Universidad Central “Marta Abreu” de Las Villas

Facultad de Ingeniería Eléctrica

CENTRO DE ESTUDIOS DE ELECTRÓNICA Y TECNOLOGÍAS DE LA INFORMACIÓN

Clasificación Automática de Neuronas Reconstruidas a partir de sus Rasgos Morfológicos

Tesis presentada en opción al Título Académico de Máster en Señales y Sistemas

Maestría en Señales y Sistemas

Autor: Ing. Duniel Delgado Castillo

Tutores: DrC. Rubén Orozco Morales

MSc. Leonardo Hernández Pérez

Santa Clara, Cuba, 2016

Universidad Central “Marta Abreu” de Las Villas

Facultad de Ingeniería Eléctrica

CENTRO DE ESTUDIOS DE ELECTRÓNICA Y TECNOLOGÍAS DE LA INFORMACIÓN

Clasificación Automática de Neuronas Reconstruidas a partir de sus Rasgos Morfológicos

Tesis presentada en opción al Título Académico de Máster en Señales y Sistemas

Maestría en Señales y Sistemas

Autor: Ing. Duniel Delgado Castillo

E-mail: duniel.delgado@etecsa.cu

Tutores: DrC. Rubén Orozco Morales

E-mail: rorozco@uclv.edu.cu

MSc. Leonardo Hernández Pérez

E-mail: leonardo.hernandez@etecsa.cu

Santa Clara, Cuba, 2016

DEDICATORIA

A mi Señor Jesucristo, Rey de mi vida y de mi hogar.

A mi amada esposa, mi ayuda idónea, por su amor, cariño y comprensión.

A mi pequeña princesa Salomé y a mi hijo por nacer, Benjamín.

A mi mamá, a mi papá, a mi hermano y su familia.

A Dana y a José que han sido como padres para mí.

A Darío, que ha sido como un hermano pequeño.

AGRADECIMIENTOS

A mi Dios porque por su gracia he llegado hasta aquí.

A mi esposa por lo que se tuvo que sacrificar para yo disponer del preciado y necesario tiempo, además de su ayuda en la redacción de la tesis.

A Leonardo Hernández Pérez y a Rainer Martín Pérez por la gran ayuda que me brindaron en el desarrollo de la investigación.

Al profesor y tutor Rubén Orozco Morales por sus valiosas sugerencias y ayuda en el desarrollo de la tesis.

A los profesores Juan Valentín Lorenzo Ginori y José Daniel López Cabrera por sus útiles consejos y ayuda en la investigación.

A todos mis compañeros de trabajo que de una manera u otra me ayudaron a tener tiempo y recursos para realizar la tesis.

A todos los que en mayor o en menor medida me ayudaron.

RESUMEN

La caracterización morfológica precisa de las múltiples clases de neuronas podría facilitar el esclarecimiento de las funciones cerebrales y los cambios funcionales que subyacen en diferentes enfermedades neurológicas. El análisis morfológico manual es muy lento y muchas veces adolece de exactitud debido a que algunas características de las células no se cuantifican fácilmente. Varias son las herramientas informáticas que se han creado para automatizar la cuantificación y comparación de los rasgos de las estructuras neuronales mediante el uso de potentes técnicas de aprendizaje automatizado.

El presente trabajo se enfoca en comparar los resultados en la clasificación automática de neuronas usando los rasgos morfológicos obtenidos con el software comercial Neurolucida y usando los rasgos extraídos con herramientas libres como el L_measure, Farsight y el TreesToolbox. Para ello se trabajó con un conjunto de reconstrucciones digitales de neuronas las cuales están divididas en dos clases, las neuronas piramidales y las interneuronas. Para evaluar los conjuntos de rasgos se utilizaron cinco algoritmos de clasificación diferentes combinados con varias técnicas de selección de atributos. También se hicieron pruebas para determinar los mejores clasificadores y para comparar los resultados dividiendo la neurona en sus dos secciones principales: axones y dendritas. Los resultados obtenidos fueron satisfactorios ya que con la incorporación de los rasgos extraídos con las herramientas libres se logró mejorar la eficacia en la clasificación, además también se demostró la superioridad de los rasgos extraídos de las dendritas sobre los obtenidos de los axones.

ÍNDICE

INTRODUCCIÓN

La comprensión de cómo funciona el cerebro es, sin duda, uno de los mayores desafíos para la ciencia moderna. La adquisición de un conocimiento profundo de la estructura, función y desarrollo del sistema nervioso a nivel molecular, celular y de sistemas es de importancia crucial en este esfuerzo, ya que los procesos en estos niveles están estrechamente vinculados con las funciones cognitivas de orden superior y son los principales objetivos de los fármacos y terapias en el tratamiento de trastornos neurológicos y psiquiátricos. Con el desarrollo de las nuevas tecnologías se han logrado muchos avances en esta materia pero todavía existen carencias respecto a un entendimiento unificado del cerebro que pueda abarcar sus múltiples niveles de organización, desde genes, cognición o comportamiento. Debido a la importancia de esta rama de la ciencia moderna, los Estados Unidos y la Unión Europea han lanzados sendos proyectos con inversiones millonarias: la "Iniciativa del Cerebro Americano" [1], iniciada en abril de 2013, disponible en www.braininitiative.nih.gov y el "Proyecto Europeo del Cerebro Humano" [2], iniciado en octubre del mismo año, disponible en www.humanbrainproject.eu. Ambos tienen como objetivo elaborar un mapa detallado y dinámico de todo el cerebro humano que sirva para conocer cómo es y cómo funciona cada detalle de nuestro cerebro, y de este modo encontrar mejores soluciones para tratar las enfermedades y lesiones del sistema nervioso. Nuestro país también está dedicando recursos a la investigación en esta área de la ciencia, un ejemplo de esto lo constituye el proyecto nacional del cual es parte este trabajo. "Herramientas de computación para estudios de morfologías neuronales en imaginología celular y análisis de imágenes de fondo de ojo o similares", cuyo objetivo general es: Desarrollar plataformas computacionales para el análisis cuantitativo de imágenes y el reconocimiento de patrones, orientada a dar respuesta a los problemas concretos de las investigaciones básicas que se desarrollan dentro del programa de neurotecnologías.

Un componente fundamental para comprender como funciona el cerebro es el estudio de la estructura y función de sus células más numerosas e importantes: las neuronas. La morfología de ellas está directamente vinculada a sus funciones, lo cual ha sido reconocido por más de un

siglo. Las neuronas se comunican a través prolongaciones en forma de árbol denominadas axones y dendritas. Estas regiones especializadas son de gran importancia en el correcto funcionamiento de las redes neuronales y su deterioro está relacionado con diversos trastornos neurológicos como son el Alzheimer y la Esquizofrenia. Estas estructuras con forma de ramificaciones se destacan por su amplia diversidad morfológica entre y dentro de las regiones cerebrales, relacionadas con el tipo específico de célula y su función [3].

Dado que la investigación en las diversas áreas de la neurociencia se basa cada vez más en las imágenes, dando lugar a grandes cantidades de datos heterogéneos adquiridos de múltiples escalas de observación, la necesidad de una informática avanzada en bioimágenes y neuroinformática, para la integración y el análisis de estos datos está aumentando rápidamente. De particular importancia en este contexto es el desarrollo de métodos computacionales y herramientas para el estudio de la anatomía neuronal. Un análisis cuantitativo de la morfología permite el estudio de los factores intrínsecos y extrínsecos que influyen en el desarrollo de la neurona, las relaciones entre la estructura y la función neuronal, los cambios neuropatológicos relacionados con síndromes específicos y los efectos de algunos compuestos, proporcionando información de gran valor para el desarrollo de fármacos [4].

En la tendencia existente en neuroinformática no solo se publican nuevas ideas, sino también las aplicaciones desarrolladas para poner a prueba estas ideas. Docenas de herramientas se pueden encontrar en la literatura reciente para realizar diversas tareas en neuroinformática como: segmentación, visualización, edición, la cuantificación, la generación, la simulación y la conversión de formatos. Las mismas permiten a los especialistas obtener reconstrucciones digitales más precisas, además de poder analizar y cuantificar estas reconstrucciones. Esto no solo trae consigo un aumento de las bases de datos de neuronas digitales sino también de los rasgos descriptivos de las mismas. Debido a este continuo crecimiento y a la complejidad de estas estructuras, realizar análisis y comparaciones de manera manual, se hace cada vez más difícil para los neurocientíficos. Por esta razón los algoritmos de aprendizaje automático son cada vez más usados en la neuroinformática. La creciente integración de las técnicas de aprendizaje automático con métodos microscópicos, químicos y funcionales ha llevado la bioinformática a nuevos niveles. Mientras que la neurociencia está en medio de una rápida transición hacia los datos digitales, los principios detrás de los algoritmos de clasificación automática se mantienen a menudo inaccesibles para los neurocientíficos, lo que limita las

posibilidades de avances. Las neuronas típicamente se caracterizan por la morfología, la fisiología y la bioquímica. Estos principales enfoques experimentales reflejan las más prominentes técnicas disponibles conocidas como: imágenes microscópicas, registro eléctrico y el análisis molecular. Estos tres dominios están íntimamente entrelazados y como un todo formar parte de la identidad neuronal, por esta razón, los métodos computacionales para clasificar automáticamente los diferentes grupos neuronales comúnmente combinan rasgos pertenecientes a los tres dominios. Son varios los estudios recientes que han aprovechado las metodologías de cálculo modernas en el estudio de los grupos neuronales. Entre los algoritmos más usados están los beyesianos, el *k*-vecinos más cercano, la regresión logística, árbol de decisión, perceptrón multicapa, máquinas de soporte vectorial, agrupamiento jerárquico, agrupamiento difuso y varios modelos estadísticos. Dentro de las plataformas más usadas se encuentra Matlab, Statistica, Weka, R y SPSS. Los conjuntos de neuronas varían desde decenas hasta miles, predominando el análisis con rasgos descriptivos pertenecientes a un solo dominio aunque en varias ocasiones se utilizan la combinación de dos y hasta tres dominios [5]. Aunque la literatura publicada ofrece varios casos de éxito en la clasificación neuronal, son aún insuficientes ante el creciente aumento de las bases de datos de reconstrucciones digitales, las cuales son diversas y complejas. Por esta razón para satisfacer las demandas actuales y futuras de la neuroinformática, se necesita perfeccionar los rasgos existentes o encontrar otros que puedan caracterizar a las neuronas de una manera más precisa, además de un desarrollo continuo de los métodos computacionales utilizados en el estudio, análisis y comparación de estas estructuras.

Con este trabajo se persigue mejorar los resultados de la clasificación entre los dos tipos de neuronas más comunes en el cerebro (piramidales e interneuronas), a partir de realizar una contribución en el análisis morfológico de sus estructuras. Además se hará una evaluación de los algoritmos implementados para la selección de rasgos y la clasificación, lo cual puede ser de utilidad para quienes necesiten realizar comparaciones similares. También se pretende vincular las plataformas de Weka y Matlab, lo cual permitirá aprovechar simultáneamente las potencialidades de ambos, permitiendo el desarrollo de aplicaciones que automáticamente puedan seleccionar rasgos, clasificar y evaluar los resultados, lo que puede ser de gran ayuda para los neuroinformáticos. Para ello se propusieron los siguientes objetivos:

Objetivo general

Determinar el método más adecuado para la clasificación automática de neuronas reconstruidas a partir de los rasgos morfológicos obtenidos de las mismas.

Objetivos específicos

- Caracterizar los métodos existentes para la clasificación automática de neuronas reconstruidas a partir de rasgos morfológicos obtenidos de las mismas.
- Diseñar el procedimiento para la selección de rasgos y su evaluación con vistas a la clasificación automática de neuronas reconstruidas a partir de sus rasgos morfológicos
- Determinar los algoritmos con mejores resultados en la clasificación de las neuronas a partir de sus rasgos morfológicos.
- Elegir cuáles métodos de selección de atributos son más eficaces para reducir la dimensionalidad de los rasgos utilizados.
- Evaluar los rasgos morfológicos más representativos en la clasificación de los grupos de neuronas analizados.

Tareas de investigación

- Realizar una búsqueda y revisión crítica de los materiales encontrados en la literatura científica sobre el tema.
- Extraer rasgos representativos de las neuronas usando herramientas libres disponibles en la web.
- Programar los algoritmos de clasificación de weka en Matlab haciendo uso de Matlab2weka para evaluar los resultados obtenidos con los conjuntos de rasgos extraídos.
- Realizar un análisis estadístico para la comparación de los algoritmos de clasificación implementados y los diferentes conjuntos de rasgos usados.

Organización del informe

Este trabajo está estructurado en introducción, tres capítulos, conclusiones, recomendaciones, bibliografía, y anexos. En la Introducción queda definida la importancia, actualidad y necesidad del tema que se aborda y se dejan explícitos los objetivos y tareas a realizar. En el

primer capítulo se recogen los principales aspectos teóricos sobre la morfología neuronal y su trazado. Además se recogen los principales datos descriptivos usados en la clasificación neuronal, así como los avances en la clasificación automática. En el segundo capítulo se analiza y describen los diferentes materiales, métodos y algoritmos utilizados en este trabajo para la clasificación neuronal a partir de sus rasgos morfológicos. En el tercer capítulo se muestran los resultados obtenidos, así como el análisis y discusión de estos. En las conclusiones se destaca el cumplimiento de los objetivos previstos en la presente investigación. Las recomendaciones propondrán indicaciones para trabajos futuros sobre el tema. La bibliografía incluye referencias utilizadas en el trabajo. Los anexos reúnen información sobre los programas y estudios realizados que no quedaron plasmados en el cuerpo del trabajo.

CAPÍTULO 1. INTRODUCCIÓN A LA CLASIFICACIÓN DE RECONSTRUCCIONES DIGITALES DE NEURONAS

En este capítulo se hará una descripción de los fundamentos del tema de investigación basado en la revisión crítica de la literatura sobre el tema. Primeramente serán descritas las neuronas, su funcionamiento y clasificación. Luego se describirán los métodos para convertir las neuronas reales en reconstrucciones digitales de neuronas. Posteriormente se mencionarán las herramientas y bases de datos disponibles para el trabajo con neuronas reconstruidas. Los siguientes epígrafes se dedicarán a la clasificación automática de neuronas, especificando los algoritmos y los datos descriptivos usados en las clasificaciones, además de hacer una revisión de los avances obtenidos hasta la fecha en este tema.

1.1 Las neuronas

El sistema nervioso regula todos los aspectos de las funciones del cuerpo y es de sorprendente complejidad. El cerebro humano adulto es el centro de control que almacena, computa, integra y transmite la información; este contiene aproximadamente 10^{11} células nerviosas, llamadas neuronas. Estas neuronas están interconectadas por unas 10^{14} sinapsis, los puntos de intersección donde dos o más neuronas se comunican. Millones de neuronas especializadas detectan características tanto de los entornos externo e interno de los organismos y transmiten esta información al cerebro para su procesamiento y almacenamiento. Millones de otras neuronas regulan la contracción de los músculos y la secreción de hormonas. El sistema nervioso también contiene células gliales, que ocupan los espacios entre las neuronas y modulan sus funciones.

A pesar de los múltiples tipos y formas de neuronas que se encuentran en los organismos metazoos, todas las células nerviosas comparten muchas propiedades comunes. La estructura y función de las células nerviosas se entiende en gran detalle, quizás, más que para cualquier otro tipo de célula. La función de una neurona es comunicar información, lo cual hace por dos métodos. El primero de ellos es mediante señales eléctricas que procesan y conducen información dentro de las neuronas, que son generalmente células altamente alargadas. Los

impulsos eléctricos que viajan a lo largo de las neuronas se denominan, potenciales de acción y la información se codifica como la frecuencia a la cual se disparan dichos potenciales. En contraste con las señales eléctricas que conducen la información dentro de una neurona, el segundo método lo forman las señales químicas que transmiten información entre las células, utilizando procesos similares a los utilizados por otros tipos de células de señalización. En conjunto, la señalización eléctrica y química del sistema nervioso le permiten detectar los estímulos externos, integrar y procesar la información recibida, transmitirlo a centros superiores del cerebro y generar una respuesta adecuada al estímulo [6].

1.1.1 Morfología de las neuronas

Aunque la morfología de diversos tipos de neuronas difiere en algunos aspectos, todas ellas contienen tres regiones principales con diferentes funciones (Figura 1.1): La primera es el cuerpo celular o soma, el cual contiene al núcleo de la neurona. La segunda son las dendritas, las cuales tienen protuberancias llamadas espinas dendríticas. Por último el axón, prolongación larga y delgada que comienza con un cono y acaba en ramificaciones llamadas terminales presinápticos.

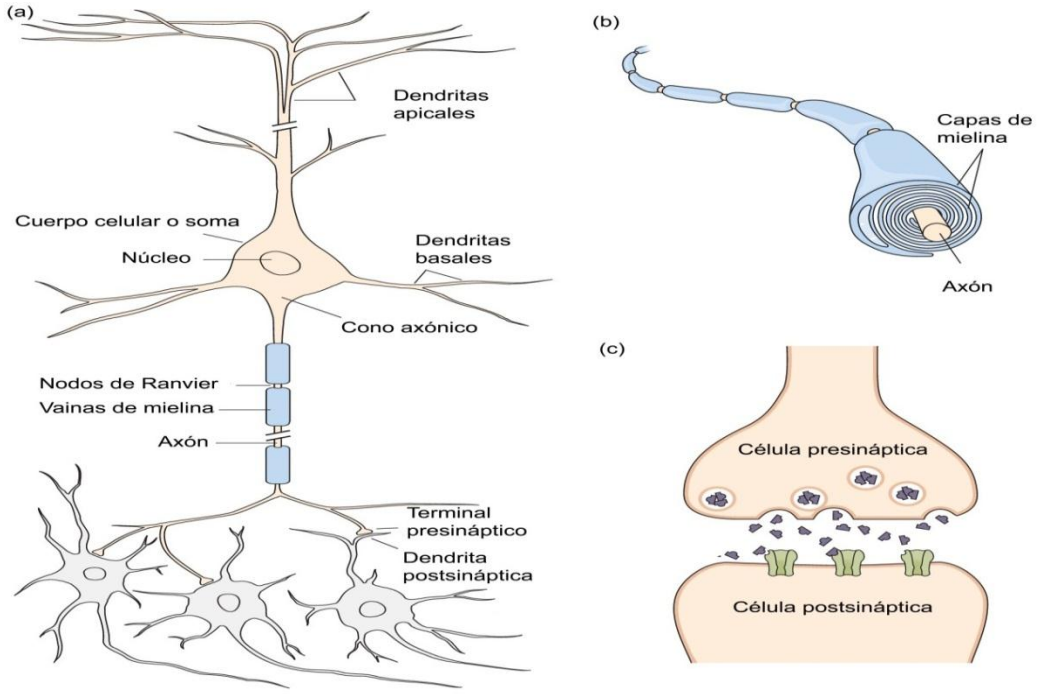

Figura 1.1: (a) Estructura de la neurona. La mayoría de las neuronas en el sistema nervioso de los vertebrados tienen las mismas características principales. (b) Axón cubierto por vainas de mielina. (c) Unión sináptica entre un terminal presináptico y una dendrita postsináptica [7].

Región I. Cuerpo celular o soma

El cuerpo celular o soma es el centro metabólico donde se fabrican las moléculas y realizan las actividades fundamentales para mantener la vida y las funciones de la célula nerviosa. El interior de la célula está constituido por una sustancia gelatinosa, el citoplasma, donde se localizan los mismos orgánulos que en otras células: el aparato de Golgi, los lisosomas, una gran cantidad de mitocondrias, retículo endoplasmático y diferentes estructuras fibrilares.

El cuerpo celular es la región que contiene el núcleo donde, al igual que en otras células, se localizan los cromosomas y el nucléolo, que fabrica los ribosomas implicados en la síntesis de proteínas. Aunque las proteínas son elementos esenciales para las funciones de cualquier célula, las neuronas requieren además proteínas específicas para desarrollar su función especializada, la transmisión de información. Para sintetizar esta gran cantidad y diversidad de proteínas, el soma cuenta con un elevadísimo número de ribosomas y un complejo sistema de membranas formado por la continuación de la membrana nuclear con los tubos del retículo endoplasmático [8].

Región II. Dendritas

Las dendritas son prolongaciones del soma con forma de árbol y constituyen las principales áreas receptoras de la información que llega a la neurona. La zona de transferencia de información de una neurona a otra es la sinapsis, la misma tiene dos componentes: el presináptico y el postsináptico, que señalan la dirección habitual del flujo de la información, que se produce desde la zona presináptica hasta la zona postsináptica. La membrana de las dendritas va a constituir generalmente el componente postsináptico. Esta membrana dendrítica (membrana postsináptica), cuenta con un elevado número de receptores, que son las moléculas especializadas sobre las que actúan los neurotransmisores liberados desde otras neuronas.

La mayoría de las neuronas tienen varios troncos dendríticos (dendritas primarias) que se ramifican varias veces, mediante bifurcación, multiplicándose de esta manera el número de ramas dendríticas y, en consecuencia, el área que ocupa cada neurona .La principal función de esta ramificación dendrítica es incrementar la superficie de recepción de información, ya que en toda la extensión del árbol dendrítico una neurona puede establecer miles de sinapsis al mismo tiempo. Las dendritas captan los mensajes y los conducen al cuerpo neuronal. Algunas

sinapsis se producen sobre pequeñas protuberancias de las dendritas denominadas espinas dendríticas.

La extensión del árbol dendrítico y la cantidad de espinas dendríticas que poseen las neuronas han atraído la atención de muchos investigadores y no es extraño si consideramos que la cantidad y diversidad de los contactos que establece una neurona van a depender del tamaño y disposición de sus dendritas. De esta manera, neuronas con escasas dendritas, cortas y poco ramificadas, tendrán menos sinapsis y esas ocurrirán en una zona más reducida que las de aquellas neuronas con una arborización dendrítica extensa, lo que le permite recibir información desde un gran número de neuronas. Además, tanto la disposición y amplitud del árbol dendrítico, como el número de espinas, parecen ser susceptibles a ser modificados por una diversidad de factores ambientales, constituyendo un ejemplo manifiesto de plasticidad neuronal [8].

Región III. Axón

EI axón es una prolongación del soma generalmente más delgada y larga que las dendritas. Es la vía a través de la cual la información se propaga hacia otras células. Esta porción de la neurona también se denomina fibra nerviosa y su longitud varía entre algunos micrómetros y varios metros, como es el caso de los axones de las neuronas motoras de las ballenas.

En el axón se pueden distinguir diferentes zonas: un segmento inicial próximo al soma denominado cono axónico, el cual desarrolla una función integradora de la información que recibe la neurona, le sigue una sección larga y delgada la cual finaliza en el terminal del axón o terminal presináptico. Del axón, lo mismo que ocurría en las dendritas, pueden surgir algunas ramificaciones colaterales, pero a diferencia de éstas, la ramificación primaria se produce ya en la zona distal, ramificándose después profusamente en su terminación para, de esta manera, transmitir la información a un mayor número de neuronas.

En los extremos de las ramificaciones axónicas se encuentran los botones terminales, denominados así por el hecho de tener forma de disco hinchado (Figura 1.1 C). Como ya se ha dicho, las neuronas se comunican unas con otras mediante sinapsis, que es el lugar de transmisión de información de una neurona a otra e implica, en consecuencia, una célula presináptica y otra postsináptica (Figura 1.1 A). Los botones terminales conforman el elemento presináptico de la sinapsis, pues a través de ellos el axón establece contacto con las

dendritas o el soma de otra neurona (o con otro tipo de células) para transmitir información [8].

Otro componente del axón lo constituyen las vainas de mielina formadas por una densa capa de membranas que se enrollan alrededor de los axones. Estas vainas, formadas en su mayor parte por lípidos, constituyen un buen aislante que mejora considerablemente la transmisión de los impulsos nerviosos. Las vainas de mielina no constituyen una cubierta continua del axón sino que están en forma segmentos regularmente espaciados por pequeñas brechas sin mielina, llamadas nódulos de Ranvier, donde la membrana plasmática del axón está expuesta al espacio extracelular durante aproximadamente 1 μm. En estos espacios sin aislamiento los potenciales de acción son regenerados [7].

1.1.2 Clasificaciones de neuronas

Es poco probable que alguna vez podamos llegar a entender cómo cada una de las cien mil millones de neuronas en el sistema nervioso contribuye de forma única a la función del cerebro. ¿Pero qué si nosotros pudiéramos demostrar que todas las neuronas en el cerebro pueden ser divididas en un pequeño número de categorías y que dentro de cada categoría, todas las neuronas funcionan de modo idéntico? La complejidad del problema podría entonces reducirse a la comprensión de la contribución única de cada categoría, en lugar de cada célula. Es con esta esperanza que los neurocientíficos han diseñado esquemas para clasificar las neuronas [9].

Basados en el número de neuritas

Las neuronas pueden clasificarse según el número total de neuritas (axones y dendritas) que se extienden desde el soma. Una neurona que tiene una sola neurita se dice que es unipolar. Si hay dos neuritas, la célula es bipolar y si hay tres o más, la célula es multipolar (Figura 1.2). La mayoría de las neuronas en el cerebro son multipolares[9].

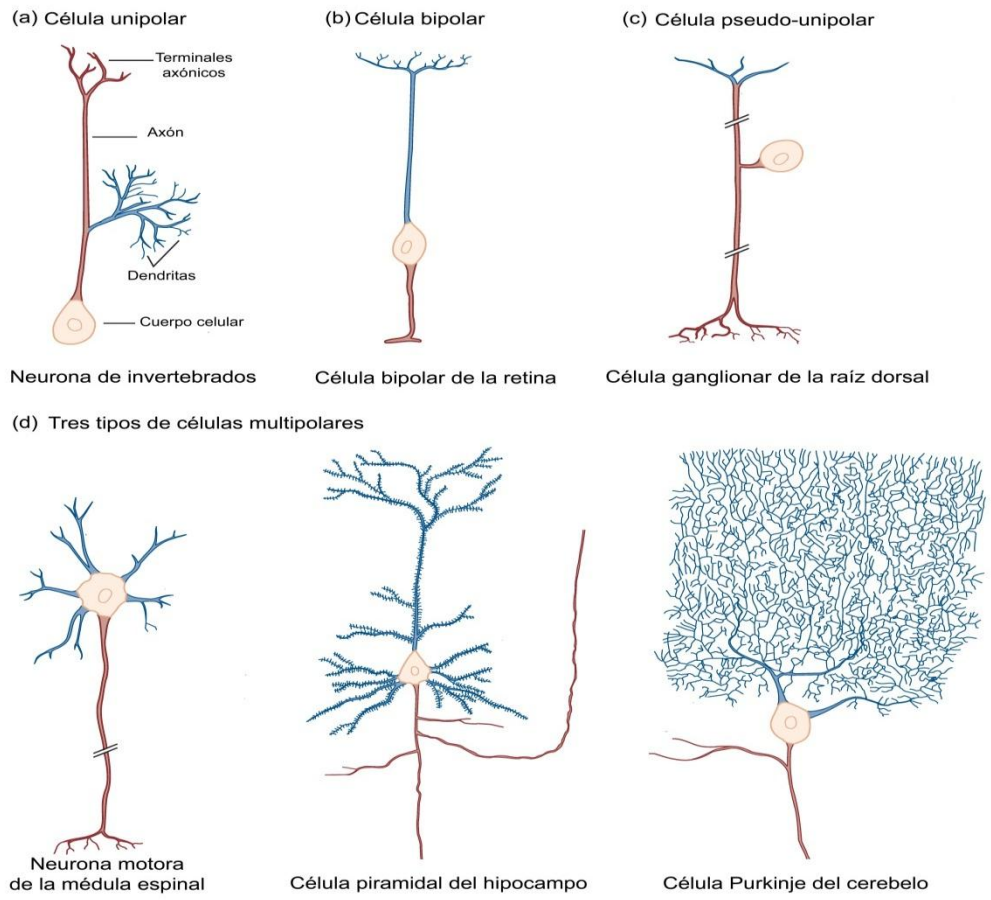

Figura 1.2: Clasificación de las neuronas de acuerdo a el número de neuritas que se extienden desde el soma [7].

Basado en las dendritas

Los árboles de dendritas pueden variar mucho de un tipo de neurona a otra. Algunos han inspirado nombres elegantes como "células de ramillete doble". Otros tienen nombres menos interesante, tales como "células alfa". La clasificación es a menudo única para una determinada parte del cerebro. Por ejemplo, en la corteza cerebral (la estructura que se encuentra justo debajo de la superficie del cerebro), hay dos grandes clases: las células estrelladas (forma de estrella) y las células piramidales (forma de pirámide).

Otra manera sencilla de clasificar las neuronas es en función de si sus dendritas tienen espinas o no. Aquellas que sí las presentan son llamadas espinosas y las que no, se denominas células aspinosas. Estos esquemas de clasificación dendríticas pueden solaparse. Por ejemplo, en la corteza cerebral, todas las células piramidales son espinosas. Las células estrelladas, por otro lado, pueden ser espinosa o aspinosas [9].

