Trabajo Fin de Máster

Sistema de identificación de vehículos basado en campo magnético

Vehicle identification system based on magnetic field

Autor/es

Rubén Calle Berges

Director/es

Rubén Blasco Marín
Roberto Casas Nebra

Escuela de Ingeniería y Arquitectura
2018

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. Rubén Calle Berges,

con nº de DNI 76972317B en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Máster, (Título del Trabajo) Sistema de identificación de vehículos basado en campo magnético

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 28 de junio del 2018

Fdo: Rubén Calle Berges

Sistema de identificación de vehículos basado en campo magnético

Resumen

En la actualidad, la sociedad demanda una alta cantidad de bienes de consumo. Para poder hacer frente a esta, son necesarios un gran número de transportes, tanto de productos como de personas. Esta situación provoca que se produzcan muchos de desplazamientos a través de las vías públicas. Para asegurarse de que se realicen de forma segura y eficiente es necesario la implementación de los sistemas de control del tráfico.

El sistema de monitorización del tráfico más extendido es el bucle inductivo, sin embargo, presenta algunos problemas en cuanto a la calidad de la información que proporciona y su vida útil. Por ello están ganando importancia como futuros sustitutos los nodos de sensores inalámbricos. Estos proporcionan una información más rica, además de una vida útil más larga. También presentan una gran capacidad para formar redes entre ellos y otros dispositivos permitiendo el desarrollo de la Smart City.

El objetivo que se persigue en este trabajo es un estudio de las posibilidades de detección de la dirección y estimación del tamaño de los vehículos mediante métodos de identificación, basados en la medida de la deformación en el campo magnético terrestre y de las vibraciones inducidas en la calzada por el paso de un vehículo.

Para ello, primeramente se realiza una investigación de los sistemas más relevantes para la monitorización de vehículos que actualmente están implantados así como las distintas líneas de desarrollo en este campo. Esta información es utilizada para la toma de las decisiones previas al desarrollo de los clasificadores

Posteriormente se realiza una experimentación de campo mediante un nodo sensor que ha permitido capturar la huella magnética y las vibraciones inducidas en el asfalto de 113 vehículos. Pre-procesando esta información se obtiene una base de datos con información de las características de los vehículos y sus correspondientes capturas.

Con la ayuda de las conclusiones obtenidas del estudio de trabajos previos y mediante la base de datos conseguida mediante la experimentación, se diseñan dos clasificadores, uno para distinguir la dirección de los vehículos y otro para diferenciar entre los mayores de 4,5m y los menores.

Como conclusión final, el principal objetivo de este trabajo se cumple satisfactoriamente. La viabilidad de esta tecnología queda demostrada. También se observa que la clasificación basada en el tamaño se puede realizar con la utilización de unas características de bajo coste computacional, mientras que la basada en la dirección necesita unas más complejas.

INDICE

CAPÍTULO 1. INTRODUCCIÓN

1. INTRODUCCIÓN

En la actualidad el sistema de monitorización del tráfico más extendido es el bucle inductivo, sin embargo, presenta algunos problemas como que la información que proporciona es muy pobre, es costoso debido a su mantenimiento y con una vida útil reducida. Los nodos de sensores inalámbricos (Wireless Sensor Network, WSN) equipados con un sensor magnético se presentan como los sustitutos de los lazos inductivos. Estos sensores poseen una gran ventaja frente a la otra tecnología, lo que abre la posibilidad de su utilización no solo para la detección de vehículos, sino también para realizar su clasificación.

En mi Trabajo Final de Grado (TFG) [1]se desarrolló un prototipo de nodo WSN basado en la captura de la "huella" magnética de un vehículo y de las vibraciones que induce en la calzada. El objetivo de este dispositivo era la realización de las pruebas de campo que permitieran comprobar esta tecnología. En este proyecto se pretende continuar con esta línea de trabajo. Para ello se desea verificar la viabilidad de desarrollar clasificadores de vehículos basados en los parámetros que permite capturar este equipo.

2. OBJETIVOS

El principal objetivo de este Trabajo Final de Master (TFM), es un estudio de las posibilidades de detección de la dirección y estimación del tamaño de los vehículos mediante métodos de identificación, basados en la medida de la deformación en el campo magnético terrestre y de las vibraciones inducidas en la calzada por el paso de un vehículo.

Para el desarrollo de este objetivo se proponen los siguientes objetivos parciales:

1. Búsqueda de información sobre métodos de clasificación.
2. Definición y realización de una prueba de campo para la captura de datos.
3. Análisis y procesamiento de los datos para la creación de una base de datos.
4. Análisis de la relevancia de las características de los datos (en adelante *features*).
5. Desarrollo de los clasificadores.
6. Validación de la calidad de los clasificadores.

3. MARCO DE DESARROLLO DEL PROYECTO

Este TFM se engloba dentro del proyecto de investigación Movilidad Verde Inteligente (MOVVI)[2]. Se presenta de la mano de dos empresas complementarias y expertas en el sector. Kepar Electrónica aporta su experiencia en el campo del diseño y la fabricación electrónica, en concreto en el desarrollo y comercialización de dispositivos para la gestión inteligente del tráfico. iA Soft proporciona y mantiene el software de varios centros de tráfico, entre ellos el de Zaragoza. En este sentido, esta alianza posibilita la aparición de un nuevo producto, modular, capaz de competir en los mercados nacionales e internacionales, existiendo ya intereses comerciales por el mismo. Este tándem, se complementa por la Universidad de Zaragoza, experta en el desarrollo de sensores de bajo consumo, modelado de dispositivos y comunicaciones inalámbricas. Y todo ello se circunscribe en la ciudad de Zaragoza encaminada hacia la línea de Smart cities para crear un conjunto de productos innovadores.

Fig. 1 Formación proyecto MOVVI

El objetivo de MOVVI es desarrollar un sistema inteligente que, apoyado en una red de sensores y actuadores distribuida por el entorno vial, realice una gestión más eficiente del tráfico y la ocupación en espacios públicos (vías, plazas de aparcamiento, etc.) y privados (parkings, monitorización de accesos u ocupaciones, etc.). La propuesta es un escenario donde los sensores y los elementos de regulación, se encuentren distribuidos, en las vías públicas, agrupados en pequeñas redes. Utilizarán internamente, comunicaciones inalámbricas de bajo consumo y se apoyarán en puntos de enrutamiento para acceder a internet y publicar la información en las bases de datos. Esta arquitectura escalable permite plantear la cobertura de tanto una pequeña área hasta la monitorización completa de una ciudad. Los datos recolectados serán procesados en la nube (Cloud Computing) generando información de alto nivel que se integrará de forma nativa en la solución de control urbano para grandes ciudades desarrollada en el ámbito del proyecto o, por medio de API ofertada, en cualquier otra solución comercial.

El ámbito de este TFM se encuadraría dentro del paquete de trabajo 4: Desarrollo de la micro-nube de sensores y Gateway, en concreto dentro de la tarea: T4.4 Análisis de datos de sensores. Coordinada por UNIZAR y con la participación de Kepar. Se analizarán los datos procedentes de los sensores, planteando algoritmos integrables en microcontrolador con la finalidad de mejorar la inteligencia de los sensores.

4. METODOLOGÍA

Para abordar este proyecto se sigue el método científico, estableciendo una hipótesis de partida y realizado unas pruebas experimentales para confirmarla. Esta hipótesis es que la información de la huella magnética de un vehículo, junto a las vibraciones que induce en la calzada, es suficiente para realizar una clasificación basada en su tamaño y su dirección. Se utiliza un planteamiento modular, descomponiendo el problema en otros más simples, facilitando su resolución. Las líneas de trabajo seguidas son las siguientes:

Como primer paso para abordar este proyecto es necesario realizar un estudio de los sistemas de clasificación de vehículos y, en especial, de los basados en la huella magnética y las vibraciones inducidas en la calzada.

Posteriormente, se genera una base de datos con huellas magnéticas y de vibraciones producidas por el paso del paso de vehículos. Esta base de datos se crea a partir del pre-procesado de los datos obtenidos en una experimentación de campo donde se capturan las medidas del campo magnético terrestre y de las vibraciones inducidas en la calzada.

Los datos generados son demasiado extensos para poder utilizarlos directamente en los clasificadores. Por tanto se debe extraer de estos diferentes *features*, que se podrán utilizar para los clasificadores. Como a priori se desconoce a ciencia cierta cuales son los mejores para realizar la clasificación, se requiere efectuar un análisis de la relevancia de estos para la selección de los más óptimos.

Con los seleccionados se desarrolla el clasificador. Análogamente al problema anterior, también se desconoce cuál es la arquitectura del clasificador más óptima, por tanto también se realiza su análisis para la selección de la más óptima.

5. ESTRUCTURA

La presente memoria recoge el trabajo realizado durante el desarrollo de este TFM. La información de carácter complementario se incluye en los anexos. Esta ordenada en los siguientes capítulos:

Capítulo 1: Se Introduce el proyecto, se presenta el marco del proyecto, se expone los objetivos y la metodología a seguir para el desarrollo del mismo.

Capítulo 2: Muestra el estado de la técnica actual de los sistemas más relevantes para la monitorización de vehículos que están implantados o que se están desarrollando actualmente.

Capítulo 3: Plantea el desarrollo de la prueba de campo realizada para la captura de los datos y el posterior pre-procesado de los mismos para obtener la base de datos.

Capítulo 4: Se realiza la construcción de los primeros clasificadores, con unas *features* básicas, y su posterior evaluación de la eficacia.

Capítulo 5: Análogamente al capítulo anterior, se realiza la construcción de otros clasificadores, en este caso con unas *features* más complejas y su posterior evaluación de eficacia.

Capítulo 6: Se abordan las conclusiones del trabajo realizado, estudiando el grado de cumplimiento de los objetivos del proyecto y planteando líneas futuras de trabajo.

Anexo 1: Se presentan las tablas con los resultados de la reducción de las *features* para los primeros clasificadores

Anexo 2: Similar al anexo anterior, en este se presentan las tablas con los resultados de la reducción de las *features* para los clasificadores ampliados.

CAPÍTULO 2. ESTADO DE LA TÉCNICA

1. INTRODUCCIÓN

Desde la revolución industrial hasta la actualidad, la demanda de bienes de consumo ha aumentado continuamente. Para poder satisfacerla, son necesarios una gran cantidad de transportes, tanto de productos como de personas. Esta situación, unida al ritmo de vida de la población actual, tiene como resultado que se produzcan millones de desplazamientos diarios. Todos estos deben producirse de forma segura y rápida. Por todo ello, desde la aparición del semáforo hasta los modernos sistemas actuales de control de tráfico, esta tecnología ha estado evolucionando continuamente.

Para el diseño y mejora de las vías de comunicación siempre ha sido necesaria la obtención de información estadística sobre el tráfico en las mismas. Esta permite la caracterización del flujo vehículos a lo largo del tiempo, con lo que se pueden planificar mejoras y reducir así los problemas de mala circulación, además de evaluar y mejorar la seguridad. Tradicionalmente, la información disponible solía ser de muy baja calidad, limitándose a contar vehículos, sin tener en cuenta otros parámetros como tipos de vehículos, velocidades o incluso destinos.

En la actualidad los Sistemas de Transporte Inteligentes (Intelligent Transport Systems, ITS) están ganando relevancia. Estos sistemas de gestión de tráfico están basados en la creación de redes de diversos sensores que, distribuidos en las vías, proporcionan una gran cantidad de información sobre el tráfico. Esta es mucho más completa, ya que incorpora información adicional como el tipo de vehículo, la velocidad, el tamaño e incluso, mediante la re-identificación de vehículos, se llegan a obtener las matrices origen-destino, que permiten obtener las trayectorias más probables para los vehículos dentro de la red de vías. El desarrollo de los ITS está siendo considerado de gran importancia por la mayoría de países. Tanto es así que la Unión Europea lo ha incluido como uno de los “retos de la sociedad” dentro de su programa marco Horizonte 2020.

Para el desarrollo de ITS se hace imprescindible la creación de nuevos dispositivos que permitan la detección de vehículos y la obtención de información de los mismos. Aunque ningún dispositivo actual es capaz de proporcionar toda la información necesaria. Cada dispositivo tiene unas características que lo hacen más propicio para la monitorización de determinados parámetros y en determinadas condiciones ambientales.

Los sistemas más comunes actualmente son los bucles inductivos. Como ya se ha comentado, estos sistemas proporcionan una información muy pobre, son caros y presentan una vida útil relativamente corta. Otro de los sistemas más utilizados son las mangueras neumáticas, consisten en colocar sobre la calzada una manguera con un líquido a presión en su interior. Cuando un vehículo circula sobre ella se detecta el incremento de presión que produce. Este sistema también proporciona información pobre y es propenso a las rupturas, además de presentar problemas a la hora de detectar vehículos muy próximos a bajas velocidades.

Sin embargo, dada la gran importancia que se les está dando a la evolución de los ITS, se están estudiando una amplia variedad de nuevos sistemas, que proporcionan una información mucho más rica. El gran aumento en la potencia de los equipos informáticos ha provocado la aparición de los sistemas de gestión de tráfico por reconocimiento visual. Estos permiten la extracción de información sobre el estado de la circulación de las imágenes de video tomadas de la calzada. Proporcionan una información muy

rica, pero necesitan una gran cantidad de potencia de procesamiento y se ven muy afectados por factores climáticos. Otros sistemas que se están desarrollando son la inclusión de nodos sensores en las vías. Estos registran el valor de algunas características físicas (vibraciones en la calzada, variación del campo magnético terrestre, etc...) que luego transmiten a centros de procesamiento, donde se interpretan, para la extracción de información sobre la circulación en las vías. Proporcionan información muy rica y son resistentes a factores climáticos.

2. SISTEMAS DE CLASIFICACIÓN DE VEHÍCULOS

2.1 BUCLES INDUCIVOS

Estos sistemas están basados en la variación de la inductancia de una espira, cuando circula sobre ella un objeto metálico de gran tamaño como sería un vehículo (Fig. 2). Su funcionamiento es el siguiente: se inserta en la calzada una espira de un material conductor, esta forma parte de un circuito eléctrico. Cuando un vehículo cruza por encima, varía su inductancia, el circuito eléctrico puede detectar esta variación y, por tanto, detectar el paso de un vehículo. La información que este sensor en su forma más básica puede proporcionar es muy pobre, solo puede realizar la detección del vehículo pero no puede proporcionar información adicional [3] [4] [5].

(a) (b)

Fig. 2 Bucle Inductivo [6]

En su versión más extendida se instalan colocando 2 espiras separadas una distancia conocida (Fig. 3). Cuando un vehículo es detectado por el primer bucle inductivo se cuenta el tiempo que transcurre hasta que es detectado por el segundo. Así se puede no solo detectar el paso de un vehículo sino que además puede estimar la velocidad y la longitud del mismo. De esta manera la información que pueden proporcionar es mucho más rica, sin embargo son más costosos y tienen una vida útil inferior al sistema de una única espira.

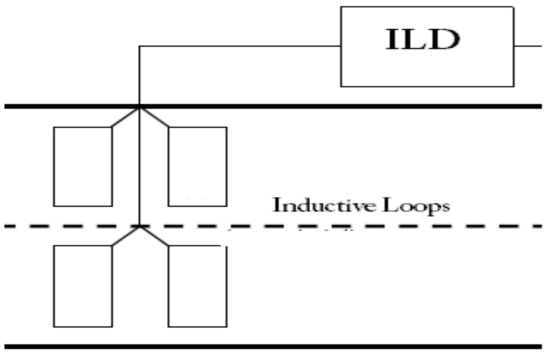

Fig. 3 Bucle inductivo de dos espiras [7]

Basándose en estos sistemas se han propuesto diferentes variaciones a la hora de tratar la información obtenida por un bucle inductivo para extraer la mayor cantidad de datos posible. En algunos trabajos [8] se propone el tratamiento en tiempo real y, mediante la extracción de una serie de parámetros estadísticos (el perfil magnético normalizado, la función de densidad de probabilidad de perfil magnético y perfil magnético transformado al dominio de la longitud del vehículo), permitiría la clasificación de vehículos según su longitud con la utilización de una única espira.

Otros trabajos se han centrado en mejorar la calidad de información que se puede extraer de una única espira. Como en la actualidad se ha producido un gran desarrollo en las redes neuronales, en numerosos trabajos han estudiado la utilización de estas técnicas para la clasificación de los vehículos en distintos grupos, según su tamaño, procesando algunos parámetros mediante redes neuronales.

En su trabajo Y. Ki y D. Baik [9] proponen la utilización de una red neuronal (Artificial Neural Networks o ANN) para procesar los datos obtenidos de una única espira, con el fin de clasificar los vehículos en 5 clases (Vehículo de pasajeros, furgoneta, autobús, camión y motocicleta). Para ello analizan dos parámetros (ratio de variación de la frecuencia y ratio de variación de la forma) de la señal obtenida. Con este método consiguen una eficacia del 91.5%. Además aseguran haber superado los anteriores métodos de J. Gajda (mediante análisis en el dominio del tiempo consiguió una eficacia del 83%)[10] [11] y de C. Sun (mediante mapas auto-organizados consiguió una eficacia del 80%)[3].

En su artículo S. Meta y M.G. Cinsdikici [6], también proponen analizar la información obtenida de una única espira mediante la utilización de una ANN para la clasificación de vehículos en las 5 mismas clases. En este caso proponen realizar la transformada discreta de Fourier (discrete Fourier transform o DFT) a la señal obtenida y después reducir su dimensionalidad mediante un análisis de componentes principales (Principal Component Analysis o PCA). La ANN es entrenada con los 16 primeros parámetros del PCA y con el máximo local de la señal. Con esta técnica han obtenido una eficacia del 94.21%.

Por otro lado, H. A. Oliveira, F. R. Barbosa[7] proponen el análisis de la información obtenida de dos espiras mediante una red neuronal. En su trabajo estudian la posibilidad de la clasificación de vehículos en cuatro grupos (Motocicletas, coches pequeños, medianos y grandes), para ello proponen analizar una serie de parámetros de ambos sensores (Variación máxima, el área debajo de la curva, el tiempo de ocupación, el tiempo de tránsito y el número de máximos locales en el primer sensor). Estos parámetros son analizados por un perceptrón multicapa (multilayer perceptron, MLP) entrenado con Levenberg-Marquardt. (LMA) Consiguiendo así una eficacia del 92.43%.

