UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DISEÑO, IMPLEMENTACIÓN Y EVALUACIÓN DE UNA RED NEURONAL CONVOLUCIONAL DE REGRESIÓN EN CLASIFICACIÓN DE NARANJAS

AUTOR: VÍCTOR SENÉN CERDÀ

TUTOR: ANTONIO JOSÉ SÁNCHEZ SALMERÓN

Curso Académico: 2019-20

AGRADECIMIENTOS

A mi familia, por apoyarme y darme su confianza a lo largo de mis estudios.

A mi tutor por introducirme en este interesante campo con mucho futuro y su indispensable ayuda en el trabajo.

A mi mejor amigo, porque fue gracias a él que escogí esta carrera y también porque la vida sería la mitad de interesante sin él.

RESUMEN

En este trabajo se han diseñado, implementado y evaluado diferentes arquitecturas de redes neuronales convolucionales de regresión. Estas redes neuronales ordenan las imágenes de naranjas capturadas en una línea de producción según su calidad.

Inicialmente, el trabajo se ha centrado en la creación y edición de forma supervisada de un conjunto de imágenes bien etiquetadas. Para ello, se ha utilizado la herramienta de etiquetado 'datasets.ai2.upv.es/oranges' disponible en el Instituto de Automática e Informática Industrial (ai2). Posteriormente, se han diseñado distintas arquitecturas de redes neuronales de regresión y se ha optimizado cada una de ellas sobre el espacio de hiperparámetros utilizando Python y Pytorch. Finalmente, se han evaluado las distintas arquitecturas propuestas, utilizando como criterios de optimización tanto las tasas de aciertos como los costes temporales.

Los resultados obtenidos muestran que las redes neuronales convolucionales son un método con potencial para la preselección y selección de cítricos en base a su calidad, la automatización del proceso de la línea y la clasificación a altas velocidades.

Palabras Clave: Regresión; edición y creación de conjunto de datos; aprendizaje profundo, redes neuronales convolucionales

RESUM

En aquest treball s'han dissenyat, implementat i avaluat diferents arquitectures de xarxes neuronals convolucionals de regressió. Aquestes xarxes neuronals ordenen les imatges de taronges capturades en una línia de producció segons la seva qualitat.

Inicialment, el treball s'ha centrat en la creació i edició de forma supervisada d'un conjunt d'imatges ben etiquetades. Per a això, s'ha utilitzat l'eina d'etiquetatge 'datasets.ai2.upv.es/oranges' disponible a l'Institut d'Automàtica i Informàtica Industrial (ai2). Posteriorment, s'han dissenyat diferents arquitectures de xarxes neuronals de regressió i s'ha optimitzat cadascuna d'elles sobre l'espai d'hiperparámetres utilitzant Python i Pytorch. Finalment, s'han avaluat les diferents arquitectures propostes, utilitzant com a criteris d'optimització tant les taxes d'encerts com els costos temporals.

Els resultats obtinguts mostren que les xarxes neuronals convolucionals són un mètode amb potencial per a la preselecció i selecció de cítrics en base a la seva qualitat, l'automatització del procés de la línia i la classificació a altes velocitats.

Paraules clau: Regressió; edició i creació de conjunt de dades; aprenentatge profund, xarxes neuronals convolucionals

ABSTRACT

In this work, different regression convolutional neural network architectures have been designed, implemented and evaluated. These neural networks order the images of oranges captured on a production line according to their quality.

Initially, work focused on supervised creation and editing of a set of well-labeled images. For this, the labeling tool 'datasets.ai2.upv.es/oranges' available at the Institute of Industrial Automation and Informatics (ai2) has been used. Subsequently, different regression neural network architectures have been designed and each one has been optimized on the hyperparameter space using Python and Pytorch. Finally, the different proposed architectures have been evaluated, using both the hit rates and the temporary costs as optimization criteria.

The obtained results show that convolutional neural networks are a method with potential for pre-selection and selection of citrus based on its quality, citrus line automatization and classification at high speeds.

Keywords: Regresion; dataset creation and editing; deep learning, convolutional neural networks

Índice de la memoria

Índice del Presupuesto

Listado de Figuras

Listado de Gráficas

DOCUMENTO I: MEMORIA

Capítulo 1:

Introducción

En este capítulo se expone el marco teórico sobre el que se apoya el proyecto. En primer lugar, se introducirá el campo de la visión artificial por computadora y se expondrá la motivación. A continuación, se expondrá qué es el aprendizaje profundo y el proceso de creación de una red neuronal. Por último, se explicará el proceso de una línea de manipulación de cítricos y el uso de sistemas de visión artificial en ella.

1.1 Introducción general

Uno de los campos de investigación que desde hace más de una década lleva cobrando relevancia en un mundo que progresa hacia una mayor automatización es, sin duda, el aprendizaje máquina (*machine learning*). Y dentro del aprendizaje máquina, una de las herramientas que ha resultado clave para la automatización de diferentes procesos ha sido la visión artificial por computadora.

Sistemas basados en una inspección visual por computadora automática pueden ser el medio perfecto para aumentar la productividad, y por tanto la competitividad, permitiendo a las industrias llevar a cabo procesos de inspección continuos de alta fiabilidad a lo largo de toda su producción. No solo supondría una mayor producción, también favorecería la calidad final de los productos junto con notables ahorros económicos y medioambientales.

Una de las herramientas que ha impulsado el campo de la visión artificial y el mundo de la inteligencia artificial en general son las redes neuronales artificiales. Su versatilidad en cuanto a las tareas de aprendizaje profundo que pueden realizar permite su aplicación en diferentes industrias y ámbitos de interés. Su uso está establecido en las principales industrias tecnológicas, y una implementación de redes neuronales en industrias más tradicionales supondría una optimización sustancial en sus procesos de control.

En el presente Trabajo Fin de Grado se propone el desarrollo de un sistema de visión artificial aplicado a la industria alimentaria, una red neuronal convolucional que es capaz de clasificar la calidad de las naranjas a lo largo de una línea de producción mediante el análisis de sus imágenes.

1.1.1 Visión artificial

Los sistemas de visión por computadora llevan décadas utilizándose en diversas aplicaciones y ámbitos del sector industrial, siendo la más prominente la inspección, o control, por imágenes. Uno de los principales inconvenientes de los sistemas tradicionales de visión por computadora es la necesidad de una selección apropiada de parámetros o características de los objetos que se quiere inspeccionar. Estos conjuntos de características son los que permiten clasificar las imágenes de interés, pero su obtención es compleja y puede representar un problema teniendo en cuenta la disparidad y variabilidad de todos los casos posibles de dichas imágenes. Debido a esta circunstancia se han elaborado técnicas que intentan disminuir la complejidad de esta tarea. Ejemplos de estas técnicas en la industria agroalimentaria serian técnicas de detección de hierbas en cultivos [1] o el control de calidad de jamones curados [2].

Los avances tecnológicos de la última década en cuanto a sensores, cámaras y escáneres han supuesto un impulso al campo de la visión artificial aplicado en la industria agroalimentaria, que está viendo incrementarse el uso de este tipo de sistemas. Gracias a estos avances, los productos pueden ser controlados mediante cámaras [3] , escáneres tridimensionales [4] , sensores espectrales [5] y otros dispositivos que realizan tareas de recogida de elevadas cantidades de información.

Junto a estos avances tecnológicos, se han desarrollado sistemas basados en redes neuronales convolucionales capaces de realizar labores de detección e inspección con altos niveles de complejidad. Con este tipo de redes se desarrollan sistemas de alta fiabilidad para el control de productos [6] . La principal ventaja que tienen las redes neuronales respecto a los métodos tradicionales es que la propia red es la que establece las características y patrones en las que basa su modelo de clasificación.

1.1.2 Motivación

El uso de sistemas de inteligencia artificial es cada vez más prevalente, no solo con la automatización de los procesos en la industria (industria 4.0), sino en numerosos otros campos de investigación, como la ciencia biomédica. Todo indica que cobrarán más relevancia en el futuro y que el mundo globalizado se adaptará a su uso en muchos aspectos de la vida.

Esta buena perspectiva de futuro del campo de la inteligencia artificial aplicada a la industria, junto a un interés personal en entender el funcionamiento de las redes neuronales, es la principal motivación de este trabajo.

1.2 Aprendizaje Profundo y Redes Neuronales Artificiales

1.2.1 Aprendizaje Máquina y Aprendizaje Profundo

Aprendizaje Máquina (*Machine Learning*) es la disciplina científica que consiste en desarrollar técnicas para producir algoritmos que permitan a un ordenador asimilar información con el fin de realizar una función. Originalmente el programa no tendría el sistema adecuado para llevar a cabo dicha función, pero aprendería a realizarla de forma eficiente modificando su propio sistema acorde a una serie de datos suministrados para su aprendizaje.

Una de las principales aplicaciones del Aprendizaje Máquina es el desarrollo de modelos predictivos en base a los patrones y correlaciones modelizados internamente durante el aprendizaje.

El Aprendizaje Profundo (*Deep Learning*) es una clase de algoritmos de Aprendizaje Máquina basados en redes neuronales que aprenden por capas de los datos que se les proporciona. Se denomina profundo por la cantidad de capas y parámetros que conforman estas redes.

Para llevar a cabo el aprendizaje de estos sistemas y cuantificar su efectividad se suelen utilizar tres conjuntos de datos:

- *Entrenamiento*, datos destinados al aprendizaje del sistema de forma iterativa. El sistema extrae información de estos datos y modifica sus elementos creando estructuras internas acorde a ellos.

- *Validación*, datos sobre los que el sistema comprueba su función en cada iteración para "medir" su precisión. En el caso de que la función sea generar predicciones, la validación permite averiguar si el sistema está produciendo las estructuras internas adecuadas o si se necesitan variar los parámetros del sistema.

- *Test,* un último conjunto de datos sobre el que se evalúa el sistema una vez finalizado y optimizado el aprendizaje. Este conjunto no lo pueden conformar datos que han sido utilizados para el entrenamiento.

1.2.1.1 Tipos de Aprendizaje y Aplicaciones

Dependiendo del tipo de función deseada para el algoritmo se realiza un tipo de aprendizaje u otro, siendo su principal clasificación la siguiente:

- ***Aprendizaje supervisado:***

El Aprendizaje supervisado (*Supervised Learning*) consiste en proporcionar al algoritmo inputs de datos junto a los outputs deseados de éstos, llamados etiquetas. El objetivo es que el algoritmo se adapte a los pares input/etiqueta de forma que al final del entrenamiento sea capaz de predecir la etiqueta de un input que no haya sido utilizado en su aprendizaje.

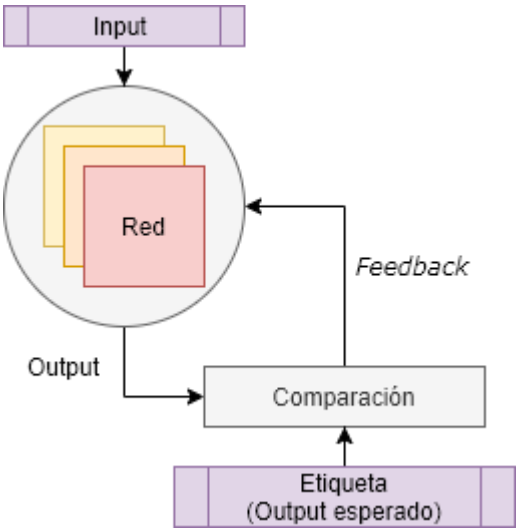


Figura 1. Principio de funcionamiento del Aprendizaje Supervisado. La red se retroalimenta de la comparación entre output y etiqueta para modificar los elementos de su sistema y mejorar.

Las aplicaciones principales que puede ofrecer este tipo de entrenamiento son la clasificación y la regresión:

- *Clasificación:*

La Clasificación en un contexto de Aprendizaje Máquina consiste en identificar un input como perteneciente a una categoría o clase.

Para conseguir esto se proporciona al sistema datos ya catalogados con sus correspondientes clases como etiquetas para que aprenda a discernir qué elementos de los inputs se asocian a su categoría y poder, una vez que el aprendizaje sea completo, categorizar correctamente nuevos inputs pertenecientes a estas clases sin etiquetar.

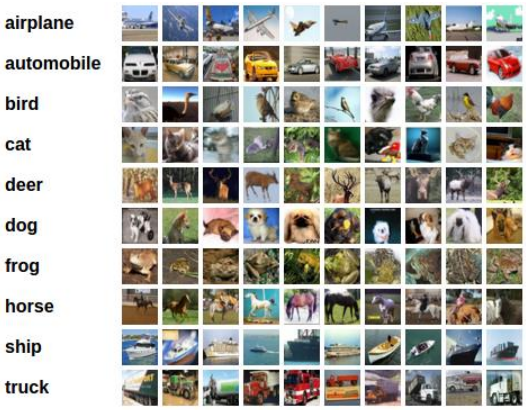


Figura 2. Dataset CIFAR-10, utilizado para testear redes de clasificación. Contiene imágenes de automóviles y animales etiquetadas.

- *Regresión:*

La Regresión consiste en estimar el valor de variables a partir de un input. La predicción se basa en las relaciones entre los elementos o variables de los datos proporcionados interpretados por el sistema. El entrenamiento se realiza proporcionando al sistema inputs junto el valor de su variable deseada.

Por tanto, en el presente trabajo de clasificación de naranjas según su calidad, estamos aplicando un tipo de aprendizaje supervisado, concretamente una regresión, ya que proporcionamos imágenes etiquetadas al sistema.

- ***Aprendizaje no supervisado**:*

El Aprendizaje no supervisado (*Unsupervised Learning*) consiste en proporcionar al algoritmo inputs de datos sin información adicional sobre estos. El sistema analiza y de forma autónoma crea un modelo basado en las relaciones que tienen los elementos de los inputs unos con otros creando así una estructura interna.

Una de las aplicaciones principales es el Análisis de grupos o *Clustering*:

- *Clustering:*

El Análisis de grupos consiste en agrupar datos similares o pertenecientes a un mismo grupo, similar a la clasificación, siendo el propio sistema quien establece las categorías. Sus aplicaciones abarcan desde detección de patrones hasta compresión de datos.

- ***Aprendizaje por refuerzo**:*

Un sistema que utiliza Aprendizaje por refuerzo (*Reinforcement Learning*) utiliza como inputs una serie de datos que describen una circunstancia o entorno específico, y produce como outputs acciones que varían dicho entorno. El sistema recibe un *feedback*, llamado recompensa (*reward*), resultante de la variación y lo interpreta mediante una función de evaluación. El objetivo es configurar al sistema para que produzca las acciones que maximicen la recompensa.

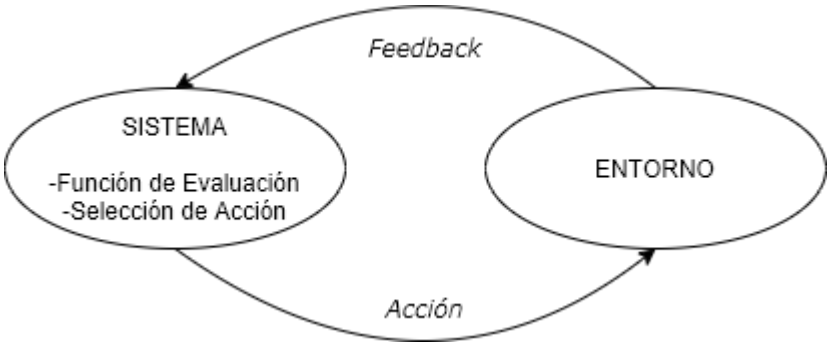


Figura 3. Principio de funcionamiento del Aprendizaje por refuerzo. Un ejemplo de utilización de este sistema son las recomendaciones personalizadas en plataformas de internet, anuncios, videos, artículos...

1.2.2 Redes Neuronales Artificiales

Las Redes Neuronales Artificiales son modelos computacionales que siguen un principio similar a las conexiones neurológicas del cerebro humano.

