Sistema de clasificación automática de sólidos mediante modelos de Machine Learning

Jorge Rivera Rueda

Autor

Francisco Javier Herraiz Martínez

Director

***Abstract*— En los últimos años se han llevado a cabo numerosos trabajos de investigación para conocer la permitividad relativa de los materiales. Estos proyectos se centraban en el desarrollo del sensor o en el desarrollo del sensor y del lector necesario. Lo que se propone en este trabajo es desarrollar un sistema completo, que además del sensor y del lector, sea capaz de clasificar productos y envases de manera automática mediante la aplicación de técnicas y modelos de Machine Learning y de Deep Learning, en base a la medida de la permitividad relativa obtenida con un sensor de metamateriales electromagnéticos. Este sistema se basa en el paradigma de IoT industrial, ya que el lector tiene una dirección IP con la que se hace una conexión de forma remota para obtener las medidas del sensor y enviar la información al ordenador para realizar el procesamiento de los datos y la clasificación de las señales. La aplicación de este sistema es la detección automática de productos o envases en entornos industriales y comerciales, como por ejemplo en supermercados o almacenes, sin necesidad de etiquetas o códigos de barras adicionales. Este documento se centra en la parte correspondiente al clasificador automático, donde se ha realizado todo el procesamiento de los datos para generar las imágenes de las señales y así poder realizar la clasificación mediante BoVW y diferentes Redes Neuronales Convolucionales (CNN). Se ha conseguido una accuracy de casi el 100% en la clasificación de los distintos productos.**

Keywords — IoT, Permitividad relativa, CNN, BoVW, Sensor, Lector, Clasificador Automático

I. Introducción

Internet ha evolucionado y se ha expandido rápidamente por todo el mundo a lo largo de los últimos 20 años, actualmente está en prácticamente todos los aspectos de nuestra vida y de nuestro día a día, lo que ha permitido el crecimiento de Internet of Things (IoT). A este crecimiento también ha contribuido en gran medida el acceso a sensores y dispositivos de bajo coste, bajo consumo y reducido tamaño, la expansión de la inteligencia artificial, la analítica de datos y de nuevos protocolos de red.

El concepto de IoT fue propuesto por Kevin Ashton en 1999 y una posible definición es la interconexión entre dispositivos, sistemas y servicios a través de una red, la cual puede ser privada o pública, sin la necesidad de intervención humana, la interacción es machine to machine (M2M). Esto permite la generación e intercambio de datos de manera continua entre de los dispositivos que posteriormente hay que procesar y analizar. Esta tecnología supone un gran cambio para la calidad de vida de las personas [1]-[5].

Las aplicaciones que tiene el IoT son casi ilimitadas y poco a poco van surgiendo nuevas formas y dispositivos que contribuyen a su expansión por todos los sectores. Por ejemplo, en el sector industrial, los dispositivos IoT permiten la monitorización y generación de alarmas en los diferentes procesos gracias a sensores y otros elementos conectados a la red que analizan los datos. Estos dispositivos son capaces incluso de aplicar acciones correctivas o protocolos de actuación sin necesidad de intervención humana.

Como cualquier tecnología, IoT tiene ventajas y desventajas. Algunas de las ventajas que presenta son la monitorización continua de los procesos, la reducción de costes, el aumento de la productividad y la eficiencia o la ayuda en la toma de decisiones. Mientras que los mayores problemas o reticencias que causa son la pérdida de privacidad, la invasión del espacio personal, el control que supone sobre los usuarios o la dificultad de que toda la comunicación sea segura y no haya brechas que puedan comprometer los datos [6]-[8].

Otro gran problema que presenta IoT es que cada fabricante utiliza su propio protocolo de comunicación, lo que hace que no todos los dispositivos sean compatibles entre sí. Para solucionar esto surgió el protocolo abierto MQTT, el cual se va a emplear en este proyecto.

Siguiendo el paradigma del IoT industrial, el objetivo de este proyecto es crear un sistema completo que sea capaz de clasificar automáticamente distintos productos. El sistema está formado por un sensor, un lector para interrogar al sensor y para proporcionar la comunicación entre el conjunto de sensor y lector con el ordenador, y un clasificador automático. Para el envío de los datos desde el lector hasta el ordenador se emplea el protocolo de comunicación máquina a máquina MQTT, un estándar abierto OASIS e ISO ampliamente utilizado en la comunicación entre dispositivos de IoT. Las medidas de los productos se toman en diferentes posiciones y condiciones. Este sistema permite hacer una clasificación de una manera rápida, sencilla y sin la necesidad de etiquetas o códigos de barras en los productos, por lo que su aplicación en la clasificación de productos o envases en supermercados o almacenes tiene un gran potencial. El esquema del sistema desarrollado se puede ver en la Figura 1.

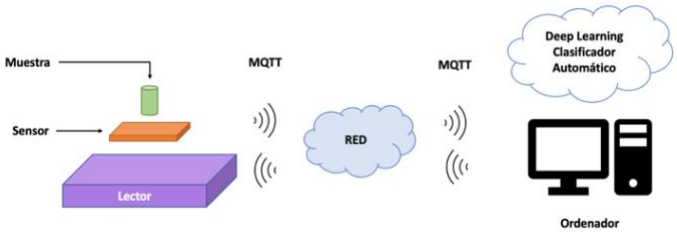

Figura 1. Esquema del sistema IoT desarrollado

A. *Sensor desarrollado*

Lo que se busca en este proyecto es la caracterización de la permitividad relativa de sólidos, para ello se ha desarrollado un novedoso sensor totalmente pasivo que utiliza resonadores con forma de espiral y que está basado en metamateriales electromagnéticos. Este sensor hace uso de tecnología planar, lo que hace que sea barato y fácil de fabricar además de muy resistente. También tiene un tamaño reducido y al ser totalmente pasivo no necesita alimentación externa para funcionar, lo que permite simplificar el proceso de toma de medidas, conseguir un sensor más versátil y sencillo y poder utilizarlo en diferentes aplicaciones donde el espacio sea un problema para emplear otros sensores más grandes. El sensor se ha diseñado para funcionar a una frecuencia de 2.45 GHz, ya que para esta banda existen multitud de componentes comerciales además de ser libre e ISM. El modo de funcionamiento de este sensor es situar los diferentes productos y envases de manera que se cubran las dos espirales. Una vez diseñado y fabricado el sensor, este se ha integrado en una "mesa" o "mostrador" para facilitar el proceso de toma de medidas. En la Figura 2 se muestra el sensor fabricado.