2.2 SENSORES PIEZOELÉCTRICOS

Otro de los sistemas más utilizados en la actualizada son los sensores piezoeléctricos (Fig. 4). Consisten en una tira de un material que, cuando experimenta una deformación brusca, cambia el potencial eléctrico entre sus extremos. Generalmente se colocan dos de estos sensores transversalmente sobre la calzada, separados una distancia conocida. De este modo, se puede detectar el paso de vehículos y estimar el número de ejes, la velocidad y el tamaño de los mismos. Son muy útiles para detectar vehículos, diferenciar entre tamaños o incluso contar el número de ejes de los camiones. Aunque presentan serias desventajas, como su alto índice de rotura o el error en la detección cuando los vehículos circulan muy cercanos. Otro de los problemas es que, estos sensores solo detectan la variación brusca de la pisada, por lo que si el vehículo se detiene no se puede detectar que sigue ahí.

Fig. 4 Sensor Piezoeléctrico [12]

Para solventar algunos de estos errores, en los sistemas que se instalan actualmente se combinan dos sensores piezoeléctricos y un bucle inductivo (Fig. 5). Su instalación se realiza separando los sensores piezoeléctricos una distancia conocida e insertando entre ellos el bucle inductivo. Esto permite solucionar el problema de que dos vehículos circulen muy cercanos, ya que el bucle inductivo permite saber la continuidad del vehículo. Igualmente se solventa el problema de que un vehículo se detenga sobre el sensor. Estos sistema presentan mejores resultados que los que solo tienen sensores piezoeléctricos o solo bucles inductivos. Sin embargo, son mucho más costosos y tienen más posibilidades de rotura.

Fig. 5 Sensor piezoeléctrico y bucle inductivo [13]

En la actualidad se está trabajando para intentar solventar los problemas que presenta la utilización de un único sensor piezoeléctrico, sin tener que recurrir a sistemas tan complejos como la combinación de sensor piezoeléctrico y bucle inductivo.

Uno de los investigadores que más está trabajando en este tema es S. A. Rajab. En 2012[14] presento un artículo en el que proponía un sistema de clasificación de vehículos con un único sensor piezoeléctrico (Fig. 6). Este se colocaría en la calzada formando un Angulo de 45º. Cuando pasa un vehículo detecta la diferencia de tiempo en pisar el piezoeléctrico entre la rueda derecha e izquierda. Entonces presupone la anchura de los vehículos como la media y puede calcular la velocidad del vehículo y la distancia entre sus ejes. Con estos valores puede realizar una clasificación de los vehículos. Sin embargo, el valor de la distancia entre ejes que el sistema calcula presenta una varianza del 39% dependiendo de que anchura media se considere.

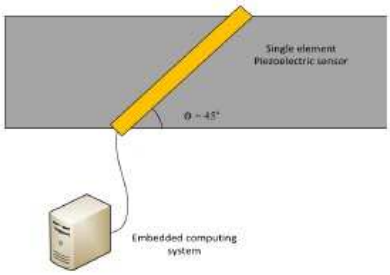

Fig. 6 Esquema del sistema de un sensor piezoeléctrico [14]

El principal problema del método anterior es que es necesario determinar la velocidad para poder calcular la distancia entre ejes, y para poder calcular la velocidad es necesario conocer el ancho del vehículo. El considerar un ancho de vehículo medio, introduce un error considerable. Por eso en 2014 presento otro artículo [15], en este solventaba el problema clasificando los vehículos por su ratio de distancia entre ejes–anchura, solucionando el problema de estimar datos. Con este método divide los ejes en 13 clases dependiendo de este ratio, consiguiendo alcanzar una eficacia del 98.9%.

También en ese mismo año presenta otro artículo[16] en el que propone otro sensor piezoeléctrico distinto, dividido en varios fragmentos independientes(Fig. 7). Esto permite un cálculo del ancho del vehículo, por tanto de la velocidad y la distancia entre los ejes. Con este sistema clasifica los vehículos en las mismas 13 clases que en los otros artículos con una eficacia del 86.9%.

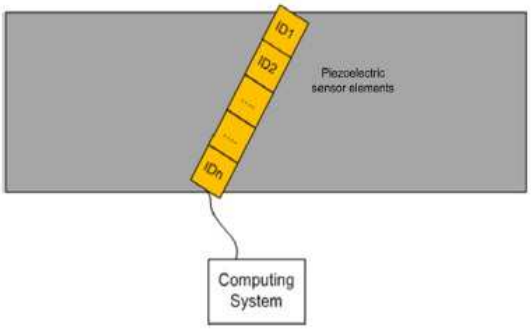

Fig. 7 Sensor piezoeléctrico dividido[16]

Esta idea de utilizar un sensor piezoeléctrico dividido en partes para mejorar el sistema de clasificación ha sido estudiada por varios investigadores. En su artículo P. R. Golla [17] plantea el diseño de un sensor piezoeléctrico dividido en varias partes. Propone un sensor piezoeléctrico dividido en 16 partes con el que se puede calcular la anchura de los vehículos y uno de 49 partes que permite asta calcular la anchura del neumático.

Por otro lado, el sistema actual de dos sensores piezoeléctricos con un bucle inductivo en su interior no se ha descartado todavía. Actualmente se sigue trabajando en conseguir mejorar la calidad de la información que nos proporciona este sensor. En su artículo S-W. Kim[18], propone el tratamiento de la información proporcionada por uno de estos sistemas mediante lógica borrosa. Los vehículos se han clasificado en 11 grupos distintos. Con el sistema actual al realizar esta clasificación se consiguen una eficacia del 87.22%, sin embargo con el nuevo sistema que se propone se consigue alcanzar un 93.44%.

2.3 SISTEMAS DE RECONOCIMIENTO VISUAL DE VEHÍCULOS

Uno de los sistemas utilizados actualmente que mejores resultados proporciona y más se están desarrollando son los sistemas de reconocimiento visual de vehículos (Fig. 8). En la actualidad en la mayoría de vías de tránsito de vehículos existe una red de cámaras de video vigilancia. Estas cámaras permiten a operarios vigilar el estado del tráfico u observar algún problema en las vías como un accidente de automóvil o el bloqueo de la vía. No obstante, esto presenta el problema de que se necesita muchas personas para controlar completamente esta red de cámaras, por ello se han desarrollado los sistemas de reconocimiento visual. Estos permiten obtener de manera autónoma información de esta red de cámaras.

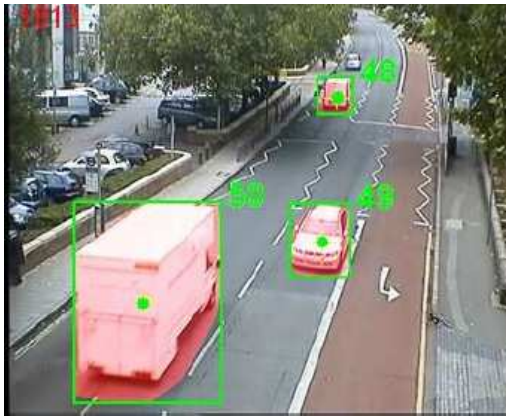

Fig. 8 Sistema de reconocimiento visual de vehículos [19]

Las investigaciones en esta rama están centradas en mejorar la cantidad y calidad de información que se puede extraer de esta red de cámaras de forma autónoma. Se han hecho muchos avances en conseguir detectar los vehículos en una imagen y en clasificarlos según su tipo. También se está consiguiendo la re-identificación de vehículos, que consiste en seguir el tránsito de un vehículo concreto a través de la red de cámaras. Esto permite la obtención de las matrices origen-destino, las cuales se utilizan para predecir el flujo del tráfico en las vías.

En la línea de realizar la clasificación de vehículos se han elaborado gran cantidad de trabajos. La clasificación de vehículos, mediante la utilización de las características extraídas de imágenes, es un trabajo muy adecuado para una red neuronal, por tanto se han realizado múltiples proyectos en los que se las utiliza para esta tarea. En su artículo A. Gepperth [20] plantea la utilización de un MLP de bajo coste de computación consiguiendo una eficacia del 90%. Por otro lado, Z.Chen [21] propone el análisis del histograma de colores de las imágenes mediante una máquina de vectores soporte (Super Vector Machine, SVM) para realizar la detección y clasificación en 3 clases. Z.Dong [22] expone la utilización de un entrenamiento por múltiples núcleos (Multi Kernel Learning, MKL) para analizar las características estructurales de la imagen para poder clasificar en 5 grupos, consiguiendo una eficacia del 91%. Por último, Y. Liu [23] presenta la idea de la estimación de características físicas de los vehículos mediante técnicas de tratamiento de imágenes convencionales, para posteriormente realizar la clasificación en 4 clases utilizando para ello una red dinámica Bayesiana (Dynamic Bayesian Network, DBN), obteniendo una eficacia del 83,75%.

Otros artículos como el de Z.Chen [19] permiten no solo la clasificación de vehículos, realizada con una SVM y un filtro de Kalman, obteniendo una eficacia del 94.69%, sino que además presenta la ventaja de que permite la realización del seguimiento del vehículo dentro de la misma imagen. Esto sería un tipo simple de re-identificación, pero para ser realmente útil se debe de poder realizar en imágenes obtenidas de distintas fuentes. Por ello, en su artículo Y. Shan [24] propone la utilización de un sistema de entrenamiento no supervisado para la discriminación de los bordes. Con este, se puede realizar la re-identificación de vehículos mediante sus características morfológicas. Otra manera es no centrarse en reconocer las características morfológicas de los vehículos, sino en intentar reconocer los números de las matrículas. Centrándose en esta idea W. Wang [25] propone realizar primero una reducción de dimensionalidad mediante un PCA, para después realizar el método de Eigenface para la obtención del número de la matrícula. De esta manera asegura haber conseguido una eficacia en detección de matrículas del 100%.

Otro tipo de imágenes de las que se puede extraer gran cantidad de información son las aéreas. Estas permiten observar gran cantidad de vías, dentro de la misma imagen. La calidad es demasiado baja para poder realizar re-identificación alguna, pero la de detección de vehículos sería muy útil para la estimación de densidad de tráfico. Con esta idea F. Yamazaki [26] propone en su artículo un sistema de detección de vehículos mediante imágenes aéreas, con él puede detectar los vehículos e incluso estimar la velocidad que llevan con una eficacia del 67%.

Otro tipo de sensor, que no son estrictamente hablando un sistema de reconocimiento visual, son los sensores láseres de rango. Estos funcionan realizando un barrido con un haz láser y midiendo el tiempo de retorno para calcular a que distancia se encuentra el fondo. En su artículo C. Harlow[27] propone la utilización de este tipo de sensor para la clasificación de vehículos conforme su longitud. Además también permiten realizar una estimación de su velocidad. Este sistema presenta una eficacia del 92%.

2.4 NODOS CON ACELERÓMETRO

El desarrollo que se ha producido en las tecnologías de comunicación inalámbrica y en los equipos autónomos, unido la evolución de los sensores, cada vez con menor tamaño y mejor precisión, ha propiciado la aparición de un nuevo tipo de sistema de monitorización del tráfico. Estos nuevos sistemas consisten en la inclusión en la vía o sus alrededores de nodos autónomos que miden algún parámetro físico. Los nodos tienen comunicación entre ellos o con un sistema central que realiza el procesamiento

conjunto de todos los datos, permitiendo la creación de forma más sencilla y económica de una red de sensores.

El tipo de estos nodos dependen del parámetro o parámetros físicos que miden. Uno de los más útiles para caracterizar el tráfico son las vibraciones producidas sobre la calzada por los vehículos. La medida de estas vibraciones se realiza mediante un acelerómetro. Al igual que los sensores piezoeléctricos, estos sensores aprovechan la interacción de los vehículos con la calzada. Sin embargo, los acelerómetros ofrecen una información más rica, sin los problemas de tener un índice de rotura tan alto. Estos nodos se insertan en la calzada (Fig. 9) de manera que sufren muchas menos roturas que los sensores piezoeléctricos comunes que tienen que estar sobre la misma.

Fig. 9 Instalación de un nodo con acelerómetro [28]

Este tipo de tecnología presenta unos resultados muy prometedores y actualmente se está trabajando mucho en este campo. Como la base de este tecnología es la interacción de los vehículos con la calzada, muchos investigadores se están centrando en detectar los ejes de los vehículos y a partir de ellos extraer la velocidad y la longitud. En su artículo R. Hostettler[29] propone la utilización de varios sensores de este tipo colocados en los extremos de ambos lados de la calzada(Fig. 10). Con ello consigue la detección de vehículos, distinguiendo entre el carril por el que circula, además de la estimación de su velocidad y longitud, consiguiendo una eficacia del 87%.

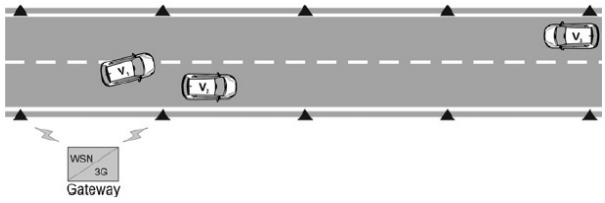

Fig. 10 Esquema del sistema[29]

Con una idea similar D. Obertov[30] propone un clasificador de vehículos, mediante el análisis de la distancia entre los ejes y la energía entregada a la calzada. Mediante un filtrado de la señal y una detección de picos, puede realizar la detección de los ejes de los vehículos. Posteriormente, mediante su algoritmo propuesto analiza esta detección, con la energía entregada por el vehículo al asfalto, para clasificarlos en 4 grupos (coche, coche con carro, camión y camión con carro) consiguiendo una eficacia del 89%.

Otra línea de trabajo, dentro de este campo, es la de no realizar la detección de los ejes, sino la de tratar la huella de vibración completa. Con esta idea M. Stocker[31], propone de forma teórica un análisis de la huella de aceleraciones mediante un sistema de aprendizaje automático ML.

Siguiendo con esa misma idea J. Lan[32], plantea el tratamiento de la huella de aceleraciones completa, mediante una NN entrenada con una BP. Con este método consigue la distinción entre vehículos diésel y gasolina, obteniendo una eficacia del 91%.

2.5 NODOS CON SENSOR MAGNÉTICO

Otro de los parámetros físicos que se suelen medir las redes de sensores es la deformación del campo magnético terrestre. El campo magnético terrestre es prácticamente constante y uniforme sobre la superficie de la tierra, pero en presencia de un objeto metálico de gran tamaño, como sería un vehículo, este campo magnético se deforma de manera notable (Fig. 11). Esta deformación produce que el flujo magnético aumente en las zonas de mayor concentración metálica, y disminuya en las zonas adyacentes. Los sistemas con sensor magnético se aprovechan de esta característica. Midiendo esta deformación no solo se consigue detectar el vehículo, sino que también se obtiene una huella magnética que es dependiente de la cantidad de material metálico del vehículo y de su disposición geométrica.

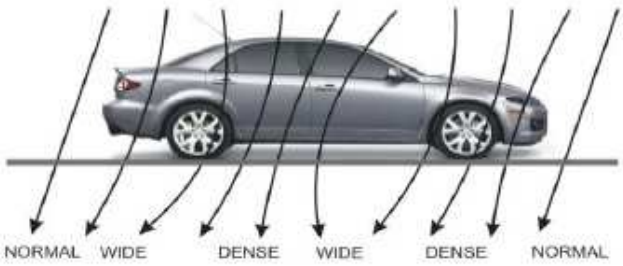

Fig. 11 Deformación del campo magnético terrestre[33]

Estos sistemas se suelen integrar en las vías en forma de red de sensores autónomos. La información extraída de estas redes permite realizar la detección y clasificación de los vehículos. Además, con varios sensores también se puede realizar la estimación de la velocidad y la longitud. Se aprovechan de un principio físico similar al de los bucles inductivos, no obstante proporcionan información más rica, siendo también menos costosos y con una vida útil más larga.

La clasificación de los vehículos mediante la utilización de un único sensor magnético supone una gran simplificación del sistema y una reducción de costes. Por ello, varios autores se centran en el desarrollo de un método eficiente que permita la clasificación de los vehículos mediante un único sensor magnético. Con esta idea G.V. Prateek [34], presenta de manera teórica un sistema de clasificación utilizando una SVM. En su trabajo expone 3 maneras distintas de pre-tratar los datos para su interpretación con las SVM: Average-Bar Transform, la transformación de Hill-Pattern y el modelo de Dipolo magnético. Finalmente se propone la utilización del modelo de Dipolo magnético. El Average-Bar Transform consiste en dividir la señal en tramos y calcular la media en cada tramo, esta media es el parámetro de entrada de la red. La transformación de Hill-Pattern consiste en contar el número de picos de Hill-Pattern, con la peculiaridad de que para que un punto se considere pico tiene que tener cierto número de puntos en aumento a la izquierda y cierto número de puntos en descenso a la derecha. Por último, el modelo de Dipolo magnético consiste en modelizar la huella del vehículo como un vector de dipolos magnéticos. La fuerza de estos dipolos y la separación entre ellos son los parámetros utilizados para la clasificación.

Siguiendo con la idea de un único sensor magnético, Z. Feng[35] propone la utilización de 4 SVM anidados para la clasificación de los vehículos en 5 clases. Este sistema utilizaría como datos de entrada longitud de la señal, la media, la varianza, los picos y los valles. Con esto se consigue una eficacia del 84,37%.

Sin embargo dada la facilidad que presentan estos nodos para formar una red y las múltiples ventajas que ofrecen, también exististe una gran investigación en sistemas de varios nodos. En su trabajo G. Burresi[33], plantea la utilización de dos nodos separados a una distancia conocida para realizar la clasificación de vehículos. Mediante un emparejamiento de las señales por alineamiento temporal dinámico (Dynamic Time Warping, DTW) se consigue una estimación de la longitud del vehículo. Con ese valor y la media en 5 tramos se entrena un MLP que consigue clasificar los vehículos en 4 clases con una eficacia del 91%.

También con la idea de una red de varios sensores W. Zhang[36], propone la creación de una red de proximidad binaria de vehículos. La red de sensores de proximidad binaria es una red especial. Todos los sensores que componen la red tienen una coordenada definida y un rango de detección finito. Estos solo comunican un bit de información, vehículo detectado o no. Con estos datos se calcula la longitud del vehículo y su velocidad. Esta información unida a la propia huella magnética del vehículo permite entrenar una NN con BP para conseguir un clasificador en 5 clases con una eficacia del 93,61%.

