UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADO EN INGENIERÍA DE LA SALUD

IDENTIFICACIÓN AUTOMÁTICA DE ORGÁNULOS CELULARES MEDIANTE REDES NEURONALES CONVOLUCIONALES DE APRENDIZAJE PROFUNDO

AUTOMATED IDENTIFICATION OF CELL ORGANELLES BY DEEP LEARNING CONVOLUTIONAL NEURAL NETWORKS

Realizado por
Carmen Aparicio Collado

Tutorizado por
Ezequiel López Rubio
Esteban José Palomo Ferrer

Departamento
Departamento de Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA

MÁLAGA, SEPTIEMBRE DE 2021

Fecha defensa: octubre de 2021

Resumen

Las proteínas en nuestro organismo son las encargadas de formar tejidos, transportar sustancias y defender al organismo contra infecciones o agentes patógenos, entre otras funciones. Conociendo la ubicación y el transporte de la proteína en el interior de la célula se puede conocer mucha información sobre su funcionalidad y los mecanismos de las enfermedades. Así, la identificación automatizada de los orgánulos celulares sirve de gran importancia para caracterizar los genes recién descubiertos o con una función desconocida. Las proteínas serían marcadas fluorescentemente para poder identificar el orgánulo donde residen y la identificación automatizada se puede realizar fácilmente con redes neuronales convolucionales mediante Inteligencia Artificial.

En este trabajo, haciendo uso de un conjunto de imágenes de microscopía de fluorescencia de células HeLa, se utilizan distintos modelos basados en redes neuronales convolucionales para identificar y clasificar los distintos orgánulos celulares. Asimismo, se presenta la precisión y el error de cada modelo y se elige el modelo más acorde a solucionar este problema.

Palabras clave: redes neuronales convolucionales, Inteligencia Artificial, aprendizaje computacional, aprendizaje profundo, transfer learning, orgánulos celulares, proteínas, PyTorch.

Abstract

Proteins in our body are responsible for composing tissues, transporting substances and defending the body against infections or pathogens, among other functions. By knowing the location and transport of the protein inside the cell, much information can be learned about its functionality and the mechanisms of disease. Thus, automated identification of cellular organelles serves of great importance to characterize newly discovered genes or genes with unknown function. Proteins would be fluorescently tagged in order to identify the organelle where they reside and automated identification can be easily performed with convolutional neural networks using Artificial Intelligence.

In this research, using a set of fluorescence microscopy images of HeLa cells, different models based on convolutional neural networks are used to identify and classify the different cell organelles. Also, the accuracy and error of each model is presented and the most appropriate model to solve this problem is chosen.

Keywords: Convolutional Neural Networks, Artificial Intelligence, computational learning, deep learning, transfer learning, cell organelles, proteins, PyTorch.

Índice

1

Introducción

Según la teoría celular, la célula es la unidad estructural y funcional de los seres vivos [1]. Todas las células cuentan con una estructura común que se trata de membrana plasmática, citoplasma y material genético. Existen dos grandes clases celulares: las células procariotas, que carecen de núcleo y no poseen orgánulos membranosos entre otras características, y las células eucariotas, mucho más evolucionadas y que cuentan con un núcleo, donde se encuentra la información genética, y orgánulos membranosos.

La estructura interna de una célula es compleja y está altamente organizada, dividiéndose en compartimentos u orgánulos celulares que desarrollan funciones específicas. Consideraremos compartimento celular al espacio, separado o no por una membrana, en el que tienen lugar las actividades necesarias o primordiales de la célula. Los orgánulos están distribuidos por todo el espacio celular, estando éste englobado por la membrana plasmática que separa el medio interno de la célula del externo.

La célula eucariota presenta un mayor grado de organización estructural que la procariota, y cuenta con orgánulos rodeados por membranas. El más característico es el núcleo, el cual posee una doble membrana o envoltura nuclear, que alberga en su

interior la información genética. El medio intracelular, situado entre la membrana plasmática y el núcleo, está formado por una solución líquida denominada hialoplasma o citosol. Aquí, se encuentran el resto de orgánulos celulares. El conjunto formado por el citosol y todos los orgánulos recibe el nombre de citoplasma.

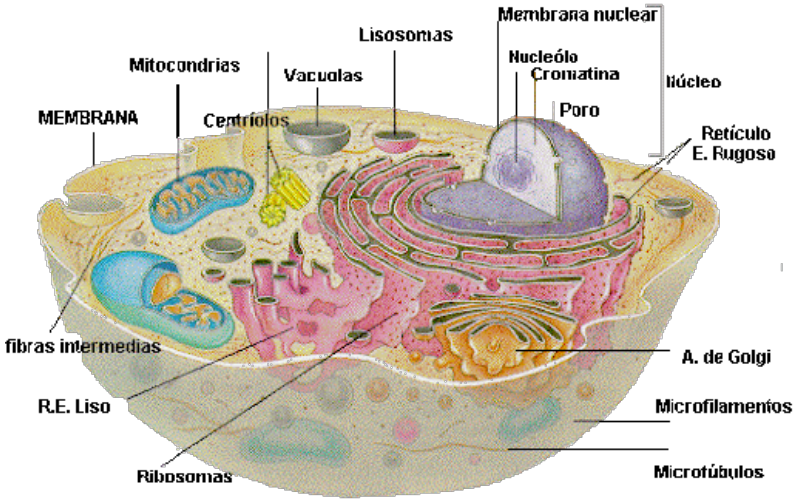


Figura 1. *Estructura de una célula eucariota.* [2]

En cada célula hay miles de proteínas diferentes. Éstas se ubican en regiones específicas de la membrana plasmática, en membranas de distintos orgánulos o dentro de ellos, en el citosol de forma fija o móvil, en su camino hacia o desde el núcleo celular, y, por último, proteínas cuyo fin es exportarlas de la célula. Las proteínas son moléculas de una gran diversidad estructural, lo que les confiere la capacidad de intervenir en numerosas y diversas funciones, pudiendo una misma proteína realizar más de una función. Estas moléculas son las encargadas de dar forma y estructura a las células y llevan a cabo la mayoría de procesos vitales. Sus funciones son específicas de cada proteína y permiten que la célula mantenga su equilibrio interno, se defienda de factores externos, repare el daño, controle y regule funciones como el metabolismo y la reproducción, entre otras. Por lo tanto, tanto la síntesis de proteínas como su transporte son actividades esenciales para la célula [3].

La ubicación subcelular de una proteína se refiere a la localización específica de un producto génico o una proteína en regiones complejas de la célula. Numerosos estudios han demostrado que la localización celular de una proteína está estrechamente relacionada con su función [4], por lo tanto, comprender la ubicación en la célula de una proteína es primordial para el entendimiento completo de su función. Toda esta información sobre la localización de las proteínas debe ser recogida y ordenada en una

base de datos que comprenda todo el proteoma. Este es el objetivo principal de la proteómica, es decir, establecer el patrón de localización de todas o casi todas las proteínas expresadas en un tipo concreto de célula [5].

Hoy en día, una de las técnicas más utilizadas para identificar la ubicación en la célula de las proteínas es a partir de imágenes de microscopía de fluorescencia. El inconveniente es que esto genera una gran cantidad de datos, y analizarlas una a una de forma convencional es lento y poco satisfactorio. Por esto, clasificar las imágenes resultantes de forma más rápida, eficiente y automática, es necesario.

La identificación automatizada de los orgánulos celulares tiene un papel importante a la hora de caracterizar genes que han sido recientemente descubiertos o que tienen una función desconocida. Las proteínas producidas por cualquier gen pueden ser marcadas con fluorescencia y así, es posible identificar el orgánulo donde residen proporcionándonos información sobre su posible función. Asimismo, si se analizan las diferencias de la localización de las proteínas y su transporte entre distintos estados como sano o varias etapas de una enfermedad nos puede aportar bastante información sobre los mecanismos de una enfermedad. Esto, es bastante útil para el nuevo campo de la citómica, cuyo objetivo es caracterizar la variación de comportamiento de los diferentes tipos celulares y la relación que estos tienen con las enfermedades [6].

La Inteligencia Artificial juega un papel importante en esto, pues es capaz de reconocer imágenes con bastante precisión, de forma más rápida y automática. Esto es en parte debido al desarrollo del Deep Learning, que usa redes neuronales artificiales basadas en el funcionamiento de las neuronas del cerebro humano. Esta técnica hace posible la identificación de patrones en los datos usando algoritmos que se encargan de identificar y clasificar las características de la imagen aprendiendo y mejorando el proceso de aprendizaje con cada ejecución. Para esto, se hace uso de las redes neuronales convolucionales (CNN) que han revolucionado el reconocimiento de imágenes, pues usan un procesamiento relativamente pequeño. El funcionamiento de este tipo de redes consiste en que la red es entrenada introduciéndole imágenes de las cuales solo usará los píxeles y las etiquetas, se le aplicarán filtros capaces de reconocer patrones y esta red aprenderá [7].

Este método nos aporta numerosas ventajas frente a los métodos convencionales de identificación pues nos permitirá la identificación automatizada de los orgánulos aumentando la precisión diagnóstica, en un menor tiempo dado a la gran cantidad de datos generados y con un menor coste. Esto permitirá obtener información sobre la función de las proteínas y obtener información sobre las enfermedades, pudiendo dar lugar a una medicina más personalizada.

2

Objetivos

El objetivo a conseguir en este trabajo consiste en la identificación automática de orgánulos celulares y su clasificación en 10 clases mediante redes neuronales convolucionales de aprendizaje profundo.

Estas clases representan los orgánulos celulares presentes en las células HeLa, que son un tipo particular que se usa para cultivo celular en investigación científica. Las clases, con el término con el que nos referiremos a ellas, son las siguientes: Actina (actin), Núcleo (DNA), Endosoma (endosome), Retículo endoplasmático (er), Giantin que forma parte de la región cis/medial del Aparato de Golgi (golgia), GPP130 región cis del Aparato de Golgi (golgpp), Lisosoma (lysosome), Tubulina (microtubules), Mitocondria (mitochondria) y Nucléolo (nucleolus). Cada imagen de microscopía de fluorescencia será identificada y clasificada en sólo una de estas 10 clases.

En este trabajo hemos centrado nuestra atención en la capacidad que tiene nuestro modelo diseñado para poder identificar correctamente las imágenes proporcionadas. Esto presenta dos dificultades añadidas que es importante mencionarlas.

La primera es que los expertos humanos encuentran especialmente complicado distinguir los endosomas y lisosomas. Ambos forman parte de un sistema de vesículas que participan en el tráfico vesicular. Pero, la principal diferencia se basa en su formación y función en la célula. El endosoma es formado a partir de la endocitosis, y, el lisosoma es una vesícula que contiene en su interior enzimas hidrolíticas degradantes [8].

La segunda, la encontramos en la dificultad para diferenciar las dos proteínas del aparato de Golgi en el conjunto de datos puesto que ambas se encuentran en la región cis.

Para resolver este problema y conseguir nuestro objetivo se hará uso de redes neuronales convolucionales capaces de identificar los orgánulos pertenecientes a cada una de las clases mencionadas anteriormente. Se creará una CNN y evaluaremos si es capaz de resolver el problema planteado y su rendimiento. Posteriormente, se probará con otras redes pre-entrenadas pero adaptándolas al problema planteado y comprobaremos si pueden identificar orgánulos celulares con precisión y buenos resultados.

3

Contexto

3.1 *Deep Learning* o Aprendizaje Profundo en el marco de la Inteligencia Artificial

Una definición simple sobre qué es la Inteligencia Artificial (IA) sería la habilidad que tienen los ordenadores para realizar tareas que generalmente requerirían inteligencia humana. Es decir, es la capacidad de las máquinas para utilizar algoritmos, adquirir conocimiento de los datos y usar ese aprendizaje en la toma de decisiones como lo haría una persona [9].

Uno de los aprendizajes en los que se centra la Inteligencia Artificial es el Aprendizaje Automático (también conocido como *Machine Learning*). Este consiste en la capacidad que tienen los dispositivos tecnológicos para aprender automáticamente sin estar programados para ello. Esto quiere decir que es capaza de identificar patrones complejos basados en datos y hacer sus predicciones [10].

El *Machine Learning* podemos dividirlo en tres subconjuntos: aprendizaje supervisado, no supervisado y de refuerzo. En el **aprendizaje supervisado**, los algoritmos utilizan datos organizados con anterioridad para indicar cómo clasificar la

nueva información siendo supervisado por una persona. En el **aprendizaje no supervisado**, al contrario que el anterior, los algoritmos tienen que buscar la forma de clasificar esa información por sí solos. Para acabar, el **aprendizaje por refuerzo**, a los algoritmos cada vez que aciertan se les da un refuerzo positivo, por lo que aprenden de la experiencia [9].

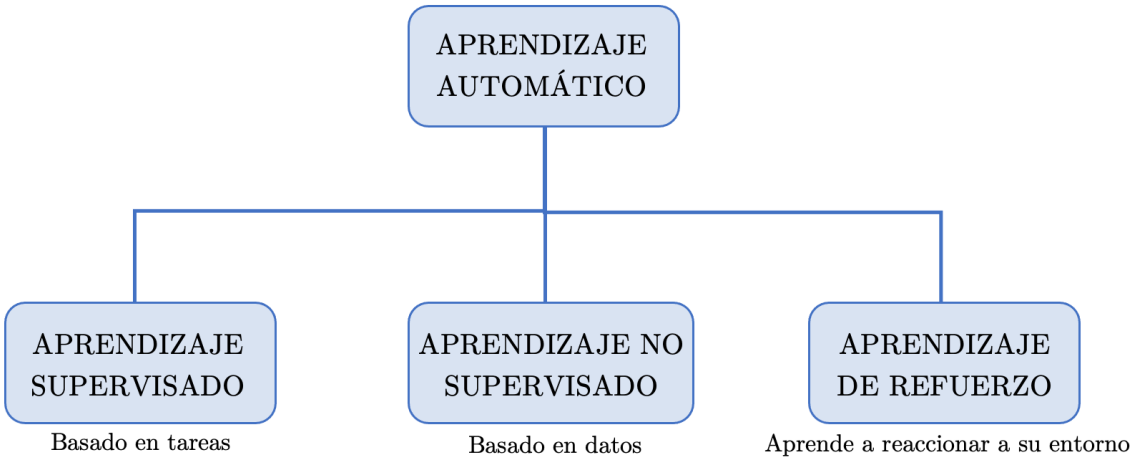


Figura 2. *Tipos de aprendizaje automático.*

Dentro de la Inteligencia Artificial, uno de los campos más relevantes en el panorama actual y de mayor crecimiento, es el Aprendizaje Profundo (o *Deep Learning*). Es un subcampo que forma parte del Aprendizaje Automático, y que se encarga de resolver problemas bastante complejos y que generalmente contienen una gran cantidad de datos.

En *Deep Learning* se utilizan estructuras lógicas que intentan simular el comportamiento del cerebro humano permitiéndole aprender de grandes cantidades de datos. Estas estructuras se denominan redes neuronales (o *neural networks*) y se encuentran conectadas entre sí y organizadas por capas, simulando así la forma en la que el cerebro humano trabaja mediante las sinapsis neuronales. Cada neurona recibe información de la anterior que se encuentra en otra capa previa, la procesa y la distribuye a las neuronas de la próxima capa (figura 3), refinando y optimizando así en cada neurona las predicciones. Esta progresión de computaciones a través de la red se denomina propagación *feed forward*.

