UNIVERSITAT POLITÈCNICA DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Politécnica Superior de Gandia

Diseño de un sistema para controlar la producción en árboles frutales

Trabajo Fin de Grado

Grado en Tecnologías Interactivas

AUTOR/A: Doménech Belda, Jordi

Tutor/a: Sendra Compte, Sandra

Cotutor/a: Lloret Mauri, Jaime

CURSO ACADÉMICO: 2023/2024

Quisiera aprovechar esta oportunidad para agradecer a Sandra Sendra Compte y Jaime Lloret Mauri, tutora y cotutor de este Trabajo de Fin de Grado, por la ayuda y apoyo durante la realización de este proyecto.

Agradecer también a todos los profesores y profesoras que me han ayudado a formarme durante toda mi etapa universitaria, por enseñarme todos los conocimientos que ahora poseo.

A mi hermana, a mis padres y a mis amigas y amigos por su apoyo y preocupación mostrada durante todo el proceso estudiantil.

Por último, agradecer a todos aquellos que me han prestado sus campos para poder realizar y probar este trabajo de final de grado.

Jordi Doménech Belda

Universidad Politécnica de Valencia (EPSG)

Índice

Tabla de ilustraciones

Resumen

Como propuesta para mi TFG, se pretende realizar un sistema utilizando una Raspberry Pi y otros componentes como: una pantalla, una cámara o un dispositivo móvil, para poder monitorizar la producción en árboles frutales. Este control se realizará utilizando librerías para el procesamiento digital de imágenes e inteligencia artificial, teniendo en cuenta el tipo de árbol que se va a analizar. Una vez tomada la foto se podrá estimar la producción del mismo.

Todo esto se realizará con el objetivo de ayudar al agricultor a mejorar la calidad de sus explotaciones y tener un análisis previo de la producción para la futura contratación de empleados en la época de recolección.

Palabras clave

Agricultura, procesamiento de imagen, inteligencia artificial, predicción, producción, árboles, monitorización.

Abstract

As a proposal for my Bachelor's Degree Final Project, the intention is to develop a system using a Raspberry Pi and different components such as a screen, a camera or a mobile device, to monitor production in fruit trees. This control will be carried out using libraries for digital image processing and artificial intelligence, tailored to the specific type of tree being analyzed. Once the photo is taken, the production estimate can be obtained.

All of this will be done with the aim of helping farmers improve the quality of their crops and have a preliminary analysis of production for future hiring of employees during the harvesting season.

Key words

Agriculture, image processing, artificial intelligence, prediction, production, trees, monitoring.

1-. Introducción

Uno de los principales problemas en la actualidad es el cambio climático. Las emisiones de gases de efecto invernadero en la Tierra conducen al calentamiento global y con ello, los patrones climáticos cambian y alteran el ciclo normal de la naturaleza. Todo esto lleva a diferentes situaciones anómalas como: elevación de las temperaturas, tormentas mucho más potentes, períodos de sequía más duraderos, desaparición de especies o escasez de alimentos [1].

El sector primario es uno de los más afectados, ya que depende mayormente del clima. Por ejemplo, si no llueve, los huertos no tienen agua o los animales no pueden beber. También, unas altas temperaturas podrían acabar con la producción de cualquier explotación agrícola o ganadera [2].

Cabe destacar que la agricultura es un sector indispensable para la supervivencia del ser humano en nuestro planeta, pero su ciclo natural se está viendo afectado. Debido al cambio climático, las frutas están madurando antes, por lo tanto, los periodos propios del árbol se están viendo alterados. Otro problema que se está observando es que el tamaño de los frutos es menor. Para que el fruto sea del mismo tamaño que siempre, los árboles se deben aclarar más intensamente (reducir el número de frutos por rama) o en consecuencia, la producción puede verse mermada.

Atendiendo a este último punto sobre la cantidad de la producción se procede a desarrollar este trabajo. Uno de los principales gastos de un agricultor es la contratación de personal en la época de recolección. Por esto, se pretende desarrollar un prototipo capaz de estimar la producción en un árbol frutal, en este caso, el melocotonero, para que el agricultor pueda hacer un análisis previo de la producción de sus explotaciones y en consecuencia, realizar una contratación de personal adecuada, es decir, acorde con la cantidad estimada que tiene.

Otra aplicación posible es utilizar el prototipo para las pólizas de seguro de un campo. Las explotaciones agrícolas se aseguran con el objetivo de que si hay una tormenta o algún fenómeno extremo que dañe las cosechas, el agricultor no pierda toda su inversión. En este caso, el agricultor podría saber la cantidad de cosecha estimada y utilizarlo como prueba para obtener la indemnización que le corresponda según el seguro contratado.

En este proyecto se va a proceder a desarrollar dos prototipos para probar y testear el modelo de reconocimiento de frutos. El primer prototipo se realizará utilizando un dispositivo móvil y el segundo una Raspberry Pi con diferentes componentes como una cámara o una pantalla táctil. Con el prototipo que se obtengan mejores resultados, se procederá a analizar su funcionamiento detalladamente.

1.1-. Objetivos

El objetivo principal de este trabajo es realizar una etiquetación o identificación de las frutas (melocotones) en un árbol. Para ello se realizará un estudio sobre las redes neuronales y el aprendizaje automático, con el objetivo de obtener el modelo que mejor se adapte a las necesidades de este proyecto.

Se realizarán y probarán distintos algoritmos preentrenados de machine learning para determinar aquel que mejor funciona y se adapta a las necesidades específicas del proyecto. Este se probará primeramente en un dispositivo móvil y posteriormente en una Raspberry Pi, que contará con una cámara, con la cual se podrá tomar las imágenes de los árboles, y una pantalla, en la cual se podrán observar los resultados obtenidos, es decir, las frutas etiquetadas por el modelo entrenado.

Una vez realizados estos pasos, se validará el prototipo. Con lo que se pretende comprobar si la detección de las frutas es correcta o no, así el agricultor podrá hacer una estimación correcta de la producción que tiene, es decir, obtener los casos en los cuales el sistema tiene un correcto funcionamiento. Además, se desea comprobar su funcionamiento en muestras que no estén en el dataset, es decir, muestras tomadas con alguno de los dos prototipos creados.

Otro objetivo es profundizar sobre el aprendizaje automático, ya que es un campo que tiene muchas aplicaciones diferentes, y además, comprobar hasta que punto pueden resultar útiles las herramientas de entrenamiento gratuitas, como puede ser Edge Impulse.

Finalmente, se pretenden documentar todas las tecnologías usadas con sus especificaciones técnicas. En esta documentación se encontrará expreso todo el proceso de entrenamiento de los sistemas de reconocimiento, las imágenes utilizadas para entrenar y validar el modelo (dataset), el prototipo utilizando la Raspberry y futuras implementaciones para una mejora de la eficiencia. Así como conclusiones y valoraciones personales del mismo.

1.2-. Estructura del proyecto

Para la realización de este trabajo, se va a seguir la siguiente estructura:

En el siguiente capítulo, el número 2, se presenta el estado del arte. En este se van a analizar diferentes artículos y trabajos para tener una idea global de los avances en este campo y aumentar así los conocimientos sobre el aprendizaje automático. Una vez analizado el estado del arte se obtendrán una serie de conclusiones a partir de dicho análisis.

En el capítulo 3 se expone la propuesta del proyecto, con la que pretende explicar el hilo conductor para la realización del trabajo y lo que se pretende lograr. Estará incluida la información sobre el árbol con el que se van a realizar las pruebas del sistema. Así mismo, se presentará la plataforma con la que se pretende realizar el entrenamiento del sistema que va a ser testeado y analizado.

A continuación, en el capítulo 4, se explicará cómo funcionan los modelos de aprendizaje automático supervisado. Asimismo, se desarrollará detalladamente como se ha obtenido y dividido el dataset. Además, en este apartado, se darán a conocer los parámetros e hiperparámetros utilizados para el entrenamiento del modelo y unas gráficas donde se mostrará el resultado de dicho entrenamiento y los valores obtenidos. Finalmente, se analizará detenidamente el conjunto de test.

En el capítulo 5 se explicará cómo se ha procedido a la implementación. Primeramente, se mostrará una propuesta de despliegue en el teléfono móvil que nos permite realizar Edge Impulse mediante el escaneo de un código QR. Seguidamente, se presentará la propuesta con el hardware (Raspberry Pi con diferentes periféricos) y se expondrá su funcionamiento detalladamente. Cabe destacar que esta implementación utiliza fotos tomadas con otro dispositivo para medir su funcionamiento, ya que debido a la calidad de la cámara de la Raspberry no se puede hacer una correcta detección de los frutos.

En el capítulo 6 se analizarán los resultados, realizando una comparación y un análisis de las diferentes pruebas realizadas con el modelo utilizando un dispositivo móvil. Se testeará y analizará el modelo con imágenes que presenten diferentes cantidades de frutos y distintos grados de madurez.

En el capítulo 7 se expondrán las conclusiones. En este apartado se expondrá cómo se ha llegado al objetivo final, las dificultades encontradas y una explicación de las diferentes posibilidades de aplicaciones futuras.

En el último capítulo se mencionarán aquellos ODS (Objetivos de Desarrollo Sostenible) que se pretenden abordar con la realización de este trabajo.

1.3-. Diagrama de Gantt

En este apartado se muestra el diagrama de Gantt, donde está expresa la planificación temporal del trabajo.

ILUSTRACIÓN 1 - DIAGRAMA DE GANTT

Como se indica claramente en este diagrama, la realización de la memoria se ha hecho en paralelo con el desarrollo del proyecto. Además, el testeo del modelo se ha realizado junto a la implementación y las pruebas.

2-. Estado del arte

Para la realización de este trabajo, en primer lugar, se procede a analizar y estudiar diferentes artículos para conocer el estado del arte, con lo objetivo de profundizar en el tema y aprender sobre las tecnologías más usadas actualmente para solucionar dicho problema.

2.1-. Artículos analizados

El primer artículo analizado [3] explica el problema en las grandes explotaciones de cítricos para hacer un conteo previo de la producción. Hace unos años con la tecnología que había resultaba complicado hacer una estimación de la producción de forma adecuada y precisa.