Otra idea de aplicación de estas redes es una red de detección de vehículos portátil. Consistiría en un conjunto de nodos portátiles que se instalarían junto a una calzada (en vez de incrustados en el pavimento) y que realizarían la clasificación de los vehículos. S.Taghvaeeyan[37] propone un sistema de estas características. Estaría compuesto por 4 sensores portátiles que se colocarían según la configuración de la imagen (Fig. 12). El emparejamiento de la señal del nodo 1 con el 4 permite obtener la altura del vehículo, y el emparejamiento de la señal obtenida de los nodos 2 y 3 permite obtener la longitud del vehículo. Estos datos se introducen en una SVM que permite la clasificación de los vehículos con una eficacia del 83% y una estimación de la velocidad con un error del 2.5%

Fig. 12 Sistema de detección portátil [37]

Un autor que ha realizado varios artículos sobre este tipo de sensores es S. Kaewkamnerd. En un primer artículo [38] propone un sistema de clasificación de un único sensor que utiliza como parámetros la energía media, número de picos de Hill-pattern y la longitud del vehículo, aunque esta no se obtiene mediante el sensor magnético. Con este, alcanza una eficacia del 80.92%. Posteriormente presenta otro artículo [39] en el sí realiza la detección de la longitud mediante 2 sensores magnéticos, consiguiendo en este caso una eficacia del 77,69%. Por último, presenta un artículo [40] en el que propone un único nodo que integra 2 sensores magnéticos. Con este nuevo sistema analiza los parámetros de los picos de Hill-pattern, los parámetros de la energía diferencial del vehículo normalizada, la velocidad y la longitud. Consiguiendo una eficacia del 81.69%.

Otra de las aplicaciones de estos sensores es la re-identificación de los vehículos. Al igual que con los sistemas de reconocimiento visual, mediante el emparejamiento de las huellas obtenidas se puede realizar la re-identificación. Con esta idea S. Charbonnier[41] propone realizar un emparejamiento de las señales mediante DTW, con lo que permite realizar una re-identificación con una eficacia del 80%.

Por último, dada la facilidad de incluir dentro del mismo nodo más de un sensor, existe un tipo de sistema que combina análisis de las perturbaciones del campo magnético y vibraciones inducidas en la calzada. La integración de ambos sensores en el mismo nodo es sencilla y mejora los resultados. Con esta idea D. Kleyko[42], estudia los resultados obtenidos al analizar estos datos con 3 clasificadores distintos, Regresión Lógica (Logistic Regression, RL), ANN y SVM. El sistema que mejor eficacia proporciona es la RL con una eficacia del 93.4%.

3. CONCLUSIONES

El presente estado del arte presenta los sistemas más relevantes para la monitorización de vehículos que actualmente están implantados así como las distintas líneas de investigación en este campo. Del estudio de este estado de la técnica se puede extraer la conclusión de que se desconoce a priori cuál es el sistema más eficaz para la clasificación de vehículos. Todos los autores proporcionan la eficacia que obtienen con sus técnicas propuestas, sin embargo, cada autor clasifica los vehículos en distintos números de grupos o utiliza una base de datos con diferente número de vehículos, además, en la mayoría de los casos, no especifican si para calcular la eficacia utilizan los mismos vehículos que para entrenar los clasificadores. Todo ello hace imposible decidir de forma inequívoca el sistema más eficaz para la clasificación de vehículos. Sin embargo, con vista a los buenos resultados que aportan tanto para este tipo de problemas como para otros similares, se opta por utilizar clasificadores basados en redes neuronales.

Para el desarrollo de este T.F.M. se utilizan, como magnitudes físicas para la clasificación, la huella magnética y las vibraciones inducidas en el asfalto. Esta decisión, junto al sistema utilizado, viene impuesta por el marco del proyecto. El dispositivo, desarrollado en mi T.F.G. [41], está basado en un WSN. Se opta por un sistema de este tipo porque, su reducido coste y la calidad de la información que proporciona, unido a su facilidad para la creación de redes de sensores inteligentes, lo convierten en un probable candidato para ser uno de los sistemas más ampliamente implantados en el futuro.

CAPÍTULO 3. BASE DE DATOS

1. INTRODUCCIÓN

En el siguiente capítulo se la preparación de las pruebas de campo realizadas para la obtención de las medidas que posteriormente permitirán la creación de la base de datos. Para la captura de los datos se utiliza un prototipo que permite detectar el valor del campo magnético y de las vibraciones. Este prototipo de trata de un WSN equipado con un acelerómetro y un sensor magnético que permite la captura de las medidas de forma autónoma y su transmisión de manera inalámbrica. La toma de medidas se realiza en la entrada de un aparcamiento en el que se ha habilitado la infraestructura necesaria.

2. DESCRIPCIÓN DEL EXPERIMENTO

2.1 OBJETIVOS

El objetivo principal de esta prueba es la captura de la huella magnética y de las vibraciones inducidas en el asfalto, producidas por varios vehículos, con el fin último de elaborar una base de datos que permita el desarrollo de algoritmos de detección y de clasificación de vehículos.

Para la consecución de este objetivo se definen los siguientes sub-objetivos:

- Capturar de forma periódica el valor del campo magnético en los 3 ejes.
- Capturar de forma periódica el valor de las vibraciones en los 3 ejes.
- Identificar el vehículo que las produce y sus características principales para relacionarlas con los datos de los sensores.

2.2 DEFINICIÓN DEL SETUP

Para la elaboración de esta experiencia se coloca un nodo sensor en la entrada de un parking. Gracias a él se capturan las medidas de la huella magnética y las vibraciones de los vehículos cuando acceden o abandonan el parking. Adicionalmente se toman imágenes de los vehículos a su paso sobre el mismo.

El nodo que se utiliza para el experimento es el prototipo desarrollado en mi T.F.G.[1]. Está compuesto por un sensor magnético ARM de 3 ejes, un acelerómetro de 3 ejes y como comunicación un módulo ZigBee. Este nodo muestrea los sensores a 100Hz, y envía las medidas por ZigBee en paquetes de 4 muestras. Estos paquetes son enviados al equipo de recepción, compuesto por un módulo de comunicaciones ZigBee conectado a un ordenador portátil. El ordenador almacena un streaming continuo de las medidas de los sensores durante toda la prueba. En el ANEXO 3 se muestra más información sobre el dispositivo utilizado.

El entorno donde se realiza la experiencia es el aparcamiento de la entrada del Instituto de Investigación en Ingeniería de Aragón (I3A), el cual dispone de 80 plazas. Se toma la decisión de este emplazamiento por proximidad y por tratarse de un aparcamiento cerrado con un único punto de entrada y salida, lo que asegura que todos los vehículos que entran y salen deban de pasar por este punto.

En la figura siguiente (Fig. 13) se observa el emplazamiento de la prueba. Se dispone de un único punto de entrada/salida con una anchura que solo permite el paso de un único vehículo. Los laterales de este camino están delimitados por unos bolardos de hormigón. El nodo se coloca en el punto de entrada al parking, bajo el asfalto en una caja estanca. El equipo de recepción se posiciona en un lateral de la entrada, suficientemente próximo para asegurar la buena comunicación. Por último, se instala una cámara, monitorizando la prueba, en un lateral.

Fig. 13 Esquema del emplazamiento

El procedimiento del experimento es el siguiente. Se realizan una serie de medidas, estas son tomadas en las dos franjas del día de mayor tráfico en el aparcamiento, la franja de apertura (7:30 a 9:00) y la franja de salida (13:00 a 14:30). Durante las experimentaciones se capturan las medidas del nodo de forma periódica y se monitoriza la prueba. Adicionalmente se toman nota de manera manual de todos los vehículos, así como de la hora y de posibles anomalías ocurridas. Con esta información, posteriormente en la etapa de procesamiento de los datos, se establece la correlación de las medidas obtenidas con las principales características de los vehículos que se observan en la grabación. En la figura siguiente (Fig. 14) se puede observar una imagen de las experimentaciones:

Fig. 14 Setup de la experimentación.

3. PROCESAMIENTO DE LOS DATOS

3.1 PROCESAMIENTO MANUAL DE LOS DATOS

Una vez finalizadas todas las experimentaciones, como resultado de ellas, se logra obtener un fichero de texto plano por cada una, en los que aparecen las medidas de los sensores muestreados de forma periódica (cada 10 milisegundos) durante toda la prueba. Además, se consigue una grabación completa de cada prueba, así como una hoja de notas, tomadas manualmente.

Para la elaboración de la base de datos se realiza un pre-procesamiento manual de los mismos. Para ello se elabora un fichero por vehículo en el que se especifica:

- Numero de vehículo detectado en la prueba
- Instante de paso por el nodo (referido al vídeo correspondiente)
- Instante de fin del paso por el nodo
- Dirección del vehículo
- Longitud del vehículo
- Peso del vehículo
- Tipo de vehículo
- Marca del vehículo
- Modelo del vehículo

Los parámetros característicos del vehículo se obtienen identificando, de forma manual, la marca, modelo y año del vehículo, mediante las imágenes grabadas, al igual que la dirección. Para la identificación de los instantes de inicio y final, se sincronizan los datos de los sensores con las imágenes. Esta sincronización se realiza mediante el golpeo del sensor al inicio de cada prueba, lo que permite una correlación temporal de los datos de los sensores con las imágenes grabadas. De esta manera, se consigue delimitar los instantes de inicio y final de detección, observando el momento en el que el vehículo pasa por el mismo.

Tras este primer pre-procesado se ha obtenido una base de datos de 113 vehículos, que posee para cada uno de ellos las huellas, tanto magnéticas como de vibración inducida en el asfalto (Fig. 15), además de un fichero con la información del mismo.

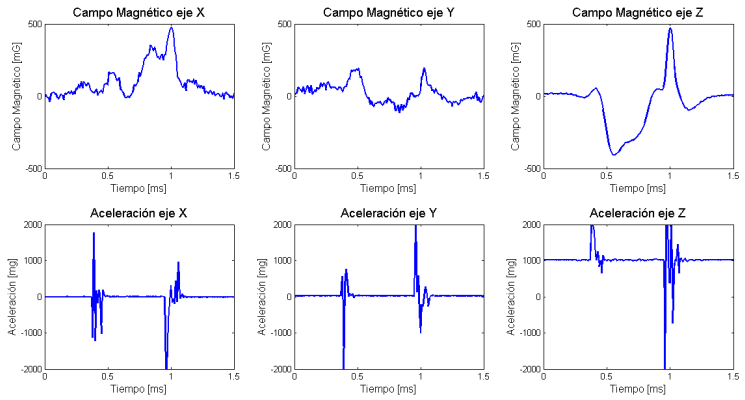

Fig. 15 Ejemplo de la captura de las huellas magnéticas y de aceleración de un vehículo

3.2 DETECCIÓN AUTOMÁTICA DE VEHÍCULOS

Para poder realizar un clasificador es necesario el poder capturar de manera automática las huellas de los vehículos. Por tanto, el primer paso es el desarrollo de un algoritmo que permita la detección y captura de las huellas de los vehículos en un streaming de datos continuos.

Este algoritmo funciona sobre el módulo de la huella magnética, basándose en un umbral de detección y en una continuidad mínima de la señal. El funcionamiento es el siguiente, en primer lugar, se detecta el inicio de un vehículo cuando se supera el umbral de 40mG durante al menos 10 muestras seguidas. Después, se identifica el final del vehículo cuando desciende del umbral de detección durante, al menos, otras 10 muestras seguidas.

Con este algoritmo, de los 113 vehículos que forman la base de datos, se detectan correctamente el 100%. Además, si se compara gráficamente los puntos de inicio y final de las detecciones calculadas, tanto manualmente como automáticamente (Fig. 16), se puede observar que el método automático realiza la captura de la huella de manera más correcta, ya que el método manual tiende a perder parte de la información. Esta pérdida de información, en el pre-procesado manual está provocada, en parte por la dificultad de decir el momento exacto del inicio y final de una detección a partir de las imágenes obtenidas. También está provocada porque el campo magnético se empieza a distorsionar un poco antes del paso del vehículo y sigue distorsionado un poco después del mismo.

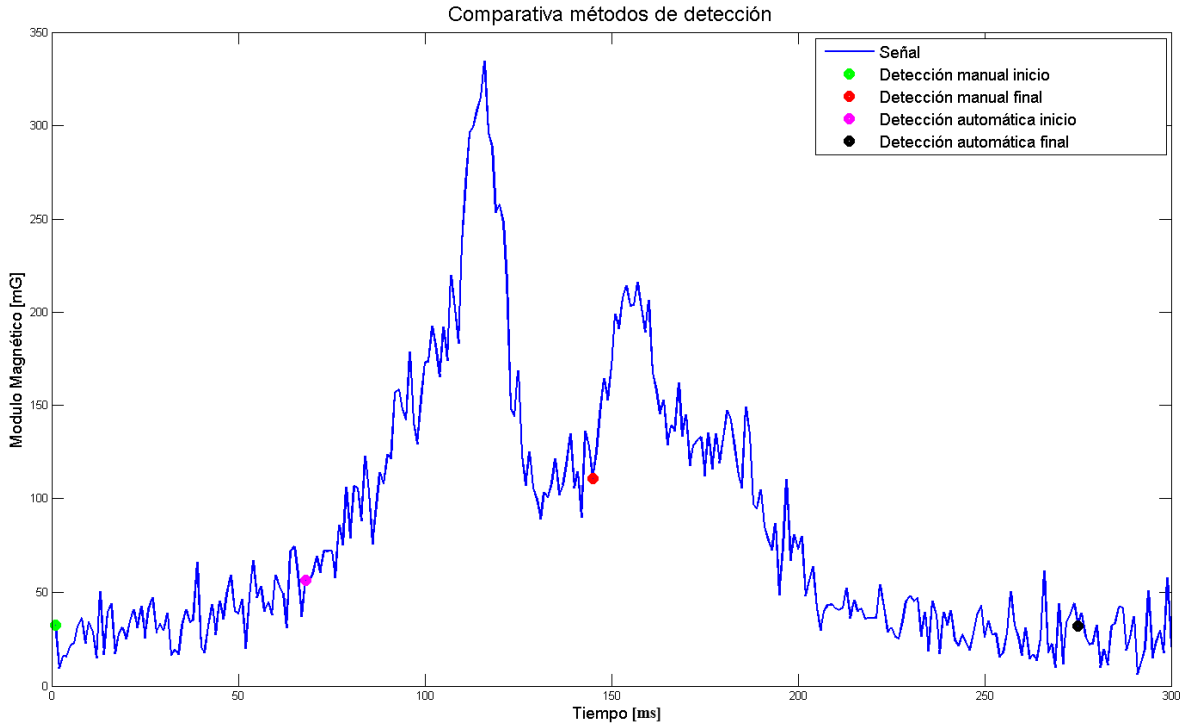

Fig. 16 Comparativa métodos de detección

4. CONCLUSIONES

Como resultado final de esta experimentación, se logra una base de datos con 113 vehículos, obteniendo para cada uno de ellos la siguiente información (Fig. 17). La base de datos está compuesta por 80 vehículos con dirección de entrada al parking y 33 de salida. Atendiendo al tamaño del vehículo, está compuesta por 37 mayores de 4,5 metros de longitud y 76 menores.

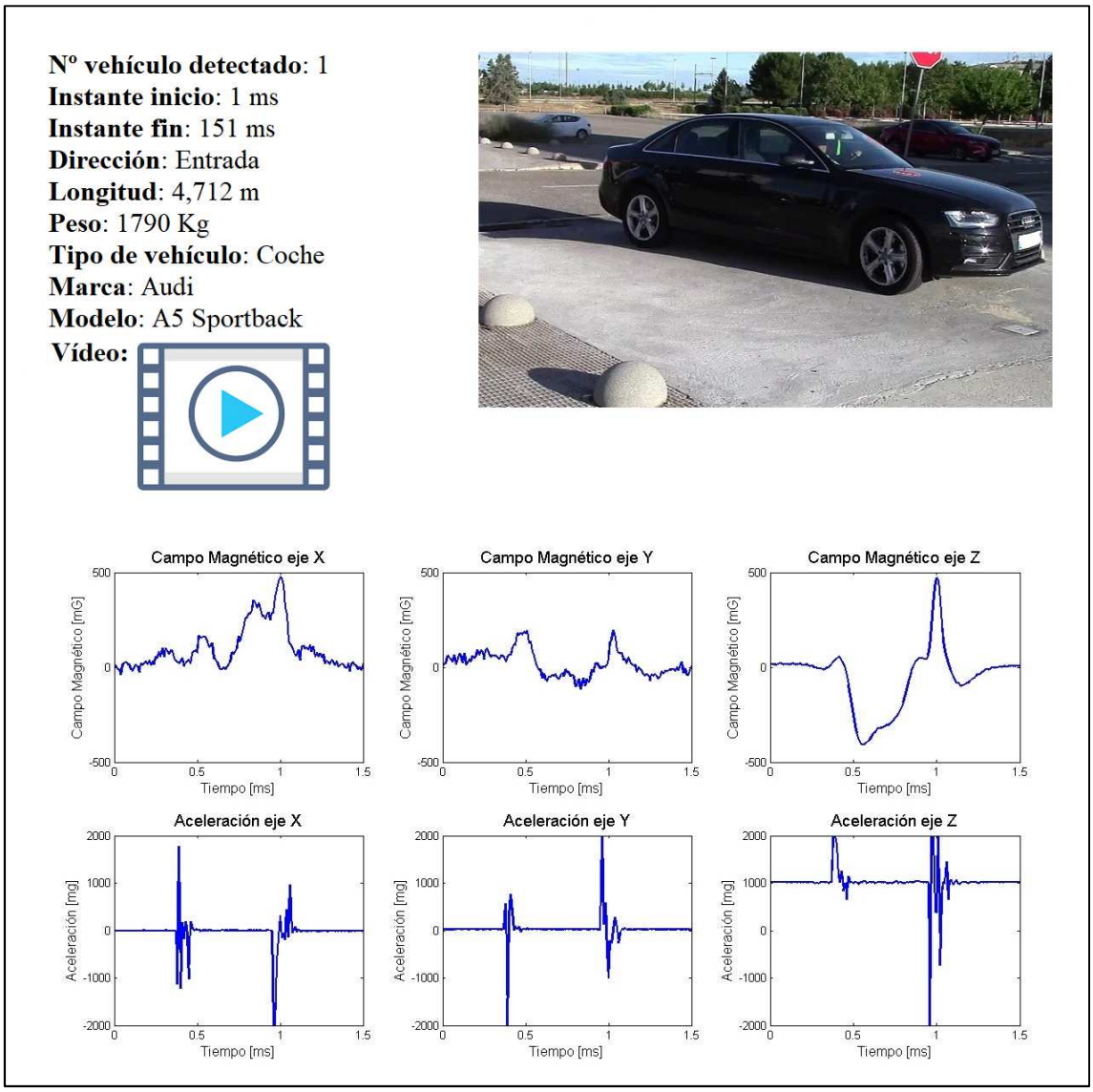

Fig. 17 Ficha de detección

CAPÍTULO 4. CLASIFICADOR

1. INTRODUCCIÓN

El siguiente capítulo recoge el proceso de creación de los diferentes clasificadores y su posterior análisis. El objetivo de este capítulo es la construcción de un clasificador de vehículos por dirección, diferenciando entre entrada o salida del parking y otro por tamaño, diferenciando entre mayores y menores de 4,5 metros. Se va a construir un MLP por cada tipo de clasificación.