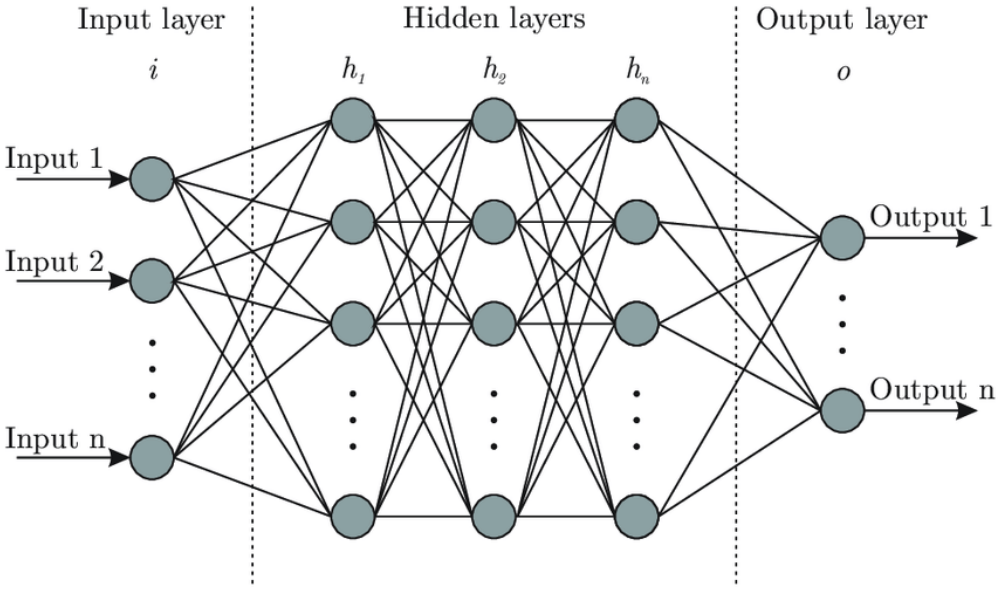


Figura 3. *Red Neuronal Artificial.* [11]

En las redes neuronales profundas tanto la capa de entrada como la de salida se denominan capas visibles. La capa de entrada es donde se introduces los datos, a lo largo de la red estos se procesan, y en la capa de salida se llevan a cabo las predicciones y el fin de la clasificación.

Después tiene lugar otro proceso, llamado *back propagation* que usa algoritmos como el gradiente de descenso, el cual se encarga de calcular los errores y ajustar los pesos y los sesgos o *bias* moviéndose a lo largo de la red hacia atrás para así, poder entrenar el modelo neuronal.

Estos dos procesos juntos permiten que la red neuronal haga predicciones y según los resultados obtenidos, corrija los errores. Por lo tanto, conforme se vaya entrenando se volverá progresivamente más preciso [12].

Las neuronas manejan una operación lineal, calculan una suma de pesos de su entrada, y le añaden un sesgo. Esta operación es seguida de una función de activación que no es lineal cuya función es decidir si una neurona debe ser activada o no, permitiendo que la salida tenga diferentes pendientes en diferentes valores, y en la última capa de la red, tiene el fin de concentrar las salidas de las operaciones lineales predichas en un determinado rango.

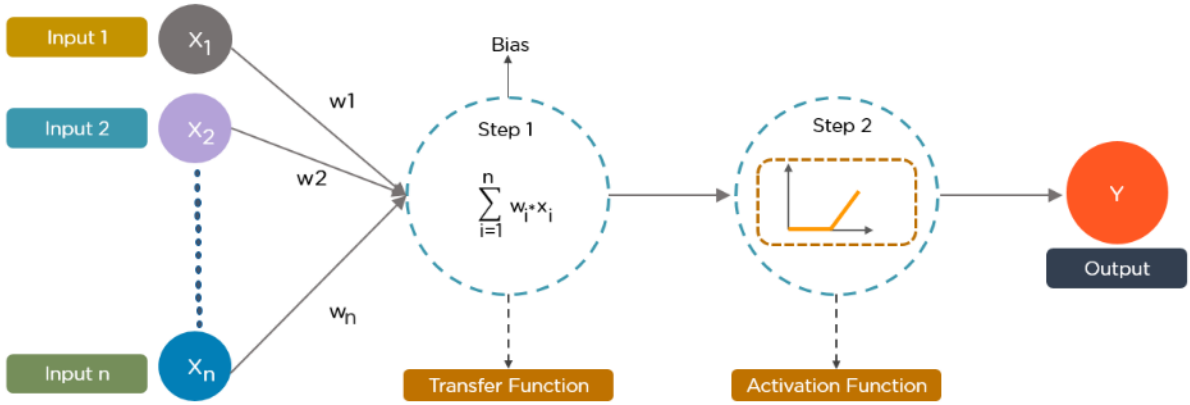


Figura 4. *Funcionamiento en una neurona.* [13]

Las funciones de activación más empleadas son *ReLU* que es actualmente considerada la más eficaz, y la función *Sigmoid*, también conocida como función logística, que se usa cuando la salida debería ser una probabilidad [14].

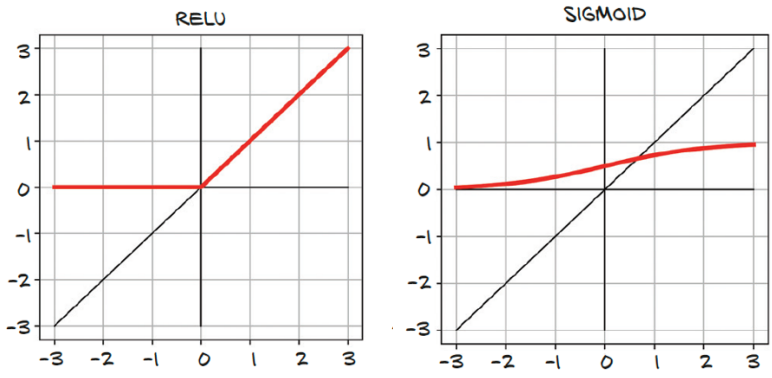


Figura 5. *Funciones de activación comunes.* [14]

Para medir la precisión del funcionamiento en cada red neuronal se usará una función de pérdida. Esta se encarga de medir el desacuerdo entre la salida deseada y la salida real. Además, se usará un optimizador denominado función estocástica de gradiente de descenso, que actualizará los parámetros entrenables de tal manera que van a modificarse en dirección opuesta de la función de pérdida [15].

Por lo tanto, un resumen del proceso de funcionamiento de la red sería: los datos de entrenamiento se introducen en la red, cada neurona hace una predicción sumando los pesos y la *bias* y se les aplica una función de activación produciendo una salida. Tras esto, se procede al *back propagation* donde se usará la función de pérdida para el error entre las predicciones y las salidas reales, el optimizador actualizará los parámetros y, por último, se modificarán los pesos. Este proceso será repetido hasta que nuestra red no pueda mejorar.

En este trabajo usaremos Aprendizaje profundo, tal y como ha sido descrito arriba, y para ello usaremos redes neuronales convolucionales. Las redes neuronales convolucionales serán definidas en el siguiente apartado.

3.2 Redes Neuronales Convolucionales (CNN)

Una red neuronal convolucional (CNN) es un tipo de red neuronal artificial con aprendizaje supervisado donde la red procesa los datos imitando de una manera muy similar a la corteza visual primaria del cerebro humano. Así, están diseñadas principalmente para trabajar con imágenes y señales. La CNN está organizada en capas, conteniendo varias capas ocultas que están especializadas y que tienen una jerarquía. De esta forma, las primeras capas irán detectando líneas y curvas, y se irán especializando hasta que sean capaces de reconocer formas complejas en las capas más avanzadas. En este tipo de red tienen lugar operaciones de convolución que consisten en filtrar una imagen usando una máscara. Esta máscara se denomina *kernel* y se aplica a una imagen para poder extraer características o patrones que presente la imagen. Así, cada pixel de salida será una combinación lineal de los píxeles de entrada [16].

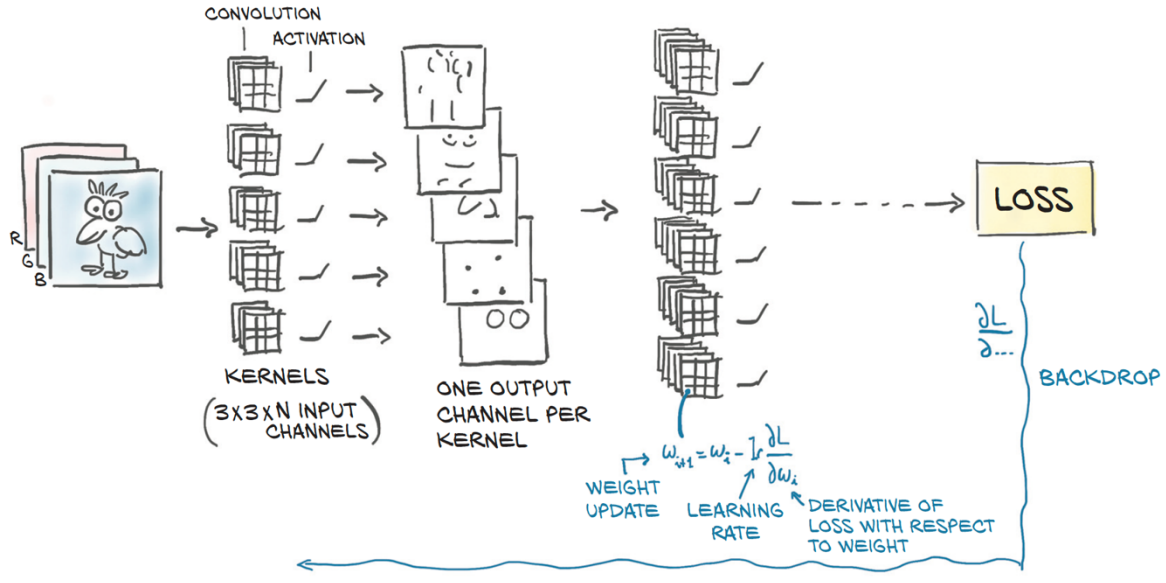


Figura 6. *Funcionamiento de una CNN.* [14]

3.2.1 Capas Convolucionales

En las capas convolucionales es en donde tienen lugar las operaciones de convolución. En ellas, su salida es el resultado de una operación de convolución entre una entrada y un *kernel*. Esta operación implica tomar un conjunto de píxeles cercanos de la imagen de entrada y usando el *kernel* opera calculando el producto escalar. Los elementos del *kernel* serán los pesos entrenables de cada capa. Así, el *kernel* actúa sobre todas las neuronas de entrada y nos permite obtener una nueva matriz, que pasará a ser una de las capas ocultas de nuestra red. Para las imágenes de colores tenemos 3 características de entrada por píxel (los canales RGB) por lo que se usarán *kernels* del mismo tamaño e iguales en todas las direcciones. Después del operador de convolución, se suele aplicar elemento a elemento una función de activación.

De esta forma, la neurona también recibirá información de los píxeles vecinos, pudiendo obtener así patrones locales. La función de cada filtro es resaltar una característica específica de la imagen, por lo conforme vayan avanzando las capas se buscarán características más complejas en función de lo que examinó la capa anterior [17].

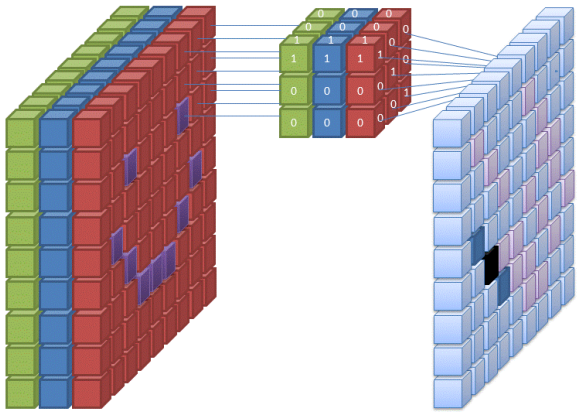


Figura 7. *Capa convolucional 3D.* [18]

3.2.2 Operación *Pooling*

La operación de *Pooling* consiste en reducir las muestras de una imagen. De esta forma, se reduce la información y el tamaño de una imagen. La capa de *Pooling* más común es de tamaño 2x2, así escalar una imagen a la mitad, es equivalente a coger 4 píxeles vecinos como entrada produciendo un pixel como salida. Tenemos tres formas de reducir la imagen:

- **Promedio de cuatro píxeles.** Hace una media de cuatro píxeles. Esto se denomina *average pooling*.
- **Tomar el máximo de cuatro píxeles.** Esta es la aproximación más usada actualmente. Se denomina *max pooling* y toma el píxel de mayor peso.
- **Realizar una convolución *strided*, en la que sólo se calcula cada enésimo píxel.**

Nosotros nos centraremos en utilizar *max pooling*.

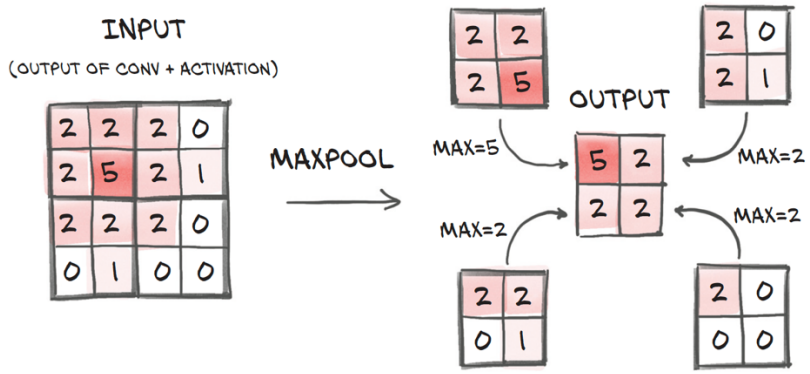


Figura 8. *Max pooling.* [14]

Las capas de *pooling* son utilizadas cuando queremos descubrir una característica, pero no nos importa la localización exacta de la característica si no la característica en sí [14].

Por lo tanto, nuestra imagen de salida será menor que la entrada como resultado tanto de las capas convolucionales como de la operación de *pooling*, descartando la información que no nos interesa y quedándose con las características que serán

determinantes a la hora de hacer predicciones. Esto es lo que hace de las CNN una gran herramienta para el análisis de imágenes.

Para evitar que la imagen se reduzca después de una convolución aplicaremos la función de *padding*, que consiste en crear píxeles fantasmas alrededor del borde de la imagen que tendrán un valor 0.