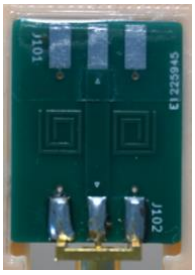

Figura 2. Sensor diseñado y fabricado

B. *Lector desarrollado*

El lector desarrollado en este proyecto se encarga de interrogar al sensor y de permitir que este pueda ser totalmente pasivo. El lector está formado por una interfaz de RF cuyos componentes son: un VCO para generar una señal senoidal de RF cuya frecuencia de salida depende de una señal triangular creada por el Arduino y que permite hacer un barrido de frecuencia continuo en el rango de interés, un detector para obtener una tensión a la salida proporcional a la potencia de la señal de RF de entrada y un circulador para separar las señales incidentes y reflejadas en el sensor, es decir, funcionando como un duplexor. El lector también cuenta con un Arduino, concretamente un Arduino MKR WIFI 1010, lo que permite convertir este sistema en un sistema IoT. Se han programado diferentes funciones en el Arduino, algunas de ellas son: un conversor digital-analógico (DAC), un conversor analógico-digital (ADC) y las funciones necesarias para la configuración y funcionamiento del protocolo de comunicación MQTT. Todo esto permite poder digitalizar las señales de los distintos productos obtenidas por el sensor y realizar el envío de los datos al ordenador. La ventaja de utilizar un lector basado en el paradigma de IoT es que se puede controlar remotamente a través de Internet, por lo que se puede mandar una orden en el momento deseado para tomar la medida o se puede hacer que estas se tomen automáticamente. Se ha elegido un Arduino al ser un dispositivo relativamente barato, fácil de conseguir y sencillo de programar, por lo que la complejidad de desarrollo y coste de este lector no es muy elevada. Una vez que se han montado los distintos componentes, el lector se ha cajeado para tener un sistema más ordenado, esto se puede ver en la Figura 3.

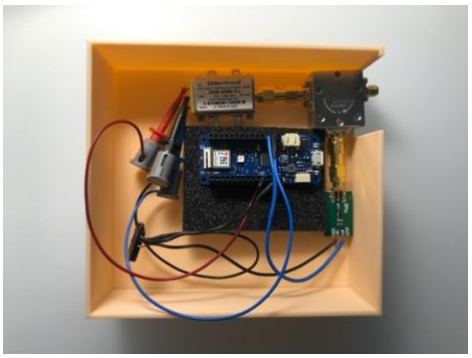

Figura 3. Lector diseñado y montado

C. *Introducción al clasificador*

Una vez que se ha finalizado el desarrollo del sensor y del lector, se puede comenzar con la parte correspondiente al clasificador automático, la cual comprende desde la selección de los productos y toma de medidas hasta el desarrollo de los diferentes modelos de clasificación, después de haber llevado a cabo el adecuado procesamiento de los datos para recomponer las señales.

II. SELECCIÓN DE PRODUCTOS A MEDIR

Para crear un sistema que sea lo más fácil de replicar y realista, se han escogido productos que cualquier persona tiene en casa o que se pueden conseguir fácilmente en un supermercado a un bajo precio. Se han elegido 9 productos o envases diferentes además de tomar las medidas cuando no hay nada sobre el sensor, por lo que en total se tienen 10 categorías o clases distintas, las cuales son:

- Aire (nada encima del sensor)
- Kiwi
- Caja de Té (envase de cartón)
- Bolsa de tela con arroz
- Paquete de Garbanzos (envase de plástico)
- Patata
- Tableta de chocolate
- Brik de leche
- Mandarina
- Bote de cristal con tomate

Figura 4. Productos seleccionados para medir

III. MEDICIÓN DE LOS PRODUCTOS

Para que el proceso de medición de los productos tenga sentido y sea realista, este se debe realizar en diferentes condiciones, por esta razón el proceso ha tenido una duración de más de una semana. A lo largo de todo ese tiempo y en diferentes momentos y horas del día se ha llevado a cabo la medición de los objetos sólidos. Aunque se haya intentado colocar los productos en una posición similar cada vez que se ha hecho una medida, por razones obvias es imposible replicar exactamente la misma posición, por lo que entre varias medidas de un producto siempre hay diferencias que pueden ser mayores o menores dependiendo de cómo se haya situado sobre el sensor en esa ocasión.

Para realizar cada una de las medidas de los diferentes objetos sólidos primero se coloca el producto sobre el sensor de manera que se cubran las dos espirales, después se ejecuta el script programado que se encarga de quedarse escuchando y de capturar la información que se publica en el topic del bróker de MQTT para posteriormente almacenarla en un CSV, por último, se manda el comando "ok" que inicia el proceso de toma de medidas y envío de los datos mediante MQTT.

El proceso completo de toma de medidas se ha repetido 12 veces, donde en cada una de las repeticiones se ha tomado un total de 10 medidas diferentes de cada uno de los productos. Es decir, cada vez que se hacía el proceso de medición se tomaban un total de 100 medidas distintas (10 medidas * 10 productos). Con cada una de las medidas se han obtenido 400 datos, ya que se ha programado el Arduino para tomar 200 valores medidos por el sensor y 200 timestamps de los instantes en los que se han tomado dichos valores. El volumen total de información capturada es de 480000 datos (12 veces * 10 medidas * 10 productos * 400 datos). Toda esta información se ha almacenado en varios archivos CSV. Debido a la gran cantidad de datos disponibles y a la dificultad de trabajar con señales e imágenes, es necesario el uso de técnicas de Big Data para el procesamiento de los datos y para la clasificación de las señales de cada uno de los productos.

En la Figura 5 y en la Figura 6 se muestra la posición de los diferentes productos sobre el sensor para realizar el proceso de toma de medidas, visto desde un lateral y visto desde arriba, respectivamente.

Figura 5. Posición de los productos para medir vista lateral

Figura 6. Posición de los productos para medir vista superior

IV. PROCESAMIENTO DE LOS DATOS

Antes de comenzar con la clasificación y entrenamiento de cada uno de los modelos que se van a probar en el proyecto, es necesario realizar el correspondiente procesamiento de los datos para generar los distintos conjuntos de imágenes de las señales.

A. Lectura y recomposición de las señales

Lo primero que hay que hacer es leer los ficheros CSV para poder recomponer las señales de cada uno de los productos medidos. Debido a que cada una de las medidas se ha realizado en un instante de tiempo diferente, cuando se ha enviado el comando "ok", el vector de tiempos tiene valores distintos, por lo que para que todas las señales del mismo producto sean lo más parecidas posibles, se ha utilizado la misma escala de tiempo. Las señales recompuestas se representan en la Figura 7.

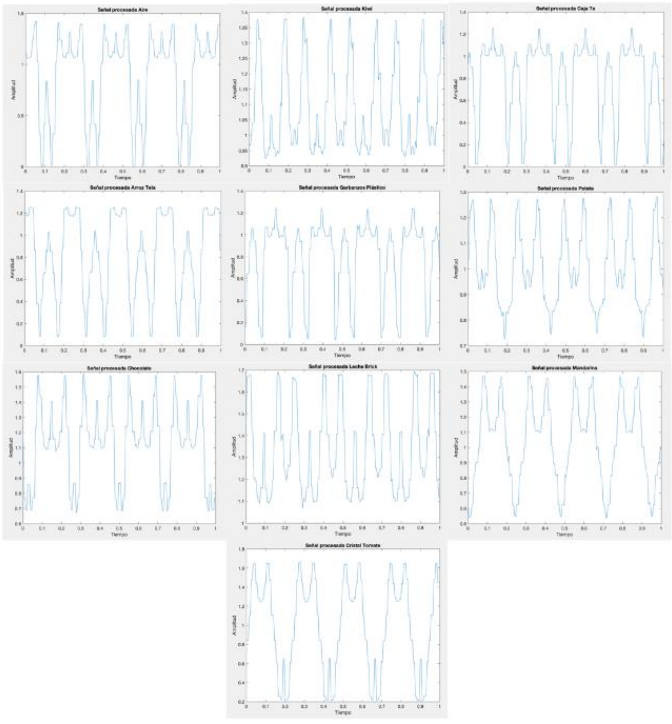

Figura 7. Señales recompuestas

B. *Análisis inicial*

Es necesario llevar a cabo una exploración inicial de los datos para conocerlos y encontrar similitudes, diferencias y características que a simple vista no se pueden detectar fácilmente. Para realizar este análisis inicial, se han utilizado las funciones *midcross* y *dtw* de Matlab.