El proceso de creación es el siguiente: en primer lugar, se seleccionan unas *features* candidatas para los clasificadores en base a los resultados obtenidos del estudio del estado del arte. Después se estudian que combinación de *features* es más efectiva para cada uno. Posteriormente se elige la mejor arquitectura para cada clasificador y finalmente se evalúa la efectividad de los mismos.

2. TÉCNICAS UTILIZADAS

Antes de proceder a la construcción de los clasificadores se expone las diferentes técnicas que se utilizan durante el diseño de los mismos. Como se menciona en el CAPÍTULO 2 en la sección 3 , se opta por utilizar técnicas basadas en Redes Neuronales Artificiales (Artificial Neural Networks o ANN). Las ANN son sistemas que imitan la estructura neuronal del cerebro, son de gran utilidad para el procesado de datos no lineales y reconocimientos de patrones, debido a su operación no lineal y su capacidad de aprendizaje a partir de muestras.

Los Mapas Auto-Organizados (Self-Organizing Maps o SOM)[43] son un sistema de clasificación de patrones basados en ANN. Los SOM son un tipo de ANN que se encuentran dentro del grupo de redes neuronales no supervisadas. Estas son especialmente útiles para la realización de análisis exploratorio y visualización de bases de datos, debido a su rapidez de entrenamiento y su capacidad de proyectar datos multidimensionales sobre un mapa bidimensional.

Otro tipo de ANN son los perceptrones multicapa (Multilayer Perceptron o MLP) [43] se encuentran dentro del grupo de redes neuronales supervisadas. Este modelo de ANN es uno de los más utilizados por tratarse de un aproximador universal de funciones. Puede aprender relaciones no lineales con lo que tiene grandes aplicaciones tanto de ajuste de función como de clasificación. Está compuesto por una capa de entrada, con una neurona por cada variable de entrada, una capa de salida con una neurona por cada clase de salida posible y entre ambas capas, una o varias capas de procesamiento.

Para validar la eficacia de los clasificadores se utiliza el índice Kappa (Cohen's Kappa Index)[44]. Este indicador es más objetivo que el porcentaje de acierto global, ya que castiga las probabilidades a priori que se tiene de acertar. Se calcula a partir del matiz de confusión, una matriz en la que las filas son las clases asignadas y las columnas las reales. El valor de Kappa que se considera "bueno" varía en función del autor[43]. Por ejemplo Landis & Koch [45] califica $0.60<Kappa<0.81$ como "sustancialmente bueno" y $Kappa>0.80$ como "casi perfecto", sin embargo Fleiss, Levin, & Cho Paik[46] califica un $Kappa>0.75$ como "excelente" y $0.4<Kappa<0.75$ como "resultado medio a bueno".

Un método de validación específico para casos como el actual, en el que la base de datos es reducida, es la validación cruzada (Cross Validation o CV) [44]. Este permite sacarle el mayor partido posible a una base de datos, por lo que es muy utilizado en caso de bases de datos pequeñas. El proceso es el siguiente:

se divide la base de datos en un número determinado de partes, siendo las partes aleatorias y homogéneas. Posteriormente se entrena el MLP con toda la base de datos menos una de estar partes y se utiliza esta parte para evaluar el clasificador, obteniendo así su matriz de confusión. Se repite el mismo proceso con todas las partes y se suma todas las matrices de confusión, obteniendo así la matriz de confusión total. Con esta matriz de confusión se puede calcular el índice Kappa del clasificador.

Otra técnica utilizada en este proyecto son los Algoritmos Genéticos (AG) [44]. Los AG son un método de búsqueda de optimización basado en los principios del Neo-Darwinismo, son de gran utilidad cuando se quiere buscar una solución óptima pero las combinaciones posibles son demasiado grandes como para comprobarlas todas. El funcionamiento es el siguiente, se define un genoma en el que se expresa todos los parámetros modificables del problema. Con este genoma se genera una población inicial aleatoria, esta población se analiza mediante una función de idoneidad que permite obtener una medida de la misma, para cada Genotipo. Posteriormente se produce la siguiente generación mediante, replicación, reproducción y mutación de los mejores. Esto produce que cada generación es más idónea que la anterior, cuando se repite el proceso un número suficiente de veces, el individuo ganador es el más o uno de los más óptimos de todas las combinaciones posibles del problema.

Un método que se utiliza en este proyecto para la reducción de dimensionalidad es el análisis de componentes principales (Principal Component Analysis o PCA). Es una técnica que permite realizar un análisis exploratorio de una base de datos[44]. Esta técnica calcula los vectores propios, que consisten en una transformación lineal ortogonal de la matriz de covarianza del conjunto de datos. Esta transformación convierte mediante una proyección lineal, los datos originales a un nuevo sistema de coordenadas, en el que las nuevas variables son combinaciones de las originales. Estas nuevas variables se ordenan por su peso estadístico, de manera que se toman las componentes con mayor varianza, conocidas como componentes principales. Tomando un número de componentes principales menor al número de variables originales, se consigue mantener un porcentaje alto de la cantidad de información. Sin embargo, este método adolece del siguiente problema, presupone que la información útil de la base de datos se encuentra en los componentes de mayor varianza, pero esto no tiene por qué ser siempre cierto.

3. FEATURES INICIALES

Para la implementación de la clasificación de los vehículos se debe de realizar en primer lugar un procesado de las señales obtenidas en la experimentación. Por ello, el primer paso en el diseño es la elección de una serie de *features* que se calculan sobre las señales obtenidas. La utilización de estas simplifica el diseño, al reducir la cantidad de parámetros de entrada. Basándose en las conclusiones obtenidas en el estado del arte, se seleccionan 5 candidatas iniciales. Cada una de estas se calcula sobre los 3 ejes de cada uno de los sensores, por lo que se producen un total de 30 *features*.

Estas son las siguientes:

- **Máximo absoluto de la señal:** Se trata del valor más alto que presenta la señal dentro de la muestra (Fig. 18). Su cálculo se ha realizado mediante la función propia de Matlab max().

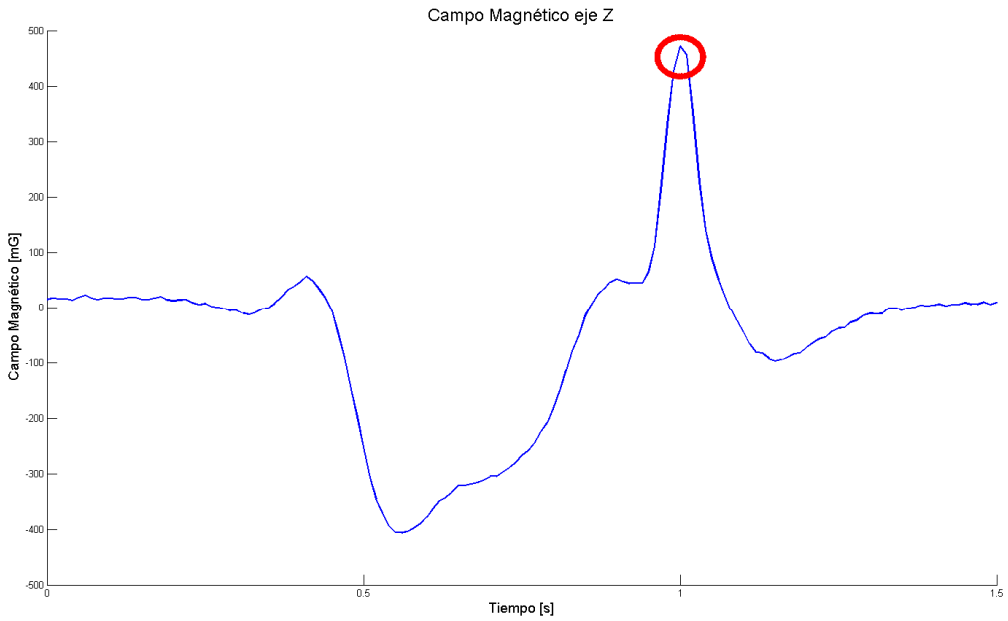

Fig. 18 Máximo

- **Mínimo absoluto de la señal:** Se trata del valor más bajo que presenta la señal dentro de la muestra (Fig. 19). Su cálculo se ha realizado mediante la función que se puede observar en el ANEXO 4 sección 1.

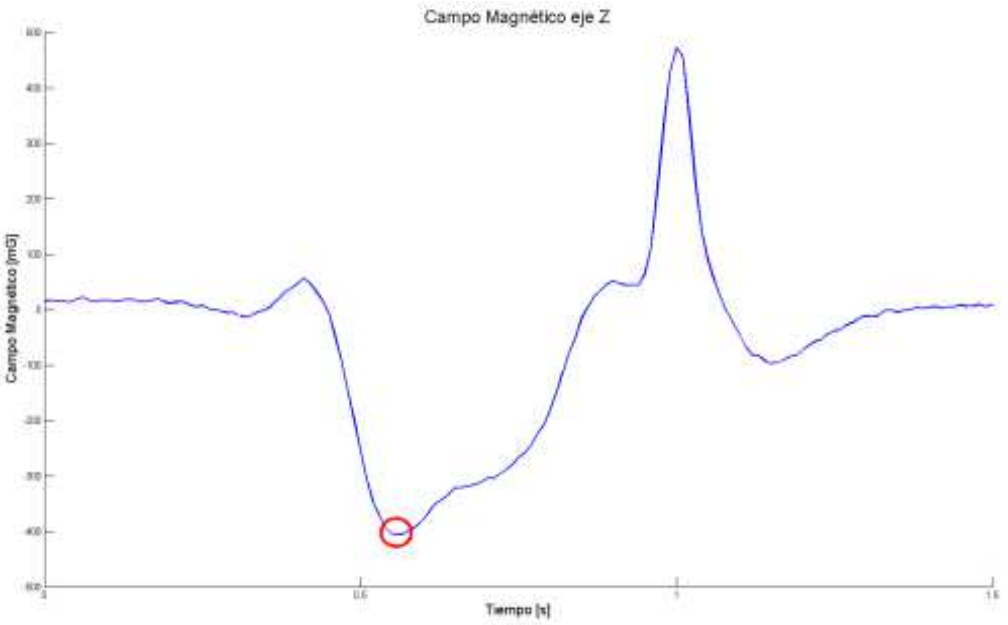

Fig. 19 Mínimo

- **Media de la señal:** Es la media del valor de todos los puntos que conforman la señal (Fig. 20). Su cálculo se ha realizado mediante la función que se puede observar en el ANEXO 4 sección 2.

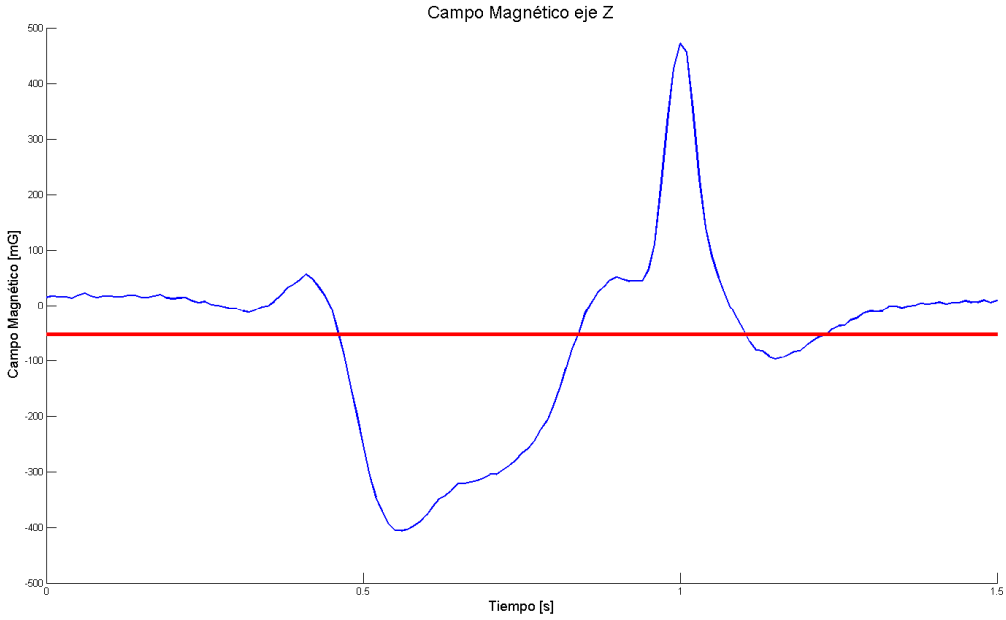

Fig. 20 Media

- **Los picos de Hill-Pattern:** [39]Se trata de un parámetro de la magnitud de la señal en función de su pendiente. Para su cálculo, se obtiene si la pendiente es positiva o negativa en cada punto. Si un punto tiene más de un número determinado de puntos con pendiente positiva a la izquierda y con pendiente negativa a la derecha se considera un pico. Este parámetro aporta información sobre los distintos tipos de estructuras y el número de ejes que conforman el vehículo (Fig. 21). Su cálculo se ha realizado mediante la función que se puede observar en el ANEXO 4 sección 3.

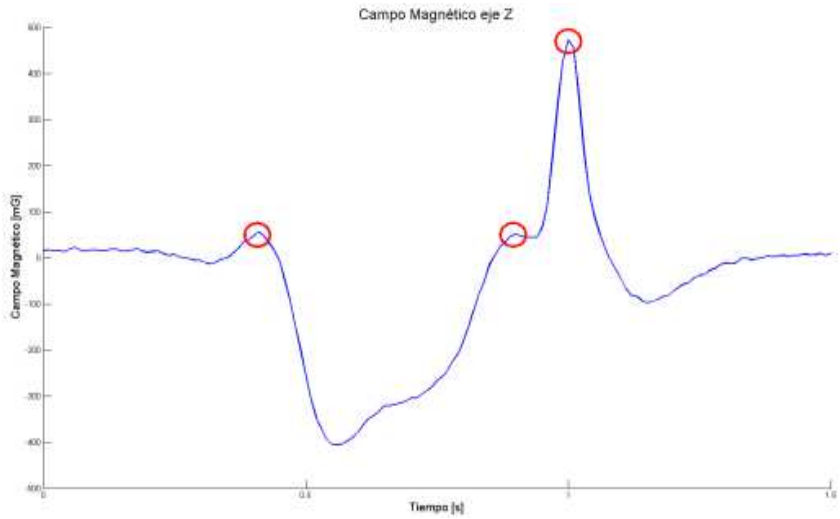

Fig. 21 Picos de Hill-Pattern

- **Average-Bar Transform:** [34]Se trata de un parámetro de la morfología de la señal, que reduce la cantidad de datos a tratar. Su cálculo consiste en dividir la señal en un número determinado de partes y calcular la media para cada una de esas partes (Fig. 22). Su cálculo se ha realizado mediante la función que se puede observar en el ANEXO 4 sección 4.

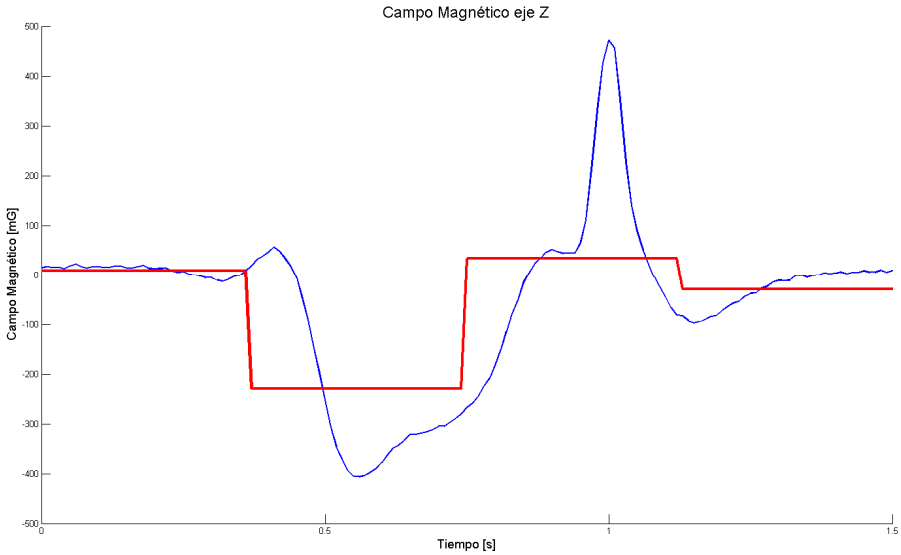

Fig. 22 Average-Bar Transform

4. REDUCCIÓN DE FEATURES

Una vez decididas las *features* iniciales, el siguiente paso es seleccionar cuales, de estas, son relevantes para cada clasificación. El objetivo de este apartado es obtener para cada uno de los dos clasificadores, las combinaciones más eficaces para la clasificación, estas combinaciones se obtienen mediante una combinación de algoritmos genéticos con un análisis mediante SOM.

Lo primero que se realiza es definir el genoma que se utiliza en el AG. Este genoma está formado por una variable binaria por cada *feature*, que especifica si se utiliza o no. Como cada *feature* especificada en el apartado anterior puede ser utilizada por cada uno de los 3 ejes de cada uno de los 2 sensores (6 ejes en total) se tiene un total de 30 variables binarias por esta parte. Además, se incluye una variable binaria más para especificar si se utiliza la longitud de la muestra tomada y 2 variables más, de tipo numérico para especificar el número de muestras que conformara cada media en barra y otra para especificar el número de muestras menores que debe de tener un punto por delante y por detrás para considerarse un pico de Hill-Pattern. Por tanto, el genoma a utilizar en el desarrollo del AG tiene 31 variables binarias y 2 numéricas.

