UNIVERSIDAD DE MÁLAGA

ESCUELA DE INGENIERÍAS INDUSTRIALES

ESCUELA DE INGENIERÍAS INDUSTRIALES

Departamento: Ingeniería de Sistemas y Automática

Área de Conocimiento: Ingeniería de Sistemas y Automática

TRABAJO FIN DE MÁSTER

APLICACIÓN DE TÉCNICAS DE INTELIGENCIA ARTIFICIAL PARA LA DETERMINACIÓN DE MARCADORES RELEVANTES EN INMUNOTERAPIA ANTITUMORAL

Máster en Ingeniería Industrial

Autor: Fernando Huelin Parras

Tutor: Irene Rivas Blanco

Cotutor: Juan Luis Onieva Zafra

MÁLAGA, noviembre de 2023

UNIVERSIDAD DE MÁLAGA

ESCUELA DE INGENIERÍAS INDUSTRIALES

DECLARACIÓN DE ORIGINALIDAD DEL PROYECTO/TRABAJO FIN DE MÁSTER

D./ Dña.: Fernando Huelin Parras.

DNI/Pasaporte: 77682639R. Correo electrónico: fernanh98@gmail.com

Titulación: Máster en Ingeniería Industrial

Título del Proyecto/Trabajo: Aplicación de técnicas de inteligencia artificial para la determinación de marcadores relevantes en inmunoterapia antitumoral.

DECLARA BAJO SU RESPONSABILIDAD

Ser autor/a del texto entregado y que no ha sido presentado con anterioridad, ni total ni parcialmente, para superar materias previamente cursadas en esta u otras titulaciones de la Universidad de Málaga o cualquier otra institución de educación superior u otro tipo de fin.

Así mismo, declara no haber trasgredido ninguna norma universitaria con respecto al plagio ni a las leyes establecidas que protegen la propiedad intelectual, así como que las fuentes utilizadas han sido citadas adecuadamente.

En Málaga, a 1 de noviembre de 2023

Fdo.: Fernando Huelin Parras

RESUMEN

Actualmente, la inteligencia artificial está transformando la manera en que se abordan nuevos desafíos, y es que, el estudio y desarrollo de este campo, pone en evidencia las ventajas que trae consigo.

Hace tiempo que se utilizan técnicas basadas en inteligencia artificial para realizar ciertas tareas, sin embargo, ha sido recientemente con el desarrollo de inteligencias artificiales generativas, cuando se comienza a entender el potencial presente en este campo aún en desarrollo.

Además, entre sus principales técnicas, existe una de utilidad especial conocida como visión artificial, que consiste en darle a un ordenador la capacidad de analizar imágenes y mostrar lo que ocurre en ellas. Esto resulta de interés ya que hay imágenes que presentan características imperceptibles al ojo humano, pero que una máquina puede procesar. En este caso se tiene como ejemplo las imágenes médicas, que en muchos casos presentan patrones que al ser identificados, pueden servir para conocer en mejor profundidad qué le ocurre al paciente.

Esta habilidad de analizar imágenes junto con la ventaja de tratarse de una máquina, hacen de la inteligencia artificial una solución eficiente para muchos problemas eliminando el error derivado del factor humano. Por otro lado, aporta a la comunidad científica otro punto de vista cuando se trata de nuevos retos e investigaciones.

Por tanto, el principal objetivo del presente Trabajo Fin de Máster es el desarrollo y evaluación de una red neuronal artificial especializada en el análisis de imágenes médicas para determinar el estado de una proteína específica, el PD-L1.

PALABRAS CLAVE

- Inteligencia Artificial
- Deep learning
- Computer vision
- Python
- Biomedicina
- Cáncer
- PD-L1
- Inmunoterapia

AGRADECIMIENTOS

Agradecer, en primer lugar, a mis tutores, Juanlu, Isabel e Irene, por haberme brindado la oportunidad de realizar este proyecto con ellos, su constante disposición y el haber adoptado en todo momento una actitud de apoyo conmigo.

A todas las personas que han despertado en mí la motivación para no conformarme y luchar por lo que considero que merece la pena.

A mi pareja, por acompañarme en el camino con su incondicional apoyo y numerosos consejos.

A mi familia, en especial a mis padres, por compartir conmigo mis ilusiones y motivarme a ser la persona que quiero llegar a ser.

Gracias a todos por estar.

Índice

Índice de figuras

Índice de tablas

1. Introducción

1.1. Antecedentes

El cáncer es una enfermedad caracterizada por el crecimiento y la proliferación descontrolada de células anormales en el cuerpo. Estas células anormales pueden formar tumores que invaden tejidos circundantes y, en etapas avanzadas, pueden diseminarse a otras partes del cuerpo, un proceso conocido como metástasis. Hoy en día, el cáncer es una de las principales causas de morbilidad y mortalidad en todo el mundo, siendo la razón por las que 113.660 personas perdieron la vida en el año 2021 en España [1] por lo que su estudio y tratamiento resulta de gran importancia para la medicina contemporánea.

Para el manejo del cáncer se emplean biomarcadores, que son indicadores, generalmente proteínas, que se producen en las células, tanto tumorales como no tumorales, en respuesta a un determinado estímulo. Ambos pueden detectarse en análisis de sangre, orina, en muestras del tumor, etc. Su presencia da información muy valiosa al equipo médico sobre el grado de malignidad del tumor, posibles tratamientos y demás.

Un biomarcador que resulta relevante es el PD-L1, por sus siglas en inglés *Programmed Death-Ligand 1*, el cual desempeña un papel crucial en el sistema inmunológico ya que cuando las células tumorales sobreexpresan PD-L1, inhiben al sistema inmunológico y por lo tanto evaden la respuesta. En otras palabras, PD-L1 permite que las células tumorales eviten ser atacadas por las células inmunológicas, lo que facilita su supervivencia y crecimiento [2]. Además, es el primer biomarcador aprobado para aplicar inmunoterapia tumoral en pacientes con cáncer de pulmón.

El estudio del estado de PD-L1 en células tumorales se ha convertido en un área crítica de investigación en oncología, ya que ayuda a identificar a los pacientes que pueden beneficiarse inmunoterapias específicas que bloquean la interacción de PD-L1 con su receptor en las células inmunológicas, lo que facilita que el sistema inmune ataque a las células tumorales.

Actualmente, este proceso se lleva a cabo por un patólogo experto que, tras realizarse una tinción sobre el tejido tumoral, este inspecciona el resultado y determina cuanta proteína de PD-L1 expresa la muestra. Este proceso en ocasiones puede resultar extenso, además del componente factor humano que afecta en el resultado. Por lo que, en los últimos años, resulta de especial interés la aplicación de técnicas computacionales que permitan clasificar el estado de PD-L1 del tumor empleando las imágenes de la tinción del tejido en una placa, de

forma que se acelere el proceso y se eliminen posibles errores derivados del factor humano. En este punto, destaca singularmente el aprendizaje profundo debido a las altas capacidades que ofrece.

El aprendizaje profundo, comúnmente conocido como *Deep Learning*, es una subdisciplina del aprendizaje automático o *Machine Learning* que se inspira en la estructura y el funcionamiento del cerebro humano para crear modelos computacionales llamados redes neuronales artificiales. A través de múltiples capas de unidades de procesamiento, la máquina puede aprender por sí sola de los errores y la información que recibe, siendo capaz de aprender y representar patrones y características complejas en datos, lo que las hace especialmente efectivas en tareas de reconocimiento de patrones y procesamiento de datos no estructurados, como imágenes, voz y texto.

Profundizando en los orígenes del *Deep Learning* se destacan principalmente 3 etapas:

- Etapa cibernética

Estos inicios arrancaron con estudios sobre el aprendizaje biológico, dando pie al entrenamiento de una única neurona, el perceptrón.

- Etapa de conexionismo

En esta época surgió el concepto de *backpropagation*, el fundamento que permite actualmente entrenar las redes neuronales y que éstas sean capaces de aprender por sí solas.

- Etapa como *Deep Learning*

Hoy en día, las redes neuronales artificiales no son una representación realista del cerebro biológico, por el contrario, mediante el *Deep Learning* se propone que las máquinas sean capaces de entender el mundo a través de una jerarquía de conceptos, de manera que cada concepto se defina a través de sus relaciones con conceptos más simples.

En su relación con la biomedicina, el aprendizaje profundo ha tenido un impacto significativo en áreas como el diagnóstico médico, la detección de enfermedades, la investigación de medicamentos y la personalización de tratamientos [3]. Algunas aplicaciones notables incluyen, entre otras, el diagnóstico por imágenes médicas, descubrimiento de fármacos, biología molecular, análisis de patrones genéticos en células tumorales, predicción de respuesta a tratamiento y detección temprana de cáncer.

1.2. Motivación

Desde hace tiempo se lleva desarrollando técnicas basadas en inteligencia artificial en diversos campos. Sin embargo, ha sido recientemente, con el avance de estas técnicas y el desarrollo de computadores con más capacidad, cuando la integración de sistemas inteligentes está suponiendo una transformación en la manera en que se abordan los desafíos.

Por otra parte, en el contexto de la oncología, la determinación del estado de PD-L1 en células tumorales es un componente crítico para la toma de decisiones clínicas en muchos pacientes. El inconveniente se halla en que este proceso se lleva a cabo mediante la interpretación visual de las imágenes de tejido por parte de patólogos expertos. Sin embargo, esta metodología manual puede resultar laboriosa, costosa y requiere tiempo, lo que a menudo retrasa la atención al paciente. También cabe destacar que en algunos casos la subjetividad inherente a la interpretación humana puede introducir variabilidad en los resultados y la no reproducibilidad de los resultados.

Por tanto, este Trabajo Fin de Máster consiste en el desarrollo de una red neuronal artificial para determinar el estado de la proteína PD-L1 en imágenes de células tumorales.

1.3. Objetivos

El principal objetivo del presente proyecto fin de Máster es el desarrollo y evaluación de una red neuronal especializada en el análisis de imágenes médicas para determinar el estado de una proteína específica, el PD-L1. Este trabajo se enmarca como un proyecto de investigación para el desarrollo y evaluación de la viabilidad de sistemas inteligentes en el tratamiento de datos clínicos con el fin de mejorar el diagnóstico y la toma de decisiones de expertos patológicos, en colaboración con el grupo de Investigación Traslacional en Inmunoterapia del Cáncer (CIMO2) de la Unidad de Oncología Intercentros e IBIMA.

Para el análisis se dispondrán de bases de datos cedidas por entidades de investigación en el ámbito de la inmunoterapia tumoral.

Así mismo, es necesario alcanzar una serie de objetivos para lograr con éxito el resultado final:

1. Documentación y estudio sobre los distintos modelos y técnicas empleadas para la inteligencia artificial.
2. Estudio de la naturaleza de los distintos marcadores de la inmunoterapia tumoral.

3. Diseño y entrenamiento de una red neuronal artificial capaz de analizar imágenes médicas.
4. Evaluación del desempeño de la metodología empleada con imágenes de PD-L1 para predecir el estado de PD-L1.
5. Evaluación el desempeño de la metodología empleada con imágenes de hematoxilina-eosina como variables indirectas para la predicción de PD-L1.

1.4. Metodología

Para la realización del presente proyecto se ha seguido SCRUM como base de metodología ágil. SCRUM se basa en una estructura de desarrollo incremental, es decir, el ciclo de desarrollo se fragmenta en pequeños proyectos con distintas etapas, de esta forma se realizan varias iteraciones que van aumentando valor al proyecto y haciéndolo posible.

Esta metodología se basa en el empirismo, esto es, todo el conocimiento se obtiene de la experiencia y las decisiones se toman con referencia a lo observado.

Además, un aspecto clave de esta metodología son las reuniones frecuentes mediante las que se revisa el desarrollo del proyecto y se planifican y desarrollan cada una de las etapas del mismo.

En la figura 1 se muestra de forma gráfica cómo se ha realizado el presente Trabajo Fin de Máster empleando la metodología SCRUM. Como se observa en la imagen, los requerimientos del proyecto se han dividido en requerimientos o proyectos de menor tamaño que se han desarrollado poco a poco han conformado el desarrollo del proyecto final.

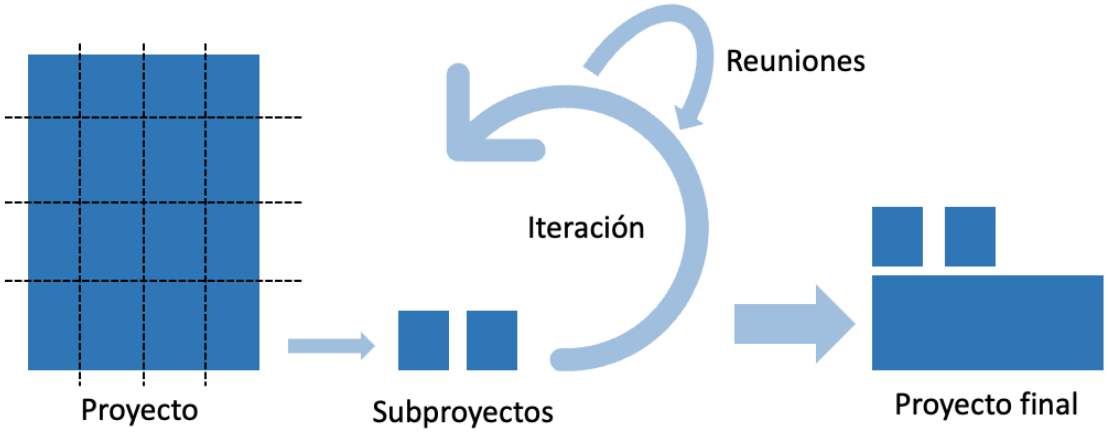


Figura 1: Metodología de trabajo SCRUM.

Aunque esta metodología se emplea sobre todo para el trabajo en equipo debido a las diversas reuniones y puesta en común, en este caso se ha optado por emplearla como metodología ágil debido las ventajas que aporta un seguimiento de los subproyectos que se van realizando, lo que permite visualizar poco a poco la exitosa realización del proyecto.

1.5. Estructura

La estructura del presente trabajo fin de Máster es la siguiente:

- En el capítulo 1 titulado **Introducción**, se incluyen los antecedentes, la motivación y los objetivos que se persiguen para el desarrollo del proyecto.
- En el capítulo 2 que lleva por nombre **Marco teórico**, se ha realizado un estudio previo sobre el funcionamiento y la base matemática tras los algoritmos y modelos de redes neuronales artificiales.
- En el capítulo 3 denominado **Materiales** se describen los materiales, bases de datos y softwares empleados para el desarrollo del proyecto.
- En el capítulo 4 titulado **Método** se exponen todos los procesos, métodos y técnicas aplicadas para lograr con éxito el resultado final.
- En el capítulo 5 nombrado **Resultados** se detallan los resultados obtenidos y sus respectivas evaluaciones.
- En el capítulo 6 denominado **Conclusiones y futuras líneas de desarrollo**, se exponen los objetivos y aprendizaje alcanzados gracias al desarrollo del presente proyecto y posibles propuestas de líneas de investigación para continuar como mejora del presente trabajo.

2. Marco teórico

En este apartado, se va a exponer de forma breve los conocimientos necesarios para entender cómo funcionan las redes neuronales artificiales sin profundizar en exceso ya que no es el objetivo de dicho proyecto.

Por su potencia y escalabilidad, las redes neuronales se han convertido en el modelo que define el aprendizaje profundo. Éstas se componen de neuronas, cada una de las cuales realiza individualmente un simple cálculo. La potencia de una red neuronal procede, en cambio, de la complejidad de las conexiones que pueden formar estas neuronas.

2.1. El perceptrón

Para comprender el funcionamiento de las redes neuronales artificiales, éstas se van a simplificar al perceptrón.

El perceptrón, mostrado en la figura 2, consiste en una única neurona artificial, es decir, una unidad de red neuronal.

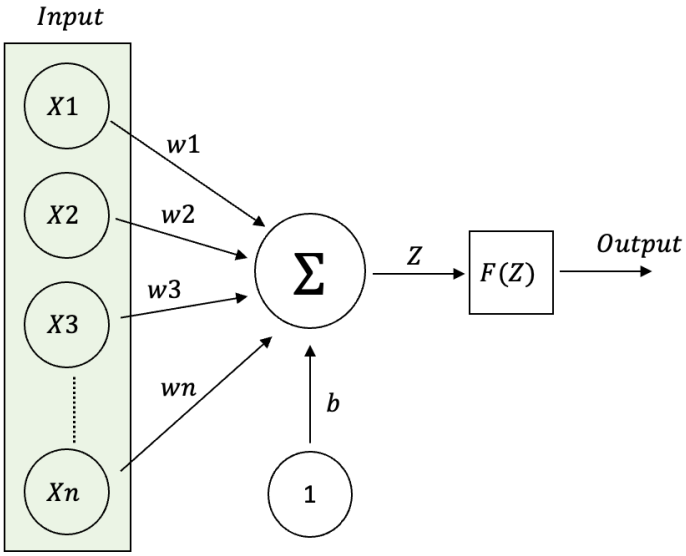


Figura 2: Diagrama del perceptrón.

De acuerdo con la figura 2, el funcionamiento del perceptrón es el siguiente. En primer lugar, las entradas (Xn) se multiplican por sus respectivos pesos asociados (wn) para posteriormente realizar un sumatorio de todas estas multiplicaciones. Cabe destacar que en el sumatorio se añade también un término denominado *bias* cuya entrada tiene valor 1, esto permite a la neurona modificar la salida independientemente de las entradas.

En este punto, la salida obtenida tras el sumatorio Z es lineal, por lo que el modelo no posee gran complejidad. Por tanto, se añade también una función F(Z) denominada función de activación. El objetivo de esta función es introducir al sistema no linealidad y, de esta forma, permite al modelo desarrollar operaciones más complejas y ajustar mejor los resultados aumentando la precisión [4].