La función *midcross* hace referencia a "Cruce de nivel de referencia media para forma de onda binivel". Esta función representa la señal y marca la ubicación de los cruces medios (instantes de nivel de referencia media) y los niveles de referencia asociados. También representa los niveles de estado con límites de estado superior e inferior [9].

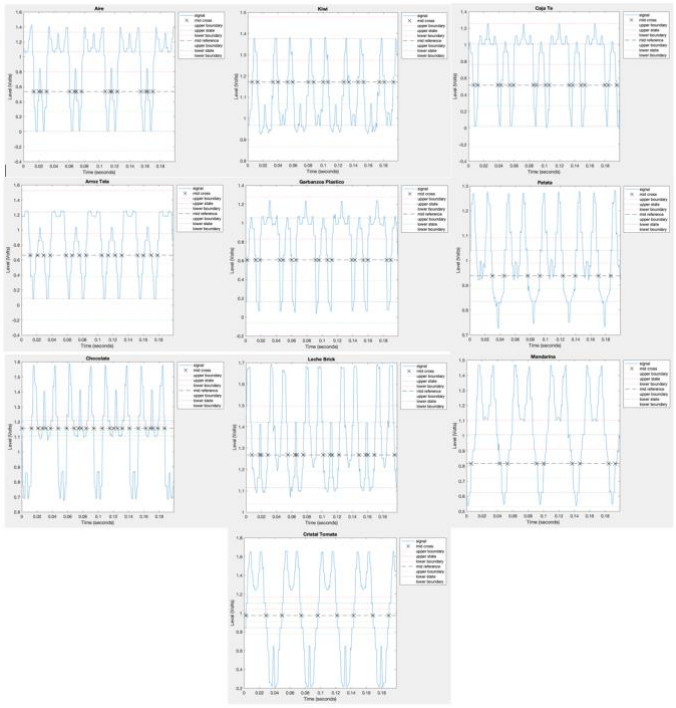

Figura 8. Midcross Señales

La función *dtw* hace referencia a "Distancia entre señales mediante deformación dinámica del tiempo". Esta función representa las dos señales originales en la parte superior y las dos señales alineadas en la parte inferior de la imagen [10].

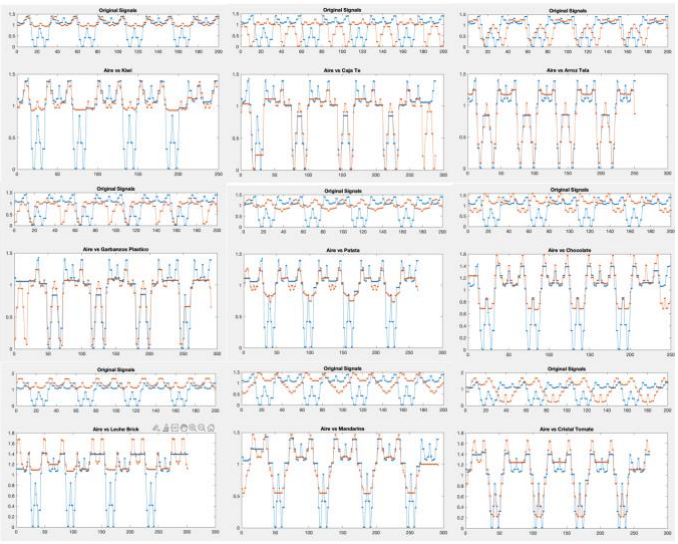

Figura 9. Dtw Señales

Aunque a simple vista ya se podían observar diferencias entre las señales de los distintos productos, después de realizar el análisis con las dos funciones explicadas, Figura 8 *midcross* y Figura 9 *dtw*, se puede comprobar que las señales difieren de manera considerable unas de otras. Hay productos, como la bolsa de tela con arroz o el bote de cristal con tomate, que tienen unas señales más parecidas a la señal de referencia, es decir, la señal que se obtiene cuando no hay nada sobre el sensor. En cambio, las señales de otros productos, como el kiwi o el brick de leche, no tienen nada que ver con esta señal de referencia.

C. *FFT*

Cuando se trabaja con señales, lo normal es representarlas tanto en el dominio temporal como frecuencial, por lo que una vez que se ha hecho el análisis de la representación de las señales en el dominio del tiempo, se van a generar las representaciones las señales de cada producto en el dominio de la frecuencia. De esta manera se van a obtener los componentes principales y se va a estudiar si la clasificación de los productos por medio de su representación frecuencial se puede realizar de manera más sencilla y rápida.

Para conseguir esta representación se ha utilizado la FFT, es decir, la transformada rápida de Fourier ("Fast Fourier Transform"), un algoritmo que permite calcular la transformada de Fourier discreta (DFT) y su inversa. La FFT permite calcular de manera más rápida y eficiente la DFT al eliminar gran parte de los cálculos repetitivos, además de conseguir una mayor precisión al disminuir los errores de redondeo.

Debido a que las señales obtenidas tienen una amplitud muy pequeña, se ha decidido utilizar una escala en dB para poder ver mejor las diferencias y los componentes en cada una de las gráficas. Solo nos interesa la parte positiva del espectro, es decir, de 0 a 500 Hz en el eje horizontal, y los valores comprendidos entre -40 dB y 0 dB en el eje vertical, ya que fuera de este rango se puede considerar ruido. Por esta razón se han filtrado las señales eliminando toda la información no relevante, el resultado se puede ver en la Figura 10.

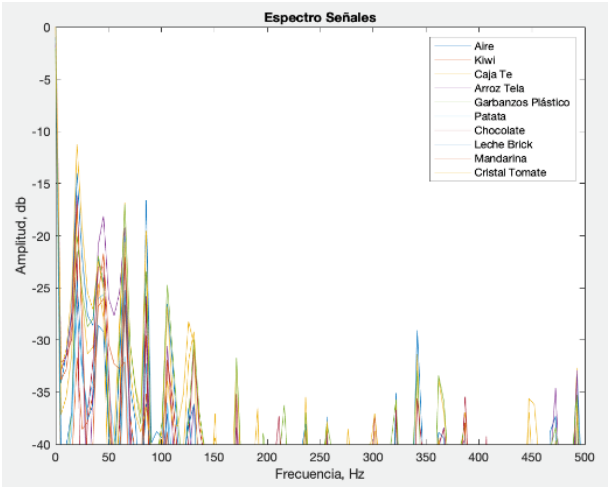

Figura 10. FFTs señales

D. *Enventanado*

Una técnica de procesamiento de señales muy común es el enventanado ("windowing"), que permite evitar las discontinuidades al principio y al final de los bloques analizados. El enventanado consiste en multiplicar la señal en el tiempo por una ventana de longitud finita con una amplitud que varía suave y gradualmente hacia cero en los bordes [11].