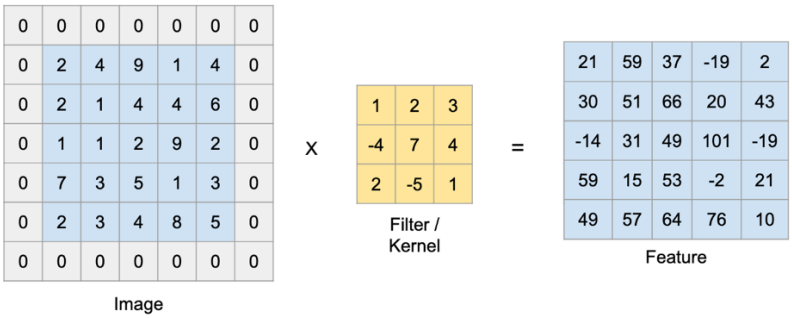


Figura 9. *Operación de padding.* [19]

3.2.3 *Flattening*

Una vez que las operaciones anteriores han sido realizadas y hemos obtenido un mapa de características, el siguiente paso es aplanarlo. Esta operación es llamada *flattening* y consiste en coger la información una a una por fila de la matriz del mapa de características y agruparla en una sola columna. El objetivo es introducirlo posteriormente en la red neuronal artificial explicada anteriormente para su posterior procesamiento. Estas capas finalmente se encargarán de clasificar la imagen con toda la información y características recogidas en las capas convolucionales [20].

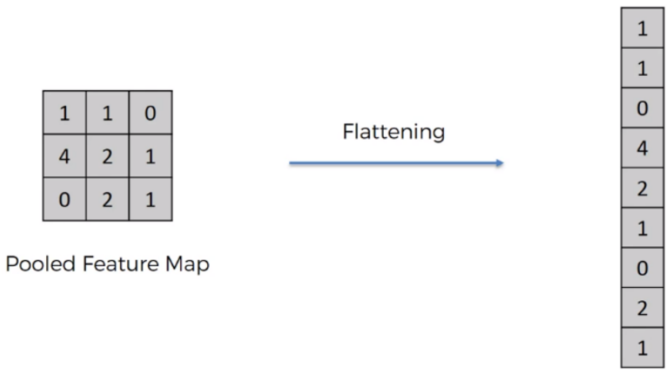


Figura 10. *Flattening.* [21]

3.2.4 Capa totalmente conectada o *Fully connected layer (FC)*

Después del *flattening,* el mapa de características aplanado pasa a través de una red neuronal artificial completa. El objetivo principal de esta red es combinar nuestras características con más atributos con el fin de predecir las clases aún mejor.

Este proceso esta compuesto por tres capas, la de entrada, la totalmente conectada y la de salida. La capa totalmente conectada se asemeja a la capa oculta de las redes neuronales artificiales, con la diferencia de que esta está completamente conectada. Conforme la información pasa a través de la red, se calcula el error de predicción. Para mejorar la predicción se usa *backpropagation.* Y en la capa de salida obtendremos las clases predichas. Se obtendrán tantas salidas como clases haya [20].

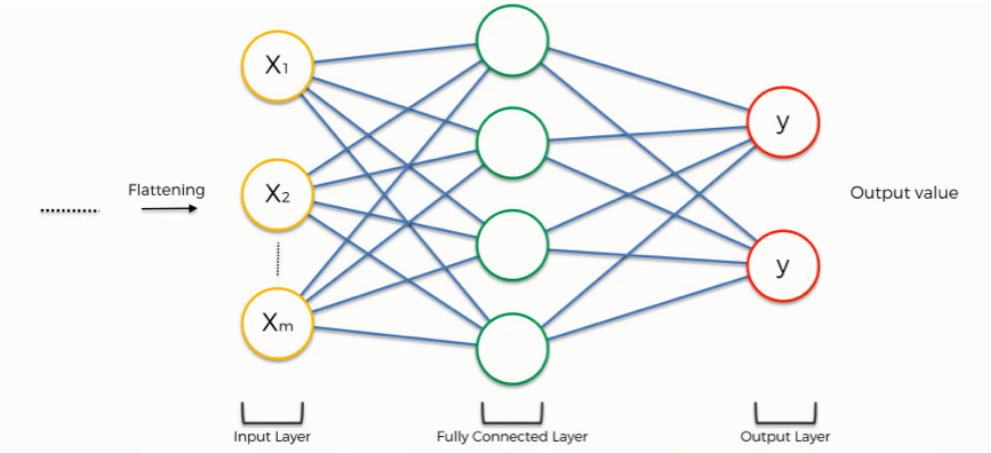


Figura 11. *Red neuronal artifical con FC.* [22]

Para acabar, a la última capa se le aplica la función *Softmax.* Es importante que los resultados finales sumen uno, pero esto es difícil de conseguir. Por lo que se aplica esta función cuyo fin es que estas cifras se reduzcan a números entre cero y uno. Esto nos permitirá obtener la predicción de cada clase y la imagen será clasificada con la clase que tenga una mayor predicción.

$$Softmax\ (x_i) = \frac{e^{x_i}}{\sum_{j=1}^{C} e^{x_j}}$$

Donde C es el número de clases e i el índice de la clase.

4

Estado del arte

4.1 Inteligencia Artificial en Biotecnología y Salud

En la actualidad, la Inteligencia Artificial tiene varias aplicaciones en nuestra sociedad. Hoy en día se ha convertido en un campo indispensable ya que es capaz de resolver problemas complejos de forma efectiva. Su aplicación está presente en numerosas industrias como la salud, finanzas, educación, ciberseguridad, entretenimiento, transportes, entre otras. En definitiva, la IA hace que nuestra vida sea más cómoda y rápida.

La IA también se está haciendo un hueco en el campo de la biotecnología participando en numerosas investigaciones. La biotecnología como su propio nombre indica es la aplicación de la tecnología en la biología. Este campo ha avanzado mucho en los últimos años debido a los avances de la biología, pero principalmente por los progresos de la Inteligencia Artificial y su ayuda a los investigadores. Esta técnica proporciona una recopilación de datos más rápida y precisa en las investigaciones. Esto está sustituyendo a los métodos antiguos, como el análisis estadístico clásico o la clasificación de imágenes manual, pues estos son poco prácticos.

Estas investigaciones también ayudan a que haya grandes avances en otros campos como es la biomedicina. Mediante la utilización de la IA, se puede aumentar la productividad de la investigación biomédica, obteniendo resultados más eficientes y precisos. Gracias a esto, estas industrias están modificando la gestión de sus procesos, tanto en la aparición de nuevos productos como servicios. Las innovaciones más relevantes son el desarrollo de medicamentos y una medicina personalizada.

Esto está sentando las bases para el desarrollo de una nueva medicina más precisa que va a revolucionar la atención sanitaria. Algunas de las formas en las que la Inteligencia Artificial está ayudando a estos campos son mejora de la precisión, descubrimiento de medicamentos, medicina personalizada, edición de genes, desarrollo de nuevas herramientas de radiología, análisis más precisos para imágenes de patologías, entre otros [23].

A continuación desarrollaré como la IA puede ayudar en áreas relacionadas con este trabajo:

La Inteligencia Artificial es usada para el entendimiento de enfermedades. Ayudando a identificar el endotipo de enfermedades, es decir, los mecanismos moleculares que separan enfermedades en diferente grupos. Entender cómo actúa una enfermedad y su forma de manifestarse es esencial para poder estudiar y determinar las causas y los mecanismos de una enfermedad. Así, los investigadores serán capaces de desarrollar nuevos tratamientos. La IA puede ayudar a analizar la gran cantidad de datos a los que se tiene acceso, proporcionando nuevas hipótesis y generando más posibles objetivos farmacológicos.

Además, también puede ayudar a agrupar enfermedades estudiando los mecanismos que las causan. En la mayoría de ocasiones, el fenotipo de una enfermedad influye en el diagnóstico médico pero estos mecanismos pueden ser más individualizados para cada persona, independientemente del fenotipo. Aquí la IA ayuda a identificar mecanismos comunes biológicos que causan la enfermedad. Investigando estas causas se identifican patrones identificando vías y rutas de señalización así como se hace uso de la genómica, transcriptómica, proteómica y metabolómica y con todos los datos generados IA ayuda analizándolos de una forma más específica y precisa. Una

vez entendida la enfermedad, el siguiente paso es descubrir y desarrollar nuevos fármacos que actúen con el sistema que está causando la enfermedad.

La Inteligencia Artificial también puede ayudar a editar los genes. A medida que las técnicas se refinan y se mejoran los modelos la ciencia será capaz de comprender mejor el genoma humano y esto les llevará a editar los genes. Esto se ha hecho ya con técnicas como el CRISPR que tiene un papel significante en el descubrimiento de medicamentos. Esta técnica y la IA nos acercará a una medicina personalizada y hará posible identificar enfermedades hereditarias que estén codificadas en nuestros genes aunque no se hayan manifestado todavía, y de esta forma, tratarlas antes de que se produzcan [24].

En conclusión, la IA ofrece numerosas ventajas frente a las técnicas de análisis tradicional y frente a la toma de decisiones clínica. Los algoritmos de aprendizaje pueden ser más precisos permitiendo importantes avances en la medicina, y, consecuentemente, mejorarán la vida de muchas personas.

4.2 Tecnología a utilizar: PyTorch y Google Colab

Este trabajo se ha desarrollado en Python con la librería de PyTorch. PyTorch es un paquete de Python el cual utiliza la programación de tensores con el fin de realizar cálculos numéricos en grandes cantidades. Para hacer más rápidos los cálculos, esta librería nos permite ejecutarlos en GPU. Normalmente se utiliza en Machine Learning, para el desarrollo de redes neuronales. La interfaz que nos proporciona es muy sencilla y la principal ventaja es que trabaja con grafos dinámicos permitiéndonos que durante el tiempo de ejecución la modificación de las funciones y la variación del gradiente sea posible.

También, se ha usado el entorno de Google Colab, una herramienta de Google Research, que permite escribir y ejecutar código en Python sin ninguna configuración y con acceso gratuito a GPU's de gran capacidad ubicadas en la nube. Esto es necesario puesto a la gran cantidad de datos y cálculos que se realizan para poder entrenar un modelo como este.

PyTorch junto con Google Colab permitirán la ejecución en tarjetas gráficas, en concreto CUDA, que es una API que hace posible la conexión entre la CPU y la GPU. Esta ha sido desarrollado por NVIDIA.

4.3 Orgánulos del problema

Dentro de nuestro dataset nos encontramos ADN (DNA) y el nucléolo (nucleolus), ambos forman parte del núcleo celular. El núcleo es un orgánulo esférico u ovalado, presente en casi todas las células eucariotas (los eritrocitos lo pierden en su maduración) que se localiza en una posición central, aunque es más o menos móvil. Su función es contener el material genético en forma de ADN, y aquí tiene lugar la replicación del ADN y la síntesis del ARN. Este se encuentra separado del citoplasma por la envoltura nuclear, y en su interior se encuentra la cromatina, que es el ADN asociado a proteínas, y el nucleoplasma. Dentro del nucleoplasma se encuentra el nucléolo, que es un orgánulo más o menos redondo compuesto por ARN y proteínas y, generalmente, está localizado próximo a la envoltura nuclear. Su principal función es la síntesis del ARNr (ribosómico) y el procesado y empaquetamiento de ribosomas, que posteriormente se exportan al citosol. Es indispensable para el desarrollo de la mitosis.

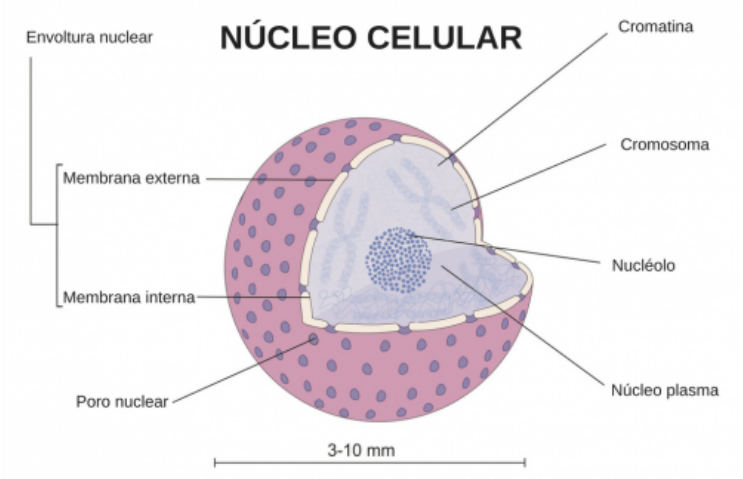


Figura 12. *Estructura del núcleo.* [25]

Posteriormente, en nuestro dataset aparece el retículo endoplasmático (er) que es un sistema membranoso extendido entre la membrana plasmática y la membrana nuclear. Como es una continuación de estas dos membranas, consecuentemente divide el citoplasma en dos compartimentos, el espacio luminal, en el interior, y el espacio citosólico, en el exterior. El retículo endoplasmático está formado por dos compartimentos interconectados, el retículo endoplasmático rugoso (RER) y el retículo

endoplasmático liso (REL). El RER tiene adheridos ribosomas y está constituido por sacos aplanados o cisternas y vesículas de diferente tamaño. Su principal función es la síntesis y almacenamiento de proteínas. El REL es una red tubular que continúa al RER, pero sin ribosomas adheridos. Su principal función es la síntesis de lípidos, entre otras.

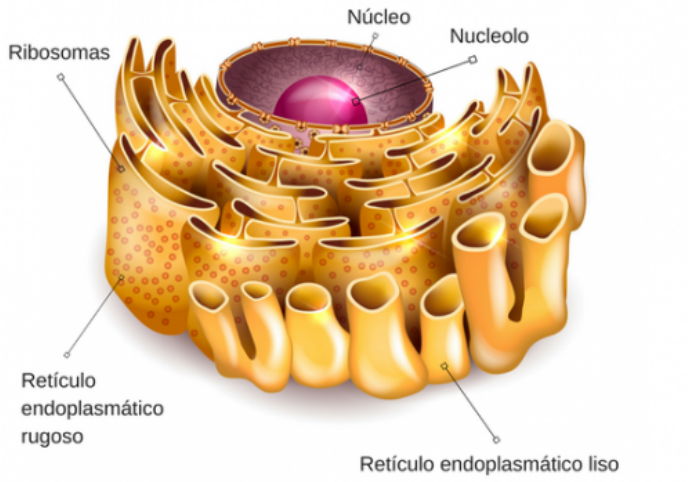


Figura 13. *Estructura del retículo endoplasmático.* [26]

Junto con los anteriores orgánulos, también nos encontramos dos proteínas que forman parte del aparato de Golgi. Este es un sistema de endomembranas constituido por una o varias unidades llamadas dictiosomas, que forman un sistema membranoso por la agrupación de cisternas y vesículas asociadas. Sus principales funciones son mecanismo de transporte, glucosilación de lípidos y proteínas, entre otros. Las dos proteínas que nos encontramos son la Gigantina (golgia) y la GPP130 (golgpp). La gigantina se encuentra en los bordes del aparato de Golgi cis/medial y forma parte de la región responsable del tráfico de membranas en la vía secretora de proteínas. La GPP130 o Golgi fosfoproteína 4 que se encuentra en la región cis.