Basado en las conexiones

La información se distribuye en el sistema nervioso por las neuronas que tienen neuritas en las superficies sensoriales del cuerpo, tales como la piel y la retina del ojo. Las células con estas conexiones se denominan neuronas sensoriales primarias. Otras neuronas tienen axones que forman sinapsis con los músculos y ordenan movimientos; éstas se llaman neuronas motoras. Pero la mayoría de las neuronas del sistema nervioso forma conexiones únicamente con otras neuronas. De acuerdo con este esquema de clasificación, estas células son todas llamadas interneuronas [9].

Basado en el largo del axón

Algunas neuronas tienen axones largos que se extienden desde una parte del cerebro a la otra, estas son llamadas neuronas Golgi de tipo I o neuronas de proyección. Otras neuronas tienen axones cortos que no se extienden más allá de la proximidad del cuerpo celular, éstas se llaman neuronas Golgi de tipo II o neuronas de circuito local. En la corteza cerebral, por ejemplo, las células piramidales por lo general tienen axones largos que se extienden a otras partes del cerebro y por lo tanto son neuronas Golgi de tipo I. En contraste, las células estrelladas tienen axones que nunca se extienden más allá de la corteza cerebral y por lo tanto son neuronas Golgi de tipo II [9].

Basado en el neurotransmisor

Los sistemas de clasificación presentados hasta el momento se basan en la morfología de las neuronas como lo revelado por la tintura de Golgi. Los nuevos métodos que permiten a los neurocientíficos identificar qué neuronas contienen neurotransmisores específicos han resultado en un esquema para clasificar las neuronas basándose en su química. Por ejemplo, las neuronas motoras que ordenan los movimientos voluntarios, todas liberan el neurotransmisor acetilcolina en sus sinapsis. Estas células son por lo tanto también clasificadas como colinérgica, lo que significa que utilizan este neurotransmisor particular. Las colecciones de las células que utilizan un neurotransmisor común conforman los sistemas de neurotransmisores del cerebro [9].

1.2 Reconstrucción digital de neuronas

La morfología neuronal es una clave determinante del procesamiento de la información en el sistema nervioso. La diversidad morfológica de axones y dendritas proporciona un sustrato esencial para la integración sináptica, la transmisión de señales, la conectividad de red y la dinámica del circuito. El comienzo de la neurociencia moderna se asocia a menudo con los primeros dibujos de árboles de neuritas y aquellos primeros ejemplos siguen apareciendo en artículos de investigación y libros de texto. Sin embargo, métodos computacionales son necesarios para cuantificar la compleja relación entre la morfología neuronal (estructura) y la fisiología (actividad). Por lo tanto, reconstrucciones digitales tridimensionales de ramificaciones axonales y dendríticas son indispensables para la exploración de la función neuronal [10].

La microscopia óptica permite un equilibrio óptimo entre la resolución y el campo de visión para rutinariamente visualizar individualmente cada rama a través de todos los arboles neuronales. Los pasos de pre-procesamiento para las reconstrucciones digitales son los mismos que para los trazados de lápiz sobre papel: preparación histológica del tejido, tinción y etiquetado [4]. Las reconstrucciones digitales se obtienen mediante el trazado de árboles neuronales en 3D con interfaces dedicadas de microscopio-computadora o con software especializado a partir de secuencias de imágenes previamente adquiridas. El formato digital más popular representa la morfología de las ramas como una secuencia de vectores interconectados, registrando las coordenadas X, Y, Z, diámetros y enlaces de conectividad. Las reconstrucciones digitales capturan la esencia de la morfología neuronal a escala de microscopía de luz en archivos compactados, normalmente al 0,01% del tamaño de las imágenes originales. Lo que es más importante, permiten una variedad casi ilimitada de medidas morfométricas, modelos computacionales y el análisis estadístico para investigar los cambios estructurales y sus efectos durante el funcionamiento normal del cerebro, desarrollo y patologías. Las reconstrucciones digitales pueden ser adquiridas de cualquier especie animal, regiones del cerebro, tipos de neuronas y una variedad de protocolos experimentales [10].

1.2.1 Métodos para convertir la neuronal real en una reconstrucción digital

La reconstrucción tridimensional de la morfología neuronal ha sido una técnica de laboratorio establecida y extendida durante tres décadas, pero los avances recientes en neurobiología,

microscopía y la tecnología de la información han ampliado tanto la amplitud y la profundidad de estos estudios. Ahora podemos etiquetar selectivamente diferentes tipos de neuronas, lo que confirma su impresionante diversidad fenotípica y permitiendo la identificación de sus propiedades distintivas. Los avances en microscopía de luz están aumentando la resolución, el contraste, la velocidad y la aplicabilidad de las imágenes neuronales, revelando detalles morfológicos más refinados que antes eran inaccesibles [3].

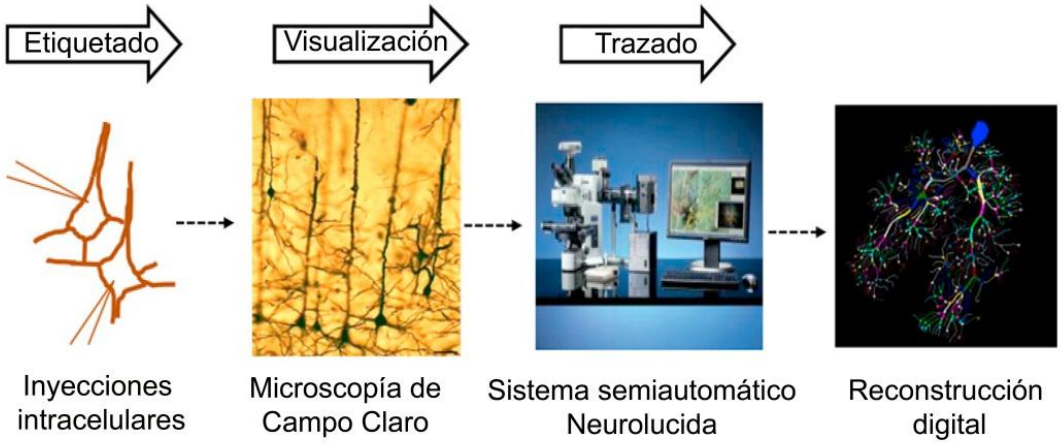

Figura 1.3: Proceso de conversión de una neurona real en una reconstrucción digital [3].

Para realizar la reconstrucción digital de la neurona existen dos métodos fundamentales, los cuales tienen aspectos en común. El primero de los métodos cuenta con tres pasos fundamentales: etiquetado de células, visualización y trazado (Figura 1.3). Para el etiquetado celular varias técnicas de tinción se han desarrollado a través de los años, su objetivo primario es lograr un claro contraste entre el árbol neuronal y el tejido de fondo. Un ejemplo de esto lo constituyen las inyecciones intracelulares de colorantes. Como segundo paso tenemos visualización o adquisición de la imagen mediante microscopía óptica, cuyas técnicas más relevantes son la microscopía de campo claro, la confocal, la de excitación dos fotones y la de superresolución. Una vez que se ha visualizado la neurona se prosigue al trazado. Las técnicas de trazado han evolucionado con los años a partir de la cámara lucida básica hasta algoritmos automatizados que generan reconstrucciones digitales de morfologías neuronales. Uno de los más potentes y populares es el Neurolucida, un sistema semiautomático que incluye el microscopio óptico controlado por un programa especializado en el ordenador, el cual crea representaciones compartiméntales en formato vectorial de los árboles dendríticos y axonales, es decir el archivo de la neurona reconstruida. Si bien los métodos de reconstrucción más

modernos son facilitados por los algoritmos de cálculo cada vez más automatizados, la intervención humana todavía se requiere en todos los casos, al menos, para asegurar la comprobación de errores y control de calidad [3]. El segundo método fundamental para el trazado de la neurona tiene los dos primeros pasos igual que el descrito anteriormente. El tercer paso, que es el trazado, lo hace de forma diferente. Después de obtenidas las imágenes, las cuales garantizarán un registro permanente de las muestras originales, se procede a la crítica tarea de segmentación (el proceso de asignar a cada elemento de la imagen una etiqueta que indique a qué segmento o parte pertenece). Esto normalmente implica cuatro pasos, comenzando con el pre-procesamiento de la imagen, seguido de la segmentación del cuerpo celular (soma), el árbol neuronal (axón y dendritas) y finalmente las espinas, todo esto haciendo uso de las técnicas de procesamiento digital de imágenes. Como último paso, ya después de estar segmentada la imagen, se procede a la conformación de la reconstrucción digital de la neurona [4].

1.3 Herramientas de software disponibles

Existe una creciente tendencia en la Neuroinformática [11], así como en otros campos de la ciencia y la ingeniería, donde no solo se publican nuevas ideas sino también las herramientas de software desarrolladas para probar esas ideas. En la más reciente literatura podemos encontrar docenas de estas herramientas de gran utilidad para los neuroinformáticos. A continuación se presentan las herramientas más conocidas, haciendo énfasis en las que están disponibles gratuitamente, se clasifican según su funcionalidad: segmentación, visualización, edición, cuantificación, generación, simulación y para la conversión de formato. Algunas de las herramientas pueden realizar varias funciones.

1.3.1 Herramientas para segmentación

Una de las primeras herramientas de software para el trazado neuronal en 3D fue el paquete Neurolucida (MBF Bioscience) que solo soportaba operaciones manuales [12]. Posteriormente le fue incorporado el módulo AutoNeuron que permite realizar trazado automático. NeuronTracer es otra de las herramientas para el trazado neuronal, esta es un módulo del paquete Imaris comercializado por la empresa Suiza Bitplane así como, HCA-Vision (CSIRO Biotech Imaging) de Australia [13]. Uno de los programas más empleados en las primeras

publicaciones que utilizaron reconstrucciones digitales, fue Eutectic [14], esta herramienta permite el trazado de las neuronas directamente desde una sola imagen. Además, el usuario también puede editar, combinar, filtrar y ver la morfología en 3D.

El trazado semiautomático de neuritas, a través de secciones ópticas puede ser hecho utilizando la herramienta Reconstruct [15]. Neuronj es una herramienta muy popular para la medición de la longitud de los parámetros de la neurita a modo de pruebas en 2D. Neuronj ha sido usado como una herramienta de referencia para pruebas de los algoritmos de trazado neuronal en 2D más recientes y ha inspirado a otros a desarrollar herramientas interactivas como Neuromantic [16]. Los mayores niveles de automatización para aplicaciones 2D se encuentran en herramientas como NeuronMetrics [17], NeuriteTracer [18] y NeuronCyto [19] y para aplicaciones 3D en NeuronStudio [20], NeuronIQ [21] y ORION [22].

1.3.2 Herramientas para visualización

Aunque las herramientas utilizadas en el trazado neuronal usualmente visualizan los resultados, existen varias herramientas que han sido desarrolladas específicamente con este fin. La mayor parte aceptan múltiples formatos de ficheros. Un ejemplo es Cvapp, el cual fue originalmente escrito para inspección y creación de datos de morfología neuronal, posteriormente fue extendido para realizar otras funciones, incluyendo la conversión entre distintos formatos. Otros ejemplos de herramientas para la visualización de datos de morfología neuronal de diferentes formatos son: NeuroGL y NL MorphologyViewer.

La herramienta de visualización más versátil es V3D [23], la cual soporta renderizado 5D (proyección espacial en 3D en el tiempo con múltiples colores) de los datos de la imagen así como los datos de superficie, datos relacionales que pueden ser descritos como un gráfico (tales como los datos de la neurona). Ofrece también varias funciones para el análisis de imágenes (como módulos adicionales). V3D soporta también complementos para el desarrollo con los cuales los usuarios pueden explotar su plataforma en el desarrollo de nuevas funciones. Su similar comercial es Amira. Otra herramienta es el TreesToolbox de Matlab el cual es de gran utilidad ya que permite la visualización, edición y cuantificación de árboles neuronales [24].

1.3.3 Herramientas para edición

Si bien la automatización total del procesamiento y análisis de datos queda como última finalidad y es de hecho un requisito previo para experimentos de elevado contenido y rendimiento, los algoritmos actuales logran conseguir esto solo bajo condiciones óptimas que son difíciles de conseguir en la práctica.

En la mayor parte de los casos el trazado neuronal contiene gran variedad de errores como:

- Se incluyen o suprimen estructuras de la imagen (falsos positivos o negativos).
- Rompimiento o ensamble de estructuras (falsa fragmentación o continuación).

El impacto que tendrán estos errores va a estar en dependencia de la naturaleza del estudio. Por ejemplo, una segmentación falsa negativa de parte del árbol de dendritas puede afectar significativamente el conteo total de dendritas o espinas pero puede que no tenga un efecto indeseado en los promedios, por ejemplo en la densidad de espinas. Por el contrario, una falsa fragmentación del árbol puede afectar los promedios, como la longitud de las dendritas, pero no afectará el conteo de la longitud total de dendritas. La mayor parte de las herramientas de segmentación permiten la edición manual para corregir los errores [4]. Una de las herramientas usadas para la edición es Farsight [25], el cual permite la inspección de la reconstrucción y la corrección de errores utilizando una interfaz gráfica.

1.3.4 Herramientas para cuantificación

Al igual que para la edición y visualización la mayor parte de las herramientas para el trazado neuronal permiten la cuantificación, pero existen un pequeño grupo desarrolladas para realizar específicamente esta tarea, tal es el caso de L_measure [26], la cual permite calcular alrededor de 100 parámetros morfológicos independientes (con respecto al soma y el árbol), de poblaciones de células o neuronas individuales. Esto permite análisis comparativos detallados sobre un gran número de neuronas, el descubrimiento de los rasgos morfológicos característicos en todas las clases de células, la detección de diferencias inducidas por factores de crecimiento específicos, el análisis de los cambios de desarrollo, la extracción de las distribuciones de parámetros necesarios para las simulaciones computacionales para generar neuronas virtuales y la evaluación de la calidad y las limitaciones de estos modelos mediante la comparación de sus propiedades emergentes con los datos experimentales originales.

Otra herramienta es XL-Calculations [27], la cual fue diseñada para distinguir entre neuronas en diferentes etapas de diferenciación y con esto facilitar el procesamiento por lotes de un gran número de neuronas trazadas de NeuronJ y el cálculo automatizado de más de 45 diferentes medidas cuantitativas de árboles neuronales.

1.3.5 Herramientas para generación

Un verdadero reto de la neuroinformática es el desarrollo de algoritmos descriptores de la morfología neuronal, estos sirven para hacer representaciones de neuronas más compactas que la reconstrucción digital así como permitir entender los procesos y parámetros fundamentales de la naturaleza y la generación automática de neuronas virtuales utilizadas en la modelación y simulación de experimentos. Existen varios tipos de estas herramientas que permiten la generación desde una neurona hasta una red neuronal completa, basados en modelos estadístico derivados de bases de datos de morfologías reales. Como ejemplo de estas herramientas tenemos L-Neuron [28], que ejecuta algoritmos locales independientes de las restricciones extrínsecas. Otro ejemplo para la generación de redes de neuronas es la herramienta NeuroConstruct [29].

1.3.6 Herramientas para simulación

El modelado y simulación de redes neuronales es importante para realizar pruebas de hipótesis derivadas de paradigmas computacionales. Génesis [30] fue el primer sistema desarrollado para la simulación de modelos biológicamente realistas a diferentes niveles, que van desde componentes subcelulares y reacciones bioquímicas, hasta neuronas individuales, grandes redes y modelos de sistemas completos. Análogamente otro entorno de simulación es Neuron [31]. Ambos están dotados de recursos para construir, probar y manejar modelos. En [32] se presentan además de los antes mencionados otros seis sistemas. Similar a la cuantificación de propiedades morfológicas, la simulación de comportamiento electrónico y electrofisiológico también puede verse afectada de manera significativa por las variabilidades en la reconstrucción digital de las neuronas, debido a esto la importancia de la reconstrucción precisa y consistente.

1.3.7 Herramientas para conversión

En vista de las diferentes tareas especializadas en neuroinformática, las reconstrucciones neuronales deberían idealmente guardarse en un formato abierto, extensible y bien documentado, lo que permite un fácil archivado e intercambio de información, no sólo entre los programas, sino también entre los diferentes laboratorios para facilitar la colaboración. Desafortunadamente existen herramientas comerciales con formatos propios. Uno de los formatos libres más usados es el 'SWC' [33], el mismo almacena en texto ASCII la cantidad mínima de parámetros requerida para representar una reconstrucción tridimensional basada en vectores, más información acerca de este formato se puede consultar en [34]. La herramienta más completa hasta la fecha es NLMorphologyConverter, la cual mediante simples líneas de comando es capaz de convertir entre la mayoría de formatos de morfologías 3D de neuronas biológicas, incluyendo todos los formatos de Neurolucida, SWC, MorphML, NeuronHOC, Genesis, Imaris, NeuroZoom, Eutectics y muchos otros. Dicha herramienta está disponible de manera gratuita en http://neuronland.org.

1.4 Bases de datos disponibles

La aceleración explosiva y continua en la recopilación y aplicación de reconstrucciones digitales de morfología neuronal ha creado una necesidad apremiante de organizar los esfuerzos en la custodia, anotación, almacenamiento y distribución de los datos. El meta-análisis de los datos primarios existentes provenientes de muchos estudios y el reanálisis de conjuntos de datos seleccionados pueden dar lugar a notables descubrimientos secundarios [35]. A su vez, el intercambio y reutilización de datos por la comunidad aumenta la visibilidad y el impacto de los estudios originales para los cuales fueron recogidos los datos. Por lo tanto, varios laboratorios están publicando libremente sus reconstrucciones digitales después de la publicación de los resultados, ya sea en su propio sitio web o en otros como es el caso del sitio público www.neuromorpho.org [3].

Iniciativas como *Neuroscience Information Framework* (NIF por sus siglas en inglés) diseñada e implementada para integrar el acceso y promover el uso de los recursos disponibles en la web para la neurociencia, cuenta con un inventario continuamente actualizado de herramientas, datos y materiales de utilidad para la comunidad científica [36]. A continuación

describiremos algunas de las bases de datos de reconstrucciones digitales disponibles de manera gratis, las cuales son mantenidas activamente y actualizadas regularmente.

1.4.1 Base de Datos NeuroMorpho.Org

NeuroMorpho.org es un repositorio central público de reconstrucciones digitales de morfologías neuronales cuyo objetivo es proporcionar una densa cobertura de los datos reconstruidos disponibles para la comunidad de neurocientíficos. Desde sus inicios ha estado creciendo continuamente y hasta la fecha es la mayor colección de reconstrucciones neuronales 3D con un total de 50356 (Figura 1.4), contribuidas por 227 laboratorios, de 36 especies y 40 regiones del cerebro.

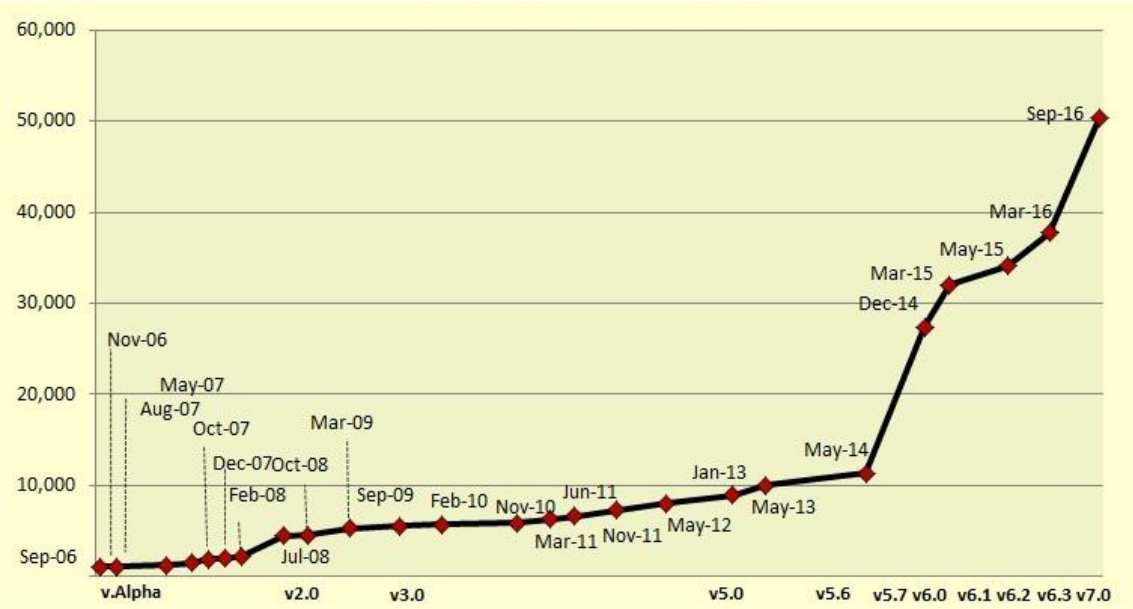

Figura 1.4: Número de reconstrucciones disponibles en cada actualización de la base de datos NeuroMorpho.org.

Las búsquedas se pueden hacer mediante menús desplegables o con una palabra clave por cualquier categoría de metadatos (por ejemplo, tipo de neurona, condiciones experimentales y el formato de archivo); así como por la morfometría (número de ramas, volumen, etc.). Todo el repositorio de neuronas está organizado por tipo de célula (por ejemplo, piramidal, motora, etc.); región del cerebro (hipocampo, cerebelo, etc.); por especies animales o por laboratorio contribuyente y puede ser examinado o descargado en el navegador a través de una pantalla intuitiva interactiva.

Según estadísticas ofrecidas en la propia base de datos, se han descargado más de 5.5 millones de archivos en más de 160.000 visitas de 153 países, además de ser citada en más de 460 publicaciones. También mantiene una amplia cobertura de literatura de las publicaciones que contienen trazados neuromorfológicos desde el inicio de la tecnología de reconstrucción digital. Las publicaciones pueden ser examinadas introduciendo el identificador de PubMed y navegado por la información de reconstrucción, año de publicación o estado de disponibilidad de los datos descritos [37] [10].

1.4.2 Base de Datos Centrada de Células

Esta base de datos disponible en www.ccdb.ucsd.edu fue creada en 2002 para proporcionar un repositorio en línea de reconstrucciones a base de microscopía óptica de alta resolución de estructuras celulares y subcelulares, permitiendo a los investigadores explorar y reutilizar estos datos. Proporciona acceso libre a reconstrucciones digitales de varios sistemas incluyendo el conjunto de imágenes de microscopía primarias. Contiene datos de alta resolución en 2D, 3D, 4D, así como archivos de imagen de localización de proteínas, mosaicos cerebrales, registros de tomografía y análisis. La base de datos también incluye ontología de anatomía subcelular enlazada al NIF [38].

1.4.3 Base de Datos de Circuito de la Mosca Drosophila

Es una base de datos pública de neuronas de la mosca Drosophila, la cual se encuentra disponible en www.flycircuit.tw. Esta contiene aproximadamente 16000 reconstrucciones de los nueve principales sistemas de neurotransmisores en el lóbulo óptico y el cerebro central de la mosca de la fruta. Etiquetando genéticamente las neuronas de proyección y locales en 29 regiones neuropilo de cada hemisferio se reconstruyeron de forma individual con un algoritmo automático personalizado después del preprocesamiento manual de las imágenes confocales. Las reconstrucciones pueden ser visualizadas en 3D en diferentes modos tales como representaciones de volumen, modo de esqueleto y por localización soma dentro del atlas. Cada reconstrucción posee metadatos los cuales incluyen el género animal y la edad, así como el neurotransmisor putativo y la distribución espacial de terminales de los axones. Esta información está codificada para permitir una rápida búsqueda por similitud. Las reconstrucciones se pueden descargar en formato Amira [39].

1.4.4 Bases de Datos de Laboratorios

Varios laboratorios mantienen disponibles públicamente las bases de datos de sus propios estudios. Tales son los casos de los Doctores:

- Alexander Borst www.neuro.mpg.de/30330/borst_modelfly_downloads,
- Brenda Claiborne http://www.utsa.edu/claibornelab/,
- Alain Destexhe www.cns.iaf.cnrs-gif.fr/alain_geometries.html,
- Attila Gulyas www.koki.hu/~gulyas/ca1cells,
- Patrick Hof http://research.mssm.edu/cnic/repository.html,
- William Kath http://dendrites.esam.northwestern.edu/
- Rafael Yuste http://www.columbia.edu/cu/biology/faculty/yuste/databases.html.

Estas bases de datos también están reflejadas en www.neuromorpho.org para el acceso centralizado a todas las reconstrucciones y los metadatos asociados.

1.5 Datos descriptivos usados en la clasificación de neuronas

La clasificación neuronal es un asunto que va en aumento cada día, pero su historia comenzó con la neurociencia en sí misma [40]. Los investigadores habitualmente se refieren a las neuronas como piramidales, estrelladas, de gránulo, bipolares o de cesta, pero estos nombres son a menudo insuficientes para describir la diversidad neuronal aún dentro de áreas limitadas del cerebro. Al darse cuenta de este problema, tanto el "Proyecto Europeo del Cerebro Humano" como la "Iniciativa del Cerebro Americano" identificaron la clasificación de tipos de células entre sus primeras prioridades [1] [2]. Para completar un censo exhaustivo de las células del cerebro humano, la última tarea es vincular tipos neuronales con el comportamiento, la computación y la cognición. Destacados esfuerzos internacionales proponen directrices iniciales para ayudar a organizar el cuerpo creciente de conocimiento [41]; sin embargo, los esfuerzos de clasificación manual están mal equipados para hacer frente a grandes volúmenes de datos. La magnitud y la complejidad de la clasificación neuronal exigen tecnologías de alto rendimiento.

La neurociencia y la informática están listas para hacer frente a la clasificación neuronal mediante métodos matemáticos potentes [5]. La creciente integración de las técnicas de aprendizaje automático con métodos de microscopía, químicos y funcionales ya han llevado a la bioinformática a un nuevo nivel [42]. Mientras que la neurociencia está pasando

rápidamente a datos digitales [3], los principios detrás de los algoritmos de clasificación automática se mantienen a menudo inaccesibles para los neurocientíficos, lo que limita las posibilidades de avances [5].

En la caracterización de las células neuronales se usan típicamente tres tipos de propiedades: las morfológicas, las bioquímicas y las electrofisiológicas (Figura 1.5). Estos principales enfoques experimentales reflejan las técnicas disponibles más prominentes, es decir, imágenes de microscopía (análisis de su estructura morfológica), la grabación eléctrica (análisis de los pulsos eléctricos que pasan a través de su estructura) y el análisis molecular (análisis de su composición bioquímica). Las estructuras axonales y dendríticas establecen los medios necesarios para la conectividad de red; los perfiles de expresión neuronales proporcionan una ventana hacia los orígenes del desarrollo y las propiedades electrofisiológicas subyacen en el procesamiento de señales. Además, estas características están íntimamente entrelazadas [5].

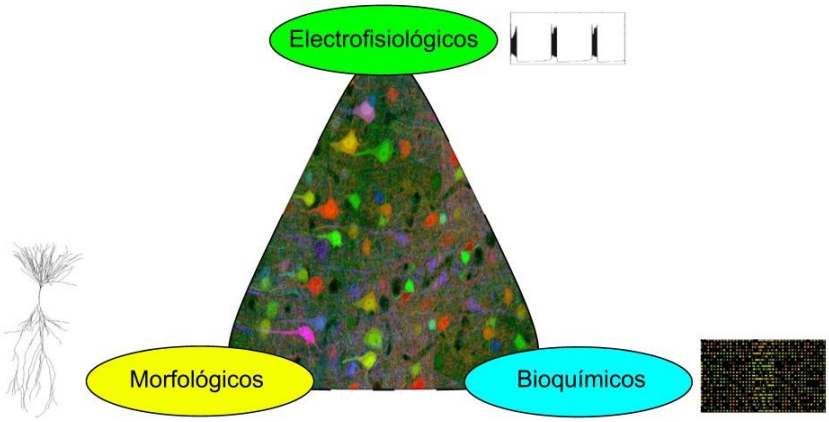

Figura 1.5: Dimensiones básicas de la caracterización neuronal. Estos tres dominios de características están estrechamente interrelacionados con otros aspectos de la identidad neuronal, como son la conectividad, el desarrollo y la plasticidad [5].

La maquinaria macromolecular esculpe tanto la excitabilidad neuronal como la circuitería y estas juntas definen funciones computacionales. La dificultad del problema aumenta aún más cuando se consideran las diferencias sistemáticas entre las especies, a través de las áreas del cerebro y durante el desarrollo. Sin embargo, incluso para los modelos animales más comunes, para las regiones bien definidas de los sistemas nerviosos y los rangos de edad confinados o etapas de desarrollo, la información disponible sobre la identidad neuronal ha fracasado hasta ahora para producir un acuerdo general sobre el procedimiento para la clasificación neuronal. Así como las listas de piezas preceden a los diagrama de despiece en los manuales de montaje,

la identificación objetiva de los tipos de neuronas es esencial para entender sus interacciones funcionales [43].