Cada neurona, una unidad computacional, está organizada en grupos denominados capas, que a su vez se interconectan con neuronas de otras capas creando así una especie de Red multicapa (con esquemas similares a la figura 4). La información que se proporciona como entrada se transmite por las neuronas de las capas y se somete a una serie de operaciones matemáticas lineales y no lineales hasta producir unos datos como salida.

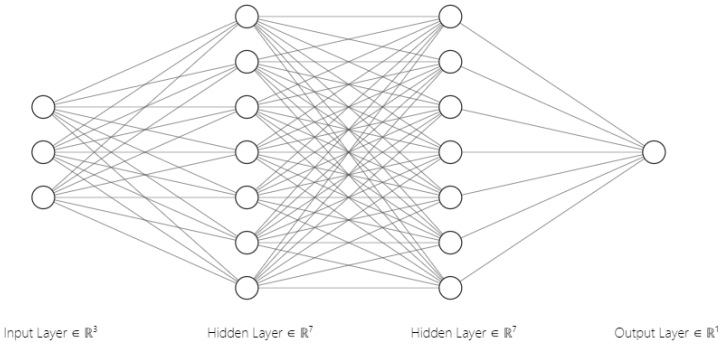


Figura 4. Esquema de una Red Neuronal Artificial. La información de la capa de entrada o inputs se interconectan con grupos de neuronas llamadas capas ocultas. Tras pasar las capas ocultas se transmiten a una última capa de salida como outputs.

1.2.2.1 Perceptrón

El Perceptrón es un modelo de neurona propuesto por Frank Rosenblatt en 1958 [7] que se utiliza como unidad básica en Redes Neuronales Artificiales. La información de entrada es tratada a lo largo de su estructura para producir una salida en función de una serie de parámetros.

- ***Pesos sinápticos:***

Los pesos sinápticos (*weights*) son una serie de parámetros que multiplican a cada una de las entradas. Se actualizan en función de la diferencia entre la salida y el resultado esperado. La capacidad de cambio de los pesos en cada iteración viene marcada por otro parámetro llamado tasa de aprendizaje (*learning rate*). Los pesos se suelen inicializar con valores aleatorios próximos a 0.

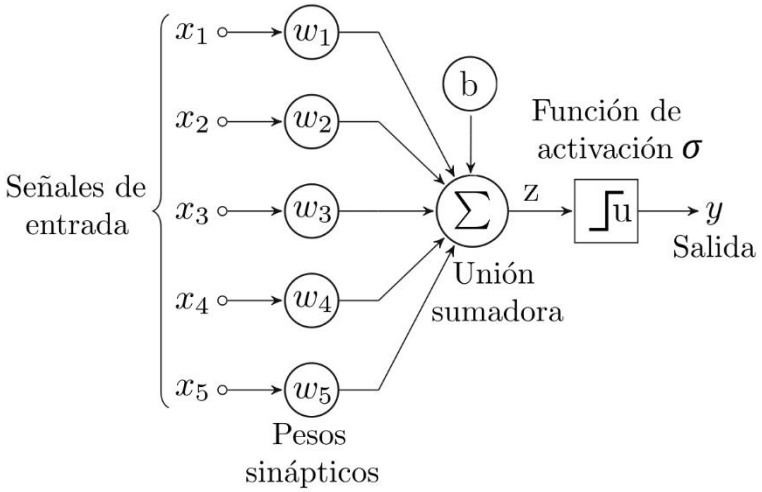


Figura 5. Esquema del Perceptrón. Las entradas son multiplicadas por los pesos y se suman junto al bias.

- ***Unión sumadora y bias:***

El resultado de los productos escalares entre pesos y entradas se suman junto a otro parámetro llamado bias, tal y como se ilustra en la figura 5. El bias adapta el resultado de la unión sumadora según la función de activación que se aplica posteriormente.

- ***Función de Activación:***

La función de activación es una función que permite el paso de información en la salida o la desactiva en función del resultado de la unión sumadora y bias.

$$\sigma(z) = \begin{cases} 1 & \text{si } z > 0 \\ 0 & \text{en otro caso} \end{cases}$$

Figura 6. Función de activación del Perceptrón. En el Perceptrón original era una función de escalón que se activa si el resultado de la unión sumadora y bias supera un umbral.

1.2.2.2 Perceptrón Multicapa

Dado que el Perceptrón no tiene la capacidad de resolver problemas no lineales, una forma de solventar esta situación es organizar las neuronas por capas e interconectarlas con otras capas con un patrón similar a la figura 7, en una organización denominada Perceptrón Multicapa.

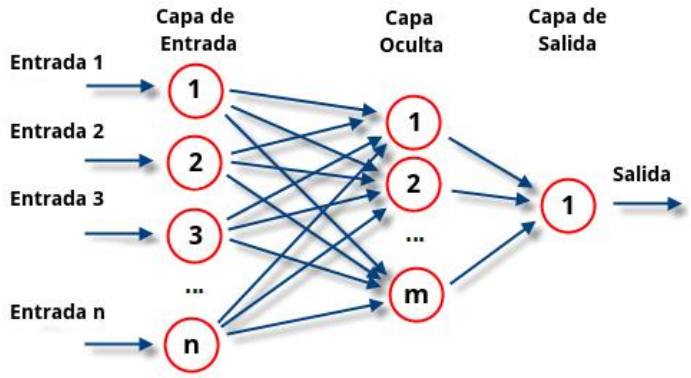


Figura 7. Esquema Perceptrón Multicapa.

La primera capa, denominada capa de entrada, es la que recibe los inputs. A cada neurona de esta capa le corresponde un dato del input, de forma que tiene que haber tantas neuronas como datos de entrada se le suministren a la red.

Las capas ocultas son las capas intermedias. Están compuestas por neuronas que reciben como entradas las salidas de cada una de las neuronas de la capa anterior. Cuantas más capas ocultas posea la red, mayor es su complejidad y su capacidad de aprendizaje.

La capa de salida la forman un número de neuronas igual al tipo deseado de salida.

En el estado del arte actual, las redes con estructura similar al Perceptrón Multicapa se denominan redes completamente conectadas (*full connected*), y suelen formar parte de redes con una estructura más compleja, como las Redes Neuronales Convolucionales que se detallan más adelante en el apartado 1.2.4.

1.2.3 Entrenamiento

Los parámetros que afectan a una red se pueden clasificar en dos grupos: los parámetros que se modifican a lo largo del aprendizaje, que forman parte del modelo como tal; y los

parámetros que se configuran antes del aprendizaje y que no se obtienen a partir de los datos de input, llamados hiperparámetros. El entrenamiento de una red neuronal artificial se entiende como el proceso por el cual los parámetros de la red se modifican para producir un mejor resultado dados unos ciertos hiperparámetros.

1.2.3.1 Retropropagación

Uno de los métodos más utilizados para el entrenamiento supervisado, inputs formados por datos con etiquetas, de redes neuronales es el algoritmo de propagación hacia atrás de errores o Retropropagación (*Backpropagation*). Este algoritmo sirve para entrenar redes neuronales prealimentadas (*feed-forward*), es decir, redes donde sus conexiones no forman un ciclo.

El entrenamiento con Retropropagación consta de dos fases. La primera fase es una propagación desde los inputs hasta generar salidas de la red. Siguiendo un proceso similar al Perceptrón Multicapa, los pesos de las neuronas se inicializan en un valor y transforman los valores de entrada para pasar a continuación por la función de activación. Respecto a la función de activación, al contrario que el Perceptrón original, no puede ser una función de escalón ya que en la siguiente fase depende de su derivada, por lo que se utilizan otro tipo de funciones de activación como las detalladas en el siguiente apartado 1.2.3.2.

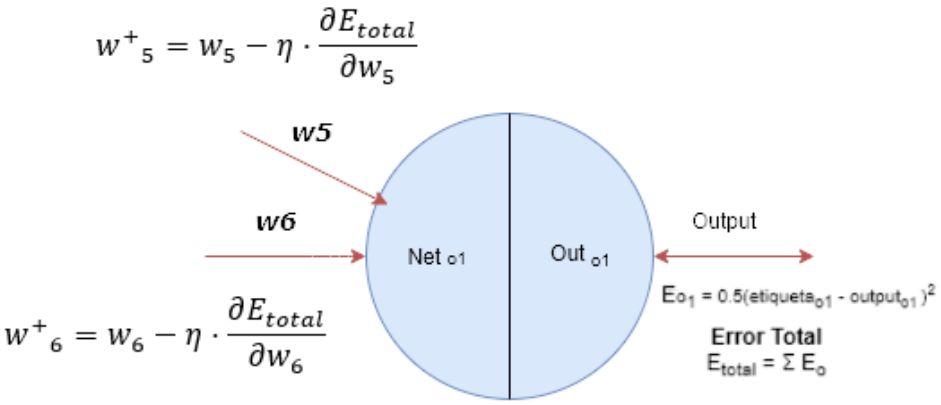


Figura 8. Esquema de Retropropagación. Tras la primera fase de propagación, con el gradiente del error respecto a los pesos, se recalculan sustrayendo la tasa de aprendizaje (η) por la derivada correspondiente del gradiente.

En la segunda fase, la Retropropagación, se compara las salidas con las etiquetas y se calcula el error mediante una función denominada función de coste, que en este caso es el error total cuadrático de la red. La forma en que ajusta los pesos es calculando el gradiente del error total respecto a todos los pesos mediante la regla de la cadena. Se calculan los nuevos pesos aplicándoles un escalón cuyo valor está en función de la derivada del error

total respecto a cada peso y la tasa de aprendizaje, siguiendo un proceso similar a la figura 8.

1.2.3.2 Funciones de Activación

Como ya se ha mencionado en el apartado anterior, para aplicar Retropropagación la función de activación no puede ser un escalón. Por este motivo se usan diversas funciones de activación, siendo una de las más utilizadas ReLU y sus derivadas.

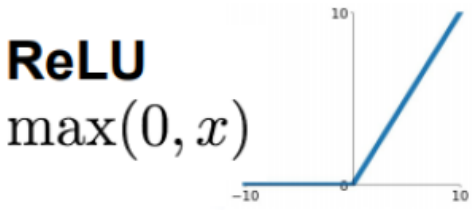


Figura 9. Gráfica ReLU.

- ***ReLU:***

ReLU (*Rectified Linear Units*) [8] , consistente en una función tipo rampa, es la función de activación más popular en el estado del arte actual debido a su simplicidad y rapidez computacional. Solo transmite outputs positivos y de forma lineal.

Derivadas de la ReLU, existen múltiples variaciones de la función que tratan outputs negativos.

Algunas variaciones notables son la Leaky ReLU [9] que permite un pequeño gradiente y la Parametric ReLU (PReLU) [10] .

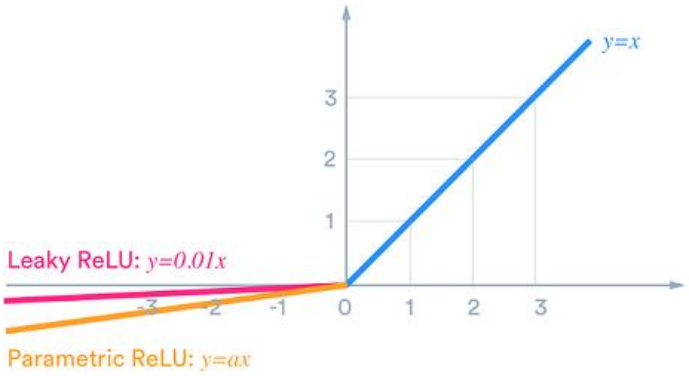


Figura 10. Gráfica Comparación Leaky ReLU y PReLU.

1.2.4 Redes Neuronales Convolucionales

Una de las aplicaciones del Aprendizaje Máquina es la Visión Artificial, el estudio de metodología para adquirir información a partir de imágenes. Una de las estructuras más útiles para el análisis de imágenes con redes neuronales son las Redes Neuronales Convolucionales, ya que utilizan una serie de estructuras de capa matriciales que reciben como entrada datos en forma de una o varias matrices, como puede ser el caso de una imagen en blanco y negro o una imagen RGB.

1.2.4.1 Capa Convolucional

La información de entrada de las Redes Convolucionales la constituyen una o múltiples matrices de datos bidimensionales. Para poder analizar la información de las matrices de entrada se utilizan filtros matriciales con el mismo principio que el Perceptrón.

Cada filtro está constituido por tantas matrices bidimensionales como matrices tenga la entrada (n), normalmente de dimensiones (*kernel*) inferiores (nx3x3, nx5x5, nx7x7…), constituidas por pesos. Su función es extraer información de los inputs recorriendo con un patrón fijo de movimiento los datos de las matrices de entrada, creando con las salidas una matriz con información extraída del input, denominada mapa de activación (*activation map*).

El movimiento del filtro viene determinado por su paso (*stride*), que determina los intervalos de salto del centro del filtro. Una forma de hacer que el centro del filtro pase por todos los datos de la matriz es aplicar *padding*, que consiste en añadir filas y columnas de relleno con valor nulo de forma que se obtiene un mapa de activación de las matrices completo sin reducir nada.

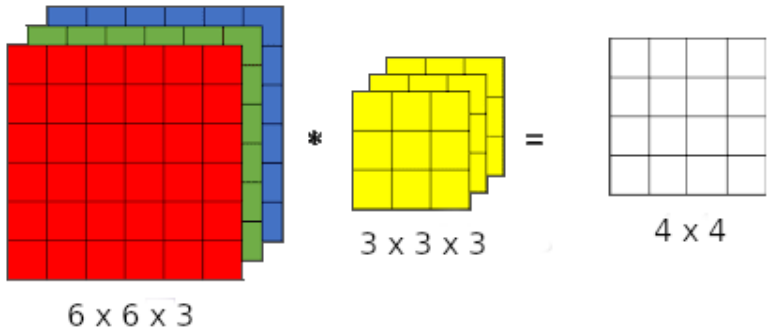


Figura 11. Esquema de funcionamiento del filtro Convolucional. La entrada es una imagen de tres matrices (RGB) y el filtro 3x3 con paso 1 sin padding. El resultado es un mapa de activación 4x4.

Se pueden aplicar más de un filtro a los datos, creando así múltiples mapas de activación. El conjunto de los mapas de activación es lo que se denomina capa convolucional.

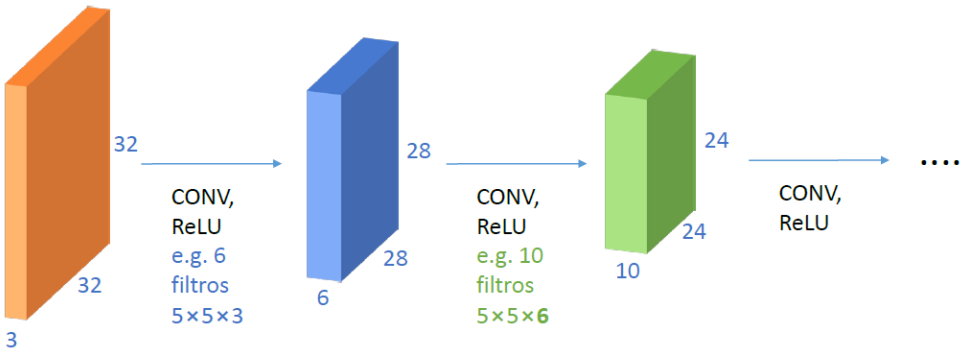


Figura 12. Esquema capas convolucionales. Los filtros generan mapas de activación agrupados como una capa convolucional. A su vez nuevos filtros generan más mapas de activación y más capas convolucionales.

1.2.4.2 Pooling

Una de las formas de evitar que la red aprenda información redundante de entradas similares y le permita producir mapas de activación con más nivel de abstracción, es la utilización de neuronas de reducción de muestreo (*pooling*). Los dos métodos de pooling más habituales son max-pooling y average-pooling.

- ***Max-pooling:***

  El max-pooling consiste en dividir una matriz bidimensional en submatrices más pequeñas de igual tamaño y generar como salida una nueva matriz formada por los valores más grandes de cada submatriz. Las dimensiones que se suelen utilizar para el max-pooling son 2x2, reduciendo los datos a la mitad y por tanto reduciendo los costes computacionales del entrenamiento.