Un ejemplo muy utilizado como función de activación es la unidad lineal rectificada o, más comúnmente conocida por sus siglas en inglés, ReLU. Esta función devuelve el máximo valor entre la entrada y cero, de esta forma, los valores negativos se ponen a cero.

Por tanto, tras lo anteriormente explicado, en la red neuronal de la figura 2, compuesta por una única neurona, el sumatorio sería el siguiente:

$$Z = X_1 w_1 + X_2 w_2 + X_3 w_3 + \cdots + X_n w_n + b$$

Si se emplea ReLU como función de activación, entonces la salida tendría el valor:

$$Output = F(Z) = \max\{Z, 0\}$$

Sin embargo, una red neuronal está formada por más de una neurona conectadas a las entradas o entre sí, lo que se conocen como capas. Varias capas conectadas entre sí es lo que se conoce en inglés como *fully connected layer*. Esto es lo que aporta a las redes neuronales su gran potencial. En la figura 3 se muestra de manera gráfica una red neuronal con una capa oculta de neuronas conectadas a las entradas y una capa de 2 neuronas de salida conectadas a la primera capa. En este caso la salida tiene 2 variables.

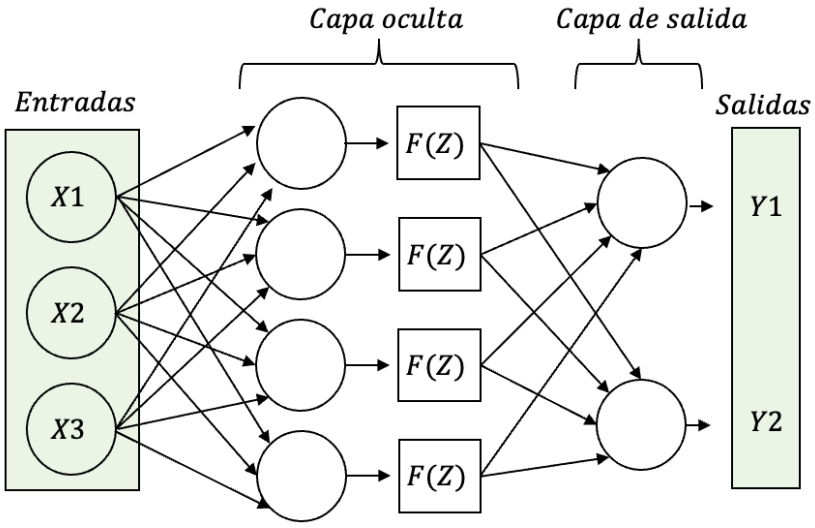


Figura 3: Diagrama de una capa totalmente conectada.

Este tipo de estructuras más complejas en la que cada capa cuenta con cientos o miles de neuronas son las que crean redes capaces de realizar complejas transformaciones y analizar los datos descubriendo patrones en ellos y obtener resultados más precisos [5].

Como se puede notar, es preciso que las entradas sean valores numéricos.

2.2. Redes neuronales convolucionales

Para el desarrollo del presente Trabajo Fin de Máster, se pretende analizar imágenes médicas. Para ello, previamente a la *fully connected layer*, es necesario añadir unas capas de neuronas que realizan una operación de convolución a partir de los valores numéricos de los píxeles de las imágenes, que actúan como entradas a la red [6]. La convolución se utiliza para extraer características de las imágenes y otros datos con estructura de cuadrícula. Tal y como se muestra en la figura 4, consiste en una operación matemática que puede entenderse como el deslizamiento de un filtro (también conocido como *kernel*) sobre la entrada [7], en este caso una imagen, y la multiplicación y suma de los valores en la región superpuesta para generar una nueva salida de dimensiones distintas conocida como mapa de características.

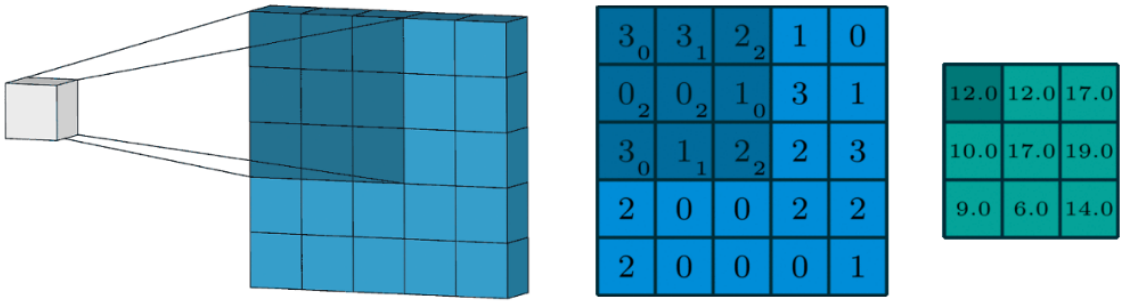


Figura 4: Ejemplo de operación de convolución.
(Fuente: Dumoulin, V., & Visin, F. (2016). A guide to convolution arithmetic for deep learning. arXiv preprint arXiv:1603.07285)

En la figura 5 se muestra de forma gráfica un ejemplo de red neuronal convolucional con la etapa de aprendizaje de características en donde se realiza las operaciones de convolución con los píxeles de las imágenes y, posteriormente, la etapa de clasificación en la que distintas capas de redes neuronales del tipo *fully connected layer* explicadas anteriormente, realizan una clasificación de las imágenes.

Además, como puede observarse en la imagen, se emplean también unas capas llamadas *Pooling* que sirven para reducir la resolución espacial de los mapas de características creados por las capas de convolución.

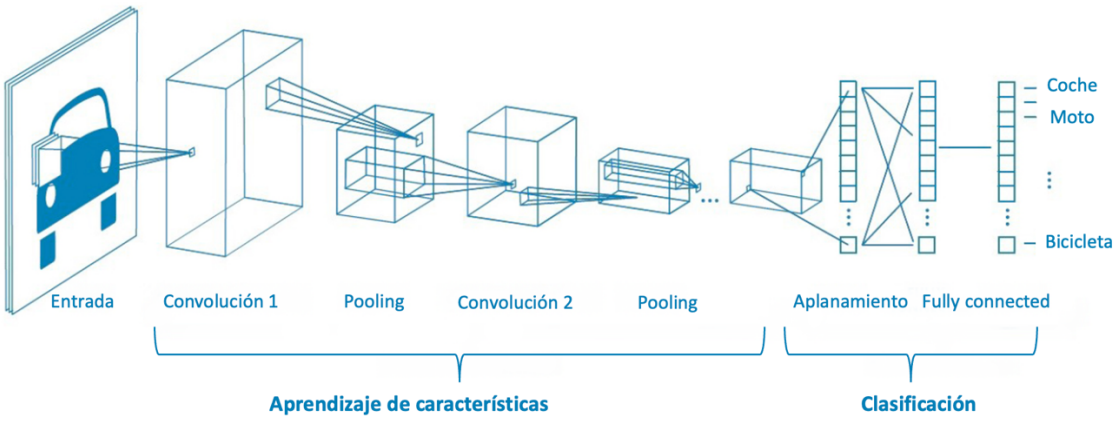


Figura 5: Ejemplo de una Red Neuronal Convolucional.
(Fuente: Hussain, M., Bird, J. J., & Faria, D. R. (2018). A Study on CNN Transfer Learning for Image Classification. Advances in Computational Intelligence Systems, 191–202. doi:10.1007/978-3-319-97982-3_16)

2.3. Proceso de aprendizaje

En una red neuronal artificial, el proceso de aprendizaje consiste en tratar de encontrar los pesos asociados a cada conexión que conlleven una salida lo más cercana posible a lo correcto.

Para ello, el sistema comienza con unos valores para los pesos aleatorios, con los que se obtiene un valor de salida. Posteriormente, se utiliza una función, denominada función de coste o función de pérdida, para evaluar la discrepancia entre la salida del sistema y el valor real, un ejemplo de esta función puede ser la entropía cruzada (*cross-entropy*) para una tarea de clasificación o el error cuadrático medio (MSE) para una tarea de regresión. La función de coste cuantifica cuán alejadas están las predicciones del modelo de los valores reales en el conjunto de entrenamiento.

El objetivo del proceso de entrenamiento es minimizar esta función de coste, lo que implica ajustar los pesos de la red. Esto se logra utilizando técnicas de optimización, siendo una de las más comunes el descenso de gradiente. En el descenso de gradiente, se calcula el gradiente de la función de coste con respecto a los pesos y se ajustan gradualmente los valores de los pesos en la dirección que reduce la pérdida. Para mejor entendimiento, en la figura 6 se muestra un ejemplo gráfico de cómo funciona la técnica de optimización. Puesto que se trata de minimizar la función de coste, se trata de localizar un mínimo en dicha función

de coste y obtener de ahí los valores de los pesos. Para ello se tiene un parámetro, la tasa de aprendizaje α, que establece cómo de rápido se produce dicha optimización.

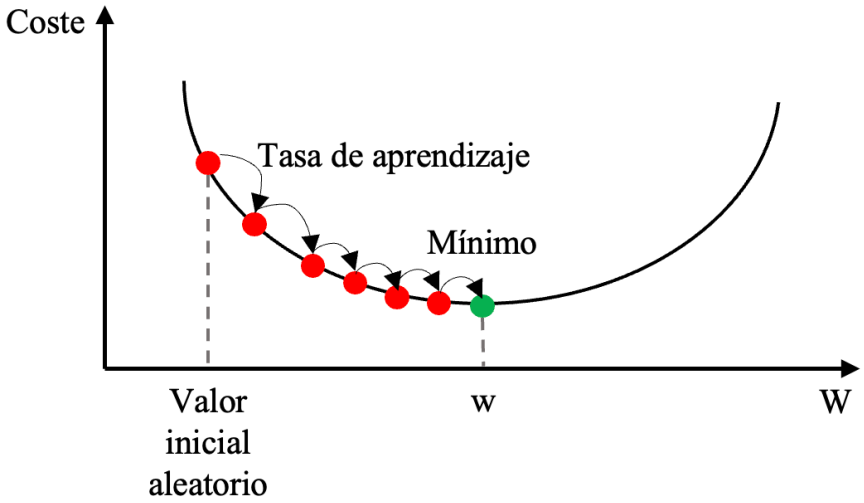


Figura 6: Optimización de la función de coste.

El proceso de optimización se realiza iterativamente a lo largo de múltiples épocas. En cada época, el conjunto de datos de entrenamiento se pasa a través de la red, y se actualizan los pesos para minimizar la función de coste. Este proceso se repite hasta que se alcance un criterio de detención predefinido, como un número máximo de épocas o cuando la pérdida converge a un valor mínimo.

A medida que la red se entrena, se ajustan los pesos de las conexiones entre las neuronas de las diferentes capas en lo que se conoce como algoritmo de *backpropagation*, que consiste en recorrer hacia atrás en la red actualizando los valores de dichos pesos [8]. La red aprende a reconocer patrones y características en los datos, y a mejorar su capacidad de hacer predicciones precisas en función de la retroalimentación proporcionada por la función de coste.

Este proceso de entrenamiento puede resultar intensivo en términos computacionales y requerir grandes conjuntos de datos para obtener buenos resultados. Además, la selección de la función de coste, la tasa de aprendizaje y otros hiperparámetros puede influir en la velocidad y el éxito del entrenamiento.

Otro aspecto importante durante el proceso de entrenamiento de un modelo de red neuronal, son las divisiones o particiones que se hacen de la base de datos disponible. En este caso, el conjunto de datos se divide en 3 subconjuntos, el de entrenamiento, el de validación y el de prueba.

Por un lado, el conjunto de entrenamiento se utiliza para entrenar al modelo para que aprenda de las imágenes, en cambio, el de validación, se emplea para ir evaluando la actuación o el desempeño del modelo a medida que éste va

aprendiendo. Esto se debe a que cuando el modelo aprende del conjunto de entrenamiento, es posible que aprenda patrones presentes en dicho conjunto que son irrelevantes y que, por tanto, no están presentes en el resto de conjunto de datos, lo que puede significar que el modelo luego no generalice de forma correcta lo que aprenda del conjunto de entrenamiento, aportando una falsa apariencia de aprendizaje. Por tanto, se emplea el conjunto de validación para evaluar el desempeño del modelo a medida que éste aprende.

También cabe destacar que, debido a las limitaciones computacionales, se suele "alimentar" al modelo con los datos de entrenamiento agrupados por lotes, ya que si éstos se entregan de una sola vez puede que ser que el ordenador encargado del entrenamiento no tenga recursos suficientes para procesarlo y provoque la interrupción del proceso.

Finalmente, el conjunto de prueba o test se utiliza una vez el modelo ha sido entrenado para evaluar el desempeño final de este en un conjunto de datos distinto. En este punto, se emplean métricas de evaluación, explicadas en el apartado 4.2, que muestran cómo es la actuación del modelo.

2.4. Curvas de aprendizaje

Las curvas de aprendizaje son gráficos que muestran la evolución, durante la etapa de entrenamiento, de las métricas empleadas para entrenar el modelo. Exponen visualmente cómo ha sido el proceso de aprendizaje del modelo y presentan a primera vista el desempeño de este [9].

Normalmente, para tareas de clasificación, se utilizan 2 tipos de curvas de aprendizaje, la de pérdida y la de precisión.

Curva de pérdida

La curva de pérdida muestra en la etapa de entrenamiento, la evolución de la función de pérdida del modelo.

La función de pérdida, explicada en el apartado 2.3, evalúa la salida obtenida del sistema respecto del valor real, indicando cuánto se alejan las predicciones del modelo de los datos reales.

Durante el proceso de entrenamiento, el objetivo principal es minimizar esta función de pérdida, buscando que el modelo haga predicciones que se acerquen lo más posible a los resultados correctos. Inicialmente, la curva de pérdida suele tener valores altos, ya que el modelo en este punto no tiene conocimiento sobre los datos y realiza predicciones aleatorias o muy inexactas. A medida que el modelo se entrena, la curva de pérdida tiende a descender. Cuando esta curva

disminuye gradualmente y se estabiliza, convergiendo a un valor, indica que el modelo ha aprendido efectivamente de los datos y hace predicciones más precisas.

En la figura 9, se muestra un ejemplo de curva de pérdida.

Curva de Precisión

Junto a la curva de pérdida, la curva de precisión es otro gráfico relevante en el entrenamiento de modelos de clasificación. Mientras que la curva de pérdida se enfoca en la medida de error, la curva de precisión evalúa la calidad de las predicciones del modelo en términos de clasificación correcta. La precisión es una métrica que cuantifica la proporción de ejemplos clasificados correctamente por el modelo en relación con el total de ejemplos.

A medida que el modelo se entrena, la curva de precisión muestra cómo evoluciona su capacidad para clasificar los datos de manera precisa. Inicialmente, la precisión puede ser baja, ya que el modelo no tiene suficiente conocimiento para hacer predicciones y por tantos estas carecen de precisión. Con el tiempo, a medida que el modelo aprende y ajusta sus parámetros, la curva de precisión tiende a aumentar. Una curva de precisión ascendente indica que el modelo está volviéndose más competente en su tarea de clasificación.

2.5. Problemas durante el entrenamiento

Cuando se crean modelos de *Machine learning* y *Deep learning*, existen dos problemas que suelen ocurrir con frecuencia [10] y que, por tanto, hay que conocerlos en profundidad para saber cómo afrontarlos y evitarlos para que el modelo tenga el desempeño esperado.

Underfitting

Underfitting o subajuste, se da cuando el modelo creado no es lo suficientemente complejo, es decir, no tiene suficientes parámetros o cuando la base de datos es pequeña. En este caso, el modelo no se termina de ajustar de forma correcta a los datos de entrenamiento y, por tanto, presenta un rendimiento deficiente tanto en los datos de entrenamiento como en los de prueba y no es capaz de capturar los patrones presentes en los datos.

En la figura 7 se muestra un ejemplo de un modelo que aplica una regresión lineal para predecir unos valores, uno presenta *underfitting* y el otro se considera "correcto". Como puede observarse, en el modelo con *underfitting*, su línea no se ajusta de forma correcta a los datos, mientras que en el modelo "simplemente correcto" presenta un mejor comportamiento.

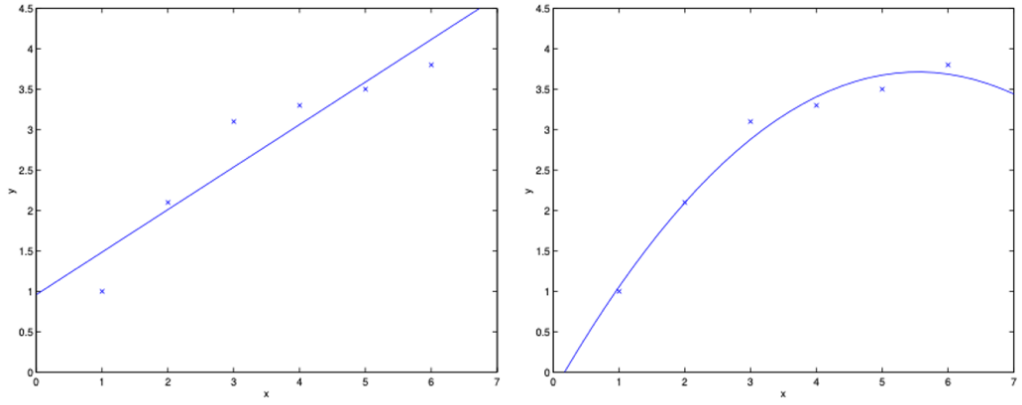


Figura 7: Comparación modelo con *underfitting*.
(a) Modelo con *underfitting*. (b) Modelo correcto.
(Fuente: Andrew Ng, Tengyu Ma. CS229 Lecture Notes, Stanford University (2023). https://cs229.stanford.edu/main_notes.pdf)

Overfitting

Por otro lado, existe otro problema de índole contraria, el *overfitting* o sobreajuste. Esto se da cuando el modelo es demasiado complejo o cuando el conjunto de datos de entrenamiento es pequeño y no presenta suficiente variabilidad. Entonces, lo que ocurre es que el modelo ajusta sus parámetros en exceso a los datos de entrenamiento y luego no es capaz de generalizar correctamente, es decir, no posee un desempeño correcto en el resto de los datos de prueba, sólo en el de entrenamiento.