1.6 Clasificación automática de neuronas

El término "clasificación" a menudo es usado con dos significados relacionados pero diferentes cuando nos referimos a tipos neuronales. En el sentido más estricto, la clasificación neuronal es el proceso de dividir un grupo de neuronas en clases conocidas, como lo ejemplificado por la tarea de distinción entre células estimulantes e inhibitorias. El segundo uso del término engloba la propia clasificación anterior así como la identificación de las clases mismas, un paso a veces referido como categorización. Esta más amplia connotación implica la definición de distintos tipos neuronales y la asignación simultánea de neuronas a cada tipo [5]. La clasificación automática principalmente es conducida por datos y por lo tanto en gran parte transparente al investigador. Formalmente un conjunto de datos de clasificación neuronal D consiste en un conjunto de k neuronas observadas, cada una descrita por $(n + 1)$ variables. Las primeras n variables, conocidas como variables predictivas, son mediciones de la neurona. La última variable, referida como variable de clase, especifica el tipo de neurona.

Un clasificador es una función asignándole etiquetas a las observaciones:

$$\beta : (x_1,\ldots,x_n) \rightarrow 1,2,\ldots,m, \qquad (1.1)$$

donde el vector n-dimensional $x = (x_1,\ldots,x_n)$ contiene los valores de todas las mediciones de una neurona particular y $\{1,2,\ldots,m\}$ son las posibles clases neuronales. La clase real de una neurona dada, usualmente denotado como c, es un valor dentro de ese rango. La asumida (pero desconocida) distribución de probabilidad conjunta $p(x_1,\ldots,x_n,c)$ subyace en que las observaciones pueden ser estimadas de las muestras $\{(x^1,c^1),\ldots,(x^k,c^k)\}$, donde el índice superior refiere a las neuronas y el índice inferior a las mediciones de esas neuronas.

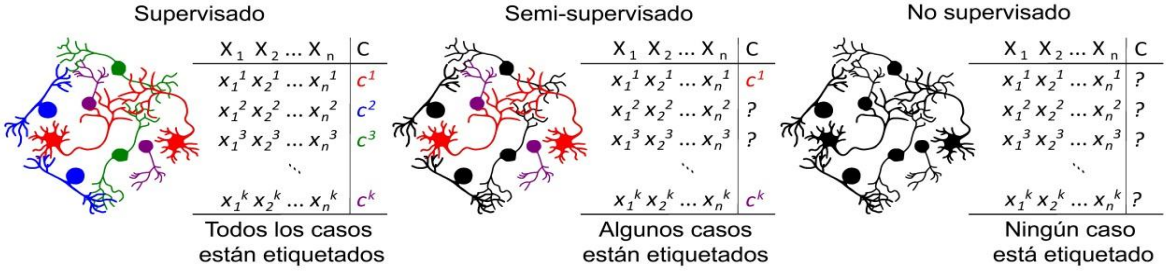

Figura 1.6: Representación de los tres esquemas de aprendizaje automático [5].

La clasificación automática puede ser conceptualizada ampliamente sobre la base del conocimiento disponible y el objetivo científico (Figura 1.6). En la clasificación supervisada, tanto las *n* mediciones y la asignación de clase *c* son conocidos para todas las neuronas *k* en el conjunto de datos. El objetivo de la clasificación supervisada es formular un mapeo predictivo o explicativo entre las mediciones y las clases. Por ejemplo, los valores de amplitud, frecuencia y duración del pulso eléctrico, de una muestra conocida de neuronas glutamatérgicas y gabaérgicas pueden ser utilizadas para asociar las características del pulso eléctrico con sus efectos postsinápticos. Este conocimiento se puede aprovechar para inferir la función de red o para deducir el neurotransmisor liberado por las neuronas en un conjunto de datos diferente a partir de sus registros eléctricos. En la clasificación supervisada el número de tipos de neuronas está predeterminado.

Por el contrario, en la clasificación o agrupamiento no supervisado sólo el conjunto de *n* mediciones está disponible, pero los valores particulares de la variable de clase son desconocidos para todas las *k* neuronas. En este caso, el objetivo es encontrar las clases que mejor explican las mediciones mediante la agrupación de las neuronas en grupos identificables. La clasificación no supervisada también determina el número de tipos de células. Por ejemplo, la variedad observada de los perfiles de expresión de proteínas en un conjunto de neuronas podrían ser reducibles a un número restringido de distintos, pero internamente consecuente, patrones de expresión, cada uno controlado por factores de transcripción específicos. En el caso intermedio de clasificación semi-supervisada, sólo algunas, pero no todas las neuronas están etiquetadas con clases conocidas. Por ejemplo, un investigador puede registrar la latencia del pulso eléctrico, la resistencia de entrada y la relación de adaptación para un conjunto de neuronas, pero sólo pudo establecer la identidad morfológica a partir de inyección de biocitina en una minoría de las células [5].

1.7 Avances en la clasificación automática de neuronas

La clasificación de neuronas en tipos se ha debatido mucho desde el inicio de la neurociencia moderna. Los recientes avances experimentales están acelerando el ritmo de recolección de datos. El aumento resultante de la información acerca de características morfológicas, fisiológicas y propiedades moleculares alienta los esfuerzos para automatizar la clasificación neuronal por potentes técnicas de aprendizaje automático [5]. Recientes estudios aplicando

estas técnicas en problemas de neurobiología han mostrado ventajas significativas. La exactitud de los datos y la sensibilidad de las técnicas de aprendizaje de máquina para resolver delicados problemas de morfometría celular sugieren su potencial para múltiples aplicaciones en futuros estudios. Esto incluiría la diferenciación entre las etapas de desarrollo específicas de neuronas y sus respuestas al medio de cultivo y la definición de cambios celulares dentro de cerebros normales y enfermos [44]. Son varios los artículos publicados con historias de éxito en la clasificación neuronal automática. Esta sección examina una selección representativa de estas publicaciones recientes, evaluando brevemente los datos biológicos, técnicas de computación y las implementaciones de software utilizadas.

1.7.1 La morfología neuronal y la conectividad de circuito

La prominencia de las características morfológicas para la identificación de tipos neuronales comenzó con el emparejamiento transformador de la tintura de Golgi y la microscopía óptica [40]. Relativo a la sensible condición de dependencia de los estados electrofisiológicos y bioquímicos, la forma de las neuronas garantiza una robustez considerable. Hasta estos días, las estructuras axonales y dendríticas siguen siendo fundamentales en el análisis del desarrollo neuronal, la patología, la computación y los circuitos [45]. Las primeras aplicaciones de reconocimiento de patrones de energía wavelet multiescala mostraron prometedoras clasificaciones de formas de tipos neuronales específicos como las células ganglionares de la retina [46].

Los primeros intentos de producir jerarquías neuronales generales basadas en las propiedades morfológicas se pueden apreciar en [47], sin embargo, su fuerza fue limitada. A pesar de los progresos recientes, el conocimiento de la diversidad morfológica y de los patrones de conectividad se encuentra todavía en su infancia. La información limitada acerca de los tipos neuronales, por no hablar de sus relaciones jerárquicas, restringe el potencial de los métodos taxonómicos. Las aplicaciones de aprendizaje automático en la clasificación de neuronas fueron iniciadas a través del análisis puramente basado en datos de interneuronas gabaérgicas corticales [49] [50] [51] [52]. Aunque estos estudios comparten el objetivo común de identificar distintos subtipos de neuronas inhibitorias de circuitos locales, se apoyaron en diferentes mediciones experimentales y algoritmos computacionales. Los resultados de clasificación automática se validaron mediante la inspección de expertos [52], el conocimiento

previo sobre las clases [51], interpretabilidad biológica [49] o el consenso de la comunidad [50].

Algunas características morfológicas fueron sorprendentemente eficaces para la clasificación automática como es el caso de análisis clásico de Sholl [53] con intervalos de 300 µm en axones [49] [50] [52] y los intervalos de 50-100 µm para dendritas [49] [51]. Otras características morfológicas relevantes fueron el área de la envolvente convexa de las dendritas [54], volumen, área de superficie [51] y la relación entre la longitud dendrítica al área de superficie [49]. Un reciente estudio incluyó varias líneas transgénicas en un análisis de 363 células ganglionares de la retina de ratón [55]. Estructuras de árboles de dendritas en 3D de coordenadas *voxel* fueron diferenciadas por agrupación jerárquica con la distancia euclidiana. El corte de agrupamiento fue elegido como el nivel más bajo que correctamente agrupó los fuertemente definidos tipos genéticos, evaluando la fiabilidad con el método de validación cruzada dejando uno fuera. Se identificaron quince clases, seis de las cuales esperan una identificación genética más exacta. La incorporación de más de 100 parámetros morfométricos del software L_measure [26] en el conjunto de herramientas Farsight [25] ayudó a tratar de clasificar automáticamente una base de datos de 1230 neuronas de roedores de múltiples fuentes [56]. La amplia muestra no uniforme ofreció grupos neuronales de consistencia limitada, pero las características morfológicas útiles sin embargo, permitieron una clasificación exitosa en casos seleccionados. En una línea similar, los árboles dendríticos de más de 5000 neuronas provenientes de la base de datos NeuroMorpho.org fueron clasificados por un modelo basado en la agrupación no supervisada con expectativa de maximización después de la reducción de dimensionalidad morfométrica mediante el análisis de componentes principales [57]. Las combinaciones específicas de medidas relacionadas con la densidad de ramificación, tamaño total, tortuosidad, ángulos de bifurcación, cuán plano es el árbol y la asimetría topológica capturadas anatómicamente son características dendríticas funcionalmente relevantes en una amplia diversidad de especies y regiones del cerebro. Enfoques similares permitieron la identificación automática de fenotipos estructurales "extrema" entre especies y regiones cerebrales [58].

A una escala de microcircuito más fina, la clasificación automática ha sido adoptada para caracterizar el tipo, la densidad y la geometría de los axones terminales del cerebelo [59] y las espinas dendríticas del hipocampo [60]. La clasificación neuronal podría, en principio, ser

obtenida directamente de la conectividad de red [61], pero el necesario sistema matemático está todavía bajo desarrollo y requerirá conectomas a nivel de neurona de gran escala para pruebas empíricas. Como un *proxy* para la conectividad, las distribuciones de similitud de las ramificaciones fueron utilizadas para clasificar 379 neuronas etiquetadas del lóbulo óptico de un solo cerebro de Drosophila [62]. Los resultados coinciden con el análisis de experto con un subconjunto de 56 categorías en un 91% la precisión estimada. Un nuevo paradigma para la clasificación morfológica adaptó la idea de bioinformática de la alineación de secuencias [63]. Cada una de las 16129 neuronas de Drosophila citadas en [39], representadas como matrices binarias ligeras [64], fueron primeramente integradas en un atlas común cerebral completo. Entonces, masivas comparaciones de similitud por pares en la localización espacial y geometría local rindieron 1052 grupos fenotípicos diferenciados por la agrupación de propagación de afinidad [65]. Este amplio conjunto de clústeres se reorganiza en superclases por agrupación jerárquica con la distancia euclidiana. Muchas de estas superclases coinciden con conocidos tipos neuronales y circuitos de Drosophila, mientras que otros se constituyen como nuevas familias todavía inexploradas [66]. La larva de Drosophila es también un poderoso sistema modelo para investigar el crecimiento dendrítico [67], estereotipia [68] y conectividad sináptica en circuitos motores [69].

En una investigación se utilizaron 17 características morfológicas para describir un conjunto de 100 neuronas dopaminérgicas pertenecientes al tronco cerebral de roedor, una región crítica la cual está implicada en varios trastornos neurológicos, incluyendo la enfermedad de Parkinson. En la misma se compara el desempeño de tres métodos de clasificación populares, máquinas de soporte vectorial, red neuronal de retro-propagación y regresión logística multinomial, con resultados en la clasificación de hasta un 97% de exactitud [44]. Un reciente artículo realizó una caracterización estadística, modelaje y clasificación de los cambios observados en neuronas mutantes gamma de cerebro de Drosophila [70]. Otro estudio aplicó métodos de agrupación semi-supervisada para discriminar con exactitud las estructuras morfológicas de 118 reconstrucciones digitales de interneuronas clasificadas en cuatro grupos por 42 de los principales neurocientíficos del mundo, cuantificadas por cinco propiedades morfométricas del axón y cuatro de las dendritas. Las interneuronas etiquetadas con más fiabilidad fueron clasificadas con más exactitud. Los resultados de clasificación correcta para los cuatro tipos fueron de 100%, 73%, 58% y 56% respectivamente [71].

1.7.2 Patrones de disparo y plasticidad

Caracterizar neuronas electrofisiológicamente puede potencialmente revelar su identidad funcional. Recientes identificaciones automáticas de tipos neuronales por patrones de pulso eléctrico se basan en anteriores análisis descriptivos [72]. Solamente el ancho del pulso eléctrico separa tres tipos principales de neuronas en el campo frontal del ojo del mono macaco, es decir, visual, movimiento y visual-movimiento [73]. El análisis se basó simplemente en el coeficiente de variación y en la prueba de los rangos con signo de Wilcoxon con las correcciones de Bonferroni para comparaciones múltiples. Tanto la corteza visual como la somato-sensorial están entre las regiones más exploradas del cerebro mamífero. Con 46 características electrofisiológicas cuantitativa de 466 interneuronas corticales previamente identificadas como ocho tipos diferentes basados en la nomenclatura Petilla [41], la clasificación jerárquica reveló seis subtipos adaptativos y de disparo rápido [74]. La forma y la duración del pulso eléctrico fueron identificadas como los principales rasgos fisiológicos, especialmente la tasa de pulsos, ligeramente después del bajo de hiperpolarización del primer pulso, así como la anchura de la altura media y latencia del estímulo del segundo pulso. Otros estudios similares reportan parámetros electrofisiológicos distintivos análogos, como la elevación, duración y decadencia de los dos primeros pulsos, más la tasa de disparo y la duración del pulso a 10 MHz [51]. Las propiedades intrínsecas pasivas y activas de la membrana resultaron ser las propiedades fisiológicas claves en la clasificación de las capas 2 y 3 de células piramidales en la corteza prefrontal dorsolateral de macacos [75], así como de capa 2/3 de interneuronas en la corteza visual de ratones [76]. Ambos trabajos utilizan agrupación jerárquica sobre 16-17 mediciones electrofisiológicas, reportando cuatro subtipos neuronales cada uno. Entre las interneuronas del ratón, los patrones de disparo en respuesta a estímulos moderados por encima de la reobase y de la entrada sináptica excitatoria constituyen los parámetros más informativos. En el caso de las células piramidales de macaco, las propiedades más indicativas de la membrana fueron la regularidad de disparo, la resistencia de entrada, la corriente de disparo mínima y la relación corriente a frecuencia. La clasificación automática de interneuronas de expresión de somatostatina, utilizando 19 rasgos fisiológicos de 36 neuronas y en paralelo con 67 rasgos morfológicas de 39 neuronas, de forma independiente identificaron tres tipos neuronales: las células de Martinotti junto con otros dos grupos con axones cortos asimétricos dirigidos a las capas 2/ 3 y medialmente flexionados

[77]. Aunque las reconstrucciones morfológicas y las grabaciones electrofisiológicas estuvieron disponibles conjuntamente sólo para 16 neuronas, los resultados de la clasificación fueron consistentes entre dominios. Tres prototipos glutamatérgicos y cuatro GABAérgicos fueron identificados a partir de 200 neuronas somato-sensoriales [48]. Reanalizar los mismos datos con conjuntos difusos no supervisados suponiendo una diversidad continua cuantitativamente confirmó la casi óptima separación previa, con la sola adición de un subtipo no identificado previamente [78].

1.7.3 Los marcadores moleculares

El tercer pilar de la caracterización neuronal es el análisis bioquímico en una sola célula o a nivel de todo el tejido. La mayoría de las propiedades biofísicas de las neuronas son dictadas por su estado transcripcional, de ahí el énfasis en los perfiles genéticos [79]. Por ejemplo, los niveles de mRNA absolutos de seis canales iónicos diferentes se midieron mediante PCR de tiempo real en y entre seis tipos identificables de las neuronas ganglionares estomatogástricas de cangrejo [6]. El coeficiente de correlación de Pearson, análisis de varianza y pruebas *t*, demostraron empíricamente que las expresiones correlacionadas de subconjuntos de canales particulares determinan salidas electrofisiológicas específicas. Otro estudio seminal utilizó varias réplicas de microarrays de ADN del cerebro de murino [80]. La agrupación jerárquica (con distancia euclidiana) de los genes expresados de forma más significativa (ANOVA, $p < 0.0001$) arrojaron una taxonomía cuantitativa de 12 principales poblaciones reconocidas de neuronas de proyección excitatorias e interneuronas inhibitorias. A pesar de la heterogeneidad de expresión dentro de las regiones individuales, tipos de células homólogas fueron reconocidas a través de neocortex y el hipocampo, por ejemplo, entre las capas somatosensoriales 5/6 y CA1 o entre capas cingulado 2/4 y CA3. El mismo conjunto de datos fue posteriormente reutilizado para investigar las relaciones entre los patrones de microARN y mRNA en cada población neuronal [81] por medio de análisis de red de la co-expresión génica ponderada [82]. Los estudios de desarrollo adoptan cada vez más clasificadores automáticos moleculares para identificar espacialmente o temporalmente grupos y precursores neuronales separados [83]. Entre los numerosos informes sobre el fenotipo neuronal por la tecnología de microarreglos de ADN, varios vincularon la norma de desarrollo morfológico de clases específicas a proteínas individuales [84] [85].

1.8 Conclusiones del Capítulo

En este capítulo se describieron los fundamentos necesarios para la investigación. Primeramente se describió la morfología neuronal con sus partes fundamentales, además de las funciones de las mismas, así como una breve reseña de los tipos de neuronas. Se pudo apreciar la significativa importancia de la morfología neuronal en las funciones cerebrales. También se analizaron los métodos de digitalización de la neurona mediante las técnicas y herramientas disponibles en la actualidad como el uso de microscopio de campo claro asistido por Neurolucida. Se mencionaron las herramientas disponibles más usadas, así como las bases de datos de acceso libre más completa como es el caso de NeuroMorpho.org. Posteriormente se describieron los tipos de datos usados en la clasificación automática de neuronas, así como los algoritmos y programas más usados. Todo esto basado en los estudios y publicación realizados hasta la fecha. El número de publicaciones sobre la clasificación neuronal automática en los últimos tres años ha superado a los de la década anterior. Igualmente notable es el conjunto de datos cada vez de mayor tamaño, de una docena de neuronas a miles en tan sólo 9 años. Esta tendencia probablemente continuará, mostrando las grandes necesidades y oportunidades de investigación en esta área.

CAPÍTULO 2. MATERIALES Y MÉTODOS

En este capítulo se abordan los materiales y métodos utilizados en el trabajo. Primero se describe la base de datos de neuronas reconstruidas usada en la presente investigación. Posteriormente se explican los programas y funciones utilizados para la extracción de rasgos morfológicos de los árboles neuronales. A continuación se exponen los algoritmos usados, primeramente para la selección de atributos y después para la clasificación. En el siguiente epígrafe se abordan los programas utilizados para implementar los algoritmos, detallando las funciones y los índices de cuantificación de los resultados usados en nuestro programa. Seguidamente se muestran la formación de las estructuras de datos y el procedimiento utilizado para la clasificación.

2.1 Base de Datos Yuste

Para el estudio se ha utilizado una base de datos de 318 neuronas divididas en dos grupos, 192 interneuronas y 126 piramidales. En la Figura 2.1 se puede apreciar un ejemplo de una neurona de cada grupo. En la corteza cerebral existen complejos circuitos neuronales compuestos por células piramidales excitatorias e interneuronas inhibidoras. A través de mecanismos inhibitorios en su mayoría, las interneuronas regulan la actividad de las células piramidales y previenen la hiperexcitabilidad. Las interneuronas también tienen importantes propiedades de interconexión, a través de las cuales se sincronizan las células piramidales.

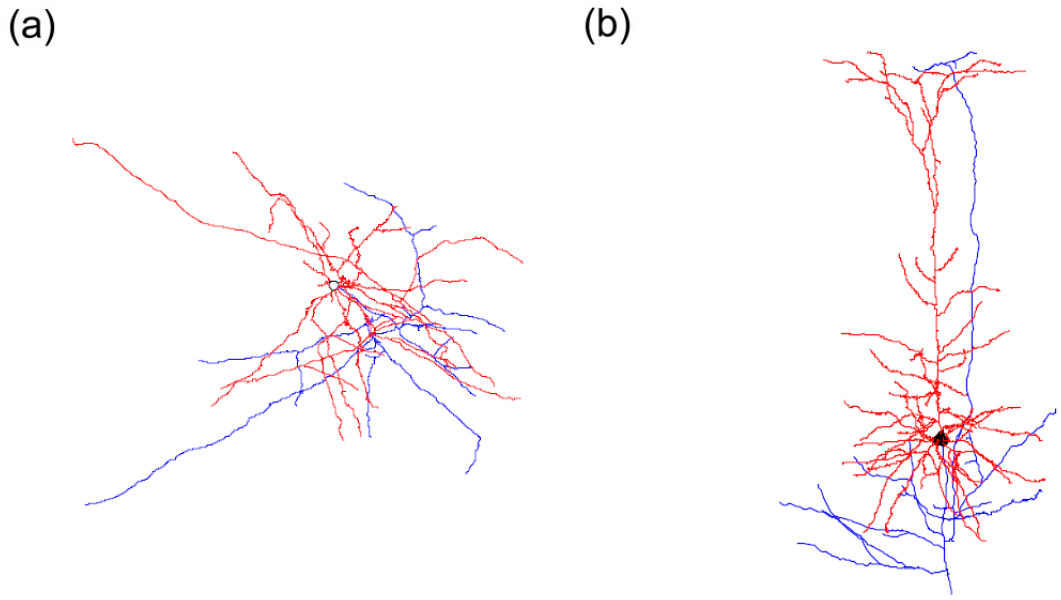

Figura 2.1: Imágenes de neuronas de la neocorteza del ratón. El árbol axonal en azul y las dendritas en rojo. (a) Interneurona, (b) Célula piramidal. Los ejemplos fueron obtenidos de http://www.columbia.edu/cu/biology/faculty/yuste/databases.html

Las 318 neuronas pertenecen a la neocorteza occipital de ratón, las mismas fueron reconstruidas utilizando Neurolucida. Los detalles del procedimiento histológico se pueden consultar en [49]. Las reconstrucciones están disponibles en la web del laboratorio de Rafael Yuste en La Universidad de Columbia en la ciudad de Nueva York (http://www.columbia.edu/cu/biology/faculty/yuste/databases.html), este es uno de los laboratorios que tiene sus reconstrucciones digitales publicadas en la base de datos pública NeuroMorpho.org.

2.1.1 Acondicionamiento de los datos

Las 318 reconstrucciones digitales de neuronas que utilizaremos en esta investigación están divididas en dos clases diferentes, las cuales serán el objetivo de los algoritmos de clasificación supervisada. Las reconstrucciones digitales pueden tener varios formatos, depende de la herramienta con la cual se realizó el trazado neuronal. Muchos de los programas que existen para trabajar con esas reconstrucciones no soportan todos los formatos. Para ayudar a la comunidad de neuroinformáticos con este problema se ha desarrollado una herramienta llamada NLMorphologyConverter descrita en el epígrafe 1.3, la misma es capaz de convertir entre los diferentes formatos sin pérdida de datos, permitiendo la compatibilidad

entre los paquetes de software libre y los comerciales, lo que será necesario en esta investigación. Además tiene otras funcionalidades que permiten manipular los datos de entrada, un ejemplo de esto es la posibilidad de eliminar partes de la neurona al convertirla, lo cual también será de utilidad en el presente trabajo. Esta herramienta junto a NLMorphologyViewer, la cual permite visualizar interactivamente el árbol neuronal en 3D, están disponible de manera gratuita en www.neuronland.org.

En nuestro caso las neuronas fueron adquiridas en formato .DAT y se usó el comando "NLMorphologyConverter neuron.dat neuron.swc SWC" para convertirlas a .SWC ya que es más usado y el necesario para las herramientas que se utilizaron en esta investigación. También se usó el comando "NLMorphologyConverter neuron_in.swc neuron_out.swc SWC --omitAllDends" para separar las dendritas de los axones y tener nuevas neuronas con un solo tipo de estas regiones respectivamente (Figura 2.2). Esto permitirá realizar análisis por separado de varios rasgos morfológicos.

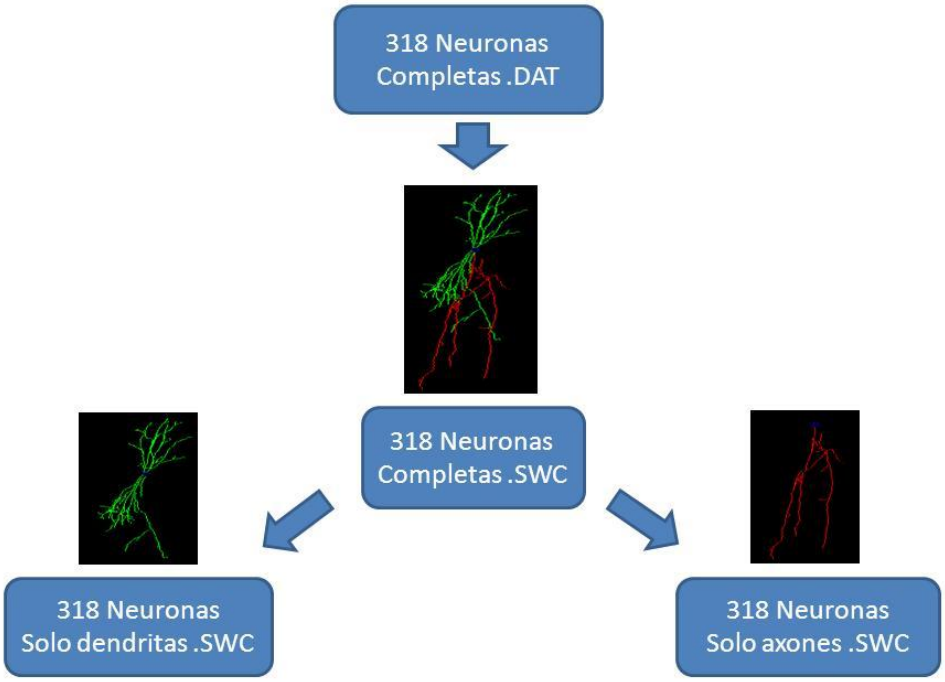

Figura 2.2: Conversión de formato y formación de nuevas neuronas solo con dendritas o axones respectivamente.

Después de este proceso los datos están listos para ser analizados. Contamos con tres grupos de 318 neuronas. En el primer grupo las neuronas están completas, en el segundo son las

mismas neuronas paro con la ausencia de los axones y en el tercer grupo con la ausencia de las dendritas. Todos los grupos con dos clases neuronales, 192 interneuronas y 126 piramidales.

2.2 Herramientas usadas para la extracción de rasgos morfológicos

Como se pudo apreciar en el capítulo anterior, para la clasificación de neuronas se pueden usar tres tipos de datos diferentes: los morfológicos, los electrofisiológicos y los moleculares. En esta investigación solo haremos uso de los rasgos morfológicos extraídos de las reconstrucciones neuronales. Son varias las herramientas que existen para la extracción de rasgos morfológicos de las neuronas, algunas como el L_measure solo cuantifican, pero otras además de cuantificar realizan otras funciones como editar, segmentar, generar, simular y visualizar. Un listado de las herramientas disponibles se puede ver en [4] y [3].

Al realizar un análisis morfométrico neuronal las regiones que más información ofrecen son las ramificaciones de axones y dendritas, esto debido al grado de complejidad morfológica que estas presentan en comparación con el soma y a su variabilidad entre las diferentes neuronas.

2.2.1 Neurolucida

Neurolucida es una poderosa herramienta para la creación y análisis realista, significativo y cuantificable de reconstrucciones neuronales a partir de las imágenes de microscopía de campo claro, confocal o de excitación de dos fotones. Al realizar análisis morfométricos detallados de las neuronas, algunas variables son medidas directamente como el área y perímetro del soma, número de axones y dendritas, el número de nodos, etc. Otras variables son valores calculados como por ejemplo el análisis fractal, análisis de Sholl entre otros.

Este programa es privado y no se pudo obtener, pero el laboratorio de Rafael Yuste, del cual proviene la base de datos de neuronas reconstruidas usadas en esta investigación, nos cedió un conjunto de 67 rasgos que fueron extraídos con Neurolucida. Este conjunto de rasgos puede verse en el Anexo 1.