Hay diferentes tipos de ventanas que se pueden utilizar, para este proyecto se han probado cuatro distintas: Hamming, Hann, Gauss, Triangular. También se ha probado la ventana Rectangular, que es equivalente a dejar la señal como está. Se han seleccionado estos tipos de ventanas ya que son los más distintos entre sí y con los que se obtienen unos resultados más diferentes.

Después de aplicar cada uno de los tipos de ventana a las señales, Figura 11, se comprobó que no es una técnica muy útil en este proyecto, ya que, al ser las señales periódicas, no se produce ninguna mejora en los resultados y por tanto no se va a utilizar en la aplicación del proyecto.

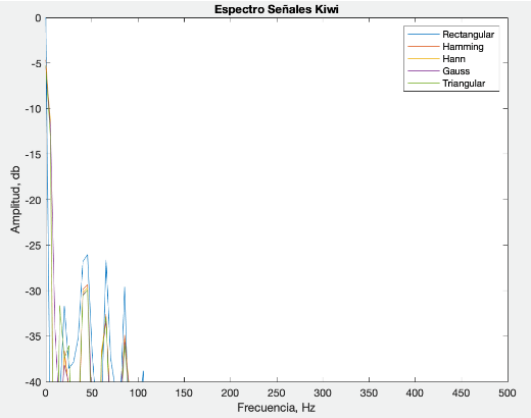

Figura 11. Enventanado Señales

E. *Espectrogramas*

Para el análisis de señales de voz y audio se utilizan a menudo los espectrogramas, los cuales consisten en una representación visual del espectro de frecuencias de señales que varían con el tiempo.

Para la creación de un espectrograma se coge un número concreto de muestras por medio de una ventana temporal que tiene un tamaño determinado. Después, se calcula la transformada de Fourier de las muestras y se representan los resultados en una gráfica tridimensional. Este proceso se repite a lo largo de toda la señal, desplazando la ventana para coger sucesivos conjuntos de muestras, calcular su contenido frecuencial y representarlo. Como resultado se obtiene una gráfica que aporta información sobre la variación de la energía y la frecuencia en función del tiempo [12]. Hay que llegar a un compromiso en la elección del tamaño de la ventana de tiempo, ya que cuanto más pequeño sea el tamaño de esta, mayor es la resolución temporal que se obtiene, pero menor es la resolución en frecuencia. Lo contrario sucede con tamaños de ventana más grandes.

A la hora de calcular los espectrogramas de las señales de cada uno de los productos, se descartó utilizar esta técnica para obtener las imágenes con las que entrenar los modelos de clasificación, ya que, al tomar las medidas en instantes de tiempo diferentes, las señales obtenidas de un mismo producto empiezan y terminan en momentos diferentes, por lo que los espectrogramas generados para un mismo objeto sólido son muy distintos entre sí y la clasificación no se podría hacer correctamente. Por esta razón, esta técnica no es adecuada para la aplicación y no se va a emplear. En la Figura 12 se muestran cuatro espectrogramas correspondientes al mismo producto.

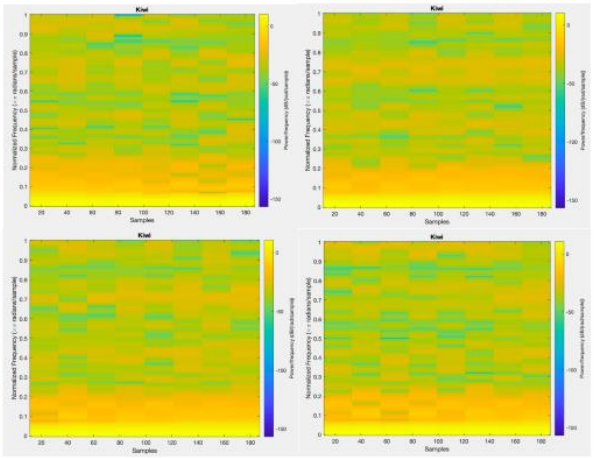

Figura 12. Espectrogramas

F. *Data Augmentation*

La última de las técnicas de procesamiento de los datos que se ha aplicado en el proyecto es Data Augmentation. Esta técnica consiste en aumentar la cantidad de datos disponibles mediante la agregación de nuevos datos que son copias de los originales, pero ligeramente modificados o mediante la creación de nuevos datos a partir de los ya existentes. Esto permite disponer de un volumen más grande y diverso de información con el que entrenar el clasificador y así reducir el *overfitting* que se pueda producir.

A pesar de tener ya un número suficiente de datos, al aplicar esta técnica se consigue duplicar la cantidad de imágenes disponibles, ya que se tienen las imágenes originales más las nuevas imágenes obtenidas con data augmentation.

Hay diferentes formas de aplicar esta técnica y conseguir nuevos datos, pero debido a que en este proyecto se trabaja con señales, estas no se pueden girar, rotar, recortar o desplazar, ya que se obtendrían señales completamente diferentes a las señales originales y todos los beneficios de usar data augmentation se perderían, además de que la clasificación no se podría hacer correctamente.

Para generar nuevas señales que difieran ligeramente de las originales, se les ha añadido ruido blanco gaussiano mediante la función *awgn* de Matlab. La cantidad de ruido a añadir no puede ser muy elevada para que la diferencia con la señal original no sea excesivamente grande, por lo que después de unas pruebas, se ha decidido añadir 30 dB de SNR. Con este valor de ruido se consiguen variaciones en las señales que también se podrían obtener realizando nuevas mediciones de los productos. Esto se ha aplicado tanto al conjunto de las señales representadas en el dominio temporal, Figura 13, como el conjunto de las señales representadas por su FFT, Figura 14.