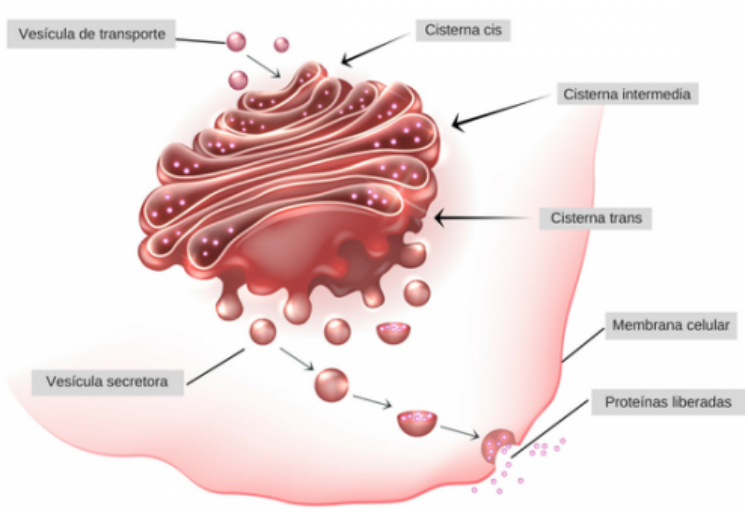


Figura 14. *Estructura del aparato de Golgi.* [27]

Otro orgánulo que aparece es el lisosoma (Lysosome), que es un orgánulo membranoso que tiene en su interior enzimas hidrolíticas que intervienen en la digestión celular. Degradan el material que captan y lo expulsan bien por fagocitosis o pinocitosis. Tenemos dos tipos de lisosomas, los primarios que son vesículas provenientes del aparato de Golgi y secundarios que se forman por la unión de un lisosoma a un endosoma. El endosoma (endosome) también es otro orgánulo que aparece en el dataset, que es una vesícula formada por invaginación de la membrana celular para captar partículas del medio externo que posteriormente y con ayuda de los lisosomas serán digeridas. [8]

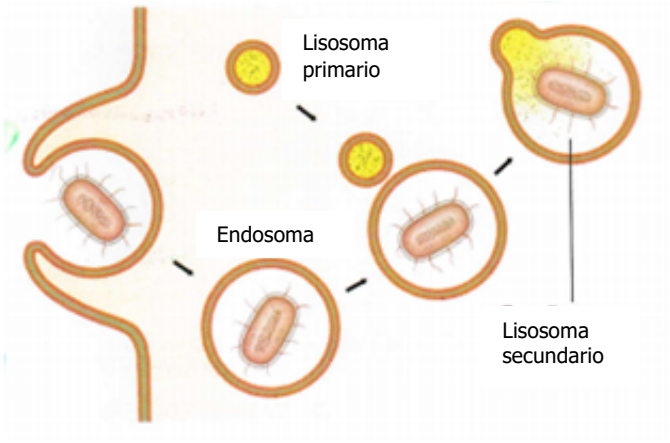


Figura 15. *Endosomas y lisosomas.* [28]

El siguiente orgánulo membranoso perteneciente al HeLa dataset es la mitocondria. Esta está presente en todas las células eucariotas y en un numero muy elevado. Tiene forma de cilindro alargado y su principal función es la respiración celular. Está compuesta por una membrana mitocondrial tanto externa como interna, un espacio intermembranoso y una matriz mitocondrial. Cada compartimento está encargado de una función, así, en la matriz mitocondrial se desarrolla el ciclo de Krebs y la β-oxidación de los ácidos grasos, en la membrana interna se desarrolla la cadena respiratoria y la fosforilación oxidativa tiene lugar en las crestas mitocondriales.

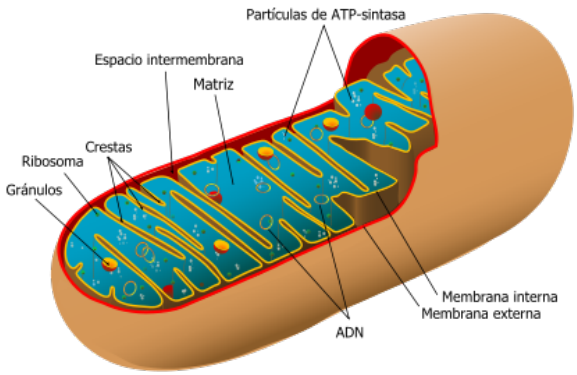


Figura 16. *Estructura de la mitocondria.* [29]

Por último, encontramos el citoesqueleto que es el conjunto de filamentos proteicos situados en el citosol y que contribuyen a la morfología celular, a la organización interna y al movimiento celular. Este está compuesto por microfilamentos de actina, filamentos intermedios y microtúbulos. En nuestro dataset nos encontramos con dos componentes que pertenecen a él, estos son los microfilamentos de actina (actin) y los microtúbulos (microtubules).

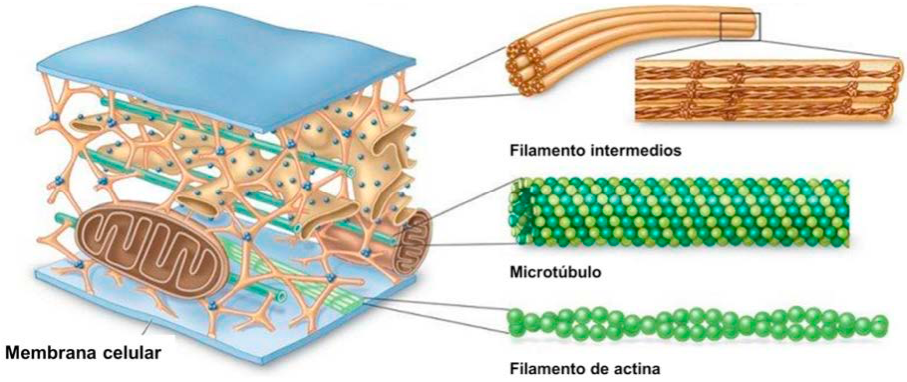


Figura 17. *Estructura del citoesqueleto.* [30]

Los microfilamentos de actina forman parte del citoesqueleto y se disponen formando una hélice compuesta por dos hebras. Estas son esenciales para el desarrollo del movimiento las células. Estos filamentos tienen extremos de diferente polaridad, y son capaces de polimerizarse y despolimerizarse fácilmente. Las funciones de estos microfilamentos son la contracción muscular, la formación del esqueleto mecánico de las microvellosidades, la citocinesis celular (división celular del citoplasma) y el movimiento ameboide.

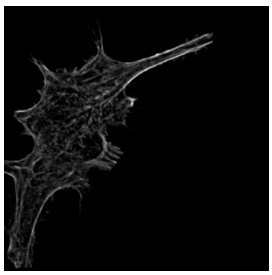
Los microtúbulos forman también parte del citoesqueleto y son formaciones cilíndricas, uniformes y rectilíneas, su localización es dispersas por el citoplasma o constituyen cilios, flagelos y centriolos. Son estructuras dinámicas, ya que se pueden formar o destruir según las necesidades que presente la célula. A nivel bioquímico, las proteínas que lo forman se llaman tubulina. Las principales funciones de los microtúbulos son la formación del huso mitótico (se encarga del movimiento de los cromosomas durante la división celular), transporte intracelular y movimiento celular. [3] [31]

5

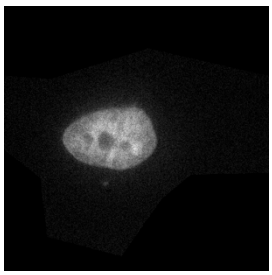
Datasets

En este trabajo hemos usado un dataset llamado *2D HeLa*, que consiste en un conjunto de imágenes de microscopía de fluorescencia de células HeLa mostrando distintos tipos de orgánulos. Para la creación de este conjunto de imágenes, estas células han sido teñidas con varios colorantes fluorescentes específicos para cada orgánulo. Las células HeLa son un tipo particular de células usadas con mucha frecuencia para cultivo celular en investigación científica.

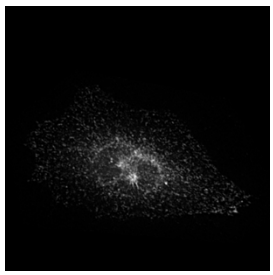
A continuación se muestran ejemplares de imágenes para cada clase:



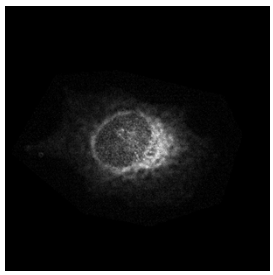
(1) Actin



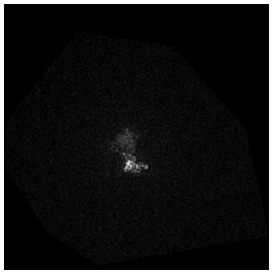
(2) DNA



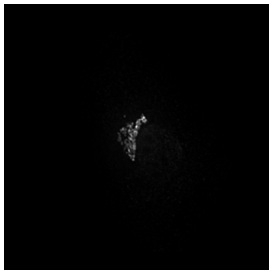
(3) Endosome



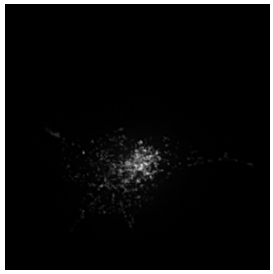
(4) ER



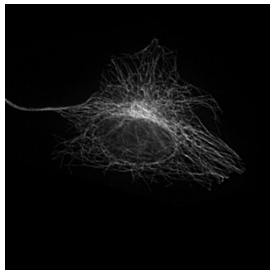
(5) Golgia



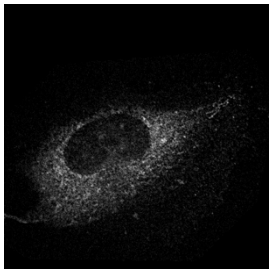
(6) Golgpp



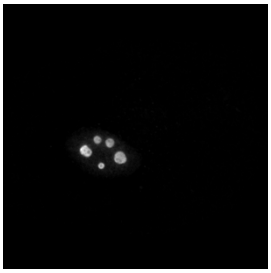
(7) Lysosome



(8) Microtubules



(9) Mitochondria



(10) Nucleolus

Nuestro dataset consta de 862 imágenes que será divididas en tres conjuntos. Estos conjunto serán de entrenamiento, validación y testeo. Cada conjunto contará con imágenes de cada clase ordenadas aleatoriamente. Así, el conjunto que usaremos para entrenar la red contará con el 60% de los datos. Cada vez que estos datos se entrenen, usaremos el 15% de los datos como conjunto de validación que servirá para comprobar tanto el error como la precisión durante el entrenamiento y, a su vez, servirá para hacer comparaciones con el conjunto de entrenamiento.

Además, una vez que el modelo esté completamente entrenado, usando los conjuntos anteriores, pasaremos a evaluar el modelo usando un conjunto de prueba. Este conjunto estará formado por el 25% de las imágenes y nos servirá para hacer predicciones y poderlas comparar con los datos reales viendo si el entrenamiento ha funcionado o no.

En la siguiente tabla se muestra el número de orgánulos con los que cuenta cada clase:

HeLa dataset			
Actin	98	**Golgpp**	85
DNA	87	**Lysosome**	84
Endosome	91	**Microtubulues**	91
ER	86	**Mitochondria**	73
Golgia	87	**Nucleolus**	80

Tabla 1. *Distribución de imágenes por clase.*

6

Metodología

6.1 Introducción de datos y preprocesado

A medida que introducimos las imágenes de nuestro dataset a estas se les irá aplicando distintas transformaciones. Se les aplicará un conjunto de transformaciones que son las siguientes:

- Reajuste de tamaño de la imagen para que sea compatible con nuestro modelo.
- Recorte de la imagen, quedándonos con la parte central.
- Conversión de imágenes *numpy* o *PIL* a tensores. También cambia las dimensiones de la imagen a C x H x W (canal x altura x ancho), que es la forma que tiene que tener nuestra imagen para trabajar con PyTorch.
- Normalización tomando la media y la desviación estándar por canal.

Estas trasformaciones se aplican cuando se carga el dataset, antes de que se introduzcan en el modelo de red neuronal.

6.2 Arquitectura de la red

La arquitectura de red usada en este trabajo consta de una red neuronal convolucional creada desde cero y, además, para obtener unos resultados más precisos se ha hecho uso de la técnica de *transfer learning* unida a *fine-tuning* en la que se han probado cinco tipo de modelos distintos.

6.2.1 Transfer Learning

Para contrastar resultados y obtener mejores predicciones, se han usado CNN que ya han sido creadas y pre-entrenadas obteniendo resultados óptimos. Esta estrategia se denomina *transfer learning*.

Un modelo pre-entrenado consiste en un modelo que ya ha sido entrenado, pero con un dataset de referencia que será similar al conjunto de datos que queremos entrenar. Las ventajas que nos proporciona esta técnica son múltiples debido al coste computacional y a la complejidad que comprende la arquitectura de una red adecuada para nuestra información.

Esta técnica suele usarse sobre todo en tareas de visión por ordenador y procesamiento del lenguaje natural debido a la gran cantidad de información y cálculos que se procesan.

En las redes neuronales, se suelen detectar primero los bordes, después las formas y, por último, características específicas. Con el *transfer learning* solo se utilizan las primeras capas y las medias, siendo las últimas reentrenadas. Es decir, en esas primeras capas, nuestra red está entrenada para reconocer los objetos, y posteriormente, en las últimas capas, se entrenaría para diferenciar ese objeto de cualquier otro. Esta primera etapa en la se extraen las características visuales se denomina *feature extraction* o extracción de características y es la parte con mayor coste computacional. El objetivo es transferir tanto conocimiento como sea posible de las tareas para las que fue creado el modelo a nuestro modelo.

Por lo tanto, el proceso que se desarrolla consiste en que de la red pre-entrenada solo se usará su parte convolucional. Las características que se extraen de esta parte pasan a un clasificador, que será el encargado de extraer características específicas y clasificar la imagen en función de ellas. Este clasificador sí será entrenado para nuestro problema [32].

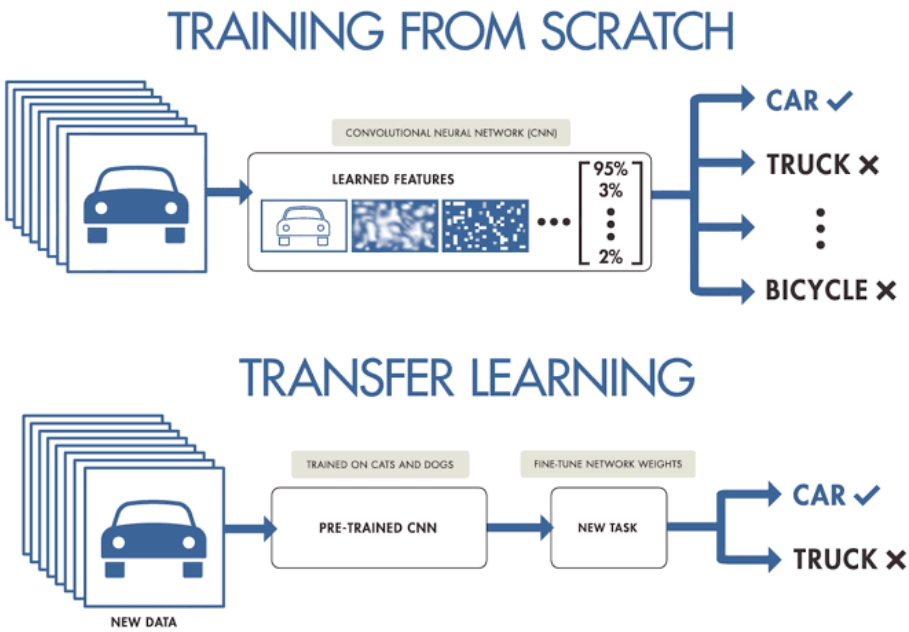


Figura 18. *Transfer Learning.* [33]

Esta técnica será acompañada de otra técnica denominada *fine-tuning*. El término *fine-tuning* hace referencia a hacer pequeños ajustes para conseguir nuestra salida deseada. Mediante esta técnica se ajustan los pesos de las capas intermedias del modelo de *transfer learning* para que se adapten a nuestro problema. Posteriormente, inicializaremos el modelo con las primeras capas congeladas para no modificar los pesos y las últimas capas del modelo serán descongeladas, es decir, entrenadas con nuestro dataset. Esta estrategia, permite que la red no tenga que ser entrenada desde el principio, ahorrando tiempo y aprovechando la información adquirida por otra red.