2.2.2 L_measure

Uno de los programas usados para la extracción de los rasgos es el L_measure, una herramienta libre que es capaz de extraer más de 30 parámetros morfológicos cuantitativos de las reconstrucciones neuronales. El programa puede ejecutarse desde comandos o a través de

una interfaz gráfica amigable escrita en Java. Una representación gráfica de algunos de los rasgos se puede ver en la Figura 2.3. En ella se aprecian un amplio conjunto de métricas dentro de las cuales se encuentran: distancia euclidiana (es la distancia desde cada compartimento al soma), número de bifurcaciones, número de ramas, número de troncos (ramas que están unidas al soma), distancia de camino, volumen y superficie de cada compartimento, orden de ramas con respecto al soma, fragmentación (cantidad de compartimentos entre dos bifurcaciones), asimetría de partición (razón que expresa la diferencia entre ramas que parten de una misma bifurcación), amplitud del ángulo en cada bifurcación (local cuando se mide entre los dos primeros compartimentos y remota cuando se mide entre las dos primeras bifurcaciones), Pk_clásico (razón entre los diámetros de los compartimentos padre e hijos en cada bifurcación). Además de las métricas antes mencionadas, se pueden extraer otras que no quedaron reflejadas en la Figura 2.3, el listado con el conjunto total se puede consultar en el Anexo 2.

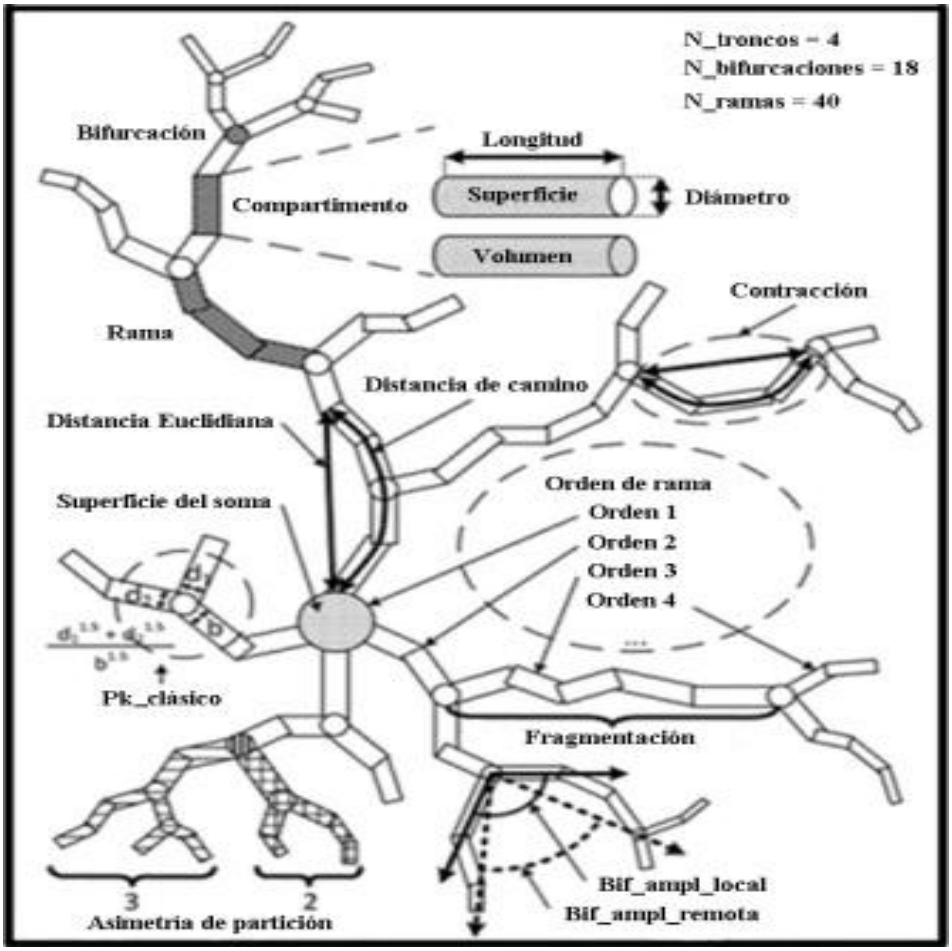

Figura 2.3: Rasgos morfológicos extraídos con L_measure.

En esta investigación para extraer los rasgos se usó la interfaz gráfica donde se cargan todas las reconstrucciones y se escogen qué parámetros deseamos. Después de cargar las neuronas se calcularon todos los rasgos, los cuales se exportan a un documento Excel donde de cada uno de los rasgos se brinda: suma total, mínimo, promedio, máximo y desviación estándar. Posteriormente esa tabla de métricas debe ser depurada ya que en algunos rasgos lo que tiene sentido es el promedio y la desviación estándar mientras que en otros puede ser la suma total o el máximo. Un ejemplo de lo mencionado son los ángulos de las bifurcaciones, en estos no tiene sentido la suma total y si el promedio.

2.2.3 Farsight

Otro de los programas libres usados para extraer rasgos de las neuronas fue el Farsight [25], el cual ofrece un conjunto de herramientas para cuantificar, visualizar, segmentar y editar. Este programa incluye rasgos del L_measure y adiciona otros como por ejemplo la asimetría euclidiana y la distancia euclidiana de los terminales al soma (Figura 2.4).

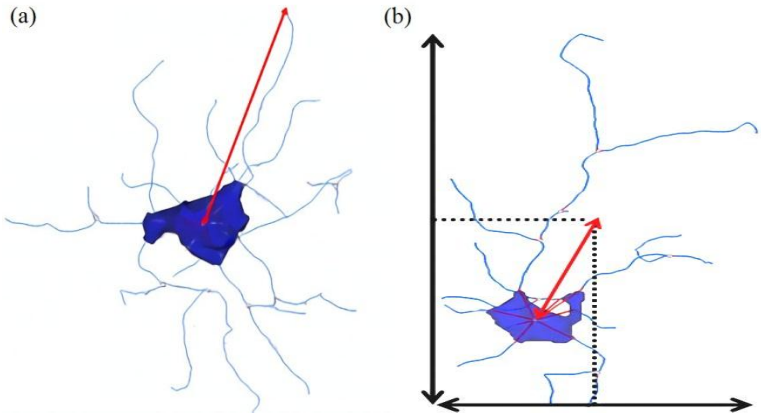

Figura 2.4: (a) Distancia euclidiana de los terminales al soma. (b) Asimetría euclidiana

La versión usada de Farsight fue la 0.4.4 con su módulo TraceEdit. En él se cargan las neuronas reconstruidas y después se busca en el menú la opción de analizar la célula, esto nos da una tabla con todas las mediciones la que se puede exportar a un Excel. Al extraer los rasgos de los tres grupos de neuronas se evitó los rasgos ya obtenidos con L_measure, el listado final se puede ver en el Anexo 3.

2.2.4 TreesToolbox

El último de los programas utilizados para extraer rasgos de la neurona es el TreesToolbox, desarrollado en Matlab, el mismo provee herramientas para reconstruir automáticamente árboles neuronales a partir de imágenes obtenidas con microscopía de excitación de dos fotones, además de editar, visualizar y analizar estas reconstrucciones. Aunque esta herramienta sirve para extraer varios rasgos neuronales, en la presente investigación solo se utilizó para hacer análisis de Sholl.

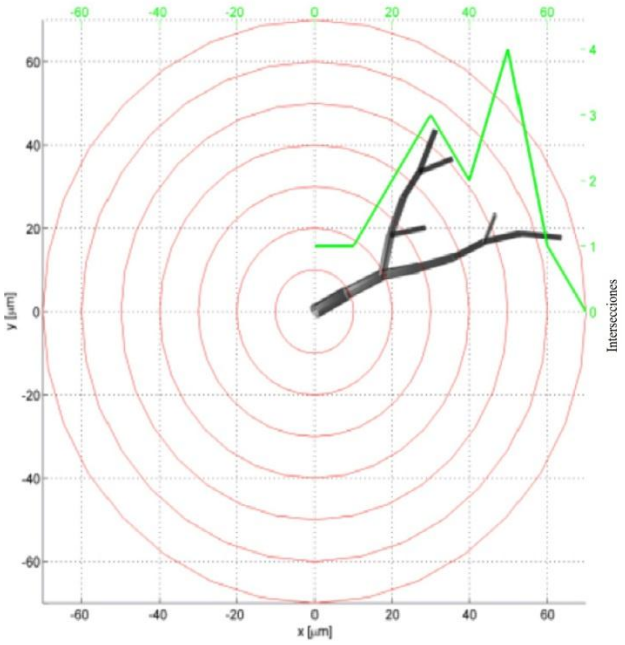

Figura 2.5: Análisis de Sholl realizado con TreesToolbox

El análisis de Sholl consiste en contar las veces que el árbol neuronal se intercepta con círculos en 2D o esferas concéntricas en 3D (Figura 2.5). Para realizar este análisis primero se carga la reconstrucción digital con *neuron =load_tree('neurona.swc')* y después se aplica el comando *sholl_tree (neuron, dd, '-3s')* donde *dd* es un vector con el valor de los diámetros de las esferas concéntricas y '-3s' para mostrar las intersecciones 3D. La lista de diámetros utilizados para los tres grupos de neuronas se puede ver en el Anexo 4.

2.3 Reducción de la dimensionalidad de los datos

Para obtener los mejores resultados en una tarea de clasificación es necesario realizar previamente un correcto proceso de selección de atributos o rasgos, mediante el cual se busca automáticamente el mejor subconjunto de atributos del conjunto de datos. La noción de

"mejor" es relativa al problema que está tratando de resolverse, pero por lo general significa una precisión más alta.

El objetivo es navegar a través de todas las combinaciones posibles de atributos que se podían elegir del conjunto de datos y localizar la mejor o una combinación suficientemente buena que mejore el rendimiento del clasificador.

Tres beneficios claves de realizar una selección de atributos sobre la base de datos son:

- Reducción del sobreajuste.
- Se mejora la precisión en la clasificación.
- Se reduce el tiempo de entrenamiento.

En esta investigación se usó la herramienta de selección de atributos del programa Weka versión 6.3.11 el cual es muy potente y tiene implementado varios algoritmos para este fin. La selección de atributos en Weka está separada en dos partes, el evaluador de atributos y el método de búsqueda [86].

2.3.1 Evaluador de atributos

El evaluador de atributos es el método por el cual un subconjunto de atributos es evaluado. A continuación se mencionan los evaluadores utilizados en esta investigación.

CfsSubsetEval: Evalúa el valor de un subconjunto de atributos considerando la capacidad de predicción individual de cada característica junto con el grado de redundancia entre ellos.

SymmetricalUncertAttributeSetEval: Evalúa el valor de un conjunto de atributos mediante la medición de la incertidumbre simétrica con respecto a otro conjunto de atributos.

SVMAttributeEval: Evalúa el valor de un atributo mediante el uso de un clasificador de *Máquinas de Soporte Vectorial*.

WrapperSubsetEval: Evalúa los subconjuntos de atributos mediante el uso de un clasificador. Emplea validación cruzada para estimar la exactitud del esquema de aprendizaje en cada conjunto. A continuación se mencionan los métodos de búsqueda utilizados en esta investigación.

2.3.2 Método de búsqueda

El método de búsqueda es la manera estructurada en que el espacio de búsqueda de posibles subconjuntos de atributos es navegado basado en el subconjunto de evaluación. Los métodos de referencia incluyen búsquedas aleatorias, búsquedas exhaustivas entre otras.

Ranker: Clasifica y ordena los atributos por sus evaluaciones individuales.

GreedyStepwise: Realiza una búsqueda voraz hacia adelante o hacia atrás a través del espacio de subconjuntos de atributos. Puede comenzar con todos los atributos o con ninguno. Se detiene cuando la adición o eliminación de cualquier atributo de los restantes resulta en una disminución en la evaluación.

GeneticSearch: Realiza una búsqueda usando un algoritmo genético simple.

2.4 Algoritmos utilizados en la clasificación de neuronas

Para la clasificación de neuronas reconstruidas en esta investigación se usaron cinco algoritmos diferentes: Naive Bayes, Regresión logística multinomial, Perceptrón Multicapa, Máquinas de soporte vectorial y *k*-vecino más cercano, los cuales han presentado buenos resultados en investigaciones semejantes [44] [49]. Todos se usaron según están implementados en el software Weka versión 3.6.11 el cual será descrito con más detalles en el siguiente epígrafe.

Naive Bayes

El clasificador Naive Bayes es una red bayesiana donde la clase no tiene padres y cada atributo tiene la clase como su único padre. El mismo está basado en el teorema de Bayes con suposición de independencia entre los predictores. El modelo es fácil de construir, con una estimación iterativa de parámetros no complicada lo que resulta especialmente útil para conjuntos de datos muy grandes. A pesar de su simplicidad, este clasificador con frecuencia trabaja sorprendentemente bien y es ampliamente utilizado debido a que, a menudo, supera a otros métodos de clasificación más sofisticados [87].

El clasificador Naive Bayes tiene una interfaz relativamente simple en Weka. Permite usar estimador de *kernel* para atributos numéricos en lugar de una distribución normal y puede utilizar discretización supervisada para convertir atributos numéricos en nominales [86].

Logistic

La regresión logística multinomial (MLR) es un popular modelo de clasificación probabilística discriminativo que tiene buenos resultados en los problemas de clasificación de bio-imágenes. El MLR es considerado como uno de los mejores clasificadores probabilísticos. Mide los dos primeros mejores y registra la pérdida de precisión de clasificación a través de un número de pasos.

$$y_i = \begin{cases} 1, si\ x\ pertenece\ a\ la\ clase\ i \\ 0, otros \end{cases} \quad (2.1)$$

Donde $i = \{2 \ldots m\}$ y m representa el número de clases de salida. Si $m = 2$ (problemas binarios), la técnica se conoce como regresión logística, mientras que cuando $m > 2$ la técnica es conocido como MLR [9]. Usando MLR, la probabilidad de que x pertenece a la clase i es:

$$P(y_i = 1|x; w) = \frac{exp(w^{\tau}{}_i x)}{\sum_{j=1}^{m} exp(w_j x)} \quad (2.2)$$

Donde como resultado de la normalización:

$$\sum_{i=1}^{m} P(y_i = 1|x; w) = 1 \quad (2.3)$$

En la fórmula anterior, P es la variable de la predicción, w_i denota el vector de peso, y el superíndice τ es el vector transposición [88].

El clasificador MLR implementado en Weka con el nombre de Logistic mejora la eficiencia del algoritmo mediante la aplicación de un estimador de cresta. Además, el modelo original de regresión logística no se ocupa de los pesos de instancia; sin embargo Weka modifica el algoritmo para que pueda manejar los pesos de instancia [86].

Perceptrón multicapa

Un perceptrón multicapa (MLP) es una red neural de retropropagación con una o más capas entre la entrada y la salida. La red se crea mediante un algoritmo MLP, además se puede supervisar y modificar durante el tiempo de entrenamiento. Los nodos de esta red son todos sigmoides (a excepción de cuando la clase es numérica, en este caso los nodos de salida se convierten en unidades lineales no umbralizadas).

La red neuronal de retropropagación es esencialmente una red de elementos simples de procesamiento que trabajan juntos para producir una salida compleja. El algoritmo de

propagación hacia atrás lleva a cabo el aprendizaje en una red neuronal de alimentación hacia delante de múltiples capas [89].

Cada capa está compuesta de unidades. Las entradas a la red corresponden a los atributos medidos para cada secuencia de entrenamiento. Las entradas alimentan a las unidades simultáneamente formando la capa de entrada. Cuando los datos pasan a través de la capa de entrada se ponderan y alimentan simultáneamente a una segunda capa de unidades, conocida como capa oculta. Las salidas de las unidades de la capa oculta se pueden introducir a otra capa oculta y así sucesivamente. Cuando termina el proceso de propagación hacia delante comienza la retropropagación, en este momento se compara la salida generada por la red con la salida deseada y el error obtenido de esta diferencia se propaga hacia atrás para ajustar los pesos en las unidades de cada capa y de esa manera obtener clasificador más eficaz, la configuración de la red neuronal usada en esta investigación se explica en el epígrafe 2.5.1.

SMO

Optimización Mínima Secuencial (SMO) es un algoritmo usado para el entrenamiento de las máquinas de soporte vectorial (SVMs). Las máquinas de soporte vectorial constituyen una poderosa y robusta herramienta destinada a labores de clasificación aunque tienen como limitante que su algoritmo básico de entrenamiento es lento y complejo, especialmente en grandes problemas. Como este problema necesitaba una solución, surgió SMO, un nuevo algoritmo de entrenamiento que perfeccionó el antiguo, el cual usaba la programación cuadrática (QP) numérica como un lazo interior. El SMO rompe este gran problema de QP en una serie de pequeños problemas QP posibles. Estos pequeños problemas de QP se resuelven analíticamente, lo que evita el uso de la QP numérica como un lazo interior, la cual es muy costosa en tiempo de ejecución. La cantidad de memoria necesaria para el SMO depende linealmente del tamaño del conjunto de entrenamiento, lo que le permite al SMO manejar grandes conjuntos de entrenamiento. El tiempo de cálculo del SMO está dominado por la evaluación de SVM; por lo tanto, el SMO es más rápido para las SVM lineales y conjuntos de datos dispersos [90].

IBK

Algoritmo de aprendizaje Basado en *K* Instancias (IBK) es un clasificador *k*-vecino más cercano (KNN) que utiliza la misma distancia métrica. KNN es un tipo de aprendizaje perezoso, donde la función sólo es aproximada a nivel local y todos los cálculos se aplazan hasta la clasificación. En este algoritmo el objeto se clasifica por el voto de la mayoría de sus vecinos.

Las muestras de entrenamiento son descritas por atributos numéricos *n*-dimensionales. Cada muestra representa un punto en el espacio *n*-dimensional. De esta manera, todas las muestras de entrenamiento se almacenan en el espacio *n*-dimensional. Cuando entra una muestra desconocida, el clasificador KNN busca en el espacio las *k* muestras de entrenamiento que están más cerca de la muestra desconocida, estas muestras de entrenamiento son los *k* vecinos más cercanos de la muestra desconocida. La cercanía puede ser definida en términos de distancia euclidiana como:

$$dist(p,q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \quad (2.4)$$

La muestra desconocida se asigna a la clase más común entre sus *k* vecinos más cercanos. Cuando *k* = 1, la muestra desconocida se le asigna la clase de la muestra de entrenamiento que está más cerca en el espacio *n*-dimensional. El clasificador asigna la misma importancia a cada atributo [91].

2.5 Uso de Weka y Matlab2weka para implementar la clasificación

Weka es un conjunto de algoritmos de aprendizaje automático para tareas de minería de datos. Los algoritmos se pueden aplicar directamente a un conjunto de datos o llamar desde su propio código Java. Weka contiene herramientas para el pre-procesamiento de datos, clasificación, regresión, agrupamiento, reglas de asociación y la visualización. También es muy adecuado para el desarrollo de nuevos sistemas de aprendizaje de máquina. Es un software de código abierto publicado bajo la Licencia Pública General de GNU.

Aunque Weka proporciona interfaces gráficas de usuario (GUI) fantásticas, a veces es necesario tener una mayor flexibilidad en la programación, lo que se puede lograr usando las funciones directamente ya que están implementadas en Java.

En este trabajo se decidió implementar las funciones de Weka en Matlab el cual permite la comunicación con Java. Esto se realizó con el objetivo de poder usar los algoritmos de Weka en conjunto con otras cajas de herramientas de Matlab como son el PRTool y el Spider entre otros, permitiendo una mayor flexibilidad e integración de las herramientas. Para la implementación de los algoritmos en Matlab se hizo uso de Matlab2weka, un toolbox que está disponible en www.mathworks.com. El mismo consta de varias funciones las cuales permiten usar desde Matlab las clases de Java implementadas por Weka. Algunas de las funciones que permiten el uso de los datos por ambos programas se muestran a continuación.

loadARFF(filename): Carga datos de un archivo Weka ARFF en un objeto Java weka de instancias para su uso por las clases Weka.

saveARFF(filename,wekaOBJ): Guarda un objeto Java weka de instancias, al formato Weka ARFF.

weka2matlab(wekaOBJ,mode): Convierte los datos Weka, almacenados en un objeto Java weka de instancias a datos de Matlab.

matlab2weka(name, feature...): Convierte los datos de Matlab en un objeto Java weka de instancias para su uso por las clases de Weka.

2.5.1 Funciones usadas para clasificar

Para la clasificación automática de neuronas en este estudio se usaron cinco clasificadores. Todo el programa se desarrolló en Matlab versión R2013a, haciendo uso de las funciones implementadas por Weka para los cinco algoritmos de clasificación, todo esto utilizando el *toolbox* Matlab2weka. En este estudio fue necesario realizar distintas comparaciones con diferentes datos de entrada, pero en todas las ocasiones el programa de clasificación fue el mismo, solo variando el clasificador (ver Anexo 5).

La primera de las funciones necesarias para realizar el proceso de clasificación es la usada para cargar los datos, es este caso se usó loadARFF(filename), la cual carga el archivo de datos Weka ARFF en una variable de Matlab de tipo weka.core.intances a la cual se podrán aplicar las funciones Java de Weka. Estas pueden consultarse en la ayuda del programa o en [86]. Como segundo paso es necesario construir el objeto clasificador lo cual se hace con las funciones mostradas a continuación:

Clasificador1 = weka.classifiers.bayes.NaiveBayes(). Tanto este clasificador como los demás se implementaron con los parámetros por defecto de Weka, en este caso sin usar el estimador de *kernel* ni la discretización supervisada.

Clasificador2 = weka.classifiers.functions.Logistic(). Con valor de la cresta de 1.0E-8 y sin límite en el número de iteraciones a realizar.

Clasificador3 = weka.classifiers.functions.MultilayerPerceptron(). Esta red neuronal está diseñada con una sola capa oculta y un número de nodos igual a la mitad de la suma de los atributos y las clases, todos los nodos son sigmoides. La razón de aprendizaje es de 0.3 y el momento de 0.2.

Clasificador4 = weka.classifiers.functions.SMO(). Para este clasificador de máquina de soporte vectorial se usó un *kernel* polinomial aunque pueden usarse otros. Un valor de épsilon de 1.0E-12 y un filtro para normalizar los datos de entrenamiento.

Clasificador5 = weka.classifiers.lazy.IBk(). Para el clasificador de *k* vecinos más cercanos se tomó como 5 el valor de *k* ya que en pruebas preliminares fue el de mejores resultados. No se usó ponderación de distancias y como algoritmo de búsqueda de los vecinos más cercanos se utilizó un algoritmo lineal basado en la distancia euclidiana.

Después de haber creado el objeto clasificador se procede a la clasificación y evaluación mediante validación cruzada:

```
eval = weka.classifiers.evaluation(data)

eval.crossvalidatemodel(classifier, data, numFolds, random, forPredictionsString)
```

En la primera línea se creó la clase de Weka que sirve para evaluar los modelos de aprendizaje automático, la cual inicializa todos los contadores necesarios para evaluar la clasificación. La segunda línea es un método de esa clase el cual realiza una validación cruzada estratificada, el mismo tiene cinco parámetros de entrada:

- classifier: Clasificador que será usado en la validación cruzada.
- data: Los datos en los cuales se realizará la validación cruzada.
- numFolds: El número de grupos para realizar la validación cruzada.
- random: Generador de números aleatorios para la aleatorización.

- forPredictionsString: Parámetros de argumentos variables que, si se proporcionan, se mantenga un StringBuffer para imprimir predicciones para un rango de atributos a la salida y un booleano (verdadero si la distribución se va a imprimir)

En nuestro estudio se usaron los cinco clasificadores descritos anteriormente, los datos dependiendo de los pruebas realizadas, una validación cruzada de 10 ya que es el valor por defecto y además comúnmente usado en investigaciones similares. Para la aleatorización se usó el java.util.Random y por último sin imprimir la distribución. Valorando la cantidad de casos de la base de datos se decidió repetir el proceso de clasificación diez veces y promediar los resultados, de esa manera asegurar que el resultado sea fiable.

2.5.2 Índices de cuantificación de los resultados

Después de realizada la validación cruzada descrita anteriormente se procede a la cuantificación de los resultados, para esto se hace uso de métodos ya implementados en la clase weka.classifiers.Evaluation(). En esta investigación se usaron los siguientes índices de desempeño.

- **Porcentaje de clasificación correcta (CC):** Nos ofrece el porcentaje de instancias clasificadas correctamente.

  CC = eval.pctCorrect().

- **Sensibilidad:** Es la razón entre los verdaderos positivos y la suma de los verdaderos positivos más los falsos negativos. Este resultado se calcula por clases y se promedia ponderado.

  Sensibilidad = eval.weightedRecall().

- **Precisión:** Es la razón entre los positivos clasificados correctamente y el total de predichos como positivos. Este resultado se calcula por clases y se promedia ponderado.

  Precision = eval.weightedPrecision().

- **Medida F:** Combina las medidas Sensibilidad y Precisión como se define en la siguiente ecuación. Este resultado se calcula por clases y se promedia ponderado.

  Medida F = eval.weightedFMeasure();

$$\text{Medida F} = \frac{2*(\text{Sensibilidad})*(\text{Precisión})}{(\text{Sensibilidad})+(\text{Precisión})} \quad (2.5)$$

- **AUC:** Nos devuelve el área bajo la curva ROC, la curva ROC es un representación gráfica entre la razón de verdaderos positivos y razón de falsos positivos, mientras mayor sea el área bajo esta curva mayor será la eficiencia del clasificador binario. Esta medida es comúnmente usada en problemas de clasificación. Este resultado se calcula por clases y se promedia ponderado.

  AUC = eval.weightedAreaUnderROC()

Además de los índices mostrados previamente, Weka también nos puede brindar la matriz de confusión, las razones de verdadero positivo, verdadero negativo, falso positivo y falso negativo, el estadístico kappa, entre otros. Todos haciendo uso de los métodos de la clase *evaluation*.

2.6 Formación de estructura de datos para la clasificación

Esta investigación se divide en tres procesos de clasificación fundamentales. El primero consiste en una comparación del desempeño de los cinco algoritmos de clasificación mencionados en el epígrafe anterior, aplicados a doce conjuntos de rasgos diferentes. El segundo consiste en comparar los resultados de la clasificación de los rasgos extraídos con Neurolucida [49], con los rasgos que ofrece el L_measure, el Farsight y el análisis de Sholl del TreesToolbox, además de la unión de todos los rasgos. Por último se realiza una comparación entre los rasgos extraídos de los axones y los rasgos extraídos de las dendritas para comprobar cuál de estas secciones diferencia más a las neuronas piramidales de las interneuronas.

Aunque los datos de entrada en cada comparación son diferentes, el algoritmo de clasificación siempre es el mismo y está estructurado de la siguiente manera:

1. Cargar el fichero ARFF que contiene la matriz de rasgos de las dos clases a clasificar.
2. Se crea la clase clasificador según el algoritmo que se vaya a utilizar.

   Clasificador = weka.classifiers.bayes.naivebayes();

3. Antes de clasificar se inicia un ciclo *for* para repetir la clasificación 10 veces y promediar los resultados.

4. Dentro del ciclo *for* se crea la clase eval = weka.classifiers.evaluation(data) para inicializar los contadores necesarios para la evaluación.
5. A continuación realiza la validación cruzada con el clasificador creado previamente: eval.crossValidateModel(Clasificador,data,10,random,array);
6. Se aplican los métodos de la clase 'evaluation' para obtener los resultados de la clasificación a través de los diferentes estadísticos:

   CC(i) = eval.pctCorrect();

   Precision(i) = eval.weightedPrecision();

   Sensibilidad(i) = eval.weightedRecall();

   MedidaF(i) = eval.weightedFMeasure();

   ROC_Area(i) = eval.weightedAreaUnderROC();
7. Se cierra el ciclo *for* y se promedian los valores de las 10 iteraciones, los resultados son exportados en formato xls.

Estas operaciones se ejecutan cinco veces, una con cada clasificador. También puede escribirse un programa que contenga una sección para cada clasificador y ejecutarlo una sola vez. El código usado en esta investigación puede verse en el Anexo 5.

2.6.1 Comparación de los clasificadores con diferentes grupos de rasgos

Esta prueba inicial tiene como objetivo principal comprobar el comportamiento de los cinco algoritmos de clasificación en los tipos de rasgos que se usan en esta investigación, lo cual servirá junto con los resultados de las pruebas que se realizarán en las secciones siguientes para determinar cuáles de estos algoritmos sobresalen o se quedan rezagados con respecto a los demás.

Para realizar esta prueba se tomaron los rasgos extraídos de cada programa por separado, también tomando de manera independiente los rasgos obtenidos de la neurona completa, solo los axones y solo las dendritas. Como se usaron cuatro programas y cada uno obtuvo rasgos tres veces, se formaron doce bases de datos con diferentes rasgos.

Para evitar el sobreajuste en la clasificación se aplicó una reducción de rasgos en aquellas bases de datos que tenían más de 25 rasgos. Para lograr esto se usó el selector de atributos de Weka, en esta ocasión con SymmetricalUncertAttributeEval como evaluador y Ranker como

método de búsqueda, la cantidad de atributos fueron ajustados hasta tener el mejor desempeño. Las doce bases de datos quedaron como se muestra en la Tabla 2.1. A cada una de las doce bases de datos se le aplicaron los cinco clasificadores siguiendo el procedimiento descrito en la sección anterior.