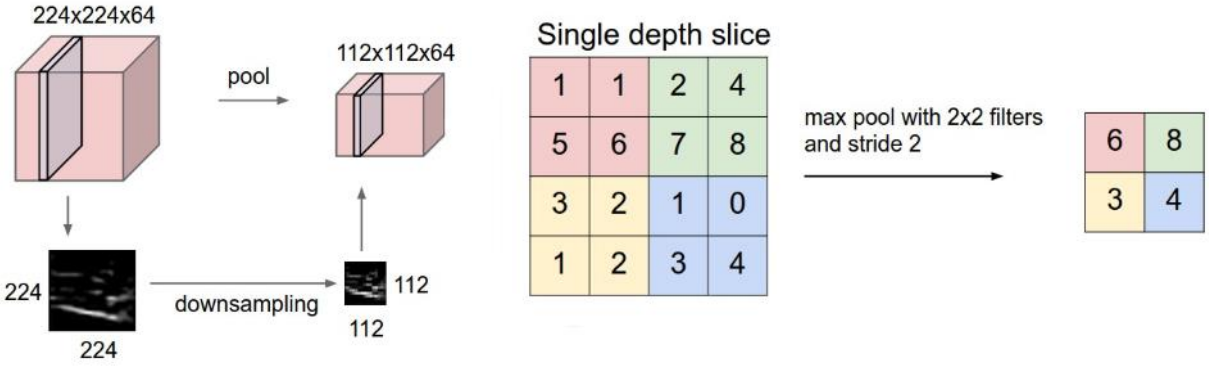


Figura 13. Esquema de funcionamiento del max-pooling. Cada matriz de la capa convolucional es dividida en parches 2x2. Se obtiene como salida una matriz compuesta por los valores más grandes de cada parche reduciendo a la mitad los datos y obteniendo mayor nivel

- ***Average-pooling:***

El principio de funcionamiento del average-pooling es similar al max-pooling, pero en lugar de dar como salida el valor más alto de cada submatriz se obtiene la media de los valores de la misma submatriz.

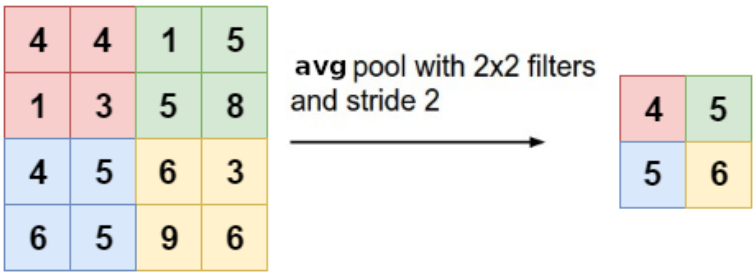


Figura 14. Esquema de funcionamiento del average-pooling.

Mientras que max-pooling se utiliza entre capas convolucionales para reducir el número de datos, el average-pooling se suele utilizar antes de pasar a la parte completamente conectada que se explica a continuación.

1.2.4.3 Combinación con completamente conectadas

Dentro del estado del arte, cuando una red neural tiene partes convolucionales, también suele tener una parte completamente conectada (*full connected*) antes de las salidas de la red. Para poder utilizar la información producida por las capas convolucionales, datos en forma matricial, es necesario convertir las matrices en un vector de una única dimensión en un proceso llamado *flattening* (aplanamiento). Una vez obtenido el vector unidimensional, la red procede con un modelo similar al Perceptrón Multicapa.

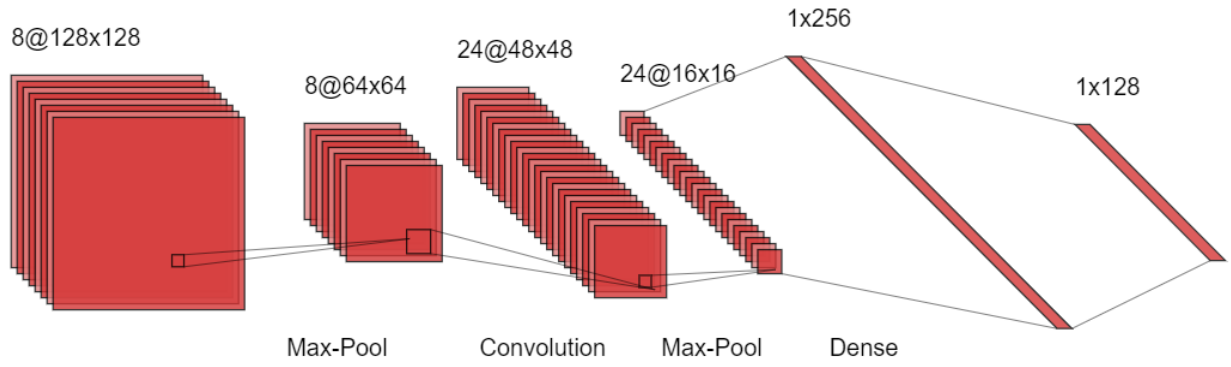


Figura 15. Esquema LeNet-5. Arquitectura interna de LeNet-5 [11] , una de las primeras redes neurales convolucionales.

1.2.5 Estado del Arte

En los últimos años las aplicaciones del aprendizaje máquina y aprendizaje profundo han crecido exponencialmente en multitud de campos de investigación. Uno de estos campos es la visión artificial, el análisis de las imágenes para obtener información. La clasificación de naranjas según su calidad por medio de fotografías entra dentro de este campo.

En visión artificial, el aprendizaje máquina abarca tareas como:

- *Segmentación:* La asignación de píxeles de una imagen a una etiqueta.
  En 2019 se consiguió una precisión del 84.5% [12] en el dataset *Cityscapes,* fotografías de calles urbanas con una resolución de 1024 x 2048.
- *Clasificación:* Categorizar imágenes por etiquetas.
  En 2019 con un mismo método basado en ResNet (un tipo de red que se explicará más adelante) se consiguió un 99.37% de precisión en CIFAR-10 y un 93.51% en CIFAR-100 [13], datasets con imágenes de 10 y 100 etiquetas correspondientemente.
- *Detección:* Detectar y delimitar un objeto de una imagen correspondiente a una etiqueta.
  En 2018 se consiguió una precisión de 86.9% [14] en el dataset VOC 2007, un dataset de imágenes con 20 clases.

Estas tareas más generales se pueden aplicar a temas más específicos como por ejemplo el control de calidad [15] , detección de tumores cerebrales [16] o traductores de texto [17] .

1.3 Visión artificial en líneas de manipulación de cítricos

La aplicación principal de las redes neuronales que se desarrollan en el presente trabajo es su uso en líneas de manipulación de cítricos. A continuación, para entender las posibles utilidades del sistema de visión artificial de clasificación planteado, se explica el proceso de una línea de manipulación de cítricos y las funciones que podrían realizar las redes en estas líneas.

1.3.1 Líneas de manipulación de cítricos

Una línea de manipulación de frutas es una línea de producción que dispone de los equipos y procesos necesarios para poder comercializar la fruta. Las funciones de estas líneas son: limpiar las piezas de fruta, tratarlas con fungicidas y desinfectantes, seleccionarlas por su calidad, calibrarlas por tamaño y envasarlas. En el caso de las líneas de manipulación de cítricos, el proceso comúnmente utilizado es el siguiente.

Antes de despaletizar las frutas y volcarlas en la una línea de rodillos transportadora, se desinfectan y se aplican fumígenos. Mientras los cítricos avanzan por la línea, se preseleccionan los frutos aptos para su comercialización y se desechan los frutos no aptos. Después de esta preselección, se lavan con agua y detergente por medio de boquillas o

cortinas de espuma y se aplica fungicida a las que se consideren aptas. Debido a que el funguicida se aplica en forma líquida es necesario un proceso de secado antes de la siguiente fase del proceso, el encerado. Mediante boquillas pulverizadoras o rodillos, se recubre superficialmente los cítricos con cera con la finalidad de alargar el tiempo de vida comercial del producto ralentizando su oxidación al mismo tiempo que se le otorga un mejor aspecto al fruto. Tras el encerado, los cítricos entran en unos túneles de secado, y al salir se realiza una selección en función de su calidad. Posteriormente se calibran por tamaño y/o peso por medios mecánicos o electrónicos y se envasan para su expedición.

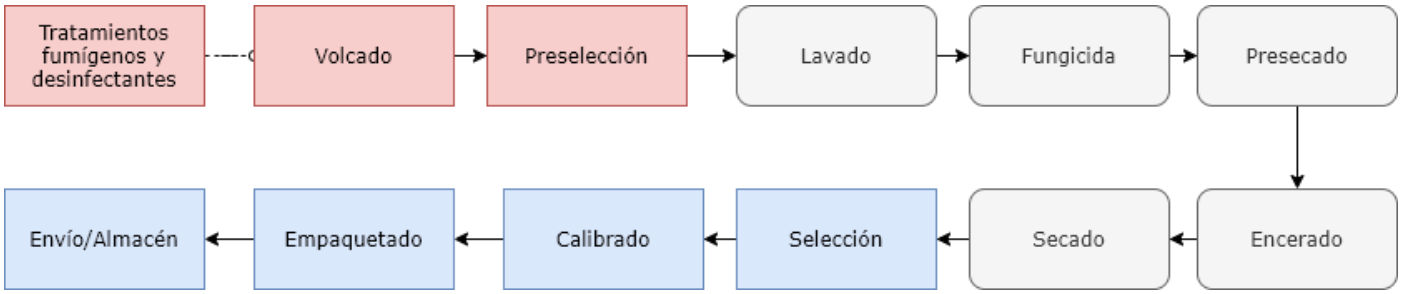


Figura 16. Esquema de una línea de manipulación de cítricos.

1.3.2 Selección por visión artificial

Los sistemas de selección por visión artificial del estado del arte aproximan la calidad de las naranjas mediante algoritmos aplicados a las imágenes que proporcionan un valor a ciertas características, como la forma o el color, con los que se calcula su calidad.

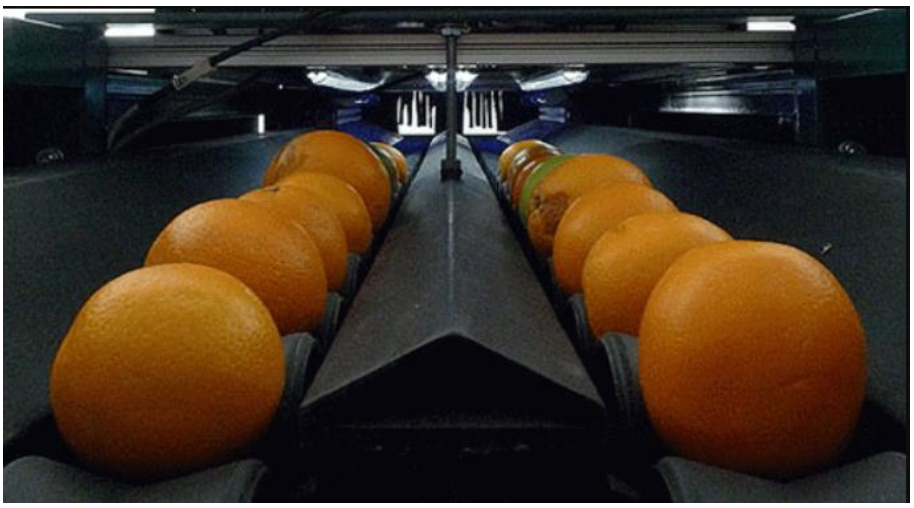


Figura 17. Sistema de clasificación por visión en una línea de manipulación de cítricos.

Un sistema por visión artificial que utiliza una red neuronal convolucional sería capaz de clasificar cítricos en la línea de manipulación, tanto en la preselección de aptas y no aptas como en la selección por calidad de las aptas tras el encerado, de forma automática y eficiente. La ventaja que supone respecto a los métodos con algoritmos es su rapidez y el hecho de que se obtendría una mayor precisión en la selección, ya que los modelos generados tras el entrenamiento de las redes serían capaces de tener en cuenta muchas más características de la naranja durante la clasificación.

Las únicas tecnologías requeridas son un sistema de captura óptica adaptado a la velocidad de la línea, un computador para procesar la red y un sistema de separación electrónico que actúe en función de las salidas de la red. Estos sistemas de visión artificial se pueden encontrar comúnmente en líneas de manipulación de frutos y son adaptables a las redes diseñadas en este trabajo.

Las funciones principales que desarrollan estos sistemas de selección por visión consisten en la purga de naranjas podridas e inservibles en la preselección, para que no contaminen el lavado, y en la clasificación de forma general las naranjas en función de su valor comercial en la selección. Este último proceso permite a la empresa conocer las calidades de los frutos y sus cantidades, así como idear su estrategia comercial en base a estos datos y la situación del mercado.

Capítulo 2:

Objetivos

En este capítulo se exponen los objetivos que se pretenden conseguir en el presente trabajo, tanto generales como específicos, y los parámetros que se analizan para obtenerlos.

2.1 Objetivo general

El objetivo principal del presente Trabajo Fin de Grado es el diseño, implementación y evaluación de distintas arquitecturas de redes neuronales artificiales que sean capaces de clasificar imágenes de naranjas de una línea de producción según su calidad mediante regresión.

2.3 Objetivos específicos

El trabajo se divide en cuatro subobjetivos específicos para conseguir el objetivo principal:

- **La correcta creación y edición de un conjunto de datos de imágenes de naranjas en una línea de producción y sus etiquetas**.
    - La elaboración de un criterio de etiquetado.
    - El etiquetado correcto de las imágenes y la creación del conjunto de datos
- **La elaboración y optimización de distintas redes neuronales artificiales que sean capaces de predecir, con cierto grado de fiabilidad, la calidad de las naranjas.**
    - Diseñar modelos de redes neuronales convolucionales.
    - Encontrar los parámetros más eficientes para las redes.
- **La evaluación del resultado de las distintas arquitecturas.**
    - Conseguir una velocidad de 15 a 20 naranjas por segundo clasificadas (velocidad estándar de inspección en líneas de manipulación de cítricos).
    - Conseguir una precisión adecuada para un proceso de preselección, la eliminación de las naranjas inservibles, y para uno de selección, la clasificación de cítricos según su calidad.

- **Valorar económicamente el proyecto.**

2.3 Parámetros de los resultados

Para poder analizar los resultados que aportan las redes y comprobar que se cumplen los objetivos específicos que se mencionan en el apartado anterior, se observan parámetros basados en el tiempo de entrenamiento y clasificación, la evolución del coste y la precisión de las predicciones:

- **Coste Temporal:**
- El tiempo de entrenamiento por iteración del conjunto de datos se utilizará para comprobar la velocidad de cada red durante su aprendizaje.
- El tiempo de clasificación real por medio de las naranjas por segundo clasificadas por la red una vez esté entrenada.

- **Coste de Entrenamiento:**
- La forma en que evoluciona el valor del coste a lo largo del entrenamiento es un indicativo del proceso de aprendizaje de la red. Siendo la función de coste MSE, el coste de entrenamiento es la media del error cuadrático.

- **Precisión de la predicción:**
- La precisión de las predicciones consistirá en el cálculo del error absoluto de cada una de ellas, tanto de entrenamiento como de validación, y su agrupación en rangos de distancia respecto a la etiqueta.
  Se calcula el porcentaje, respecto al total, de predicciones que distan de un valor menor que 20, 10 y 5 de la etiqueta correspondiente.

Capítulo 3:

Métodos

En este capítulo se exponen los métodos utilizados para llevar a cabo el proyecto. Primero, se explica el proceso seguido para crear un conjunto de datos correctamente etiquetado y el preprocesamiento de las imágenes. Posteriormente, se expone las redes utilizadas, los parámetros a optimizar de ellas y la plataforma de procesamiento de las redes.

3.1 Etiquetado

En Aprendizaje Máquina con aprendizaje supervisado, se entiende como conjunto de datos (*Dataset*) una base de datos matricial donde cada fila la componen dos elementos, los datos de input y su etiqueta.