En este proyecto se utiliza un dron con una cámara de muy alta resolución. Con él, lo que se pretende es poder recorrer la explotación rápidamente para realizar el conteo de los frutos. A este se le añadió una red neuronal creada con Keras y TensorFlow y se utilizó un dataset etiquetado para la detección de los frutos.

En el análisis de los resultados se obtiene que el error del sistema es del 1.54%. También se manifiesta la dificultad de reconocer los frutos, ya que estos pueden estar ocultos por otros o tener un color diferente a causa de su grado de madurez. El sistema creado arroja mejor precisión que otros métodos utilizados anteriormente, como, por ejemplo, la estimación realizada por un técnico profesional.

El objetivo del siguiente trabajo leído [4] es desarrollar un modelo de deep learning para discriminar los frutos del granado y exponer un algoritmo para estimar el tamaño del fruto en píxeles. De manera que el fruto estaría monitorizado durante todo el día y se podría saber el peso del mismo, con lo cual, se podría calcular la producción en quilogramos.

Para este trabajo se utilizó el lenguaje de programación Python, con el entorno de Spyder. Y se utilizó un modelo etiquetado, al igual que en artículo comentado anteriormente. Para el entrenamiento se utilizó la técnica de aumentado de datos que, con pequeñas rotaciones o transformaciones del dataset, se consigue aumentar el mismo.

En el análisis de resultados, el fondo se cambió a blanco y negro, dejando los frutos destacados del color original. Los resultados obtenidos fueron óptimos, aunque se destaca que hay confusión en la predicción si dos frutos se solapan entre sí.

La siguiente tesis [5] trata de cómo reconocer en un durazno los frutos que estén maduros, inmaduros o dañados.

Para su realización, se utilizó Anaconda con JupyterLab junto al lenguaje Python. Para la red neuronal se utilizó Keras. Primeramente, se realizó la obtención del dataset de forma manual, obteniendo una cantidad de más de mil doscientas muestras, tanto de frutos maduros, como inmaduros o dañados. Para la clasificación se utilizó una red neuronal convolucional.

Una vez entrenado el modelo se obtuvo la matriz de confusión de los duraznos y, se obtuvo una precisión superior al 82%. Se propone utilizar este sistema para la recolección automática de frutos.

Seguidamente, en esta publicación de la Universitat de Lleida [6] se habla de la dificultad de realizar el conteo de las frutas en una explotación agrícola debido a las oclusiones de las mismas o a las diferentes condiciones de luminosidad en la zona a analizar. En esta publicación se utiliza un LiDAR 3D que permite, gracias a la intensidad de luz reflejada, distinguir entre los frutos y otras partes del árbol.

Por otra parte, también se utilizan sensores de RGB-D que proporcionan datos simultáneamente de color, profundidad y señal infrarroja. Esta información se utiliza para entrenar las redes neuronales.

Para la estimación de la cosecha, según el artículo, es necesario que el error de muestreo no supere el 10%. Esto es debido a que el conteo de las frutas es el parámetro que se utiliza para calcular la producción habitualmente. Para la correcta estimación de la cosecha, no es suficiente con una pequeña muestra de árboles, sino que se deberá realizar el conteo para cada uno de los ejemplares de la explotación. Por lo que se propone la utilización de este tipo de sensores junto a avionetas o drones.

2.2-. Análisis

Después del análisis de los diferentes artículos se puede intuir que realizar una estimación de la producción en quilogramos de una explotación de árboles frutales es una tarea difícil. Esto se debe a que esta estimación depende del conteo de los frutos realizado con anterioridad.

Para realizar dicho conteo con precisión se deben tomar datos de cada uno de los ejemplares. Aun así, puede haber frutos que se encuentran ocultos por otras partes del árbol. También, los frutos pueden ser confundidos en el momento de la detección con otros elementos del árbol.

Atendiendo a esta información, lo que se pretende es realizar y optimizar el modelo de la mejor forma posible, para poder detectar la mayor parte de las frutas del árbol.

En el siguiente apartado se detallará detenidamente todo el proceso para la realización del proyecto, teniendo en cuenta toda la información obtenida en los artículos anteriores.

3-. Propuesta del proyecto

Una vez analizado el estado del arte, se procede a explicar detalladamente la propuesta del trabajo y como se va a proceder a la realización del mismo.

3.1-. Líneas de investigación

Como se ha mencionado en el apartado anterior, realizar una detección de frutos en un árbol frutal es una tarea bastante tediosa. Para realizar dicha tarea se tienen que tener en cuenta diferentes parámetros como, por ejemplo: el tamaño de la fruta, el contraste de color del fruto con las diferentes partes del árbol o la cantidad de frutas, entre otros.

Para esto se van a realizar dos implementaciones diferentes, una utilizando un dispositivo móvil, y otra una Raspberry Pi. Además, se van a entrenar y probar diferentes modelos hasta quedarse con el más óptimo. Después de entrenar los diferentes modelos, se probará el mejor y se realizará un testeo para comprobar su funcionamiento.

Antes de la realización de este entrenamiento se va a obtener un dataset de fotos de melocotoneros con frutos, que estará compuesto por imágenes tomadas por mí mayoritariamente y algunas imágenes obtenidas de Internet. Este dataset estará compuesto también por imágenes de melocotones con un diferente grado de madurez. Una vez creado el dataset se etiquetarán las fotos para realizar el entrenamiento utilizando Edge Impulse.

Seguidamente, una vez creada la colección de imágenes y entrenados los diferentes modelos, se realizará una pequeña implementación en un dispositivo móvil y, posteriormente, en una Raspberry Pi, como propuesta de implementación. Además, se comparará con otras posibles implementaciones ficticias del modelo para que el sistema sea aplicable a diferentes ámbitos y a distintos árboles.

Finalmente, se desea comparar diferentes parámetros de los diferentes modelos, como, por ejemplo, la capacidad del sistema para reconocer los melocotones de los árboles, el tiempo que tarda en detectarlos, entre otros. También se pretende analizar la capacidad de la plataforma Edge Impulse para entrenar modelos en su versión gratuita.

3.2-. Árbol a analizar

El primer paso antes de empezar a recolectar el dataset es elegir el tipo de árbol frutal con el que se van a realizar las pruebas.

Como se ha mencionado anteriormente, para reconocer una fruta es necesario que haya un buen contraste con las otras partes del árbol, por lo tanto, se ha elegido el melocotonero, ya que por fuera los frutos son de tonalidades amarillas y rojas. Aparte de esta característica, es una fruta que se recolecta durante los primeros meses del verano, por lo que se van a poder hacer pruebas y tomar fotos sobre los mismos, con el objetivo de realizar la validación del sistema y elaborar un dataset extenso y hecho a medida para este proyecto.

El melocotonero (*Prunus Persica*) [7] es una especie de árbol del género *Prunus* de la familia *Rosaceae.* Su fruto es conocido como melocotón, aunque en algunos países de le llama durazno o piesco. Los principales productores de este fruto son por orden decreciente: China, España e Italia.

Su fruto, el melocotón, contiene su semilla encerrada en una cáscara dura, llamada hueso. Habitualmente tiene la piel aterciopelada y posee una carne amarilla de sabor dulce.

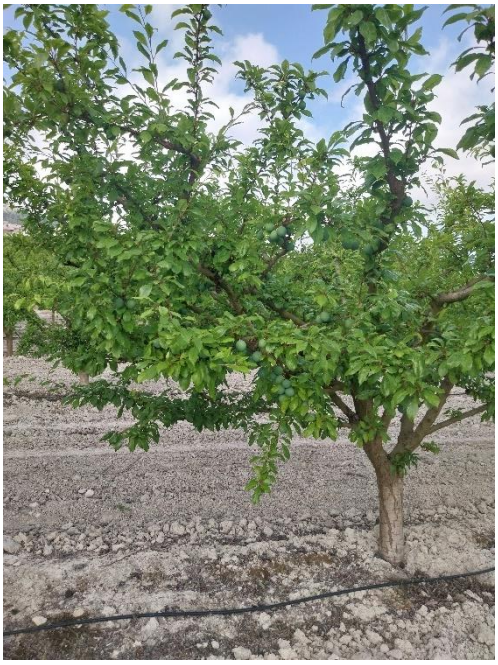

ILUSTRACIÓN 2 - MELOCOTONERO CON FRUTOS VERDES

3.3-. Edge Impulse

Edge Impulse es una plataforma que sirve para realizar algoritmos de aprendizaje automático para su posterior implementación en dispositivos como microcontroladores u ordenadores con recursos reducidos.

A diferencia de otras plataformas como Tensor Flow, en Edge Impulse no es necesario involucrarse en el código, aunque existe la opción de modificar el código para optimizar y personalizar el modelo con más profundidad. Mediante su interfaz gráfica se pueden entrenar, crear y ajustar los modelos. Cabe destacar que es necesario tener conocimiento sobre este tipo de algoritmos para su correcta implementación y funcionamiento.

Edge Impulse también destaca por su gran compatibilidad, ya que permite embeber el modelo en tu plataforma final (dispositivo móvil, Raspberry Pi o Arduino).

Esta plataforma está en constante crecimiento, por lo que resultará útil para la realización de este proyecto. Así pues, destacar que se va a explotar su versión gratuita para ver las prestaciones que ofrece.

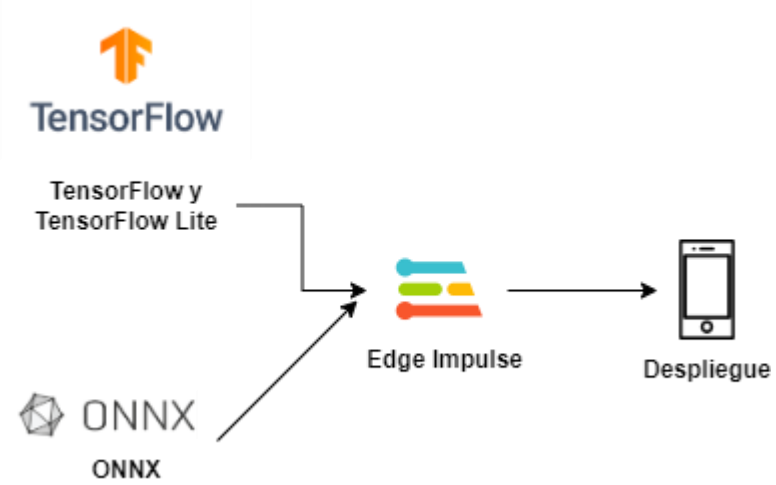

ILUSTRACIÓN 3 - ESQUEMA DE EDGE IMPULSE

En la ilustración anterior se muestra un pequeño diagrama de flujo que muestra como Edge Impulse utiliza Tensor Flow y ONNX como entrada para su kit de desarrollo (SDK). Posteriormente, se puede ver que Edge Impulse ofrece el modelo entrenado para su despliegue.