En la figura 8 se muestra el mismo ejemplo del caso anterior pero esta vez el modelo está sobreajustado.

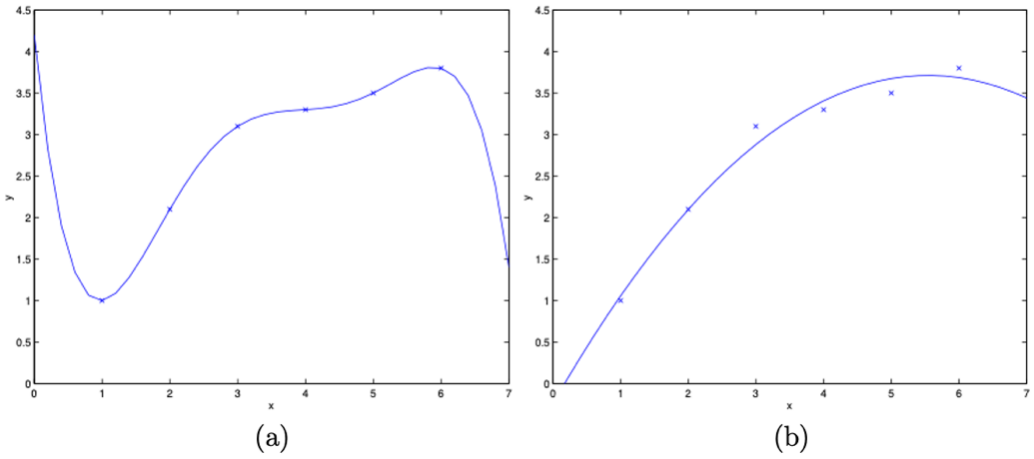


Figura 7: Comparación de modelo con *overfitting*.
(a) Modelo con *overfitting*. (b) Modelo correcto.
(Fuente: Andrew Ng, Tengyu Ma. CS229 Lecture Notes, Stanford University (2023). https://cs229.stanford.edu/main_notes.pdf)

En el caso particular de redes neuronales, estos dos posibles comportamientos se pueden identificar observando las curvas de aprendizaje de pérdida, tal y como se explica a continuación.

Como se comentó en el apartado 2.3, en el proceso de entrenamiento se utiliza el conjunto de entrenamiento para que el modelo aprenda y el de validación para ir comprobando su actuación durante dicho proceso.

En primer lugar, en la figura 9, se muestra la curva de aprendizaje de un modelo que es entrenado durante 50 épocas y que no presenta ni *overfitting* ni *underfitting*, es decir, muestra un buen ajuste.

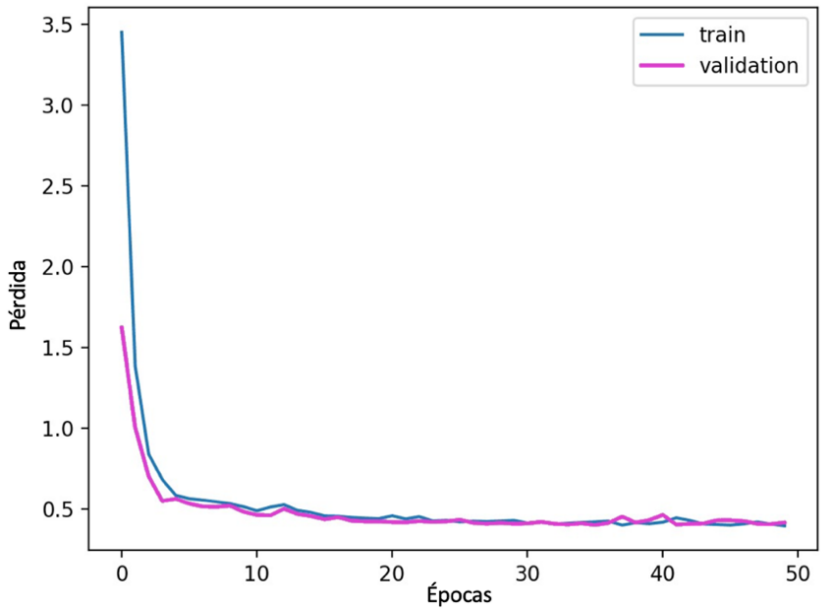


Figura 8: Curva de aprendizaje de pérdida correcta.

Como puede observarse, tanto para el conjunto de entrenamiento como el de validación, la función de perdida va disminuyendo de manera coherente y paralela a medida que avanza el proceso de entrenamiento. Esta convergencia simultánea de las curvas de pérdida sugiere que el modelo está aprendiendo de manera efectiva de los datos de entrenamiento y, al mismo tiempo, puede generalizar bien a datos no vistos. Por lo que este comportamiento refleja un equilibrio adecuado entre la capacidad del modelo para ajustarse a los datos de entrenamiento y su habilidad para evitar el *overfitting* al mantener un rendimiento sólido en el conjunto de validación.

Por otro lado, en la figura 10, se muestran las curvas de pérdidas de un modelo con *underfitting*.

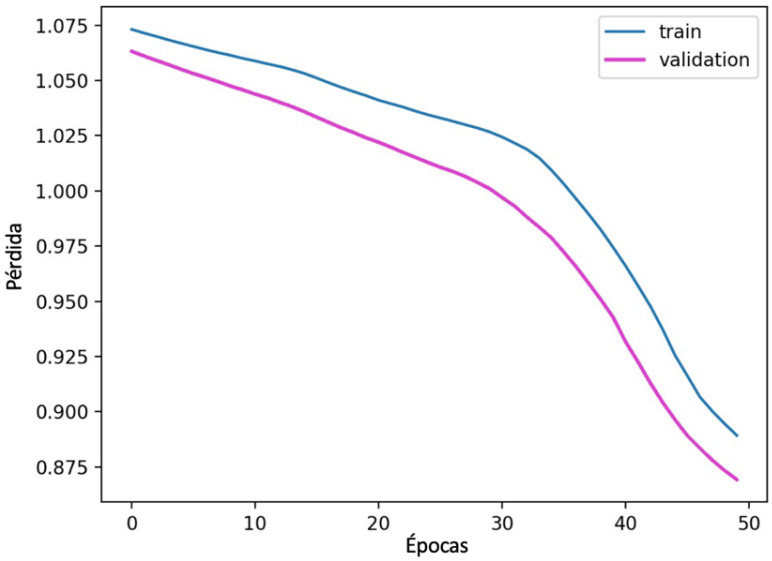


Figura 9: Curva de pérdida con *underfitting*.

Como puede observarse en la gráfica, la función de pérdida de ambos conjuntos posee una tendencia de bajada sin converger a un valor, lo que sugiere que le falta aprendizaje y que, por tanto, se encuentra subajustado.

Finalmente, en la figura 11, se muestra un modelo con *overfitting*.

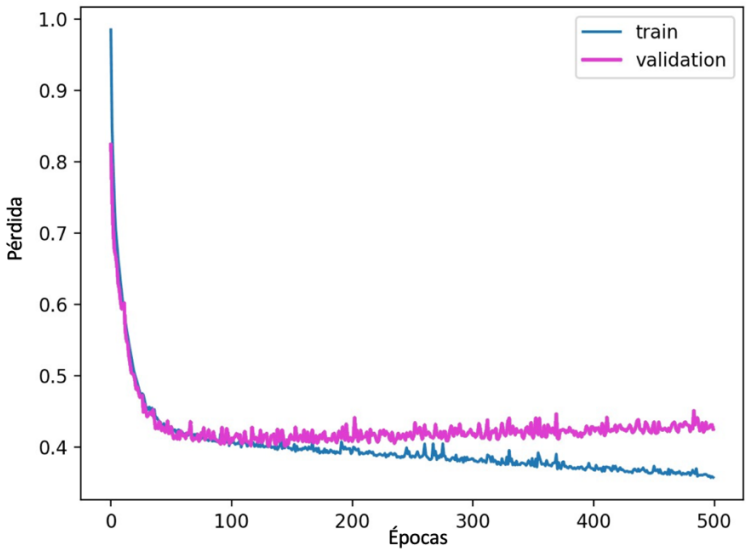


Figura 10: Curva de pérdida con *overfitting*.

En este caso, durante las 500 épocas de entrenamiento, la pérdida del conjunto de entrenamiento ha ido disminuyendo mientras que en el del conjunto de validación, ha llegado a un punto que ha dejado de disminuir he incluso ha tenido un cierto aumento. Esto indica que el modelo ha aprendido patrones presentes en los datos de entrenamiento que son irrelevantes para la tarea en cuestión y, por tanto, al analizar el conjunto de validación no muestra signos de que continue aprendiendo realmente.

3. Materiales

3.1. Base de datos

Para la realización del presente trabajo se ha empleado una base de datos generada en el contexto del proyecto de investigación PI18/01592, aprobado por el Comité Provincial Ético de Málaga (nº 26/10/2017), concedido al grupo de Investigación Traslacional en Inmunoterapia del Cáncer (CIMO2) de la Unidad de Oncología Intercentros e IBIMA. Este set de datos se ha denominado CIMO2.

Esta base de datos está compuesta por 136 imágenes de células tumorales de pacientes. En concreto, se tienen imágenes con tinción hematoxilina-eosina y PD-L1. La tinción hematoxilina-eosina es un método que supone la aplicación de la tinción de hematoxilina, que, por ser catiónica o básica, tiñe estructuras ácidas (basófilas) en tonos azul y púrpura, como por ejemplo los núcleos celulares y, el uso de eosina que tiñe componentes básicos (acidófilos), como, por ejemplo, estructuras citoplasmáticas y sustancias intercelulares, en tonos de color rosa. En la figura 12 se muestra un ejemplo de este tipo de imágenes.

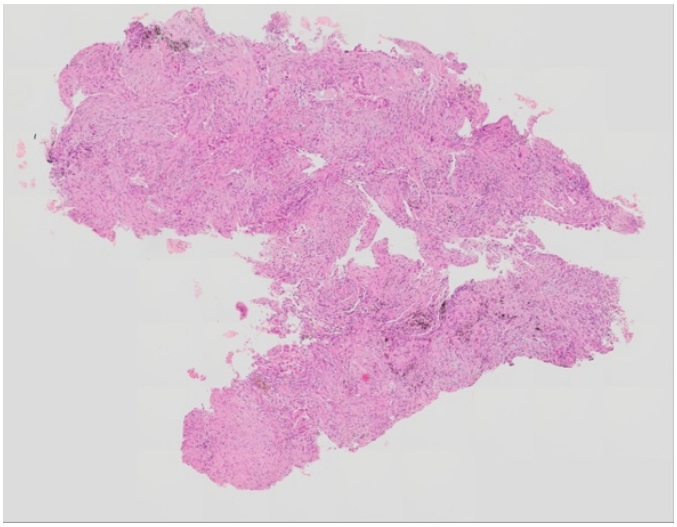


Figura 11: Imagen de muestra de Hematoxilina-eosina.

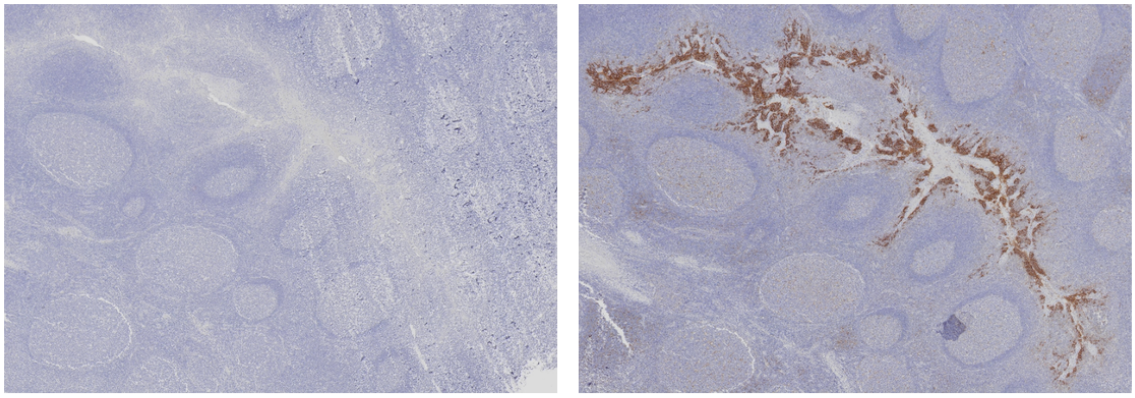
En el caso de las imágenes de PD-L1, éstas exponen cuánta expresión de la proteína PD-L1 tiene la muestra del paciente. De esta forma, un patólogo especializado observa la imagen y determina el estado de PD-L1 empleando el siguiente baremo:

- **PD-L1 negativo**: menos del 1% de las células cancerígenas expresan PD-L1.
- **PD-L1 bajo**: entre el 1 y el 49% de las células cancerígenas expresan PD-L1.

- **PD-L1 alto**: más de la mitad (50%) de las células cancerígenas expresan PD-L1.

Por lo que, entre los objetivos se encuentra que el modelo que se desarrolle sea capaz de clasificar cada imagen entre una de las 3 categorías anteriores y emplear tanto imágenes de hematoxilina-eosina como de PD-L1 para predecir el estado de éste.

En la figura 13 se muestra un ejemplo de imagen de control negativo de PD-L1 y la misma imagen con tinción de PD-L1.



(a) (b)

Figura 12: Muestra de tinción PD-L1.
(a) Células del control negativo de la tinción PDL1. (b) Células tras la prueba de PDL1.

Manejo de aspectos éticos y privacidad

Al tratar con información de pacientes, es necesario asegurar la privacidad de estos.

Por tanto, la presente propuesta ha cumplido las normas éticas y los códigos de conducta profesional nacionales, de la UE y códigos de conducta profesional, y ha seguido la Declaración de Helsinki de la Asociación Médica Mundial (2002), la Asociación Médica Mundial, las Directrices de Buenas Prácticas Clínicas (1996), el "Convenio de Oviedo Convenio del Consejo de Europa para la Protección de los Derechos Humanos y la Dignidad del Ser Humano con respecto a la Aplicación de la Biología y la Medicina (2005), Directrices éticas internacionales para la investigación biomédica en seres humanos (2002).

Por lo que, respecto al set de datos CIMO2, toda información, referida tanto a datos personales como clínicos, que se ha recogido durante el curso de la investigación, ha sido almacenada de manera estrictamente confidencial, de conformidad con lo establecido en la regulación vigente Andaluza, Española y Europea (ver Consentimiento Informado). Los registros médicos no se harán

públicos en ningún momento. La información ha sido anonimizada, de manera que un código asignado sirve para su identificación. La información recogida sólo se ha utilizado para fines de estudio. Incluso si los resultados del estudio son publicados, la identidad siempre permanecerá confidencial. Las muestras y datos recogidos no podrán ser utilizados con fines comerciales.

Por otro lado, tras realizar una documentación sobre la determinación de PD-L1 aplicando *Deep learning*, se encontró un artículo en el que se desarrolla un modelo que determina si una muestra es positiva o negativa en PD-L1 a partir de imágenes de Hematoxilina-eosina [11]. Esto resulta de gran interés ya que actualmente esta tarea la realizan patólogos expertos y para ello requieren de imágenes con tinción PD-L1, por lo que el proceso resulta más largo y laborioso. Se contactó con los autores de dicho artículo y se consiguió acceso a una parte de la base de datos utilizada en dicho artículo.

Esta es una base de datos de 2528 imágenes del Centro de Evaluación de Patología Genética (GPEC) consistente en una librería pública de tejidos microarray. En este caso, se trata únicamente de imágenes con tinción hematoxilina-eosina y el estado de PD-L1 se clasifica en positivo o negativo únicamente, por lo que no se distinguen 3 clases como las imágenes de nuestro set de datos (CIMO2).

3.2. Software

Para analizar la base de datos y realizar los modelos de redes neuronales se va a emplear Python como lenguaje de programación. Se ha optado por este lenguaje principalmente por la gran cantidad de librerías que posee especializadas en técnicas de *Machine learning* y *Deep learning* [12].

También se ha hecho uso del supercomputador Picasso el cual es uno de los nodos que forman la Red Española de Supercomputación (RES), ya que, para entrenar modelos de *Deep learning* que requieran altos recursos computacionales, es necesario utilizar unidades de procesamiento gráfico o GPU con suficiente almacenamiento RAM.

Las principales características de este superordenador son que actualmente dispone de 4096 cores, 23 TB de RAM y aproximadamente 6.5 PB de almacenamiento. Todo unificado con una red InfiniBand de baja latencia y un único sistema de colas:

- Cluster de 768 cores Intel y 3TB de RAM y 32 tarjetas GPU Tesla M2075. Red IB 56Gbps.

- Cluster de 7 máquinas Intel con 80 procesadores y 2 TB de RAM cada una. 560 cores y 14TB de RAM en total. Red IB 56Gbps.
- Cluster de 2668 cores Intel y 5.4 TB de RAM. Red IB 40Gbps.
- Almacenamiento compartido (GPFS) con 550 TB netos por InfiniBand 56 Gbps.
- Almacenamiento a largo plazo de alta redundancia con 6 PB brutos de Object Store por Ethernet 10G.