Para calcular la función de idoneidad primeramente se divide la base de datos en 10 partes homogéneas. Después se entrena un SOM por cada genotipo, se opta por este tipo de ANN por su rapidez de entrenamiento. Este SOM se define de un tamaño de 13 por 8 neuronas y se entrena con 9 de las 10 partes de la base de datos. Una vez entrenado se procede a su etiquetado, para ello se le vuelve a presentar a la red SOM las 9 partes de la base de datos y se etiqueta la neurona ganadora (Best Matching Unit o BMU), en caso de que un neurona gane para más de un tipo se etiqueta por la clase más votada. Posteriormente se evalúa, con la parte no utilizada, la base de datos con la red etiquetada obteniendo así la matriz de confusión y con ella se calcula el índice Kappa, este índice es el valor de idoneidad que se utiliza en el

AG. Es importante aclarar que, en este caso, este índice no es un indicador de la eficacia del clasificador final, solo nos sirve para realizar una comparación de eficacia entre las distintas combinaciones.

Para el AG como generación inicial se crean los 100 primeros genotipos aleatoriamente. Estos, después de calcularse su idoneidad (índice Kappa) y clasificarse de mejor a peor, se utilizan para el desarrollo de la siguiente generación de la esta manera: Los 20 mejores pasan a la siguiente generación sin modificación. Otros 60 se generan mediante reproducción, es decir cortando el genoma por un punto aleatorio y cogiendo cada parte del genotipo de un individuo distinto. La posibilidad de que un individuo se utilice como reproductor es directamente proporcional a su idoneidad, siendo los más reproducidos los más idóneos. Los 20 restantes se obtienen cogiendo uno de los 20 mejores y aplicándole una mutación, es decir cambiando uno de sus genes de forma aleatoria. En la figura siguiente (Fig. 23) se puede el diagrama de este algoritmo.

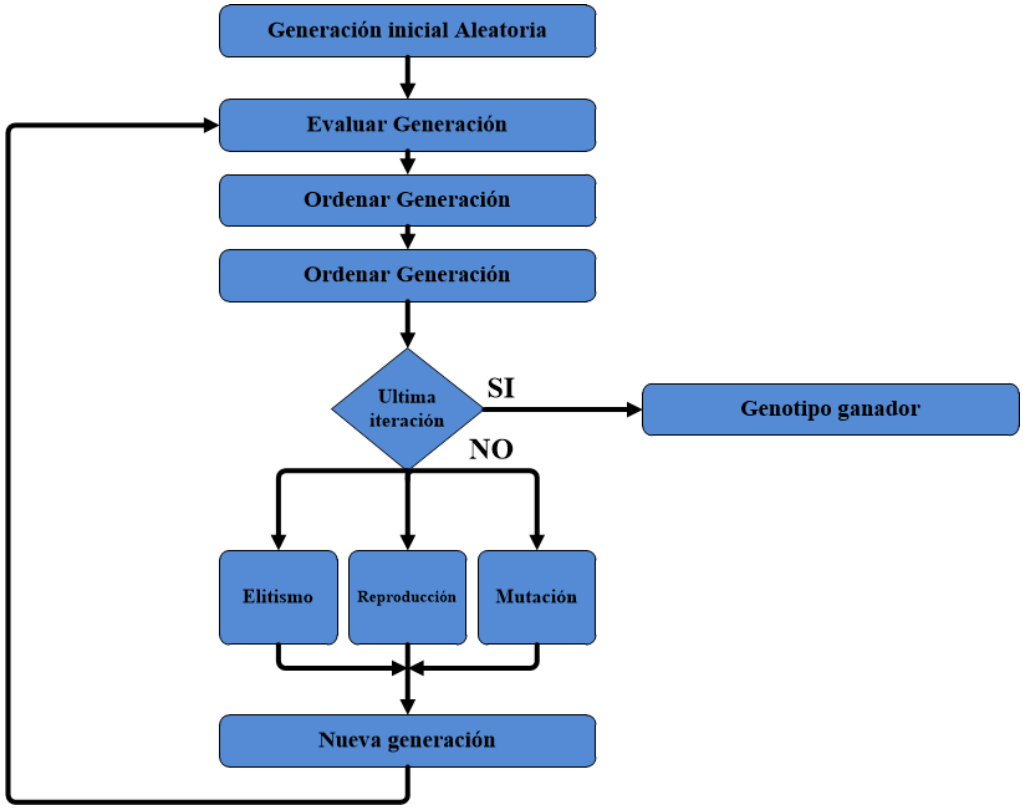

Fig. 23 Diagrama del AG

Tras repetir el proceso 200 veces, se obtiene que prácticamente la totalidad de la población posee el mismo genotipo, el de mayor idoneidad. Esto puede querer decir dos cosas, este genotipo es el mejor absoluto de las posibles combinaciones o se trata de un máximo local en el que la evolución se estanca sin poder progresar, debido a que el proceso se repite pocas veces o a que la cantidad de mutación aplicada es insuficiente. Para solventar el problema de un máximo local, se repite el proceso 4 veces. Si el resultado final es el mismo en las 4 veces, con condiciones iniciales distintas, es muy probable que estemos ante el máximo absoluto. El código de este algoritmo se puede observar en el ANEXO 5.

Como se puede ver en los resultados (ANEXO 1, apartados 1 y 2), hay varios genotipos que aparecen varias veces mientras que otros solo aparecen una vez. Se podría suponer que los que aparecen repetidos son el máximo absoluto, mientras que los que aparecen una sola vez forman un máximo local. Sin embargo, para mayor seguridad de los resultados, se repite el proceso una vez más, pero en esta ocasión los cuatro ganadores entran en los genotipos iniciales. Finalmente, con este proceso se obtiene el máximo absoluto de las posibles combinaciones, es este el que se utilizara en los clasificadores (ANEXO 1, apartados 3 y 4)

5. SELECCIÓN DE ARQUITECTURA

El siguiente paso en la construcción, es la elección del propio clasificador. Como se menciona anteriormente, el tipo que se utiliza es un MLP. Un paso importante en la construcción de estos clasificadores es la elección de la arquitectura neuronal. En este apartado se selecciona la arquitectura neuronal para cada uno.

Para la selección de las arquitecturas de los MLP que se utilizan en los clasificadores, se opta por probar con hasta un máximo de 2 capas de procesamiento. El proceso de selección de la arquitectura es el siguiente: se calcula todas las combinaciones posibles para 2 capas y 25 neuronas por capa. La experiencia dicta que el número de neuronas óptimo para la capa de procesamiento es un término medio entre las variables de entrada y las clases de salida, por ello se decide probar solo hasta 25 neuronas. Para todas estas combinaciones de arquitecturas se entrena un MLP, mediante un entrenamiento tipo Levenberg-Marquardt y utilizando toda la base de datos para entrenar, en la figura siguiente (Fig. 24) se puede observar la arquitectura del MLP. Se calcula la matriz de confusión para cada uno de estos entrenamientos y con ella el índice Kappa. Es importante destacar que este índice Kappa calculado no es un indicador de la eficacia del clasificador, solo sirve para comparar entre las distintas arquitecturas. El código utilizado para esta selección se puede observar en el ANEXO 6.

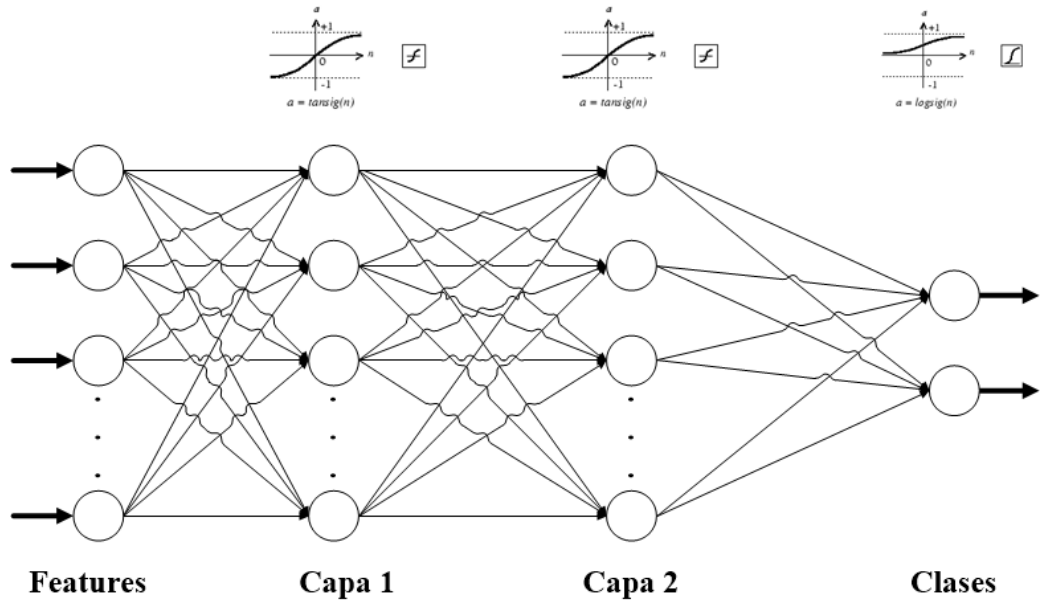

Fig. 24 Arquitectura MLP

Para el clasificador de dirección se obtienen la siguiente arquitectura (Tabla 1):

Capa intermedia 1	Capa intermedia 2	Índice Kappa	Entrenamiento
21	5	0.9827	L-M

Tabla 1 Arquitectura MLP para dirección

Para el clasificador por tamaño se obtienen la siguiente arquitectura (Tabla 2):

Capa intermedia 1	Capa intermedia 2	Índice Kappa	Entrenamiento
5	20	0.9703	L-M

Tabla 2 Arquitectura MLP para tamaño

6. EVALUACIÓN DE LOS CLASIFICADORES

El último punto en la elaboración de un clasificador es evaluar la calidad del mismo. En este apartado se evalúa la calidad de los clasificadores construidos. Como existe el problema de que la base de datos es poco extensa, se decide utilizar como método de evaluación la CV.

Para aplicar esta CV a la evaluación de los clasificadores, se decide dividir la base de datos en 10 grupos de validación. Se realizara el entrenamiento del MLP utilizando 9 de estos para entrenar la red y se utiliza el restante para su evaluación, obteniendo así su matriz de confusión. Este proceso se repite para cada uno de los 10 grupos de validación y posteriormente se suman las 10 matrices de confusión para conseguir la matriz de confusión total. Con esta se puede calcular el índice Kappa del clasificador.

Se ha replicado el proceso anteriormente mencionado para ambos clasificadores obteniendo los siguientes resultados

Para el clasificador de dirección se obtiene la siguiente matriz de confusión y el siguiente índice Kappa (Tabla 3):

		Clase real	
		Salida	Entrada
Clase asignada	Salida	9	0
	Entrada	24	80
Kappa		-0,3886	

Tabla 3 Matriz de confusión e índice Kappa para dirección

Para el clasificador por tamaño se obtiene la siguiente matriz de confusión y el siguiente índice Kappa (Tabla 4):

		Clase real	
		Mediano	Grande
Clase asignada	Mediano	73	34
	Grande	0	6
Kappa		0,6982	

Tabla 4 Matriz de confusión e índice Kappa para tamaño

7. CONCLUSIONES

Como conclusiones de este capítulo, como se puede apreciar en las tablas del apartado anterior, se obtiene que para una clasificación basada en el tamaño se dispone de un buen clasificador. Sin embargo, para la clasificación basada en la dirección el resultado obtenido no es adecuado. Por tanto, en el capítulo siguiente, se repite el proceso de diseño de los clasificadores, pero ampliando las *features* iniciales con otras más complejas.

CAPÍTULO 5. CLASIFICADOR AMPLIADO

1. INTRODUCCIÓN

Como se observa en el capítulo anterior (CAPÍTULO 4), con unas *features* básicas se puede construir un buen clasificador para discriminar vehículos por tamaño. Sin embargo, no se consigue estos resultados para identificar la dirección de circulación. Por ello, en este capítulo se repite el proceso de construcción, ampliando las *features* iniciales con otras más complejas.

El proceso de creación es análogo al capítulo anterior: en primer lugar se plantean unas *features* candidatas. Después se estudia cual es la combinación de *features* más efectiva para cada tipo de clasificación. Como en este caso la dimensionalidad es demasiado grande se realiza una reducción de la misma. Posteriormente, se selecciona la mejor arquitectura para cada clasificador y finalmente se evalúa la eficacia de los mismos.

2. AMPLIACIÓN DE FEATURES INICIALES

Para la mejora de los clasificadores expuestos en el capítulo anterior (CAPÍTULO 4) se amplían las *features* iniciales con el fin de ampliar la cantidad de información que se introduce en el sistema. Se seleccionan 8 nuevas características que se aplican a los 3 ejes de los 2 sensores.

Las nuevas *features* son:

- **Desviación estándar:** Se trata de una medida de la dispersión de los datos con respeto al valor promedio.). Su cálculo se ha realizado mediante la función que se puede observar en el ANEXO 7 sección 1.

$$(1)\, s = \left(\frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 \right)^{\frac{1}{2}}$$

EC. 1 Desviación estándar[47]

- **Desviación media absoluta:** Consiste en la media de la distancia del valor de cada dato con el valor promedio. Este indicador permite describir la variabilidad del conjunto de datos. Su cálculo se ha realizado mediante la función que se puede observar en el ANEXO 7 sección 2.

- **Rango intercuartil:** Es la diferencia entre el tercer cuartil (la mediana de la mitad superior de los datos) y el primer cuartil (la mediana de la mitad inferior) de una distribución. Proporciona un indicador estadístico robusto de la medida de la dispersión. Su cálculo se ha realizado mediante la función que se puede observar en el ANEXO 7 sección 3.

- **Entropía espectral:** [47] Este concepto está basado en la entropía de Shannon, consiste en tratar la distribución de potencia normalizada de la señal en el dominio de la frecuencia como una distribución de probabilidad y calcular la entropía de Shannon de esta. Su cálculo se ha realizado mediante la función que se puede observar en el ANEXO 7 sección 4.

$$EI(s) = -\sum_i s_i^2 \log(s_i^2)$$

EC. 2 Entropía espectral[47]

- **Medida energética:** Este indicador consiste en calcular la suma de los cuadrados de los valores de la señal y dividirlo para el número de valores. Su cálculo se ha realizado mediante la función que se puede observar en el ANEXO 7 sección 5.

- **Asimetría de la señal en el dominio de la frecuencia:**[47] Esta *feature* es una medida de la asimetría de los datos entorno al punto medio de la muestra. Si la asimetría es negativa, los datos se distribuyen más a la izquierda y si es positiva se distribuyen más a la derecha. En una distribución perfectamente simétrica, la asimetría es cero. Su cálculo se ha realizado mediante la función que se puede observar en el ANEXO 7 sección 6.

$$s = \frac{E(x-\mu)^3}{\sigma^3}$$

EC. 3 Asimetría de la señal en el dominio de la frecuencia[47]

- **Curtosis:** Es una característica de forma de una distribución que mide cuan atípica es. Una mayor Curtosis implica una mayor concentración de puntos cerca del pico de la distribución y lejos de las colas. La distribución normal tiene una Curtosis de 3, las distribuciones con Curtosis mayores de 3 son más apuntadas y con colas menos gruesas. Por contrario las distribuciones con Curtosis inferiores a 3 tienen los picos menores y unas colas más gruesas. Su cálculo se ha realizado mediante la función que se puede observar en el ANEXO 7 sección 7.

- **Transformada rápida de Fourier:** Es un eficiente algoritmo que permite calcular la transformada discreta de Fourier. Esta transformación permite obtener una representación en el dominio de la frecuencia de la señal inicial, la cual se encontraba en el dominio del tiempo. Para la ampliación de los *features* se utilizan las 4 primeras frecuencias y sus potencias espectrales, además también se utiliza la media ponderada de las frecuencias espectrales. Su cálculo se ha realizado mediante la función que se puede observar en el ANEXO 7 sección 8.

$$X(k) = \sum_{j=1}^{N} x(j)\omega_N^{(j-1)(k-1)}$$

$$x(j) = (1/N)\sum_{k=1}^{N} X(k)\omega_N^{-(j-1)(k-1)}$$

EC. 4 Transformada rápida de Fourier[47]

3. REDUCCIÓN DE FEATURES

Una vez se tienen todas las *features*, el siguiente paso es seleccionar cuál de estas es relevante para cada clasificador. Este paso es análogo al desarrollado en el CAPÍTULO 4 en el apartado 4, por tanto el procedimiento que se sigue es el mismo.

En este nuevo proceso de reducción de las *features* del problema se vuelve a definir un genoma distinto para el AG. Este genoma está compuesto por el mismo que en el capítulo anterior, 31 variables binarias y 2 variables numéricas, más las necesarias para las nuevas *features*. Para estas se incluye una variable binaria para especificar si se va a utilizar o no. Como son un total de 16 nuevas *features* y cada una de ellas puede ser aplicada en cada uno de los 3 ejes de los 2 sensores, se amplía con 96 nuevas variables binarias. Por lo que el genoma final que se obtiene está compuesto por 127 variables binarias y 2 variables numéricas.

En el ANEXO 2 se presentan los resultados de esta reducción. Como se puede observar, finalmente se obtienen 61 *features* para el clasificador de dirección y 68 para el clasificador de tamaño. Sin embargo, dependiendo del valor de la variable de número de muestras para la media de barra, aumenta o disminuye el número de *features*. Esto es debido a que, contra menor número de muestras por media, más medias de barras salen por cada detección. Por ello, finalmente se obtiene 178 *features* para el clasificador de dirección y 82 para el clasificador de tamaño.

4. REDUCCIÓN DE LAS DIMENSIONES

Al aumentar el número de *features*, aparece un nuevo problema, el aumento de la dimensionalidad. Con las hasta 178 *features* que tienen los nuevos clasificadores se hace inviable las simulaciones necesarias para la construcción. Por ello se aplica PCA para reducir la dimensionalidad y así, poder abordar el diseño.