4-. Aprendizaje automático

El aprendizaje automático permite que un sistema aprenda y mejore de forma autónoma mediante redes neuronales y deep learning. Se va a explicar detalladamente qué es y todo el proceso a seguir hasta obtener un modelo de reconocimiento totalmente funcional y aplicable.

4.1-. Aprendizaje automático supervisado

El aprendizaje automático supervisado [8] se engloba en una de las subcategorías de la inteligencia artificial y machine learning. Su principal característica es la utilización de datos etiquetados que son utilizados para el entrenamiento de algoritmos que se encargan de clasificar datos con precisión.

Un dato etiquetado es aquel que se adjunta con una etiqueta que indica claramente de que elemento que se trata. Por ejemplo, en nuestro caso, los melocotones de las imágenes irán encuadrados junto a la etiqueta de “fruta”.

Existen principalmente dos tipos de problemas que se pueden solucionar con este tipo de sistemas:

- **Problemas de clasificación:** son los encargados de detectar anomalías en una imagen, detectar objetos o clasificar muestras en diferentes clases. Es el que se va a utilizar en la realización de este proyecto.
- **Problemas de regresión:** la salida son vectores continuos, es decir, son capaces de predecir, por ejemplo, el precio de una casa basándose en diferentes elementos de la misma.

Algunos de los algoritmos más habituales son:

- **Árboles de decisión:** se encargan de dividir las características en regiones para hacer las predicciones.
- **Máquinas de soporte vectorial (SVM):** encuentran el hiperplano que mejor separa las clases en el espacio de características.
- **Redes neuronales:** son programas o modelos que toman decisiones de forma similar al cerebro humano. Son las que se van a utilizar en la realización de este proyecto.

4.2-. Elaboración del dataset

En la creación del modelo de reconocimiento de frutos, es necesario, en primera instancia, la recolección de un dataset de imágenes. En este caso, se ha creado una colección de imágenes obtenidas mayoritariamente mediante una cámara, y además, otras pocas imágenes de internet.

En total, el conjunto de datos está formado por 213 muestras, teniendo en cuenta el conjunto de entrenamiento y test.

Las muestras de este dataset se han obtenido mediante la cámara de la Samsung Galaxy Tab8+, tomando fotos de diferentes ejemplares.

Las fotografías se han tomado durante diferentes días, con diferente clima y condiciones de luminosidad. Esto se debe a que el sistema debe ser capaz de reconocer las frutas en cualquier condición climática, ya sea lluvia o sol.

También se han realizado fotografías a frutos con diferentes estados de maduración, ya que se encontró, en el mismo campo, frutos mucho más maduros que otros. Con esto, en el momento de utilizar el modelo, se podrán detectar todas las frutas independientemente de su estado de maduración.

Cuando ya se tiene el conjunto de imágenes que se va a utilizar, se deben subir a la plataforma de Edge Impulse. Las imágenes deben ser representativas, es decir, el fruto se debe de distinguir de forma adecuada de los otros elementos a descartar (ramas u hojas). Además, es interesante que los frutos se encuentren separados, ya que el algoritmo de reconocimiento va a reconocer mejor las características.

En la siguiente imagen se presenta una muestra del dataset.

ILUSTRACIÓN 4 - MUESTRA DEL DATASET

En el caso de esta muestra se observa que únicamente hay un melocotón. Aunque el dataset está compuesto por imágenes con una cantidad variable de frutos.

4.3-. División del dataset

Una vez obtenido el dataset y realizados los pasos descritos anteriormente, se debe realizar una correcta división del mismo. La división realizada para el aprendizaje automático es la siguiente:

- **Conjunto de entrenamiento:** este conjunto sirve para que el modelo pueda aprender sobre los datos etiquetados, con el propósito de ajustar los parámetros. En el caso de este proyecto se utiliza un 82% del conjunto de las muestras iniciales.
- **Conjunto de validación:** se realiza una evaluación no sesgada para ajustar el modelo con sus hiperparámetros. Es como una evaluación intermedia del proceso de entrenamiento. Del 82% de los datos utilizados para el entrenamiento, un 20% se utiliza para el set de validación.
- **Conjunto de test:** en esta parte se utilizan los datos etiquetados anteriormente. El algoritmo debe encargarse de identificar o detectar las frutas sin ver las etiquetas. Posteriormente, se observan y se comparan las etiquetas predichas por el sistema y las etiquetadas manualmente. Atendiendo a los aciertos, se obtienen las métricas del conjunto del test. En este caso concreto se realiza con un 18% de los datos de entrada.

En la siguiente ilustración se muestran las imágenes utilizadas para la creación del modelo, al igual que su split entre entrenamiento y test.

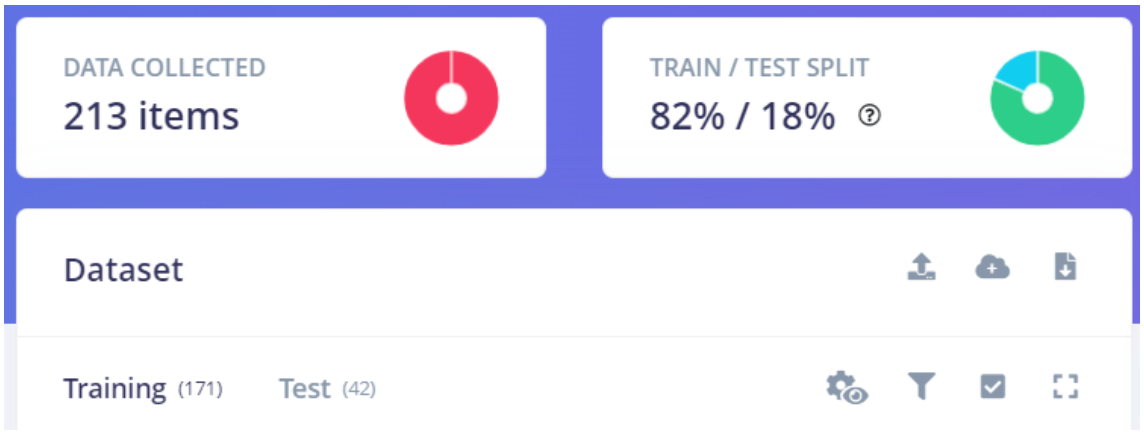

ILUSTRACIÓN 5 - DIVISIÓN DE ENTRENAMIENTO Y TEST

4.4-. Etiquetado del dataset

El etiquetado es el proceso previo que se realiza antes del entrenamiento del modelo. Este proceso consiste en ir imagen a imagen, encuadrando el objeto de interés, en este caso en concreto la fruta, y poniendo el nombre de la etiqueta (fruta).

Este proceso de etiquetado es un componente básico en los modelos de visión por computador, ya que estas etiquetas servirán para que el modelo de aprendizaje automático reconozca y aprenda que objeto es, en este caso preciso, si se trata de una fruta o se trata de otros componentes del árbol.

El etiquetado de un dataset es largo y tedioso y se debe realizar de forma manual y minuciosa para que el modelo aprenda únicamente aquellas características que son necesarias. En ocasiones, si el resultado del test no es el esperado, se debe revisar el etiquetado para comprobar que se ha hecho de forma correcta.

El etiquetado se hace sobre todas las imágenes, independientemente de si son datos de entrenamiento, validación o test.

En la plataforma de Edge Impulse se ofrece una herramienta para realizar el etiquetado manualmente, mediante el proceso de encuadrado mencionado anteriormente.

A continuación, se muestra un ejemplo de una imagen etiquetada utilizando la herramienta de etiquetado de Edge Impulse.

ILUSTRACIÓN 6 - EJEMPLO ETIQUETADO DE IMAGEN

Como se observa en la imagen, el etiquetado se ha hecho sobre frutos sin ningún tipo de oclusión, ya que así, como se ha mencionado con anterioridad, el algoritmo puede aprender mejor las características de la fruta.

Cabe destacar que, en ocasiones, una pequeña oclusión puede ser interesante, ya que si en la implementación real aparece algo que oculta una parte de la fruta, puede no detectarla de forma adecuada.

El etiquetado se ha realizado sobre 213 muestras, es decir, todas las muestras que hay en el conjunto de datos que utiliza el modelo.

4.5-. Entrenamiento

Después de realizar el etiquetado de las imágenes correctamente, se procede a realizar diferentes entrenamientos para la obtención del modelo más óptimo.

Como entrada de la red neuronal, están las imágenes etiquetadas siguiendo los pasos explicados anteriormente. En la salida, únicamente se tiene una posible etiqueta, que es “fruta”. El esquema que se tiene en Edge Impulse, cuenta con una imagen de entrada que se preprocesa, en este caso se encuadra y se redimensiona.

El esquema a priori es el que se muestra en la siguiente ilustración, aunque se utiliza una red neuronal convolucional, optimizada para la detección de las frutas.