Tabla 2.1 Conjuntos de rasgos separados por programas y sección neuronal

Programa	Sección neuronal	# Rasgos
Neurolucida	Soma	6
Neurolucida	axón	20
Neurolucida	Dendritas	20
L_measure	Completa	22
L_measure	axón	22
L_measure	Dendritas	22
Farsight	Completa	14
Farsight	axón	14
Farsight	Dendritas	14
TreesToolbox	Completa	14
TreesToolbox	axón	14
TreesToolbox	Dendritas	14

2.6.2 Comparación de rasgos de axones con rasgos de dendritas

Para realizar la segunda prueba se conformó una base de datos con todos los rasgos obtenidos de las dendritas y otra con todos los rasgos obtenidos de los axones, no importando el programa con que fueron extraídos. El objetivo consiste en identificar qué rasgos aporta más a la clasificación, es decir, qué sección neuronal diferencia más las neuronas piramidales de las interneuronas.

Antes de aplicar los algoritmos de clasificación se redujo la cantidad de atributos de cada grupo hasta 50 usando SVMAttributeEval como evaluador y Ranker como método de búsqueda. Posteriormente se usó el mismo procedimiento de clasificación y los mismos filtros de selección de atributos que los aplicados en la sección anterior (ver Tabla 2.2).

2.6.3 Comparación de Neurolucida con Lmeasure-Farsight-TreesToolbox y ambos

Las pruebas desarrolladas en este epígrafe constituyen la parte más importante de la investigación, ya que se tratan de mejorar los resultados en la clasificación con los rasgos obtenidos con el programa neurolucida como se usó en [49]. Para esto se tomaron los rasgos extraídos de otros tres programas, L-measure, Farsight y TreesToolbox como se explica en el epígrafe 2.2.

Antes de aplicar los clasificadores se formaron tres bases de datos principales. La primera está conformada por 67 rasgos obtenidos con Neurolucida. La segunda está formada por la suma de todos los rasgos extraídos con los tres programas usados en esta investigación. La última está formada por la unión de los dos conjuntos anteriores. Para equilibrar la cantidad de rasgos de los tres conjuntos, los dos últimos fueron reducidos a 67 rasgos usando SVMAttributeEval como evaluador y Ranker como método de búsqueda.

Una vez que las tres bases de datos han sido igualadas, se comienza la clasificación utilizando cada uno de los cinco algoritmos descritos en el epígrafe 2.4. Primeramente sin aplicar reducción de rasgos, después utilizando seis filtros supervisados de selección de atributos formados por la combinación de diferentes métodos de evaluación y búsqueda (ver Tabla 2.2). El procedimiento para la clasificación es el mismo que el aplicado en la sección anterior.

Tabla 2.2 Filtros utilizados para la reducción de atributos.

Método de evaluación	**Método de búsqueda**
SVMAttributeEval	Ranker
CfsSubsetEval	GreedyStepwise-Forward
CfsSubsetEval	Genetic
WrapperSubsetEval	GreedyStepwise-Forward
WrapperSubsetEval	GreedyStepwise-Backward
WrapperSubsetEval	Genetic

2.7 Prueba de hipótesis estadísticas en la evaluación

En la primera sección, donde se evalúan varios algoritmos de clasificación en varias bases de datos, la prueba estadística que se usó es la recomendada por Demsar [92] ya que es una de las

más usadas en este tipo de análisis, el cual propone la prueba no paramétrica de Fridman. Notamos que las pruebas no paramétricas son apropiadas en el ámbito del Aprendizaje Automático pues estas no asumen distribución normal ni homogeneidad en las varianzas y los resultados empíricos sugieren que estas son más fuertes o potentes que otras pruebas paramétricas como ANOVA [92]. Para implementar la prueba estadística se usó el paquete de R llamado "scmamp" (Statistical Comparison of Multiple Algorithms in Multiple Problems) descrito en [93], el cual tiene como objetivo asistir en el análisis estadístico de una forma simple a los investigadores, además incluye algunas herramienta para mostrar los resultados que resultan muy cómodas cuando hacemos las comparaciones entre diferentes algoritmos.

La segunda sección tiene algunas semejanzas con la primera ya que se aplicaron varios algoritmos de clasificación en tres bases de datos, pero en esta ocasión no deseamos comparar los algoritmos sino comparar si algunos de los tres grupos de rasgos mostraron resultados significativamente superiores a los demás, esto implica un análisis trasversal y para esto se usó la prueba de Kruskal-Wallis, un método no paramétrico para tres o más poblaciones. Para realizar este análisis estadístico también se usó el paquete R.

2.8 Conclusiones del Capítulo

En el presente capítulo primeramente se describió la base de datos utilizada y su acondicionamiento, separando los axones y las dendritas en las reconstrucciones neuronales. A continuación se explicaron las herramientas que se usaron para la extracción de los rasgos morfológicos: Neurolucida, L_measure, Farsight y TreesToolbox.

Después se expusieron los algoritmos de Weka usados para realizar la selección de atributos y los cinco clasificadores implementados. Estos algoritmo fueron Naive Bayes, regresión logística multinomial, perceptrón multicapa, máquinas de soporte vectorial y *k*-vecino más cercano. Posteriormente se abordaron el uso de Weka y matlab2weka para implementar los algoritmos, detallando las funciones e índices de cuantificación que se utilizaron. Seguidamente se describió cómo se formaron las estructuras de datos a ser clasificadas. Este proceso de dividió en tres etapas, primero la evaluación de los clasificadores en varios conjuntos de rasgos, segundo la comparación de los resultados entre tres grupos de rasgos: los extraídos con Neurolucida, los obtenidos con Lmeasure-Farsitgh-TreesToolbox y la unión de

ambos conjuntos. Por último se enunció la comparación entre los rasgos de los axones y de las dendritas.

CAPÍTULO 3. RESULTADOS Y DISCUSIÓN

En este capítulo se analizan los resultados de todas las pruebas realizadas en cuanto a los índices de efectividad: porciento de clasificación correcta (CC), sensibilidad, precisión, medida F y área bajo la curva ROC (*AUC por sus siglas en inglés*). Estos índices fueron obtenidos después de promediar los resultados de diez iteraciones con validación cruzada de diez. Primeramente se comparan los algoritmos de clasificación aplicando cada uno a todos los conjuntos de rasgos por separado, para validar los resultados se aplica la prueba estadística de Friedman. En segundo lugar se separarán los rasgos de las dendritas y de los axones para comprobar cuales aportaban más a la clasificación. A continuación se comparan los resultados usando varios clasificadores y varias estrategias para la selección de atributos, aplicados a tres conjuntos: los rasgos de Neurolucida, los rasgos de Lmeasure-Farsight-TreesToolbox y la unión de ambos conjuntos. Para determinar cuál conjunto resulta más eficiente se realiza un análisis estadístico mediante la prueba de Kruskal-Wallis y Mann-Whitney. Además se muestra un listado con las combinaciones que obtuvieron los mejores resultados y los rasgos morfológicos más repetidos en las mismas.

3.1 Resultados de los algoritmos aplicados a las conjuntos de rasgos individuales

El primer experimento realizado en este trabajo se realiza con el objetivo de comparar el desempeño de 5 clasificadores aplicados a rasgos morfológicos neuronales. Esto, junto a comparaciones entre los diferentes grupos de rasgos mostrados en los epígrafes siguientes, nos ayudará a sacar conclusiones sobre los mejores algoritmos y los rasgos morfológicos en la tarea de clasificar automáticamente las neuronas piramidales y las interneuronas.

Los cinco algoritmos seleccionados fueron Naive Bayes (NB), regresión logística multinomial (LR), perceptrón multicapa (MLP), optimización mínima secuencial (SMO) y *k*-vecinos más cercanos, implementados en Weka como (IBK). Estos clasificadores se aplicaron a 12 conjuntos de rasgos los cuales provienen de 4 herramientas para la extracción de rasgos morfológicos y 3 conjuntos de neuronas.

Tabla 3.1. Resultados de la clasificación en múltiples conjuntos usando Naive Bayes.

Grupos de Rasgos	CC	Precisión	Sensibilidad	Medida F	AUC
farsightAxón14	71.57	0.73	0.72	0.72	0.80
farsightDend14	73.99	0.74	0.74	0.73	0.79
farsightNeuron14	77.01	0.77	0.77	0.77	0.83
LmeasAxón22	71.20	0.72	0.71	0.71	0.79
LmeasDend22	74.81	0.75	0.75	0.74	0.81
LmeasNeuron22	75.82	0.77	0.76	0.76	0.83
lucidaAxón20	72.11	0.75	0.72	0.72	0.82
lucidaDend20	74.43	0.74	0.74	0.74	0.80
lucidaNeuron6	60.88	0.62	0.61	0.61	0.64
sholl_Neuron14	54.06	0.67	0.54	0.51	0.67
shollAxón14	57.30	0.69	0.57	0.56	0.67
shollDend14	67.42	0.67	0.67	0.67	0.66

Los resultados mostrados en la Tabla 3.1 provienen de aplicar el calificador Naive Bayes 10 veces con validación cruzada de 10 a cada conjunto de rasgos y promediarlos. El mismo procedimiento se aplicó a los 4 restantes algoritmos de clasificación y sus resultados se pueden ver en el Anexo 6. En estos se puede apreciar que en ningún conjunto los resultados fueron buenos y que los análisis de Sholl realizados con la herramienta TreesToolbox fueron los de peor desempeño.

Tabla 3.2. Promedio de las clasificaciones sobre los 12 conjuntos de rasgos.

Clasifi-cador	CC		Sensibilidad		Precisión		Medida F		AUC	
	Media	DesvE	Media	DesvE	Media	DesvE	Media	DesvE	Media	DesvE
NB	69.22	7.67	0.69	0.08	0.72	0.05	0.69	0.08	0.76	0.07
LR	74.48	6.59	0.74	0.07	0.74	0.07	0.74	0.07	0.80	0.07
MLP	73.89	6.45	0.74	0.06	0.74	0.06	0.74	0.07	0.79	0.07
SMO	72.20	6.85	0.72	0.07	0.69	0.14	0.69	0.12	0.68	0.10
IBK	73.62	6.49	0.74	0.06	0.74	0.07	0.74	0.07	0.79	0.08

En la Tabla 3.2 se pueden apreciar la media y la desviación estándar de los índices de desempeño en los 12 conjuntos de rasgos. En la misma se puede observar que para estos grupos, ninguno de los algoritmos obtuvo resultados sobresalientes. A continuación se

expondrán los resultados de la prueba estadística de Friedman propuesta por Demsar en [92] para este tipo de comparaciones.

3.1.1 Análisis estadístico usando CC

Al realizar el análisis se usó el porcentaje de clasificación correcta (CC) mostrado en la Tabla 3.3, y se aplicó la prueba estadística de Friedman con el objetivo de determinar si existen diferencias estadísticamente significativas entre los algoritmos enfrentados. Para implementar la prueba se usó el paquete "scmamp" de R, el mismo tiene como objetivo simplificar el análisis de los resultados por medio de gráficos y tablas.

Tabla 3.3. Resultados de la clasificación CC usando todos los clasificadores

Grupos de Rasgos	NB	LR	MLP	SMO	IBK
farsightAxón14	71.57	74.03	71.10	74.50	75.94
farsightDend14	73.99	77.67	76.07	76.45	77.61
farsightNeuron14	77.01	77.11	74.47	76.67	75.50
LmeasAxón22	71.20	75.06	73.43	73.02	73.30
LmeasDend22	74.81	81.35	79.43	76.82	78.18
LmeasNeuron22	75.82	76.04	78.93	77.45	80.75
lucidaAxón20	72.11	77.11	73.27	75.72	71.04
lucidaDend20	74.43	83.55	84.91	80.66	83.24
lucidaNeuron6	60.88	60.69	60.03	60.38	62.86
sholl_Neuron14	54.06	71.42	75.22	65.94	72.45
shollAxón14	57.30	63.58	65.38	60.03	61.64
shollDend14	67.42	76.13	74.47	68.71	70.88

En la Figura 3.1 se representa de forma gráfica el comportamiento de los cinco algoritmos según el porcentaje de instancias clasificadas correctamente. En la misma se puede apreciar que ninguno sobresale sobre los demás y que el Naive Bayes es el de peores resultados.

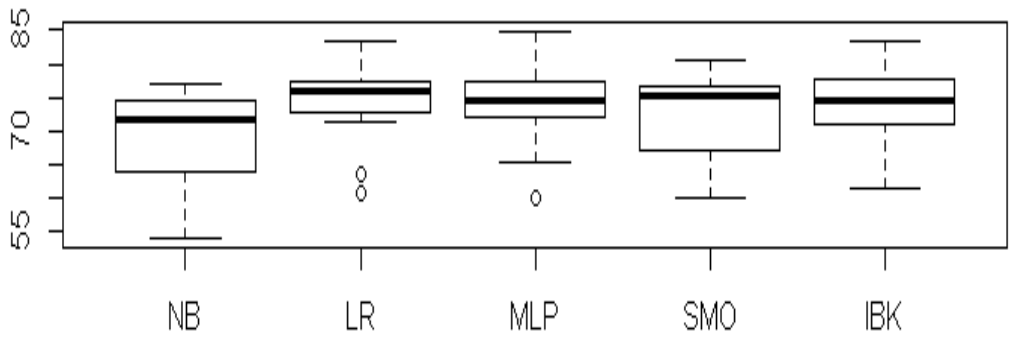

Figura 3.1: Gráfico de cajas, se observa en cada caja el comportamiento de cada clasificador según el porcentaje de instancias clasificadas correctamente.

Para aplicar la prueba no paramétrica Friedman en el programa R usamos el comando friedmanTest(MyData) el cual nos proporcionó como resultado:

"Friedman's chi-squared = 16.133, df = 4, p-value = 0.002845"

Se observa que el valor de Chi-cuadrado es alto y el valor de *p* está por debajo de 0.05 por tanto existen diferencias significativas entre al menos uno de los algoritmos comparados, pero no podemos saber cuál, por lo que se hace necesario aplicar una prueba *post hoc*. En este caso usamos la función postHocTest de la forma:

test <- postHocTest(MyData, test="friedman", correct="bergmann", use.rank=TRUE)

Aquí usamos a Bergman y Hommel una de las pruebas más potentes en estos casos que queremos comparar todos contra todos. El paquete, además, cuenta con una forma muy elegante para mostrar los resultados de esta prueba usando para esto la función: plotRanking(pvalues=test$corrected.pval, summary=test$summary, alpha=0.05)

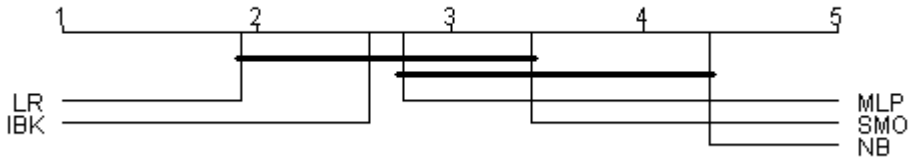

Figura 3.2: Gráfica obtenida al aplicar la función plotRanking, en la misma se puede ver las diferencias significativas entre los clasificadores.

En la Figura 3.2 se puede apreciar que no existen diferencias significativas entre los cuatro mejores algoritmos. Por su parte el clasificador Naive Bayes el cual obtuvo la última posición, no tiene diferencias estadísticamente significativas con sus dos vecinos más cercanos.

3.2 Resultados de clasificación de dendritas y axones

En esta sección se muestran los resultados obtenidos separando los rasgos de las dendritas y los axones. Esto nos ayudará, junto con otros argumentos, a llegar a conclusiones al final del capítulo. Como los conjuntos de rasgos son grandes debido a la suma de los obtenidos por cada programa, se debe aplicar métodos de selección de rasgos antes de aplicar los algoritmos de clasificación. Las técnicas de selección de atributos utilizadas son 6 y se mostraron en la Tabla 2.2.

Tabla 3.4 Resultados de los clasificadores en los conjuntos de dendritas y axones según el porcentaje de clasificación correcta.

NB		LR		MLP		SMO		IBK	
Axón	**Dend**	**Axón**	**Dend**	**Axón**	**Dend**	**Axón**	**Dend**	**Axón**	**Dend**
75.85	79.31	82.14	88.90	80.82	87.96	81.54	88.33	79.75	84.78
74.78	80.35	81.16	86.16	78.18	83.43	76.92	84.59	77.14	84.03
75.57	80.19	81.01	85.66	79.06	84.15	77.96	84.69	78.43	84.12
78.14	84.40	81.16	82.39	77.52	86.48	75.57	82.99	72.17	84.12
72.04	83.81	76.32	81.73	76.86	79.12	77.01	78.55	75.69	79.97
80.16	82.58	82.58	90.50	83.08	89.53	82.17	89.03	79.69	88.71

Tabla 3.5 Resultados de los clasificadores en los conjuntos de dendritas y axones según el AUC.

NB		LR		MLP		SMO		IBK	
Axón	**Dend**	**Axón**	**Dend**	**Axón**	**Dend**	**Axón**	**Dend**	**Axón**	**Dend**
0.86	0.85	0.88	0.96	0.88	0.95	0.82	0.86	0.87	0.91
0.88	0.87	0.88	0.92	0.88	0.91	0.81	0.82	0.89	0.92
0.87	0.87	0.87	0.91	0.86	0.90	0.77	0.82	0.86	0.91
0.82	0.88	0.87	0.82	0.87	0.89	0.77	0.80	0.87	0.88
0.76	0.87	0.83	0.87	0.84	0.85	0.77	0.75	0.72	0.88
0.87	0.87	0.88	0.96	0.84	0.96	0.75	0.87	0.73	0.93

En las Tablas 3.4 y 3.5 se pueden apreciar los resultados de cada clasificador aplicado 6 veces a cada conjunto de rasgos, uno por cada método de selección de atributos. Tanto con el índice de clasificación correcta, como con el AUC se puede observar que en la inmensa mayoría de los casos los resultados obtenidos con las dendritas fueron mejores que los obtenidos con los

axones, lo cual también puede verse en los gráficos de cajas de la Figura 3.3. Esto quiere decir que las interneuronas y las neuronas piramidales se diferencian más en las ramificaciones de dendritas que en los axones. Este resultado fue el esperado ya que aunque los axones son comúnmente más largos que las dendritas, las estructuras de las últimas son morfológicamente más complejas y esto trae consigo una mayor divesidad entre los rasgos obtenidos de los dos grupos neuronales.

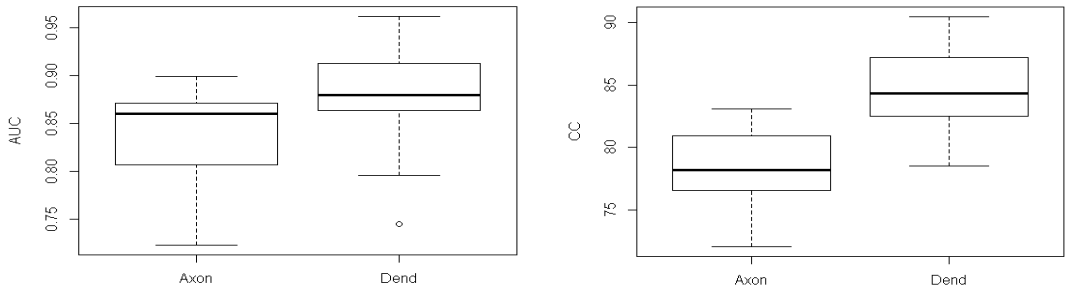

Figura 3.3: Gráficos de cajas donde se observa el comportamiento de la clasificación con rasgos de axones y de dendritas según el AUC en la izquierda y el CC en la derecha.

3.3 Resultados de la clasificación con rasgos de Neurolucida, Lmeasure-Farsight-TreesToolbox y la unión de ambos

En esta sección se muestran y analizan los resultados de la prueba principal de la investigación. Ello consiste en la comparación de los resultados obtenidos con los rasgos extraídos con el software comercial Neurolucida y los rasgos extraídos con las herramientas libres Lmeasure-Farsight-TreesToolbox, además de la unión de ambos conjuntos.

Después de tener los tres conjuntos de rasgos se aplicaron 6 estrategias de selección de atributos y a cada subconjunto se le aplicaron los 5 algoritmos de clasificación. Los valores mostrados corresponden al promedio después de repetir 10 veces la clasificación con validación cruzada de 10. En las tablas mostradas a continuación solo muestran los porcentajes de clasificación correcta, los resultados con el resto de los índices pueden consultarse en el Anexo 7.

Tabla 3.6 Resultados de la clasificación con rasgos de Neurolucida según el porcentaje de clasificación correcta.

Método de Evaluación	Método de Búsqueda	NB	LR	MLP	SMO	IBK

SVMAttributeEval	Ranker	82.76	88.89	87.51	86.72	85.37
CfsSubsetEval	GreedyStepwise-Forward	84.71	83.93	84.71	81.98	84.30
	Genetic	82.70	87.70	88.86	85.22	82.95
WrapperSubsetEval	GreedyStepwise-Forward	83.42	81.10	86.94	83.01	86.66
	GreedyStepwise-Backward	76.38	76.25	75.47	75.31	81.44
	Genetic	88.08	89.68	87.51	87.98	85.72

La Tabla 3.6 muestra los resultados de la clasificación en los rasgos morfológicos extraídos por el programa Neurolucida, en la misma se puede observar que en la mayoría de los casos el porcentaje de clasificación estuvo entre 80-89%, en unos pocos casos por debajo de 80% y en ninguna ocasión por encima de 90%.

Tabla 3.7 Resultados de la clasificación con rasgos de Lmeasure-Farsight-TreesToolbox según el porcentaje de clasificación correcta.

Método de Evaluación	**Método de Búsqueda**	**NB**	**LR**	**MLP**	**SMO**	**IBK**
SVMAttributeEval	Ranker	84.81	91.00	88.99	90.44	84.74
CfsSubsetEval	GreedyStepwise-Forward	82.86	84.62	84.84	86.66	86.28
	Genetic	82.92	85.72	83.42	85.15	85.47
WrapperSubsetEval	GreedyStepwise-Forward	85.47	88.58	85.25	86.72	85.44
	GreedyStepwise-Backward	72.89	87.01	84.96	82.86	81.94
	Genetic	86.25	90.59	90.37	91.10	86.47

En la Tabla 3.7 se muestran los resultados de la clasificación efectuada sobre los rasgos extraídos de los programas L_measure, Farsight y TreesToolbox. Aunque en la mayoría de los casos el porcentaje se mantuvo entre 80-90%, en varias ocasiones se logró alcanzar un 90%.

Tabla 3.8 Resultados de la clasificación con la unión de los rasgos de Neurolucida y Lmeasure-Farsight-TreesToolbox según el porcentaje de clasificación correcta.

Método de Evaluación	**Método de Búsqueda**	**NB**	**LR**	**MLP**	**SMO**	**IBK**
SVMAttributeEval	Ranker	86.41	92.57	92.67	93.45	90.06
CfsSubsetEval	GreedyStepwise-Forward	83.58	87.73	85.66	85.56	85.62
	Genetic	83.77	88.55	86.85	85.47	87.54
WrapperSubsetEval	GreedyStepwise-Forward	91.25	90.75	88.99	83.52	88.83
	GreedyStepwise-Backward	75.88	85.97	81.47	83.01	78.93
	Genetic	90.53	93.55	93.14	94.52	90.28

En la última clasificación efectuada con la unión de los dos conjuntos de rasgos anteriores la cual se muestra en la Tabla 3.8 se puede observar mejores resultados ya que en esta ocasión fueron más numerosos los porcentajes por encima de 90% llegando hasta un 94% de efectividad.

3.3.1 Resultados de mejores combinaciones de clasificador y selector de atributos

En esta sección se mostrará un listado con los mejores resultados obtenidos, analizando cuáles conjuntos de rasgos, clasificadores y estrategias de selección de atributos fueron los responsable de los más altos índices de instancias clasificadas correctamente.

Tabla 3.9 Listado de las combinaciones con más alto por ciento de instancias clasificadas correctamente.

Conjunto Rasgos	Clasificador	Método de avaluación	Método de búsqueda	% CC	# Rasgos
Lucid_LmeaFarsTrees	SMO	WrapperSubsetEval	Genetic	94.52	23
Lucid_LmeaFarsTrees	LR	WrapperSubsetEval	Genetic	93.55	15
Lucid_LmeaFarsTrees	SMO	SVMAttributeEval	Ranker	93.45	24
Lucid_LmeaFarsTrees	MLP	WrapperSubsetEval	Genetic	93.14	26
Lucid_LmeaFarsTrees	MLP	SVMAttributeEval	Ranker	92.67	24
Lucid_LmeaFarsTrees	LR	SVMAttributeEval	Ranker	92.57	24
Lucid_LmeaFarsTrees	NB	WrapperSubsetEval	GreedyStepwise-Forward	91.25	12
LmeaFarsTrees	SMO	WrapperSubsetEval	Genetic	91.10	25
LmeaFarsTrees	LR	SVMAttributeEval	Ranker	91.00	25
Lucid_LmeaFarsTrees	LR	WrapperSubsetEval	GreedyStepwise-Forward	90.75	10
LmeaFarsTrees	LR	WrapperSubsetEval	Genetic	90.59	23
Lucid_LmeaFarsTrees	NB	WrapperSubsetEval	Genetic	90.53	21
LmeaFarsTrees	SMO	SVMAttributeEval	Ranker	90.44	25
LmeaFarsTrees	MLP	WrapperSubsetEval	Genetic	90.37	27
Lucid_LmeaFarsTrees	IBK	WrapperSubsetEval	Genetic	90.28	15
Lucid_LmeaFarsTrees	IBK	SVMAttributeEval	Ranker	90.06	24

Neurolucida	LR	WrapperSubsetEval	Genetic	89.68	16

Para confeccionar el listado mostrado en la Tabla 3.9 se tomaron todas las combinaciones que obtuvieron clasificaciones con menos de 5% de diferencia con respecto al mejor valor alcanzado. Con esta condición fueron 17 las combinaciones clasificadas. En este listado se puede observar que los mejores resultados pertenecen a la unión de Neurolucida y Lmeasure-Farsight-TreesToolbox logrando varias clasificaciones por encima de 91%. Esto fue posible debido a la obtención de nuevos rasgos usando los softwares libres Farsight, TreesToolbox y L_measure (este último especializado en la extracción de rasgos morfológicos), que junto a los extraídos con el software comercial Neurolucida formaron un nuevo conjunto que permitió mejorar la eficacia de los clasificadores. A estos resultados también contribuyó de manera significativa la extracción de rasgos separando la neurona en dendritas y axones, debido a que los rasgos que más diferenciaron estas estructuras entre los dos grupos neuronales no fueron los mismos.

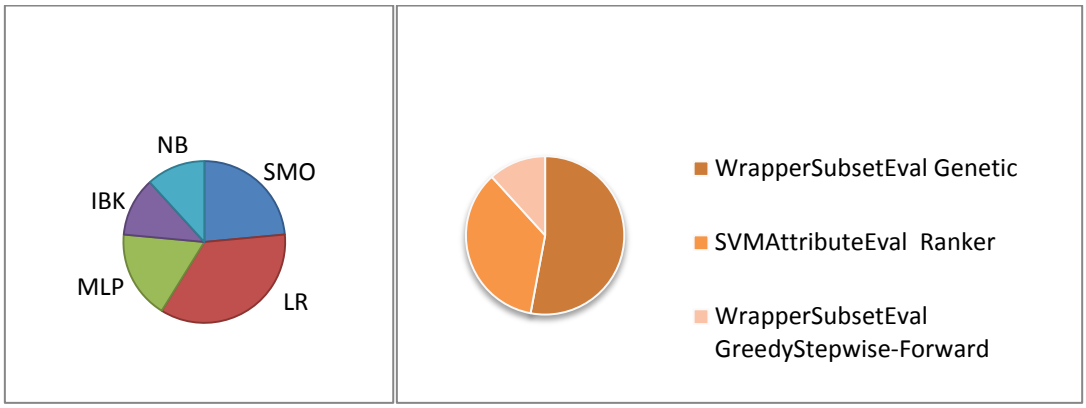

Figura 3.4: Gráficas mostrando la representación de los clasificadores y las estrategias de selección de atributos en las combinaciones que lograron los mejores resultados.

Como se puede apreciar en los gráficos de pastel mostrados en la Figura 3.4 de las 6 estrategias de selección de atributos usadas en la investigación solo 3 se vieron representadas en la Tabla 3.9 y de estas las mejores fueron las combinaciones del evaluador WrapperSubsetEval con el método de búsqueda Genetic y el evaluador SVMAttributeEval con el método de búsqueda Ranker. En cuanto a los algoritmos de clasificación todos tuvieron al menos 2 clasificaciones por encima de los 90%, pero los tres con mejores resultados fueron el SMO, LR y MLP sin poder afirmar que uno de ellos fue el ideal. Esto quiere decir que la eficacia en la clasificación no dependió de algún algoritmo en particular sino de los datos. En

cuanto a la selección de rasgos se observó que el método WrapperSubsetEval fue el más usado, porque este utiliza un algoritmo de clasificación para evaluar el conjunto de datos lo cual lo hace eficaz, como desventaja presenta un alto costo computacional. El otro método que mostró buenos resultados fue el SVMAttributeEval, el cual también hace uso de un clasificador para evaluar los atributos.