El autor agradece al Centro de Supercomputación y Bioinnovación (SCBI) de la Universidad de Málaga la provisión de recursos computacionales (el superordenador Picasso) y el soporte técnico (www.scbi.uma.es/site).

4. Método

En este apartado se procede a explicar con detalle todas las técnicas aplicadas y el proceso seguido para el desarrollo del trabajo.

4.1. Reconocimiento óptico de caracteres

Un Reconocimiento Óptico de Caracteres (OCR) es un sistema dedicado a la detección y extracción de texto en imágenes [13]. Su funcionamiento se basa en emplear un escáner para procesar la forma física de un documento o imagen. Posteriormente, convierte lo escaneado en una versión en dos colores o en blanco y negro. La imagen o el mapa de bits escaneados se analizan en busca de zonas claras y oscuras, y las zonas oscuras se identifican como caracteres que hay que reconocer, mientras que las zonas claras se identifican como fondo. A continuación, las zonas oscuras se procesan para encontrar letras alfabéticas o dígitos numéricos. En esta etapa se suele seleccionar un carácter, una palabra o un bloque de texto cada vez. A continuación, los caracteres se identifican mediante un reconocimiento de patrones.

En este proyecto, se ha empleado un OCR para asociar las imágenes en la base de datos CIMO2 con su respectivo etiquetado en el archivo Excel de datos. En el apartado 5.1. se explica de mejor forma por qué esto ha sido necesario y en qué ha consistido.

En este proyecto, se ha empleado usando la librería de Python “easyocr”.

4.2. Preprocesamiento y aumento de datos

Es muy importante tener en cuenta que, a la hora de entrenar una red neuronal, ésta muestra un mejor desempeño cuando se entrenan con una cantidad de datos lo suficientemente grande y variada [14].

Por tanto, antes de entrenar una red neuronal convolucional, resulta buena práctica aplicar algunas técnicas de preprocesamiento y aumento de datos a las imágenes principalmente por dos motivos:

- Cuando se tiene una base de datos pequeña, esto sirve para obtener nuevas imágenes a partir de las que ya se disponen. De esta forma es posible ampliar en cierta medida la base de datos.
- El uso de estas técnicas aporta variabilidad a las imágenes, evitando así que el modelo sobreajuste sus parámetros a los datos de entrenamiento. El sobreajuste ocurre cuando un modelo ha aprendido

mucho los patrones presentes en el conjunto de datos de entrenamiento, por lo que luego no da buenos resultados al resto de conjunto de datos. En el apartado 2.5 se explica en mayor profundidad el concepto de sobreajuste.

Además, es importante tener en cuenta la escasez de disponibilidad de imágenes en el ámbito médico debido a diversos factores como la escasez de equipamiento necesario, privacidad de los pacientes e incapacidad para conseguir imágenes con ciertos criterios entre otros [15]. Por tanto, en este sector resulta casi imperativo aplicar estas técnicas para aumentar en cierta medida estos datos.

En este proyecto, además de giros y volteos, se van a emplear las siguientes técnicas de aumento de datos: giros, volteos, corrección gamma, modificación de la temperatura y la saturación, adición de ruido Gaussiano, y aumento del desenfoque.

Corrección gamma

Esta técnica se emplea para ajustar el brillo y el contraste de las imágenes. Se basa en una corrección no lineal que se aplica a los valores de brillo de una imagen de acuerdo con la siguiente expresión.

$$V_{salida} = A \cdot V_{entrada}^{\gamma}$$

En dicha expresión, 'A' se corresponde con una constante y V con los valores de brillo de la imagen.

En la figura 14 se muestra qué efectos tiene la corrección gamma para valores de gamma mayores y menores de 1.

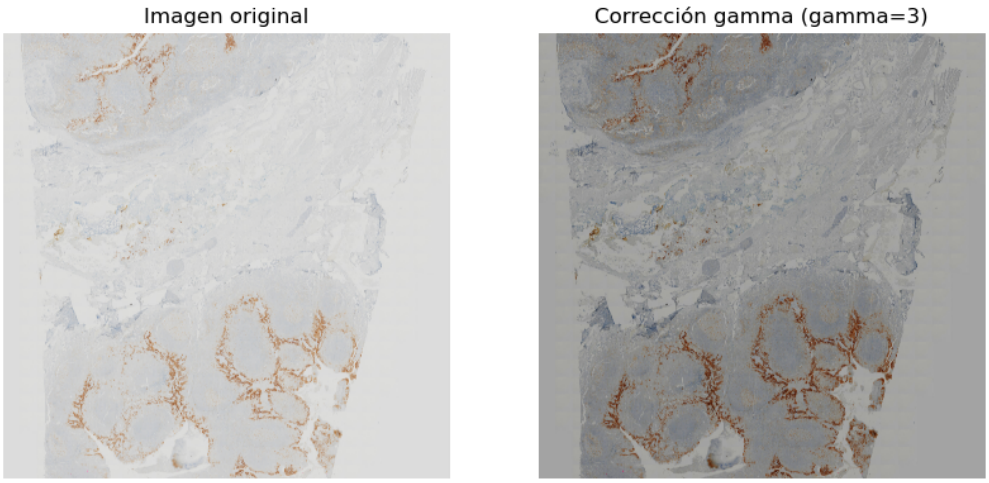


Figura 13: Efecto de la corrección Gamma.

Modificación de la temperatura y la saturación

La temperatura de color en las imágenes se refiere a la apariencia cálida o fría de la luz. La modificación de la temperatura de una imagen se basa en el ajuste de la temperatura de color de la luz que ilumina la escena. Esto se logra aplicando una transformación a los canales de color de la imagen (RGB) para simular diferentes fuentes de luz o condiciones de iluminación. Las transformaciones más comunes incluyen el aumento o la disminución de la temperatura de color para simular iluminación cálida o iluminación fría.

Por otro lado, la saturación es la intensidad de un matiz específico y se basa en la pureza del color. Una imagen con saturación completa tiene colores vivos y puros, mientras que una imagen con baja saturación aparece en tonos de gris.

La modificación de la saturación implica ajustar los valores de saturación en los canales de color de una imagen (RGB) para aumentar o disminuir la intensidad de los colores en la imagen.

En las figuras 15 y 16 se muestra qué efecto tiene la modificación de la temperatura y la saturación en una imagen.

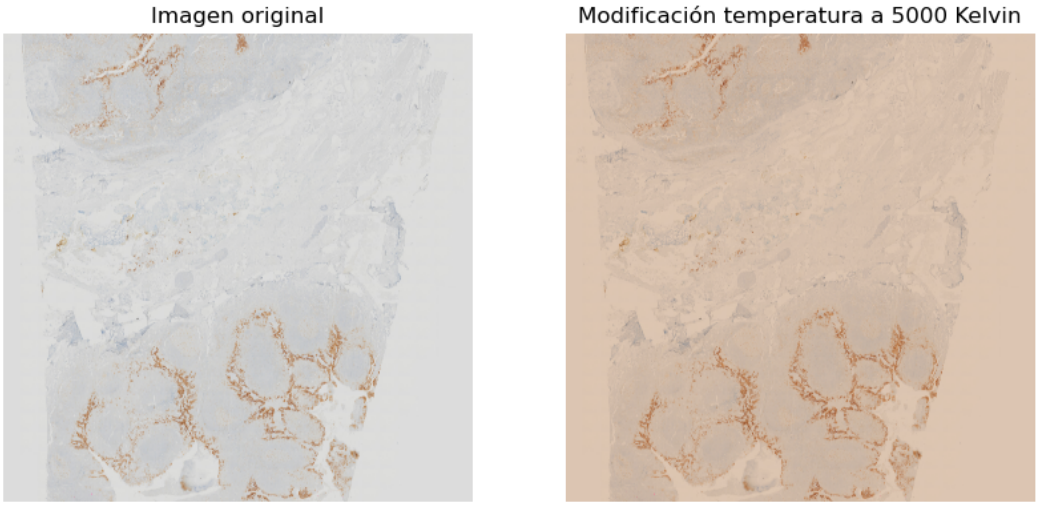


Figura 14: Efecto de la modificación de la temperatura.

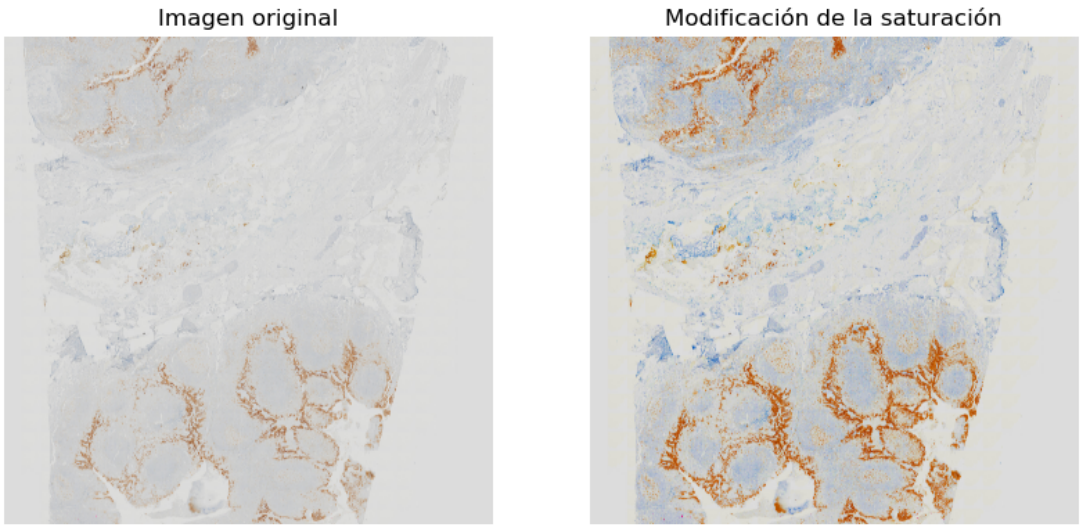


Figura 15: Efecto de la modificación de la saturación.

Ruido Gaussiano

Añadir ruido gaussiano a una imagen se emplea para introducir variaciones aleatorias en los valores de píxeles de una imagen utilizando una distribución gaussiana o normal. Para cada píxel de la imagen original, se agrega un valor aleatorio muestreado de una distribución gaussiana centrada en cero con una desviación estándar específica. La magnitud de la desviación estándar controla la cantidad de ruido que se añade a la imagen. Cuanto mayor sea la desviación estándar, mayor será el ruido agregado.

En la figura 17 se muestra el ejemplo de una imagen con ruido gaussiano.

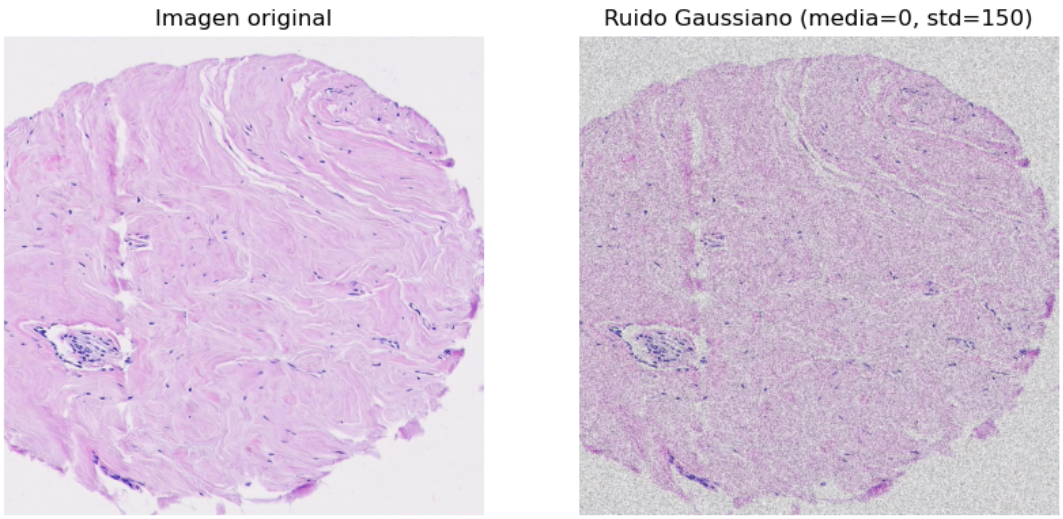


Figura 16: Efecto de la adición de ruido Gaussiano.

Aumento del desenfoque

El aumento del desenfoque de una imagen consiste en aplicar un filtro de desenfoque a la imagen original de forma que se suavicen los detalles y bordes de la imagen al promediar los valores de píxeles en regiones vecinas. De esta forma se difuminan sus detalles y se reduce la nitidez.

En la figura 18 se muestra cómo afecta a una imagen el aumento del desenfoque.

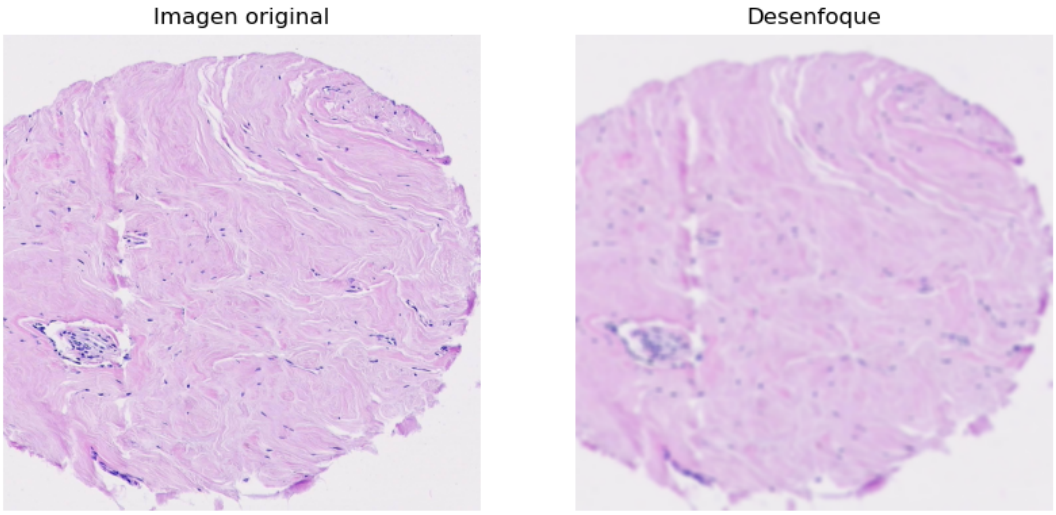


Figura 17: Efecto del aumento del desenfoque.

Para el desarrollo del presente proyecto, se han empleado las técnicas de preprocesamiento anteriormente descritas junto con giros y volteos de imágenes. De esta forma, se logra ampliar la base de datos en caso de ser necesario, obteniendo nuevas imágenes modificadas a partir de las originales, y también se obtiene un conjunto de imágenes de entrenamiento con una mayor variabilidad evitando que el modelo de red neuronal aprenda patrones irrelevantes presentes en el conjunto de datos de entrenamiento.

4.3. Métodos estadísticos

Para evaluar la actuación de los modelos de redes neuronales, se van a emplear los siguientes métodos estadísticos.

Matriz de confusión

La matriz de confusión es una herramienta que muestra una clasificación en forma de matriz de las salidas obtenidas por un modelo predictivo. En dicha matriz se muestran los siguientes datos:

- Verdaderos negativos (VN).
- Falsos negativos (FN).
- Verdaderos positivos (VP).
- Falsos positivos (FP).

Estos valores se encuentran dispuestos de la siguiente forma en la matriz de confusión.

$$\begin{bmatrix} VN & FP \\ FN & VP \end{bmatrix}$$

A partir de la matriz de confusión, se pueden calcular las siguientes medidas estadísticas para evaluar el desempeño del modelo.

Exactitud

La exactitud de un modelo se refiere a lo cerca que está el resultado del valor verdadero. Representa la proporción de predicciones correctas sobre el total de predicciones. Por tanto, se calcula de la siguiente forma:

$$Exactitud = \frac{VP + VN}{VP + FP + VN + FN}\ x\ 100$$

La exactitud comprende valores en porcentaje de 0% a 100% siendo 0% que todos los resultados son erróneos y 100% que todos son correctos.

Precisión

La precisión de un modelo se refiere a la capacidad de este para hacer predicciones correctas o verdaderas entre todas las predicciones que realiza. Es una de las métricas clave utilizadas en problemas de clasificación y proporciona información sobre la calidad de las predicciones positivas del modelo. Además, resulta útil cuando se busca minimizar los falsos positivos.

Representa la fracción de predicciones positivas correctas entre todas las predicciones positivas, por lo que se calcula siguiendo la siguiente ecuación:

$$Precisión = \frac{VP}{VP + FP}\ x\ 100$$

Sensibilidad y Especificidad

La sensibilidad mide la capacidad de un modelo para identificar de manera correcta los casos positivos de entre todos los casos positivos reales en el conjunto de datos.

Se calcula de acuerdo con la siguiente expresión:

$$Sensibilidad = \frac{VP}{VP + FN}\ x\ 100$$

En porcentaje, se expresa como un valor entre 0% y 100%, donde 100% significa que el modelo identifica correctamente todos los casos positivos, y 0% que el modelo pasa por alto todos los casos positivos.

Por otro lado, la especificidad mide la capacidad de un modelo para identificar de manera correcta los casos negativos de entre todos los casos negativos reales en el conjunto de datos.

Se calcula de acuerdo con la siguiente expresión:

$$Especificidad = \frac{VN}{VN + FP}\ x\ 100$$

En porcentaje, se expresa como un valor entre 0% y 100%, donde 100% significa que el modelo identifica correctamente todos los casos negativos, y 0% que el modelo pasa por alto todos los casos negativos.