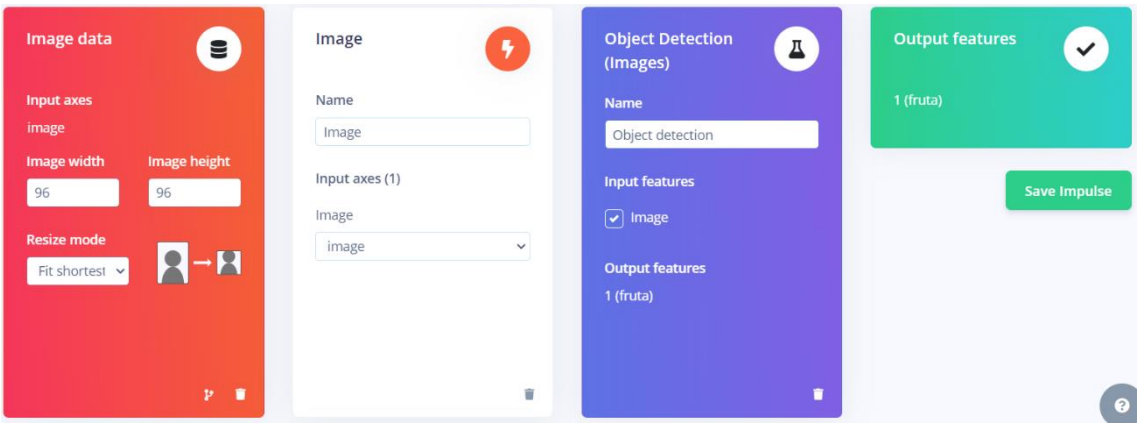

ILUSTRACIÓN 7 - ESQUEMA DE EDGE IMPULSE PARA EL ENTRENAMIENTO

Seguidamente, se procede a personalizar los parámetros e hiperparámetros del modelo. Estos son los siguientes:

- **Number of cycles:** también conocido como epoch. Se refiere al número de iteraciones completas que se realizan sobre el conjunto de datos de entrenamiento. Cabe destacar que, a mayor valor en este campo, mayor probabilidad de sobreentrenar el modelo. En este caso se ha elegido 170 epoch. Este valor se debe combinar adecuadamente con el learning rate.

- **Learning rate:** es uno de los hiperparámetros más importantes. Hace referencia a como de rápido la red neuronal aprende, es decir, cuánto se ajustan los pesos del modelo en cada iteración. Su objetivo es minimizar la función de pérdida, es decir, minimizar la diferencia entre las predicciones del modelo y las etiquetas reales. Si la red neuronal se sobreentrena rápidamente, suele ser necesario disminuir este valor. El valor utilizado es 0.0015.

- **Training processor:** por defecto, en la versión gratuita de Edge Impulse es la CPU. Si se desea utilizar una GPU se debe hacer uso de la versión de pago.

- **Data augmentation:** este campo se puede seleccionar o no. En este caso concreto ha sido marcada la casilla. Con esta chequeada, se aumenta el tamaño y la diversidad del conjunto de datos de entrada. Esto se obtiene realizando diferentes transformaciones como rotaciones o escalados. Ayuda a generalizar el modelo y reducir el sobreajuste durante el entrenamiento.

- **Validation set size:** se puede personalizar y se trata del tamaño del set de validación. Como se ha mencionado con anterioridad, el set de validación es del 20% sobre el conjunto de datos de entrenamiento.

- **Neural network architecture:** se ha utilizado FOMO MobileNewV2 0.35. Está diseñada para la detección rápida y eficiente de elementos en una imagen, utilizando dispositivos con recursos limitados o hardware de poca potencia, como puede ser una Raspberry Pi. Conjuntamente con MobileNetV2 [9], que es una red neuronal convolucional de 53 capas de profundidad, que permite mantener un alto rendimiento en tareas de visión por computado. Puede lograr clasificar 1000 categorías de elementos diferentes.

Para lograr un modelo correcto, y que no esté sesgado a los datos de entrenamiento, se deben probar diferentes combinaciones de parámetros e hiperparámetros hasta encontrar la más óptima. Además, es necesario saber cuál es el objetivo del modelo para poder elegir correctamente entre las redes neuronales preentrenadas que nos ofrece Edge Impulse.

Cabe destacar que los campos mencionados son aquellos que se pueden modificar con la versión gratuita de Edge Impulse. Además, se puede personalizar también el código de Keras manualmente para obtener una salida personalizada.

Algunas limitaciones de la versión gratuita de Edge Impulse son: el máximo tiempo de entrenamiento es de 20 minutos y solo se pueden utilizar 4Gb de datos de entrenamiento.

A continuación, se muestra un fragmento de código con todos los valores comentados en este apartado:

```
model = train(num_classes=classes,
              learning_rate=LEARNING_RATE,
              num_epochs=EPOCHS,
              alpha=0.35,
              object_weight=100,
              train_dataset=train_dataset,
              validation_dataset=validation_dataset,
              best_model_path=BEST_MODEL_PATH,
              input_shape=MODEL_INPUT_SHAPE,
              batch_size=BATCH_SIZE,
              use_velo=False,
              ensure_determinism=ensure_determinism)
```

ILUSTRACIÓN 8 - CÓDIGO PARA EL ENTRENAMIENTO

Como bien se observa en la imagen, los valores que se han mencionado anteriormente son aquellos que están reflejados en el código.

4.6-. Valores a analizar

Cuando se realiza el entrenamiento se deben tener en cuenta y entender varios parámetros. Estos son los siguientes:

1. **Precisión:** la precisión (fórmula 1) es la proporción de casos que son verdaderos positivos, sobre la cantidad total de todo lo que se estableció como positivo.

$$1)\ \text{Precisión} = \frac{\text{Verdaderos Positivos (TP)}}{\text{Verdaderos Positivos (TP)} + \text{Falsos Positivos (FP)}}$$

2. **Recall:** el recall, representado en la fórmula 2, es la proporción de verdaderos positivos entre el total de todo aquello que era positivo.

$$2)\ \text{Recall} = \frac{\text{Verdaderos Positivos (TP)}}{\text{Verdaderos Positivos (TP)} + \text{Falsos Negativos (FN)}}$$

3. **F1-Score:** el F1-score es la media armónica entre la precisión y el recall, fórmula 3.

$$3)\ \mathrm{F1-Score} = \frac{2 * \mathrm{Precisión} * \mathrm{Recall}}{\mathrm{Precisión} + \mathrm{Recall}}$$

En la siguiente tabla se muestra la relación entre las etiquetas reales (etiquetadas a mano) y las etiquetas que predice el sistema. También es conocida como la matriz de confusión del modelo. Esta matriz nos ayuda a identificar donde se encuentran los errores y aciertos del sistema, para poder así corregir los parámetros del modelo o el etiquetado del dataset.

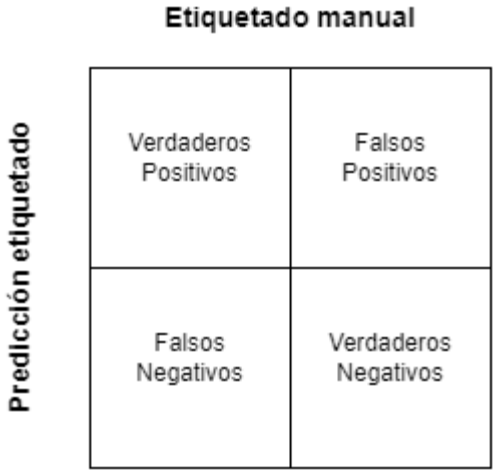

ILUSTRACIÓN 9 - MATRIZ DE CONFUSIÓN GENÉRICA

Es importante lograr un equilibrio entre las métricas obtenidas en el entrenamiento y las métricas obtenidas en el test. En el caso de obtener un valor de F1-score muy alto en el entrenamiento y el del test muy bajo, nos indicaría que el modelo ha sido sobreentrenado y que está sesgado a los datos de entrenamiento, por lo que se deberían modificar los parámetros e hiperparámetros mencionados en el apartado anterior, con el objetivo de corregir este problema.

4.7-. Valores obtenidos en el conjunto de entrenamiento

Para obtener un entrenamiento óptimo, como se ha mencionado anteriormente, se deben probar diferentes combinaciones de parámetros e hiperparámetros. En este caso, después de realizar diferentes entrenamientos se ha conseguido entrenar un modelo con un funcionamiento correcto.

Con este modelo se va a proceder a realizar diferentes pruebas para analizarlo detalladamente.

Los valores obtenidos en el entrenamiento del modelo se muestran en la siguiente imagen:

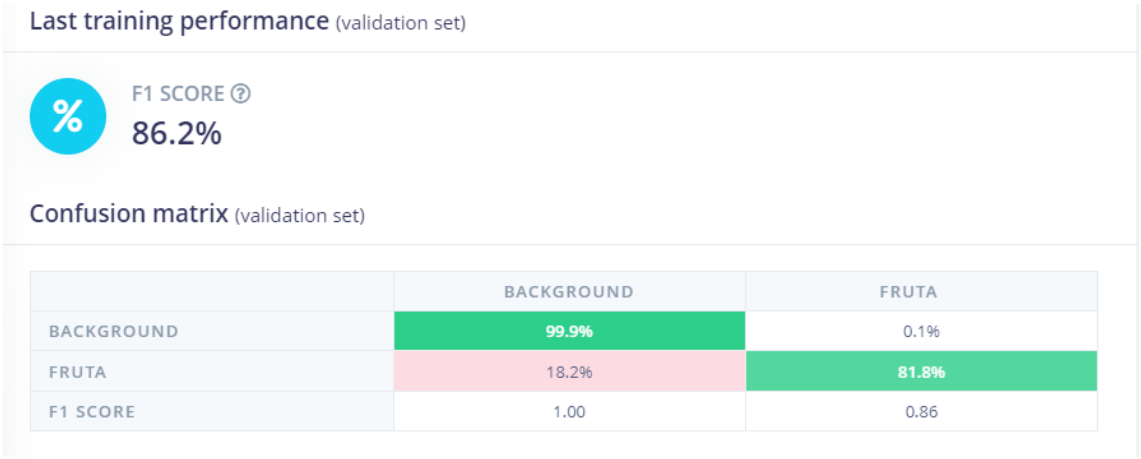

ILUSTRACIÓN 10 - MATRIZ DE CONFUSIÓN DEL ENTRENAMIENTO

Como se observa, el valor de F1-score es bastante elevado. En la matriz de confusión se observa que el fondo siempre lo detecta correctamente. En el caso de las frutas, detecta el 81,8% correctamente.

La confusión puede ser debida al etiquetado, ya que la mayor parte de las frutas que están ocultas no se han etiquetado como fruta. En el caso de etiquetar las frutas mayormente ocultas, puede influir en el entrenamiento y detectar otros elementos del árbol como fruta. Estos errores se van a analizar en los siguientes apartados para comprobar la causa del fallo del modelo.

4.8-. Valores obtenidos en el conjunto de test

Una vez realizado el entrenamiento, es necesario testear el modelo para que este no esté sesgado al conjunto de entrenamiento. Cuando esto ocurre, se dice que el modelo está sobreentrenado.