3.3.2 Análisis estadístico para determinar el mejor conjunto de rasgos usando CC

Para mostrar de manera gráfica el resultado de los clasificadores en los tres conjuntos de rasgos se usó el diagrama de cajas de la Figura 3.5, en el mismo se puede apreciar que los mejores resultados los obtuvo la unión de ambos conjuntos, seguido de Lmeasure-Farsight-TreesToolbox y terminando con el Neurolucida. Pero para conocer si las diferencias son significativas se debe aplicar alguna prueba estadística.

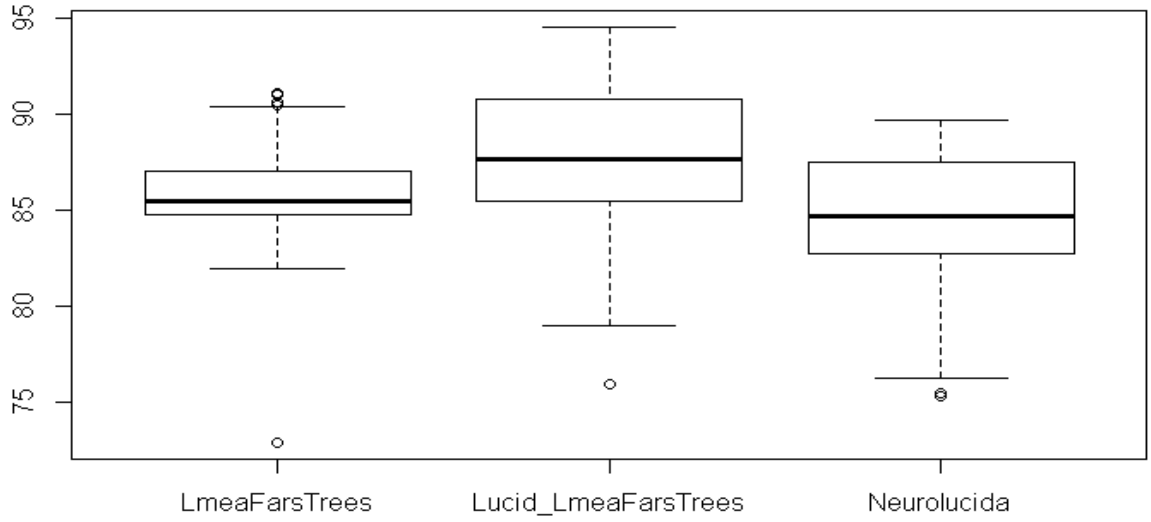

Figura 3.5: Gráficos de cajas donde se observa el comportamiento de la clasificación en los tres conjuntos de rasgos según el porcentaje de clasificación correcta.

Para realizar el análisis estadístico se conformó una tabla semejante a la utilizada en el epígrafe 3.1.1, la cual tiene por un eje los 3 grupos de rasgos utilizados y por el otro los diferentes algoritmos de clasificación utilizados. Como en esta ocasión lo que deseamos comparar son las bases de datos en lugar de los algoritmos y no podemos suponer normalidad debemos utilizar la prueba no paramétrica de Kruscal-Wallis. Para implementar la misma se usó función kruskal.test(data) del paquete estadístico R, la cual dio como resultado:

"Kruskal-Wallis chi-squared = 9.8386, df = 2, p-value = 0.007304"

Como se puede apreciar el valor de *p* es menor que 0.05, podemos decir que existen diferencias estadísticamente significativas entre al menos 2 de los grupos pero no podemos saber cuáles. Por lo tanto debemos realizar un análisis *post-hoc*, en esta ocasión se aplicará la prueba de Mann–Whitney. Al realizar esta prueba se debe aplicar el ajuste de Bonferroni para evitar el aumento de error de tipo I. La función con el método de ajuste queda de la siguiente manera:

```
pairwise.wilcox.test(data,p.adjust.method="bonferroni")
```

Tabla 3.10 Resultado de la prueba de Mann–Whitney con la corrección de bonferroni.

	LmeaFarsTrees	Lucid_LmeaFarsTrees
Lucid_LmeaFarsTrees	0.1753	-
Neurolucida	0.4061	0.0089

Como se puede apreciar en los valores de *p* mostrados en la Tabla 3.10 solo existen diferencias estadísticamente significativas entre los rasgos obtenidos con el Neurolucida y la unión de estos con Lmeasure-Farsight-TreesToolbox.

3.3.3 Análisis estadístico para determinar el mejor conjunto de rasgos usando AUC

El análisis estadístico realizado en esta sección será semejante al realizado en el epígrafe anterior pero en esta ocasión se usará en lugar del porcentaje de clasificación correcta, el índice de desempeño AUC, esto nos ayudará a validar los resultados obtenidos. En la Figura 3.6 se puede observar mediante un gráfico de cajas el comportamiento de los tres conjuntos de rasgos según al AUC, en la misma se aprecia un comportamiento similar al mostrado en la Figura 3.6.

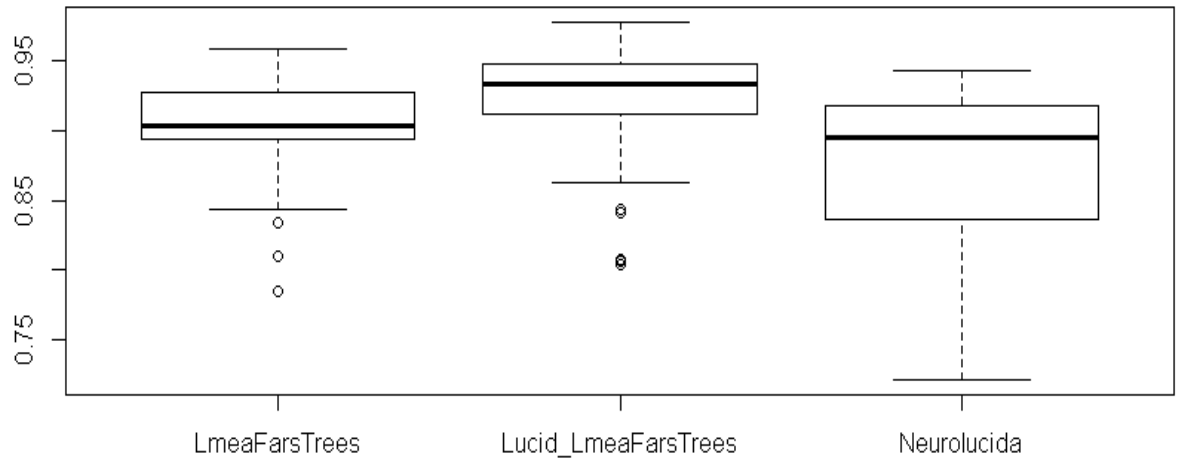

Figura 3.6: Gráficos de cajas donde se observa el comportamiento de la clasificación en los tres conjuntos de rasgos según el AUC.

Para comprobar si existen diferencias estadísticamente significativas entre algunos de los grupos debemos aplicar, al igual que en el caso anterior, la prueba no paramétrica de Kruscal-Wallis. Para implementar la misma se usó función kruskal.test(data) del paquete estadístico R, la cual proporcionó como resultado:

"Kruskal-Wallis chi-squared = 11.647, df = 2, p-value = 0.002958"

El valor de *p* nuevamente estuvo por debajo de 0.05 lo que implica diferencias estadísticamente significativas entre algunos de los grupos. Procedemos entonces a realizar un análisis *post-hoc*, usando la prueba Mann–Whitney con la corrección de Bonferroni para evitar el aumento de error de tipo I. Los resultados pueden verse en la Tabla 3.11

Tabla 3.11 Resultado de la prueba de Mann–Whitney con la corrección de Bonferroni.

	LmeaFarsTrees	Lucid_LmeaFarsTrees
Lucid_LmeaFarsTrees	0.0645	-
Neurolucida	0.5355	0.0041

Como se puede apreciar en los valores de *p* solo existen diferencias estadísticamente significativas entre los rasgos obtenidos con el Neurolucida y la unión de estos con Lmeasure-Farsight-TreesToolbox. Este resultado concuerda con el obtenido en el epígrafe anterior, lo que ratifica su veracidad.

3.4 Mejores rasgos morfológicos usados en la clasificación

En esta sección se analizan los rasgos más usados por las combinaciones que lograron los mejores resultados en la clasificación mostrados en el epígrafe 3.3.1. Para conformar la Tabla 3.12 se buscaron los rasgos de los 17 mejores subconjuntos y se tabularon los que se repitieron 4 o más veces, incluyendo el programa con que fueron obtenidos y a la sección de la neurona a la cual pertenecen.

Tabla 3.12 Rasgos más usados en las clasificación con mejores resultados.

Rasgos	**Cant**	**Programa**	**Región**
dimensión_fractal	11	L_measure	dend
taper_2 de Dendritas_Avg	9	L_measure	dend
num_secciones_sholl_dendritas	9	Neurolucida	dend
alto de neurona_avg	7	L_measure	neuron
superficie de neurona_avg	7	L_measure	neuron
Perímetro_env_ convexa_dendritas	6	Neurolucida	dend
intersecc_Sholl_dendritas_en_100	6	TreesToolbox	dend
intersecc_Sholl_neurona_en_800	6	TreesToolbox	neuron
ampl_bif_local_dendritas_avg	5	L_measure	dend
HillmanThreshold_neurona_avg	5	L_measure	neuron
Diámetro_última_bif_Dendritas_ Stdv	5	L_measure	dend
Razón_diámetro_hijos_padre_Dendritas_ Stdv	5	L_measure	dend
Distancia_euclidiana_rama_a_soma_Dendritas_Ave	5	Farsight	dend
Ampl_bif_remota_Axón_Stdv	5	Lmeasure	Axón
Distancia_euclidiana_rama_a_soma_Dendritas_Max	5	Farsight	dend
intersecc_Sholl_neurona_en_450	5	TreesToolbox	neuron
Orden_de_rama_dendritas_stdv	4	L_measure	dend
Razón_hijas_dendritas_avg	4	L_measure	dend
Pk de neurona_avg	4	L_measure	neuron
Pk_clásico_de_dendritas_stdv	4	L_measure	dend
Elevación_neurona_ave	4	Farsight	neuron
longitud_Sholl_dendritas_en_150	4	Neurolucida	dend
segmento_de_más_alto_orden_dendrita	4	Neurolucida	dend
Max_nivel_hoja_Axón	4	Farsight	Axón
Max_nivel_hoja_Dendritas	4	Farsight	dend
Magnitud_total_de_terminales_Neurona	4	Farsight	neuron

Factor_forma_soma	4	Neurolucida	neuron
Grado_terminal_Axón_stdv	4	L_measure	Axón
Total_segmento_terminales_Axón	4	L_measure	Axón

Como se puede apreciar en la Tabla 3.12 cada uno de los programas usados en la investigación están representados al igual que las tres regiones de la neurona. Los tres rasgos más repetidos fueron la dimensión fractal, la métrica Taper_2 y el número de secciones de Sholl, todos estos del árbol de dendritas.

Tabla 3.13 Cantidades de rasgos que pertenecen a cada programa y a cada región de la neurona.

Programa	**Cantidad**	**Región**	**Cantidad**
L_measure	15	Dendrita	16
Farsight	6	Axón	4
Neurolucida	5	Neurona	9
TreesToolbox	3		

Como se puede apreciar en la Tabla 3.13 el programa con mejores resultados fue L_measure ya que más de la mitad de los rasgos fueron suyos, incluyendo los dos primeros. Estos resultados mostraron las potencialidades de este programa especialmente diseñado para extraer los rasgos morfológicos de las estructuras neuronales. La región de la neurona de la cual se obtuvieron la mayor cantidad de rasgos fue las dendritas, con más de la mitad del total. Esto ocurrió debido a la complejidad morfológica de estas estructuras y a su diversidad entre los diferentes tipos de neuronas.

3.5 Conclusiones del Capítulo

En el presente capítulo se expusieron los resultados de la investigación. Primeramente se evaluaron los algoritmos de clasificación en múltiples conjuntos de rasgos morfológicos dando como resultados que no obtuvieron diferencias estadísticamente significativas entre los primeros cuatro, solo el Naive Bayes que ocupó la última posición tuvo diferencias con los dos primeros. Posteriormente se realizó la clasificación automática separando los rasgos provenientes de los axones y de las dendritas, donde los últimos mostraron ser superiores. La última prueba realizada fue la comparación de los resultados obtenidos a partir de los rasgos morfológicos extraídos con Neurolucida, los extraídos con Lmeasure-Farsight-TreesToolbox y

la unión de ambos grupos, donde los últimos mostraron los mejores resultados con diferencia estadísticamente significativa respecto a los obtenidos con Neurolucida. Por último se realizó un análisis de las clasificaciones más elevadas donde se pudo apreciar que los algoritmos con mejor desempeño fueron el SMO, MLP y LR; las estrategias de selección de atributos más eficaces fueron las combinaciones del evaluador WrapperSubsetEval con el método de búsqueda Genetic y el evaluador SVMAttributeEval con el método de búsqueda Ranker; además se comprobó que los mejores rasgos provenían de los árboles de dendritas.

CONCLUSIONES

- Mediante la comparación de los resultados de la clasificación de tres conjuntos de rasgos extraídos con Neurolucida, Lmeasure-Farsight-TreesToolbox y la unión de ambos conjuntos, estos últimos mostraron los mejores resultados alcanzando hasta un 94.52% de efectividad.

- Al realizar una evaluación de los algoritmos más apropiados para la clasificación de rasgos morfológicos de las neuronas reconstruidas usadas sobresalieron los clasificadores, máquina de soporte vectorial, perceptrón multicapa y regresión logística multinomial, quedando algo rezagado el Naive Bayes.

- Después de realizar una búsqueda, se comprobó que las estrategias de selección de atributos más eficaces para reducir la dimensionalidad de los rasgos utilizados, fueron las combinaciones del evaluador WrapperSubsetEval con el método de búsqueda Genetic y el evaluador SVMAttributeEval con el método de búsqueda Ranker.

- Al analizar los rasgos más representativos de la morfología de la neurona se obtuvo como resultado que los obtenidos del árbol de dendritas diferenciaron más los dos grupos neuronales que los obtenidos de los axones.

RECOMENDACIONES

- Aplicar el mismo procedimiento en otras bases de datos de neuronas reconstruidas para evaluar los resultados obtenidos en este estudio.

- Explotar con más profundidad las potencialidades de las herramientas TreesToolbox y L_measure, además de probar otras herramientas que permitan la extracción de rasgos morfológicos.

- Modificar los algoritmos implementados en esta investigación o utilizar otros que logren mejorar los resultados de la clasificación de estas clases neuronales.

BIBLIOGRAFÍA

[1] T. R. Insel, S. C. Landis, and F. S. Collins, “The NIH BRAIN Initiative,” *Science*, vol. 340, no. 6133, pp. 687–688, May 2013.

[2] H. Markram, “The human brain project,” *Sci. Am.*, vol. 306, no. 6, pp. 50–55, Jun. 2012.

[3] R. Parekh and G. A. Ascoli, “Neuronal morphology goes digital: a research hub for cellular and system neuroscience,” *Neuron*, vol. 77, no. 6, pp. 1017–1038, Mar. 2013.

[4] E. Meijering, “Neuron tracing in perspective,” *Cytometry A*, vol. 77A, no. 7, pp. 693–704, Jul. 2010.

[5] R. Armañanzas and G. A. Ascoli, “Towards the automatic classification of neurons,” *Trends Neurosci.*, vol. 38, no. 5, pp. 307–318, May 2015.

[6] H. Lodish and A. Berk, *Molecular Cell Biology*, 7 edition. W. H. Freeman, 2012.

[7] E. R. Kandel, J. H. Schwartz, T. M. Jessell, S. A. Siegelbaum, and A. J. Hudspeth, Eds., *Principles of Neural Science, Fifth Edition*, 5th edition. New York: McGraw-Hill Education / Medical, 2012.

[8] del Abril Alonso, A., Ambrosio Flores, E., de Blas Calleja, M. R., Caminero Gómez, Á., García Lecumberri, C., & de Pablo González, J. M., *Fundamentos de psicobiología*. Madrid: Sanz y Torres, 2009.

[9] M. F. Bear, B. W. Connors, and M. A. Paradiso, *Neuroscience: Exploring the Brain, 3rd Edition*, 3rd edition. Philadelphia, PA: Lippincott Williams and Wilkins, 2006.

[10] M. Halavi, K. A. Hamilton, R. Parekh, and G. A. Ascoli, “Digital Reconstructions of Neuronal Morphology: Three Decades of Research Trends,” *Front. Neurosci.*, vol. 6, Apr. 2012.

[11] E. De Schutter, G. A. Ascoli, and D. N. Kennedy, “Review of papers describing neuroinformatics software,” *Neuroinformatics*, vol. 7, no. 4, pp. 211–212, Dec. 2009.

[12] J. R. Glaser and E. M. Glaser, “Neuron imaging with Neurolucida--a PC-based system for image combining microscopy,” *Comput. Med. Imaging Graph. Off. J. Comput. Med. Imaging Soc.*, vol. 14, no. 5, pp. 307–317, Oct. 1990.

[13] P. Vallotton, R. Lagerstrom, C. Sun, M. Buckley, D. Wang, M. De Silva, et al., “Automated analysis of neurite branching in cultured cortical neurons using HCA-Vision,” *Cytom. Part J. Int. Soc. Anal. Cytol.*, vol. 71, no. 10, pp. 889–895, Oct. 2007.

[14] J. J. Capowski, “Computer-aided reconstruction of neuron trees from several serial sections,” *Comput. Biomed. Res.*, vol. 10, no. 6, pp. 617–629, Diciembre 1977.

[15] J. C. Fiala, “Reconstruct: a free editor for serial section microscopy,” *J. Microsc.*, vol. 218, no. Pt 1, pp. 52–61, Apr. 2005.

[16] D. R. Myatt and S. J. Nasuto, “Improved automatic midline tracing of neurites with Neuromantic,” *BMC Neurosci.*, vol. 9, no. 1, pp. 1–2, 2008.

[17] M. L. Narro, F. Yang, R. Kraft, C. Wenk, A. Efrat, and L. L. Restifo, “NeuronMetrics: software for semi-automated processing of cultured neuron images,” *Brain Res.*, vol. 1138, pp. 57–75, Mar. 2007.

[18] M. Pool, J. Thiemann, A. Bar-Or, and A. E. Fournier, “NeuriteTracer: a novel ImageJ plugin for automated quantification of neurite outgrowth,” *J. Neurosci. Methods*, vol. 168, no. 1, pp. 134–139, Feb. 2008.

[19] W. Yu, H. K. Lee, S. Hariharan, W. Bu, and S. Ahmed, “Quantitative neurite outgrowth measurement based on image segmentation with topological dependence,” *Cytom. Part J. Int. Soc. Anal. Cytol.*, vol. 75, no. 4, pp. 289–297, Apr. 2009.

[20] S. L. Wearne, A. Rodriguez, D. B. Ehlenberger, A. B. Rocher, S. C. Henderson, and P. R. Hof, “New techniques for imaging, digitization and analysis of three-dimensional neural morphology on multiple scales,” *Neuroscience*, vol. 136, no. 3, pp. 661–680, 2005.

[21] X. Xu and S. T. C. Wong, “Optical microscopic image processing of dendritic spines morphology,” *IEEE Signal Process. Mag.*, vol. 23, no. 4, pp. 132–135, Jul. 2006.

[22] B. E. Losavio, Y. Liang, A. Santamaría-Pang, I. A. Kakadiaris, C. M. Colbert, and P. Saggau, “Live neuron morphology automatically reconstructed from multiphoton and confocal imaging data,” *J. Neurophysiol.*, vol. 100, no. 4, pp. 2422–2429, Oct. 2008.

[23] H. Peng, Z. Ruan, F. Long, J. H. Simpson, and E. W. Myers, “V3D enables real-time 3D visualization and quantitative analysis of large-scale biological image data sets,” *Nat. Biotechnol.*, vol. 28, no. 4, pp. 348–353, Apr. 2010.

[24] H. Cuntz, F. Forstner, A. Borst, and M. Häusser, “The TREES toolbox--probing the basis of axonal and dendritic branching,” *Neuroinformatics*, vol. 9, no. 1, pp. 91–96, Mar. 2011.

[25] J. Luisi, A. Narayanaswamy, Z. Galbreath, and B. Roysam, “The FARSIGHT trace editor: an open source tool for 3-D inspection and efficient pattern analysis aided editing of automated neuronal reconstructions,” *Neuroinformatics*, vol. 9, no. 2–3, pp. 305–315, Sep. 2011.

[26] R. Scorcioni, S. Polavaram, and G. A. Ascoli, “L-Measure: a web-accessible tool for the analysis, comparison and search of digital reconstructions of neuronal morphologies,” *Nat. Protoc.*, vol. 3, no. 5, pp. 866–876, 2008.

[27] J. Popko, A. Fernandes, D. Brites, and L. M. Lanier, “Automated analysis of NeuronJ tracing data,” *Cytom. Part J. Int. Soc. Anal. Cytol.*, vol. 75, no. 4, pp. 371–376, Apr. 2009.

[28] G. A. Ascoli and J. L. Krichmar, “L-neuron: A modeling tool for the efficient generation and parsimonious description of dendritic morphology,” *Neurocomputing*, vol. 32–33, pp. 1003–1011, Jun. 2000.

[29] P. Gleeson, V. Steuber, and R. A. Silver, “neuroConstruct: a tool for modeling networks of neurons in 3D space,” *Neuron*, vol. 54, no. 2, pp. 219–235, Apr. 2007.

[30] J. M. Bower and D. Beeman, *The Book of GENESIS*. New York, NY: Springer New York, 1998.

[31] M. L. Hines and N. T. Carnevale, “NEURON: a tool for neuroscientists,” *Neurosci. Rev. J. Bringing Neurobiol. Neurol. Psychiatry*, vol. 7, no. 2, pp. 123–135, Apr. 2001.

[32] R. Brette, M. Rudolph, T. Carnevale, M. Hines, D. Beeman, J. M. Bower, et al., “Simulation of networks of spiking neurons: a review of tools and strategies,” *J. Comput. Neurosci.*, vol. 23, no. 3, pp. 349–398, Dec. 2007.

[33] G. A. Ascoli, J. L. Krichmar, S. J. Nasuto, and S. L. Senft, “Generation, description and storage of dendritic morphology data.,” *Philos. Trans. R. Soc. Lond. Ser. B*, vol. 356, no. 1412, pp. 1131–1145, Aug. 2001.

[34] R. C. Cannon, D. A. Turner, G. K. Pyapali, and H. V. Wheal, “An on-line archive of reconstructed hippocampal neurons,” *J. Neurosci. Methods*, vol. 84, no. 1–2, pp. 49–54, Oct. 1998.

[35] G. A. Ascoli, “Successes and rewards in sharing digital reconstructions of neuronal morphology,” *Neuroinformatics*, vol. 5, no. 3, pp. 154–160, 2007.

[36] D. Gardner, H. Akil, G. A. Ascoli, D. M. Bowden, W. Bug, D. E. Donohue, et al., “The Neuroscience Information Framework: A Data and Knowledge Environment for Neuroscience,” *Neuroinformatics*, vol. 6, no. 3, pp. 149–160, Sep. 2008.

[37] G. A. Ascoli, D. E. Donohue, and M. Halavi, “NeuroMorpho.Org: A Central Resource for Neuronal Morphologies,” *J. Neurosci.*, vol. 27, no. 35, pp. 9247–9251, Aug. 2007.

[38] M. E. Martone, “Cell Centered Database,” in *Encyclopedia of Computational Neuroscience*, D. Jaeger and R. Jung, Eds. New York, NY: Springer New York, 2015, pp. 574–577.

[39] A.-S. Chiang, C.-Y. Lin, C.-C. Chuang, H.-M. Chang, C.-H. Hsieh, C.-W. Yeh, et al., “Three-dimensional reconstruction of brain-wide wiring networks in Drosophila at single-cell resolution,” *Curr. Biol. CB*, vol. 21, no. 1, pp. 1–11, Jan. 2011.

[40] S. Ramón y Cajal, “El nuevo concepto de la histología de los centros nerviosos.,” *Rev. Cienc. Med*, no. 15, pp. 457–476, 1892.

[41] Petilla Interneuron Nomenclature Group, G. A. Ascoli, L. Alonso-Nanclares, S. A. Anderson, G. Barrionuevo, R. Benavides-Piccione, et al., “Petilla terminology: nomenclature of features of GABAergic interneurons of the cerebral cortex,” *Nat. Rev. Neurosci.*, vol. 9, no. 7, pp. 557–568, Jul. 2008.

[42] P. Larrañaga, B. Calvo, R. Santana, C. Bielza, J. Galdiano, I. Inza, et al., “Machine learning in bioinformatics,” *Brief. Bioinform.*, vol. 7, no. 1, pp. 86–112, Mar. 2006.

[43] J. Stone, *Parallel processing in the visual system: the classification of retinal ganglion cells and its impact on the neurobiology of vision*. Plenum Press, 1983.

[44] A. Alavi, B. Cavanagh, G. Tuxworth, A. Meedeniya, A. Mackay-Sim, and M. Blumenstein, “Automated classification of dopaminergic neurons in the rodent brain,” presented at the International Joint Conference on Neural Networks, 2009. IJCNN 2009, 2009, pp. 81–88.

[45] R. Parekh and G. A. Ascoli, “Quantitative investigations of axonal and dendritic arbors: development, structure, function, and pathology,” *Neurosci. Rev. J. Bringing Neurobiol. Neurol. Psychiatry*, vol. 21, no. 3, pp. 241–254, Jun. 2015.

[46] R. M. Cesar Júnior and L. da F. Costa, “Neural cell classification by wavelets and multiscale curvature,” *Biol. Cybern.*, vol. 79, no. 4, pp. 347–360, Oct. 1998.

[47] M. Bota and L. W. Swanson, “BAMS Neuroanatomical Ontology: Design and Implementation,” *Front. Neuroinformatics*, vol. 2, May 2008.

[48] A. Karagiannis, T. Gallopin, C. Dávid, D. Battaglia, H. Geoffroy, J. Rossier, et al., “Classification of NPY-Expressing Neocortical Interneurons,” *J. Neurosci.*, vol. 29, no. 11, pp. 3642–3659, Mar. 2009.

[49] L. Guerra, L. M. McGarry, V. Robles, C. Bielza, P. Larrañaga, and R. Yuste, “Comparison between supervised and unsupervised classifications of neuronal cell types: a case study,” *Dev. Neurobiol.*, vol. 71, no. 1, pp. 71–82, Jan. 2011.

[50] J. DeFelipe, P. L. López-Cruz, R. Benavides-Piccione, C. Bielza, P. Larrañaga, S. Anderson, et al., “New insights into the classification and nomenclature of cortical GABAergic interneurons,” *Nat. Rev. Neurosci.*, vol. 14, no. 3, pp. 202–216, Mar. 2013.

[51] R. Santana, L. M. McGarry, C. Bielza, P. Larrañaga, and R. Yuste, “Classification of neocortical interneurons using affinity propagation,” *Front. Neural Circuits*, vol. 7, p. 185, 2013.

[52] P. L. López Cruz, P. Larrañaga Múgica, J. De Felipe Oroquieta, and M. C. Bielza Lozoya, “Bayesian network modeling of the consensus between experts: an application to neuron classification,” *Int. J. Approx. Reason.*, vol. 55, no. 1, pp. 3–22, 2014.

[53] D. A. Sholl, “Dendritic organization in the neurons of the visual and motor cortices of the cat,” *J. Anat.*, vol. 87, no. Pt 4, p. 387–406.1, Oct. 1953.

[54] B. Mihaljević, R. Benavides-Piccione, C. Bielza, J. DeFelipe, and P. Larrañaga, “Bayesian network classifiers for categorizing cortical GABAergic interneurons,” *Neuroinformatics*, vol. 13, no. 2, pp. 193–208, Apr. 2015.

[55] U. Sümbül, S. Song, K. McCulloch, M. Becker, B. Lin, J. R. Sanes, et al., “A genetic and computational approach to structurally classify neuronal types,” *Nat. Commun.*, vol. 5, p. 3512, Mar. 2014.

[56] Y. Lu, L. Carin, R. Coifman, W. Shain, and B. Roysam, “Quantitative arbor analytics: unsupervised harmonic co-clustering of populations of brain cell arbors based on L-measure,” *Neuroinformatics*, vol. 13, no. 1, pp. 47–63, Jan. 2015.