En el caso de la clasificación de naranjas por su calidad, el *dataset* estaría compuesto por filas de imágenes de naranjas capturadas en una línea de producción y su etiqueta. Las etiquetas de las imágenes representarían la calidad de las naranjas en un valor numérico.

Se dispone inicialmente de 7730 imágenes sin etiquetar para la creación del conjunto de datos. Las imágenes tienen una resolución de 112x560 y constan de 5 fotografías 112x112 de la misma naranja concatenadas de forma horizontal vista desde diferentes orientaciones.

Figura 18. Imagen del dataset utilizado.

Un etiquetado correcto es fundamental para un entrenamiento efectivo de la red, sobre todo en una tarea susceptible de ser subjetiva como la cuantificación de la calidad de un fruto solo con su apariencia. Uno de los problemas que se puede derivar de un etiquetado incorrecto, es que la red reciba información conflictiva respecto a un mismo defecto o característica de la naranja. Esta situación se traduciría en una red incapaz de crear un modelo preciso.

Para evitar este problema, se establecen unos criterios de clasificación, con el fin de objetivar en lo posible el etiquetado, y se establece un proceso de creación del conjunto de datos con cierto grado de autocorrección.

3.1.1 Criterio de Etiquetación

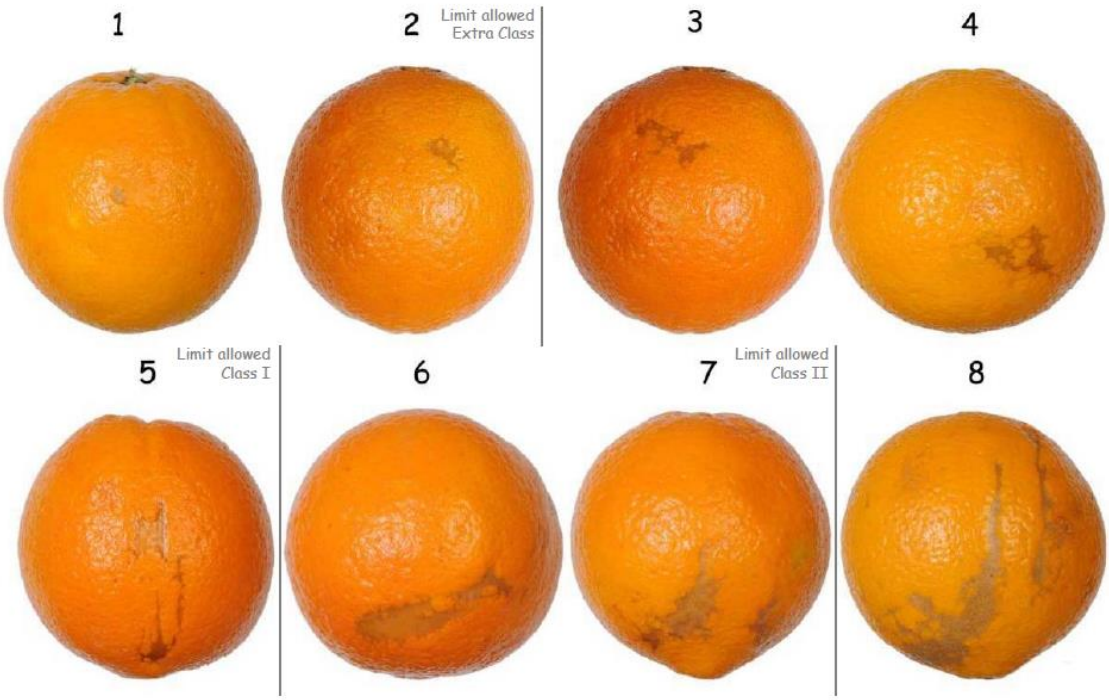
La calidad de la naranja se cuantifica en un valor numérico en gradiente de 0 a 100, siendo 0 el mínimo de calidad y 100 el máximo. Se determina el número aproximado de la calidad de las naranjas en base al ampliamente utilizado estándar de frutas y vegetales de la OECD [18] publicado en 2010. Este documento clasifica las calidades de las naranjas por clases:

-*Extra class:* Naranjas de calidad superior, buen color, sin defectos de forma y defectos superficiales negligibles.

-*Class I:* Naranjas de buena calidad. Los límites de calidad son menos estrictos que en la *extra class*, admitiendo defectos denominados de clase 1.

-*Class II:* Naranjas que no cualifican para las clases superiores, pero cumplen los requisitos mínimos de calidad, admitiendo defectos denominados de clase 2.

-*Out of Grade:* Naranjas no aptas, ya sea por acumulación de defectos clase 1 y 2, porque un defecto sobrepase el umbral de la clase 2 o por defectos llamados eliminatorios, defectos no tolerables (por ejemplo, que la naranja esté podrida).



Oranges	Extra Class	Class I	Class II	Out of Grade
Superficial Dark Coloured Silver Scurfs	1, 2	3, 4, 5	6, 7	8

Figura 19. Clasificación de Naranjas con defecto de marca plateada oscura, OECD.

Tal y como se observa en la figura 19, el documento de la OECD muestra para cada tipo de defecto comúnmente hallado en naranjas imágenes divididas por clase, formando un gradiente similar al que se pretende conseguir con la clasificación por calidad. Se crea y se establece entonces, partiendo de estas clasificaciones del documento, un criterio similar para el trabajo.

Para las naranjas aptas se le otorga un valor a la etiqueta de entre 50 y 100 en función de las clases y cantidades de defectos (indicados en la figura 20). Las naranjas que muestren un defecto que sobrepasa un umbral de clase 2 (que este fuera de rango/*out of grade*), o muestren algún defecto eliminatorio, se consideran no aptas y se les otorga un valor a las etiquetas de entre 0 y 49. Como el documento no da un orden de calidad a los defectos que están fuera del rango se ha propuesto una categorización de éstos en función de si afectan solo a la piel o también al interior (indicados también en la figura 20). En las no aptas, en el caso de que haya múltiples defectos, solo se considera el que reduzca más la calidad.

TABLA CLASIFICACIÓN

Valor	DEFECTOS
100	Ninguno
90	Class I [1]
80	Class I [2]
70	Class II [1]
60	Class II [1] + Class I [1-2]
50	Class II [2]
40	Out of range **(Pitting Superficial/Scratching / External Colour [1] /Other superficial/Shape)**
30	Out of range **(Pitting Non-Superficial/External Colour[2]/Skin texture[1]/Protruding navels/Internal navels)**
20	Out of range **(Unclean/Skin Texture[2]/Creasing/External Colour [3])**
10	Out of range **(Brown rot/Oleocellosis)**
0	Out of range **(Green Mould)**

Figura 20. Tabla de Clasificación. Tabla orientativa de los criterios de clasificación de calidad.

Las naranjas aptas se clasifican en función de la combinación de la clase de defectos. Cada valor tiene asociado la cantidad del tipo de defecto entre corchetes. Por ejemplo, para la calidad 70 le corresponde [1] defecto de clase II.

Para aproximar en clases el valor de las clasificaciones, podría suponerse clase extra entre 90 /100, clase 1 entre 70/90, clase 2 entre 50/70 y fuera de rango todo lo inferior a 50.

3.1.2 Proceso de Etiquetado

El criterio de clasificación basado en defectos es solo una guía, no es infalible. Como se ha mencionado antes, esta tarea de etiquetado es subjetiva hasta cierto punto. Aunque se tengan referencias visuales de los defectos, depende en última instancia del criterio propio del etiquetador. Con este hecho en mente, se ha establecido un proceso de etiquetado que pretende corregir instancias de etiquetado incorrecto.

Para llevar a cabo la etiquetación del conjunto de datos (7730 imágenes) se ha utilizado la herramienta de etiquetado 'dataset.ai2.upv.es/oranges', una plataforma del Instituto de Automática e Informática Industrial de la UPV. Esta herramienta alberga conjuntos de

datos de cítricos y plataformas de etiquetado para tareas de visión artificial, concretamente para regresión, detección y segmentación.

La etiquetación se ha llevado a cabo por 3 etiquetadores (usuarios), entre los cuales se encuentra el autor del trabajo, en 3 fases explicadas por orden a continuación.

- ***Etiquetado individual:***

  En la primera fase, cada etiquetador asigna manualmente una etiqueta a todas las imágenes del conjunto de datos en base a los criterios establecidos. Para ello se utiliza el modo *Standard* de la herramienta, que muestra por orden todas las imágenes y guarda las etiquetas introducidas por cada usuario en un registro individual.

  Una vez etiquetadas todas las imágenes, se activa el modo *Revision*. En este modo se puede navegar todo el conjunto de datos etiquetado por el usuario, mostrando la imagen y la etiqueta asignada, pudiendo ser posible cambiar la etiqueta si es necesario.

- ***Corrección de discrepancias:***

  En la segunda fase se utiliza el modo *Error Fixing* de la herramienta, que compara las etiquetas de los usuarios en todas las imágenes. Si existen diferencias notables entre las etiquetas de algunas imágenes, se vuelven a mostrar por orden para su reevaluación. Iterando este proceso se corrigen las discrepancias del etiquetado causadas por una etiquetación incorrecta, una confusión en los criterios o errores al introducir la etiqueta.

Una vez realizadas estas dos fases se crea un conjunto de datos asociando a cada imagen la media de las etiquetas de los usuarios, pero el etiquetado no acaba aquí.

- ***Comparación:***

  El conjunto de datos generado se utiliza para entrenar una serie de redes, generando modelos que permitan una clasificación automática. Obtenidos ya los modelos, se cargan en el servidor de la plataforma para que clasifiquen todo el conjunto de datos y se almacenan las etiquetas, procediendo a la tercera y última fase del etiquetado, la comparación.

  En el modo *Comparison* de la plataforma, se muestran las imágenes cuyas etiquetas predichas por el modelo distan un error absoluto |etiqueta-predicción| superior a un valor marcado. De esta forma se evalúan estas imágenes y etiquetas analizando si ha fallado el modelo por no reconocer un patrón o si la etiqueta de los usuarios es la que es incorrecta. Si este último es el caso, se modifica la etiqueta.

  Si las imágenes están bien etiquetadas pero la red no relaciona un tipo de defecto con su valor correspondiente de calidad, se catalogan junto a una descripción del defecto en cuestión.

  Una vez catalogadas las imágenes, se aplica un proceso de aumento de datos. Se duplican varias veces las imágenes con los defectos que la red tiene problemas de

modelización con transformaciones de posición, giradas con ciertos ángulos. De esta forma el conjunto de datos dispone de más casos del defecto para que la red aprenda, esta vez correctamente, cómo predecir con mejores resultados imágenes similares.

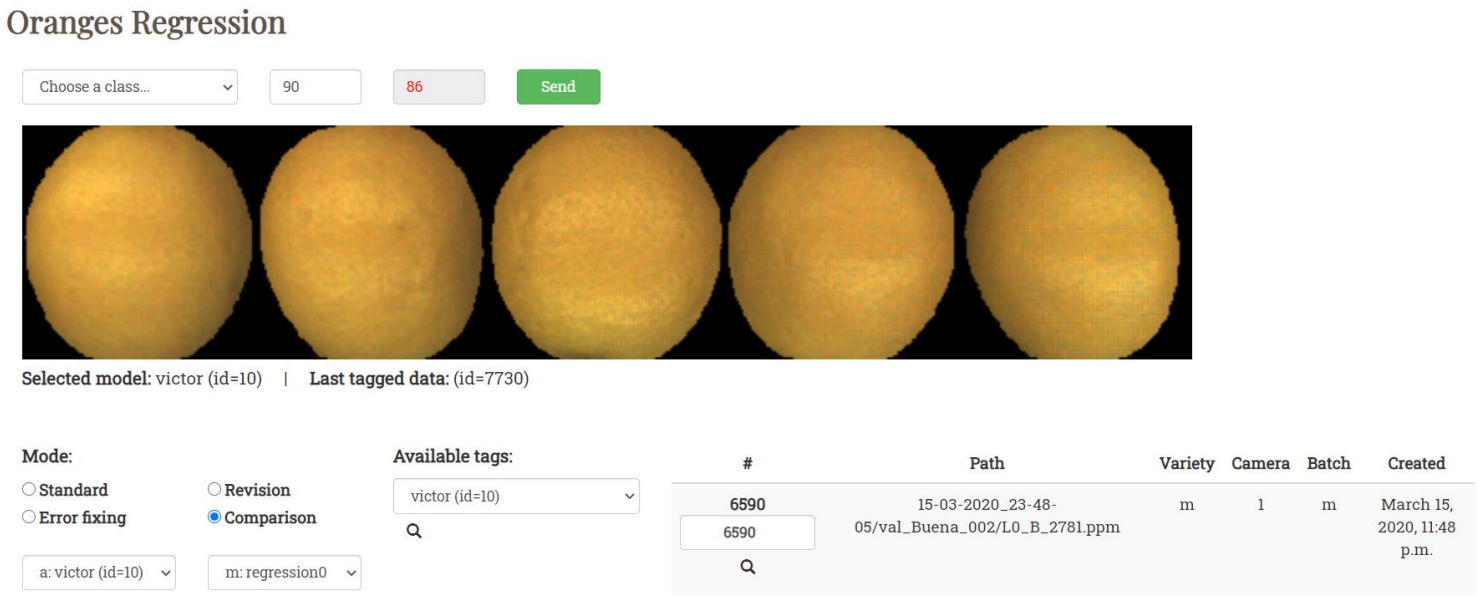


Figura 21. Plataforma de etiquetado utilizada 'datasets.ai2.upv.es/oranges'

3.1.3 Preprocesamiento de las imágenes

Antes de proporcionar las imágenes a la red se someten a transformaciones para facilitar su aprendizaje.

La primera de ellas es la concatenación en planos de las imágenes de una misma naranja. Como se ha mencionado en el apartado anterior, una imagen en realidad contiene 5 fotografías concatenadas horizontalmente. La transformación consiste en cortar la imagen en 5 imágenes cuadradas, cada una con sus 3 planos RGB (112x112x3), y solapar las imágenes de forma que el resultado sea una matriz 112x112x15. De esta forma un mismo filtro afecta a las 5 imágenes simultáneamente.

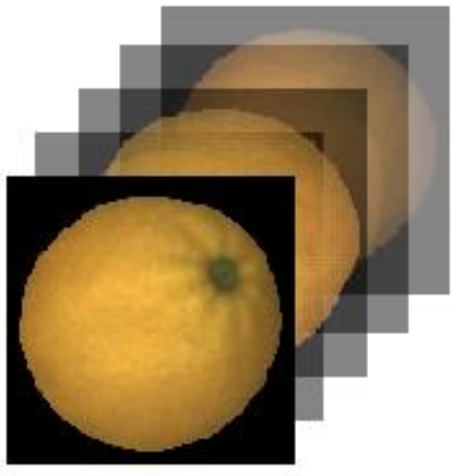


Figura 22. Esquema de la concatenación.

Luego, las imágenes se pasan a tensores para poder ser introducidas en la red, pero antes de transmitirlas como inputs, se normalizan en función de una media y desviación típica. El normalizar las entradas facilita que la red encuentre relaciones entre los datos durante el entrenamiento, sobre todo en un entrenamiento por regresión donde todas las imágenes comparten comúnmente las mismas características.

3.2 Arquitecturas de redes utilizadas

La arquitectura de una red neural consiste en la organización de las capas dentro de la red junto con sus conexiones y sus parámetros. Dependiendo de la tarea que desarrollan, las arquitecturas poseen unas características u otras. En el caso de la clasificación según la calidad de naranjas mediante regresión, las estructuras con el fin más semejante son las de clasificación. Este tipo de redes siguen un fundamento similar a la regresión que se quiere conseguir, que sería como una clasificación en infinitas clases. La principal ventaja de este tipo de regresión frente a una clasificación por clases convencional es que ofrece más información con respecto al rango de calidad de la naranja, pudiendo diferenciar dentro de una misma clase de naranja si tiende a una clase inferior o superior.