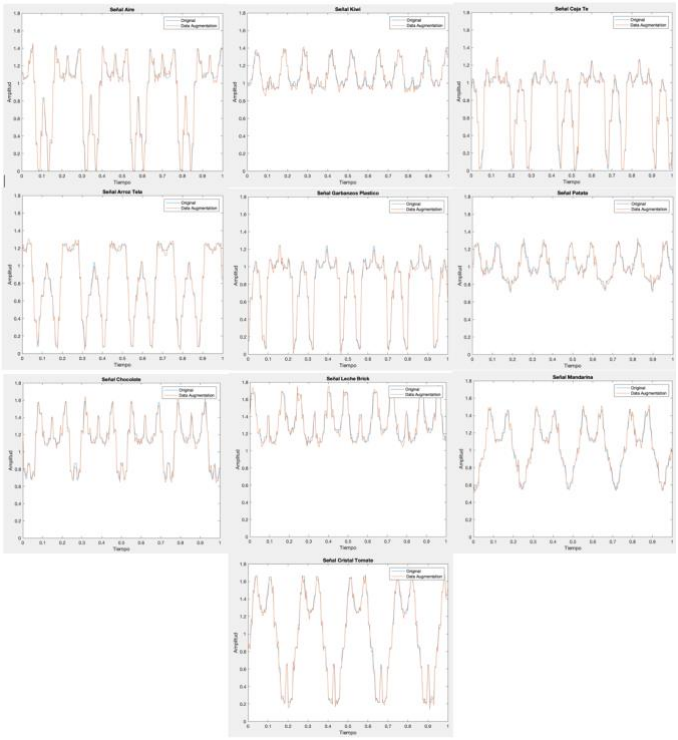

Figura 13. Data Augmentation Tiempo

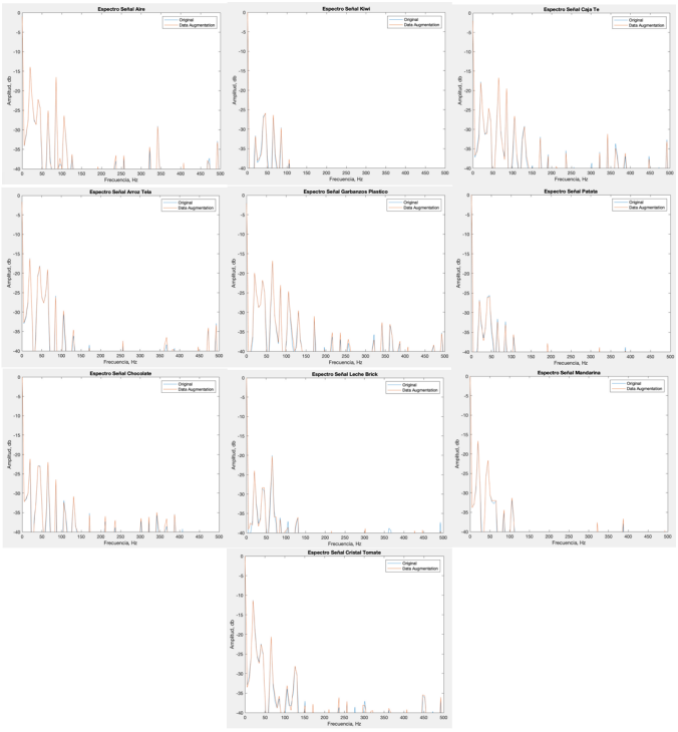

Figura 14. Data Augmentation Frecuencia

G. Generación de conjuntos de imágenes

Una vez que se ha hecho el procesamiento de las señales y se han probado todas las técnicas mostradas en los apartados anteriores, el siguiente paso es generar los diferentes conjuntos de imágenes que se van a emplear para entrenar los modelos de clasificación. Se han generado cuatro conjuntos diferentes: el de las señales en el dominio del tiempo, el de las señales en el dominio del tiempo aplicando data augmentation, el de las señales en el dominio de la frecuencia, el de las señales en el dominio de la frecuencia aplicando data augmentation.

Los conjuntos de imágenes sin data augmentation contienen 120 imágenes para cada uno de los productos, mientras que los que tienen data augmentation contienen 240 imágenes cada uno. El total de imágenes para los conjuntos a los que no se ha aplicado data augmentation es de 1200 imágenes (120 imágenes * 10 productos), mientras que para los que si se ha aplicado es de 2400 imágenes (240 imágenes * 10 productos).

Cada uno de los algoritmos y modelos de clasificación que se han utilizado en este proyecto han sido entrenados con estos cuatro conjuntos de imágenes. Se ha ajustado la escala de todas las señales, tanto la escala horizontal como la vertical, y se han recortado las imágenes para eliminar aquellas partes que no contienen información relevante, todo esto con la finalidad de facilitar el proceso de clasificación. En la Figura 15 se puede ver un ejemplo de imagen que se va a utilizar para realizar la clasificación de los productos.

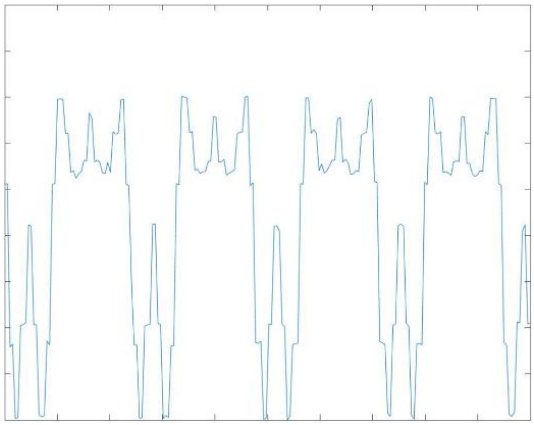

Figura 15. Ejemplo Imagen a clasificar

V. MODELOS DE CLASIFICACIÓN

Para conseguir que el proyecto sea lo más completo posible, se ha decidido emplear una técnica de Machine Learning tradicional como es Bag Of Visual Words (BoVW) y una técnica de Deep Learning como son las Redes Neuronales Convolucionales (CNN), de esta manera se pueden comparar los resultados obtenidos con cada uno de los métodos y determinar cuál es el mejor para realizar la clasificación de los productos.

En el caso de las CNN se ha probado dos aproximaciones distintas, la primera de ellas ha sido crear una Red Neuronal Convolucional desde cero y la otra ha sido emplear Transfer Learning para hacer la clasificación mediante una red ya preentrenada en la que se cambian únicamente las últimas capas para adaptarla a los datos del proyecto. Dentro de las redes preentrenadas se ha utilizado AlexNet, GoogleNet e InceptionV3, las cuales van desde la más simple y con menos capas de profundidad a la más compleja y con mayor número de capas, respectivamente.

Para todos los modelos se han seleccionado aleatoriamente un 80% de las imágenes como conjunto de entrenamiento y un 20% como conjunto test. Los nombres de

las 10 categorias a clasificar son: Aire, ArrozTela, CajaTe, Chocolate, CristalTomate, GarbanzosPlastico, Kiwi, LecheBrick, Mandarina y Patata.

En el caso de las CNNs, los parámetros empleados para realizar el entrenamiento de cada uno de los modelos son: *sgdm* o descenso de gradiente estocástico con impulso como algoritmo de optimización, 64 como tamaño del mini batch, 15 como número máximo de epochs y 0,0001 como tasa inicial de aprendizaje.

El ordenador empleado para llevar a cabo el entrenamiento de los modelos ha sido un MacBook Pro de 2015 con un procesador Intel i5 de doble núcleo a 2,9 GHz y 16 GB de RAM. Si se hubiese empleado un dispositivo que contase con una GPU, se podrían haber reducido los tiempos de entrenamiento de las CNN.

A. *BoVW*

BoVW es la primera de las técnicas de clasificación que se ha empleado en el proyecto, las siglas hacen referencia a "Bag of Visual Words". Esta técnica está basada en Bag of Words, un modelo que se emplea en el procesamiento de lenguaje natural (NLP).

BoVW lo que hace es representar una imagen como un conjunto de características o "features". Las características consisten en puntos clave (keypoints) y descriptores, donde los puntos clave son puntos destacados de una imagen. Independientemente de que la imagen se gire, se expanda o se rote, estos puntos clave siempre son los mismos. Los descriptores recogen información sobre los píxeles que rodean a los puntos clave [13]. Algunos de los "features extractors" más utilizados son SURF, HoG o SIFT, en este proyecto se emplea solamente SURF.

Una vez que se han obtenido los keypoints y los descriptores, se procede a construir el diccionario o bolsa de palabras de los puntos clave, para ello se emplean técnicas de clustering como K-Means. El siguiente paso es representar cada una de las imágenes del dataset mediante un histograma, donde se muestra la frecuencia con la que aparece en la imagen cada una de las features del diccionario [14].