[57] S. Polavaram, T. A. Gillette, R. Parekh, and G. A. Ascoli, “Statistical analysis and data mining of digital reconstructions of dendritic morphologies,” *Front. Neuroanat.*, vol. 8, p. 138, 2014.

[58] K. Zawadzki, C. Feenders, M. P. Viana, M. Kaiser, and L. da F. Costa, “Morphological homogeneity of neurons: searching for outlier neuronal cells,” *Neuroinformatics*, vol. 10, no. 4, pp. 379–389, Oct. 2012.

[59] K. M. Brown, I. Sugihara, Y. Shinoda, and G. A. Ascoli, “Digital morphometry of rat cerebellar climbing fibers reveals distinct branch and bouton types,” *J. Neurosci. Off. J. Soc. Neurosci.*, vol. 32, no. 42, pp. 14670–14684, Oct. 2012.

[60] F. Sündermann, N. Golovyashkina, C. Tackenberg, R. Brandt, and L. Bakota, “High-resolution imaging and evaluation of spines in organotypic hippocampal slice cultures,” *Methods Mol. Biol. Clifton NJ*, vol. 846, pp. 277–293, 2012.

[61] David J. Marchette, “Investigation of a random graph model for neuronal connectivity,” *Jt. Mtg Am Math Soc*, p. 91, 2012.

[62] T. Zhao and S. M. Plaza, “Automatic Neuron Type Identification by Neurite Localization in the Drosophila Medulla,” *ArXiv14091892 Cs Q-Bio*, Sep. 2014.

[63] S. McGinnis and T. L. Madden, “BLAST: at the core of a powerful and diverse set of sequence analysis tools,” *Nucleic Acids Res.*, vol. 32, no. Web Server issue, pp. W20-25, Jul. 2004.

[64] N. Y. Masse, S. Cachero, A. D. Ostrovsky, and G. S. X. E. Jefferis, “A mutual information approach to automate identification of neuronal clusters in Drosophila brain images,” *Front. Neuroinformatics*, vol. 6, p. 21, 2012.

[65] B. J. Frey and D. Dueck, “Clustering by passing messages between data points,” *Science*, vol. 315, no. 5814, pp. 972–976, Feb. 2007.

[66] M. Costa, J. D. Manton, A. D. Ostrovsky, S. Prohaska, and G. S. X. E. Jefferis, “NBLAST: Rapid, Sensitive Comparison of Neuronal Structure and Construction of Neuron Family Databases,” *Neuron*, Jun. 2016.

[67] M. D. Kim, Y. Wen, and Y.-N. Jan, “Patterning and organization of motor neuron dendrites in the Drosophila larva,” *Dev. Biol.*, vol. 336, no. 2, pp. 213–221, Dec. 2009.

[68] G. Tsechpenakis, R. E. Gamage, M. D. Kim, and A. Chiba, “Motor neuron morphology estimation for its classification in the Drosophila brain,” *Conf. Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. IEEE Eng. Med. Biol. Soc. Annu. Conf.*, vol. 2011, pp. 7755–7758, 2011.

[69] X. Chang, M. D. Kim, R. Stephens, T. Qu, A. Chiba, and G. Tsechpenakis, “Part-based motor neuron recognition in the Drosophila ventral nerve cord,” *NeuroImage*, vol. 90, pp. 33–42, Apr. 2014.

[70] A. Razetti, X. Descombes, C. Medioni, and F. Besse, “Statistical Characterization, Modelling and Classification of Morphological Changes in imp Mutant Drosophila Gamma Neurons,” presented at the BIOSTEC 2016 - The 9h International Joint Conference on Biomedical Engineering Systems and Technologies, 2016.

[71] B. Mihaljević, R. Benavides-Piccione, L. Guerra, J. DeFelipe, P. Larrañaga, and C. Bielza, “Classifying GABAergic interneurons with semi-supervised projected model-based clustering,” *Artif. Intell. Med.*, vol. 65, no. 1, pp. 49–59, Sep. 2015.

[72] M. Salganicoff, M. Sarna, L. Sax, and G. L. Gerstein, “Unsupervised waveform classification for multi-neuron recordings: a real-time, software-based system. I. Algorithms and implementation,” *J. Neurosci. Methods*, vol. 25, no. 3, pp. 181–187, Oct. 1988.

[73] J. Y. Cohen, P. Pouget, R. P. Heitz, G. F. Woodman, and J. D. Schall, “Biophysical support for functionally distinct cell types in the frontal eye field,” *J. Neurophysiol.*, vol. 101, no. 2, pp. 912–916, Feb. 2009.

[74] S. Druckmann, S. Hill, F. Schürmann, H. Markram, and I. Segev, “A hierarchical structure of cortical interneuron electrical diversity revealed by automated statistical analysis,” *Cereb. Cortex N. Y. N 1991*, vol. 23, no. 12, pp. 2994–3006, Dec. 2013.

[75] A. V. Zaitsev, N. V. Povysheva, G. Gonzalez-Burgos, and D. A. Lewis, “Electrophysiological classes of layer 2/3 pyramidal cells in monkey prefrontal cortex,” *J. Neurophysiol.*, vol. 108, no. 2, pp. 595–609, Jul. 2012.

[76] J. Helm, G. Akgul, and L. P. Wollmuth, “Subgroups of parvalbumin-expressing interneurons in layers 2/3 of the visual cortex,” *J. Neurophysiol.*, vol. 109, no. 6, pp. 1600–1613, Mar. 2013.

[77] L. M. McGarry, A. M. Packer, E. Fino, V. Nikolenko, T. Sippy, and R. Yuste, “Quantitative classification of somatostatin-positive neocortical interneurons identifies three interneuron subtypes,” *Front. Neural Circuits*, vol. 4, p. 12, 2010.

[78] D. Battaglia, A. Karagiannis, T. Gallopin, H. W. Gutch, and B. Cauli, “Beyond the frontiers of neuronal types,” *Front. Neural Circuits*, vol. 7, p. 13, 2013.

[79] S. B. Nelson, K. Sugino, and C. M. Hempel, “The problem of neuronal cell types: a physiological genomics approach,” *Trends Neurosci.*, vol. 29, no. 6, pp. 339–345, Jun. 2006.

[80] K. Sugino, C. M. Hempel, M. N. Miller, A. M. Hattox, P. Shapiro, C. Wu, et al., “Molecular taxonomy of major neuronal classes in the adult mouse forebrain,” *Nat. Neurosci.*, vol. 9, no. 1, pp. 99–107, Jan. 2006.

[81] J. Tsang, J. Zhu, and A. van Oudenaarden, “MicroRNA-mediated feedback and feedforward loops are recurrent network motifs in mammals,” *Mol. Cell*, vol. 26, no. 5, pp. 753–767, Jun. 2007.

[82] K. D. Winden, M. C. Oldham, K. Mirnics, P. J. Ebert, C. H. Swan, P. Levitt, et al., “The organization of the transcriptional network in specific neuronal classes,” *Mol. Syst. Biol.*, vol. 5, p. 291, 2009.

[83] L. Tricoire, K. A. Pelkey, B. E. Erkkila, B. W. Jeffries, X. Yuan, and C. J. McBain, “A blueprint for the spatiotemporal origins of mouse hippocampal interneuron diversity,” *J. Neurosci. Off. J. Soc. Neurosci.*, vol. 31, no. 30, pp. 10948–10970, Jul. 2011.

[84] M. J. Hawrylycz, E. S. Lein, A. L. Guillozet-Bongaarts, E. H. Shen, L. Ng, J. A. Miller, et al., “An anatomically comprehensive atlas of the adult human brain transcriptome,” *Nature*, vol. 489, no. 7416, pp. 391–399, Setiembre 2012.

[85] S. A. Sorensen, A. Bernard, V. Menon, J. J. Royall, K. J. Glattfelder, T. Desta, et al., “Correlated gene expression and target specificity demonstrate excitatory projection neuron diversity,” *Cereb. Cortex N. Y. N 1991*, vol. 25, no. 2, pp. 433–449, Feb. 2015.

[86] R. R. Bouckaert, E. Frank, M. Hall, R. Kirkby, P. Reutemann, A. Seewald, et al., “WEKA Manual for Version 3-7-8,” *Hamilt. N. Z.*, 2013.

[87] D. Grossman and P. Domingos, “Learning Bayesian Network Classifiers by Maximizing Conditional Likelihood,” in *Proceedings of the Twenty-first International Conference on Machine Learning*, New York, NY, USA, 2004, p. 46–.

[88] S. Le Cessie and J. C. Van Houwelingen, “Ridge Estimators in Logistic Regression,” *J. R. Stat. Soc. Ser. C Appl. Stat.*, vol. 41, no. 1, pp. 191–201, 1992.

[89] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques, Third Edition*, 3 edition. Haryana, India; Burlington, MA: Morgan Kaufmann, 2011.

[90] J. Platt, “Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines,” *Microsoft Res.*, Apr. 1998.

[91] D. W. Aha, D. Kibler, and M. K. Albert, “Instance-based learning algorithms,” *Mach. Learn.*, vol. 6, no. 1, pp. 37–66.

[92] J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” *J. Mach. Learn. Res.*, vol. 7, no. Jan, pp. 1–30, 2006.

[93] B. Calvo and G. Santafe, “scmamp: Statistical Comparison of Multiple Algorithms in Multiple Problems,” *R J.*, 2015.

ANEXOS

Anexo 1. Conjunto de rasgos extraídos con Neurolucida

'axon node density'
'dendrite node density'
'somatic perimeter (?m)'
'somatic area (?m2)'
'axonal node total'
'total axonal length (?m)'
'total surface area of axon (?m2)'
'ratio of axonal length to surface area (1/?m)'
'number of dendrites'
'dendritic node total'
'total dendritic length (?m)'
'ave length of dendrites (?m)'
'total surface area of dendrites (?m2)'
'ratio of dendritic length to surface area (1/?m)'
'somatic aspect ratio'
'somatic compactness'
'somatic form factor'
'somatic roundness'
'highest order axon segment'
'highest order dendritic segment'
'dendritic torsion ratio '
'axonal torsion ratio '
'k-dim (fractal analysis)-axon'
'k-dim (fractal analysis)-dendrites'
'axonal planar angle average'
'axonal planar angle stdv'
'axonal local angle average'
'axonal local angle stdv'
axonal spline angle average'
'axonal spline angle stdv'
'dendritic planar angle average'
'dendritic planar angle stdv'
'dendritic local angle average'
'dendritic local angle stdv'
'dendritic spline angle average'
'dendritic spline angle stdv'
'ave tortuosity of axonal segments'
'stdv of tortuosity of axonal segments'
'axonal segment length average (?m)'
'axonal segment length average stdv (?m)'
'ave tortuosity of dendritic segments'
'stdv of tortuosity of dendritic segments'
'dendritic segment length average (?m)'
'dendritic segment length stdv (?m)'
'average tortuosity of axonal nodes'
'stdv of tortuosity of axonal nodes'
'average tortuosity of dendritic nodes'
'stdv of tortuosity of dendritic nodes'
'number dendritic sholl sections'
'dendritic sholl length at 50µm '
'dendritic sholl length at 100µm'
'dendritic sholl length at 150µm'
'number axonal sholl sections'
'axonal sholl length at 100µm '
'axonal sholl length at 200µm '
'axonal sholl length at 300µm '
'axonal sholl length density'
'axonal sholl node density'
'convex hull axon area (?m2)'
'convex hull axon perimeter (?m)'
'convex hull axon volume (?m3)'
'convex hull axon surface area (?m2)'
'convex hull dendrite area (?m2)'
'convex hull dendrite perimeter (?m)'
'convex hull dendrite volume (?m3)'
'convex hull dendrite surface area (?m2)'
'relative distance to pia'

Anexo 2. Conjunto de rasgos extraídos con L_measure

L_measure neurona	L_measure dendritas	L_measure axones
' Bif_ampl_local of Neuron_ Avg'	' Bif_ampl_local of Dendrites_ Avg'	'Bif_ampl_local of Axon_ Avg'
' Bif_ampl_local of Neuron_ Stdv'	' Bif_ampl_local of Dendrites_ Stdv'	'Bif_ampl_local of Axon_ Stdv'
' Bif_ampl_remote of Neuron_ Avg'	' Bif_ampl_remote of Dendrites_ Avg'	'Bif_ampl_remote of Axon_ Avg'
' Bif_ampl_remote of Neuron_ Stdv'	' Bif_ampl_remote of Dendrites_ Stdv'	'Bif_ampl_remote of Axon_ Stdv'
' Bif_tilt_local of Neuron_ Avg'	' Bif_tilt_local of Dendrites_ Avg'	'Bif_tilt_local of Axon_ Avg'
' Bif_tilt_local of Neuron_ Stdv'	' Bif_tilt_local of Dendrites_ Stdv'	'Bif_tilt_local of Axon_ Stdv'
' Bif_tilt_remote of Neuron_ Avg'	' Bif_tilt_remote of Dendrites_ Avg'	'Bif_tilt_remote of Axon_ Avg'
' Bif_tilt_remote of Neuron_ Stdv'	' Bif_tilt_remote of Dendrites_ Stdv'	'Bif_tilt_remote of Axon_ Stdv'
' Bif_torque_local of Neuron_ Avg'	' Bif_torque_local of Dendrites_ Avg'	'Bif_torque_local of Axon_ Avg'
' Bif_torque_local of Neuron_ Stdv'	' Bif_torque_local of Dendrites_ Stdv'	'Bif_torque_local of Axon_ Stdv'
' Bif_torque_remote of Neuron_ Avg'	' Bif_torque_remote of Dendrites_ Avg'	'Bif_torque_remote of Axon_ Avg'
' Bif_torque_remote of Neuron_ Stdv'	' Bif_torque_remote of Dendrites_ Stdv'	'Bif_torque_remote of Axon_ Stdv'
' Branch_Order of Neuron_ Avg'	' Branch_Order of Dendrites_ Avg'	'Branch_Order of Axon_ Avg'
' Branch_Order of Neuron_ Stdv'	' Branch_Order of Dendrites_ Stdv'	'Branch_Order of Axon_ Stdv'
' Branch_pathlength of Neuron_ Avg'	' Contraction of Dendrites_ Avg'	'Contraction of Axon_ Avg'
' Branch_pathlength of Neuron_ Stdv'	' Contraction of Dendrites_ Stdv'	'Contraction of Axon_ Stdv'
' Contraction of Neuron_ Avg'	' Daughter_Ratio of Dendrites_ Avg'	'Daughter_Ratio of Axon_ Avg'
' Contraction of Neuron_ Stdv'	' Daughter_Ratio of Dendrites_ Stdv'	'Daughter_Ratio of Axon_ Stdv'
' Daughter_Ratio of Neuron_ Avg'	' Depth of Dendrites_ Avg'	'Depth of Axon_ Avg'
' Daughter_Ratio of Neuron_ Stdv'	' Diam_threshold of Dendrites_ Avg'	'Diam_threshold of Axon_ Avg'
' Depth of Neuron_ Avg'	' Diam_threshold of Dendrites_ Stdv'	'Diam_threshold of Axon_ Stdv'
' Diam_threshold of Neuron_ Avg'	' Diameter of Dendrites_ Avg'	'Diameter of Axon_ Avg'
' Diam_threshold of Neuron_ Stdv'	' Diameter of Dendrites_ Stdv'	'Diameter of Axon_ Stdv'
' Diameter of Neuron_ Avg'	' Diameter_pow of Dendrites_ Avg'	'Diameter_pow of Axon_ Avg'
' Diameter of Neuron_ Stdv'	' Diameter_pow of Dendrites_ Stdv'	'Diameter_pow of Axon_ Stdv'
' Diameter_pow of Neuron_ Avg'	' EucDistance of Dendrites_ Avg'	'EucDistance of Axon_ Avg'
' Diameter_pow of Neuron_ Stdv'	' EucDistance of Dendrites_ Stdv'	'EucDistance of Axon_ Stdv'
' EucDistance of Neuron_ Avg'	' Fractal_Dim of Dendrites_ Avg'	'Fractal_Dim of Axon_ Avg'
' EucDistance of Neuron_ Stdv'	' Fractal_Dim of Dendrites_ Stdv'	'Fractal_Dim of Axon_ Stdv'
' Fractal_Dim of Neuron_ Avg'	' Fragmentation of Dendrites_ Avg'	'Fragmentation of Axon_ Avg'
' Fractal_Dim of Neuron_ Stdv'	' Fragmentation of Dendrites_ Stdv'	'Fragmentation of Axon_ Stdv'
' Fragmentation of Neuron_ Avg'	' Height of Dendrites_ Avg'	'Height of Axon_ Avg'
' Fragmentation of Neuron_ Stdv'	' Helix of Dendrites_ Avg'	'Helix of Axon_ Avg'
' Height of Neuron_ Avg'	' Helix of Dendrites_ Stdv'	'Helix of Axon_ Stdv'
' Helix of Neuron_ Avg'	' HillmanThreshold of Dendrites_ Avg'	'HillmanThreshold of Axon_ Avg'
' Helix of Neuron_ Stdv'	' HillmanThreshold of Dendrites_ Stdv'	'HillmanThreshold of Axon_ Stdv'

' HillmanThreshold of Neuron_ Avg'	' Last_parent_diam of Dendrites_ Avg'	'Last_parent_diam of Axon_ Avg'
' HillmanThreshold of Neuron_ Stdv'	' Last_parent_diam of Dendrites_ Stdv'	'Last_parent_diam of Axon_ Stdv'
' Last_parent_diam of Neuron_ Avg'	' N_branch of Dendrites_ Total'	'N_branch of Axon_ Total'
' Last_parent_diam of Neuron_ Stdv'	' N_tips of Dendrites_ Total'	'N_tips of Axon_ Total'
' Length of Neuron_ Total'	' Parent_Daughter_Ratio of Dendrites_ Avg'	'Parent_Daughter_Ratio of Axon_ Avg'
' N_bifs of Neuron_ Total'	' Parent_Daughter_Ratio of Dendrites_ Stdv'	'Parent_Daughter_Ratio of Axon_ Stdv'
' N_branch of Neuron_ Total'	' Partition_asymmetry of Dendrites_ Avg'	'Partition_asymmetry of Axon_ Avg'
' N_stems of Neuron_ Total'	'Partition_asymmetry of Dendrites_ Stdv'	'Partition_asymmetry of Axon_ Stdv'
'N_tips of Neuron_ Total'	' PathDistance of Dendrites_ Avg'	'PathDistance of Axon_ Avg'
' Parent_Daughter_Ratio of Neuron_ Avg'	' PathDistance of Dendrites_ Stdv'	'PathDistance of Axon_ Stdv'
' Parent_Daughter_Ratio of Neuron_ Stdv'	' Pk of Dendrites_ Avg'	'Pk of Axon_ Avg'
' Partition_asymmetry of Neuron_ Avg'	' Pk of Dendrites_ Stdv'	'Pk of Axon_ Stdv'
' Partition_asymmetry of Neuron_ Stdv'	' Pk_2 of Dendrites_ Avg'	'Pk_2 of Axon_ Avg'
' PathDistanceof Neuron_ Avg'	' Pk_2 of Dendrites_ Stdv'	'Pk_2 of Axon_ Stdv'
' PathDistanceof Neuron_ Stdv'	' Pk_classic of Dendrites_ Avg'	'Pk_classic of Axon_ Avg'
' Pk of Neuron_ Avg'	' Pk_classic of Dendrites_ Stdv'	'Pk_classic of Axon_ Stdv'
' Pk of Neuron_ Stdv'	' Rall_Power of Dendrites_ Avg'	'Rall_Power of Axon_ Avg'
' Pk_2 of Neuron_ Avg'	' Rall_Power of Dendrites_ Stdv'	'Rall_Power of Axon_ Stdv'
' Pk_2 of Neuron_ Stdv'	' SectionArea of Dendrites_ Avg'	'SectionArea of Axon_ Avg'
' Pk_classic of Neuron_ Avg'	' SectionArea of Dendrites_ Stdv'	'SectionArea of Axon_ Stdv'
' Pk_classic of Neuron_ Stdv'	' Taper_1 of Dendrites_ Avg'	'Taper_1 of Axon_ Avg'
' Rall_Power of Neuron_ Avg'	' Taper_1 of Dendrites_ Stdv'	'Taper_1 of Axon_ Stdv'
' Rall_Power of Neuron_ Stdv'	' Taper_2 of Dendrites_ Avg'	'Taper_2 of Axon_ Avg'
' SectionArea of Neuron_ Avg'	' Taper_2 of Dendrites_ Stdv'	'Taper_2 of Axon_ Stdv'
'SectionArea of Neuron_ Stdv'	' Terminal_degree of Dendrites_ Avg'	'Terminal_degree of Axon_ Avg'
' Soma_Surface of Neuron_ Total'	' Terminal_degree of Dendrites_ Stdv'	'Terminal_degree of Axon_ Stdv'
' Surfaceof Neuron_ Total'	' TerminalSegment of Dendrites_ Total'	'TerminalSegment of Axon_ Total'
' Surface of Neuron_ Avg'	' Volume of Dendrites_ Total'	'Volume of Axon_ Total'
' Surface of Neuron_ Stdv'	' Volume of Dendrites_ Avg'	'Volume of Axon_ Avg'
' Taper_1 of Neuron_ Avg'	' Volume of Dendrites_ Stdv'	'Volume of Axon_ Stdv'
' Taper_1 of Neuron_ Stdv'	' Width of Dendrites_ Avg'	'Width of Axon_ Avg'
' Taper_2 of Neuron_ Avg'	'Branch_pathlength of Dendrites_ Max'	'Branch_pathlength of Axon_ Max '
' Taper_2 of Neuron_ Stdv'	'EucDistance of Dendrites_ Max'	'EucDistance of Axon_ Max '
' Terminal_degree of Neuron_ Avg'	'PathDistance of Dendrites_ Max'	'PathDistance of Axon_ Max '
' Terminal_degreeof Neuron_ Stdv'	'Terminal_degree of Dendrites_ Max'	'Terminal_degree of Axon_ Max '

' TerminalSegment of Neuron_ Total'
' Volume of Neuron_ Total'
' Volume of Neuron_ Avg'
' Volume of Neuron_ Stdv'
' Width of Neuron_ Avg'
'Branch_Order of Neuron_ Max '
'Branch_pathlength of Neuron_ Max '
'EucDistance of Neuron_ Max '
'PathDistance of Neuron_ Max '
'Terminal_degree of Neuron_ Max'

Anexo 3. Conjunto de rasgos extraídos con Farsight

Farsight neurona	Farsight dendritas	Farsight axones
Euclidean_Skewness_Neuron	Leaf_Nodes_Dendrites	Leaf_Nodes_Axon
Branching_Stems_Neuron	Ave_Azimuth_Dendrites	Ave_Azimuth_Axon
Leaf_Nodes_Neuron	Ave_Elevation_Dendrites	Ave_Elevation_Axon
Ave_Azimuth_Neuron	Ave_Leaf_Level_Dendrites	Ave_Leaf_Level_Axon
Ave_Elevation_Neuron	Ave_Leaf_Path_Length_Dendrites	Ave_Leaf_Path_Length_Axon
Ave_Leaf_Level_Neuron	Max_Leaf_Level_Dendrites	Max_Leaf_Level_Axon
Ave_Leaf_Path_Length_Neuron	Max_Leaf_Path_Length_Dendrites	Max_Leaf_Path_Length_Axon
Max_Leaf_Level_Neuron	Ave_Branch_Pt_to_Soma_Euclidean_Distance_Dendrites	Ave_Branch_Pt_to_Soma_Euclidean_Distance_Axon
Max_Leaf_Path_Length_Neuron	Max_Branch_Pt_to_Soma_Euclidean_Distance_Dendrites	Max_Branch_Pt_to_Soma_Euclidean_Distance_Axon
Ave_Branch_Pt_to_Soma_Euclidean_Distance_Neuron	Ave_Tip_to_Soma_Euclidean_Distance_Dendrites	Ave_Tip_to_Soma_Euclidean_Distance_Axon
Max_Branch_Pt_to_Soma_Euclidean_Distance_Neuron	Max_Tip_to_Soma_Euclidean_Distance_Dendrites	Max_Tip_to_Soma_Euclidean_Distance_Axon
Ave_Tip_to_Soma_Euclidean_Distance_Neuron	Overall_Magnitude_of_Tips_Dendrites	Overall_Magnitude_of_Tips_Axon
Max_Tip_to_Soma_Euclidean_Distance_Neuron	Overall_Azimuth_of_Tips_Dendrites	Overall_Azimuth_of_Tips_Axon
Overall_Magnitude_of_Tips_Neuron	Overall_Elevation_of_Tips_Dendrites	Overall_Elevation_of_Tips_Axon
Overall_Azimuth_of_Tips_Neuron		
Overall_Elevation_of_Tips_Neuron		

Anexo 4. Conjunto de rasgos extraídos con TreesToolbox

TreesToolbox neurona	TreesToolbox dendritas	TreesToolbox axones
'neuron sholl intersections at 50'	'dendritic sholl intersections at 50'	'axonal sholl intersections at 50'
'neuron sholl intersections at 100'	'dendritic sholl intersections at 100'	'axonal sholl intersections at 100'
'neuron sholl intersections at 150'	'dendritic sholl intersections at 150'	'axonal sholl intersections at 150'

'suma of sholl intersections 50-150'	'suma of sholl intersections 50-150'	'suma of sholl intersections 50-150'
'neuron sholl intersections at200'	'dendritic sholl intersections at200'	'axonal sholl intersections at200'
'neuron sholl intersections at250'	'dendritic sholl intersections at250'	'axonal sholl intersections at250'
'neuron sholl intersections at300'	'dendritic sholl intersections at300'	'axonal sholl intersections at300'
'suma of sholl intersections 50-300'	'suma of sholl intersections 50-300'	'suma of sholl intersections 50-300'
'neuron sholl intersections at350'	'dendritic sholl intersections at350'	'axonal sholl intersections at350'
'neuron sholl intersections at400'	'dendritic sholl intersections at400'	'axonal sholl intersections at400'
'neuron sholl intersections at450'	'dendritic sholl intersections at450'	'axonal sholl intersections at450'
'neuron sholl intersections at500'	'dendritic sholl intersections at500'	'axonal sholl intersections at500'
'suma of sholl intersections 50-500'	'suma of sholl intersections 50-500'	'suma of sholl intersections 50-500'
'neuron sholl intersections at550'	'suma of sholl intersections 50-1000'	'axonal sholl intersections at550'
'neuron sholl intersections at600'		'axonal sholl intersections at600'
'neuron sholl intersections at650'		'axonal sholl intersections at650'
'neuron sholl intersections at700'		'axonal sholl intersections at700'
'neuron sholl intersections at750'		'axonal sholl intersections at750'
'neuron sholl intersections at800'		'axonal sholl intersections at800'
'neuron sholl intersections at850'		'axonal sholl intersections at850'
'neuron sholl intersections at900'		'axonal sholl intersections at900'
'neuron sholl intersections at950'		'axonal sholl intersections at950'
'neuron sholl intersections at1000'		'axonal sholl intersections at1000'
'suma of sholl intersections 50-1000'		'suma of sholl intersections 50-1000'

Anexo 5. Programa en Matlab para la clasificación

```
%% Inicializando
% Añadiendo el camino a los códigos de matlab2weka
addpath([pwd filesep 'matlab2weka']);
% Añadiendo el archivo Weka Jar
javaaddpath('C:\Program Files\Weka-3-6\weka.jar');
% Añadiendo el archivo matlab2weka Jar que convierte las matrices de matlab
% a instancias de weka
javaaddpath([pwd filesep 'matlab2weka' filesep 'matlab2weka.jar']);
%importar funciones
import java.util.Enumeration
import java.lang.String
import matlab2weka.*
import weka.core.converters.ArffLoader.*
import weka.core.range.*
import weka.classifiers.*;
import java.util.*
import weka.classifiers.evaluation.*
import weka.classifiers.Evaluation.numInstances.*
import weka.core.range.*
import weka.classifiers.*;
import weka.classifiers.evaluation.output.prediction.*
% Cargar el datos .arff de weka
 [file, path] = uigetfile( ...
```

```
{  '*.arff','xls-files (*.arff)'; ...
   '*.*',  'All Files (*.*)'}, ...
   'Pick a file', ...
   'MultiSelect', 'off');
datos = loadARFF([path file]);
%%
%prepara opciones para validación cruzada
random = javaObject('java.util.Random');
buffer = javaObject('java.lang.StringBuffer');
range = javaObject('weka.core.Range','1');
bool = javaObject('java.lang.Boolean',false);
array = javaArray('java.lang.Object',3);
array(1) = buffer;
array(2) = range;
array(3) = bool;
%% Clasificación con NaiveBayes
%si se desea usar otro clasificador solo se debe habilitar.
%Crea el objeto clasificador
clasificador = weka.classifiers.bayes.NaiveBayes();
%clasificador = weka.classifiers.functions.Logistic();
%clasificador = weka.classifiers.functions.MultilayerPerceptron();
%clasificador = weka.classifiers.functions.SMO();
%clasificador = weka.classifiers.lazy.IBk(5);
%
%Ciclo para repetir 10 veces la clasificación.
for i = 1:10
%myrand = Random(1);%para semilla igual al weka
myrand = Random; %para aleatorio siempre
eval = weka.classifiers.Evaluation(datos); %objeto evaluador
eval.crossValidateModel(clasificador,datos,10,myrand,array);%ValCr = 10
%== Sumario ==
%Índices de la clasificacion
CC(i) = eval.pctCorrect();
TN_Rate(i) = eval.weightedTrueNegativeRate();
TP_Rate(i) = eval.weightedTruePositiveRate();
FN_Rate(i) = eval.weightedFalseNegativeRate();
FP_Rate(i) = eval.weightedFalsePositiveRate();
Precision(i) = eval.weightedPrecision();
Sensibilidad(i) = eval.weightedRecall();
MedidaF(i) = eval.weightedFMeasure();
AUC(i) = eval.weightedAreaUnderROC();
end
%Después de clasificar 10 veces se calcula la media
CCmean = mean(CC);
TN_Ratemean = mean(TN_Rate);
TP_Ratemean = mean(TP_Rate);
FN_Ratemean = mean(FN_Rate);
FP_Ratemean = mean(FP_Rate);
Precisionmean = mean(Precision);
Sensibilidadmean = mean(Sensibilidad);
MedidaFmean = mean(MedidaF);
ROC_Areamean = mean(AUC);
%Agregar media al final del vector.
CC(11) = CCmean;
TN_Rate(11) = TN_Ratemean;
TP_Rate(11) = TP_Ratemean;
FN_Rate(11) = FN_Ratemean;
```

```
FP_Rate(11) = FP_Ratemean;
Precision(11) = Precisionmean;
Sensibilidad(11) = Sensibilidadmean;
MedidaF(11) = MedidaFmean;
AUC(11) = ROC_Areamean;
%para conformar la tabla de resultados
nombres={'CC' 'Precisión' 'Sensibilidad' 'Medida F' 'AUC'};
tabla=[CC' Precision' Sensibilidad' MedidaF' AUC'];
tabla1 = num2cell(tabla);
tabla1=cat(1,nombres,tabla1); %uno los nombres y los datos
xlswrite('ResultadoClasf.xls',tabla1); %exportar resultados
clear all;
```

Anexo 6. Resultados de los cinco algoritmos de clasificación aplicados a todos los grupos de rasgos

Tabla A6.1: Resultados de la clasificación en todos los grupos usando NB.