Por esta razón se han adaptado arquitecturas de redes neuronales usadas para tareas de clasificación del estado del arte actual.

3.2.1 VGG

Las redes VGG (*Visual Geometry Group*) son unas redes neuronales propuestas en 2014 [19] para el reconocimiento de imágenes a gran escala. Todas tienen el mismo patrón de estructura, capas convolucionales de 64, 128, 256 y 512 filtros de dimensión 3. Las diferencias entre unas y otras residen en el número de capas totales, por ejemplo, la red VGG-13 tiene 13 capas en total, contando convolucionales y completamente conectadas.

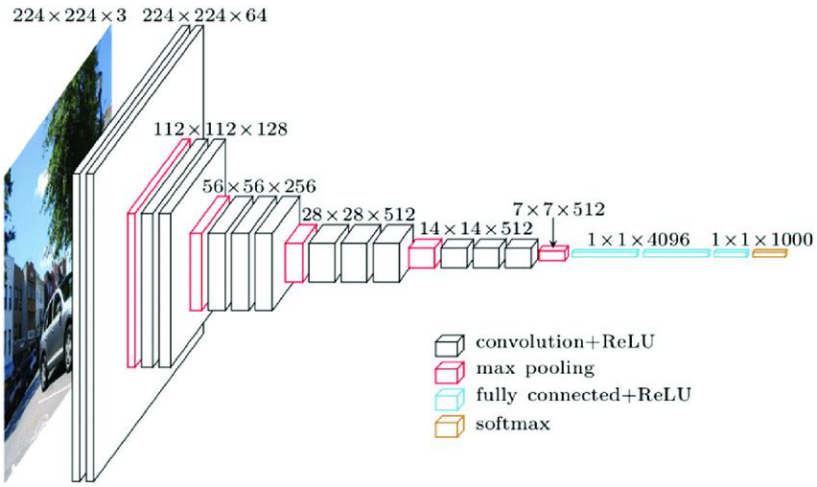


Figura 23. Esquema de la red VGG-16. [20]

En la figura 24 están expuestas todas las configuraciones de las redes VGG. De entre estas redes se ha escogido VGG-16 para la clasificación.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figura 24. Arquitecturas redes VGG.

Las redes VGG están diseñadas para recibir inputs de imágenes 224x224x3 y para dar como salidas una clasificación de 1000 clases. Se ha adaptado VGG-16 para el caso de regresión, modificando la primera y última capa de la red para que acepte un input de 112x112x15 y se obtenga como output un solo valor.

Uno de los inconvenientes de este tipo de redes es el tiempo de entrenamiento y el tamaño del archivo de la red, soliendo ser ambos elevados debido al número de parámetros involucrados (138 millones en el caso de VGG-16).

3.2.2 ResNet

Uno de los problemas con las redes neuronales es el problema del desvanecimiento del gradiente. Se observó que, en las redes profundas, los pesos de las primeras capas apenas cambiaban de sus valores aleatorios iniciales. Esto se debe a que los gradientes calculados en la Retropropagación iban disminuyendo a medida que pasaban de capas y se aplicaban sucesivamente de la regla de la cadena.

Las redes ResNet (*Residual Network*) [21] , permiten que los gradientes de Retropropagación se salten capas al calcularlos, disminuyendo eficazmente el desvanecimiento de los mismos gradientes. Esto se consigue creando mapas de activación copia con funciones de identidad, funciones que devuelven el mismo valor utilizado como argumento, y sumándolas a las salidas de cada capa convolucional.

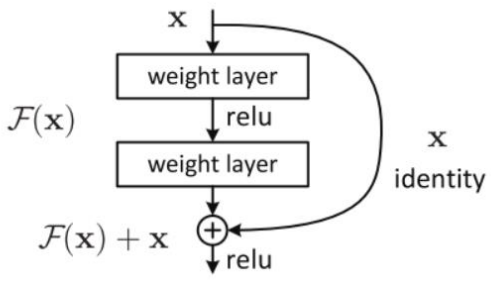
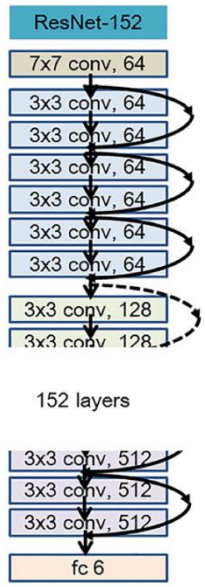


Figura 25. Bloque identidad de una red residual.

ResNet sigue un patrón parecido a VGG en cuanto a capas, formando bloques de 64, 128, 256 y 512 convoluciones con filtros de dimensión 3, y max-pooling entre bloques. Dependiendo del número de capas totales recibe su nombre específico. De entre las redes ResNet se escoge ResNet-152 para la clasificación.



De la misma forma que se han adaptado la red VGG-16, se alteran los parámetros de la primera y última capa de la red acomodando los inputs ya comentados en el apartado 3.1.3.

Aparte de la ventaja sobre la VGG en cuanto al desvanecimiento de los gradientes, las redes ResNet también son más rápidas en el entrenamiento a pesar de tener mayor profundidad. Esto se debe a una menor cantidad de operaciones en las capas convolucionales o FLOP (*floating point operation per second*), sobre todo en la primera capa, donde ResNet aplica una reducción por max-pooling del total de parámetros después de la primera convolución. Otra de las razones de su rapidez computacional es el número de parámetros, rondando los 23 millones en el caso de ResNet-50.

Figura 26. Esquema ResNet 152.

3.2.3 DenseNet

Siguiendo un proceso similar de interconectividad entre capas, se propuso DenseNet [22] en 2017, consistente en redes con bloques donde cada bloque tiene acceso directo al gradiente de la función de coste, la función que proporciona el error, y al input inicial del bloque.

La forma en que DenseNet realiza estas conexiones es diferente de ResNet. Mientras que ResNet suma el mapa de activación resultante de cada convolución con el mapa de activación entrante de dicha capa, tal y como se ve en la figura 25, DenseNet utiliza como entrada para cada capa la concatenación de los inputs de las capas anteriores del mismo bloque.

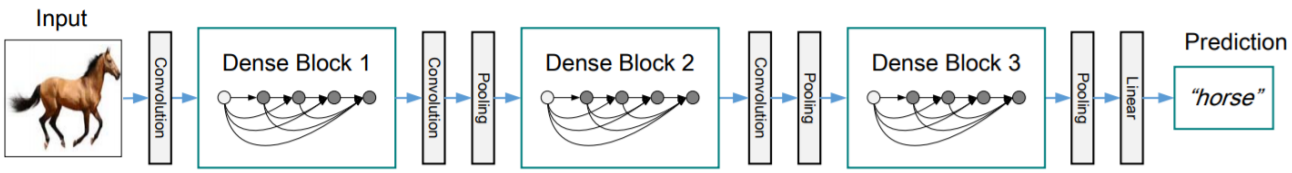


Figura 27. Esquema arquitectura DenseNet de 3 bloques. Se realiza una primera convolución a la imagen. El mapa de activación resultante pasa al primer bloque, donde los mapas de salida de cada capa se concatenan para servir de entrada a la capa siguiente. Tras cada bloque se aplica una capa de transición que realiza pooling.

Dentro de un bloque de DenseNet, las convoluciones se organizan en parejas formando una unidad. Esta unidad la compone una convolución con filtros 1x1 seguida de otra convolución con filtros 3x3. La primera convolución actúa como cuello de botella (*Bottleneck*) y reduce el número de mapas de activación de entrada para la segunda, mejorando así la eficiencia computacional.

Layers	Output Size	DenseNet-121
Convolution	112 × 112	
Pooling	56 × 56	
Dense Block (1)	56 × 56	[1 × 1 conv, 3 × 3 conv] × 6
Transition Layer (1)	56 × 56	
	28 × 28	
Dense Block (2)	28 × 28	[1 × 1 conv, 3 × 3 conv] × 12
Transition Layer (2)	28 × 28	
	14 × 14	
Dense Block (3)	14 × 14	[1 × 1 conv, 3 × 3 conv] × 24
Transition Layer (3)	14 × 14	
	7 × 7	
Dense Block (4)	7 × 7	[1 × 1 conv, 3 × 3 conv] × 16
Classification Layer	1 × 1	
		fully-connected,

Figura 28. Esquema de la arquitectura de la red DenseNet 121.

Los bloques están formados por un número concreto de estas unidades, dependiendo de la profundidad deseada de la red. Para la clasificación, la red escogida es DenseNet-121, con aproximadamente 9 millones de parámetros.

Las redes DenseNet están diseñadas para recibir inputs 112x112, por lo que solo se adapta la profundidad de la primera convolución y la capa de salida de la parte completamente conectada.

3.3 Ajuste de hiperparámetros

El proceso de ajustar los hiperparámetros, introducidos en el apartado 1.2.3, tiene como objetivo la mayor optimización posible del entrenamiento de la red (*fine-tuning*). La forma convencional de optimizar los hiperparámetros consiste en ir probando posibles combinaciones de éstos, analizando los resultados y seleccionar la combinación que proporcione los mejores. Para ajustar los hiperparámetros de las redes anteriormente expuestas se hacen pruebas con un subconjunto de datos de menor tamaño al que se utilizará una vez entrenada la red.

Veremos a continuación los hiperparámetros para la optimización del entrenamiento y las regularizaciones que se aplican para evitar posibles problemas en la generación de modelos durante el entrenamiento.

3.3.1 Función de Coste

Como se ha comentado en el apartado 1.2.3.1, el algoritmo de Retropropagación modifica los pesos de las capas en función de la variación de un error. La función de coste es la que calcula ese error, también llamado coste.

Dependiendo de las salidas de la red y la tarea a la que la red está destinada se utiliza una función de coste u otra. En la tarea de regresión que se quiere conseguir usaremos la función del error cuadrático medio (*Mean Squared Error),* MSELoss.

$$MSE = \underbrace{\frac{1}{n}\sum_{i=1}^{n}}_{\text{Conjunto de datos}} \underbrace{(y_i}_{\text{Output}} - \underbrace{\hat{y}_i)^2}_{\text{Etiqueta}}$$

Figura 29. Formula del error cuadrático medio para un conjunto de n datos.

3.3.2 Optimizador

El optimizador es el que se encarga de la modificación de los pesos con el fin de reducir el coste. En un principio, el optimizador altera los pesos de forma negativa en función de la derivada parcial del error para cada peso y la tasa de aprendizaje durante la Retropropagación (siguiendo un esquema similar a la figura 8), intentando llegar al mínimo de la función de coste. Este método de optimización se denomina gradiente descendiente (*Gradient Descend*).

Uno de los problemas del gradiente descendiente es la irregularidad de la función de coste, pudiendo estancar el aprendizaje en un mínimo local o en zonas de poca variación (puntos de silla). Algunas formas de solventar este problema es limitar el cálculo de la derivada y aplicar un "momento" a las modificaciones para que tengan tendencias similares a las anteriores variaciones, tal es el caso del optimizador SGD (*Stochastic Gradient Descent*), que permite la implementación de un valor de momento.

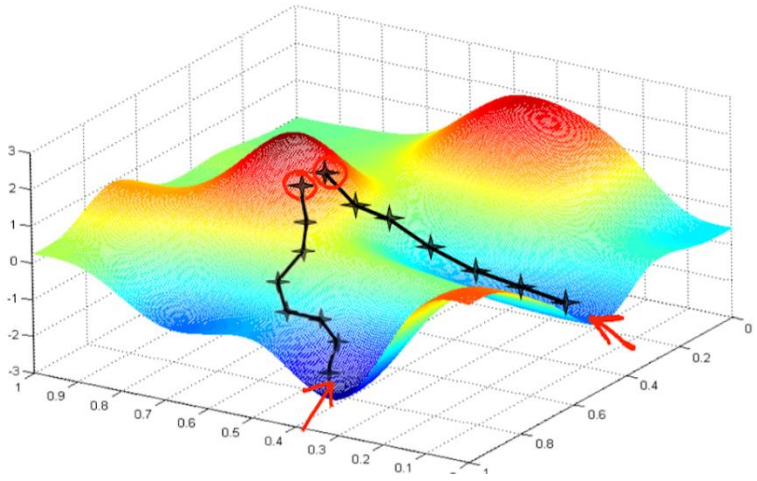


Figura 30. Representación gráfica del gradiente descendiente. Esta forma de optimización busca los mínimos del coste, pero puede estancarse en un mínimo local o puntos de silla. También dependiendo de la tasa de aprendizaje tarda más tiempo en alcanzar un mínimo, si la tasa es demasiado pequeña, o no es capaz de encontrarlo y estancarse, si la tasa es demasiado grande.

Otra forma adicional de mejorar el gradiente descendiente es la aplicada por el optimizador Adagrad, que atribuye a cada peso un factor de aprendizaje específico. Inicialmente estos factores son valores aleatorios, pero se modifican en función del gradiente acumulado en cada iteración.

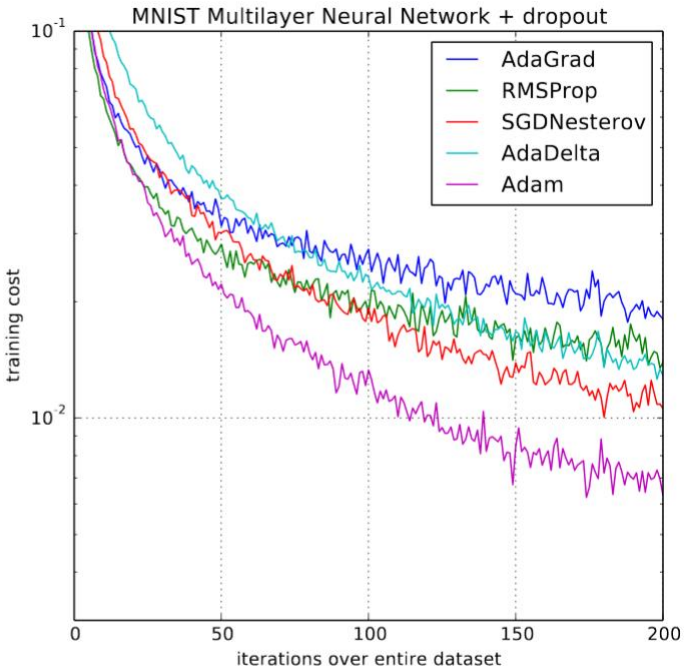


Figura 31. Comparación entre optimizadores por iteración. Adam es el que otorga un descenso del coste con mayor velocidad. El comportamiento de seria Adamax es casi igual. [24]

El optimizador que utilizaremos es una versión de Adam [23] , un optimizador que combina todo lo anteriormente expuesto y añade algunas mejoras más, como la adaptación automática de la tasa de aprendizaje a lo largo del entrenamiento y el uso de la media de los gradientes anteriores con una degradación aplicada. Concretamente, se utilizará Adamax, una modificación de Adam propuesta en el mismo artículo [23]. La diferencia entre Adam y Adamax es que Adamax toma los valores más altos de Adam en cuanto a la degradación de la media de los gradientes anteriores.

3.3.3 Regularización

La regularización consiste en aplicar métodos que eviten problemas en el entrenamiento como el sobreajuste (*overfitting*). El sobreajuste se produce cuando una red pierde capacidad de generalización para clasificar debido a una especialización de los pesos a los datos de entrenamiento, perdiendo como consecuencia precisión en la validación y test.