Durante la realización de este proyecto se han creado diferentes modelos que, en el momento de testear, el valor de F1-score era relativamente inferior al obtenido con los datos de entrenamiento, por lo que se tenían que modificar los parámetros para evitar el sobreentrenamiento.

En el caso del modelo analizado anteriormente, el valor de F1-score difiere de forma muy sutil al obtenido en el conjunto de entrenamiento, por lo que el modelo no está sesgado.

A continuación, se muestra un gráfico que indica las clasificaciones realizadas correctamente sobre el conjunto de test:

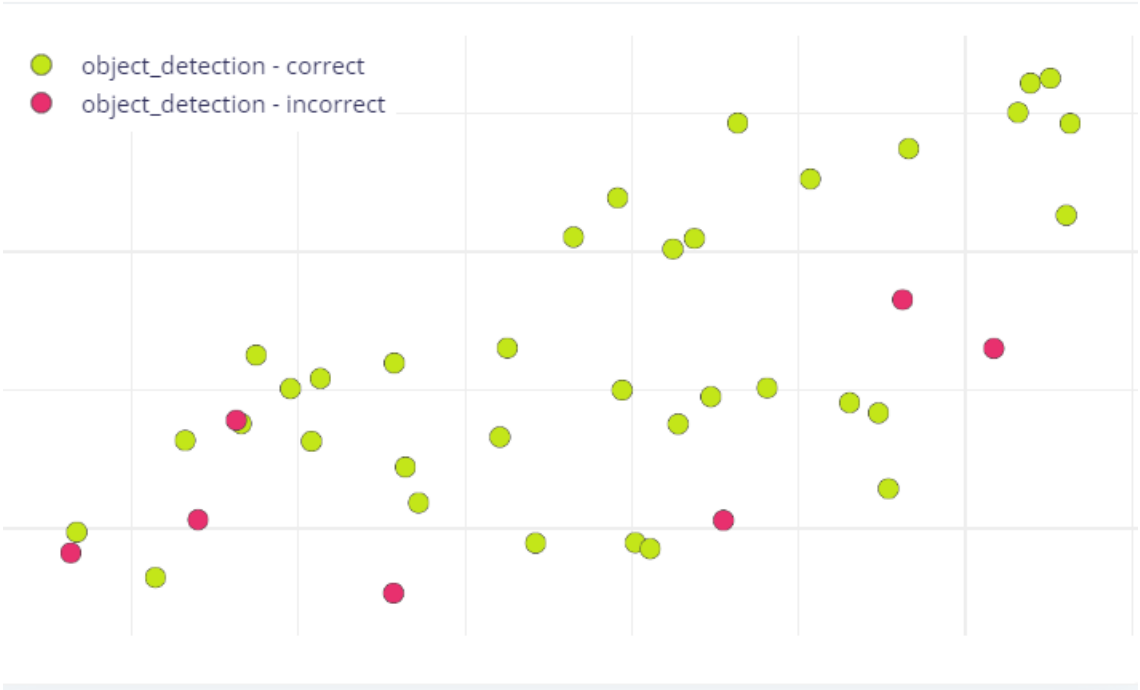

ILUSTRACIÓN 11 - DIAGRAMA DEL CONJUNTO DE TEST

Como se observa en este diagrama de puntos, cada punto representa una imagen del conjunto de test, la mayor parte de las muestras de test están identificadas correctamente (color verde). Posteriormente se realizará un estudio sobre las muestras que no están detectadas correctamente.

Las que están clasificadas como correctas es porque están etiquetadas correctamente más de un 80%. En el caso de las etiquetadas incorrectamente, no llegan a este valor.

Se muestran los valores obtenidos en el conjunto del test:

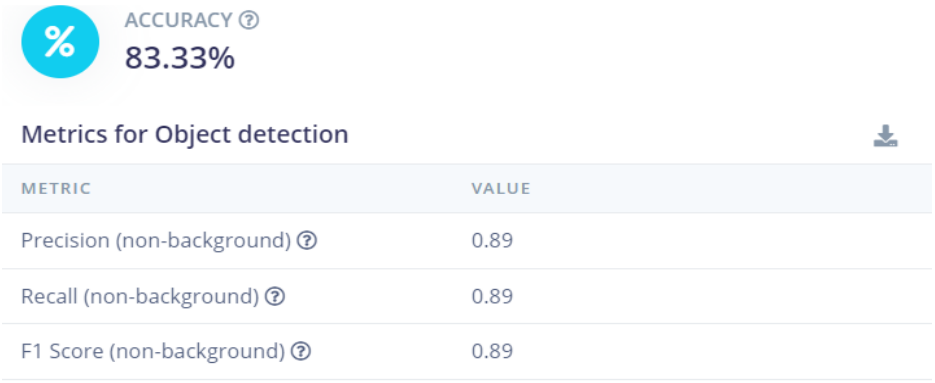

ILUSTRACIÓN 12 - VALORES DEL CONJUNTO DE TEST

Se procede a realizar un estudio sobre aquellas muestras del conjunto de test que no están etiquetadas correctamente.

Seguidamente, se muestra una imagen que ha sido detectada según el modelo con un 50% de precisión:

ILUSTRACIÓN 13 - EJEMPLO DE MUESTRA DEL TEST

En la ilustración anterior se observan dos imágenes. En primera instancia, la foto etiquetada a mano; en segunda instancia, la imagen sometida al modelo.

Como bien se observa en la primera imagen únicamente hay un fruto etiquetado, esto se debe a que el color del fruto no etiquetado es más amarillento y se puede confundir, además, también está bastante oculto. Cabe destacar que se han etiquetado imágenes con frutos poco maduros, pero en este caso, se ha creído conveniente no etiquetarlo debido a las sombras y las oclusiones.

En la segunda imagen, se observa que sí que es detectado por el modelo entrenado, además el valor que ofrece el modelo es de 1.00, por lo que está seguro de que es un fruto. Esto se debe a que el conjunto de imágenes utilizado para el entrenamiento es extenso y se han etiquetado melocotones de diferente tamaño y estado de maduración.

Por lo tanto, aunque esta muestra tiene una precisión del 50%, realmente sí que está bien etiquetada por el modelo.

Por todo esto, una vez se ha sometido el modelo al conjunto de test, se deben analizar minuciosamente los resultados. Aunque el valor de precisión puede ser inferior al deseado, el modelo puede funcionar adecuadamente, ya que en el conjunto de test simplemente se comparan las etiquetas predichas por el sistema con las que se han puesto manualmente.

5-. Implementación

En este punto se procede a estudiar y analizar las diferentes tecnologías con las que se proponen dos posibles implementaciones para llevar a la práctica el modelo entrenado con anterioridad y realizar diferentes pruebas para comprobar su correcto funcionamiento.

5.1-. Implementación con un dispositivo móvil

Edge Impulse ofrece una opción para poder testear los modelos entrenados en un dispositivo móvil. Este modelo que se ha entrenado y gracias a la utilización de FOMO MobileNewV2 0.35 explicada anteriormente, se puede utilizar escaneando simplemente un código QR y utilizando la cámara del mismo dispositivo.

El modelo está optimizado para su correcto funcionamiento a tiempo real. Una vez detectada una fruta, aparece junto a dicha etiqueta la probabilidad de que sea la misma.

La principal ventaja de esta implementación es que, si se tiene un dispositivo móvil de alta gama, se pueden obtener imágenes de buena calidad y realizar una detección minuciosa a tiempo real.

A continuación, se muestra el panel de Edge Impulse. Mediante el escaneo del QR se puede probar el modelo o añadir más imágenes al dataset (que posteriormente se deberían etiquetar).

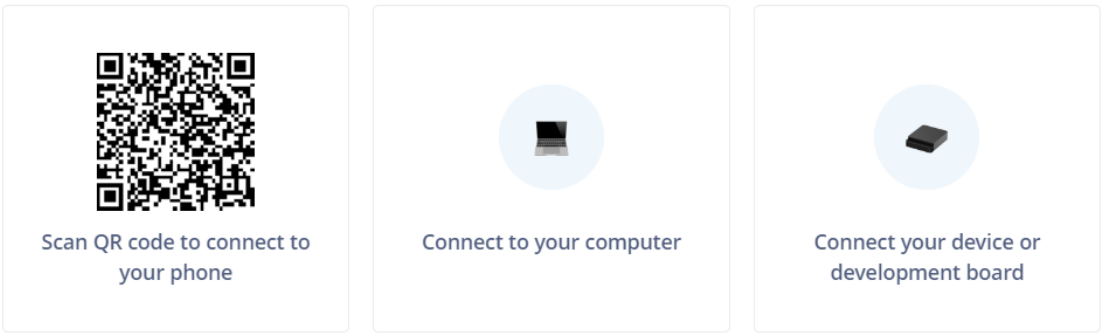

ILUSTRACIÓN 14 - OPCIONES DE EDGE IMPULSE PARA EL MODELO

Esta implementación puede resultar interesante para la aplicación que se comentó en la introducción. Simplemente, mientras se hacen las labores en el campo, se podría dejar la cámara del móvil encendida y mediante una capturadora de pantalla, grabar toda la detección de los frutos. Al final de la jornada, el agricultor tendría una grabación donde se le mostrarían todas las frutas detectadas.

La época de recolecta del melocotón y sus variedades son en los meses de verano principalmente (de finales mayo a julio). En esta época puede haber tormentas en forma de granizo o lluvias fuertes, por lo tanto, en caso de daños en la cosecha, se podría presentar el vídeo al seguro, como prueba de la cosecha que se tenía.

Para dicha implementación se ha usado una tableta de gama alta, en concreto la Samsung Galaxy Tab8+. Este dispositivo cuenta con altas prestaciones, entre las que destaca:

- Procesador de 2.99GHz Octa-Core.
- Cámara de 13MP, con vídeo UHD 4K.
- Batería de 10090 mAh.
- Resolución de pantalla UHD 8K (7680 x 4320).
- Conectividad Wi-Fi hasta 6GHz.
- Bluetooth v5.2.

Por todas estas características del dispositivo, se va a proceder a realizar la implementación y las pruebas con dicho dispositivo.