Grupos de Rasgos	CC	Precisión	Sensibilidad	Medida F	AUC
farsightAxón14	71.57	0.73	0.72	0.72	0.80
farsightDend14	73.99	0.74	0.74	0.73	0.79
farsightNeuron14	77.01	0.77	0.77	0.77	0.83
LmeasAxón22	71.20	0.72	0.71	0.71	0.79
LmeasDend22	74.81	0.75	0.75	0.74	0.81
LmeasNeuron22	75.82	0.77	0.76	0.76	0.83
lucidaAxón20	72.11	0.75	0.72	0.72	0.82
lucidaDend20	74.43	0.74	0.74	0.74	0.80
lucidaNeuron6	60.88	0.62	0.61	0.61	0.64
sholl_Neuron14	54.06	0.67	0.54	0.51	0.67
shollAxón14	57.30	0.69	0.57	0.56	0.67
shollDend14	67.42	0.67	0.67	0.67	0.66

Tabla A6.2: Resultados de la clasificación en todos los grupos usando LR.

Grupos de Rasgos	CC	Precisión	Sensibilidad	Medida F	AUC
farsightAxón14	74.03	0.74	0.74	0.74	0.81
farsightDend14	77.67	0.78	0.78	0.77	0.81
farsightNeuron14	77.11	0.77	0.77	0.77	0.83
LmeasAxón22	75.06	0.75	0.75	0.75	0.82
LmeasDend22	81.35	0.81	0.81	0.81	0.86
LmeasNeuron22	76.04	0.76	0.76	0.76	0.85
lucidaAxón20	77.11	0.77	0.77	0.77	0.83
lucidaDend20	83.55	0.83	0.84	0.83	0.90
lucidaNeuron6	60.69	0.59	0.61	0.59	0.64
sholl_Neuron14	71.42	0.71	0.71	0.70	0.75
shollAxón14	63.58	0.63	0.64	0.63	0.73
shollDend14	76.13	0.76	0.76	0.75	0.79

Tabla A6.3: Resultados de la clasificación en todos los grupos usando MLP

Grupos de Rasgos	CC	Precisión	Sensibilidad	Medida F	AUC
farsightAxón14	71.10	0.72	0.71	0.71	0.78
farsightDend14	76.07	0.76	0.76	0.76	0.80
farsightNeuron14	74.47	0.74	0.74	0.74	0.79
LmeasAxón22	73.43	0.74	0.73	0.73	0.80
LmeasDend22	79.43	0.79	0.79	0.79	0.86
LmeasNeuron22	78.93	0.79	0.79	0.79	0.86
lucidaAxón20	73.27	0.73	0.73	0.73	0.80
lucidaDend20	84.91	0.85	0.85	0.85	0.91
lucidaNeuron6	60.03	0.59	0.60	0.59	0.62
sholl_Neuron14	75.22	0.75	0.75	0.75	0.79
shollAxón14	65.38	0.67	0.65	0.66	0.71
shollDend14	74.47	0.74	0.74	0.74	0.77

Tabla A6.4: Resultados de la clasificación en todos los grupos usando SMO

Grupos de Rasgos	CC	Precisión	Sensibilidad	Medida F	AUC
farsightAxón14	74.50	0.74	0.74	0.74	0.72
farsightDend14	76.45	0.78	0.76	0.75	0.72
farsightNeuron14	76.67	0.76	0.77	0.76	0.75
LmeasAxón22	73.02	0.73	0.73	0.73	0.71
LmeasDend22	76.82	0.79	0.77	0.75	0.72
LmeasNeuron22	77.45	0.77	0.77	0.77	0.76
lucidaAxón20	75.72	0.76	0.76	0.76	0.75
lucidaDend20	80.66	0.82	0.81	0.80	0.77
lucidaNeuron6	60.38	0.36	0.60	0.45	0.50
sholl_Neuron14	65.94	0.67	0.66	0.61	0.59
shollAxón14	60.03	0.44	0.60	0.46	0.50
shollDend14	68.71	0.69	0.69	0.66	0.63

Tabla A6.5: Resultados de la clasificación en todos los grupos usando IBk con k = 5

Grupos de Rasgos	CC	Precisión	Sensibilidad	Medida F	AUC
farsightAxón14	75.94	0.77	0.76	0.76	0.82
farsightDend14	77.61	0.77	0.78	0.77	0.80
farsightNeuron14	75.50	0.76	0.76	0.76	0.81
LmeasAxón22	73.30	0.73	0.73	0.73	0.80
LmeasDend22	78.18	0.78	0.78	0.78	0.87
LmeasNeuron22	80.75	0.81	0.81	0.81	0.87

lucidaAxón20	71.04	0.72	0.71	0.71	0.78
lucidaDend20	83.24	0.83	0.83	0.83	0.90
lucidaNeuron6	62.86	0.62	0.63	0.62	0.62
sholl_Neuron14	72.45	0.72	0.72	0.72	0.76
shollAxón14	61.64	0.62	0.62	0.62	0.66
shollDend14	70.88	0.70	0.71	0.71	0.76

Anexo 7. Resultados de la clasificación a partir de los rasgos obtenidos con Neurolucida, Lmeasure-Farsight-TreesToolbox y la unión de ambos conjuntos

Tabla A7.1: Resultados de la clasificación en rasgos obtenidos con Neurolucida usando NB.

Evaluador	M.Búsqueda	CC %	Precisión	Sensibilidad	Medida F	AUC	Rasgos
SVMAttributeEval	Ranker	82.7673	0.8285	0.8277	0.8280	0.8946	24
CfsSubsetEval	GreedyStepwise-Forward	84.7170	0.8498	0.8472	0.8479	0.9095	12
WrapperSubsetEval	Genetic	82.7044	0.8341	0.8270	0.8284	0.8849	21
	GreedyStepwise-Forward	83.4277	0.8337	0.8343	0.8338	0.8922	10
	GreedyStepwise-Backward	76.3836	0.7621	0.7638	0.7624	0.8163	25
	Genetic	88.0818	0.8817	0.8808	0.8811	0.9201	17

Tabla A7.2: Resultados de la clasificación en rasgos obtenidos con Neurolucida usando LR.

Evaluador	M.Búsqueda	CC %	Precisión	Sensibilidad	Medida F	AUC	Rasgos
SVMAttributeEval	Ranker	88.8994	0.8889	0.8890	0.8889	0.9421	24
CfsSubsetEval	GreedyStepwise-Forward	83.9308	0.8385	0.8393	0.8383	0.9180	12
WrapperSubsetEval	Genetic	87.7044	0.8772	0.8770	0.8771	0.9397	21
	GreedyStepwise-Forward	81.1006	0.8165	0.8110	0.8046	0.8093	4
	GreedyStepwise-Backward	76.2579	0.7617	0.7626	0.7619	0.8361	25
	Genetic	89.6855	0.8967	0.8969	0.8964	0.9427	16

Tabla A7.3: Resultados de la clasificación en rasgos obtenidos con Neurolucida usando MLP.

Evaluador	M.Búsqueda	CC %	Precisión	Sensibilidad	Medida F	AUC	Rasgos
SVMAttributeEval	Ranker	87.5157	0.8751	0.8752	0.8750	0.9367	24
CfsSubsetEval	GreedyStepwise-Forward	84.7170	0.8471	0.8472	0.8469	0.9120	12
WrapperSubsetEval	Genetic	88.8679	0.8887	0.8887	0.8886	0.9375	21
	GreedyStepwise-Forward	86.9497	0.8705	0.8695	0.8679	0.8951	5
	GreedyStepwise-Backward	75.4717	0.7562	0.7547	0.7549	0.8302	27
	Genetic	87.5157	0.8748	0.8752	0.8748	0.9298	27

Tabla A7.4: Resultados de la clasificación en rasgos obtenidos con Neurolucida usando SMO.

Evaluador	M.Búsqueda	CC %	Precisión	Sensibilidad	Medida F	AUC	Rasgos
SVMAttributeEval	Ranker	86.7296	0.8669	0.8673	0.8664	0.8555	24
CfsSubsetEval	GreedyStepwise-Forward	81.9811	0.8225	0.8198	0.8152	0.7942	12
	Genetic	85.2201	0.8523	0.8522	0.8504	0.8360	21
WrapperSubsetEval	GreedyStepwise-Forward	83.0189	0.8454	0.8302	0.8222	0.7953	6
	GreedyStepwise-Backward	75.3145	0.7524	0.7531	0.7454	0.7216	27
	Genetic	87.9874	0.8816	0.8799	0.8781	0.8633	19

Tabla A7.5: Resultados de la clasificación en rasgos obtenidos con Neurolucida usando IBk.

Evaluador	M.Búsqueda	CC %	Precisión	Sensibilidad	Medida F	AUC	Rasgos
SVMAttributeEval	Ranker	85.3774	0.8543	0.8538	0.8539	0.9179	24
CfsSubsetEval	GreedyStepwise-Forward	84.3082	0.8490	0.8431	0.8443	0.9082	12
	Genetic	82.9560	0.8309	0.8296	0.8300	0.8971	21
WrapperSubsetEval	GreedyStepwise-Forward	86.6667	0.8679	0.8667	0.8647	0.8935	9
	GreedyStepwise-Backward	81.4465	0.8136	0.8145	0.8138	0.8836	26
	Genetic	85.7233	0.8603	0.8572	0.8580	0.9115	19

Tabla A7.6: Resultados de la clasificación en rasgos obtenidos con Lmeasure-Farsight-TreesToolbox usando NB.

Evaluador	M.Búsqueda	CC %	Precisión	Sensibilidad	Medida F	AUC	Rasgos
SVMAttributeEval	Ranker	84.8113	0.8476	0.8481	0.8477	0.9001	25
CfsSubsetEval	GreedyStepwise-Forward	82.8616	0.8278	0.8286	0.8269	0.8883	25
	Genetic	82.9245	0.8287	0.8292	0.8271	0.9032	21
WrapperSubsetEval	GreedyStepwise-Forward	85.4717	0.8543	0.8547	0.8535	0.9005	9
	GreedyStepwise-Backward	72.8931	0.7282	0.7289	0.7285	0.7851	27
	Genetic	86.2579	0.8621	0.8626	0.8619	0.9037	20

Tabla A7.7: Resultados de la clasificación en rasgos obtenidos con Lmeasure-Farsight-TreesToolbox usando LR.

Evaluador	M.Búsqueda	CC %	Precisión	Sensibilidad	Medida F	AUC	Rasgos
SVMAttributeEval	Ranker	91.0063	0.9099	0.9101	0.9099	0.9562	25
CfsSubsetEval	GreedyStepwise-Forward	84.6226	0.8457	0.8462	0.8457	0.9332	25
	Genetic	85.7233	0.8568	0.8572	0.8568	0.9201	21
WrapperSubsetEval	GreedyStepwise-Forward	88.5849	0.8855	0.8858	0.8855	0.9336	8
	GreedyStepwise-Backward	87.0126	0.8701	0.8701	0.8701	0.9285	27

	Genetic	90.5975	0.9060	0.9060	0.9059	0.9478	23

Tabla A7.8: Resultados de la clasificación en rasgos obtenidos con Lmeasure-Farsight-TreesToolbox usando MLP.

Evaluador	**M.Búsqueda**	**CC %**	**Precisión**	**Sensibilidad**	**Medida F**	**AUC**	**Rasgos**
SVMAttributeEval	Ranker	88.9937	0.8900	0.8899	0.8899	0.9578	25
CfsSubsetEval	GreedyStepwise-Forward	84.8428	0.8491	0.8484	0.8486	0.9267	25
	Genetic	83.4277	0.8343	0.8343	0.8340	0.9061	21
WrapperSubsetEval	GreedyStepwise-Forward	85.2516	0.8524	0.8525	0.8519	0.9091	7
	GreedyStepwise-Backward	84.9686	0.8500	0.8497	0.8496	0.9269	27
	Genetic	90.3774	0.9039	0.9038	0.9038	0.9551	27

Tabla A7.9: Resultados de la clasificación en rasgos obtenidos con Lmeasure-Farsight-TreesToolbox usando SMO.

Evaluador	**M.Búsqueda**	**CC %**	**Precisión**	**Sensibilidad**	**Medida F**	**AUC**	**Rasgos**
SVMAttributeEval	Ranker	90.4403	0.9045	0.9044	0.9038	0.8944	25
CfsSubsetEval	GreedyStepwise-Forward	86.6667	0.8686	0.8667	0.8645	0.8483	25
	Genetic	85.1572	0.8523	0.8516	0.8494	0.8336	21
WrapperSubsetEval	GreedyStepwise-Forward	86.7296	0.8730	0.8673	0.8640	0.8440	10
	GreedyStepwise-Backward	82.8616	0.8285	0.8286	0.8261	0.8096	27
	Genetic	91.1006	0.9109	0.9110	0.9107	0.9038	25

Tabla A7.10: Resultados de la clasificación en rasgos obtenidos con Lmeasure-Farsight-TreesToolbox usando IBK.

Evaluador	**M.Búsqueda**	**CC %**	**Precisión**	**Sensibilidad**	**Medida F**	**AUC**	**Rasgos**
SVMAttributeEval	Ranker	84.7484	0.8484	0.8475	0.8451	0.9031	25
CfsSubsetEval	GreedyStepwise-Forward	86.2893	0.8625	0.8629	0.8626	0.9192	25
	Genetic	85.4717	0.8544	0.8547	0.8535	0.9154	21
WrapperSubsetEval	GreedyStepwise-Forward	85.4403	0.8540	0.8544	0.8541	0.8955	8
	GreedyStepwise-Backward	81.9497	0.8190	0.8195	0.8190	0.8868	26
	Genetic	86.4780	0.8648	0.8648	0.8647	0.8935	14

Tabla A7.11: Resultados de la clasificación uniendo los rasgos obtenidos con Neurolucida y Lmeasure-Farsight-TreesToolbox usando NB.

Evaluador	**M.Búsqueda**	**CC %**	**Precisión**	**Sensibilidad**	**Medida F**	**AUC**	**Rasgos**
SVMAttributeEval	Ranker	86.4151	0.8637	0.8642	0.8638	0.9139	24
CfsSubsetEval	GreedyStepwise-	83.5849	0.8359	0.8358	0.8335	0.9147	23

	Forward						
	Genetic	83.7736	0.8369	0.8377	0.8364	0.9113	18
WrapperSubsetEval	GreedyStepwise-Forward	91.2579	0.9126	0.9126	0.9126	0.9535	12
	GreedyStepwise-Backward	75.8805	0.7563	0.7588	0.7564	0.8078	25
	Genetic	90.5346	0.9053	0.9053	0.9053	0.9480	21

Tabla A7.12: Resultados de la clasificación uniendo los rasgos obtenidos con Neurolucida y Lmeasure-Farsight-TreesToolbox usando LR.

Evaluador	**M.Búsqueda**	**CC %**	**Precisión**	**Sensibilidad**	**Medida F**	**AUC**	**Rasgos**
SVMAttributeEval	Ranker	92.5786	0.9258	0.9258	0.9258	0.9619	24
CfsSubsetEval	GreedyStepwise-Forward	87.7358	0.8771	0.8774	0.8771	0.9543	23
	Genetic	88.5535	0.8855	0.8855	0.8855	0.9451	18
WrapperSubsetEval	GreedyStepwise-Forward	90.7547	0.9076	0.9075	0.9070	0.9503	10
	GreedyStepwise-Backward	85.9748	0.8592	0.8597	0.8592	0.9295	26
	Genetic	93.5535	0.9356	0.9355	0.9355	0.9707	15

Tabla A7.13: Resultados de la clasificación uniendo los rasgos obtenidos con Neurolucida y Lmeasure-Farsight-TreesToolbox usando MLP.

Evaluador	**M.Búsqueda**	**CC %**	**Precisión**	**Sensibilidad**	**Medida F**	**AUC**	**Rasgos**
SVMAttributeEval	Ranker	92.6730	0.9268	0.9267	0.9267	0.9777	24
CfsSubsetEval	GreedyStepwise-Forward	85.6604	0.8570	0.8566	0.8566	0.9321	23
	Genetic	86.8553	0.8687	0.8686	0.8685	0.9368	18
WrapperSubsetEval	GreedyStepwise-Forward	88.9937	0.8907	0.8899	0.8897	0.9481	9
	GreedyStepwise-Backward	81.4780	0.8155	0.8148	0.8148	0.8931	27
	Genetic	93.1447	0.9314	0.9314	0.9313	0.9700	26

Tabla A7.14: Resultados de la clasificación uniendo los rasgos obtenidos con Neurolucida y Lmeasure-Farsight-TreesToolbox usando SMO.

Evaluador	**M.Búsqueda**	**CC %**	**Precisión**	**Sensibilidad**	**Medida F**	**AUC**	**Rasgos**
SVMAttributeEval	Ranker	93.4591	0.9347	0.9346	0.9343	0.9278	24
CfsSubsetEval	GreedyStepwise-Forward	85.5660	0.8550	0.8557	0.8548	0.8440	23
	Genetic	85.4717	0.8542	0.8547	0.8536	0.8415	18
WrapperSubsetEval	GreedyStepwise-Forward	83.5220	0.8453	0.8352	0.8290	0.8043	5
	GreedyStepwise-Backward	83.0189	0.8323	0.8302	0.8264	0.8069	27
	Genetic	94.5283	0.9453	0.9453	0.9453	0.9428	23

Tabla A7.15: Resultados de la clasificación uniendo los rasgos obtenidos con Neurolucida y Lmeasure-Farsight-TreesToolbox usando IBK.

Evaluador	M.Búsqueda	CC %	Precisión	Sensibilidad	Medida F	AUC	Rasgos
SVMAttributeEval	Ranker	90.0629	0.9009	0.9006	0.8999	0.9457	24
CfsSubsetEval	GreedyStepwise-Forward	85.6289	0.8562	0.8563	0.8562	0.9292	23
	Genetic	87.5472	0.8752	0.8755	0.8747	0.9341	18
WrapperSubsetEval	GreedyStepwise-Forward	88.8365	0.8887	0.8884	0.8874	0.9295	11
	GreedyStepwise-Backward	78.9308	0.7876	0.7893	0.7875	0.8627	27
	Genetic	90.2830	0.9028	0.9028	0.9028	0.9383	15

Anexo 8. Resultados de la clasificación en rasgos de las dendritas y los axones.

Tabla A8.1: Resultados de la clasificación en rasgos de los axones usando NB.

Evaluador	M.Búsqueda	CC %	Precisión	Sensibilidad	Medida F	AUC	Rasgos
SVMAttributeEval	Ranker	75.8491	0.7894	0.7585	0.7609	0.8617	25
CfsSubsetEval	GreedyStepwise-Forward	74.7799	0.7929	0.7478	0.7495	0.8788	13
	Genetic	75.5660	0.7901	0.7557	0.7580	0.8723	12
WrapperSubsetEval	GreedyStepwise-Forward	78.1447	0.7859	0.7814	0.7828	0.8150	6
	GreedyStepwise-Backward	72.0440	0.7367	0.7204	0.7234	0.7637	18
	Genetic	80.1572	0.8076	0.8016	0.8030	0.8694	20

Tabla A8.2: Resultados de la clasificación en rasgos de los axones usando LR.

Evaluador	M.Búsqueda	CC %	Precisión	Sensibilidad	Medida F	AUC	Rasgos
SVMAttributeEval	Ranker	82.1384	0.8204	0.8214	0.8205	0.8811	25
CfsSubsetEval	GreedyStepwise-Forward	81.1635	0.8112	0.8116	0.8113	0.8764	13
	Genetic	81.0063	0.8102	0.8101	0.8101	0.8678	12
WrapperSubsetEval	GreedyStepwise-Forward	81.1635	0.8104	0.8116	0.8101	0.8702	9
	GreedyStepwise-Backward	76.3208	0.7613	0.7632	0.7617	0.8327	17
	Genetic	82.5786	0.8257	0.8258	0.8256	0.8810	19

Tabla A8.3: Resultados de la clasificación en rasgos de los axones usando MLP.

Evaluador	M.Búsqueda	CC %	Precisión	Sensibilidad	Medida F	AUC	Rasgos
SVMAttributeEval	Ranker	80.8176	0.8099	0.8082	0.8086	0.8757	25
CfsSubsetEval	GreedyStepwise-Forward	78.1761	0.7826	0.7818	0.7819	0.8601	13
	Genetic	79.0566	0.7933	0.7906	0.7914	0.8673	12
WrapperSubsetEval	GreedyStepwise-	77.5157	0.7773	0.7752	0.7757	0.8388	6

	Forward						
	GreedyStepwise-Backward	76.8553	0.7700	0.7686	0.7691	0.8356	15
	Genetic	83.0818	0.8302	0.8308	0.8299	0.8996	26

Tabla A8.4: Resultados de la clasificación en rasgos de los axones usando SMO.

Evaluador	**M.Búsqueda**	**CC %**	**Precisión**	**Sensibilidad**	**Medida F**	**AUC**	**Rasgos**
SVMAttributeEval	Ranker	81.5409	0.8167	0.8154	0.8159	0.8099	25
CfsSubsetEval	GreedyStepwise-Forward	76.9182	0.7755	0.7692	0.7709	0.7682	13
	Genetic	77.9560	0.7824	0.7796	0.7805	0.7746	12
WrapperSubsetEval	GreedyStepwise-Forward	75.5660	0.7781	0.7557	0.7583	0.7678	4
	GreedyStepwise-Backward	77.0126	0.7684	0.7701	0.7687	0.7543	21
	Genetic	82.1698	0.8211	0.8217	0.8194	0.8034	20

Tabla A8.5: Resultados de la clasificación en rasgos de los axones usando IBK.

Evaluador	**M.Búsqueda**	**CC %**	**Precisión**	**Sensibilidad**	**Medida F**	**AUC**	**Rasgos**
SVMAttributeEval	Ranker	79.7484	0.7998	0.7975	0.7983	0.8866	25
CfsSubsetEval	GreedyStepwise-Forward	77.1384	0.7819	0.7714	0.7735	0.8619	13
	Genetic	78.4277	0.7922	0.7843	0.7861	0.8666	12
WrapperSubsetEval	GreedyStepwise-Forward	72.1698	0.7325	0.7217	0.7243	0.7227	3
	GreedyStepwise-Backward	75.6918	0.7546	0.7569	0.7520	0.7312	16
	Genetic	79.6855	0.7957	0.7969	0.7959	0.7838	21

Tabla A8.6: Resultados de la clasificación en rasgos de las dendritas usando NB.

Evaluador	**M.Búsqueda**	**CC %**	**Precisión**	**Sensibilidad**	**Medida F**	**AUC**	**Rasgos**
SVMAttributeEval	Ranker	79.3082	0.7933	0.7931	0.7883	0.8465	24
CfsSubsetEval	GreedyStepwise-Forward	80.3459	0.8036	0.8035	0.7994	0.8692	17
	Genetic	80.1887	0.8019	0.8019	0.7979	0.8709	18
WrapperSubsetEval	GreedyStepwise-Forward	84.4025	0.8459	0.8440	0.8411	0.8845	5
	GreedyStepwise-Backward	83.8050	0.8381	0.8381	0.8358	0.8696	15
	Genetic	82.5786	0.8248	0.8258	0.8245	0.8728	18

Tabla A8.7: Resultados de la clasificación en rasgos de las dendritas usando LR.

Evaluador	**M.Búsqueda**	**CC %**	**Precisión**	**Sensibilidad**	**Medida F**	**AUC**	**Rasgos**
SVMAttributeEval	Ranker	88.8994	0.8888	0.8890	0.8888	0.9577	24
CfsSubsetEval	GreedyStepwise-Forward	86.1635	0.8611	0.8616	0.8611	0.9228	17
	Genetic	85.6604	0.8566	0.8566	0.8551	0.9147	18

WrapperSubsetEval	GreedyStepwise-Forward	82.3899	0.8297	0.8239	0.8183	0.8248	3
	GreedyStepwise-Backward	81.7296	0.8166	0.8173	0.8150	0.8711	22
	Genetic	90.5031	0.9050	0.9050	0.9047	0.9606	26

Tabla A8.8: Resultados de la clasificación en rasgos de las dendritas usando MLP.

Evaluador	**M.Búsqueda**	**CC %**	**Precisión**	**Sensibilidad**	**Medida F**	**AUC**	**Rasgos**
SVMAttributeEval	Ranker	87.9560	0.8792	0.8796	0.8791	0.9498	24
CfsSubsetEval	GreedyStepwise-Forward	83.4277	0.8348	0.8343	0.8343	0.9054	17
	Genetic	84.1509	0.8410	0.8415	0.8407	0.9016	18
WrapperSubsetEval	GreedyStepwise-Forward	86.4780	0.8656	0.8648	0.8632	0.8881	5
	GreedyStepwise-Backward	79.1195	0.7909	0.7912	0.7907	0.8460	23
	Genetic	89.5283	0.8956	0.8953	0.8953	0.9623	27

Tabla A8.9: Resultados de la clasificación en rasgos de las dendritas usando SMO.

Evaluador	**M.Búsqueda**	**CC %**	**Precisión**	**Sensibilidad**	**Medida F**	**AUC**	**Rasgos**
SVMAttributeEval	Ranker	88.3333	0.8922	0.8833	0.8800	0.8591	24
CfsSubsetEval	GreedyStepwise-Forward	84.5912	0.8553	0.8459	0.8406	0.8170	17
	Genetic	84.6855	0.8560	0.8469	0.8416	0.8182	18
WrapperSubsetEval	GreedyStepwise-Forward	82.9874	0.8430	0.8299	0.8224	0.7962	7
	GreedyStepwise-Backward	78.5535	0.7994	0.7855	0.7733	0.7451	22
	Genetic	89.0252	0.8975	0.8903	0.8875	0.8681	24

Tabla A8.10: Resultados de la clasificación en rasgos de las dendritas usando IBK.

Evaluador	**M.Búsqueda**	**CC %**	**Precisión**	**Sensibilidad**	**Medida F**	**AUC**	**Rasgos**
SVMAttributeEval	Ranker	84.7799	0.8517	0.8478	0.8442	0.9074	24
CfsSubsetEval	GreedyStepwise-Forward	84.0252	0.8397	0.8403	0.8388	0.9184	17
	Genetic	84.1195	0.8427	0.8412	0.8383	0.9102	18
WrapperSubsetEval	GreedyStepwise-Forward	84.1195	0.8440	0.8412	0.8377	0.8798	6
	GreedyStepwise-Backward	79.9686	0.7984	0.7997	0.7971	0.8752	24
	Genetic	88.7107	0.8885	0.8871	0.8857	0.9261	23