Tras aplicar PCA a los resultados de ambos clasificadores se obtiene la siguiente relación entre la cantidad de información que se conserva (es decir, cantidad de varianza original que describen) y el número de auto vectores a utilizar (Fig. 25 y Fig. 26). Se decide utilizar 40 auto vectores, los cuales mantienen un 97,97% de información para la clasificación de dirección y un 99,25% para la de tamaño.

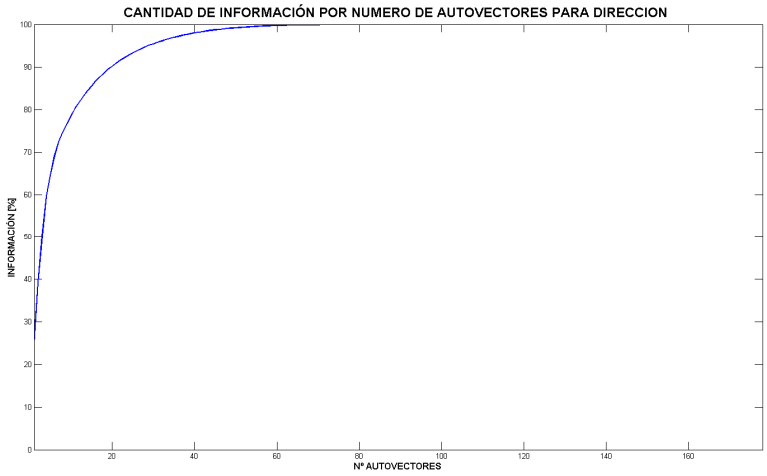

Fig. 25 Cantidad de información por número de auto-vectores para dirección

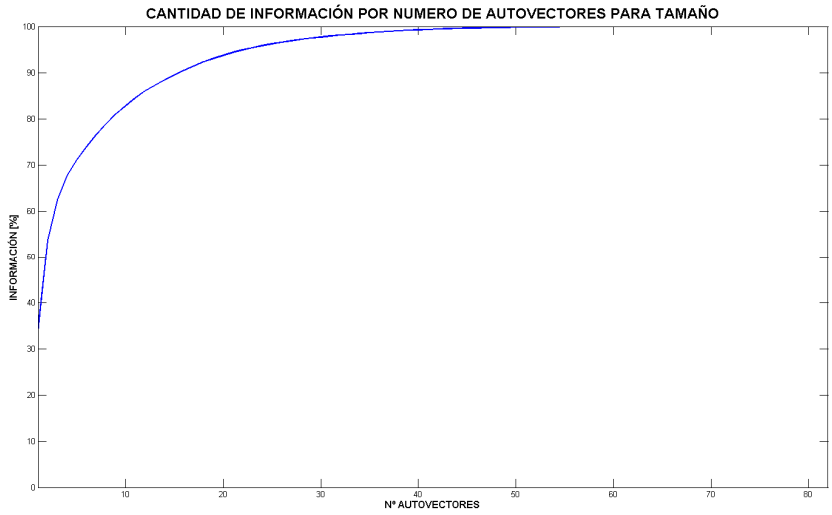

Fig. 26 Cantidad de información por número de auto-vectores para tamaño

5. SELECCIÓN DE ARQUITECTURA

Una vez reducida la dimensión de las entradas de los clasificadores se procede a la construcción del clasificador. Análogamente ha como se realiza en el capítulo anterior, en este apartado se estudia la arquitectura óptima para los MLP.

Al igual que en el CAPÍTULO 4, en el apartado 5, se decide por probar un máximo de 2 capas de procesamiento. Sin embargo, en esta ocasión como el número entradas son 40, las combinaciones probadas son con un mínimo de 10 neuronas por capa y un máximo de 40. En la figura siguiente (Fig. 27Fig. 24) se puede observar la arquitectura del MLP

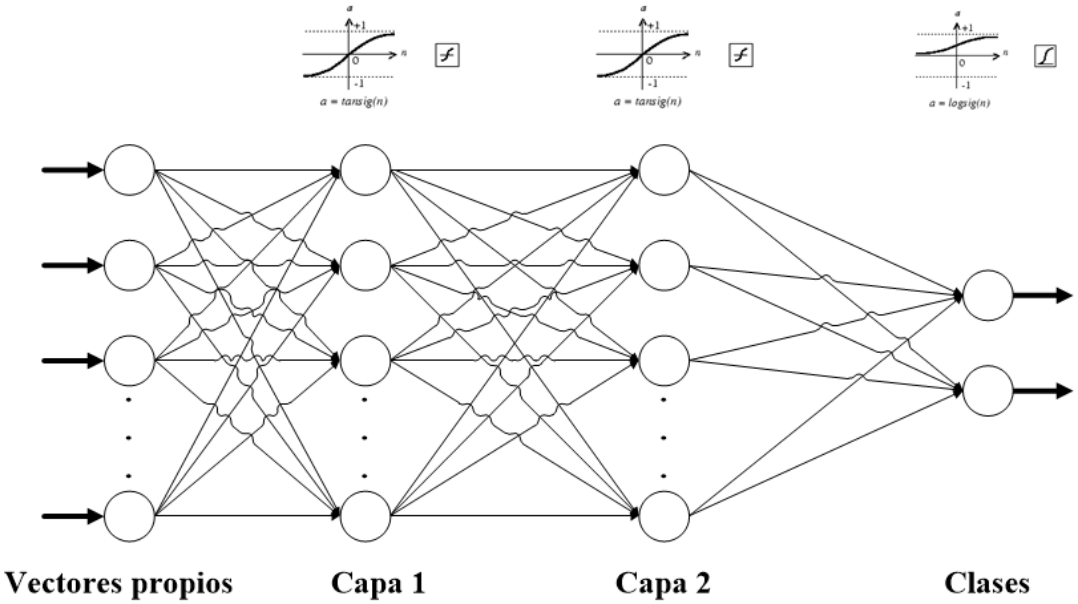

Fig. 27 Arquitectura MLP

Para el clasificador de dirección se obtienen las siguientes arquitecturas ganadoras (Tabla 5):

Capa intermedia 1	Capa intermedia 2	Índice Kappa	Entrenamiento
22	38	0,7	L-M
25	37	0,7	L-M
19	29	0,7	L-M
10	29	0,7	L-M
23	28	0,7	L-M
15	16	0,7	L-M
18	15	0,7	L-M
32	12	0,7	L-M

Tabla 5 Arquitecturas MLP para dirección

Para el clasificador por tamaño se obtiene la siguiente arquitectura (Tabla 6):

Capa intermedia 1	Capa intermedia 2	Índice Kappa	Entrenamiento
19	28	0,4211	L-M
14	26	0,4211	L-M

Tabla 6 Arquitectura MLP para tamaño

6. EVALUACIÓN DE LOS CLASIFICADORES

Llegados a este punto del diseño de los nuevos clasificadores se procede a evaluar su calidad. Esta evaluación se realiza con un procedimiento análogo al utilizado en el capítulo anterior (CAPÍTULO 4, apartado 6), pero en este caso se usan los vectores propios como entradas a la red neuronal. Además, como en el apartado anterior se tiene varias arquitecturas ganadoras con el mismo índice Kappa, se repite el proceso para todas ellas y se selecciona la mejor.

Para el clasificador de dirección se obtiene que la mejor arquitectura es con 18 neuronas en la primera capa y 15 en la segunda. Para esta arquitectura se obtiene la siguiente matriz de confusión y el siguiente índice Kappa (Tabla 7):

		Clase real	
		Salida	Entrada
Clase asignada	Salida	24	5
	Entrada	9	75
Kappa		0,7230	

Tabla 7 Matriz de confusión e índice Kappa para dirección

Para el clasificador de tamaño se obtiene que la mejor arquitectura es con 14 neuronas en la primera capa y 26 en la segunda. Para esta arquitectura se obtiene la siguiente matriz de confusión y el siguiente índice Kappa (Tabla 8):

		Clase real	
		Mediano	Grande
Clase asignada	Mediano	75	26
	Grande	1	11
Kappa		0,7583	

Tabla 8 Matriz de confusión e índice Kappa para tamaño

Como se puede apreciar en las tablas anteriores, no solo se soluciona el problema que se tenía con el clasificador basado en la dirección, sino que se presenta una mejora en ambos clasificadores, obteniéndose así unos buenos clasificadores para ambos tipos.

CAPÍTULO 6. CONCLUSIONES

1. CONCLUSIONES

El principal objetivo de este TFM se cumple satisfactoriamente. Como se puede comprobar en los capítulos anteriores, es posible clasificar los vehículos, tanto por su tamaño como por su dirección. Por lo que, la viabilidad de este tipo de tecnología queda claramente demostrada. La clasificación de vehículos, basada en el tamaño, se puede conseguir con unas *features* básicas con un bajo coste de computación. Sin embargo, para una buena clasificación basada en la dirección del vehículo, es necesaria la utilización de unas *features* más complejas, que requieren un mayor tiempo de computación.

A continuación, se exponen los objetivos parciales de este trabajo y su grado de cumplimento:

Búsqueda de información sobre métodos de clasificación: Se realiza una búsqueda de información sobre los sistemas más importantes para la monitorización de vehículos que están implantados o que se están desarrollando actualmente. Esta información es utilizada para la toma de las decisiones previas al desarrollo de los clasificadores. Todo este estudio de trabajo previo se puede observar en el CAPÍTULO 2.

Definición y realización de una prueba de campo para la captura de datos: Con la ayuda de la información obtenida en el objetivo anterior, se define una prueba de campo que permite la obtención de la información el crudo que posteriormente conformara la base de datos. La definición de esta prueba se puede observar en el CAPÍTULO 3, en el apartado 2.

Análisis y procesamiento de los datos para la creación de una base de datos: Con los datos obtenidos en el objetivo anterior se desarrolla una base de datos que puede ser utilizada tanto para el desarrollo del trabajo actual como para el desarrollo de futuros estudios. El pre-procesamiento de los datos, para la obtención de la base de datos, se puede observar en el CAPÍTULO 3, en el apartado 3.

Análisis de la relevancia de los parámetros de los datos (*features*): A los datos que conforman la base de datos, se les realiza un procesamiento, para extraer una serie de *features*, que son posteriormente utilizadas, en el desarrollo de los clasificadores. Estas son inicialmente unas básicas en el CAPÍTULO 4, sin embargo tienen que ser ampliadas posteriormente, en el CAPÍTULO 5, para poder desarrollar un clasificador basado en la dirección de los vehículos. Todas las *features* utilizadas, así como su relevancia para cada tipo de clasificación, se pueden observar en los apartados 2 y 3 de los CAPITULOS 4 y 5.

Desarrollo de los clasificadores: Con las *features* relevantes para cada tipo de clasificación, se desarrollan finalmente los clasificadores. Se utilizan como clasificadores un MLP y como entradas a estos, las *features* seleccionadas o en el caso de los clasificadores ampliados los vectores propios obtenidos de aplicar PCA a estas. El desarrollo de estos clasificadores puede observarse en los apartados 4 y 5 de los CAPITULOS 4 y 5.

Validación de la calidad de los clasificadores: A los clasificadores obtenidos en el objetivo anterior, se les aplica un proceso de validación para obtener su calidad. Este proceso consiste en una CV y como parámetro para medir su calidad se utiliza su índice KAPPA. Las validaciones de estos clasificadores se pueden observar en los apartados 5 y 6 de los CAPITULOS 4 y 5.

2. TRABAJO FUTURO

Como se observa en el apartado anterior, el objetivo principal de este TFM es cumplido de manera satisfactoria. Sin embargo, este trabajo abre un abanico de posibles caminos para trabajos futuros.

A continuación, se exponen los posibles caminos de futuros trabajos:

Base de datos: Como se puede ver a lo largo de todo este trabajo, una de las principales limitaciones con la que se contaba era el reducido tamaño de la base de datos. Uno de los posibles trabajos futuros seria la ampliación de la base de datos, tanto en número como en otros tipos de vehículos.

Nuevos tipos de clasificación: Ligado al posible trabajo futuro anteriormente mencionado, se tiene otro posible camino de mejora. Consistiría en que, contando una base de datos más numerosa y con otros tipos de vehículos, se podría plantear el estudio de otros tipos de clasificaciones (diferenciar entre coches y camiones, contar número de ejes en camiones) o incluso la re-identificación de vehículos.

Integración: Otro de los posibles caminos de mejora seria el estudio de las diferentes posibles integraciones de estos clasificadores en un dispositivo. Estas pueden ser tanto incluyendo toda la lógica de procesamiento en un nodo autónomo, como utilizando el nodo solo para muestrear los datos y realizar el procesamiento de manera externa.

Smart City: Ligado a todos los puntos anteriores, otro de los posibles caminos de mejora seria el estudio de la integración completa de estos nodos sensores, con otros sistemas ya existentes de control de vehículos, para formar una red inteligente de gestión del tráfico.

REFERENCIAS

[1] R. Calle Berges, «Sistema de monitorización de vehículos basado en sensor magnético», Escuela de Ingeniería y Arquitectura, 2015.

[2] «Proyecto MOVVI». [En línea]. Disponible en: http://movvi.iasoft.es/.

[3] C. Sun, «An Investigation in the Use of Inductive Loop Signatures for Vehicle Classification», *Calif. Partn. Adv. Transp. Technol. UC Berkeley*, ene. 2000.

[4] J. J. Lamas-Seco, P. M. Castro, A. Dapena, y F. J. Vazquez-Araujo, «Vehicle Classification Using the Discrete Fourier Transform with Traffic Inductive Sensors», *Sensors*, vol. 15, pp. 27201-27214, oct. 2015.

[5] D. D. Desai y S. B. Somani, «Evaluation of intrusive vehicle detection and classification technique for traffic management», en *2014 Annual IEEE India Conference (INDICON)*, 2014, pp. 1-6.

[6] S. Meta y M. G. Cinsdikici, «Vehicle-Classification Algorithm Based on Component Analysis for Single-Loop Inductive Detector», *IEEE Trans. Veh. Technol.*, vol. 59, n.º 6, pp. 2795-2805, jul. 2010.

[7] H. A. Oliveira, F. R. Barbosa, O. M. Almeida, y A. P. S. Braga, «A vehicle classification based on inductive loop detectors using artificial neural networks», en *Industry Applications (INDUSCON), 2010 9th IEEE/IAS International Conference on*, 2010, pp. 1-6.

[8] J. Gajda, R. Sroka, M. Stencel, A. Wajda, y T. Zeglen, «A vehicle classification based on inductive loop detectors», en *Instrumentation and Measurement Technology Conference, 2001. IMTC 2001. Proceedings of the 18th IEEE*, 2001, vol. 1, pp. 460-464 vol.1.

[9] Y.-K. Ki y D.-K. Baik, «Vehicle-Classification Algorithm for Single-Loop Detectors Using Neural Networks», *IEEE Trans. Veh. Technol.*, vol. 55, n.º 6, pp. 1704-1711, nov. 2006.

[10] J. Gajda y R. Sroka, «Vehicle classification by parametric identification of the measured signals», presentado en 16th IMEKO, 2000, vol. 9, pp. 199–204.

[11] J. Gajda, R. Sroka, M. Stencel, y T. Zeglen, «An Eastern European example of the identification of moving vehicle parameters using the tried and trusted method of weigh in motion», *Traffic Technol*, pp. 87–90, sep. 2000.

[12] «TECBAS S.L.» [En línea]. Disponible en: http://www.tecbas.es.

[13] «INGETRA INGENIERIA DE TRANSITO». [En línea]. Disponible en: http://www.tyssatransito.com.

[14] S. A. Rajab, A. S. Othman, y H. H. Refai, «Novel vehicle and motorcycle classification using single element piezoelectric sensor», en *2012 15th International IEEE Conference on Intelligent Transportation Systems*, 2012, pp. 496-501.

[15] S. A. Rajab y H. H. Refai, «A single element piezoelectric sensor for vehicle classification using the ratio of track width to length», en *2014 IEEE Intelligent Vehicles Symposium Proceedings*, 2014, pp. 1463-1468.

[16] S. A. Rajab, A. Mayeli, y H. H. Refai, «Vehicle classification and accurate speed calculation using multi-element piezoelectric sensor», en *2014 IEEE Intelligent Vehicles Symposium Proceedings*, 2014, pp. 894-899.

[17] P. R. Golla, A. Mukherjee, y B. Harvey, «Extruded segmented sensor for road traffic classification», en *Southeastcon, 2013 Proceedings of IEEE*, 2013, pp. 1-5.

[18] S.-W. Kim, K. Kim, J. Lee, y D. Cho, «Application of fuzzy logic to vehicle classification algorithm in loop/piezo-sensor fusion systems», presentado en Asian Journal of Control, 2001, vol. 3, pp. 64-68.

[19] Z. Chen, T. Ellis, y S. A. Velastin, «Vehicle detection, tracking and classification in urban traffic», en *2012 15th International IEEE Conference on Intelligent Transportation Systems*, 2012, pp. 951-956.

[20] A. Gepperth, J. Edelbrunner, y T. Bucher, «Real-time detection and classification of cars in video sequences», en *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.*, 2005, pp. 625-631.

[21] Z. Chen, N. Pears, M. Freeman, y J. Austin, «Road vehicle classification using Support Vector Machines», en *Intelligent Computing and Intelligent Systems, 2009. ICIS 2009. IEEE International Conference on*, 2009, vol. 4, pp. 214-218.

[22] Z. Dong y Y. Jia, «Vehicle type classification using distributions of structural and appearance-based features», en *2013 IEEE International Conference on Image Processing*, 2013, pp. 4321-4324.

[23] Y. Liu y K. Wang, «Vehicle classification system based on dynamic Bayesian network», en *Service Operations and Logistics, and Informatics (SOLI), 2014 IEEE International Conference on*, 2014, pp. 22-26.

[24] Y. Shan, H. S. Sawhney, y R. Kumar, «Unsupervised learning of discriminative edge measures for vehicle matching between non-overlapping cameras», en *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 2005, vol. 1, pp. 894-901 vol. 1.

[25] W. Wang, Y. Shang, J. Guo, y Z. Qian, «Real-time vehicle classification based on eigenface», en *Consumer Electronics, Communications and Networks (CECNet), 2011 International Conference on*, 2011, pp. 4292-4295.

[26] F. Yamazaki, W. Liu, y T. T. Vu, «Vehicle Extraction and Speed Detection from Digital Aerial Images», en *IGARSS 2008 - 2008 IEEE International Geoscience and Remote Sensing Symposium*, 2008, vol. 3, p. III-1334-III-1337.