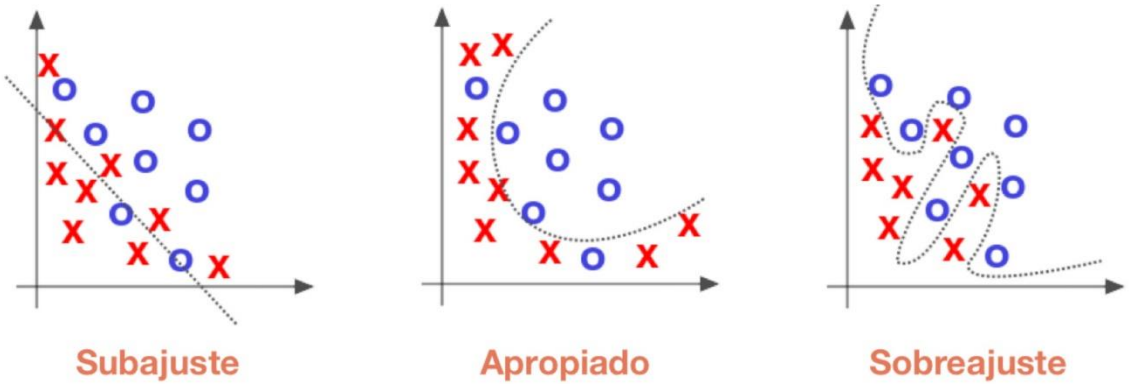


Figura 32. Ejemplo gráfico de sobreajuste y subajuste. En un problema de clasificación, el sobreajuste evita buenas predicciones ya que la red aprende un modelo demasiado a medida del conjunto de datos de entrenamiento.

El caso contrario es el subajuste (*underfitting*), que no es capaz de producir generalizaciones por falta de complejidad de la red.

Con el fin de evitar el sobreajuste se han aplicado las siguientes regularizaciones:

- ***Dropout:***

El *dropout* [25] es un tipo de regularización estática que impide, con una probabilidad determinada, la transmisión de información de neuronas seleccionadas aleatoriamente. Normalmente, y también en el caso de las redes utilizadas, se aplica en las capas completamente conectadas. El hecho de "ignorar" una proporción de neuronas fuerza a la red a aprender características más robustas de los inputs. Una desventaja de aplicar *dropout* es que el entrenamiento tarda más en converger [26], aunque cada iteración dure menos por la reducción de parámetros.

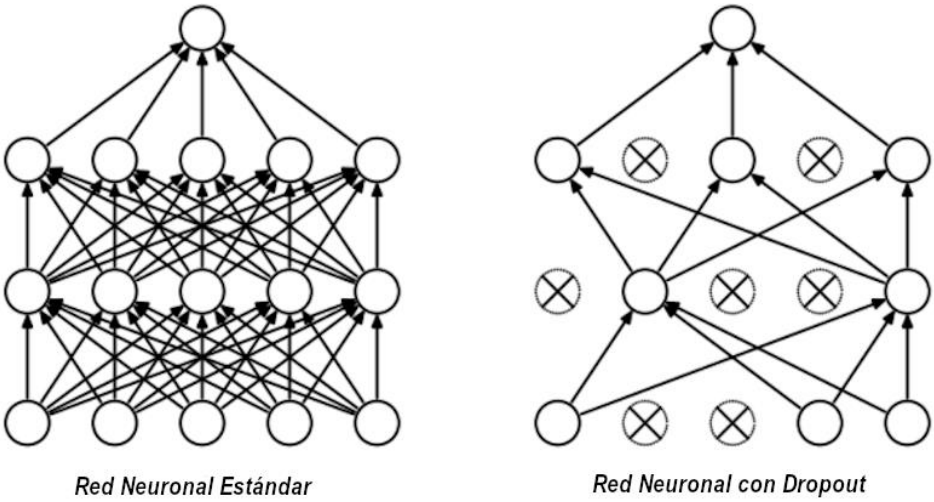


Figura 33. Funcionamiento del Dropout.

- ***Normalización por Lotes:***

La normalización por lotes (*Batch Normalization*) [27] consiste en calcular la media y desviación típica de las salidas de la capa anterior y normalizar dichas salidas restando la media y dividiendo por la desviación típica. Esta forma de regularización se puede aplicar tanto a las capas convolucionales como a las capas completamente conectadas.

Los dos principales efectos de la normalización por lotes son que permite el uso de tasas de aprendizaje más altas a la vez que reduce el sobreajuste. También evita que se modifique la distribución inicial de los inputs al pasar los datos capa tras capa, problema que se puede dar en redes neuronales profundas.

- ***Weight Decay:***

El *weight decay* consiste en penalizar la función de coste de forma que se generan modelos que generalicen mejor. Este efecto se consigue añadiendo términos a la función de coste.

$$J = \frac{1}{M}\sum_{i=1}^{M}(real_i - estimado_i)^2 + \alpha\frac{1}{2N}\sum_{j=1}^{N}w_i^2$$

Figura 34. Weight decay en función de coste medio cuadrático.

Durante la Retropropagación, no solo se substrae a los pesos la derivada parcial del coste, también se resta la derivada del nuevo término añadido.

- ***Aumento de Datos:***

  Otra forma de evitar sobreajuste es aumentando el tamaño del conjunto de datos. De esta forma, cuanta más variedad de inputs reciba la red, menos probable es que desarrolle un modelo solo acorde a esos datos y no generalice.

  Una forma de aumentar el conjunto de datos sin necesidad de nuevas imágenes son las transformaciones, funciones con una probabilidad de modificar las imágenes. De esta forma, a cada iteración del conjunto de datos, algunas imágenes se ven modificadas y la red se entrena con un conjunto distinto cada vez. En el entrenamiento para la clasificación de naranjas se han aplicado volteos horizontales y verticales a las imágenes con una probabilidad del 50%.

3.3.4 Búsqueda en Cuadrícula

El método para encontrar los hiperparámetros óptimos que se utiliza es la búsqueda en cuadrícula (*grid search*). Este método consiste en asignar varios valores a cada hiperparámetro y testear todas las posibles combinaciones de éstos en la red. Una vez probadas todas, se analizan los resultados de la validación y se escoge la combinación que mejor resultado ha proporcionado. Esta búsqueda se realiza en primer lugar en un conjunto de datos de un tamaño inferior y busca determinar un rango óptimo de hiperparámetros para posteriormente aplicarlo al *dataset* entero.

3.4 Ejecución

3.4.1 Pytorch

El lenguaje de programación usado para la confección de las redes es Python, y la biblioteca de aprendizaje profundo utilizada es Pytorch, una biblioteca de código abierto basada en Torch lanzada en 2016. Pytorch ofrece funciones que facilitan la creación y optimización de redes neuronales apoyándose en el uso de unidades de procesamiento gráfico (GPU) para la aceleración del procesamiento. Pytorch dispone de una variada gama de funciones de coste, funciones de activación, formas de regularización y optimizadores.

También dispone de otras funciones que pueden ser útiles para el entrenamiento de la red. De entre ellas se utilizan las que permiten administrar el conjunto de datos y las que permiten modificar la tasa de aprendizaje a lo largo del entrenamiento (LRStep), que la disminuye en función del número de iteraciones que lleva realizadas. Con esta modificación de la tasa de aprendizaje (*learning rate decay)* se consigue una mayor precisión una vez que el entrenamiento se estanca.

Otra de las funciones que ofrece Pytorch es la posibilidad de cargar pesos pre-entrenados de redes modelo que están disponibles en su plataforma online [1]. Estos pesos pre-entrenados para visión artificial ya tienen predisposición a detectar patrones, de forma que, si se cargan en una red, el entrenamiento sería más rápido que la misma red con pesos iniciales aleatorios. Como las redes utilizadas son adaptaciones de modelos que ofrece Pytorch, se han adaptado pesos pre-entrenados a las redes ResNet-50 y VGG-16 en las capas que no se han modificado. Por un problema de compatibilidad no se han adaptado en el caso de DenseNet-121.

3.4.2 Entorno de ejecución

Para la realización del trabajo han sido utilizados dos entornos de ejecución alojados.

El primero es el servidor del Instituto de Automática e Informática Industrial de la Universidad Politécnica de Valencia, que se utiliza para la confección del conjunto de datos tal y como se ha explicado en el apartado 3.1.2. Originalmente, se pretendía que este entorno fuera sobre el que se realizaran también los entrenamientos y evaluaciones de las

[1] *https://pytorch.org/hub/research-models*

redes, pero debido a los problemas derivados de la COVID-19 la accesibilidad del servidor se vio limitada.

Por esta razón, y para trabajar completamente en la nube, se utiliza el segundo entorno alojado para la ejecución del código, la plataforma Google Colab [2]. Esta plataforma gratuita permite ejecutar y programar en Python, y está diseñada para la realización de tareas de aprendizaje profundo, por lo que dispone de las bibliotecas necesarias para ello como Pytorch.

Una de las principales utilidades que ofrece Google Colab es el acceso a una GPU ("Tesla K80") de gran potencia y capacidad. Las características de la GPU en entrenamientos de aprendizaje profundo pueden resultar un factor limitante en cuanto al tiempo de entrenamiento, y es debido a este factor por lo que no se ha optado por un entorno local para la ejecución del código. Aunque la plataforma limite según la cantidad de memoria utilizada y el tiempo de ejecución de uso de la GPU anteriormente mencionada, para la realización del proyecto, supone una mejor opción frente al entorno local.

[2] *https://colab.research.google.com/*

Capítulo 4:

Resultados

En este capítulo se exponen y se comentan los resultados de las redes diseñadas en el trabajo.

Las redes se entrenan con el conjunto de datos etiquetado de 7730 imágenes, de las cuales se destina un 80% (6275) para el entrenamiento y un 20% (1454) para validación. En los dos grupos hay una igual proporción de naranjas aptas (50-100) y no aptas (0-49).

En cada red se especifican los hiperparámetros de tamaño de lote (número de imágenes por iteración durante el entrenamiento), la tasa de aprendizaje y su reducción en escalón (si se le aplica), *weight decay* y *drop out* (probabilidad de ignorar cada neurona).

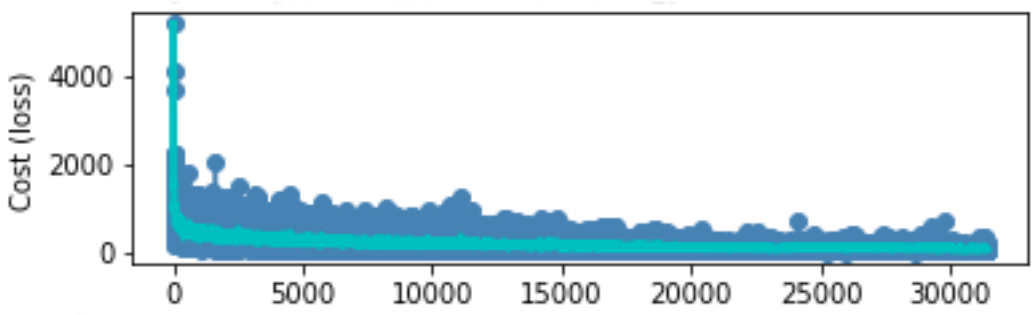
Se muestran para cada red los costes de entrenamiento y las precisiones de las predicciones tal y como se explica en el capítulo 2, en rangos de 20, 10 y 5.

4.1 VGG-16

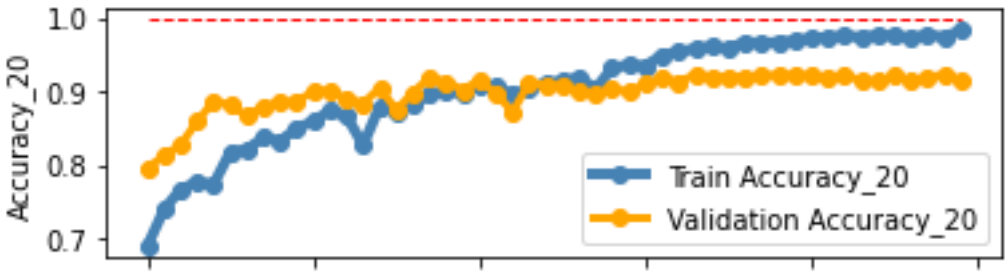
Tamaño de lote (*batch size*): 10 **Weight decay**: 0

Tasa de aprendizaje (*learning rate*): 0.0005 **Drop out**: 0.7

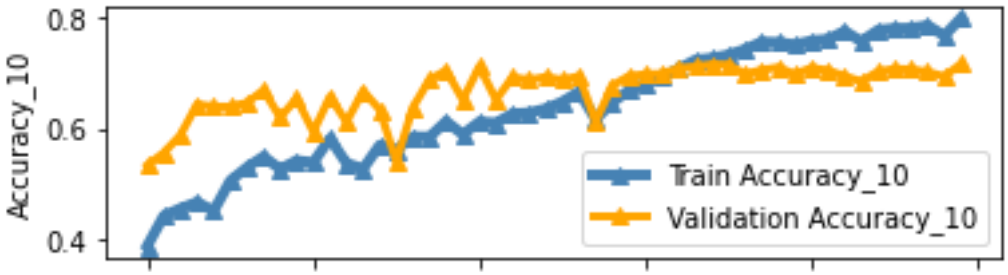
Reducción tasa de aprendizaje: tasa*0.2 cada 30 epochs



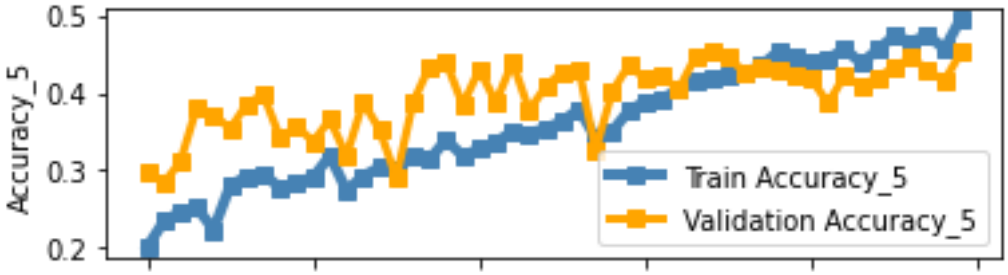
Gráfica 1. Coste de entrenamiento VGG-16.



Gráfica 2. Precisión VGG-16 rango ≤20.



Gráfica 3. Precisión VGG-16 rango ≤10.



Gráfica 4. Precisión VGG-16 rango ≤5.