Finalmente, a partir de los histogramas, se pueden encontrar imágenes similares o predecir la categoría de la imagen, para ello se emplea la técnica de clasificación supervisada máquina de vectores de soporte o "Support Vector Machines" (SVM). SVM es un clasificador biclase, por lo que busca la frontera o hiperplano que separe los puntos en dos grupos o clases diferentes, con la particularidad de que la distancia de los grupos con el hiperplano sea la mayor posible [14].

El proceso de extracción de características y de representación de los histogramas se muestra en la Figura 16.

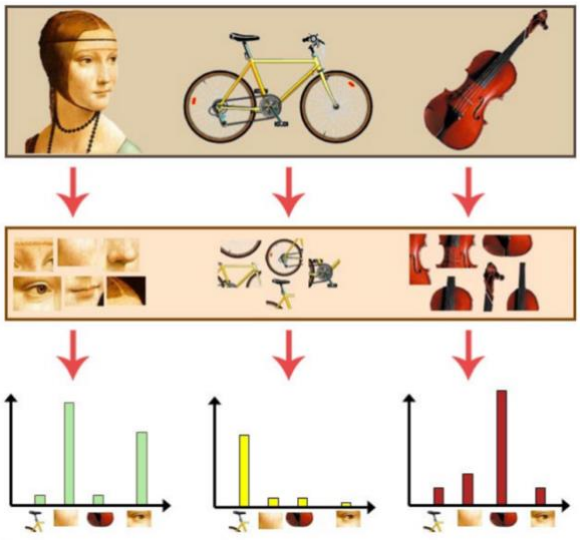

Figura 16. Extracción características y representación histogramas BoVW [13]

Después de aplicar BoVW a cada uno de los cuatro conjuntos de imágenes, se ha comprobado que se obtienen mejores resultados cuando se utilizan las imágenes con la representación temporal de las señales que cuando se emplean las imágenes con su representación frecuencial. El uso de data augmentation permite conseguir una mayor accuracy en ambas representaciones, siendo mayor la diferencia en el caso de las señales en el tiempo.

En ninguno de los casos se ha producido *overfitting* y la accuracy conseguida es prácticamente igual o mayor al 0.8 con todos los conjuntos de imágenes, por lo que se puede confirmar que Bag of Visual Words (BoVW) funciona correctamente y permite realizar la clasificación de los productos de una forma adecuada.

Los resultados obtenidos con cada uno de los conjuntos de imágenes se muestran en la Tabla 1.

Modelo/Accuracy	**Entrenamiento**	**Test**
Conjunto señales tiempo sin DA	0.925	0.9
Conjunto señales tiempo con DA	0.9458	0.9365
Conjunto señales frecuencia sin DA	0.8313	0.7667
Conjunto señales frecuencia con DA	0.8234	0.7771

Tabla 1. Resumen modelos BoVW

B. *CNN Diseñada*

Debido a que las CNNs preentrenadas que se van a probar en el proyecto constan de multitud de capas y son relativamente complejas, se ha decidido diseñar una CNN sencilla con pocas capas de profundidad para así contemplar todas las posibilidades.

Esta red neuronal consta de una capa de entrada donde las imágenes tienen que ser 227x227x3, lo que significa que las imágenes con las que se va a entrenar el modelo tienen que

ser a color (RGB) y deben tener un tamaño de 227 píxeles por 227 píxeles.

Después de la capa de entrada, hay una capa convolucional con una función de activación de tipo ReLU y una capa de normalización de canal transversal (cross channel). Luego se encuentra la capa de max pooling, que permite reducir la dimensionalidad y controlar el *overfitting*. Por último, este conjunto se repite de nuevo y finalmente se encuentra la capa completamente conectada con la función Softmax, donde se indican las 10 categorías a clasificar.

El esquema de las capas de esta red se puede ver en la Figura 17.

```
 1   'data'     Image Input                   227×227×3 images with 'zerocenter' normalization
 2   'conv1'    Convolution                   8 5×5 convolutions with stride [1  1] and padding [0  0  0  0]
 3   'relu1'    ReLU                          ReLU
 4   'norm1'    Cross Channel Normalization   cross channel normalization with 5 channels per element
 5   'pool1'    Max Pooling                   3×3 max pooling with stride [1  1] and padding [0  0  0  0]
 6   'conv2'    Convolution                   16 5×5 convolutions with stride [1  1] and padding [0  0  0  0]
 7   'relu2'    ReLU                          ReLU
 8   'norm2'    Cross Channel Normalization   cross channel normalization with 5 channels per element
 9   'pool2'    Max Pooling                   3×3 max pooling with stride [1  1] and padding [0  0  0  0]
10   'fc8'      Fully Connected               10 fully connected layer
11   'prob'     Softmax                       softmax
12   'output'   Classification Output         crossentropyex
```

Figura 17. Esquema capas CNN diseñada

A pesar de que la CNN diseñada es muy simple y se podía llegar a pensar que no se iban a conseguir unos buenos resultados, ha ocurrido lo contrario. La accuracy obtenida es superior al 90% para todos los conjuntos de imágenes, siendo en el mejor de los casos superior al 98%, y no se ha producido *overfitting*. A diferencia de BoVW, la clasificación de las imágenes representadas por medio de sus FFTs se realiza mejor que cuando se representan las señales en el dominio del tiempo. Los conjuntos con data augmentation consiguen unos resultados ligeramente mejores, pero al requerir el doble tiempo para completar el entrenamiento, se puede concluir que no merece la pena usarlos. Todo esto se puede comprobar en la Tabla 2.

Modelo/Resultados	Tiempo	Entrenamiento	Test
Conjunto señales tiempo sin DA	21 mins y 44 secs	1	0.9292
Conjunto señales tiempo con DA	50 mins y 38 secs	0.999	0.9375
Conjunto señales frecuencia sin DA	21 mins y 15 secs	0.9927	0.9792
Conjunto señales frecuencia con DA	52 mins y 46 secs	0.9995	0.9812

Tabla 2. Resumen modelos CNN Diseñada

C. *AlexNet*

La primera de las redes que se va a utilizar en este proyecto mediante la aplicación de Transfer Learning es AlexNet. Esta CNN fue diseñada por Alex Krizhevsky en colaboración con Ilya Sutskever y Geoffrey Hinton para la ImageNet Large Scale Visual Recognition Challenge de 2012 (ILSVRC 2012). Es una de las más famosas y utilizadas en el mundo. La red ha sido entrenada para clasificar 1000 categorías distintas de objetos, algunos de ellos son un teclado, un ratón, un lápiz o diferentes tipos de animales.