6.2.2 Arquitectura usada

- **CNN creada**

Para este trabajo hemos creado una CNN propia. Nuestra CNN está formada por dos métodos uno en el que se definen las capas que queremos usar en la red, y otro en el que se crea la red usando las capas definidas en el método anterior.

Hemos definido nueve capas en nuestra red neuronal, tres de estas capas son capas de convolución con un kernel 3x3, y estas están seguidas cada una de una capa *pooling* en la que se le aplica la operación *max pooling* definida en los apartados superiores. Por último, se han definido tres capas lineales, a dos de ellas le aplicaremos la función de activación *ReLu* y a la última se le aplicará la función *Softmax* que nos permitirá obtener la predicción de cada clase y un resultado final.

La red ha sido creada conforme a mis parámetros de entrada, considerando el tamaño de imagen y, los canales de entrada y salida. En la siguiente imagen, podemos observar las transformaciones que sufre una imagen a lo largo de nuestra CNN:

```
torch.Size([3, 64, 64])
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1          [-1, 16, 64, 64]             448
         MaxPool2d-2          [-1, 16, 32, 32]               0
            Conv2d-3          [-1, 32, 32, 32]           4,640
         MaxPool2d-4          [-1, 32, 16, 16]               0
            Conv2d-5          [-1, 64, 16, 16]          18,496
         MaxPool2d-6            [-1, 64, 8, 8]               0
            Linear-7                [-1, 1024]       4,195,328
            Linear-8                 [-1, 512]         524,800
            Linear-9                  [-1, 10]           5,130
================================================================
Total params: 4,748,842
Trainable params: 4,748,842
Non-trainable params: 0
----------------------------------------------------------------
```

Figura 19. *Arquitectura de la red creada.*

- **DenseNet**

Mediante la técnica de *transfer learning* hemos implementado la red *DenseNet-121* que ha demostrado tener gran eficacia en estudios anteriores. En esta red cada capa además de obtener información de la capa anterior la obtiene de todas las que están anteriores a ellas y a su vez, transfiere su mapa de características a todas las capas siguientes. Para ello, se hace uso de la concatenación. Así, se hace uso de un conocimiento colectivo [34].

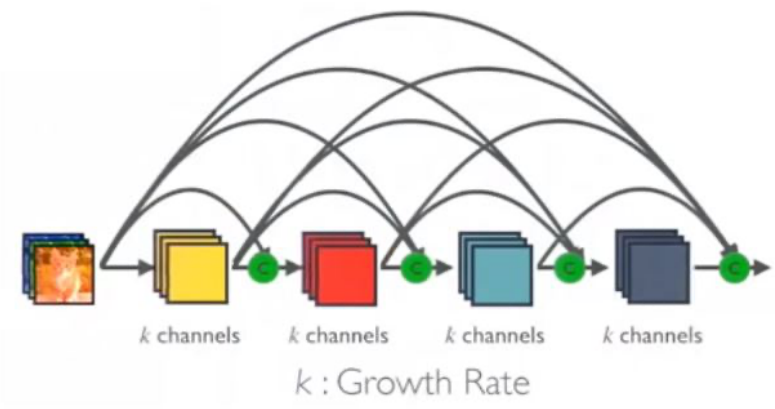


Figura 19. *Arquitectura DenseNet.* [34]

- **Inception v3**

Otro modelo pre-entrenado usado ha sido *Inception v3*. Lo que hace especial a esta red es que durante el entrenamiento usa dos capas de salida. La segunda capa consiste en una salida auxiliar, mientras que la primera es una capa lineal. Esta red consta de 42 capas y su arquitectura se define en la siguiente imagen: [35]

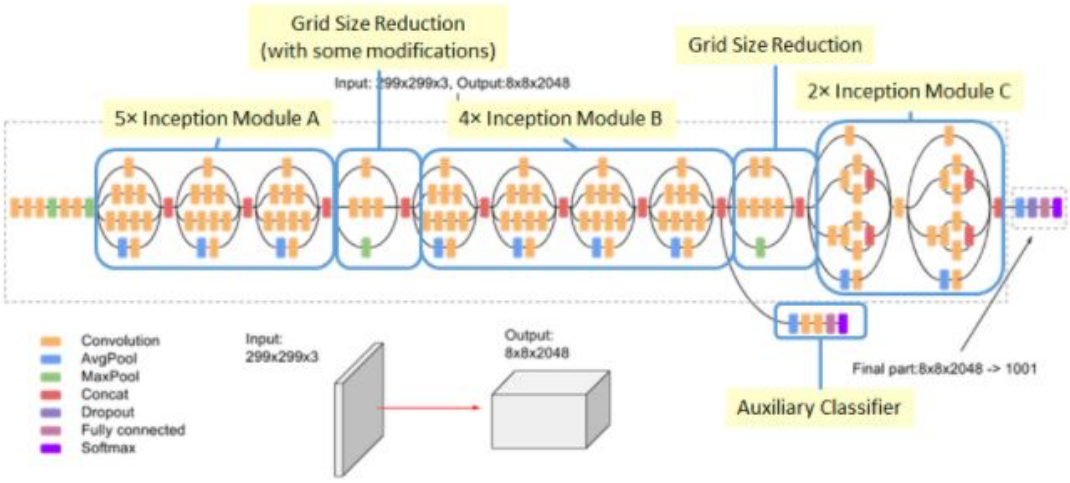


Figura 20. *Arquitectura Inception v3.* [35]

- **ResNet**

El siguiente modelo utilizado es *ResNet-18*. Esta red consta de unas 152 capas en las que se aprenden las funciones residuales de representación en lugar de aprender la señal de representación directamente. De forma resumida, el número de capas va aumentando introduciendo una conexión residual mediante capas de identidad. Esta red usa conexiones de salto lo que le permite avanzar a las siguientes capas evitando dañar el gradiente. [36]

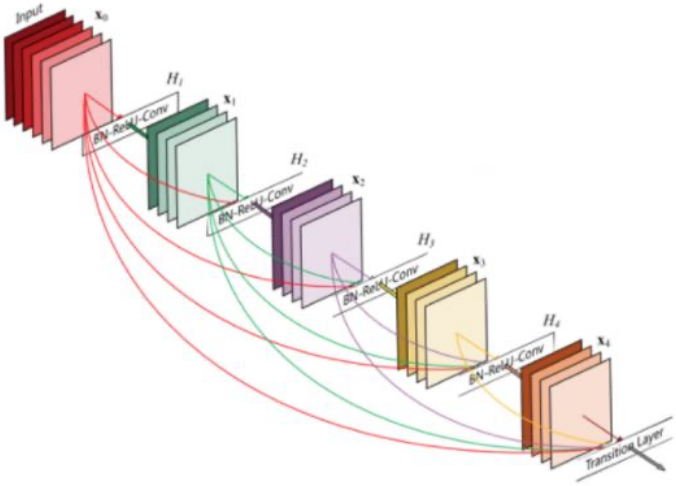


Figura 21. *Arquitectura ResNet.* [36]

- **AlexNet**

La siguiente red usada es *AlexNet* que es considerada la primera CNN exitosa. Su arquitectura consiste en ocho capas, cinco capas convolucionales y tres capas totalmente conectadas. En su capa de salida se le aplica la función *Softmax*. A todas las salidas de las capas se le aplica la función de activición *ReLu*. Su arquitectura se muestra en la siguiente imagen: [37]

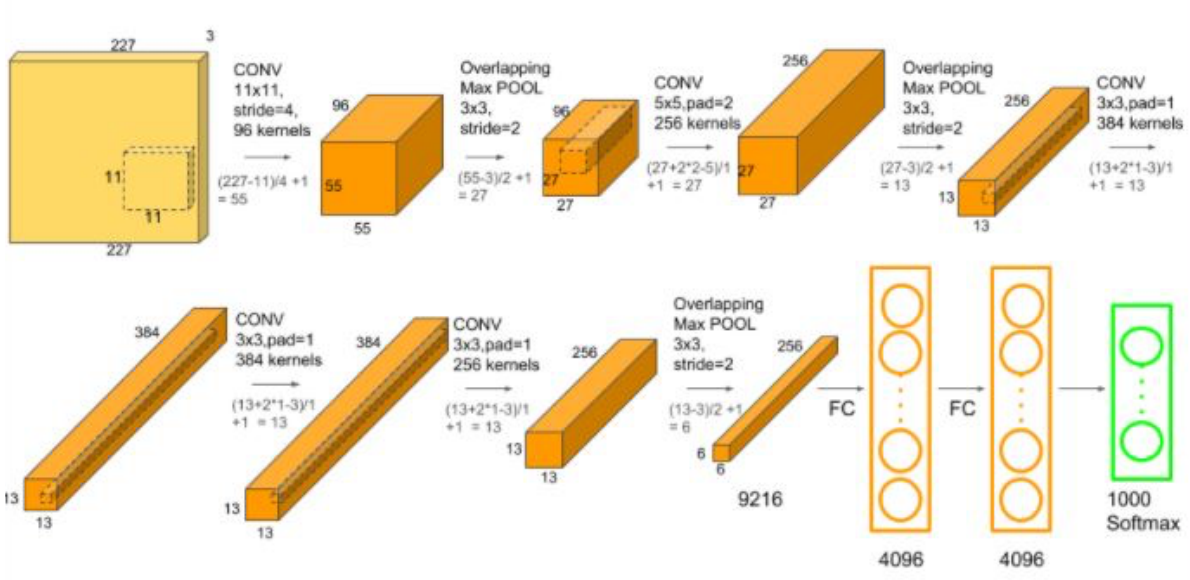


Figura 22. *Arquitectura AlexNet.* [38]

- **VGG**

Por último, acabaremos con *VGG-18*. Esta es una red muy simple, usa capas convolucionales 3x3 apiladas convirtiéndola así en una red profunda. El volumen se reduce usando *max pooling*. Tiene dos capas totalmente conectadas y están seguidas de una función *Softmax*. Su arquitectura se puede ver en la siguiente imagen: [39]

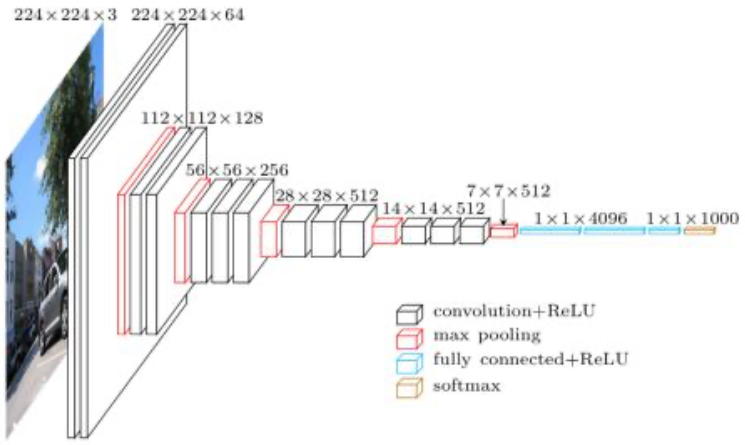


Figura 22. *Arquitectura VGG.* [40]

6.3 Optimizador y función de pérdida

Para optimizar el modelo se ha utilizado un optimizador SGD. Este optimizador implementa el descenso de gradiente estocástico. Cuenta con los parámetros tasa de aprendizaje y momento opcional. La tasa de aprendizaje se encarga de determinar el tamaño de paso o *step size* en cada iteración mientras que el algoritmo de optimización se mueve a lo largo de la función de pérdida. Es decir, representa la velocidad a la que el modelo aprende. Una tasa de aprendizaje pequeña hará que un modelo sea más preciso pero, a su vez, hará que el entrenamiento sea más lento. Por otro lado, el momento se encarga de almacenar el gradiente de los pasos anteriores para determinar en qué dirección seguir.

Como función de pérdida hemos usado *Cross Entropy*. Esta función es generalmente usada para tareas de clasificación multi-clase. Se encarga de medir el error de un modelo que tiene como salida una probabilidad entre 0 y 1. Conforme la predicción no coincide con el valor real, la pérdida aumenta.

6.4 Evaluación del modelo

Para evaluar el modelo creado emplearemos varios métodos que serán descritos a continuación. En primer lugar, graficaremos la curva de aprendizaje del modelo que nos mostrará la precisión y la pérdida que han tenido tanto los conjuntos de entrenamiento como de validación en cada época. Esta además de indicarnos si nuestro modelo está obteniendo buenos resultados nos mostrará el punto en el que modelo ha llegado a su máximo nivel de aprendizaje y ya no mejorará más.

En segundo lugar, tal y como se ha comentado en apartados anteriores, se ha creado un conjunto de prueba que servirá para evaluar el modelo una vez entrenado. Este medirá el error y la precisión del modelo tras su entrenamiento y validación. Asimismo, se encargará de hacer predicciones y compararlas con las etiquetas reales viendo si coinciden. También mostrará la actuación de cada clase en esas predicciones.

Por último, se creará una matriz de confusión sobre el conjunto test. La matriz de confusión nos permitirá observar la actuación del algoritmo y determinar los aciertos y errores que tiene nuestro modelo a la hora del aprendizaje de la información. Esta es una herramienta muy útil pues nos permite ver si una clase se está confundiendo con otra

7

Resultados

En este capítulo se presentan los resultados obtenidos de cada una de las redes entrenadas. Se le irán aplicando varios cambios para evaluar el modelo como la variación de la tasa de aprendizaje y se le hará una *cross-validation*. Se irán comentando los resultados y se representará el mejor resultado para cada red usada.

Para evaluar los resultados se hará uso de las medidas de rendimiento presentadas a continuación. Estas medidas de rendimiento serán sacadas para cada modelo probado con los diferentes parámetros propuestos.

- **Medidas de precisión y pérdida.** Se presentará un valor medio de las medidas de precisión y pérdida del conjunto de prueba, calculado a partir de la pérdida generada dividida entre la longitud del conjunto de prueba.
- **Curvas de aprendizaje.** Se representarán las curvas de aprendizaje de cada modelo midiendo tanto su precisión como su error para los conjuntos de entrenamiento y validación. Esto nos permitirá determinar si un modelo está aprendiendo bien o no, así como apreciar la velocidad a la que aprende y determinar si hay algún fallo en alguno de los conjuntos que haga que el modelo no realice un buen aprendizaje.