Tiempo de entrenamiento: 859.859 minutos total; 17.197 minutos/epoch

Tiempo de clasificación: 40.84 naranjas/segundo

Coste final de entrenamiento MSE: 70.2015

Mejor Resultado VGG-16:

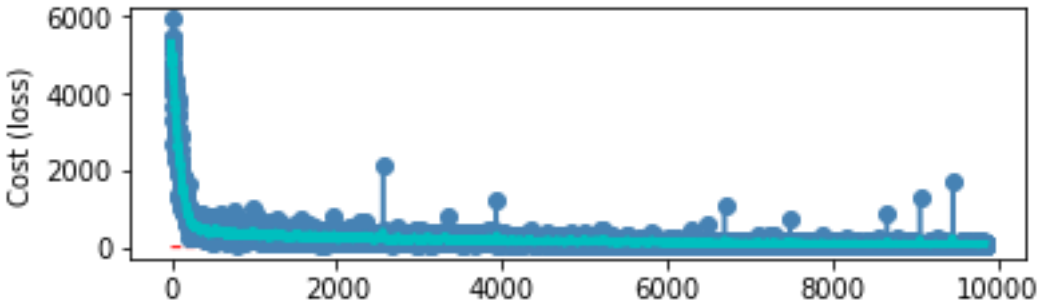
Epoch	**Precisión**	**≤20**	**≤10**	**≤5**
34	%	92.09	71.66	45.39

4.3 ResNet-50

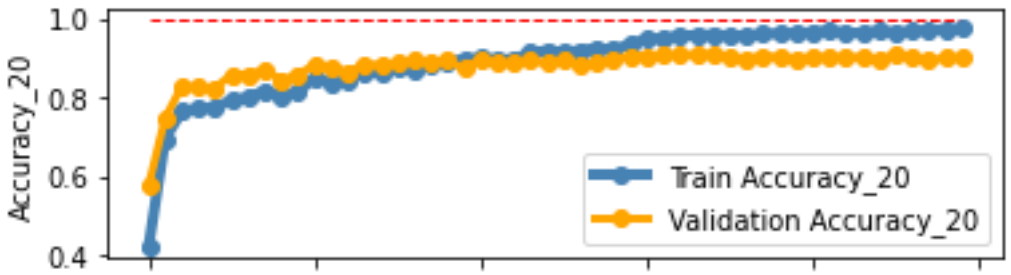
Tamaño de lote (*batch size*): 32 **Weight decay**: 0.001

Tasa de aprendizaje (*learning rate*): 0.0005 **Drop out**: 0.9

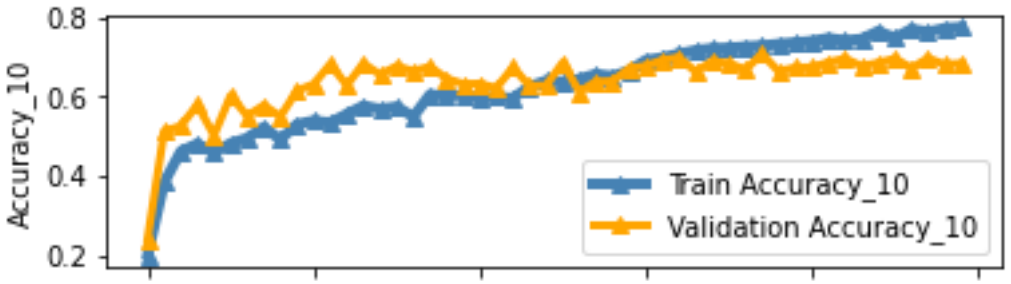
Reducción tasa de aprendizaje: tasa*0.2 cada 30 epochs



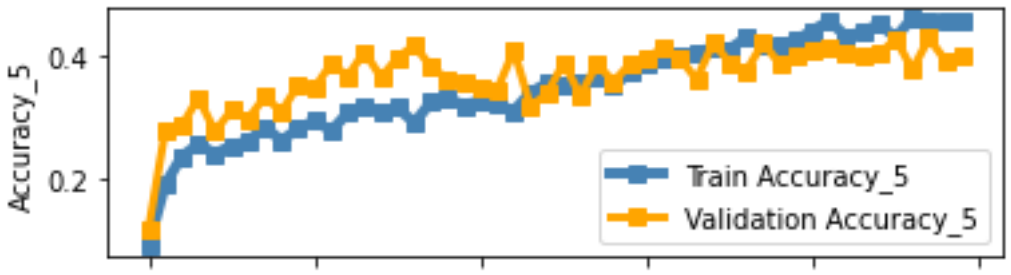
Gráfica 5. Coste de entrenamiento ResNet-50.



Gráfica 6. Precisión ResNet-50 rango ≤20.



Gráfica 7. Precisión ResNet-50 rango ≤10.



Gráfica 8. Precisión ResNet-50 rango ≤5.

Tiempo de entrenamiento: 224.919 minutos; 4.4983 minutos/epoch

Tiempo de clasificación: 51.27 naranjas/segundo

Coste final de entrenamiento MSE: 78.6734

Mejor Resultado ResNet-50:

Epoch	Precisión	≤20	≤10	≤5
37	%	90.72	70.7	42.23

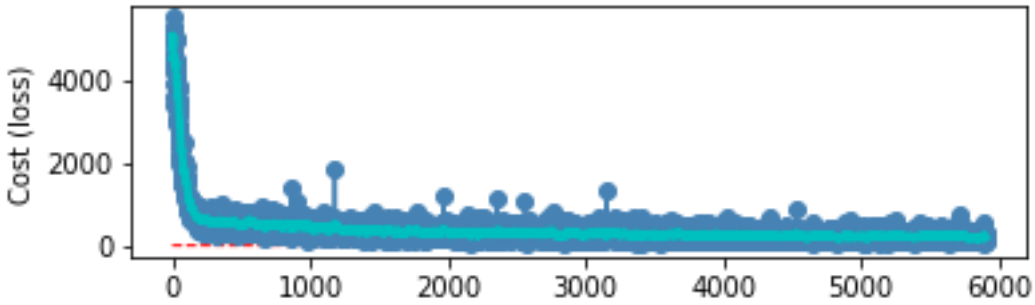
4.4 DenseNet-121

Tamaño de lote (*batch size*): 32

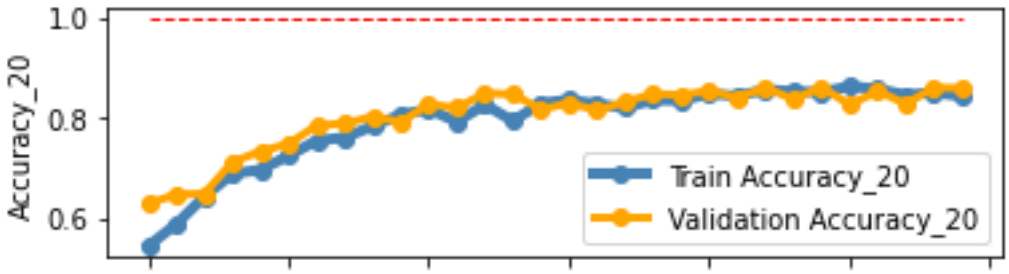
Weight decay: 0.001

Tasa de aprendizaje (*learning rate*): 0.001

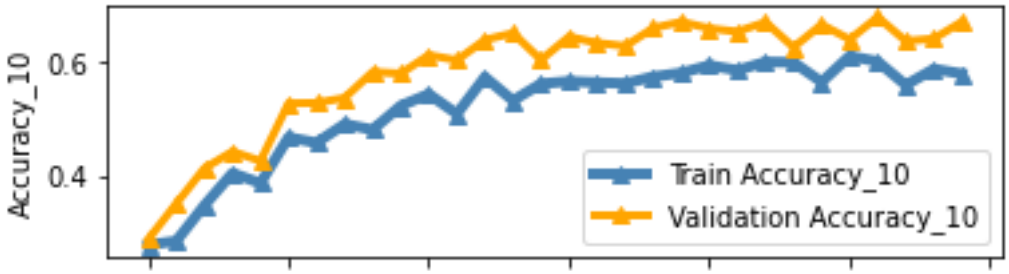
Drop out: 0



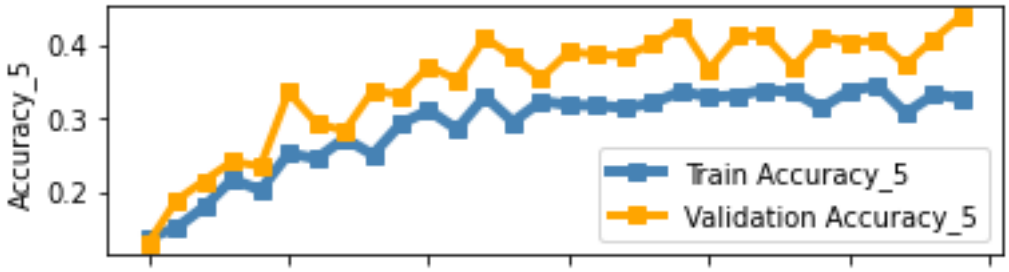
Gráfica 9. Coste de entrenamiento DenseNet-121.



Gráfica 10. Precisión DenseNet-121 rango ≤20.



Gráfica 11. Precisión DenseNet-121 rango ≤10.



Gráfica 12. Precisión DenseNet-121 rango ≤5.

Tiempo de entrenamiento: 290.24 minutos; 9.6746 minutos/epoch

Tiempo de clasificación: 24.31 naranjas/segundo

Coste de Entrenamiento MSE: 203.129

Mejor Resultado DenseNet-121:

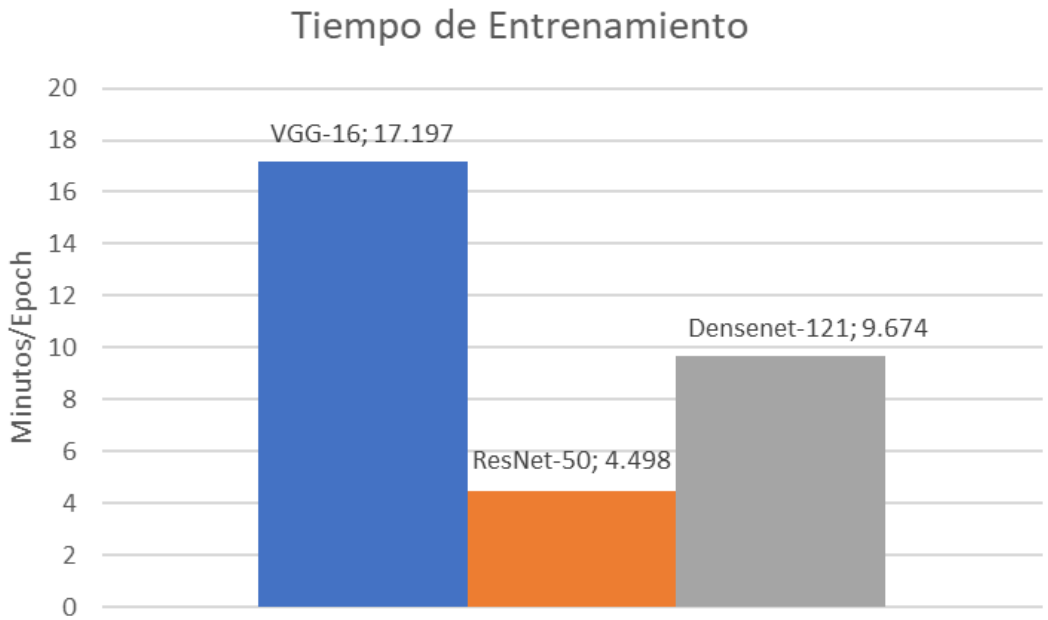
Epoch	**Precisión**	**≤20**	**≤10**	**≤5**
29	%	86.04	67.06	43.88

4.5 Análisis de los resultados

El objetivo que se pretende conseguir es que varias redes consigan un resultado óptimo en la regresión, superando el margen de precisión y coste temporal de clasificación especificados en el capítulo 2.

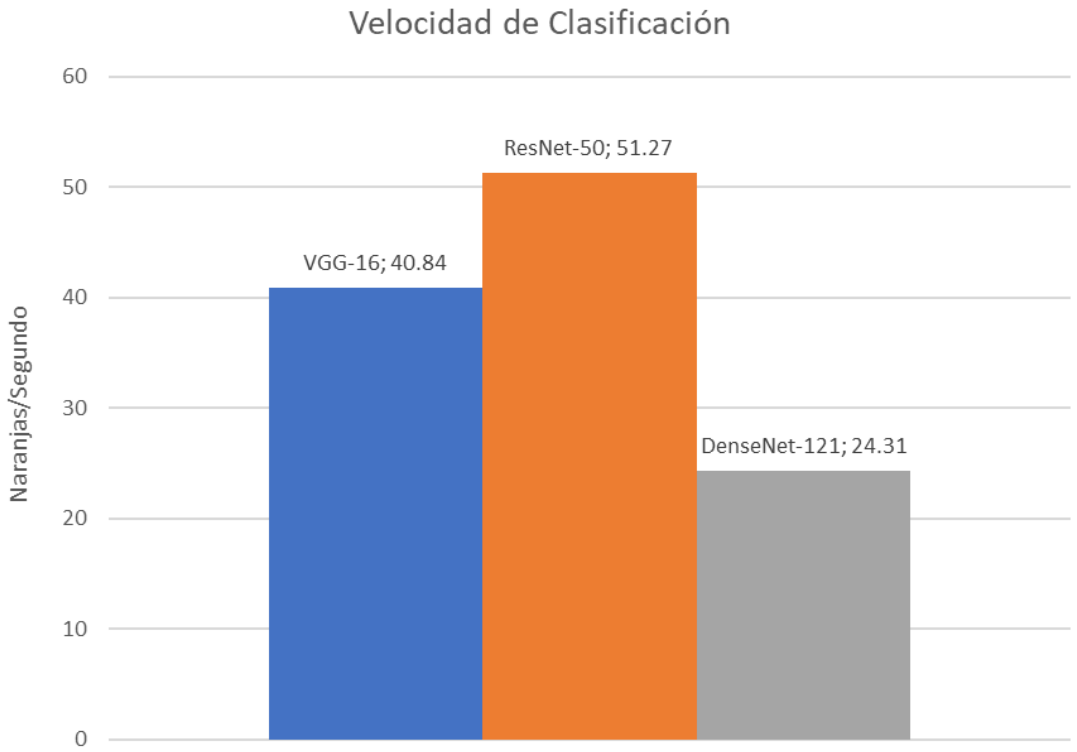
4.5.1 Tiempo de entrenamiento y clasificación

La diferencia más notable, en cuanto al entrenamiento entre las tres redes, es el tiempo de entrenamiento. Tal y como se explica en el capítulo anterior, las redes VGG tienen tiempos de entrenamiento más largos que las otras dos redes debido a una mayor cantidad de datos calculados en cada capa. Esto se puede comprobar en los resultados, siendo el tiempo por *epoch* de la red VGG-16 un 300% del tiempo de ResNet-50 y un 115% del tiempo de DenseNet-121. En términos de eficiencia en el tiempo de entrenamiento, a pesar de que DenseNet-121 posee un menor número de parámetros que ResNet-50, el tiempo medio por capa es casi similar, por lo que ResNet-50 demuestra ser más eficiente.



Gráfica 13. Tiempo de Entrenamiento. Previsiblemente, VGG-16 tiene el mayor coste temporal de aprendizaje. ResNet-50 y DenseNet-121 muestran un tiempo inferior, demostrando que son una mejora en cuanto a eficiencia respecto a les redes VGG.

Respecto a los tiempos de clasificación, las redes muestran velocidades de clasificación que superan los umbrales marcados como objetivos, de 15 a 20 naranjas por segundo. Si bien esta velocidad depende en última instancia de la capacidad del entorno de ejecución, puede servir de indicativo para hacer una comparación entre las redes.

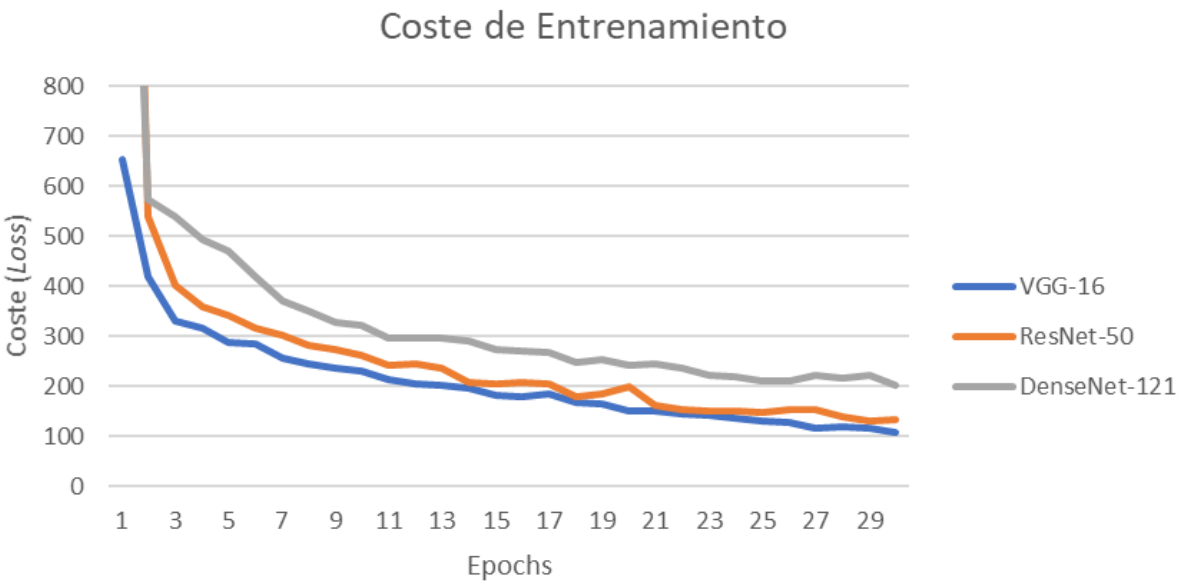


Gráfica 14. Tiempo de Clasificación. Para su cálculo simplemente se ha dividido la duración de clasificación del conjunto de validación entre el total de imágenes de ese conjunto.