AlexNet tiene 8 capas de profundidad y el tamaño de las imágenes a introducir en la capa de entrada es de 227x227x3. Las primeras cinco capas son capas convolucionales, algunas de ellas seguidas de una capa de max pooling, y las últimas tres capas son capas completamente conectadas. La función de activación que utiliza es de tipo ReLU. En la Figura 18 se muestra el esquema de la red y en la Figura 19 su arquitectura.

```
 1   'data'     Image Input                   227×227×3 images with 'zerocenter' normalization
 2   'conv1'    Convolution                   96 11×11×3 convolutions with stride [4  4] and padding [0  0  0  0]
 3   'relu1'    ReLU                          ReLU
 4   'norm1'    Cross Channel Normalization   cross channel normalization with 5 channels per element
 5   'pool1'    Max Pooling                   3×3 max pooling with stride [2  2] and padding [0  0  0  0]
 6   'conv2'    Grouped Convolution           2 groups of 128 5×5×48 convolutions with stride [1  1] and padding [2  2  2  2]
 7   'relu2'    ReLU                          ReLU
 8   'norm2'    Cross Channel Normalization   cross channel normalization with 5 channels per element
 9   'pool2'    Max Pooling                   3×3 max pooling with stride [2  2] and padding [0  0  0  0]
10   'conv3'    Convolution                   384 3×3×256 convolutions with stride [1  1] and padding [1  1  1  1]
11   'relu3'    ReLU                          ReLU
12   'conv4'    Grouped Convolution           2 groups of 192 3×3×192 convolutions with stride [1  1] and padding [1  1  1  1]
13   'relu4'    ReLU                          ReLU
14   'conv5'    Grouped Convolution           2 groups of 128 3×3×192 convolutions with stride [1  1] and padding [1  1  1  1]
15   'relu5'    ReLU                          ReLU
16   'pool5'    Max Pooling                   3×3 max pooling with stride [2  2] and padding [0  0  0  0]
17   'fc6'      Fully Connected               4096 fully connected layer
18   'relu6'    ReLU                          ReLU
19   'drop6'    Dropout                       50% dropout
20   'fc7'      Fully Connected               4096 fully connected layer
21   'relu7'    ReLU                          ReLU
22   'drop7'    Dropout                       50% dropout
23   'fc8'      Fully Connected               1000 fully connected layer
24   'prob'     Softmax                       softmax
25   'output'   Classification Output         crossentropyex with 'tench' and 999 other classes
```

Figura 18. Esquema capas AlexNet

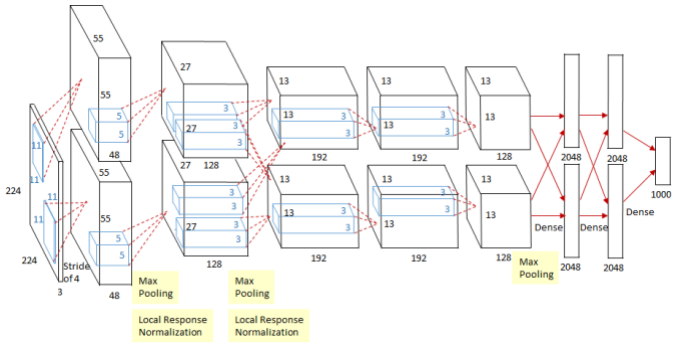

Figura 19. Arquitectura AlexNet [15]

Mirando el esquema de la red, se puede observar que se puede reutilizar desde la capa 1 hasta la capa 22 incluida, ya que la capa 23 es la capa completamente conectada y esta hay que adaptarla a las categorías existentes en el proyecto.

Con AlexNet se consiguen unos resultados muy buenos, con una accuracy de casi el 100% en el caso de las imágenes de las señales representadas en el tiempo sin data augmentation. Con la representación de las FFTs de las señales se consigue una peor accuracy y el empleo de data augmentation no ayuda a mejorar los resultados. Con ninguno de los conjuntos se ha producido *overfitting* y hasta ahora es el modelo que mejor realiza la clasificación de los productos. El resumen de este modelo se puede ver en la Tabla 3.

Modelo/Resultados	Tiempo	Entrenamiento	Test
Conjunto señales tiempo sin DA	16 mins y 1 sec	0.9979	0.9958
Conjunto señales tiempo con DA	38 mins y 10 secs	0.9958	0.9896
Conjunto señales frecuencia sin DA	16 mins y 10 secs	0.9542	0.9427
Conjunto señales frecuencia con DA	31 mins	0.9755	0.9479

Tabla 3. Resumen modelos AlexNet

D. *GoogleNet*

La siguiente CNN que se va a probar en el proyecto es GoogleNet, la cual fue desarrollada en 2014 por una investigación de Google en colaboración con varias universidades. Esta red fue la ganadora de la ImageNet Large

Scale Visual Recognition Challenge de ese año (ILSVRC 2014). Proporciona una disminución significativa de la tasa de error en comparación con otras redes ganadoras del ILSVRC como AlexNet [16].

Esta red consta de 22 capas de profundidad y se diseñó para tener en cuenta la eficiencia computacional, pudiendo ejecutarse en dispositivos que tengan pocos recursos computacionales. Las imágenes que se introducen en la capa de entrada deben tener un tamaño de 224x224 y todas las funciones de activación de las capas convolucionales son de tipo ReLU. En la Figura 20 se puede ver la arquitectura completa de GoogleNet.

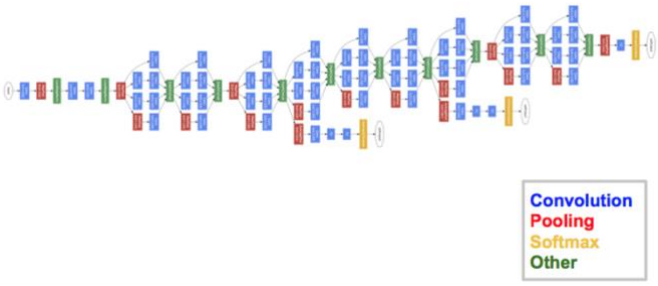

Figura 20. Arquitectura GoogleNet [16]

GoogleNet ofrece un buen rendimiento en la clasificación de las imágenes al conseguir una accuracy de 0.87 de media entre los cuatro conjuntos de imágenes. Al ser una red mucho más compleja y con más capas que AlexNet o que la CNN diseñada, los tiempos de entrenamiento son considerablemente superiores.

La accuracy conseguida con las imágenes de la representación temporal de las señales es bastante mejor que la obtenida con la representación en frecuencia. En el caso de las FFTs, el empleo de data augmentation sí que permite mejorar los resultados. Al igual que en los apartados anteriores, no se ha producido *overfitting*. En la Tabla 4 se encuentra el resumen de los resultados conseguidos con esta red.

Modelo/Resultados	Tiempo	Entrenamiento	Test
Conjunto señales tiempo sin DA	47 mins y 25 secs	0.9885	0.9875
Conjunto señales tiempo con DA	93 mins y 58 secs	0.9854	0.9771
Conjunto señales frecuencia sin DA	47 mins y 4 secs	0.8771	0.8417
Conjunto señales frecuencia con DA	92 mins y 58 secs	0.9240	0.9062

Tabla 4. Resumen modelos GoogleNet

E. InceptionV3

InceptionV3 es la última CNN que se va a probar en el proyecto, esta red comenzó como un módulo para GoogleNet y está formada por distintos bloques tanto simétricos como asimétricos, que incluyen convoluciones, avg pooling, max pooling, concatenaciones y capas completamente conectadas. Softmax es la función de pérdidas que se utiliza y ReLU la función de activación. Las imágenes empleadas para entrenar la red deben tener un tamaño de 299x299x3 [17].

La arquitectura completa de InceptionV3 se puede ver en la Figura 21.