- **Matriz de confusión.** Se proporcionará una matriz que tome las predicciones y las imágenes verdaderas del conjunto test para ver tanto la cantidad como el tipo de aciertos y fallos que nuestro modelo está cometiendo. Esta matriz no será normalizada.

El resto de medidas de rendimiento utilizadas para evaluar la eficacia de un modelo pueden ser encontradas en el Apéndice A, así como totalidad de los resultados. Estas medidas no son presentadas a continuación puesto a que aportan información menos relevante a la hora de evaluar. Y el resto de resultados se recogen en ese mismo apéndice pues como ya he comentado solo se presentará el modelo que actúe mejor y aporte beneficios a este estudio.

7.1 CNN creada

En primer lugar, probaremos con tres tasas de aprendizaje que serán 0.005, 0.05 y 0.5 y veremos cual es la más adecuada. El entrenamiento se ejecutará por 15 épocas.

Modelo	**Tasa de aprendizaje**	**Precisión Test**	**Pérdida Test**
A	0.005	8.7500	54.4661
B	0.05	10.7500	52.1979
C	0.5	4.0000	66.5360

Tabla 2. *Resultados con distintas tasas de aprendizaje.*

Como se puede observar en la tabla superior la precisión es muchísimo peor en el tercer caso, por lo que este modelo queda descartado. En cuanto a las otras dos, la que presenta una tasa de aprendizaje mayor muestra unas ligeras mejoras. Pero nos basaremos en la curva de aprendizaje para ver cual ha tenido un mejor desarrollo.

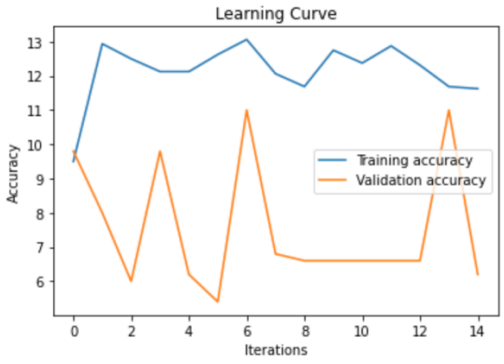


Figura 23. *Modelo A.*

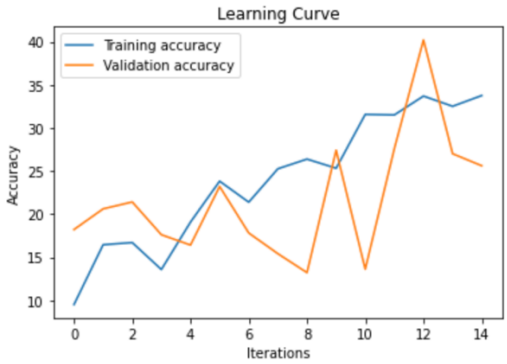


Figura 24. *Modelo B.*

Observando sus curvas de aprendizaje se puede observar que el modelo A no está aprendiendo mucho pues tiene una curva muy inestable y un poco horizontal. Por lo que descartaremos este caso. En cuanto a la otra, muestra una gráfica un poco inestable pero parece que si está aprendiendo.

A continuación, mostraremos el comportamiento del modelo no descartado en su matriz de confusión.

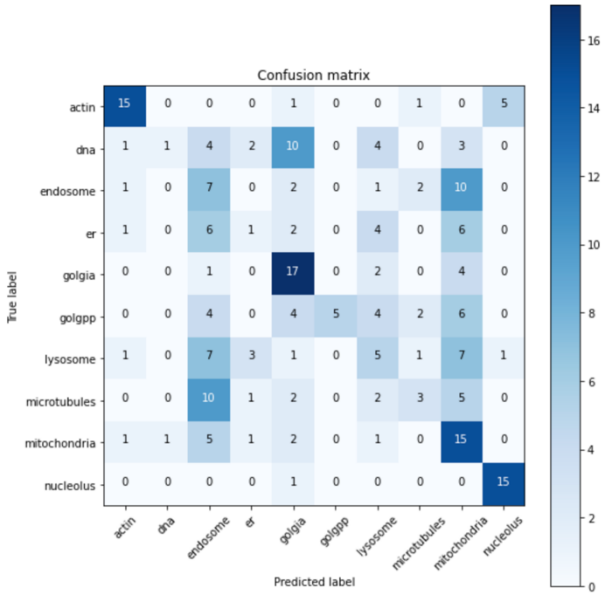


Figura 25. *Matriz de confusión.*

Como podemos observar el modelo aprende bien en ciertas clases, pero otras, como por ejemplo, *DNA* o *microtubules* tiene bastantes problemas de confusión con otras clases. Por lo que esta CNN creada no sería la más óptima.

7.2 DenseNet

Como la primera red neuronal convolucional no ha obtenido muy buenos resultados, hemos optado por aplicar *transfer learning* con *fine-tuning* utilizando *DenseNet*. Como con el modelo anterior hemos probado otras tres tasas de aprendizaje durante 15 épocas y veremos cuales son los parámetros más óptimos.

Modelo	Tasa de aprendizaje	Precisión Test	Pérdida Test
D	0.001	21.6250	19.5608
E	0.003	24.3750	14.1539
F	0.005	23.8750	15.3104

Tabla 3. *Resultados con distintas tasas de aprendizaje.*

Como podemos observar, esta red muestra mucho mejores resultados que la red anterior. La precisión es bastante alta en los tres modelos, pero seleccionaremos los dos con mayor precisión y menor pérdida.

A continuación para ver como se comporta cada modelo y quedarnos con el más preciso mostremos sus matrices de confusión. Se puede ver que ambas son bastante precisas en casi todas las clases. No obstante, aunque en la clase *endosome* parece ser ligeramente mejor el modelo F, en el resto de clases el modelo E parece ser más preciso.

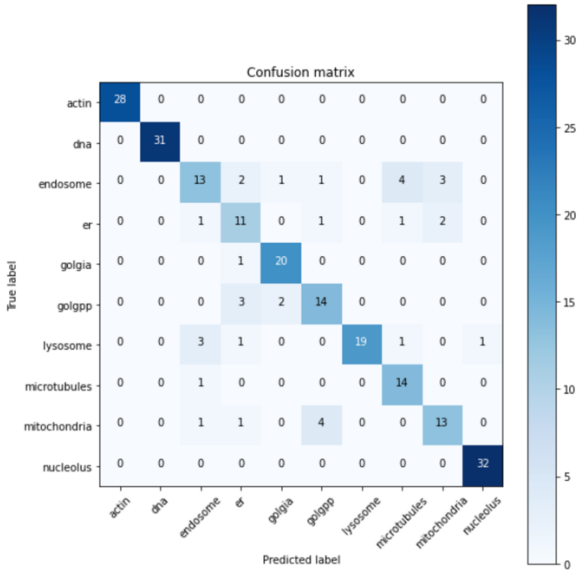


Figura 26. *Modelo E.*

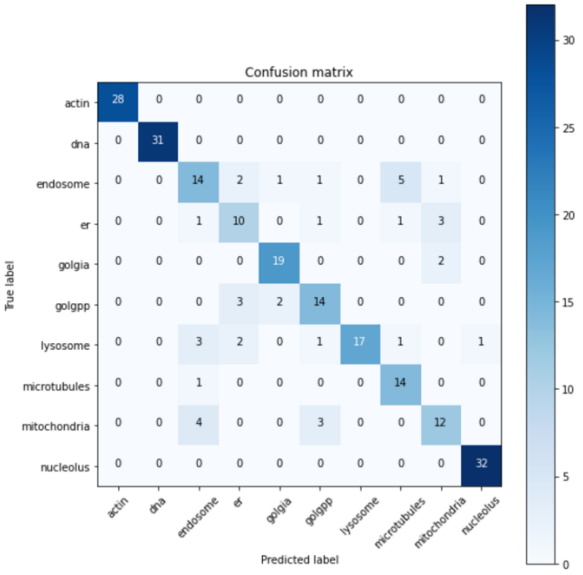


Figura 27. *Modelo F.*

Para comprobar su actuación a lo largo del entrenamiento, mostraremos las curvas de aprendizaje del modelo E. Podemos ver que las curvas representadas son bastante estable disminuyendo la pérdida a medida que el modelo se va entrenando y aumentando la precisión en cada época.

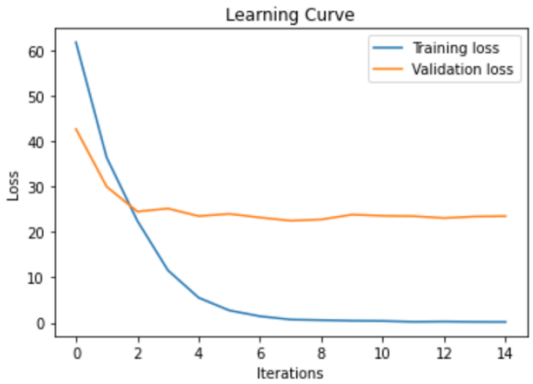


Figura 28. *Pérdida.*

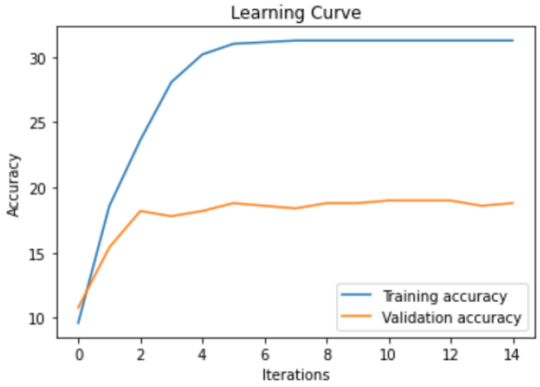


Figura 29. *Precisión.*

7.3 Inception V3

En esta sección procederemos a probar otra red distinta. También, aplicaremos *transfer learning* con *fine-tuning* probando otras tres tasas de aprendizaje durante 15 épocas y veremos cuales son los parámetros más óptimos.

Modelo	**Tasa de aprendizaje**	**Precisión Test**	**Pérdida Test**
G	0.001	22.6250	19.1961
H	0.003	23.6250	17.9486
I	0.005	22.2500	22.6352

Tabla 4. *Resultados para Inception V3.*

Todos los modelos presentan una buena precisión y muy parecida, en cuanto a la pérdida presentan ciertas diferencias. Basándonos en la estabilidad de su curva de aprendizaje nos quedaremos con el modelo H pues muestra una mayor estabilidad tanto en la pérdida como en la precisión. Estas gráficas se muestran a continuación.

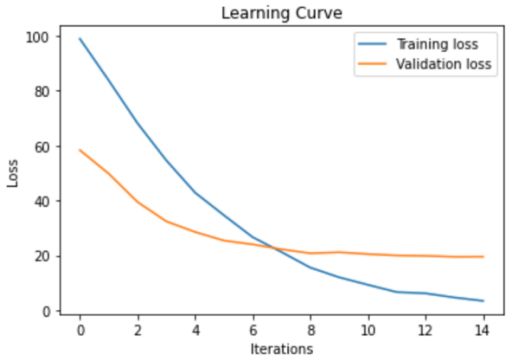


Figura 30. *Pérdida.*

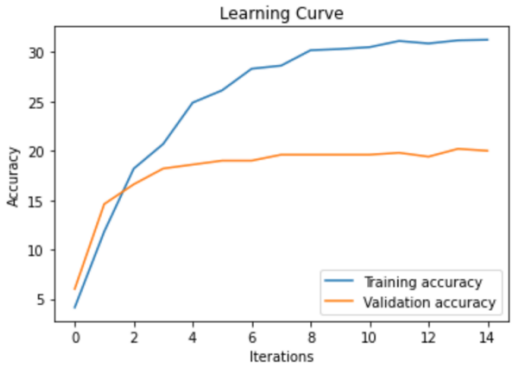


Figura 31. *Precisión.*

Ahora observaremos qué tipo de aciertos y errores esta cometiendo nuestro modelo mediante la matriz de confusión.

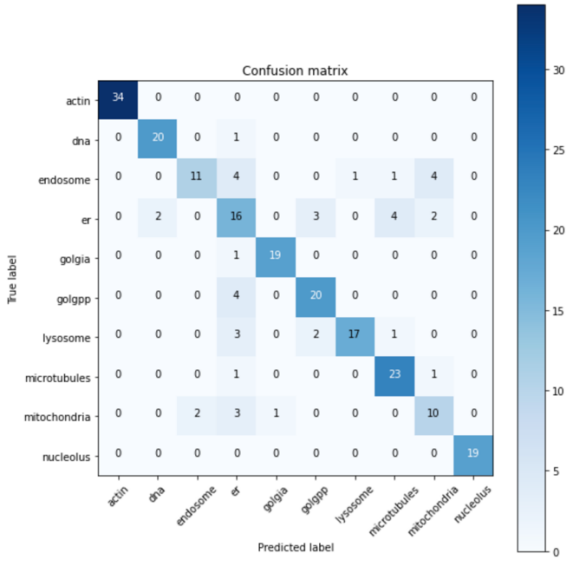


Figura 32. *Matriz de confusión.*

Como podemos ver, nuestra matriz no comete muchos errores, es bastante precisa, de hecho hay clases en las que no comete errores como son *actin* y *nucleolus*. Y en el resto de clases los errores que comete son bastante bajos.

7.4 ResNet

En este apartado procederemos a probar una red residual ya creada y pre-entrenada. Para ello, tomaremos el mismo procedimiento que con las anteriores,

ajustaremos la tase de aprendizaje probando distintos valores y la entrenaremos durante 15 épocas. Tras ello, evaluaremos la eficacia de los distintos modelos.

Modelo	Tasa de aprendizaje	Precisión Test	Pérdida Test
J	0.001	20.7500	21.6581
K	0.003	21.8750	20.7980
L	0.005	21.1250	21.3241

Tabla 5. *Resultados para ResNet.*

Tanto los valores de precisión como los de pérdida presentados en la tabla superior, son bastante parecidos entre sí. No obstante, optaremos por elegir los dos modelos que tienen mayor precisión. A continuación, veremos cómo funcionan sus curvas de aprendizaje para el error y determinar el desarrollo de los modelos.

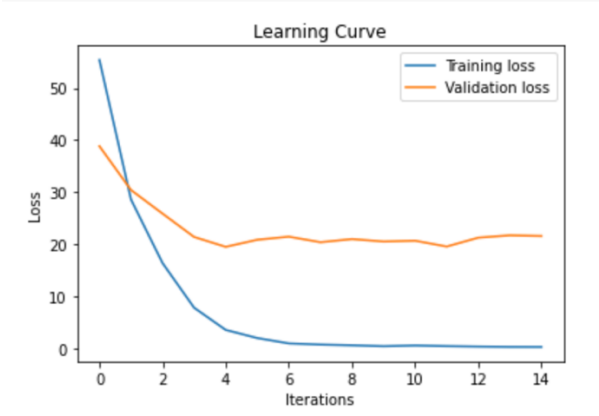


Figura 33. *Error del modelo K.*

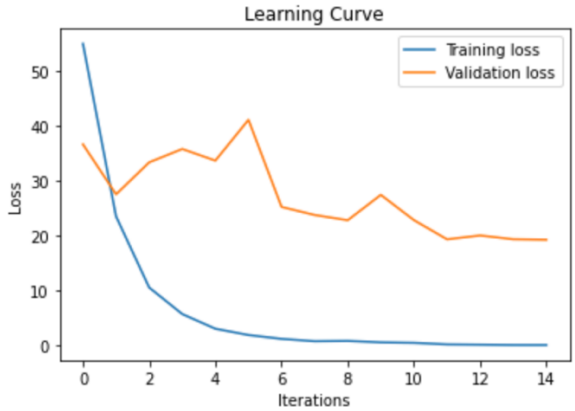


Figura 34. *Error del modelo L.*

Si observamos la curva de aprendizaje del modelo L no presenta buenos resultados para el conjunto de validación obteniendo un error bastante ruidoso, esto puede ser debido a que el modelo ha aprendido demasiado bien, incluido los ruidos. Por este motivo, descartaremos este modelo y optaremos por el modelo K que muestra un error adecuado.