[27] C. Harlow y S. Peng, «Automatic vehicle classification system with range sensors», presentado en Transportation research part C, 2001, vol. 9, pp. 231-247.

[28] R. Bajwa, R. Rajagopal, P. Varaiya, y R. Kavaler, «In-pavement wireless sensor network for vehicle classification», en *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, 2011, pp. 85-96.

[29] R. Hostettler, W. Birk, y M. L. Nordenvaad, «Feasibility of road vibrations-based vehicle property sensing», *IET Intell. Transp. Syst.*, vol. 4, n.º 4, pp. 356-364, dic. 2010.

[30] D. Obertov y B. Andrievsky, «Vehicle classification using measurements from accelerometers mounted on the road surface», en *Methods and Models in Automation and Robotics (MMAR), 2014 19th International Conference On*, 2014, pp. 413-417.

[31] M. Stocker, M. Rönkkö, y M. Kolehmainen, «Situational Knowledge Representation for Traffic Observed by a Pavement Vibration Sensor Network», *IEEE Trans. Intell. Transp. Syst.*, vol. 15, n.º 4, pp. 1441-1450, ago. 2014.

[32] J. Lan, T. Lan, y S. Nahavandi, «A novel application of a microaccelerometer for target classification», *IEEE Sens. J.*, vol. 4, n.º 4, pp. 519-524, ago. 2004.

[33] G. Burresi y R. Giorgi, «A field experience for a vehicle recognition system using magnetic sensors», en *2015 4th Mediterranean Conference on Embedded Computing (MECO)*, 2015, pp. 178-181.

[34] G. V. Prateek, K. Nijil, y K. V. S. Hari, «Classification of Vehicles Using Magnetic Field Angle Model», en *2013 4th International Conference on Intelligent Systems, Modelling and Simulation*, 2013, pp. 214-219.

[35] Z. Feng y W. Mingzhe, «A New SVM Algorithm and AMR Sensor Based Vehicle Classification», en *Intelligent Computation Technology and Automation, 2009. ICICTA '09. Second International Conference on*, 2009, vol. 2, pp. 421-425.

[36] W. Zhang, G. Tan, N. Ding, Y. Shang, y M. Lin, «Vehicle Classification Algorithm based on Binary Proximity Magnetic Sensors and Neural Network», en *2008 11th International IEEE Conference on Intelligent Transportation Systems*, 2008, pp. 145-150.

[37] S. Taghvaeeyan y R. Rajamani, «Portable Roadside Sensors for Vehicle Counting, Classification, and Speed Measurement», *IEEE Trans. Intell. Transp. Syst.*, vol. 15, n.º 1, pp. 73-83, feb. 2014.

[38] S. Keawkamnerd, J. Chinrungrueng, y C. Jaruchart, «Vehicle Classification with low computation magnetic sensor», en *ITS Telecommunications, 2008. ITST 2008. 8th International Conference on*, 2008, pp. 164-169.

[39] S. Kaewkamnerd, R. Pongthornseri, J. Chinrungrueng, y T. Silawan, «Automatic vehicle classification using wireless magnetic sensor», en *Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 2009. IDAACS 2009. IEEE International Workshop on*, 2009, pp. 420-424.

[40] S. Kaewkamnerd, J. Chinrungrueng, R. Pongthornseri, y S. Dumnin, «Vehicle classification based on magnetic sensor signal», en *Information and Automation (ICIA), 2010 IEEE International Conference on*, 2010, pp. 935-939.

[41] S. Charbonnier, A. C. Pitton, y A. Vassilev, «Vehicle re-identification with a single magnetic sensor», en *Instrumentation and Measurement Technology Conference (I2MTC), 2012 IEEE International*, 2012, pp. 380-385.

[42] D. Kleyko, R. Hostettler, W. Birk, y E. Osipov, «Comparison of Machine Learning Techniques for Vehicle Classification Using Road Side Sensors», en *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, 2015, pp. 572-577.

[43] A. Bono Nuez, «Redes Neuronales Artificiales para Evaluar la Calidad de Vida», Tesis Doctoral, Universidad de Zaragoza, 2015.

[44] D. Buldain, «Apuntes prácticas de redes neuronales artificiales», Universidad de Zaragoza, 2016.

[45] J. R. Landis, y G. G. Koch, «The measurement of observer agreement for categorical data», *biometrics*, vol. 159-174, 1977.

[46] J. L. Fleiss, B. Levin, y M. Cho Paik, «Statistical methods for rates and proportions», *Wiley series in probability and statistics*, 2003.

[47] «Manual de MATLAB», *MathWorks*. [En línea]. Disponible en: //es.mathworks.com/help/matlab/index.html.

ANEXOS

ANEXO 1. RESULTADOS REDUCCIÓN DE FEATURES

1. RESULTADOS DEL AG PARA LA CLASIFICACIÓN DE DIRECCIÓN

Nº Repetición	Genotipos																															Kappa
Repetición 1	0	0	1	1	1	0	1	0	0	0	0	1	0	1	0	0	0	0	1	1	0	0	0	0	1	1	1	1	1	0	1	23	362	0,9613
Repetición 2	0	0	1	1	1	0	1	0	0	0	0	1	0	1	0	0	0	0	1	1	0	0	0	0	1	1	1	1	0	1	1	23	369	0,9613
Repetición 3	0	0	1	1	1	0	1	0	0	0	0	1	0	1	0	0	0	0	1	1	0	0	0	0	1	1	1	1	0	1	1	23	369	0,9613
Repetición 4	1	1	1	1	1	0	1	0	0	1	1	1	1	1	0	0	1	0	1	1	0	0	1	1	0	1	1	1	1	0	0	143	376	0,9524

2. GENOTIPO FINAL PARA LA CLASIFICACIÓN DE DIRECCIÓN

Genotipos																																Kappa
0	0	1	1	1	0	1	0	1	0	0	1	0	1	0	0	0	0	1	1	0	0	0	0	1	0	0	1	1	1	1	23	362	0,9613

3. RESULTADOS DEL AG PARA LA CLASIFICACIÓN DE TAMAÑO

Nº Repetición	Genotipos																																Kappa
Repetición 1	1	1	0	1	1	0	1	1	1	0	1	1	0	1	1	1	1	0	1	0	1	0	1	1	1	0	0	0	0	0	1	27	614	0,9115
Repetición 2	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	1	0	1	0	1	1	0	0	1	1	1	362	23	0,9380
Repetición 3	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	1	0	1	0	1	1	0	0	1	1	1	362	23	0,9380
Repetición 4	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	1	0	1	0	1	1	0	0	1	1	1	362	23	0,9380

4. GENOTIPO FINAL PARA LA CLASIFICACIÓN DE TAMAÑO

Genotipos																																Kappa
0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	1	0	1	0	1	1	0	0	1	1	1	362	23	0,9380

ANEXO 2. RESULTADOS REDUCCIÓN DE FEATURES AMPLIADOS

1. RESULTADOS DEL AG PARA LA CLASIFICACIÓN DE DIRECCIÓN

GENOMAS	KAPPA
0 1 1 0 1 0 0 0 0 1 1 1 1 1 1 0 1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 1 1 1 0 1 1 0 1 1 1 0 1 0 1 1 1 0 0 0 0 1 0 0 1 1 1 0 0 0 0 0 0 1 0 1 0 0 1 1 1 0 1 1 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 1 0 0 1 1 1 24 353	0,9617
0 1 1 0 1 0 0 0 0 1 1 1 1 1 1 0 1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 1 1 1 0 1 1 0 1 1 1 0 1 0 1 1 1 0 0 0 0 1 0 0 1 1 1 0 0 0 0 0 0 1 0 1 0 0 1 1 1 0 1 1 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 1 0 0 1 1 1 24 353	0,9617
0 1 1 0 1 0 0 0 0 1 1 1 1 1 1 0 1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 1 1 1 0 1 1 0 1 1 1 0 1 0 1 1 1 0 0 0 0 1 0 0 1 1 1 0 0 0 0 0 0 1 0 1 0 0 1 1 1 0 1 1 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 1 0 0 1 1 1 24 353	0,9617
0 1 1 0 1 0 0 0 0 1 1 1 1 1 1 0 1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 1 1 1 0 1 1 0 1 1 1 0 1 0 1 1 1 0 0 0 0 1 0 0 1 1 1 0 0 0 0 0 0 1 0 1 0 0 1 1 1 0 1 1 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 1 0 0 1 1 1 24 353	0,9617

2. GENOTIPO FINAL PARA LA CLASIFICACIÓN DE DIRECCIÓN

GENOMAS	KAPPA
0 1 1 0 1 0 0 0 0 1 1 1 1 1 1 0 1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 1 1 1 0 1 1 0 1 1 1 0 1 0 1 1 1 0 0 0 0 1 0 0 1 1 1 0 0 0 0 0 0 1 0 1 0 0 1 1 1 0 1 1 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 1 0 0 1 1 1 24 353	0,9617

3. RESULTADOS DEL AG PARA LA CLASIFICACIÓN DE TAMAÑO

GENOMAS	KAPPA
1 1 1 1 0 1 1 1 0 1 1 0 1 0 1 1 1 1 0 0 0 1 0 1 0 0 1 1 0 1 1 1 0 0 1 1 1 1 0 1 0 1 1 0 0 1 0 1 1 1 1 0 1 0 1 0 0 1 1 0 0 0 0 1 1 1 0 1 1 0 1 0 1 0 1 1 1 0 1 1 0 1 0 1 0 1 1 1 0 1 1 0 1 0 1 0 1 0 0 0 0 1 1 0 0 1 1 0 1 1 0 115 108	0,9026
1 1 1 1 0 1 1 1 0 1 1 0 1 0 1 1 1 1 0 0 0 1 0 1 0 0 1 1 0 1 1 1 0 0 1 1 1 1 0 1 0 1 1 0 0 1 0 1 1 1 1 0 1 0 1 0 0 1 1 0 0 0 0 1 1 1 0 1 1 0 1 0 1 0 1 1 1 0 1 1 0 1 0 1 0 1 1 1 0 1 1 0 1 0 1 0 1 0 0 0 0 1 1 0 0 1 1 0 1 1 0 115 108	0,9026
1 1 1 1 0 1 1 1 0 1 1 0 1 0 1 1 1 1 0 0 0 1 0 1 0 0 1 1 0 1 1 1 0 0 1 1 1 1 0 1 0 1 1 0 0 1 0 1 1 1 1 0 1 0 1 0 0 1 1 0 0 0 0 1 1 1 0 1 1 0 1 0 1 0 1 1 1 0 1 1 0 1 0 1 0 1 1 1 0 1 1 0 1 0 1 0 1 0 0 0 0 1 1 0 0 1 1 0 1 1 0 115 108	0,9026
1 1 1 1 0 1 1 1 0 1 1 0 1 0 1 1 1 1 0 0 0 1 0 1 0 0 1 1 0 1 1 1 0 0 1 1 1 1 0 1 0 1 1 0 0 1 0 1 1 1 1 0 1 0 1 0 0 1 1 0 0 0 0 1 1 1 0 1 1 0 1 0 1 0 1 1 1 0 1 1 0 1 0 1 0 1 1 1 0 1 1 0 1 0 1 0 1 0 0 0 0 1 1 0 0 1 1 0 1 1 0 115 108	0,9026

4. GENOTIPO FINAL PARA LA CLASIFICACIÓN DE TAMAÑO

GENOMAS	KAPPA
1 1 1 1 0 1 1 1 0 1 1 0 1 0 1 1 1 1 0 0 0 1 0 1 0 0 1 1 0 1 1 1 0 0 1 1 1 1 0 1 0 1 1 0 0 1 0 1 1 1 1 0 1 0 1 0 0 1 1 0 0 0 0 1 1 1 0 1 1 0 1 0 1 0 1 1 1 0 1 1 0 1 0 1 0 1 1 1 0 1 1 0 1 0 1 0 1 0 0 0 0 1 1 0 0 1 1 0 1 1 0 115 108	0,9026

ANEXO 3. CARACTERÍSTICAS DISPOSITIVO

El dispositivo utilizado para la realización de las pruebas de campo durante este proyecto es el siguiente (Fig. 28). Se trata de un nodo sensor con capacidad para muestrear un sensor magnético y un acelerómetro y transmitir esta información de manera inalámbrica mediante una comunicación ZigBee.

Fig. 28 Dispositivo

Además, también posee sensores de entorno (temperatura y humedad), posibilidad de otras comunicaciones (SPI, I2C y UART) y una memoria EPROM, en la Tabla 9 se muestra los componentes de este dispositivo. En el modo de funcionamiento que se ha utilizado en este trabajo, muestreo de ambos sensores y transmisión en streaming de los datos de manera inalámbrica, la frecuencia máxima de muestreo es 100Hz.

Componentes	
Microcontrolador	PIC18F26J11-I/ML
Sensor magnético	HMC5983
Acelerómetro	MMA8652FC
Sensores de entorno	SHT21
Comunicación (zigbee)	ETRX357
Otras comunicaciones	SPI, I2C UART
Memoria	24AA1025-I/SM
Alimentación	Batería (3,6V)

Tabla 9 Componentes

ANEXO 4. FUNCIONES FEATURES BÁSICAS

1. MÍNIMO ABSOLUTO DE LA SEÑAL

```
function Salida = minimos(matriz, longitud)
    Salida = [];
    for u=1:1:size(matriz,1)
        valor = matriz (u,1);
        for i=1:1:longitud(u)
            if(valor > matriz (u,i))
                valor = matriz (u,i);
            end
        end
        Salida = [Salida; valor];
    end
end
```

2. MEDIA DE LA SEÑAL

```
function Salida = medias(matriz, longitud)
    Salida = [];
    for u=1:1:size(matriz,1)
        valor = matriz (u,1);
        for i=2:1:longitud(u)
          valor = valor + matriz (u,i);
        end
        Salida = [Salida; valor/longitud(u)];
    end

end
```

3. LOS PICOS DE HILL-PATTERN

```
function [BarraMAGX BarraMAGY BarraMAGZ BarraACX BarraACY BarraACZ] =
Mbarra(MagEjeX, MagEjeY, MagEjeZ, ACEjeX, ACEjeY, ACEjeZ,Genomas,
longitud)

 numeroMuestras = Genomas(G,32);
nmax = MaxTam/numeroMuestras;
nmax=floor(nmax);

for u=1:size(MagEjeX,1)                    %Para cada vehículo

        n = Longitudes(u)/numeroMuestras;
        n=floor(n);
        if n==0
            n=1;
        end
```

```
        %%%%%%%%%%%%%%%%%%%%%%%
        if (Genomas(G,19))
            if n<=1
                auxMAGX = [];
                aux = MagEjeX(u,1:Longitudes(u));
                auxMAGX = [auxMAGX, mean(aux,2)];
            else
                auxMAGX = [];
                aux = MagEjeX(u,1:(numeroMuestras));
                auxMAGX = [auxMAGX, mean(aux,2)];
                for i=1:n-2
                aux =
MagEjeX(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxMAGX = [auxMAGX, mean(aux,2)];
                end
                aux = MagEjeX(u,(numeroMuestras*(n-1)):Longitudes(u));
                auxMAGX = [auxMAGX, mean(aux,2)];
            end
             if n<nmax
                 BarraMAGX = [BarraMAGX; auxMAGX zeros(1,nmax-n)];
             else
                 BarraMAGX = [BarraMAGX; auxMAGX];
             end
        end
         %%%%%%%%%%%%%%%%%%%%%%%
        if (Genomas(G,20))
           if n<=1
                auxMAGY = [];
                aux = MagEjeY(u,1:Longitudes(u));
                auxMAGY = [auxMAGY, mean(aux,2)];
           else
                auxMAGY = [];
                aux = MagEjeY(u,1:(numeroMuestras));
                auxMAGY = [auxMAGY, mean(aux,2)];
                for i=1:n-2
                aux =
MagEjeY(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxMAGY = [auxMAGY, mean(aux,2)];
                end
                aux = MagEjeY(u,(numeroMuestras*(n-1)):Longitudes(u));
                auxMAGY = [auxMAGY, mean(aux,2)];
           end
             if n<nmax
                 BarraMAGY = [BarraMAGY; auxMAGY zeros(1,nmax-n)];
             else
                 BarraMAGY = [BarraMAGY; auxMAGY];
             end
        end
        %%%%%%%%%%%%%%%%%%%%%%%
        if (Genomas(G,21))
           if n<=1
                auxMAGZ = [];
                aux = MagEjeZ(u,1:Longitudes(u));
                auxMAGZ = [auxMAGZ, mean(aux,2)];
           else
```

```
                auxMAGZ = [];
                aux = MagEjeZ(u,1:(numeroMuestras));
                auxMAGZ = [auxMAGZ, mean(aux,2)];
                for i=1:n-2
                aux =
MagEjeZ(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxMAGZ = [auxMAGZ, mean(aux,2)];
                end
                aux = MagEjeZ(u,(numeroMuestras*(n-1)):Longitudes(u));
                auxMAGZ = [auxMAGZ, mean(aux,2)];
            end
              if n<nmax
                  BarraMAGZ = [BarraMAGZ; auxMAGZ zeros(1,nmax-n)];
              else
                  BarraMAGZ = [BarraMAGZ; auxMAGZ];
              end
        end
        %%%%%%%%%%%%%%%%%%%%%%
        if (Genomas(G,22))
            if n<=1
                auxACX = [];
                aux = ACEjeX(u,1:Longitudes(u));
                auxACX = [auxACX, mean(aux,2)];
            else
                auxACX = [];
                aux = ACEjeX(u,1:(numeroMuestras));
                auxACX = [auxACX, mean(aux,2)];
                for i=1:n-2
                aux =
ACEjeX(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxACX = [auxACX, mean(aux,2)];
                end
                aux = ACEjeX(u,(numeroMuestras*(n-1)):Longitudes(u));
                auxACX = [auxACX, mean(aux,2)];
            end
              if n<nmax
                  BarraACX = [BarraACX; auxACX zeros(1,nmax-n)];
              else
                  BarraACX = [BarraACX; auxACX];
              end
        end
        %%%%%%%%%%%%%%%%%%%%%%
        if (Genomas(G,23))
            if n<=1
                auxACY = [];
                aux = ACEjeY(u,1:Longitudes(u));
                auxACY = [auxACY, mean(aux,2)];
            else
                auxACY = [];
                aux = ACEjeY(u,1:(numeroMuestras));
                auxACY = [auxACY, mean(aux,2)];
                for i=1:n-2
                aux =
ACEjeY(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxACY = [auxACY, mean(aux,2)];
```

```
                end
                aux = ACEjeY(u,(numeroMuestras*(n-1)):Longitudes(u));
                auxACY = [auxACY, mean(aux,2)];
            end
              if n<nmax
                  BarraACY = [BarraACY; auxACY zeros(1,nmax-n)];
              else
                  BarraACY = [BarraACY; auxACY];
              end
        end
        %%%%%%%%%%%%%%%%%%%%%%%%
        if (Genomas(G,24))
           if n<=1
                auxACZ = [];
                aux = ACEjeZ(u,1:Longitudes(u));
                auxACZ = [auxACZ, mean(aux,2)];
           else
                auxACZ = [];
                aux = ACEjeZ(u,1:(numeroMuestras));
                auxACZ = [auxACZ, mean(aux,2)];
                for i=1:n-2
                aux =
ACEjeZ(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxACZ = [auxACZ, mean(aux,2)];
                end
                aux = ACEjeZ(u,(numeroMuestras*(n-1)):Longitudes(u));
                auxACZ = [auxACZ, mean(aux,2)];
           end
              if n<nmax
                  BarraACZ = [BarraACZ; auxACZ zeros(1,nmax-n)];
              else
                  BarraACZ = [BarraACZ; auxACZ];
              end
        end
    end
end
```