5.2-. Prototipo Raspberry Pi

5.2.1-. Presupuesto prototipo Raspberry

En la siguiente tabla se muestran los precios para la implementación del modelo de reconocimiento de frutos en la Raspberry Pi:

Producto	Cantidad	Precio Unidad	Total
Raspberry Pi 4 Model B, 8 Gb	1	89,95€	89,95€
Raspberry Pi Camera Module 2	1	27,95€	27,95€
Tarjeta microSD 64 Gb	1	24,95€	24,95€
Pantalla Raspberry LCD 3.5”	1	16,95€	16,95€
		Total:	159,80€

El coste de esta implementación depende de los componentes que se elijan. Para mejorar el funcionamiento se debería utilizar una cámara de mayor resolución, por lo que el coste aumentaría.

5.2.2-. Raspberry Pi 4 Model B y periféricos

Una Raspberry Pi [10] es un ordenador de bajo coste integrado en una placa desarrollada en Reino Unido por Raspberry Pi Fundation, cuya fundación fue en 2009. En el 2006, se elaboraron los primeros diseños basados en el microcontrolador Atmel ATmega644. Después de varios desarrollos, en febrero de 2012, se lanzó el primer modelo, la Raspberry Pi 1 Model B, que se vendió por 35$. Desde ese momento se han desarrollado y lanzado diferentes modelos. En este caso, se utilizará la Raspberry Pi 4 Model B, que es el último lanzamiento de esta marca.

Este modelo cuenta con las siguientes características técnicas:

- Procesador Broadcom de 64 bits a 1,5 GHz.
- Memoria RAM de 8GB.
- 2 USB 3.0 y 2 USB 2.0.
- Conexión Gigabit Ehernet.
- Soporte de tarjeta SD (micro SD).
- 2 puertos micro HDMI (hasta 4Kp60 admitidos).

Está opción resulta interesante, ya que debido a su alta capacidad de procesamiento, puede contener y ejecutar con una velocidad adecuada modelos de redes neuronales.

ILUSTRACIÓN 15 - RASPBERRY PI 4

Únicamente con este dispositivo se puede probar el modelo para analizar sus prestaciones, pero no se pueden tomar imágenes ni ver los resultados.

Por esto, es necesario una cámara para su funcionamiento de forma autónoma. Con ella se puede tomar fotos o un videos a tiempo real y someter dichas imágenes al modelo para la detección de frutos.

Para lograr una independencia total, se debe introducir una pantalla con la cual se van a poder ver los resultados. En caso de no disponer de ella, no se podrían ver los resultados.

5.2.3-. Diagrama y funcionamiento

A continuación, se muestra el diagrama para la implementación en la Raspberry Pi:

ILUSTRACIÓN 16 - DIAGRAMA DE LA IMPLEMENTACIÓN EN LA RASPBERRY

Como se puede observar en el diagrama, la Raspberry Pi lleva embebido el modelo creado anteriormente, por lo tanto, será la encargada de procesar las imágenes. Estas serán obtenidas utilizando la cámara incorporada a la Raspberry Pi. Para ver los resultados de la detección se dispone de una pantalla táctil LCD de 3’5 pulgadas.

Una vez entendido el diagrama se procede a la implementación. El primer paso es montar a la Raspberry Pi los componentes. Seguidamente, se instalará el sistema operativo de la Raspberry Pi mediante una tarjeta micro SD y se procederá a la configuración de la cámara y de la pantalla.

Edge Impulse ofrece una opción para poder conectar directamente la plataforma a la Raspberry Pi mediante comandos. El principal problema de esta implementación es la resolución de la cámara de la Raspberry Pi, ya que para que el sistema funcione correctamente es necesario que las imágenes captadas por la cámara sean de alta calidad.

Testear el modelo con la Raspberry Pi resulta muy complicado, tal y como se ha indicado, por la resolución de la cámara. Esta cámara puede resultar interesante para modelos de reconocimiento de elementos más grandes como, por ejemplo, coches. En este caso, debido a su poca resolución, no es posible probar correctamente el modelo con las fotos que realiza. Por lo tanto, se ha testeado el modelo mediante imágenes tomadas en otro dispositivo, pero procesadas con la Raspberry Pi.

El resultado de la implementación es el que se muestra a continuación. Así pues, cabe destacar que la cámara va en la parte trasera de la Raspberry, pero para mostrar todos los componentes queda de la siguiente forma:

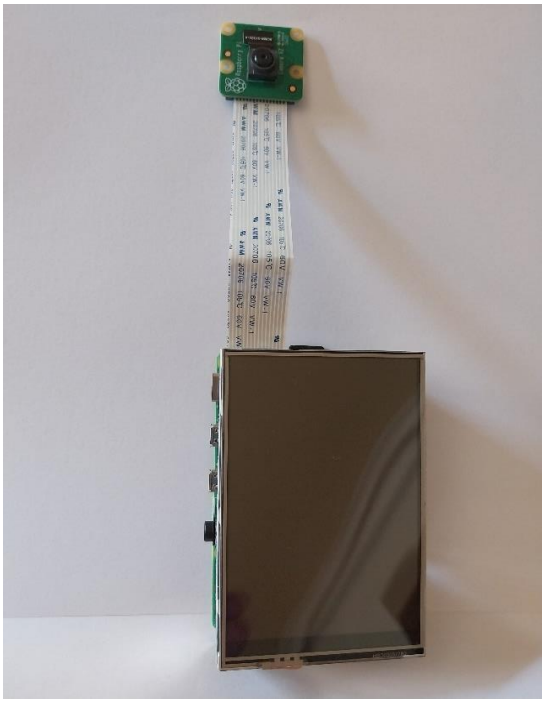

ILUSTRACIÓN 17 - PROTOTIPO MONTADO

5.2.4-. Pruebas realizadas con la Raspberry Pi

Para comprobar el funcionamiento y el tiempo de procesamiento del modelo se cargaron unas imágenes de prueba (tomadas con una cámara de alta resolución) en la Raspberry Pi y se procedió a analizar los tiempos de detección del modelo.

ILUSTRACIÓN 18 - TIEMPOS DE EJECUCIÓN DEL MODELO EN LA RASPBERRY

En la ilustración anterior se muestra que el tiempo de inferencia en el dispositivo es de unos 3 ms. Además, utiliza poca memoria RAM. Estos parámetros se deben a que el modelo de Raspberry Pi utilizado es de los más punteros del mercado y tiene unas altas prestaciones. Además, el modelo entrenado en Edge Impulse está optimizado de forma correcta y exportado con Tensor Flow Lite para su correcto funcionamiento en este tipo de dispositivos.

Atendiendo a esta información, se ha logrado obtener un modelo de reconocimiento optimizado para su correcto funcionamiento en dispositivos con pocos recursos como la Raspberry Pi 4.

6-. Análisis de resultados

Una vez realizado el modelo se procede a analizar los resultados y el funcionamiento del mismo.

Como se ha mencionado anteriormente, las pruebas de detección en la Raspberry Pi no han resultado exitosas debido a la calidad de la cámara. Por lo tanto, no resulta interesante probar el modelo con este prototipo. Para la realización de dichas pruebas se va a utilizar un dispositivo móvil con una cámara de mayor resolución, en este caso la cámara de una Samsung Galaxy Tab 8+.

En los apartados anteriores se han mostrado los valores de precisión y f1-score del modelo, entre otros. Seguidamente, se van a realizar pruebas de detección de frutos para comprobar si dichos valores coinciden con la realidad.

6.1-. Pruebas realizadas

En este caso se va a proceder a realizar diferentes tomas de imágenes para comprobar el funcionamiento de la detección de los frutos. Aunque el modelo esté validado por el conjunto de test, resulta interesante hacer diferentes pruebas para comprobar el correcto funcionamiento.

Dichas pruebas se van a realizar utilizando el dispositivo móvil de gama alta mencionado anteriormente. Se desean tomar diferentes imágenes a tiempo real y comparar la detección realizada por el modelo con la cantidad de frutas que hay realmente. También, se va a analizar cómo afectan las oclusiones al sistema, al igual que el grado de maduración o la cantidad de frutos por rama.

Las imágenes se muestran pixeladas debido al preprocesado que realiza previamente el sistema antes de analizar las imágenes. Este preprocesado consiste en un encuadrado y un redimensionado y es el primer paso que realiza el modelo.

Cuando el sistema detecta una fruta, marca su centroide con un círculo y añade la etiqueta de “fruta” junto a la probabilidad de que sea una fruta.

Para realizar dichas pruebas y, como se ha mencionado anteriormente, se ha utilizado la cámara de un dispositivo de alta gama.

En primera instancia se escanea el código QR como se muestra en la ilustración.

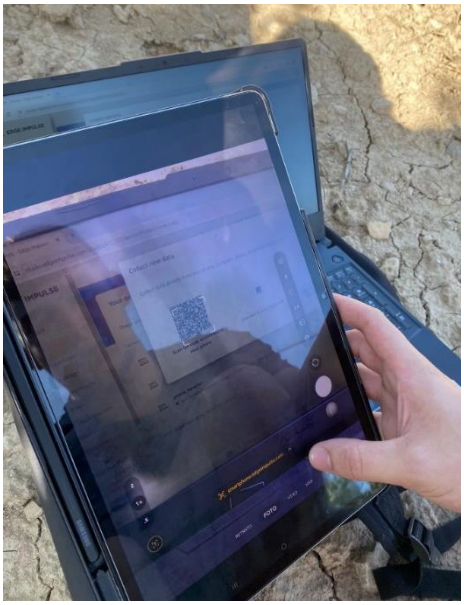

ILUSTRACIÓN 19 - ESCANEO DEL CÓDIGO QR

Seguidamente, se muestra cómo se han ido tomando las imágenes para realizar las pruebas del modelo.