Para comprobar que este modelo funciona bien, se muestra su matriz de confusión. Como podemos ver abajo, nuestro modelo ha aprendido bien, aunque muestra ciertas confusiones con otras clases. Cabe mencionar el problema que podría tener confundiendo endosomas con mitocondrias.

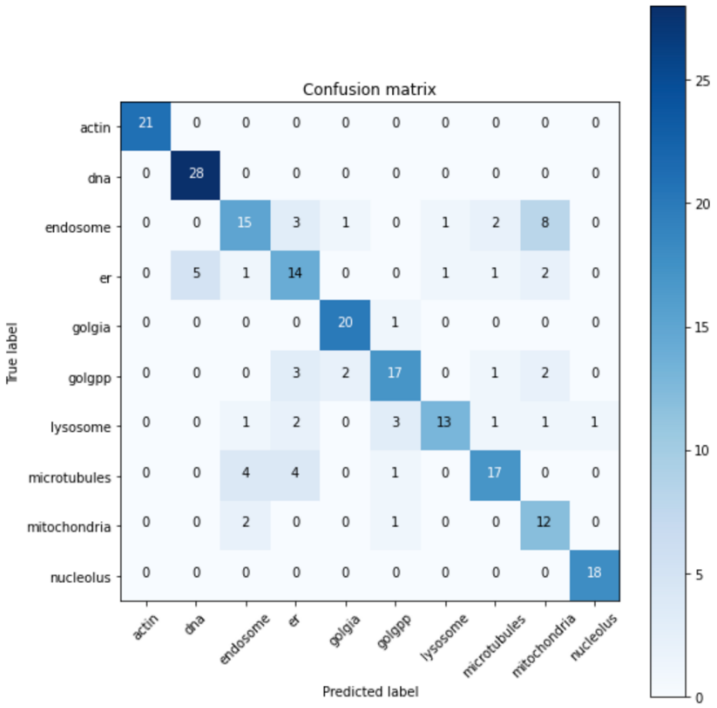


Figura 35. *Matriz de confusión del modelo K.*

7.5 AlexNet

Ahora procedemos a probar una red menos densa que la anterior, también usando *fine-tuning*. Probaremos con tres tasas de aprendizaje distintas y mostraremos los resultados obtenidos.

Modelo	Tasa de aprendizaje	Precisión Test	Pérdida Test
M	0.001	20.0000	24.5707
N	0.003	20.0000	32.8539
Ñ	0.005	18.7500	43.6520

Tabla 6. *Resultados para AlexNet.*

Este tipo de red parece que no es la más adecuada para nuestro conjunto de datos, pues presenta errores mucho peores que en el resto de redes probadas. Aunque la precisión es bastante parecida en ambos casos, me basaré en la pérdida tan elevada para descartar los modelos N y Ñ.

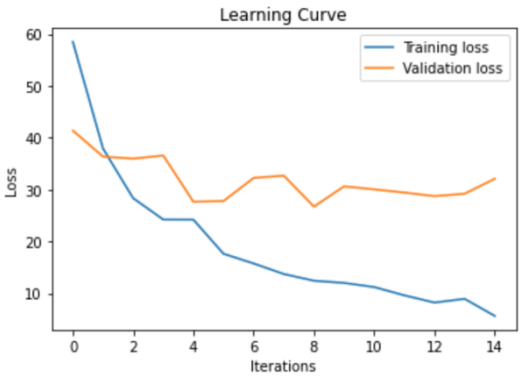


Figura 36. *Error del modelo M.*

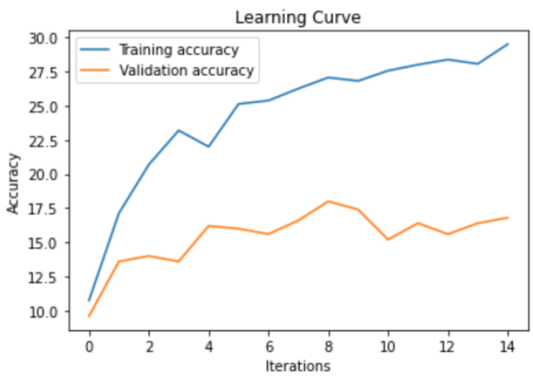


Figura 37. *Precisión del modelo M.*

En las gráficas se puede observar que el modelo no está teniendo un buen proceso de aprendizaje pues muestra unas gráficas muy inestables. Veamos la matriz de confusión para este modelo. La matriz de confusión muestra el mal aprendizaje de la red pues aunque hay clases que reconoce bastante bien, son varias clases en las que presenta problemas, como es el caso de la confusión de endosomas y golgpp con mitocondrias, teniendo prácticamente la misma probabilidad de equivocarse que de acertar.

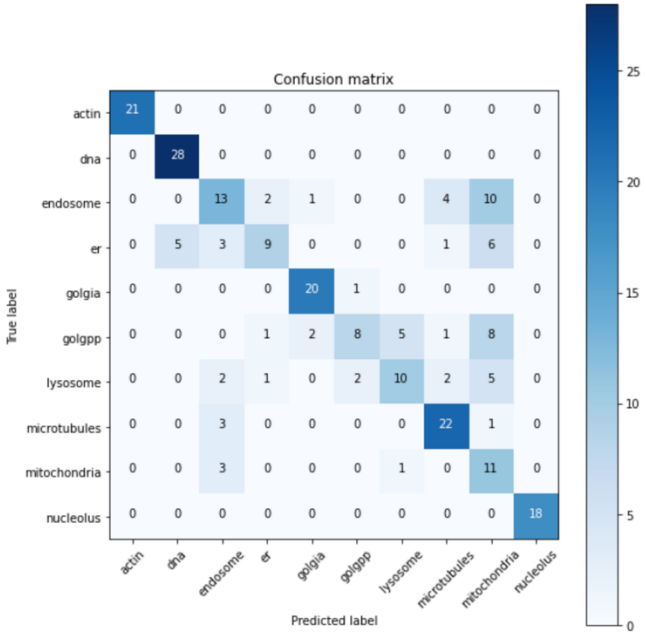


Figura 38. *Matriz de confusión del modelo M.*

7.6 VGG

Por último, se procederá a probar con la red pre-entrenada *VGG*. Siguiendo el mismo procedimiento que en las redes anteriores, ajustando los parámetros hasta conseguir el modelo más óptimo.

Modelo	Tasa de aprendizaje	Precisión Test	Pérdida Test
O	0.001	22.1250	20.6318
P	0.003	22.5000	26.1553
Q	0.005	22.5000	25.0720

Tabla 6. *Resultados para VGG.*

La tabla presenta unas precisiones muy similares entre los tres modelos. Sin embargo, presentan bastante diferencia entre las pérdidas de los distintos modelos. Teniendo en cuenta que esta pérdida es el error entre las predicciones y la salida real, nos quedaremos con el modelo O, pues presenta menor pérdida y no mucha diferencia con el resto de precisiones.

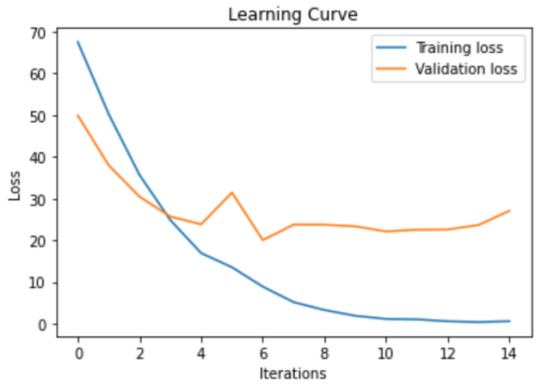


Figura 39. *Error del modelo O.*

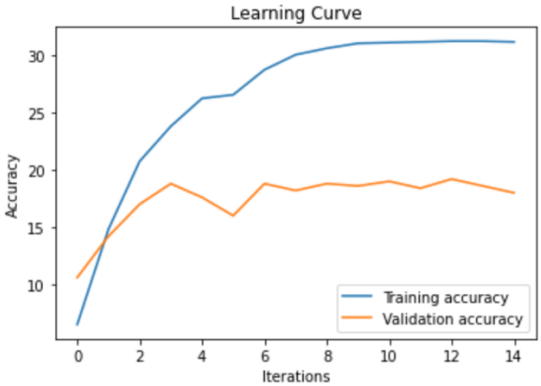


Figura 40. *Precisión del modelo O.*

En la curva de aprendizaje se puede observar que el conjunto de entrenamiento aprende bien. Sin embargo, el conjunto de validación no está desarrollándose del todo bien pues las gráficas se muestran un poco inestables.

Vamos a ver como ha actuado sobre el conjunto de prueba con la matriz de confusión. En general, la red identifica bastante bien las imágenes en las clases. Es

cierto, que tienes algunas confusiones con algunas clases pero no es un porcentaje bastante alto.

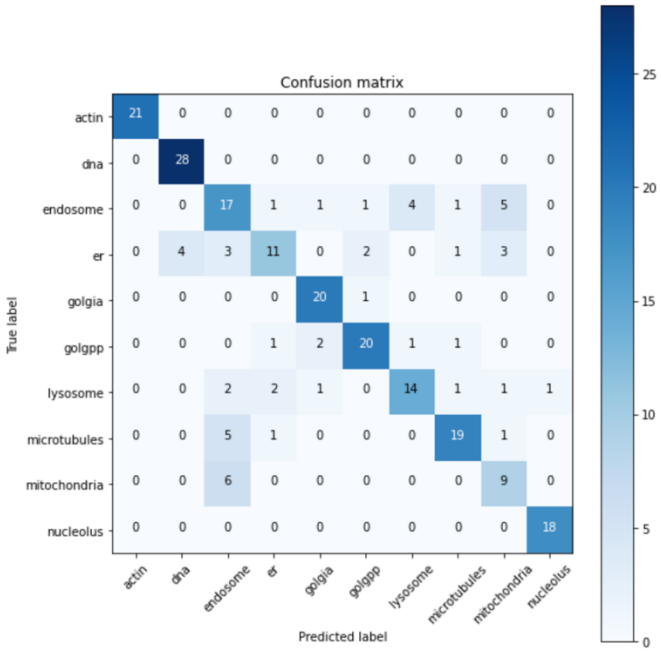


Figura 41. *Matriz de confusión del modelo O.*

8

Discusión

Tras haber realizado las diferentes pruebas con todos los modelos facilitados, se han obtenido en general buenos resultados de identificación de los orgánulos celulares presentes en el HeLa dataset. Si bien, es cierto que algunos modelos han presentado mayor precisión y menor pérdida que otros, me dispongo a compararlos y discutirlos.

En cuanto a la red neuronal convolucional creada por mí para resolver este problema, cabe mencionar que no es una red competente pues los resultados que ofrece son bastante poco precisos aún ajustando las distintas tasas de aprendizaje. Hay algunas clases que reconoce fácilmente, pero en otras comete muchos errores. Consecuentemente, este modelo de red no es factible.

El modelo *DenseNet* parece ser el que mejor se adapta a nuestro problema, tanto con una tasa de aprendizaje de 0.003 como de 0.005 presenta valores de precisión y de error muy buenos. A la hora de distinguir clases presenta muy buenos resultados, si bien puede llegar a confundir una clase con otra pero en una medida muy pequeña. En cuanto a la evolución de su aprendizaje, se observa que aprende a una buena velocidad y de una forma eficiente, siendo la brecha de generalización entre los conjuntos de validación y entrenamiento no muy grande, lo que es bastante bueno.

El modelo *Inception V3* junto con el anterior, son los mejores modelos para nuestro problema. Sus valores de error y precisión son un poco inferiores a los de *DenseNet*, aún así, presenta muy buenos valores de precisión y un error bajo. Si nos fijamos en sus curvas de aprendizaje podemos observar que aprende a una velocidad constante y no encuentra muchas dificultades. En cuanto a la matriz de confusión, también obtiene muy buenos valores teniendo muy pocas confusiones.

El modelo *ResNet* con una tasa de aprendizaje de 0.003 ha obtenido una buena precisión y error, podría considerarse como el tercer modelo que mejor se adapta a nuestro problema. Presenta un buen modelo de aprendizaje pues su error tiende gradualmente hacia un valor constante. Aunque su matriz de confusión, muestre confusiones no muy elevadas y mayoritariamente aciertos, es motivo de preocupación el hecho de que pueda confundir lisosomas con mitocondrias.

La red *AlexNet* es la que peores resultados nos aporta de las redes probadas con *transfer learning*. Si bien su precisión es un poco inferior a la del resto de modelos, los valores de error que presenta son muy elevados, considerando cualquiera de las tasas de aprendizaje probadas. Confunde bastantes clases, hasta el punto de tener las mismas probabilidades de acertar que un orgánulo es *golgpp* que de confundirlo con una mitocondria. También, presenta bastantes problemas a la hora de identificar endosomas. Por estos motivos, este modelo no debería ser considerado como válido para este problema o habría que hacer más investigaciones acerca de su mejora.

Por último, el modelo *VGG* no tiene un mal rendimiento. Es cierto que trabaja mejor con una tasa de aprendizaje más pequeña y aprendiendo más lento, de lo contrario sus niveles de error aumentan. Tal y como este error muestra, se puede ver en su matriz de confusión que presenta algunas clases que pueden dar lugar a fallos o predicciones erróneas.

A continuación, a modo de resumen se presenta una tabla que presenta el mejor modelo de cada red creada:

Tipo de Red	Modelo	Tasa de aprendizaje	Precisión Test	Pérdida Test
CNN creada	B	0.05	10.7500	52.1979
DenseNet	E	0.003	24.3750	14.1539
Inception V3	H	0.003	23.6250	17.9486
ResNet	K	0.003	21.8750	20.7980
AlexNet	M	0.001	20.0000	24.5707
VGG	O	0.001	22.1250	20.6318

Tabla 7. *Mejor modelo de cada red.*

Tras haber discutido el rendimiento de cada modelo, podemos concluir sin lugar a dudas que el modelo mas idóneo para identificar orgánulos en el dataset es el *DenseNet* con una tasa de aprendizaje del 0.003 seguido por el mismo con una tasa de 0.005. Este modelo consigue hacer predicciones acertadas con un aprendizaje rápido. También consigue resolver los problemas planteados en el capítulo 2, sobre la dificultad de diferenciar endosomas y lisosomas, y, GPP130 y gigantina. También cabe destacar que probándolo con 10000 imágenes del conjunto test, obtuvo un 84% de aciertos, acertando mas de 3 de cada 4 imágenes.