F1 score

La puntuación F1 o más comúnmente conocido como F1-score, combina las métricas de precisión y sensibilidad en un solo valor para evaluar el rendimiento del modelo. Resulta útil cuando se desea equilibrar la importancia de ambas métricas y cuando se busca una medida única que refleje la calidad general de las predicciones del modelo.

Se calcula siguiendo la siguiente ecuación:

$$F1 - score = \ 2 \cdot \frac{Precisión \cdot Sensibilidad}{Precisión + Sensibilidad}$$

El F1-Score se encuentra en un rango de valores de 0 a 1, donde 1 representa un F1-Score perfecto, indicando un modelo que tiene tanto una alta precisión como una alta sensibilidad. Cuanto más cercano esté el F1-Score a 1, mejor será el rendimiento del modelo en términos de equilibrar la capacidad de hacer predicciones precisas y la capacidad de identificar correctamente los casos positivos.

El F1-Score es especialmente útil en situaciones donde hay un desequilibrio entre las clases, en cuyo caso, constituye una métrica equilibrada para evaluar el rendimiento del modelo.

Curva ROC y AUC

La curva ROC (*Reciever Operating Characteristic*) sirve para determinar la capacidad discriminativa del modelo [16]. Tal y como se muestra en la figura 19, consiste en una representación gráfica de cómo un modelo clasifica los datos en función de diferentes umbrales de decisión. En el eje X, se representa la tasa de falsos positivos (FP) y, en el eje Y, se representa la tasa de verdaderos positivos (VP) o sensibilidad. Cuanto mayor desplazada hacia arriba sea la curva, mejor ya que indica que se tiene un mayor número de verdaderos positivos que falsos positivos.

Para cuantificar el resultado de la curva ROC y obtener así un valor que indique la capacidad discriminatoria del modelo, se emplea el AUC o área bajo la curva. Este es el valor de la integral bajo la curva ROC. En la figura 19, se muestran ejemplos de distintos modelos con sus respectivas curvas ROC y AUC.

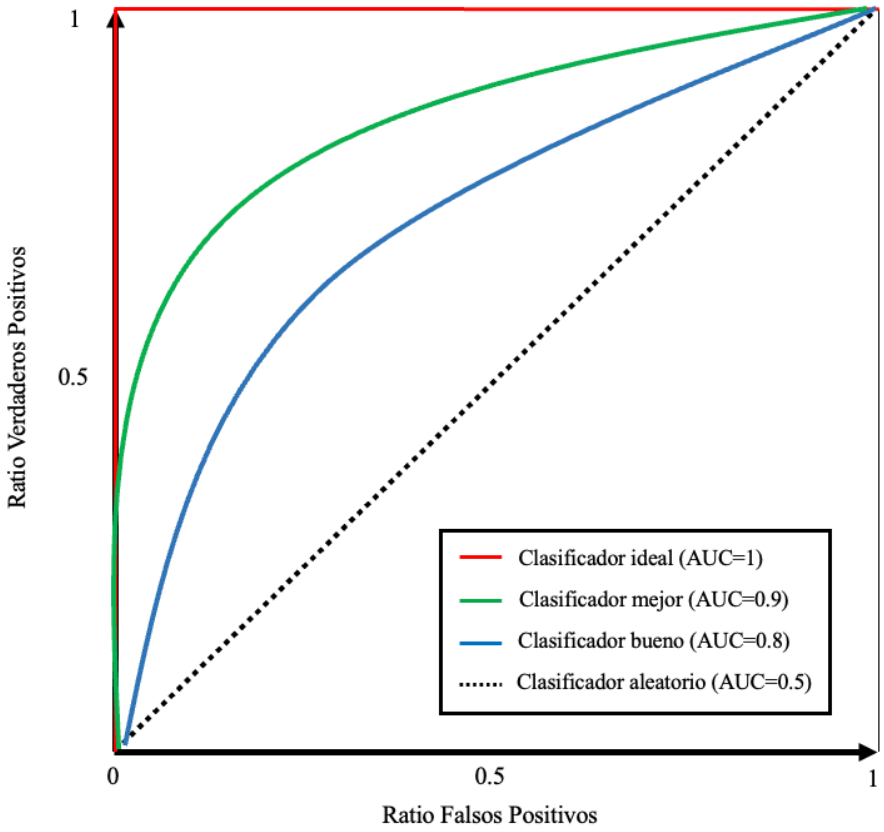


Figura 18: Curvas ROC de distintos tipos de modelos.

4.4. Transferencia de aprendizaje

La transferencia de aprendizaje o *transfer learning*, es una técnica que consiste en emplear como base una red neuronal ya entrenada previamente y

adaptarla para crear una nueva para, posteriormente, entrenarla con nuestra propia base de datos.

Esta técnica aporta una serie de ventajas, entra las que destacan las siguiente:

- **Aprovechamiento de conocimientos adquiridos**: La principal ventaja de la transferencia de aprendizaje es que permite aprovechar los conocimientos previamente adquiridos por modelos entrenados en conjuntos de datos masivos y tareas similares. Esto resulta beneficioso cuando se dispone de una cantidad limitada de datos de entrenamiento para una tarea específica. En lugar de entrenar un modelo desde cero, se puede emplear uno pre-entrenado en un conjunto de datos grande como punto de partida. El modelo base ya ha aprendido a detectar características visuales genéricas, como bordes, texturas y patrones, que son útiles en una amplia variedad de tareas.
- **Ahorro computacional**: El entrenamiento de redes neuronales convolucionales desde cero es un proceso que consume mucho tiempo y recursos computacionales. La transferencia de aprendizaje acelera significativamente el proceso, ya que se puede partir de un modelo pre-entrenado y afinar sus pesos en lugar de entrenar todo el modelo desde cero. Esto ahorra tiempo y recursos, especialmente en situaciones en las que el hardware supone una limitación.
- **Mejora del rendimiento**: La transferencia de aprendizaje a menudo conduce a un mejor rendimiento en comparación con entrenar un modelo desde cero, especialmente cuando se tienen pocos datos de entrenamiento. Al inicializar el modelo con pesos pre-entrenados, el modelo ya tiene una comprensión sólida de características visuales comunes, lo que lo hace más competente desde el principio. Posteriormente se pueden afinar estas características para que se ajusten mejor a la tarea específica que se requiera.
- **Reducción del riesgo de *overfitting***: Cuando se tiene un conjunto de datos pequeño, existe un alto riesgo de *overfitting* si se entrena una red desde cero. La transferencia de aprendizaje, al aprovechar una red ya entrenada, reduce este riesgo, ya que el modelo comienza con una inicialización más robusta y puede evitar ajustarse en exceso a los datos de entrenamiento.

Con esta serie de características, la transferencia de aprendizaje ha demostrado en diversos ámbitos, entre ellos la medicina debido a la dificultad de disponer de grandes bases de datos, ser una técnica eficaz y efectiva para el entrenamiento de redes neuronales [17].

4.5. Diseño de la red neuronal

En este apartado se va a explicar el modelo de red neuronal convolucional desarrollado en el presente proyecto fin de máster, cuya arquitectura puede dividirse en dos partes, por un lado, las capas de convolución, y por otro las capas *fully-connected* o totalmente conectadas.

Para las capas de convolución, se ha aplicado transferencia de aprendizaje tomando como base una red neuronal ya entrenada, por tanto, antes de explicar la arquitectura del modelo creado, se va a proceder con la explicación de las características de la red neuronal tomada como base, que en este caso ha sido una red con arquitectura DenseNet.

4.5.1. DenseNet

A diferencia de una red neuronal convolucional tradicional, como la mostrada anteriormente en la figura 5, en la que la etapa de aprendizaje de características es secuencial, en una arquitectura DenseNet, cada capa está conectada al resto de capas que le siguen, actuando como entrada de cada una de éstas, tal y como se muestra en la figura 20, de ahí el nombre de *densily connected network* o red densamente conectada [18].

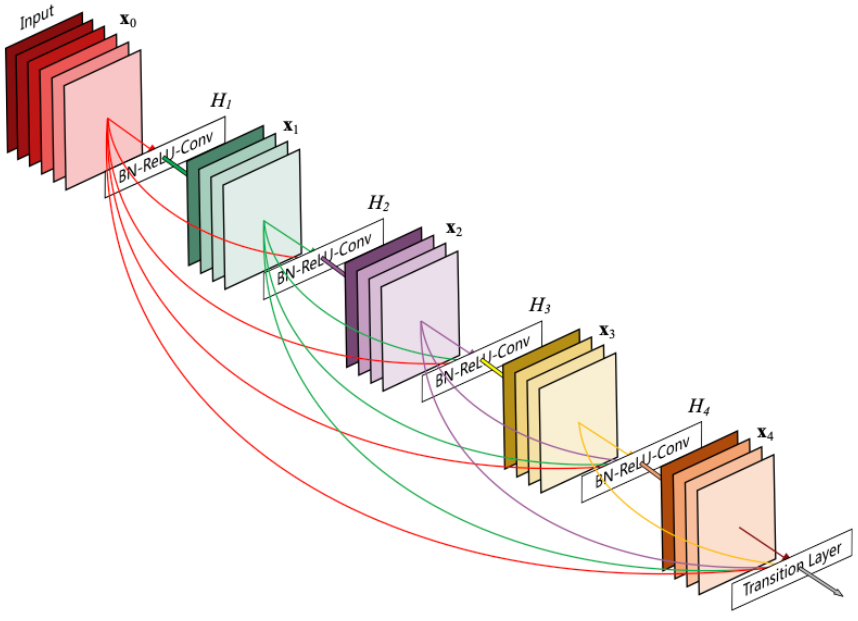


Figura 19: Ejemplo de red densamente conectada.
(Fuente: Huang, G., Liu, Z., Maaten, L. van der, & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2017.243)

En ciertos casos, esto aporta algunas ventajas ya que, en redes puramente secuenciales, a medida que la red se hace cada vez más profunda, es decir, cuenta con más capas, aumentan las probabilidades de que se dé el problema del desvanecimiento del gradiente.

Este problema hace referencia a una situación en la que los gradientes (las derivadas parciales de la función de pérdida con respecto a los pesos de la red) se vuelven cada vez más pequeños a medida que se retropropagan hacia las capas iniciales de la red durante el proceso de entrenamiento, y, cuando los gradientes se vuelven muy pequeños, las actualizaciones de los pesos en las capas iniciales son insignificantes, lo que hace que estas capas no aprendan efectivamente [19]. Como resultado, la red puede tener dificultades para converger o aprender patrones importantes en los datos.

En el caso de una red con arquitectura DenseNet, este problema se reduce de forma significativa facilitando el aprendizaje. Además, este tipo de redes posee otras características como que refuerzan la propagación de características, fomentan la reutilización de características y reducen sustancialmente el número de parámetros. Por estos motivos, se ha optado por emplear una arquitectura de red basada en DenseNet.

4.5.2. Arquitectura del modelo

Una vez expuesto el modelo de red DenseNet, se procede a explicar la arquitectura de la red desarrollada.

En la figura 21 se muestra una representación gráfica de dicha arquitectura.

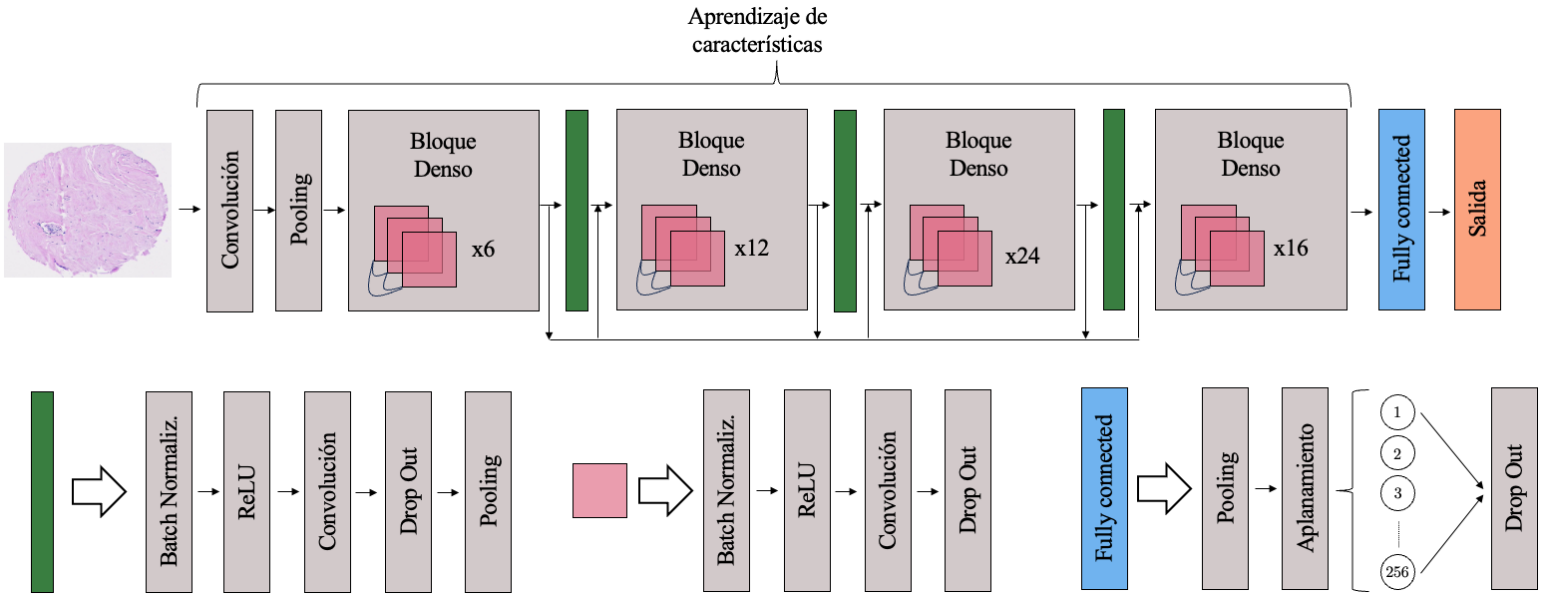


Figura 20: Arquitectura del modelo desarrollado.

Como se ha comentado anteriormente la arquitectura se divide en dos partes, las capas de convolución y capas totalmente conectadas.

Convolución

Las capas de convolución, que en el diagrama se corresponden con la etapa de aprendizaje de características, se han creado aplicando transferencia de aprendizaje, es decir, se ha implementado la arquitectura del modelo DenseNet121 [18] con sus pesos ya aprendidos. De esta forma, el modelo parte de un punto en el que ya ha aprendido a analizar e identificar objetos y patrones en imágenes por lo que se reducen computacionalmente los recursos y se puede lograr una mayor eficiencia y rendimiento del modelo.

Esta etapa está compuesta por una primera capa de convolución y agrupación o *Pooling*. Posteriormente, cuenta con bloques densos y capas de transición.

Los bloques densos, están formados por un determinado número de capas de convolución densamente conectadas. Dentro de estos bloques, las dimensiones del mapa de características permanecen constantes para permitir su concatenación, pero su volumen cambia

Por otro lado, las capas de transición, representadas en verde en el diagrama, realizan un muestreo descendente entre los bloques densos mediante una convolución y agrupación, es decir, reducen las dimensiones espaciales a la mitad.

Esta arquitectura de red tiene un hiperparámetro, la tasa de crecimiento (definido por k), que controla el número de mapas de características añadidos por cada bloque denso.

Capas totalmente conectadas

Tras la etapa de aprendizaje de características, se ha implementado una capa de 256 neuronas para que el modelo aprenda a clasificar las imágenes.

La capa de salida se corresponde con una única neurona cuya función de activación es la función sigmoide, que convierte sus entradas a un rango de valor entre 0 y 1, significando esto la probabilidad de que la imagen se corresponda con un caso de PD-L1 positivo o negativo.

El resto de las funciones de activación empleadas ha sido la función ReLU, explicada en el apartado 2.1.

Además, como se muestran en los diagramas, se han aplicado las técnicas de normalización por lote y *drop out* [20]. La normalización por lote, denominado *Batch normalization* en el diagrama, se ha utilizado para estandarizar las salidas intermedias, de esta forma, se estabiliza y acelera el entrenamiento de la red.

El *drop out* es una técnica que se emplea con el propósito de evitar que se produzca *overfitting* omitiendo un cierto porcentaje de las conexiones intermedias entre capas en cada iteración.

En la tabla 1 se muestran las características de las capas de convolución y las dimensiones de salida de cada capa en la arquitectura. Por características se refiere a las dimensiones de los *kernel* que tiene cada capa de convolución.

Capas	**Dimensiones de salida**	**Características (k = 32)**
Convolución	112 x 112	7 x 7 conv, stride 2
Pooling	56 x 56	3 x 3 max pool, stride 2
Bloque denso (1)	56 x 56	[1 x 1 conv 3 x 3 conv] x6
Capa de transición (1)	56 x 56	1 x 1 conv
	28 x 28	2 x 2 average pool, stride 2
Bloque denso (2)	28 x 28	[1 x 1 conv 3 x 3 conv] x12
Capa de transición (2)	28 x 28	1 x 1 conv
	14 x 14	2 x 2 average pool, stride 2
Bloque denso (3)	14 x 14	[1 x 1 conv 3 x 3 conv] x24
Capa de transición (3)	14 x 14	1 x 1 conv
	7 x 7	2 x 2 average pool, stride 2
Bloque denso (4)	7 x 7	[1 x 1 conv 3 x 3 conv] x16
Capa de clasificación	1 x 1	7 x 7 global average pool
	1 x 1	256 fully-connected, sigmoid

Tabla 1: Descripción de las características de las capas de la red.

4.5.3. Proceso de entrenamiento

El modelo se ha entrenado de manera independiente en las dos bases de datos disponibles que se indicaron en el apartado 3.1.

Base de datos CIMO2

El objetivo con esta base de datos es crear un modelo que clasifique las imágenes de PD-L1 en 3 categorías, menor de 1%, entre 1 y 49% y mayor de 49%. Por lo que se trata de un proceso de clasificación multiclase.