ResNet-50 resulta la red más rápida para la clasificación, seguida de VGG-16 y DenseNet-121. Así que, en términos temporales de entrenamiento y clasificación, la red que mejores resultados ha demostrado es Resnet-50.

4.5.2 Coste de Entrenamiento

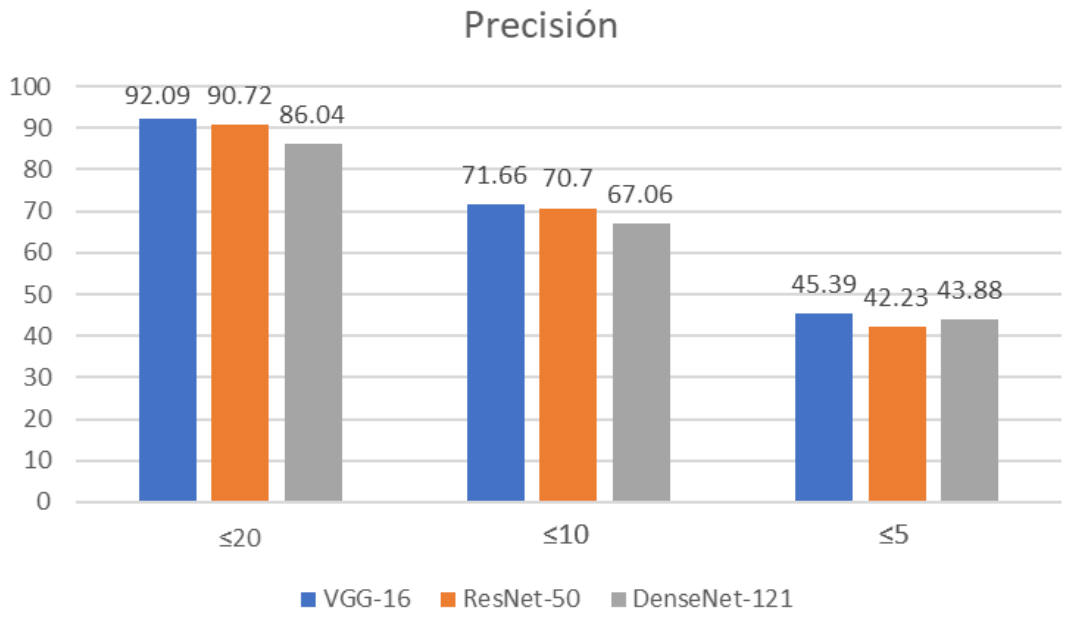
Como ya se ha mencionado en el apartado 3.4.1, VGG-16 y ResNet-50 parten de pesos pre-entrenados en casi todas sus capas, mientras que DenseNet-121 se inicia con pesos aleatorios. Si bien esto no altera el resultado final de la red, el iniciar con pesos pre-entrenados agiliza el entrenamiento, tal y como se ve reflejado en las curvas de la función de coste en la gráfica 14. DenseNet-121 reduce su coste de entrenamiento a un ritmo inferior que ResNet-50 y VGG-16, por lo que requiere más *epochs* para converger.



Gráfica 15. Coste de Entrenamiento. Se muestra el coste de entrenamiento de 30 epochs. Esta gráfica representa la media móvil de los costes de cada iteración, ya que tal y como se observa en las gráficas 1, 5 y 9 el coste puede presentar mucho ruido. Se observa que mientras ResNet-50 y VGG-16 convergen casi al mismo ritmo, DenseNet-121 tarda más.

4.5.3 Precisión

Las predicciones de las distintas redes han dado resultados similares, siendo la red VGG-16 la que obtiene más precisión en sus predicciones, seguida de ResNet-50 y DenseNet-121.



Gráfica 16. Precisión de las redes. La precisión se mide como el porcentaje de predicciones que están en un rango de su etiqueta correspondiente.

El primer objetivo específico de precisión que se perseguía con las redes era la clasificación entre aptas y no aptas para una posible aplicación en la preselección dentro de una línea de manipulación de cítricos. En esta clasificación se pretende descartar las naranjas inservibles como las podridas o mancilladas por insectos, que en el criterio considerado se corresponden a las naranjas con una calidad de entre 0 y 30.

Este objetivo se consigue ya que, dados los rangos de probabilidades resultantes plasmados en la gráfica 16, las naranjas podridas (0 a 20) tienen más de un 90% de probabilidad de ser predichas en ese rango y en el caso de 20 a 30 alcanza aproximadamente un 70%.

El segundo objetivo específico consistía en aproximar la clase de calidad de una naranja para el proceso de selección de la línea. Tal y como se explica en las anotaciones de la figura 20, valores entre situados entre 90 y 100 se consideran clase extra, 70 y 90 clase I, 50 y 70 clase II, y el resto fuera de rango.

Con unas probabilidades del ≈70% de que las predicciones estén en un rango de 10 unidades de las valoraciones reales, las redes puede que no otorguen una predicción exacta, pero si ofrece una aproximación a las clases estandarizadas correspondientes, sobre todo en la clasificación de la clase extra (la de superior calidad), que aun obtendría mejores predicciones ya que las predicciones no sobrepasan los extremos (0 y 100).

Capítulo 5:

Conclusiones

En este último capítulo se exponen las conclusiones del proyecto, comentando las metas alcanzadas, posibles formas de mejora y las limitaciones encontradas durante su realización.

En el presente Trabajo Fin de Grado, se ha creado y editado un conjunto de datos bien etiquetado y se han diseñado tres redes neuronales convolucionales de clasificación automática de cítricos en una línea de producción.

Se comprueba que los objetivos específicos marcados se cumplen:

- Con la elaboración de un criterio de calidad basado en las recomendaciones de la OECD y la concepción de un proceso de etiquetado utilizando una plataforma online se consigue crear satisfactoriamente un conjunto de datos etiquetados.
- Diferentes redes neuronales convolucionales capaces de clasificar por regresión han sido diseñadas y se han optimizado empíricamente sus parámetros.
- Dichas redes son capaces, cumpliendo los objetivos de velocidad, de clasificar satisfactoriamente por su calidad las naranjas para un proceso de preselección y selección. De entre las tres redes diseñadas, la mejor opción es VGG-16, ya que con ella se obtiene la mejor precisión de las tres y demuestra una velocidad de clasificación óptima.

La regresión mediante redes neuronales de imágenes ha demostrado ser un método con potencial para la selección de cítricos, la automatización del proceso y la clasificación a altas velocidades. Si bien las redes resultarían de utilidad para la preselección de naranjas comercialmente aptas/no aptas y la aplicación para una selección, se podría mejorar su precisión para hacerlas aún más viables para la selección por clases de calidad de las aptas.

Una de las formas de mejora tiene que ver con los criterios de etiquetación. Mientras que el proceso de etiquetado utilizando la herramienta del Instituto de Automática e Informática Industrial ha demostrado ser eficiente, el criterio propuesto es relativamente subjetivo en cuanto al efecto de las numerosas formas en que se puede presentar los defectos sobre la calidad. Un criterio más específico y objetivo ayudaría a las redes a conseguir una mejor clasificación.

El tamaño y la variedad del conjunto de datos también ha influido en las capacidades de predicción de las redes. De las imágenes utilizadas para el conjunto de datos, aproximadamente la mitad son naranjas aptas y la otra mitad no aptas, pero no hay semejante equidad en cuanto a la representación de los distintos tipos de defectos y su posible variabilidad. Un conjunto de datos de mayor tamaño implicaría necesariamente más variedad de casos de clasificación para un mejor aprendizaje y como consecuencia una mayor precisión. También se podrían aplicar transformaciones a las imágenes semejantes a las que se utilizan en la segmentación, resaltando la forma, defectos y color para un aprendizaje de menor complejidad.

Cabe destacar el tiempo de entrenamiento en el desarrollo de las redes neuronales como una de sus principales limitaciones ya que, dependiendo de la arquitectura de la red, la cantidad de parámetros involucrados en el aprendizaje y el tamaño del *dataset*, su duración puede ser de horas o incluso días dependiendo de la capacidad del entorno de ejecución. A los factores anteriores, también se le suman las limitaciones de uso temporal y capacidad de procesamiento del entorno alojado. Debido a estas razones, junto a la naturaleza empírica del refinamiento de los hiperparámetros y métodos de regularización de cada una de las redes, el entrenamiento ha ocupado la mayor parte de la duración del proyecto.

Bibliografía

[1] Sánchez-Salmerón, Antonio-José & Benlloch-Dualde, J.-V & Christensen, S. & Walter, Mette. (1996). Weed mapping in cereal crops using image analysis techniques. Proceedings of AgEng96.

[2] Sánchez-Salmerón, Antonio-José & Albarracin, W. & Grau, Raul & Ricolfe-Viala, Carlos & Barat, Jose. (2008). Control of ham salting by using image segmentation. Food Control. 135-142. 10.1016/j.foodcont.2007.02.012.

[3] Ricolfe-Viala, Carlos & Sánchez-Salmerón, Antonio-José. (2011). Optimal conditions for camera calibration using a planar template. Proceedings - International Conference on Image Processing, ICIP. 853-856. 10.1109/ICIP.2011.6116691.

[4] Ivorra, Eugenio & Amat, Samuel & Sánchez-Salmerón, Antonio-José & Barat, Jose & Grau, Raul. (2014). Continuous monitoring of bread dough fermentation using a 3D vision Structured Light technique. Journal of Food Engineering. 130. 8–13. 10.1016/j.jfoodeng.2013.12.031.

[5] Ivorra, E., Sánchez, A. J., Verdú, S., Barat, J. M., & Grau, R. (2016). Shelf life prediction of expired vacuum-packed chilled smoked salmon based on a KNN tissue segmentation method using hyperspectral images. Journal of Food Engineering, 178, 110-116.

[6] Acevedo Correa, Diofanor & Castillo, Piedad & Martelo, Raul. (2018). Neural networks in food industry. Contemporary Engineering Sciences. 11. 1807-1826. 10.12988/ces.2018.84141.

[7]. ROSENBLATT, Frank. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 1958, vol. 65, no 6, p. 386

[8] KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. Imagenet classification with deep convolutional neural networks. En *Advances in neural information processing systems*. 2012. p. 1097-1105.

[9] Maas, Andrew L, Hannun, Awni Y, and Ng, Andrew Y. Rectifier nonlinearities improve neural network acoustic models. In ICML, volume 30, 2013.

[10] He, K., Zhang, X., Ren, S. and Sun, J., 2015. *Delving Deep Into Rectifiers: Surpassing Human-Level Performance On Imagenet Classification*. [online] arXiv.org.

[11] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.

[12] YUAN, YUHUI, CHEN, XILIN and WANG, JINGDONG, 2020, Object-Contextual Representations for Semantic Segmentation. *arXiv.org* [online].

[13] KOLESNIKOV, ALEXANDER, BEYER, LUCAS, ZHAI, XIAOHUA, PUIGCERVER, JOAN, YUNG, JESSICA, GELLY, SYLVAIN and HOULSBY, NEIL, 2020, Big Transfer (BiT): General Visual Representation Learning. *arXiv.org* [online].

[14] SINGH, BHARAT, NAJIBI, MAHYAR and DAVIS, LARRY S., 2020, SNIPER: Efficient Multi-Scale Training. *arXiv.org* [online].

[15] Wang, T., Chen, Y., Qiao, M. *et al.* A fast and robust convolutional neural network-based defect detection model in product quality control. *Int J Adv Manuf Technol* **94,** 3465-3471 (2018).

[16] Zhou, Chenhong & Ding, Changxing & Wang, Xinchao & Lu, Zhentai & Tao, Dacheng. (2019). One-pass Multi-task Networks with Cross-task Guided Attention for Brain Tumor Segmentation.

[17] Edunov, Sergey & Ott, Myle & Auli, Michael & Grangier, David. (2018). Understanding Back-Translation at Scale.

[18] OECD (2010), *Citrus Fruits*, International Standards for Fruit and Vegetables, OECD Publishing, Paris, https://doi.org/10.1787/9789264083745-en-fr.

[19] Simonyan, Karen & Zisserman, Andrew. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv 1409.1556.

[20] Nash, Will & Drummond, Tom & Birbilis, Nick. (2018). A review of deep learning in the study of materials degradation. npj Materials Degradation. 2. 10.1038/s41529-018-0058-x.

[21] He, Kaiming & Zhang, Xiangyu & Ren, Shaoqing & Sun, Jian. (2016). Deep Residual Learning for Image Recognition. 770-778. 10.1109/CVPR.2016.90.

[22] Huang, Gao & Liu, Zhuang & van der Maaten, Laurens & Weinberger, Kilian. (2017). Densely Connected Convolutional Networks. 10.1109/CVPR.2017.243.

[23] Kingma, Diederik & Ba, Jimmy. (2014). Adam: A Method for Stochastic Optimization. International Conference on Learning Representations.

[24] Bock, Sebastian & Goppold, Josef & Weiß, Martin. (2018). An improvement of the convergence proof of the ADAM-Optimizer.

[25] Srivastava, Nitish & Hinton, Geoffrey & Krizhevsky, Alex & Sutskever, Ilya & Salakhutdinov, Ruslan. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research. 15. 1929-1958.

[26] Senen-Cerda, A., & Sanders, J. (2020, February 06). Almost Sure Convergence of Dropout Algorithms for Neural Networks. *arXiv 2002.02247*

[27] Ioffe, Sergey; Szegedy, Christian (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift".

[28] González-Muñiz, Ana. (2018). Aplicaciones de técnicas de inteligencia artificial basadas en aprendizaje profundo (deep learning) al análisis y mejora de la eficiencia de procesos industriales. 10.13140/RG.2.2.15144.72967

DOCUMENTO II: PRESUPUESTO

Contenido del Presupuesto

En esta parte del documento se muestra el presupuesto, una valoración económica del proyecto "Diseño, implementación y evaluación de una red neuronal convolucional de regresión en clasificación de naranjas" acorde al documento recomendado por la UPV: "*Recomendaciones en la elaboración de presupuestos es actividades de I+D+I. Revisión 2018*".

El presupuesto se compone de tres conceptos diferenciados:

-Coste de Personal

De cada participante del proyecto se calcula el coste según la siguiente formula:

$$Coste\ (€)\ =\ Coste\ horario\ (€)\ \cdot\ Dedicación\ en\ horas$$

La dedicación en horas de cada parte ha sido estimada.

-Material Inventariable

Amortizaciones de equipos y licencias de software informático calculadas según la siguiente formula:

$$Coste\ (€)\ =\ \frac{Tiempo\ de\ uso\ (meses)\ \cdot\ Coste\ del\ equipo/software\ (€)}{Periodo\ de\ amortización\ (años)\cdot 12}$$

Clasificación económica del gasto	**Amortización (años)**
Adquisición de equipos para procesos de información	6
Adquisición de aplicaciones informáticas	6

Coste de Personal

Núm	Denominación de Personal	Precio	Cantidad	Total
1	Tutor/Profesor Catedrático de Universidad	51,4	30	1542
2	Graduado en ingeniería de Tecnologías Industriales	20	300	6000
			Total Personal:	7542

Coste de Material Inventariable

Núm	Concepto	Precio	T. amortización	T. uso	Cantidad	Total
1	Ordenador portátil	740,16	6	4	1	41,12
2	Internet/Wifi	28,95	6	4	4	6,43
3	Windows 10	135	6	4	1	7,50
4	Office 365	69	6	4	1	3,83
5	Costes Indirectos				10%	5,89
					Total Material Inventariable:	64,78

Concepto	**Coste Total**
Coste de Personal	7542€
Coste de Material Inventariable	64,78€
Presupuesto de Ejecución Material	7606,78€

	Importe
Presupuesto de Ejecución Material	7606,78€
Gastos Generales (13%)	988,88€
Beneficio Industrial (6%)	456,41€
Presupuesto de Ejecución por Contrata	9052,06€
IVA (21%)	1900,93€
Presupuesto de Base de Licitación	10953,00€

El coste total del proyecto asciende a **DIEZ MIL NOVECIENTOS CINCUENTA Y TRES EUROS**.