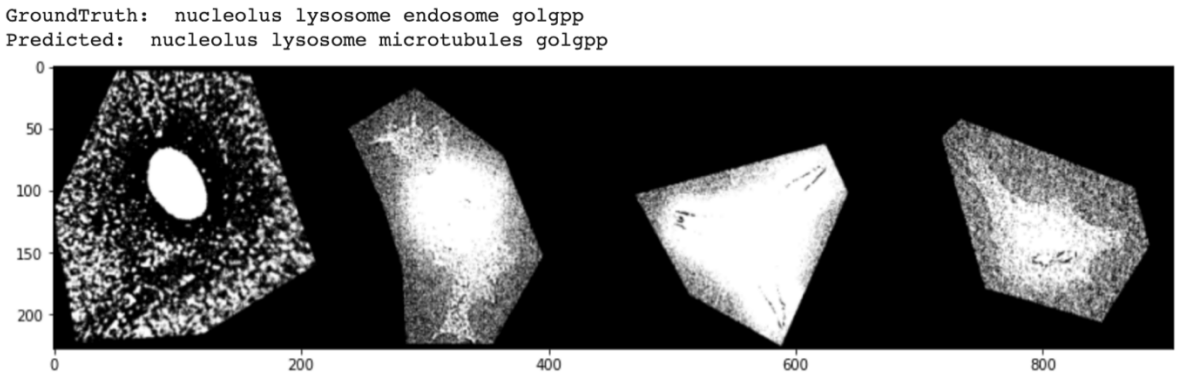


Figura 42. *Comparación de imágenes reales con las predichas por DenseNet.*

También cabe destacar la actuación de la red *Inception V3*, pues consigue unos valores muy similares al modelo anterior, aunque ligeramente inferiores. Este modelo ha conseguido acertar un 82% de 10000 imágenes probadas del conjunto de prueba, consiguiendo acertar 3.28 imágenes de cada 4.

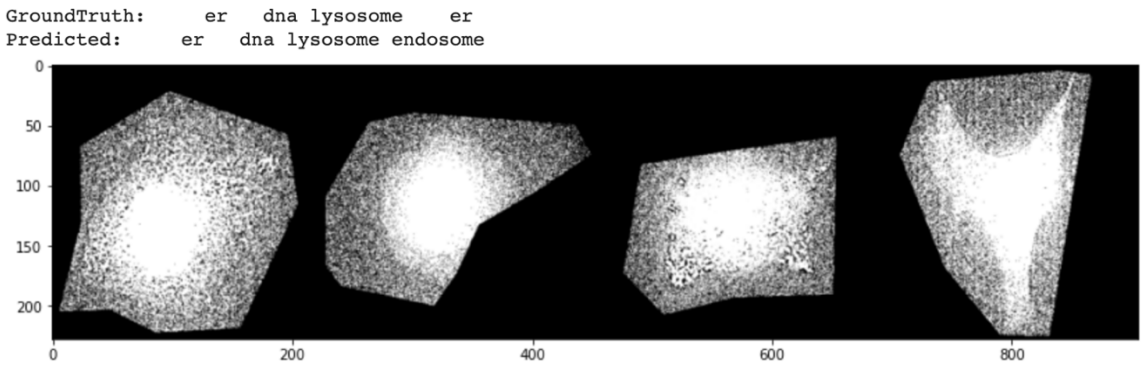


Figura 43. *Comparación de imágenes reales con las predichas por Inception V3.*

9

Conclusión

La localización de proteínas en los orgánulos celulares es importante para estudiarla, puesto que esto nos puede aportar mucha información sobre su función. El desarrollo de este campo es uno de los principales objetivos de la proteómica de localización. Actualmente, los investigadores hacen uso del marcado de proteínas junto con la microscopía de fluorescencia para ver e identificar patrones de localización y así, registrarlos sacando conclusiones razonadas sobre ellos. Es cierto que esta técnica ha funcionado bien a lo largo de los años, pero, es necesario mejorarla debido a la cantidad de datos que se generan y a las proteínas descubiertas y caracterizadas cada año.

Para ello, la aplicación de métodos informáticos puede presentar una gran mejora, pues son eficaces, menos costosos, objetivos y capaces de diferenciar las características más imperceptibles por el ojo humano. La Inteligencia Artificial y en concreto, las Redes Neuronales Convolucionales, son capaces de detectar patrones sobre la localización de proteínas y manejar una gran cantidad de datos no requiriendo mucho tiempo. Los resultados obtenidos de estas comparaciones, podrían proporcionar al instante conocimiento acerca de la estructura y función de una proteína.

Con este trabajo, se ha desarrollado un sistema para la identificación y predicción de orgánulos celulares en imágenes de microscopía de fluorescencia y hemos demostrado que este desarrollo puede producir clasificadores capaces de distinguir en su mayoría patrones en los orgánulos de células HeLa. Para ello, se ha creado una red neuronal convolucional y se han probado distintas redes previamente entrenadas aplicándoles pequeñas modificaciones para ver la precisión de cada modelo y determinar cual se adapta mejor al problema.

Considerando los resultados obtenidos tanto la precisión como la tasa de error, los modelos *DenseNet* como *Inception V3* podrían ser llevados a la práctica en la tecnología de clasificación de imágenes y sería conveniente más investigación sobre esta línea, pues podrían llegar a ser una herramienta prometedora mejorando la precisión y eficacia de la clasificación de imágenes subcelulares de microscopía de fluorescencia. Estas serían capaces de procesar una gran cantidad de datos, sustituyendo la forma tradicional de clasificación de este tipo de imágenes.

Las ventajas de un sistema automatizado como este son numerosas. En primer lugar, con todos los datos obtenidos, se podría construir una base de datos que permitan comparar los nuevos patrones de localización con el proteínas ya existentes, viendo proteínas que se localizan de forma parecida. En segundo lugar, podrían proporcionar una lista completa de todos los patrones de proteínas de una célula. También, se podrían descubrir nuevos genes y editarlos.

Aunque por el momento estos objetivos no están al alcance, en un futuro cercano los avances en este campo serán múltiples haciendo posibles muchas de las cosas mencionadas en este trabajo. Esto requiere más investigación y el uso de herramientas informáticas como GPU's más potentes, pues analizar tal cantidad de datos requiere tiempo, así como mejorar un poco la precisión de los distintos modelos de CNN para que así el índice de fallo sea el mínimo posible. Tras haber mejorado estas pautas, sería interesante comparar los resultados obtenidos con la clasificación hecha por investigadores y determinar cómo de beneficiosa es su aplicación en un ámbito laboral.

Referencias

[1] A. Albarracín, La teoría celular en el siglo XIX, Madrid - España: Ediciones Akal, 1992.

[2] J. S. Raisman y A. M. Gonzalez, «Célula Eucariota I: Generalidades,» *Biología edu,* 2008.

[3] B. Alberts y D. Bray, Introducción a la Biología Celular, Nueva York: Medica Panamericana, 1998.

[4] B. L. Roberts, W. D. Richardson y A. E. Smith, «The effect of protein context on nuclear location signal function,» *Science Direct,* 1987.

[5] W. P. Blackstock y M. P. Weir, «Proteomics: quantitative and physical mapping of cellular proteins,» *Trends in Biotechnology,* 1999.

[6] G. K. Valet y A. Tárnok, «Cytomics in predictive medicine,» *Wiley online Library,* 2003.

[7] J. Ghanchi, «¿Cómo está revolucionando la inteligencia artificial el desarrollo de software ?,» *BBVA Open Mind,* 2019.

[8] J. P. Luzio, «Relación entre endosomas y lisosomas,» *Biochemical Society Transactions,* 2001.

[9] L. Rouhiainen, Inteligencia Artificial: 101 cosas que debes saber hoy sobre nuestro futuro, Barcelona: Planeta, 2018.

[10] A. González, «¿Qué es Machine Learning?,» *cleverdata.*

[11] F. Bre, J. M. G. Gimenes y V. D. Fachinotti, «Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks,» *Research Gate ,* 2017.

[12] I. C. Education, «Deep Learning,» *IBM,* 2021.

[13] D. Calvo, «Definición de red neuronal artificial,» *Diego Calvo,* 2017.

[14] E. Stevens, L. Antiga y T. Viehmann, Deep Learning with PyTorch, Nueva York: Manning Shelter Island, 2020.

[15] V. Divakar, «Quantinsti,» 2018. [En línea]. Available: https://blog.quantinsti.com/backpropagation/.

[16] S. Albawi, T. A. Mohammed y S. Al-Zawi, «Understanding of a convolutional neural network,» *IEEE Xplore,* 2018.

[17] S. Silva y E. Freire, «Intro a las redes neuronales convolucionales,» *Bootcamp AI,* 2019.

[18] Cecbur, «Convolutional Neural Network with Color Image Filter,» *Wikipedia,* 2019.

[19] K. Patel, «Convolutional Neural Networks — A Beginner's Guide,» *Toward Data Science,* 2019.

[20] D. Mwiti, «Tutorial, Convolutional Neural Networks: An Intro,» *Heartbeat,* 2018.

[21] S. Team, «Convolutional Neural Networks (CNN): Step 3 - Flattening,» *SuperDataScience,* 2018.

[22] S. Team, «Convolutional Neural Networks (CNN): Step 4 - Full Connection,» *SuperDataScience,* 2018.

[23] R. Kazmi, «Artificial Intelligence (AI) in the Biotech Industry,» *Koombea,* 2021.

[24] V. Rees, «Beyond the phenotype: using AI to better understand disease,» *Drug Target Review,* 2019.

[25] ". celular", «Significados.com,» 2019. [En línea]. Available: https://www.significados.com/nucleo-celular/.

[26] ". endoplasmático", «Significados.com,» 2019. [En línea]. Available: https://www.significados.com/reticulo-endoplasmatico/.

[27] ". d. Golgi", «Significados.com,» 2019. [En línea]. Available: https://www.significados.com/aparato-de-golgi/.

[28] p. y. v. Lisosomas, «PAU I.E.S. José Caballero,» 2016. [En línea]. Available: http://paucurso15-16jc.blogspot.com/2016/01/lisosomas-peroxisomas-y-vacuolas.html.

[29] M. Ruiz, «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Mitocondria#/media/Archivo:Animal_mitochondrion_diagram_es.svg.

[30] J. A. P. Medina y V. M. Carmen, «Citoesqueleto y tráfico vesicular,» *Saber Mas.*

[31] H. Lodish, Biología Celular y Molecular, Nueva York: Medica Panamericana, 2006.

[32] N. Donges, «What Is Transfer Learning? Exploring the Popular Deep Learning Approach.,» *Built in,* 2019.

[33] V. Team, «El transfer learning y las redes convolucionales,» *Viewnext,* 2021.

[34] S. H. Tsang, «Review: DenseNet — Dense Convolutional Network (Image Classification),» *Towards Data Science,* 2018.

[35] N. Inkawhich, «Finetuning Torchvision Models,» *PyTorch.*

[36] D. S. Team, «Redes neuronales residuales – Lo que necesitas saber (ResNet),» *Data Science,* 2020.

[37] Q. Hua, «[NIPS 2012] AlexNet: Review and Implementation,» *Medium,* 2019.

[38] P. Networks, «AlexNet – ImageNet Classification with Deep Convolutional Neural Networks,» *Neurohive,* 2018.

[39] K. Simonyan y A. Zisserman, «Very Deep Convolutional Networks for Large-Scale Image Recognition,» *Cornell University,* 2014.

[40] M. E. Casademunt, «Detección de objetos utilizando redes neuronales profundas,» *Marescas,* 2019.

Apéndice A

Tabla completa de resultados

En este apéndice se presentan todos los resultados obtenidos tras el proceso de entrenamiento con todos los modelos. Los modelos se nombran según el nombre dado en el capítulo 7. Sobre ellos, se presenta su tasa de aprendizaje, el error y precisión sobre el conjunto de entrenamiento. También, se presenta el porcentaje de acierto sobre un conjunto de 10000 imágenes en pruebas sobre el conjunto test. Y se proporciona el porcentaje de precisión sobre cada clase.

Modelo	**Lr**	**Error test**	**Acc. Test**	**Acc. Test [%]**	**Acc. Actin [%]**	**Acc. DNA [%]**	**Acc. Endo. [%]**	**Acc. ER [%]**	**Acc. Golgia [%]**	**Acc. Golgpp [%]**	**Acc. Lyso. [%]**	**Acc. Micro. [%]**	**Acc. Mito. [%]**	**Acc. Nucl. [%]**
A	0.005	54.4661	8.7500	30	68.2	4.0	30.4	5.0	70.8	20.0	19.2	13.0	57.7	93.8
B	0.05	52.1979	10.7500	36	68.2	4.0	30.4	5.0	70.8	20.0	19.2	13.0	57.7	93.8
C	0.5	66.5360	2.8750	10	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
D	0.001	19.5698	21.6250	75	100.0	86.4	40.0	82.6	91.7	50.0	61.5	62.5	66.7	100.0
E	0.003	14.1539	24.3750	84	100.0	100.0	54.2	68.8	95.2	73.7	76.0	93.3	68.4	100.0
F	0.005	15.3104	23.8750	83	100.0	100.0	58.3	62.5	90.5	73.7	68.0	93.3	63.2	100.0
G	0.001	19.1961	22.6250	78	100.0	95.2	47.6	55.6	95.0	75.0	69.6	80.0	62.5	100.0
H	0.003	17.9486	23.6250	82	100.0	95.2	52.4	59.3	95.0	83.3	73.9	92.0	62.5	100.0
I	0.005	22.6352	22.2500	77	100.0	96.2	31.6	73.1	90.9	72.7	50.0	80.0	70.6	100.0
J	0.001	21.6581	20.7500	72	100.0	92.9	53.3	41.7	100.0	60.0	63.6	76.9	33.3	100.0
K	0.003	20.7980	21.8750	76	100.0	100.0	50.0	58.3	95.2	68.0	59.1	65.4	80.0	100.0
L	0.005	21.3242	21.1250	73	100.0	100.0	40.0	45.8	95.2	80.0	63.6	65.4	53.3	100.0
M	0.001	24.5707	20.0000	69	100.0	100.0	43.3	37.5	95.2	32.0	45.5	84.6	73.3	100.0
N	0.003	32.8539	20.0000	69	100.0	100.0	30.0	62.5	95.2	52.0	50.0	69.2	46.7	100.0
Ñ	0.005	43.6520	18.7500	65	100.0	96.4	36.7	50.0	90.5	28.0	31.8	73.1	60.0	100.0
O	0.001	20.6318	22.1250	76	100.0	100.0	56.7	45.8	95.2	80.0	63.6	73.1	60.0	100.0
P	0.003	26.1553	22.5000	78	100.0	95.0	28.6	54.2	96.3	76.0	76.5	74.1	75.0	100.0
Q	0.005	25.0720	22.500	78	100.0	100.0	43.3	79.2	90.5	64.0	77.3	84.6	46.7	100.0

Tabla 8. *Resultados completos de cada ensayo.*