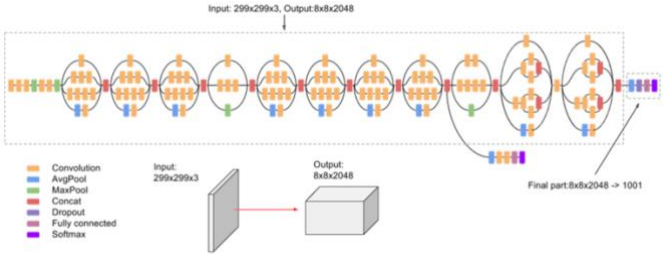

Figura 21. Arquitectura InceptionV3 [17]

Lo primero que llama la atención es la cantidad de minutos que se necesitan para completar el entrenamiento de esta red, esto es debido a la profundidad de la red y la gran cantidad de capas que tiene. La accuracy media que se consigue en el caso de la representación temporal de las señales es de 0.98, mientras que en el caso de la representación frecuencial es de 0.87, ambos valores muy elevados y buenos. Para ninguno de los cuatro conjuntos de imágenes se produce *overfitting*.

A pesar de tener una accuracy muy buena, en comparación con las otras redes que se han probado, esta no es la mejor opción, ya que requiere mucho más tiempo de entrenamiento sin que esto signifique que sea el modelo que proporciona los mejores resultados a la hora de hacer la clasificación de los productos. Esto se puede comprobar mirando la Tabla 5.

Modelo/Resultados	Tiempo	Entrenamiento	Test
Conjunto señales tiempo sin DA	201 mins y 15 secs	0.9948	0.9542
Conjunto señales tiempo con DA	387 mins y 54 secs	0.9979	0.9438
Conjunto señales frecuencia sin DA	165 mins y 55 secs	0.9854	0.8792
Conjunto señales frecuencia con DA	327 mins y 34 secs	0.9984	0.9333

Tabla 5. Resumen modelos InceptionV3

VI. Conclusiones y Trabajo futuro

Todos los modelos que se han probado en este proyecto funcionan correctamente y son capaces de clasificar con una elevada accuracy los distintos productos. En la Tabla 6 se puede ver el resumen de los resultados obtenidos con cada uno de los modelos para cada uno de los conjuntos de imágenes empleados.

Excepto la CNN diseñada desde cero, el resto de los modelos clasifican mejor las imágenes de las señales representadas en el dominio del tiempo que de la frecuencia. Todos los modelos que realizan la clasificación de los conjuntos de imágenes con las señales representadas en el tiempo tienen una accuracy superior o igual a 0.9, mientras que los modelos que clasifican las FFTs de las señales

presentan una accuracy más baja, sobre todo BoVW, que no supera el 0.8 en ninguno de los dos conjuntos.

Se ha demostrado que aplicar data augmentation no permite mejorar en gran medida la accuracy de los modelos, ya que en muchas ocasiones la diferencia es mínima y, por tanto, no merece la pena emplear el doble de tiempo en completar el entrenamiento para conseguir una accuracy solo un poco más alta.

Las CNNs presentan mejores resultados que Bag of Visual Words (BoVW), lo que quiere decir que las técnicas de Deep Learning tiene un mejor desempeño respecto a las técnicas tradicionales de Machine Learning a la hora de llevar a cabo la clasificación de imágenes de señales, tanto en el dominio del tiempo como de la frecuencia.

La red AlexNet y el conjunto de las imágenes de las señales representadas en el dominio del tiempo sin data augmentation es el caso con en el que se realiza mejor la clasificación de los productos, la accuracy obtenida es de casi el 100%. Además, es el modelo que menos tiempo de entrenamiento necesita.

Modelo/ Accuracy Test	**BoVW**	**CNN Diseñada**	**Alex Net**	**Google Net**	**Inception V3**
Conjunto señales tiempo sin DA	0.9	0.9292	0.9958	0.9875	0.9542
Conjunto señales tiempo con DA	0.9365	0.9375	0.9896	0.9771	0.9438
Conjunto señales frecuencia sin DA	0.7667	0.9792	0.9427	0.8417	0.8792
Conjunto señales frecuencia con DA	0.7771	0.9812	0.9479	0.9062	0.9333

Tabla 6. Resumen modelos

Como trabajo futuro quedaría probar si un diseño diferente de CNN al que se ha desarrollado en este proyecto permite conseguir una accuracy mejor que AlexNet con un menor tiempo de entrenamiento.

En este proyecto se ha demostrado que el empleo de técnicas de Machine Learning y de Deep Learning para la clasificación de señales de la medida de la permitividad relativa de distintos productos es todo un éxito. Estas señales se obtienen por medio de un sensor basado en metamateriales electromagnéticos. Antes de hacer la clasificación es necesario llevar a cabo el correspondiente procesamiento de los datos y recomposición de las señales. Todo esto hace que la aplicación de este sistema en entornos industriales y comerciales tenga mucho potencial y sea algo novedoso, ya que no son necesarios códigos de barras, QR, etiquetas u otros elementos adicionales.

VII. REFERENCIAS

[1] (Jan, 2021). Internet de las cosas. Available: https://es.wikipedia.org/wiki/Internet_de_las_cosas

[2] M. Gracia (Nov, 2020). IoT – Internet Of Things. Available: https://www2.deloitte.com/es/es/pages/technology/articles/IoT-internet-of-things.html

[3] jjtorres (Oct, 2014). ¿Qué es y cómo funciona el Internet de las cosas? Available: https://hipertextual.com/archivo/2014/10/internet-cosas/

[4] D. Ballard (May, 2016). Intro to the Internet of Things. Available: https://www.redhat.com/es/blog/intro-internet-things

[5] R. Alonso (Dec, 2020). ¿Qué es el Internet de las cosas (IoT) y por qué se le llama así? Available: https://hardzone.es/reportajes/que-es/internet-cosas-iot/

[6] (Dec, 2019). ¿Internet de las Cosas? Ventajas y Desventajas. Available: https://www.winecta.com/internet-de-las-cosas-ventajas-desventajas/

[7] P. Seguí (Nov, 2014). Internet de las cosas; Qué es y cuáles son sus ventajas y desventajas. Available: https://ovacen.com/internet-de-las-cosas/

[8] (Nov, 2019). 10 beneficios del IoT para empresas. Available: https://www.adeccoinstitute.es/futuro-del-trabajo-y-tecnologia/10-beneficios-del-iot-para-empresas/

[9] Midcross. Available: https://es.mathworks.com/help/signal/ref/midcross.html

[10] Dtw. Available: https://es.mathworks.com/help/signal/ref/dtw.html

[11] (Apr, 2020). Ventana (función). Available: https://es.wikipedia.org/wiki/Ventana_(función)

[12] (Feb, 2021). Espectrograma. Available: https://es.wikipedia.org/wiki/Espectrograma

[13] B. Davida (Jul, 2018). Bag of Visual Words in a Nutshell. Available: https://towardsdatascience.com/bag-of-visual-words-in-a-nutshell-9ceea97ce0fb

[14] V. González Castro, E. Alegre, E. Fidalgo, "Clasificación de imágenes con Bag of Visual Words," Grupo de Visión del Comité Español de Automática (CEA), Cap. 10, pp. 181-200, Jun. 2016.

[15] Programador clic. AlexNet Comprensión. Available: https://programmerclick.com/article/34611003618/

[16] S. Das (Nov, 2017). CNN Architectures: LeNet, AlexNet, VGG, GoogLeNet, ResNet and more… Available: https://medium.com/analytics-vidhya/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5

[17] Google (Mar, 2021). Guía avanzada de Inception v3 para Cloud TPU. Available: https://cloud.google.com/tpu/docs/inception-v3-advanced