A continuación, se expone el proceso de entrenamiento.

En primer lugar, las 57 imágenes se han dividido en 3 subconjuntos. El conjunto de entrenamiento con 45 imágenes, el de validación con 6 y el de test con 6. Puesto que las imágenes pueden ser de 3 tipos, esta división se ha realizado manteniendo la misma proporción de imágenes de cada clase en los 3 subconjuntos para evitar un desequilibrio entre el conjunto de entrenamiento y los de prueba.

Puesto que los datos de entrenamiento son muy escasos, contando con 45 imágenes, se ha realizado un aumento de datos a este conjunto. Para ello, se han realizado varias pruebas analizando con qué técnicas de aumentos de datos se obtienen mejores resultados y, finalmente, se ha concluido obteniendo 180 imágenes nuevas a partir de las 45 iniciales aplicando giros, inversiones y ruido gaussiano, por lo que el conjunto de entrenamiento ha resultado ser de 180 imágenes.

Respecto a los conjuntos de validación y prueba, únicamente se les ha aplicado un corte para que la imagen sea cuadrada y un redimensionamiento a 224x224 píxeles en la etapa de preprocesamiento.

Posteriormente, se ha procedido con el entrenamiento del modelo. Las características del proceso de entrenamiento han sido las siguientes:

- Tasa de aprendizaje: $\alpha = 0.0001$.
- Optimizador: Adam.
- Función de pérdida: Pérdida focal.
- Tamaño del lote de entrenamiento: 10.

Base de datos del GPEC

En cuanto al entrenamiento con la base de datos del GPEC, el proceso ha sido el siguiente.

En primer lugar, las 2528 imágenes se han dividido en 3 subconjuntos. El conjunto de entrenamiento con 2047 imágenes, el conjunto de validación con 228 y finalmente el conjunto de test con 253 imágenes. Esta división se ha realizado manteniendo en los 3 subconjuntos la misma proporción de casos positivos y negativos de PD-L1 para que no se diese un desequilibrio entre los subconjuntos.

Posteriormente se ha realizado un preprocesamiento de las imágenes. En concreto, a cada imagen del conjunto de entrenamiento se le han aplicado las técnicas de aumento de datos explicadas en el apartado 4.1. Esto se ha realizado para introducir más variabilidad en cada imagen y así evitar que se produjese *overfitting*, pero no se ha aumentado la cantidad de imágenes. En orden de acciones aplicadas, el flujo de dicho preprocesamiento ha sido el siguiente:

- Corte de la imagen de forma cuadrada.
- Redimensionamiento a 224x224 píxeles.
- Rotación de la imagen un cierto ángulo aleatorio con una probabilidad del 50%.
- Inversión de la imagen de izquierda a derecha y de arriba a abajo con un 50% de probabilidad en cada caso.
- Corrección gamma con un valor aleatorio entre 0 y 3 para el factor gamma.
- Modificación de la saturación con un valor aleatorio entre 0 y 2.
- Modificación de la temperatura con un valor aleatorio entre 2000 y 8000 Kelvin.
- Adición de ruido Gaussiano siguiendo una distribución normal de media cero y desviación estándar 20.
- Ajuste del desenfoque con un valor aleatorio entre 0 y 2.

De esta forma, cada imagen ha sido modificada con distintos factores en cada caso, debido a que los valores de estos han sido aleatorios para cada imagen con el objetivo de introducir la mayor variabilidad posible y evitar así que el modelo aprenda patrones específicos del conjunto de entrenamiento.

En la figura 22, se muestra un ejemplo de dicho proceso aplicado a una imagen siendo el resultado final la imagen que recibe el modelo como entrenamiento. En dicha imagen, los ejes representan los píxeles.

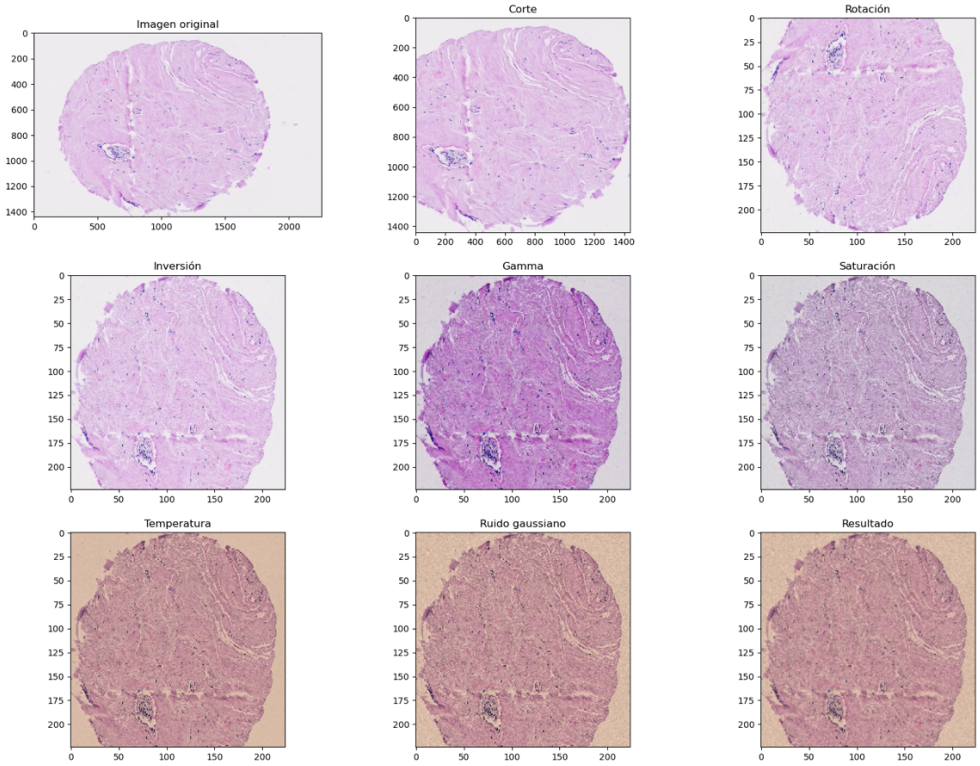


Figura 22: Preprocesamiento de las imágenes de entrenamiento.

En cuanto a los conjuntos de validación y test únicamente se les ha aplicado un el corte y redimensionamiento a 224 píxeles.

Tras realizar este preprocesamiento de las imágenes, se ha procedido con el entrenamiento del modelo. Las características del proceso de entrenamiento han sido las siguientes:

- Tasa de aprendizaje: $\alpha = 0.0001$.
- Optimizador: Adam.
- Función de pérdida: Pérdida focal.
- Tamaño del lote de entrenamiento: 10.

Se ha utilizado Adam como optimizador debido a su adaptabilidad, eficiencia en la convergencia y robustez en términos de hiperparámetros.

También, debido al desequilibrio presente en las etiquetas de los datos tal y como se muestra en el apartado 5.2, se ha utilizado la función de pérdida focal como objetivo principal durante el entrenamiento del modelo.

Cuando existe un desequilibrio de clases, la función pérdida de entropía cruzada estándar puede generar un sesgo no deseado. Esto significa que el modelo tiende a inclinarse hacia la clase dominante en lugar de adaptarse a la distribución real de los datos. Con la pérdida focal se resuelve este problema mediante una modificación de la función entropía cruzada estándar, reduciendo la pérdida asignada a los ejemplos bien clasificados.

Además, se ha implementado una parada temprana para evitar que el modelo realizase un sobreajuste. Esta parada temprana se basa en para el entrenamiento cuando las pérdidas en el conjunto de validación no mejoran durante 5 épocas, lo que daría indicio de *overfitting* en el modelo. Debido a esto, el modelo ha sido entrenado durante 14 épocas.

Nota: El código escrito en Python con el desarrollo de todo lo anteriormente explicado se encuentra disponible en el siguiente repositorio de GitHub: https://github.com/fernanh98/TFM.git

5. Resultados

En este apartado, se van a analizar los resultados obtenidos y realizar así, una evaluación del desempeño de las técnicas aplicadas en el proyecto.

5.1. Análisis de la base de datos

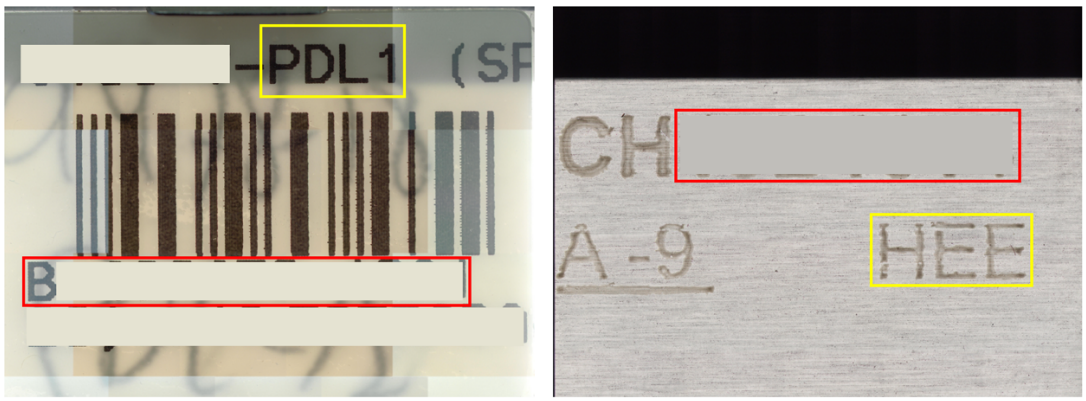
Base CIMO2

La base de datos CIMO2 disponible para la realización del proyecto consta, por un lado, de las imágenes de las muestras de las tinciones PD-L1 y hematoxilina-eosina (HE) y, por otro, un archivo Excel que contiene los resultados de cada muestra estableciendo un código para cada número de biopsia de manera que sea posible realizar una trazabilidad del paciente, tal y como se muestra en la tabla 2. En dicha tabla se ha modificado el número de muestra por motivos de privacidad.

Muestra	HISTOLOGÍA	Tipo Histológico	EGFR/ALK	PDL1-cat	PDL1-group	PDL1
649	ESCAMOSO	ESCAMOSO	NP	POS	>49	50
52	ESCAMOSO	ESCAMOSO	NP	POS	>49	60
123	NO ESCAMOSO	ADENOCARCINOMA	MUTneg	POS	>49	50
486	NO ESCAMOSO	CPNCP	MUTneg	POS	>49	50
65	NO ESCAMOSO	ADENOCARCINOMA	MUTneg	POS	>49	50

Tabla 2: Base de datos de las muestras.

En primer lugar, fue necesario vincular cada imagen con su muestra en la base de datos, ya que no existía previamente dicha relación. Para ello, como se muestra en la figura 23, cada imagen contiene una etiqueta que indica de qué muestra se trata y que tipo de prueba es, si PD-L1 o HE (hematoxilina-eosina). Además, como puede observarse en dicha figura, en rojo se indica la muestra a la que pertenece (omitido por motivos de privacidad) y en amarillo el tipo de imagen que es.



(a) (b)

Figura 23: Etiquetado de las muestras.
(a) Etiqueta de una muestra PD-L1. (b) Etiqueta de una muestra HE.

Por tanto, se empleó un Reconocimiento Óptico de Caracteres (OCR) programado en Python, que extrajo el texto contenido en cada etiqueta y así posteriormente se vinculó cada imagen con su correspondiente muestra en la base de datos.

En total se disponían de 136 imágenes, tras analizar las etiquetas de cada una como se ha mencionado anteriormente, se obtuvo que 88 eran de PD-L1 y 48 de HE.

Hay que tener en cuenta que, en ocasiones, los patólogos se ayudan de la imagen original sin tinción de PD-L1 para compararla con la que sí tiene tinción y así determinar el porcentaje de PD-L1, por tanto, dentro de las 88 imágenes de PD-L1, había algunas que no tenían tinción y que, por tanto, no resultaban de utilidad. Además, había otras imágenes que no habían sido analizadas y por tanto no se disponía del resultado de la prueba de PD-L1.

Tras revisar las muestras, el número se redujo a 57 imágenes de PD-L1 disponibles para el desarrollo del proyecto.

Base del GPEC

En cuanto a la base de datos obtenida del GPEC, tal y como se muestra en la figura 27, el archivo Excel ya contiene la ruta a cada imagen y el etiquetado de PD-L1, por lo que no ha sido necesario realizar ningún previo ajuste a dicha base de datos.

ID	PDL1 label	path
625	0	HE_C34_v3_s1_001.jpg
1229	1	HE_E12_v3_b3_161.jpg
878	0	HE_D12_v3_b3_110.jpg
1639	1	HE_F34_v3_s1_091.jpg
2449	1	HE_I12_v3_b3_145.jpg

Tabla 3: Base de datos del GPEC.

5.2. Descriptiva de las bases de datos

Seguidamente, se realizó un breve análisis exploratorio de las bases de datos, puesto que se trata de imágenes, no es necesario realizar un análisis exploratorio en profundidad de las variables.

Como se mencionó en el apartado 2.1, el patólogo determina el porcentaje de células que expresan PD-L1 y de esta forma, se clasifica cada muestra en 3 categorías: menor del 1% (es decir, negativo), entre 1 y 49% y, mayor del 49%. Tras analizar las 57 imágenes de CIMO2, se obtuvo lo siguiente:

- Menor del 1%: 15 muestras (26.32% del total).
- Entre 1 y 49%: 16 muestras (28.07% del total).
- Mayor de 49%: 26 muestras (45.61% del total).

Es decir, un 73.68% de las muestras se consideraban positivas y un 26.32% negativas.

Como se observa, clasificando las imágenes en las 3 categorías anteriores, éstas no se encuentran muy desbalanceadas, en cambio, si se realiza una única clasificación binaria entre negativo o positivo, sí estarían más desbalanceadas. Esto es importante tenerlo en cuenta a la hora de entrenar una red neuronal.

En cuanto a las 2528 imágenes del GPEC, las proporciones son de 66% con PD-L1 positivo y 34% con PD-L1 negativo.

5.3. Resultados de entrenamiento

A continuación, se van a exponer los resultados obtenidos tras realizar los entrenamientos del modelo en ambas bases de datos.

Base de datos CIMO2

En primer lugar, se analizan las curvas de aprendizaje para evaluar el proceso de entrenamiento del modelo.

Para ello, en la figura 24 se muestra la curva de aprendizaje de pérdida del modelo.

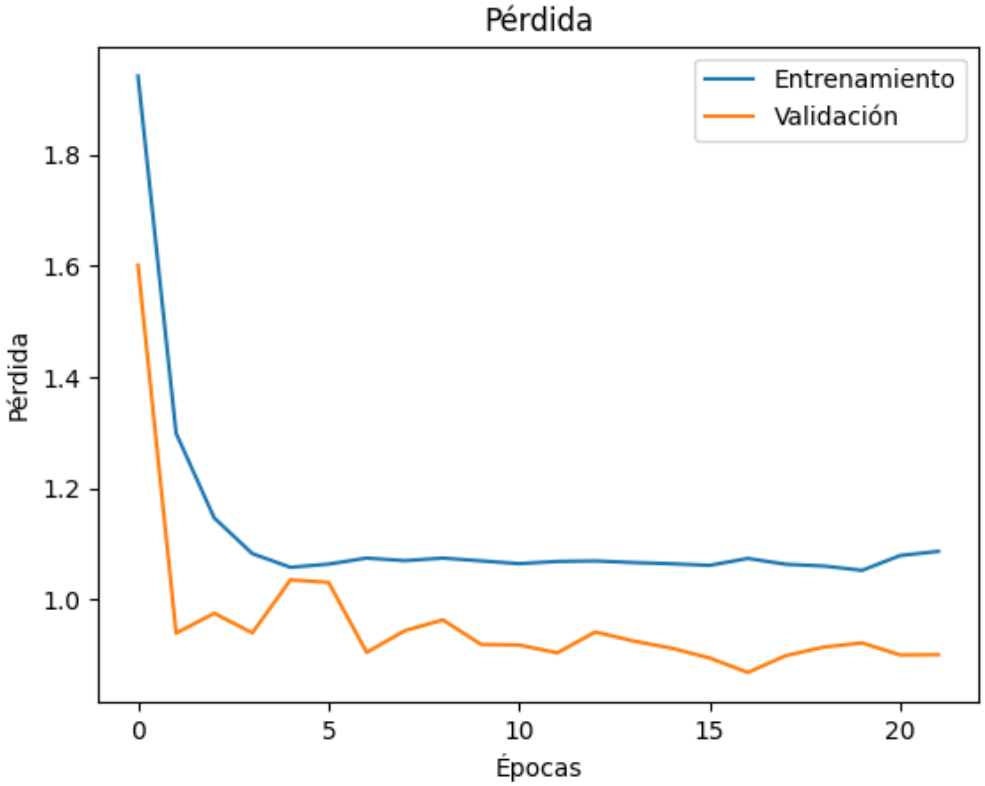


Figura 24: Curva de aprendizaje de pérdida durante el entrenamiento.

Como se observa en la imagen, tanto el conjunto de entrenamiento como el de validación comienzan con un valor aleatorio elevado que disminuye de forma significativa convergiendo a un valor, aunque con ciertas variaciones.

Un aspecto destacable es el hecho de que la pérdida de validación es menor que la pérdida de entrenamiento. Esto puede deberse a que el conjunto de datos de validación sea más fácil de predecir para el modelo que el de entrenamiento o, lo que es más probable en este caso, al ser el conjunto de validación más pequeño respecto al de entrenamiento, tiene una menor variabilidad.

No obstante, la curva de aprendizaje de pérdida muestra un proceso de entramiento correcto.

A continuación, se analiza la curva de precisión mostrada en la figura 25.