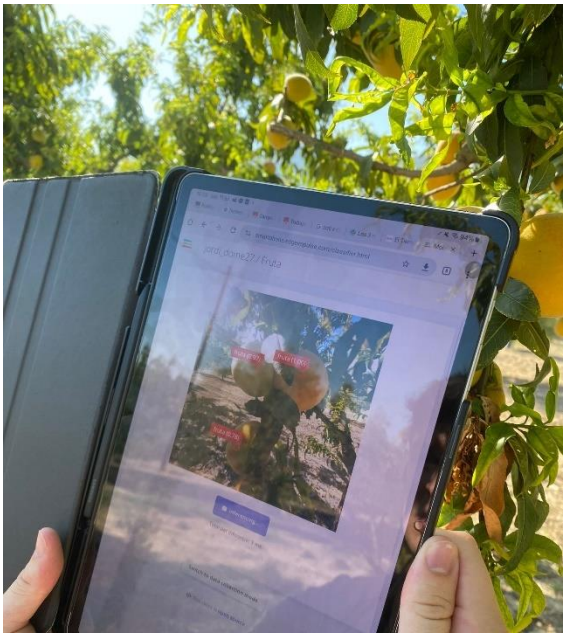

ILUSTRACIÓN 20 - MUESTRA DE LA TOMA DE IMÁGENES

Se van a mostrar diferentes ejemplos para testear y probar el funcionamiento del modelo.

El primer ejemplo se hace con frutos dispersos y maduros:

ILUSTRACIÓN 21 - MUESTRA DE MELOCOTONES MADUROS

ILUSTRACIÓN 22 - MUESTRA ETIQUETADA DE MELOCOTONES MADUROS

Como primera prueba se muestra una imagen con cuatro frutos dispersos, totalmente visibles o con algún tipo de oclusión. Los frutos ya están maduros y pueden ser recolectados en cualquier momento.

En la primera imagen se observa claramente que hay 4 frutos. Uno en primer plano que no presenta ningún tipo de oclusión. Hay 2 melocotones más en la imagen que sí presentan algún tipo de oclusión. Finalmente, en la esquina inferior derecha, se muestra un fruto pequeño que está cortado por el borde de la imagen.

En la segunda imagen, se observa que se detectan todos los frutos con un valor de probabilidad mayor o igual al 70%. Esto se debe a que, durante el etiquetado, se ha mostrado especial atención a posicionar las etiquetas de la forma más significativa para el sistema de reconocimiento. Por lo tanto, se logran detectar los frutos de forma correcta.

En el siguiente ejemplo se va a mostrar cómo se detectan los frutos con una madurez menor a la mostrada anteriormente:

ILUSTRACIÓN 23 - MUESTRA DE MELOCOTONES POCO MADUROS

ILUSTRACIÓN 24 - MUESTRA ETIQUETADA DE MELOCOTONES POCO MADUROS

La imagen mostrada contiene frutos poco maduros, es decir, con un color amarillento aún. Este color en principio puede resultar un problema, ya que se puede confundir el fruto con las hojas. Esta imagen resulta también interesante debido a las sombras de las mismas hojas del árbol. También hay un fruto que presenta una oclusión severa.

Se puede observar en la segunda imagen que la detección de los frutos es totalmente correcta. Los frutos más escondidos son detectados con total claridad por el sistema. En este caso, el menos visible, el sistema lo detecta con un 1, es decir, hay un 100% de probabilidades de que el fruto sea fruto.

Pese a tener sombras en la imagen y los melocotones no tener ese color naranja que les representa, el sistema detecta claramente los frutos. Como se mencionó en los apartados anteriores, el dataset está elaborado para poder detectar los frutos en cualquiera de sus estados. Por lo tanto, y como bien se ha observado en este ejemplo, el estado de madurez del melocotón no es un problema para que el sistema lo detecte como lo que es.

En el siguiente ejemplo se va a mostrar el principal problema que hay en la detección de este tipo de frutos.

ILUSTRACIÓN 25 - MUESTRA DE MELOCOTONES MADUROS AGRUPADOS

ILUSTRACIÓN 26 - MUESTRA ETIQUETADA DE MELOCOTONES MADUROS AGRUPADOS

Primeramente, en este ejemplo se muestra claramente el preprocesado que se hace de la imagen. En el caso de la detección de frutos a tiempo real, el preprocesado lo hace directamente la cámara, por lo que no se pierde información.

Seguidamente, se observa que cuando los frutos están muy cercanos, se muestra el centroide en el centro de la agrupación de los frutos, pero no se detectan todos. Esto se debe a que en los modelos FOMO, las etiquetas colapsan a los centroides, es decir, los píxeles no se tratan individualmente, sino que se consideran una representación centralizada (centroide) para cada una de las etiquetas. En este caso, como hay frutos que están muy cercanos entre sí, los centroides colapsan en uno, ya que el sistema no puede distinguir el uno del otro claramente.

Cabe destacar, que esta imagen está tomada en un melocotonero abandonado. Cuando este tipo de árbol está cuidado, para que el fruto llegue a tener el tamaño deseado, se realiza un proceso de aclarado, en el cual se dejan los frutos separados entre sí para que lleguen a conseguir el tamaño deseado.

Por todo esto, sí que resultaría un problema en la detección de los frutos en etapas primarias, cuando el árbol aún no ha sido aclarado. Pero, en etapas posteriores del melocotonero, serían detectados con total claridad.

Podría resultar una herramienta útil para ayudar al agricultor al aclarado del árbol. En el caso de no tener muy claro qué cantidad dejar, se podría apuntar con la cámara a dicha agrupación de frutos y dejar la cantidad que el sistema detecte. Es decir, si hay una agrupación de 6 frutos y se detectan dos centroides, se dejaría esa cantidad de frutos.

6.2-. Análisis de las pruebas realizadas

Como se ha observado en el apartado anterior, se pueden distinguir dos posibles casos con dos posibles variaciones:

- **Frutos poco maduros:** en este caso los frutos son de un tamaño pequeño y con un color amarillento o verdoso, se pueden confundir claramente con las hojas del árbol u otros elementos del mismo. Las variaciones que se presentan son las siguientes:
    - **Árbol aclarado:** Los frutos están bien distantes entre sí. La detección del fruto es buena debido a que los centroides de la detección de los frutos no colapsan.
    - **Árbol no aclarado:** los frutos se presentan en agrupaciones, por lo tanto, en la detección, los centroides se colapsan y se pierden detecciones. La detección no es buena.
- **Frutos maduros:** los frutos presentan un color anaranjado y un tamaño grande. En este caso es difícil que se confundan con otros elementos del árbol, ya que el color tiene un alto contraste. Puede haber dos variaciones posibles:
    - **Árbol aclarado:** los frutos están bien distantes entre sí, además destacan sobre los demás elementos. La detección es buena debido a las condiciones mencionadas anteriormente.
    - **Árbol no aclarado:** este tipo de árbol no está cuidado, por lo que los frutos presentan un menor tamaño y se encuentran agrupados. La detección no es buena. Por lo tanto, no resulta un caso interesante para la aplicación, ya que el campo está abandonado y el agricultor no invertirá en este tipo de sistemas de reconocimiento.

En la siguiente tabla se muestra de forma visual y esquemática los posibles casos que puede encontrar el sistema:

	Árbol aclarado	**Árbol no aclarado**
Fruto no maduro	-Tamaño del fruto pequeño -Color amarillento/verdoso -Frutos separados -Buena detección	-Tamaño del fruto pequeño -Color amarillento/verdoso -Frutos juntos -Pésima detección
Fruto maduro	-Tamaño del fruto grande -Color anaranjado -Frutos separados -Buena detección	-Tamaño del fruto mediano -Color anaranjado -Frutos juntos -Pésima detección

Como se observa, la detección de frutos es buena si la explotación agrícola está bien cuidada. El grado de madurez del fruto no es ningún problema para que el sistema lo pueda detectar de forma adecuada.

El principal problema del sistema es cuando los frutos se encuentran muy juntos, ya que se pierde información, es decir, al juntarse los centroides, hay una cantidad de frutos que no es detectada por el sistema.

Como bien se observa en la siguiente gráfica, el valor que predice el sistema se adapta de forma bastante correcta a la realidad, aunque en algunas ocasiones difiere del valor real. Esto es debido a todo lo mencionado anteriormente.

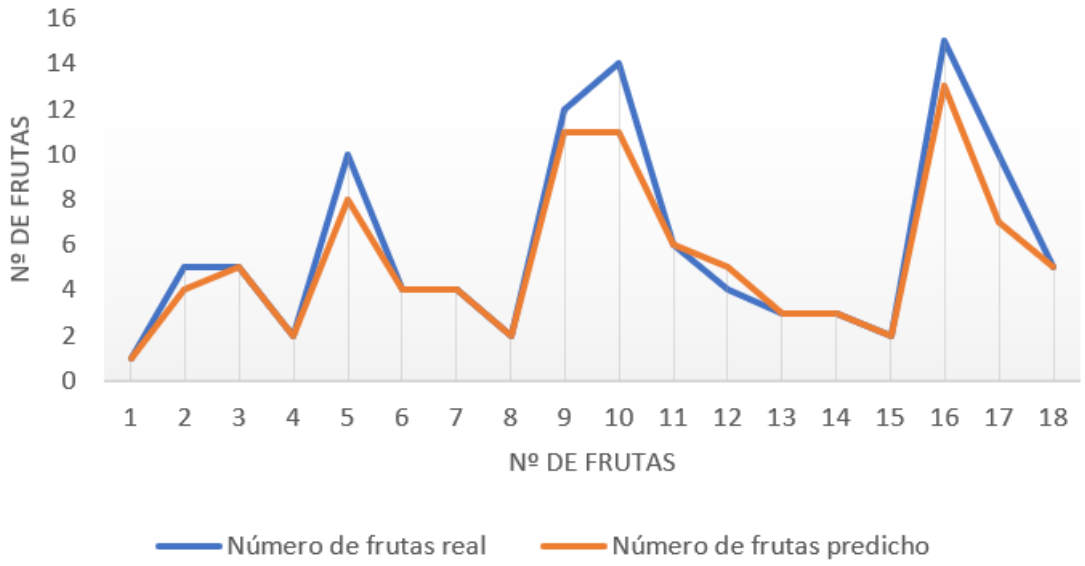

ILUSTRACIÓN 27 - COMPARACIÓN DEL MODELO CON LA REALIDAD

7-. Conclusiones

En este apartado se van a exponer las conclusiones finales sobre la realización del proyecto, así como las dificultades encontradas y las futuras líneas de investigación.

7.1-. Cumplimiento de los objetivos

Una vez realizado y testeado el proyecto se repasan los objetivos para comprobar su cumplimiento.