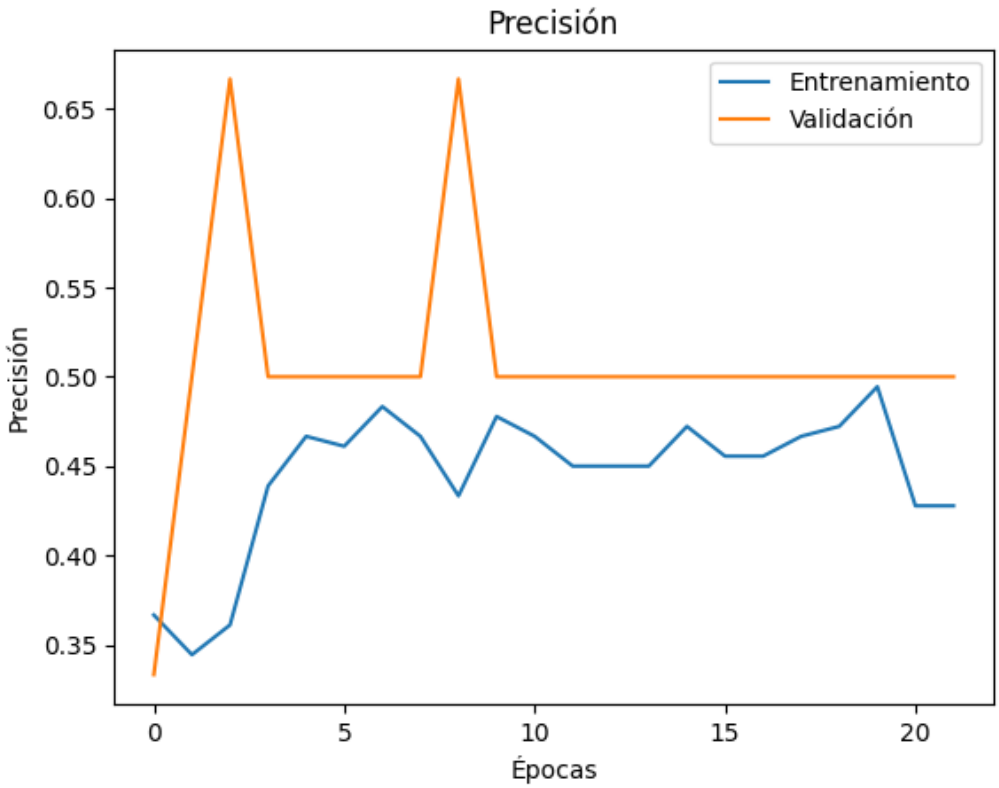


Figura 25: Curva de aprendizaje de precisión durante el entrenamiento.

En este caso, la curva de precisión muestra que el modelo no ha aprendido suficientes patrones correctos para la tarea de clasificación.

En cuanto al conjunto de entrenamiento, la precisión tuvo un aumento hasta la sexta época, a partir de la cuál muestra un comportamiento con algunas variaciones, pero con una tendencia de estabilidad en torno al 45% de precisión, indicando que el modelo ha alcanzado un punto de saturación o ha quedado atrapado en un mínimo local, por lo que no está aprendiendo los patrones correctos.

Respecto al conjunto de validación, presenta cierta inestabilidad hasta la novena época, esto es común y puede deberse a fluctuaciones naturales en los datos o al proceso de entrenamiento. Desde este punto permanece constante en 50%, lo que puede significar que el modelo ha encontrado una solución que le permite generalizar de manera constante en lugar de simplemente memorizar el conjunto de entrenamiento.

Tras analizar las curvas de aprendizaje, se calculan las métricas estadísticas para terminar de evaluar la actuación del modelo.

Para ello, en primer lugar, en la figura 26 se muestra la distribución de los casos en los conjuntos de test y predicción.

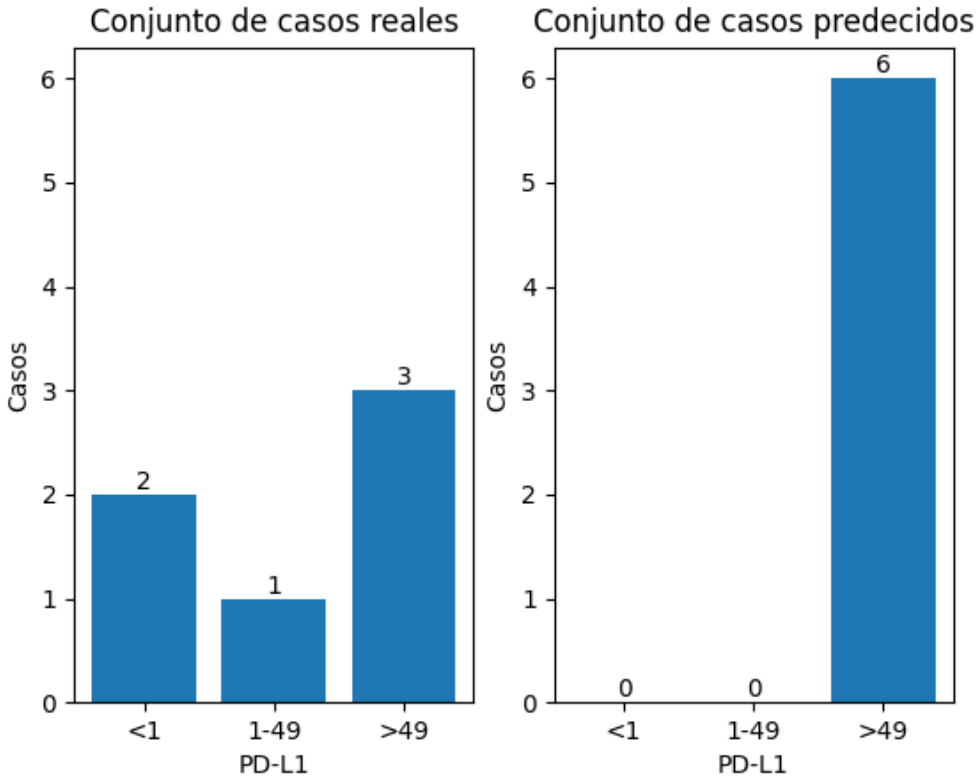


Figura 26: Distribución de casos en los conjuntos.

Como se observa en la imagen, las 6 imágenes del conjunto de test han sido determinadas por el modelo como PD-L1 mayor del 49%, esto puede deberse a que, en la base de datos, esta es la clase con mayor número de imágenes. Este hecho es importante tenerlo en cuenta a la hora de analizar las métricas estadísticas.

Puesto que en este punto se conoce que el desempeño del modelo no es realmente eficaz, se procede a analizar únicamente la curva ROC para comprobar la capacidad de discriminación entre las 3 clases.

En la figura 27 se muestra la curva ROC con el valor AUC para cada una de las 3 clases posibles.

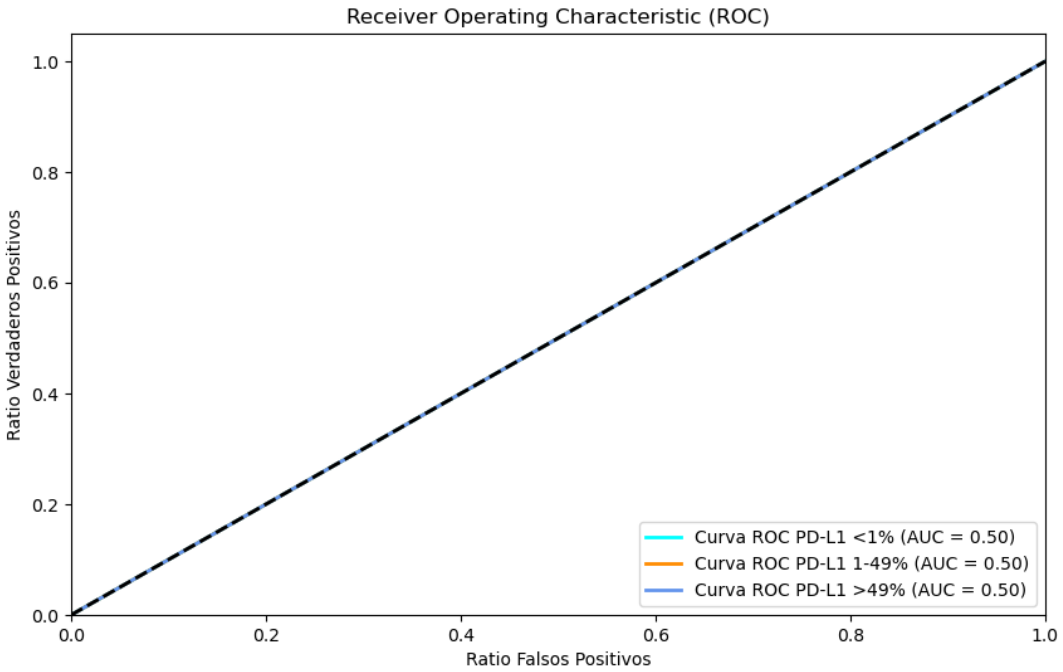


Figura 27: Curva ROC para las 3 clases posibles.

Como se muestra en las curvas ROC, para los 3 posibles casos de PD-L1, el modelo actúa como un clasificador aleatorio con un AUC de 0.5, indicando que efectivamente el modelo no ha aprendido suficientes características para ser capaz de clasificar entre las 3 clases.

Este comportamiento se debe a la escasez de suficientes imágenes para entrenar el modelo. Como se comentó anteriormente, para entrenar redes neuronales y obtener un desempeño correcto es necesario emplear bases de datos lo suficientemente grandes. A continuación, tras analizar los resultados con la base de datos del GPEC, en la cual sí se dispone de un número considerable de imágenes, se puede observar la diferencia entre ambos casos.

Base de datos del GPEC

En primer lugar, se analizan las curvas de aprendizaje para evaluar el proceso de entrenamiento del modelo.

En la figura 28 se muestra la curva de aprendizaje de pérdida del modelo.

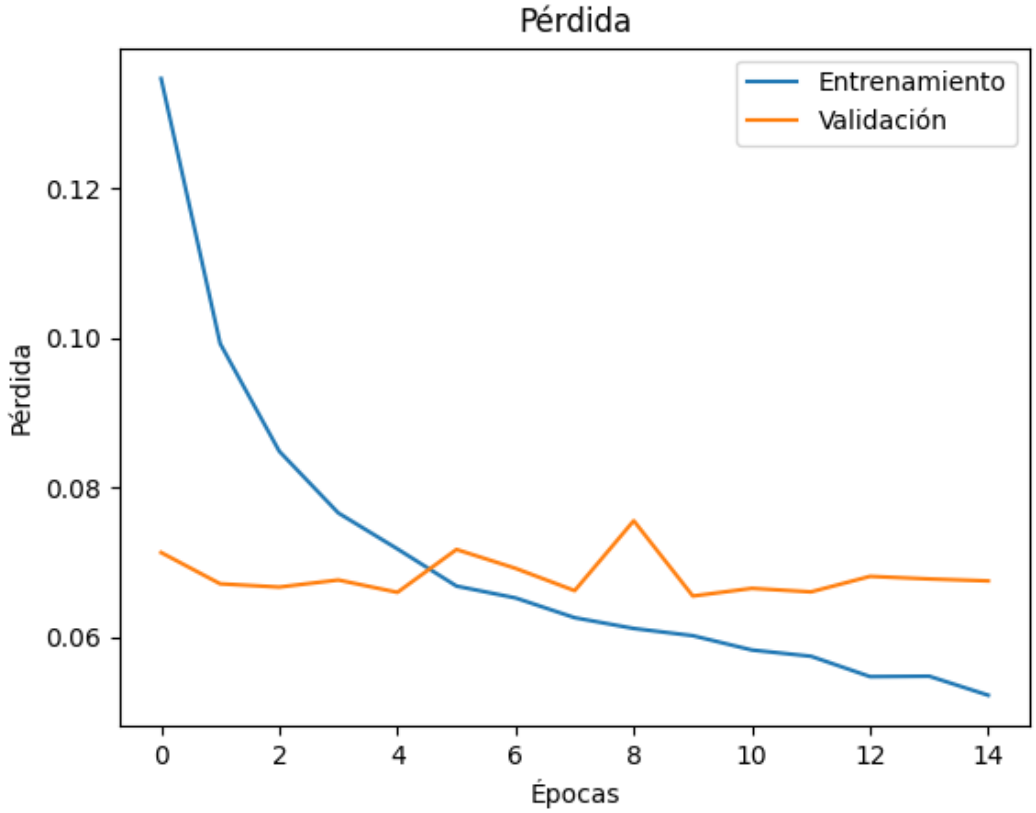


Figura 28: Curva de aprendizaje de pérdida durante el entrenamiento.

Como se observa en la imagen, el conjunto de entrenamiento comienza con unas pérdidas elevadas y disminuye de forma significativa durante el entrenamiento. En cambio, las pérdidas del conjunto de validación comienzan con un valor bajo. Esto puede ser indicio que, desde la primera época, el modelo ha actuado bien y ha generalizado de manera correcta. Además, hay que tener en cuenta que, a diferencia del conjunto de validación, las imágenes del conjunto de entrenamiento fueron sometidas a un preprocesamiento en el que se les introdujo ruido y desenfoque entre otros, por lo que, para el modelo, este conjunto puede resultar más complejo de determinar debido a la gran cantidad de variabilidad añadida.

Es importante destacar que el proceso de entrenamiento ha finalizado cuando el modelo continuaba aprendiendo del conjunto de entrenamiento sin mostrar señales de mejora en el de validación, hecho que implicaría un overfitting. Esto puede observarse en el espacio que se genera entre ambas curvas al final del entrenamiento.

A continuación, se analiza la curva de precisión mostrada en la figura 29.

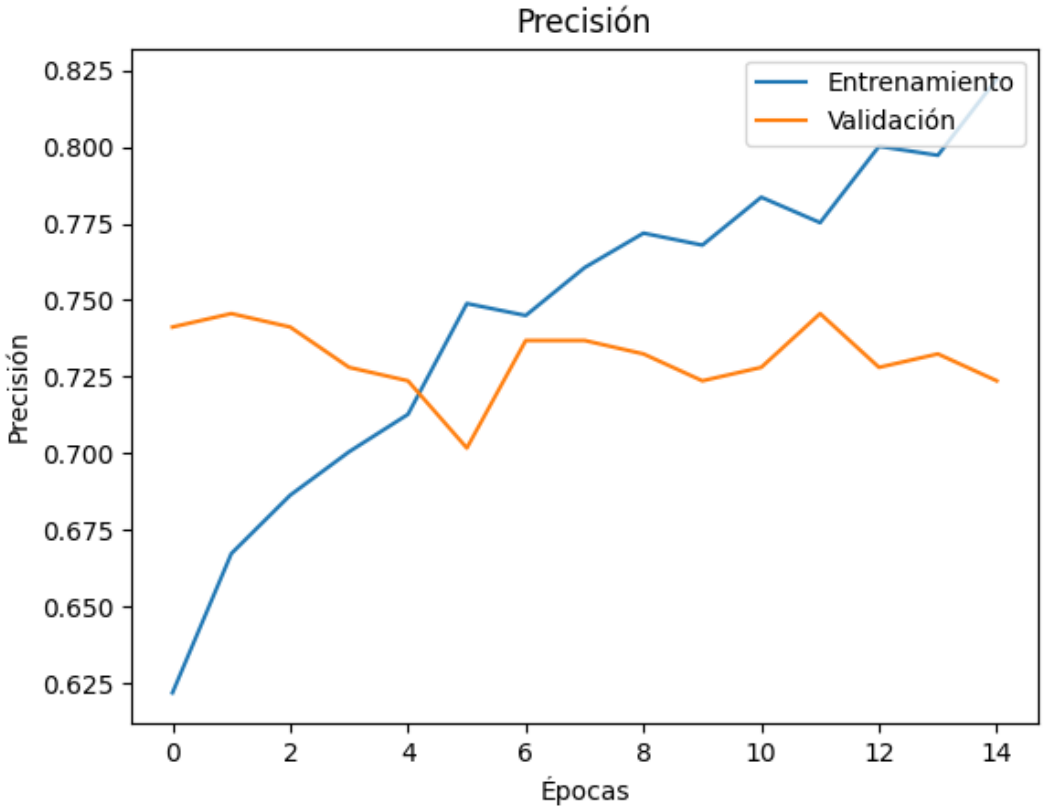


Figura 29: Curva de aprendizaje de precisión durante el entrenamiento.

En esta curva, al igual que se observó en la de pérdida, el conjunto de entrenamiento presenta una mejoría significativa durante el entrenamiento mientras que el de validación comienza con una mejor precisión y no muestra indicios de mejora. Lo que refuerza la idea que desde las primeras épocas el modelo ha generalizado de forma correcta y el resto de las características que va aprendiendo no resultan relevantes para la tarea.

También se observa en esta curva la importancia de detener el entrenamiento antes de que se produzca *overfitting* ya que, si se hubiese continuado con el proceso, la precisión en los datos de entrenamiento hubiese aumentado pero el de validación presenta cierta tendencia a disminuir, indicando que estaría el modelo sobreajustado al conjunto de entrenamiento.

Tras analizar las curvas de aprendizaje, se procede a evaluar el desempeño del modelo empleando las métricas estadísticas anteriormente explicadas.

Para ello, en primer lugar, se muestra en la figura 30 la distribución de casos de PD-L1 tanto negativo como positivo presente en el conjunto de datos de prueba y en la predicción del modelo.

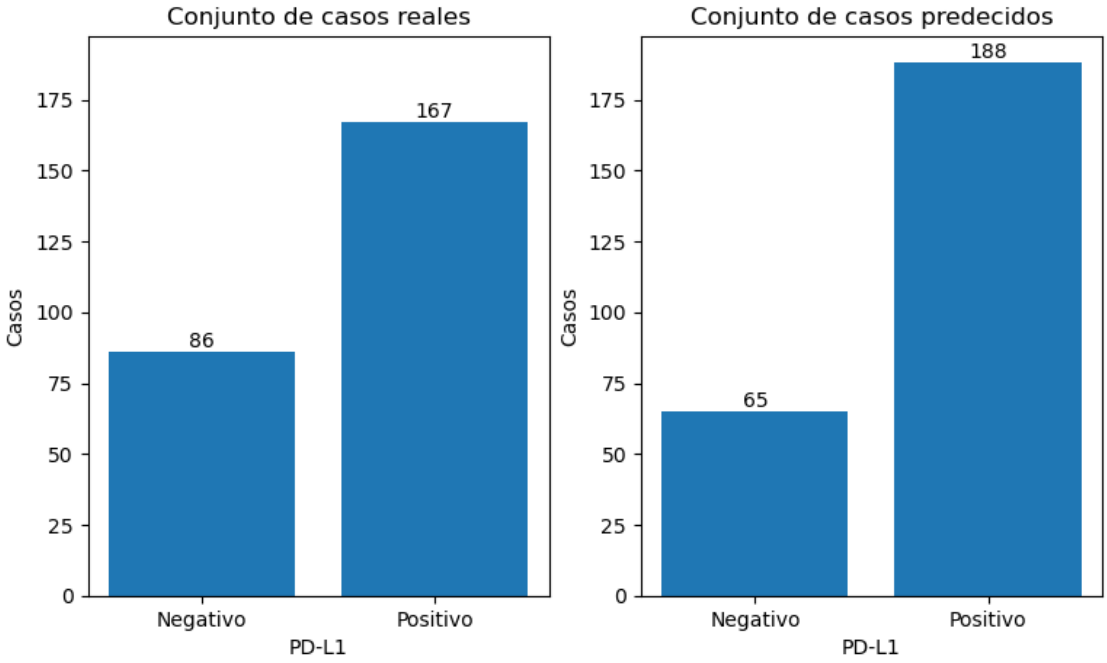


Figura 30: Distribución de casos en los conjuntos.

Y con ello, la matriz de confusión obtenida de las predicciones en el conjunto de test o prueba para los casos de PD-L1 positivo es la siguiente.

$$\begin{bmatrix} 52 & 34 \\ 13 & 154 \end{bmatrix}$$

Es decir:

- Verdaderos negativos: 52.
- Falsos negativos: 13.
- Verdaderos positivos: 154.
- Falsos positivos: 34.

Con estos datos, se obtienen las siguientes métricas.

$$Exactitud = \frac{VP + VN}{VP + FP + VN + FN} = 81.42\%$$

Como se explicó anteriormente, la exactitud mide la proporción de medidas correctas sobre el total de mediciones. En este caso se tiene que un 81.42% de los resultados del modelo son correctos, lo cual muestra capacidad de mejora, pero es un valor bastante bueno.

$$Precisión = \frac{VP}{VP + FP} = 82.91\%$$

Por otro lado, se tiene una precisión de 82.91%, siendo esto la proporción de predicciones correctas positivas entre el total de predicciones correctas que realiza el modelo, dejando poco porcentaje a posibles falsos positivos.

$$Sensibilidad = \frac{VP}{VP + FN} = 91.21\%$$

En cuanto a la sensibilidad, este valor indica que el modelo es capaz de identificar el 91.21% de casos positivos.

$$Especificidad = \frac{VN}{VN + FP} = 60.46\%$$

En cambio, el modelo muestra un peor desempeño a la hora de identificar los casos negativos ya que cuenta con una especificidad de 60.46%.

$$F1 - score = 2 \cdot \frac{Precisión \cdot Sensibilidad}{Precisión + Sensibilidad} = 0.86$$

Otra métrica relevante analizada es el F1-score que muestra la relación entre la precisión y sensibilidad, es decir, en este caso, un F1-score de 0.86 muestra un buen equilibrio entre ambas métricas. El modelo identifica con una alta probabilidad los casos positivos manteniendo igualmente una alta precisión.

Por último, en la figura 31, se procede a analizar la curva ROC para finalizar con la evaluación global del desempeño del modelo.

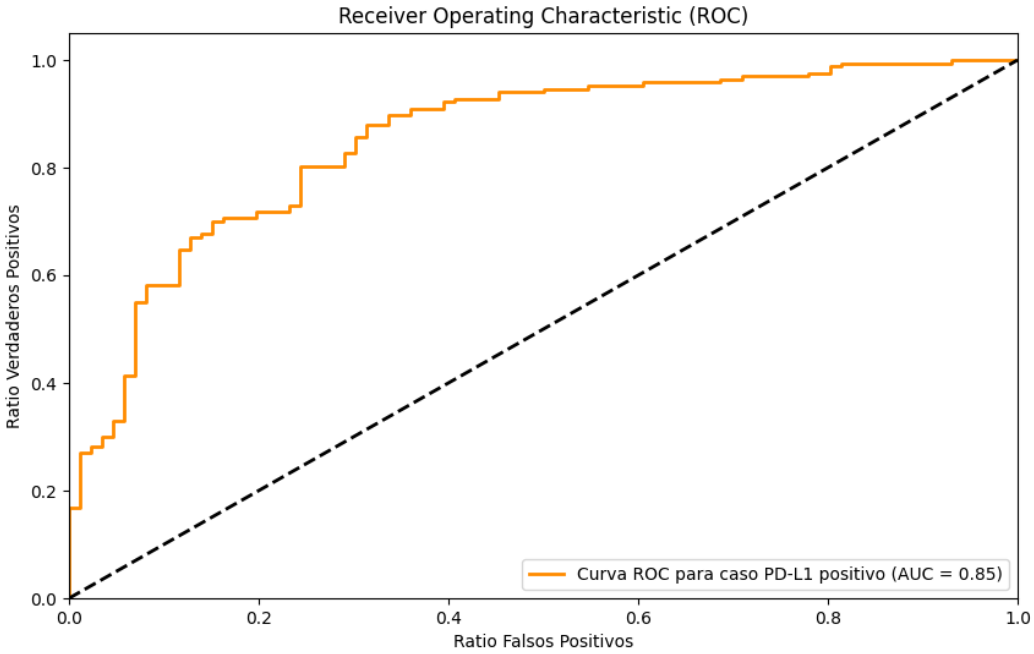


Figura 31: Curva ROC del modelo.

En este caso, observando la curva ROC, se puede determinar que el modelo posee una buena capacidad de discriminación entre casos positivos y negativos ya que posee un AUC de 0.85, estando lejos de tener un comportamiento de clasificador aleatorio.

Tras analizar los resultados obtenidos al evaluar las distintas métricas estadísticas, se concluye que el modelo entrenado con la base de datos del GPEC muestra un desempeño bastante correcto. Por supuesto muestra signos de mejoría en distintos aspectos, pero, desde un punto de vista global, el proceso de entrenamiento ha resultado con una

Otro aspecto importante que hay que tener en cuenta es, que, aunque el modelo muestre un desempeño efectivo en la mayoría de los escenarios. Al tratarse de aplicaciones médicas, el umbral de decisión específico que se elija puede variar según las necesidades clínicas y los costos asociados con los falsos positivos y negativos. Por tanto, en estos casos siempre es necesario realizar una evaluación conjunta con expertos en el ámbito médico que validen y aporten umbrales de decisión.

6. Conclusiones y futuras líneas de desarrollo

En este apartado se van a detallar las conclusiones obtenidas tras el desarrollo, implementación y evaluación de las técnicas empleadas durante el proyecto.

Además, se va a realizar una propuesta de mejoras para futuras líneas de desarrollo del proyecto.

6.1. Conclusiones

El objetivo planteado por el presente Trabajo Fin de Máster ha consistido en el desarrollo y evaluación de una red neuronal especializada en el análisis de imágenes médicas para determinar el estado de una determinada proteína, en este caso el PD-L1.

Durante el proceso de desarrollo se han ido aplicando distintas técnicas utilizadas en el *Deep learning*, en concreto, cuando se trabaja con redes neuronales convolucionales. Por tanto, para obtener el correcto diseño del modelo ha sido necesario el cumplimiento de distintos objetivos entre los que destacan:

- Se han extraído las conclusiones necesarias tras un análisis exploratorio de ambas bases de datos para tomar decisiones sobre cómo sería el tratamiento de las imágenes, qué características se tendrían en cuenta para entrenar el modelo, como funciones de activación, funciones de pérdida, optimizador etc.
- Se ha diseñado el preprocesamiento de las imágenes, creando una función propia en Python para ello que estuviese especialmente adaptada a las imágenes que se utilizan. Para ello se hicieron varias pruebas sobre qué modificaciones, incluido el valor de sus parámetros daban mejor resultado.
- Se ha elegido una arquitectura de red neuronal base sobre la que aplicar transferencia de aprendizaje la cual resultase en un correcto desempeño del modelo. Para ello se realizó una exhausta documentación sobre distintos tipos de arquitecturas utilizadas hoy día con sus respectivas ventajas y desventajas para el entrenamiento de ciertos datos.

Tras un análisis al proceso de entrenamiento del modelo con la base de datos CIMO2 y evaluar su desempeño, se obtiene como conclusión que, durante el entrenamiento, el modelo muestra un correcto proceso de aprendizaje como se pudo observar con las curvas de aprendizaje. No obstante, la base de datos es muy

pequeña. Para entrenar modelo de redes neuronales de muy alta eficacia son necesarias bases de datos amplias y que cuenten con suficiente variedad. En este caso no ha sido suficiente con aplicar las técnicas de aumento de datos.

Otro aspecto importante que se destaca con este entrenamiento es la importancia de que las imágenes se encuentren normalizadas. En este contexto, estas imágenes han sido evaluadas por distintos patólogos que no siempre entre ellos siguen los mismos criterios y, por tanto, no todas las imágenes se evalúan de una forma consistente, esto puede afectar a que el modelo aprenda de forma exacta los patrones relevantes en dichas imágenes. Este factor humano es importante tenerlo en cuenta.

Desde un punto de vista más técnico, también se destaca la importancia de elegir las características e hiperparámetros de las redes neuronales, tales como la función de pérdida, tasa de aprendizaje, porcentaje de dropo ut etc. En este caso ha sido necesaria la documentación y experimentación para llegar a los valores que suponen un mejor desempeño del modelo.

Finalmente, en cuanto al entrenamiento con la base del GPEC, se concluye que el modelo efectivamente muestra un correcto desempeño para evaluar si un paciente es positivo o negativo en la prueba de PD-L1 aun cuando presenta una posible mejora en cuanto al rendimiento.

Con este entrenamiento, también se pone en evidencia un hecho que puede resultar un punto de inflexión sobre cómo se llevan realizando ciertas tareas hasta el momento, ya que, en este caso en concreto, para determinar el estado de PD-L1, es necesario obtener en primer lugar la imagen de Hematoxilina-eosina para que le sirva al patólogo como guía para decidir qué zonas de la muestra analizar posteriormente aplicando la tinción concreta de PD-L1, finalizando con el análisis de esta muestra. La base de datos del GPEC contiene únicamente imágenes de Hematoxilina-eosina, es decir, el modelo ha sido entrenado para determinar el estado de PD-L1 analizando estas imágenes. Actualmente no se conocen métodos clínicos para determinar el PD-L1 con estas imágenes, por lo que el empleo de la inteligencia artificial en este caso ofrece una oportunidad como herramienta para los patólogos expertos a la hora de determinar el estado de PD-L1 de forma mucho más eficiente y si se mejoran rendimientos, con un muy elevado nivel de eficacia eliminando posibles factores humanos.

Como conclusión general del proyecto, cabe mencionar que éste ha tenido mucha labor de aprendizaje aplicado a la investigación. Con el desarrollo del proyecto se han aprendido muchas técnicas para crear un sistema de visión artificial poniéndolo en práctica con imágenes médicas, lo que en ciertos aspectos ha presentado un verdadero reto que ha aportado muchos conocimientos poniendo en práctica el potencial y habilidades adquiridas al estudiar una ingeniería.

6.2. Futuras líneas de desarrollo

Tras analizar las conclusiones del proyecto, se proponen a continuación una serie de mejoras y futuras líneas de investigación.

- Realizar una extensa base de datos con criterios de evaluación para PD-L1 unificados que pueda emplearse para el entrenamiento de un modelo con un elevado rendimiento.
- Analizar el proceso de implementación y puesta en producción de un modelo de inteligencia artificial de este tipo en hospitales, de forma que sirva como herramienta de apoyo a expertos en la medicina para la toma de decisiones sobre tratamientos.
- Realizar modelos semejantes al desarrollado en el presente proyecto, pero enfocados al estudio, que permitan abrir nuevas vías para descubrir otros marcadores relevantes y cómo emplearlos en la clínica.
- Actualizar con nuevas técnicas y algoritmos. En el ámbito de la inteligencia artificial se están llevando a cabo el desarrollo y descubrimiento de nuevos modelos cada vez más complejos y capaces. Por lo que resulta vital mantenerse actualizado para afrontar nuevos retos.
- Realizar más y diversos experimentos para lograr un ajuste más fino de los parámetros que caracterizan el modelo con objeto de aumentar su rendimiento.

7. Bibliografía

[1] Número de muertes por tumores en España de 2006 al 2021. (2022). Recuperado 25 de septiembre de 2023, de Statista website: https://es.statista.com/estadisticas/590245/numero-de-muertes-por-tumores-en-espana/#statisticContainer.

[2] Sandip Pravin Patel, Razelle Kurzrock; PD-L1 Expression as a Predictive Biomarker in Cancer Immunotherapy. Mol Cancer Ther 1 April 2015; 14 (4): 847–856. https://doi.org/10.1158/1535-7163.MCT-14-0983.

[3] Anwar, S.M., Majid, M., Qayyum, A. et al. Medical Image Analysis using Convolutional Neural Networks: A Review. J Med Syst 42, 226 (2018). https://doi.org/10.1007/s10916-018-1088-1.

[4] Sharma, S., Sharma, S., & Athaiya, A. (2017). Activation functions in neural networks. Towards Data Sci, 6(12), 310-316.

[5] Andrew Ng, Tengyu Ma. CS229 Lecture Notes, Stanford University (2023). https://cs229.stanford.edu/main_notes.pdf.

[6] O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks. arXiv preprint arXiv:1511.08458.

[7] Dumoulin, V., & Visin, F. (2016). A guide to convolution arithmetic for deep learning. arXiv preprint arXiv:1603.07285.

[8] Rojas, R. (1996). The Backpropagation Algorithm. In: Neural Networks. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-61068-4_7.

[9] Hoiem, D., Gupta, T., Li, Z. & Shlapentokh-Rothman, M.. (2021). Learning Curves for Analysis of Deep Networks. Proceedings of the 38th International Conference on Machine Learning, in Proceedings of Machine Learning Research 139:4287-4296 Available from https://proceedings.mlr.press/v139/hoiem21a.html.

[10] Koehrsen, W. (2018). Overfitting vs. underfitting: A complete example. Towards Data Science, 405.

[11] Shamai, G., Livne, A., Polónia, A. et al. Deep learning-based image analysis predicts PD-L1 status from H&E-stained histopathology images

in breast cancer. Nat Commun 13, 6753 (2022). https://doi.org/10.1038/s41467-022-34275-9.

[12] Raschka, S., & Mirjalili, V. (2017). Python machine learning: Machine learning and deep learning with python. Scikit-Learn, and TensorFlow. Second edition ed, 3.

[13] Hamad, K. & Kaya, M. (2016). A Detailed Analysis of Optical Character Recognition Technology. International Journal of Applied Mathematics Electronics and Computers, Special Issue (2016), 244-249. DOI: 10.18100/ijamec.270374.

[14] L. Taylor and G. Nitschke, "Improving Deep Learning with Generic Data Augmentation," 2018 IEEE Symposium Series on Computational Intelligence (SSCI), Bangalore, India, 2018, pp. 1542-1547, doi: 10.1109/SSCI.2018.8628742.

[15] Goceri, E. Medical image data augmentation: techniques, comparisons and interpretations. Artif Intell Rev 56, 12561–12605 (2023). https://doi.org/10.1007/s10462-023-10453-z.

[16] Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern recognition, 30(7), 1145-1159.

[17] Kim, H.E., Cosa-Linan, A., Santhanam, N. et al. Transfer learning for medical image classification: a literature review. BMC Med Imaging 22, 69 (2022). https://doi.org/10.1186/s12880-022-00793-7.

[18] G. Huang, Z. Liu, L. Van Der Maaten and K. Q. Weinberger, "Densely Connected Convolutional Networks," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 2261-2269, doi: 10.1109/CVPR.2017.243.

[19] Roodschild, M., Gotay Sardiñas, J., & Will, A. (2020). A new approach for the vanishing gradient problem on sigmoid activation. Progress in Artificial Intelligence, 9(4), 351-360.

[20] Garbin, C., Zhu, X., & Marques, O. (2020). Dropout vs. batch normalization: an empirical study of their impact to deep learning. Multimedia Tools and Applications, 79, 12777-12815.

[21] El origen de deep learning. (2022). Recuperado 26 de septiembre de 2023, de Universidad de Alcalá website: https://master-

deeplearning.com/origen-deep-learning/#:~:text=El%20Deep%20Learning%20es%20el,aprender%20y%20prevenir%20futuros%20errores.

[22] ¿Qué es scrum, cuál es su finalidad y sus principales ventajas? (2021). Recuperado 26 de septiembre de 2023, de Zendesk website: https://www.zendesk.com.mx/blog/que-es-scrum/#:~:text=Scrum%20es%20una%20metodología%20ágil,el%20proceso%20sea%20más%20eficiente.