En primera instancia destacar que se ha logrado obtener un dataset suficientemente significativo, es decir, con las muestras necesarias para poder entrenar un modelo funcional.

Seguidamente, se ha logrado crear el código y entender todos sus parámetros para poder conseguir que el modelo funcione correctamente sin estar sesgado a los datos de entrenamiento, por lo tanto, el modelo obtenido no está sobrentrenado.

Además, se ha logrado exitosamente probar su funcionamiento en un dispositivo móvil, utilizando la cámara del mismo. La detección de las frutas ha sido satisfactoria debido a la calidad de la cámara.

Por otra parte, la implementación de la Raspberry Pi ha sido enriquecedora en cuanto a aprender sobre el funcionamiento de este tipo de modelos en este tipo de dispositivos. Aunque no se ha podido probar el funcionamiento del modelo mediante la cámara de esta, sí que se ha podido testear el comportamiento y los tiempos de procesamiento del modelo. Por lo tanto, se ha logrado obtener un modelo optimizado para este tipo de dispositivos.

Finalmente, se ha podido explotar al máximo la versión gratuita de la herramienta Edge Impulse, entrenando el modelo al límite de las características que ofrece su versión gratuita.

7.2-. Problemas encontrados

Uno de los problemas encontrados fue el etiquetado de las imágenes, ya que dependiendo de cómo se hiciesen, afectaba de forma muy severa a los resultados del entrenamiento y del test. Finalmente, mediante el sistema de ensayo y error se consiguió un resultado del etiquetado bastante correcto y significativo. Cuando se comprobaba el modelo con el conjunto de test, se obtenía que el modelo estaba sobreentrenado, por lo que modificando el etiquetado se solucionó.

Seguidamente, otro de los problemas fue la elección de la herramienta para el entrenamiento, ya que debido al tamaño de las fotos, era dificultoso su entrenamiento de forma gratuita. Se probaron diferentes herramientas y, finalmente, con la utilización de Edge Impulse, y gracias a los conocimientos adquiridos, se consiguió resolver el problema y se pudo entrenar un modelo con un dataset amplio.

Por otro lado, otro de los problemas fue la detección de frutas ocultas, ya que mediante el tipo de cámaras que se disponía resultaba una tarea complicada. Con la corrección y la modificación del etiquetado, se logró detectar una mayor cantidad de frutas, aunque estuviesen mayormente ocultas.

Las agrupaciones de frutos también resultan un problema, aunque si el frutal está bien cuidado es difícil encontrarlas, ya que los frutos se encuentran dispersos para que conserven bien su tamaño.

Finalmente, en la implementación del modelo con la Raspberry Pi, resultaba complicada la detección de las frutas, ya que debido a la baja calidad de la cámara, las imágenes quedaban pixeladas y el algoritmo no las podía detectar, por lo que se optó por probar cómo procesaba fotos obtenidas por otro medio. Con esto se observó que el rendimiento era óptimo y el modelo funcionaba correctamente en este tipo de dispositivos.

La tarea de realizar este tipo de algoritmos no es sencilla, ya que muchas veces para obtener un modelo correcto se deben de ir probando diferentes parámetros y etiquetados.

7.3-. Futuras implementaciones

Para finalizar este proyecto, se desean analizar las futuras implementaciones del sistema para mejorarlo e incluso poder llegar a comercializarlo e implantarlo en grandes explotaciones agrícolas.

Primeramente, para la correcta detección de los frutos, y como se indicaba en uno de los textos analizado en el apartado de “estado del arte”, con una cámara como la del dispositivo móvil o la Raspberry Pi, es difícil detectar los frutos que presentan oclusiones severas. Por lo tanto, una futura implementación sería migrar este sistema a un LiDAR, o una cámara de profundidad, con la cual mediante la nube de puntos que crea durante el escaneo, permitiría detectar una mayor cantidad de frutos.

Otra posible implementación sería embeber este sistema en un dron con un algoritmo de conteo, con el cual, mediante vuelos automatizados, se podría realizar un seguimiento de los frutos, y en el caso de que esta cantidad de fruto se viese afectada, se podrían tomar medidas al instante. Esta implementación sería muy útil para fincas con una gran cantidad de hectáreas con árboles frutales, ya que no sería necesario que el agricultor los recorriese con su vehículo.

En este apartado se observa como la implementación de algoritmos de visión por computador puede facilitar mucho la tarea de los agricultores. Además, la agricultura es un trabajo aún muy manual, y con este tipo de sistemas se puede ayudar a los agricultores a desempeñar de una forma mucho más sencilla su trabajo.

7.4-. Conclusiones y aportaciones personales

Como se ha ido mencionando en todo el proyecto las tareas de reconocimiento son tareas tediosas y complicadas. En el inicio de la realización de este proyecto resultaba complicado entrenar un modelo con imágenes de alta calidad y, además, que este funcionara de forma correcta. A medida que se ha ido investigando, se ha podido implementar un algoritmo mediante Edge Impulse que funciona de forma correcta.

Gracias a este proyecto, se ha podido profundizar sobre el aprendizaje automático y la creación de modelos. También, se han mostrado herramientas muy útiles y sencillas de aplicar, con las cuales se puede testear un sistema entrenado.

La implementación en la Raspberry Pi, ha servido para familiarizarse un poco más con este tipo de hardware. Con una correcta optimización del modelo de reconocimiento, este tipo de productos pueden convertirse en herramientas muy potentes.

Además, se ha logrado exprimir al máximo la versión gratuita de la plataforma Edge Impulse y se ha mostrado que para proyectos de investigación puede ser muy interesante.

Como conclusión, este proyecto me ha servido como colofón final de la carrera de Tecnologías Interactivas, y me ha acercado un poco más al mundo actual de la inteligencia artificial.

8-. ODS

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No procede
ODS 1. Fin de la pobreza				x
ODS 2. Hambre cero				x
ODS 3. Salud y bienestar			x	
ODS 4. Educación de calidad				x
ODS 5. Igualdad de género				x
ODS 6. Agua limpia y saneamiento				x
ODS 7. Energía asequible y no contaminante				x
ODS 8. Trabajo decente y crecimiento económico			X	
ODS 9. Industria, innovación e infraestructuras			x	
ODS 10. Reducción de las desigualdades				x
ODS 11. Ciudades y comunidades sostenibles				x
ODS 12. Producción y consumo responsables			x	
ODS 13. Acción por el clima	x			
ODS 14. Vida submarina				x
ODS 15. Vida de ecosistemas terrestres	x			
ODS 16. Paz, justicia e instituciones sólidas				x
ODS 17. Alianzas para lograr objetivos				x

Los Objetivos de Desarrollo Sostenible son un conjunto de 17 objetivos adoptados por todos los estados miembro de las Naciones Unidas en 2015 y forman parte de la Agenda 2030 de Desarrollo Sostenible. Estos abordan una serie de desafíos globales, como la pobreza, el cambio climático o la degradación ambiental.

Los ODS que se abordan en este proyecto son los siguientes:

- **ODS 13. Acción por el clima:** con la implantación de estos sistemas se tendría un cálculo de la producción antes de la etapa de recolección, por lo que las emisiones de CO2 y el consumo de combustibles fósiles se verían reducidos. Esto es debido a que como se sabe la producción no se utilizarían más medios (tractores, furgonetas o maquinaria) de los necesarios.

- **ODS 15. Vida de ecosistemas terrestres:** gracias a la utilización de estos sistemas la vida útil de las tierras aumenta, ya que con la implementación del sistema de reconocimiento con drones no sería necesario el desplazamiento de un agricultor con su respectivo vehículo a la explotación. Esto hace que el terreno no sufra erosión debido a las ruedas de los vehículos.

9-. Bibliografía

[1] J.A López Feldman, D. Hernández Cortés. “Cambio climático y agricultura: una revisión de la literatura con énfasis en América Latina”. El trimestre económico, vol. 83, no. 332, p. 459-496, 2016.

[2] F. Carrasco Choque. “Efectos del cambio climático en la producción y rendimiento de la quinua en el distrito de Juli, periodo 1997 - 2014”. *Comun@cción*, vol. 87, no. 2, p. 38-47, dic 2016.

[3] O.E Apolo Apolo, M. Pérez-Ruiz, G. Egea, y J. Martínez-Guanter, “Estimación de producción en cítricos usando técnicas de aprendizaje automático”, presentado en X Congreso Ibérico de Agroingeniería, Zaragoza (Huesca)-España, 3-6 de septiembre de 2019.

[4] J. Giménez-Gallego, J.D González-Teruel, A.B Toledo-Moreno, M. Jiménez-Buendía, F. Soto-Valles, R. Torres-Sánchez, “Segmentación en imagen de frutos de granado usando deep learning con aplicación en agriculuta de precisión”, presentado en las *XLIII Jornadas de Automática*, Cartagena.

[5] M.D Arévalo Zenteno, “Clasificación de la fruta del Durazno mediante Redes Neuronales”, Tesis doctoral, Universidad Autónoma del Estado de México, Texcoco, Estado de México, 2021.

[6] E. Gregorio, J.R. Rosell-Polo, J. Arnó, J. Gené, A. Escolà, R. Sanz, J.C. Miranda, “Detección de frutos y estimación de cosecha”. (Artículo en línea). Disponible en: https://www.grap.udl.cat/es/recerca/linies-de-recerca/deteccio-de-fruits-i-estimacio-de-collita/. [Último acceso: 13 abril 2024].

[7] J. A. F. Martínez, “MELOCOTONERO. Prunus perisca [Rosaceae]". (Artículo en línea). Disponible en: https://www.regmurcia.com/servlet/s.Sl?sit=c,365,m,1050&r=ReP-5157-DETALLE_REPORTAJES. [Último acceso: 13 abril 2024].

[8] A. Torres, “freeCodeCamp”. (Artículo en línea). Disponible en: https://www.freecodecamp.org/espanol/news/introduccion-al-aprendizaje-automatico/. [Último acceso: 23 mayo 2024].

[9] Keras, “keras.io”. (Artículo en línea). Disponible en: https://keras.io/api/applications/mobilenet/. [Último acceso: 12 junio 2024].

[10] R. Pi, “Raspberry Pi”. (Artículo en línea). Disponible en: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/. [Último acceso: 10 junio 2024].