4. AVERAGE-BAR TRANSFORM

```
function [picosMAGX picosMAGY picosMAGZ picosACX picosACY picosACZ] =
Mbarra(MagEjeX, MagEjeY, MagEjeZ, ACEjeX, ACEjeY, ACEjeZ, Genomas,
longitud)
    auxiliar = 0;
    numpicos = Genomas(G,33);

    for j=1:size(MagEjeX,1)

        %%%%%%%%%%%%%%%%%%%%%%%%
        if (Genomas(G,25))
            auxMAGX = 0;
            for i=numpicos+1:size(MagEjeX,2)-(numpicos+1)
                auxiliar = 1;
                for d=1:numpicos
```

```
            if(MagEjeX(j,i)<=MagEjeX(j,i+d))
                auxiliar = 0;
            end
            if(MagEjeX(j,i)<=MagEjeX(j,i-d))
                auxiliar = 0;
            end
        end
        auxMAGX =  auxMAGX+auxiliar;
    end
    picosMAGX = [picosMAGX; auxMAGX];
end
%%%%%%%%%%%%%%%%%%%%%%%%
if (Genomas(G,26))
    auxMAGY = 0;
    for i=numpicos+1:size(MagEjeY,2)-(numpicos+1)
        auxiliar = 1;
        for d=1:numpicos
            if(MagEjeY(j,i)<=MagEjeY(j,i+d))
                auxiliar = 0;
            end
            if(MagEjeY(j,i)<=MagEjeY(j,i-d))
                auxiliar = 0;
            end
        end
        auxMAGY =  auxMAGY+auxiliar;
    end
    picosMAGY = [picosMAGY; auxMAGY];
end
%%%%%%%%%%%%%%%%%%%%%%%%
if (Genomas(G,27))
    auxMAGZ = 0;
    for i=numpicos+1:size(MagEjeZ,2)-(numpicos+1)
        auxiliar = 1;
        for d=1:numpicos
            if(MagEjeZ(j,i)<=MagEjeZ(j,i+d))
                auxiliar = 0;
            end
            if(MagEjeZ(j,i)<=MagEjeZ(j,i-d))
                auxiliar = 0;
            end
        end
        auxMAGZ =  auxMAGZ+auxiliar;
    end
    picosMAGZ = [picosMAGZ; auxMAGZ];
end
 %%%%%%%%%%%%%%%%%%%%%%%%
if (Genomas(G,28))
    auxACX = 0;
    for i=numpicos+1:size(ACEjeX,2)-(numpicos+1)
        auxiliar = 1;
        for d=1:numpicos
            if(ACEjeX(j,i)<=ACEjeX(j,i+d))
                auxiliar = 0;
            end
            if(ACEjeX(j,i)<=ACEjeX(j,i-d))
```

```
                    auxiliar = 0;
                end
            end
            auxACX =  auxACX+auxiliar;
        end
        picosACX = [picosACX; auxACX];
    end
     %%%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(G,29))
        auxACY = 0;
        for i=numpicos+1:size(ACEjeY,2)-(numpicos+1)
            auxiliar = 1;
            for d=1:numpicos
                if(ACEjeY(j,i)<=ACEjeY(j,i+d))
                    auxiliar = 0;
                end
                if(ACEjeY(j,i)<=ACEjeY(j,i-d))
                    auxiliar = 0;
                end
            end
            auxACY =  auxACY+auxiliar;
        end
        picosACY = [picosACY; auxACY];
    end
    %%%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(G,30))
        auxACZ = 0;
        for i=numpicos+1:size(ACEjeZ,2)-(numpicos+1)
            auxiliar = 1;
            for d=1:numpicos
                if(ACEjeZ(j,i)<=ACEjeZ(j,i+d))
                    auxiliar = 0;
                end
                if(ACEjeZ(j,i)<=ACEjeZ(j,i-d))
                    auxiliar = 0;
                end
            end
            auxACZ =  auxACZ+auxiliar;
        end
        picosACZ = [picosACZ; auxACZ];
    end
  end
end
```

ANEXO 5. AG REDUCCIÓN DE FEATURES

1. AG PARA TAMAÑO

```
clear all;
close all;
fprintf('Analisis completo de las Features mediante Algoritmos
Geneticos\n');
fprintf('Para muestras capturadas automaticamente\n');
fprintf('Para el TAMAÑO\n');


%% CONFIGURAR
%%%%%%%%%%%%%%%%%%%%%%%%%%
nombreArchivo = 'C:\Users\Ruben\Documents\universidad\Master\Trabajo
Final de Master\Base de datos\Datos-Detectados-
Automaticos\DatosAutomaticos.mat';
%%%%%%%%%%%%%%%%%%%%%%%%%%
Ciclos = 300;
%%%%%%%%%%%%%%%%%%%%%%%%%%

%% CARGA DE DATOS
load(nombreArchivo');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Extraccion de las features
fprintf('Extraccion de las features\n');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Features estadisticasa

MAXMAGX= max(MagEjeX,[],2);
MAXMAGY= max(MagEjeY,[],2);
MAXMAGZ= max(MagEjeZ,[],2);
MAXACEX= max(ACEjeX,[],2);
MAXACEY= max(ACEjeY,[],2);
MAXACEZ= max(ACEjeZ,[],2);

MINMAGX= minimos(MagEjeX,Longitudes);
MINMAGY= minimos(MagEjeY,Longitudes);
MINMAGZ= minimos(MagEjeZ,Longitudes);
MINACEX= minimos(ACEjeX,Longitudes);
MINACEY= minimos(ACEjeY,Longitudes);
MINACEZ= minimos(ACEjeZ,Longitudes);

MEANMAGX= medias(MagEjeX,Longitudes);
MEANMAGY= medias(MagEjeY,Longitudes);
MEANMAGZ= medias(MagEjeZ,Longitudes);
MEANACEX= medias(ACEjeX,Longitudes);
MEANACEY= medias(ACEjeY,Longitudes);
MEANACEZ= medias(ACEjeZ,Longitudes);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Creacion de los genomas iniciales
fprintf('Creacion de los genomas iniciales\n');

NuevaGeneracion = randi([0,1],100,31);
auxi = randi([2,400],100,2);
NuevaGeneracion = [NuevaGeneracion, auxi];
KappaMax = [];

MaxTam = max(Longitudes,[],1)


              %%%%%%%%%%%%%%%%%%%%%%%%%%
%      1     MAXMAGX       MAXMAGY       MAXMAGZ       MAXACEX
MAXACEY       MAXACEZ
%      7     MINMAGX       MINMAGY       MINMAGZ       MINACEX
MINACEY       MINACEZ
%      13    MEANMAGX      MEANMAGY      MEANMAGZ      MEANACEX
MEANACEY      MEANACEZ
%      19    BarraMAGX     BarraMAGY     BarraMAGZ     BarraACX
BarraACY      BarraACZ
%      25    picosMAGX     picosMAGY     picosMAGZ     picosACX
picosACY      picosACZ
%      31    Longitud      NumBarras     NumPicos

% Reproductor1 = [1 1 1 1 1 1 0 0 1 0 0 0 0 1 1 1 1 0 1 1 0 0 0 0 0 0
0 0 0 1 212 27];
% Reproductor2 = [0 1 0 1 0 0 1 0 1 1 0 0 0 1 1 0 1 1 1 0 0 0 1 0 0 1
0 1 0 0 228 47];
% Reproductor3 = [0 1 1 1 1 1 0 1 0 1 1 0 0 0 0 0 0 0 0 1 0 0 1 1 0 0
0 1 0 1 48   9];
%
% NuevaGeneracion = [NuevaGeneracion; Reproductor1; Reproductor2;
Reproductor3];

while(Ciclos > 0)
  Genomas =   NuevaGeneracion;
  fprintf('Ciclos:\n');
  Ciclos
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %% Calculo de las Kappas
    fprintf('Calculo de las Kappas\n');
    Kappas = [];


    for G=1:size(Genomas,1)
            % Features de media en barra
            BarraMAGX = [];
            BarraMAGY = [];
            BarraMAGZ = [];
            BarraACX = [];
            BarraACY = [];
            BarraACZ = [];
```

```
            numeroMuestras = Genomas(G,32);
            nmax = MaxTam/numeroMuestras;
            nmax=floor(nmax);

            for u=1:size(MagEjeX,1)                    %Para cada
vehículo

                n = Longitudes(u)/numeroMuestras;
                n=floor(n);
                if n==0
                    n=1;
                end

                %%%%%%%%%%%%%%%%%%%%%%
                if (Genomas(G,19))
                    if n<=1
                        auxMAGX = [];
                        aux = MagEjeX(u,1:Longitudes(u));
                        auxMAGX = [auxMAGX, mean(aux,2)];
                    else
                        auxMAGX = [];
                        aux = MagEjeX(u,1:(numeroMuestras));
                        auxMAGX = [auxMAGX, mean(aux,2)];
                        for i=1:n-2
                            aux =
MagEjeX(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                            auxMAGX = [auxMAGX, mean(aux,2)];
                        end
                        aux = MagEjeX(u,(numeroMuestras*(n-
1)):Longitudes(u));
                        auxMAGX = [auxMAGX, mean(aux,2)];
                    end
                     if n<nmax
                         BarraMAGX = [BarraMAGX; auxMAGX zeros(1,nmax-
n)];
                     else
                         BarraMAGX = [BarraMAGX; auxMAGX];
                     end
                end

                %%%%%%%%%%%%%%%%%%%%%%
                if (Genomas(G,20))
                   if n<=1
                        auxMAGY = [];
                        aux = MagEjeY(u,1:Longitudes(u));
                        auxMAGY = [auxMAGY, mean(aux,2)];
                   else
                        auxMAGY = [];
                        aux = MagEjeY(u,1:(numeroMuestras));
                        auxMAGY = [auxMAGY, mean(aux,2)];
                        for i=1:n-2
                            aux =
MagEjeY(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
```

```
                        auxMAGY = [auxMAGY, mean(aux,2)];
                    end
                    aux = MagEjeY(u,(numeroMuestras*(n-
1)):Longitudes(u));
                    auxMAGY = [auxMAGY, mean(aux,2)];
                end
                  if n<nmax
                      BarraMAGY = [BarraMAGY; auxMAGY zeros(1,nmax-
n)];
                  else
                      BarraMAGY = [BarraMAGY; auxMAGY];
                  end
            end
            %%%%%%%%%%%%%%%%%%%%%%%%
            if (Genomas(G,21))
                if n<=1
                    auxMAGZ = [];
                    aux = MagEjeZ(u,1:Longitudes(u));
                    auxMAGZ = [auxMAGZ, mean(aux,2)];
                else
                    auxMAGZ = [];
                    aux = MagEjeZ(u,1:(numeroMuestras));
                    auxMAGZ = [auxMAGZ, mean(aux,2)];
                    for i=1:n-2
                        aux =
MagEjeZ(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                        auxMAGZ = [auxMAGZ, mean(aux,2)];
                    end
                    aux = MagEjeZ(u,(numeroMuestras*(n-
1)):Longitudes(u));
                    auxMAGZ = [auxMAGZ, mean(aux,2)];
                end
                  if n<nmax
                      BarraMAGZ = [BarraMAGZ; auxMAGZ zeros(1,nmax-
n)];
                  else
                      BarraMAGZ = [BarraMAGZ; auxMAGZ];
                  end
            end
            %%%%%%%%%%%%%%%%%%%%%%%%
            if (Genomas(G,22))
                if n<=1
                    auxACX = [];
                    aux = ACEjeX(u,1:Longitudes(u));
                    auxACX = [auxACX, mean(aux,2)];
                else
                    auxACX = [];
                    aux = ACEjeX(u,1:(numeroMuestras));
                    auxACX = [auxACX, mean(aux,2)];
                    for i=1:n-2
                        aux =
ACEjeX(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                        auxACX = [auxACX, mean(aux,2)];
                    end
```

```
                        aux = ACEjeX(u,(numeroMuestras*(n-
1)):Longitudes(u));
                        auxACX = [auxACX, mean(aux,2)];
                    end
                      if n<nmax
                          BarraACX = [BarraACX; auxACX zeros(1,nmax-
n)];
                      else
                          BarraACX = [BarraACX; auxACX];
                      end
                end
                %%%%%%%%%%%%%%%%%%%%%%
                if (Genomas(G,23))
                   if n<=1
                        auxACY = [];
                        aux = ACEjeY(u,1:Longitudes(u));
                        auxACY = [auxACY, mean(aux,2)];
                   else
                        auxACY = [];
                        aux = ACEjeY(u,1:(numeroMuestras));
                        auxACY = [auxACY, mean(aux,2)];
                        for i=1:n-2
                            aux =
ACEjeY(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                            auxACY = [auxACY, mean(aux,2)];
                        end
                        aux = ACEjeY(u,(numeroMuestras*(n-
1)):Longitudes(u));
                        auxACY = [auxACY, mean(aux,2)];
                   end
                     if n<nmax
                         BarraACY = [BarraACY; auxACY zeros(1,nmax-
n)];
                     else
                         BarraACY = [BarraACY; auxACY];
                     end
                end
                %%%%%%%%%%%%%%%%%%%%%%
                if (Genomas(G,24))
                   if n<=1
                        auxACZ = [];
                        aux = ACEjeZ(u,1:Longitudes(u));
                        auxACZ = [auxACZ, mean(aux,2)];
                   else
                        auxACZ = [];
                        aux = ACEjeZ(u,1:(numeroMuestras));
                        auxACZ = [auxACZ, mean(aux,2)];
                        for i=1:n-2
                            aux =
ACEjeZ(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                            auxACZ = [auxACZ, mean(aux,2)];
                        end
                        aux = ACEjeZ(u,(numeroMuestras*(n-
1)):Longitudes(u));
                        auxACZ = [auxACZ, mean(aux,2)];
```

```
                    end
                      if n<nmax
                          BarraACZ = [BarraACZ; auxACZ zeros(1,nmax-
n)];
                      else
                          BarraACZ = [BarraACZ; auxACZ];
                      end
                 end
             end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%
        % Features de picos

        picosMAGX = [];
        picosMAGY = [];
        picosMAGZ = [];
        picosACX = [];
        picosACY = [];
        picosACZ = [];
        auxiliar = 0;
        numpicos = Genomas(G,33);

        for j=1:size(MagEjeX,1)

            %%%%%%%%%%%%%%%%%%%%%%%
            if (Genomas(G,25))
                auxMAGX = 0;
                for i=numpicos+1:size(MagEjeX,2)-(numpicos+1)
                    auxiliar = 1;
                    for d=1:numpicos
                        if(MagEjeX(j,i)<=MagEjeX(j,i+d))
                            auxiliar = 0;
                        end
                        if(MagEjeX(j,i)<=MagEjeX(j,i-d))
                            auxiliar = 0;
                        end
                    end
                    auxMAGX =  auxMAGX+auxiliar;
                end
                picosMAGX = [picosMAGX; auxMAGX];
            end
            %%%%%%%%%%%%%%%%%%%%%%%
            if (Genomas(G,26))
                auxMAGY = 0;
                for i=numpicos+1:size(MagEjeY,2)-(numpicos+1)
                    auxiliar = 1;
                    for d=1:numpicos
                        if(MagEjeY(j,i)<=MagEjeY(j,i+d))
                            auxiliar = 0;
                        end
                        if(MagEjeY(j,i)<=MagEjeY(j,i-d))
                            auxiliar = 0;
                        end
```

```
        end
        auxMAGY =  auxMAGY+auxiliar;
    end
    picosMAGY = [picosMAGY; auxMAGY];
end
%%%%%%%%%%%%%%%%%%%%%%%%
if (Genomas(G,27))
    auxMAGZ = 0;
    for i=numpicos+1:size(MagEjeZ,2)-(numpicos+1)
        auxiliar = 1;
        for d=1:numpicos
            if(MagEjeZ(j,i)<=MagEjeZ(j,i+d))
                auxiliar = 0;
            end
            if(MagEjeZ(j,i)<=MagEjeZ(j,i-d))
                auxiliar = 0;
            end
        end
        auxMAGZ =  auxMAGZ+auxiliar;
    end
    picosMAGZ = [picosMAGZ; auxMAGZ];
end

%%%%%%%%%%%%%%%%%%%%%%%%
if (Genomas(G,28))
    auxACX = 0;
    for i=numpicos+1:size(ACEjeX,2)-(numpicos+1)
        auxiliar = 1;
        for d=1:numpicos
            if(ACEjeX(j,i)<=ACEjeX(j,i+d))
                auxiliar = 0;
            end
            if(ACEjeX(j,i)<=ACEjeX(j,i-d))
                auxiliar = 0;
            end
        end
        auxACX =  auxACX+auxiliar;
    end
    picosACX = [picosACX; auxACX];
end

%%%%%%%%%%%%%%%%%%%%%%%%
if (Genomas(G,29))
    auxACY = 0;
    for i=numpicos+1:size(ACEjeY,2)-(numpicos+1)
        auxiliar = 1;
        for d=1:numpicos
            if(ACEjeY(j,i)<=ACEjeY(j,i+d))
                auxiliar = 0;
            end
            if(ACEjeY(j,i)<=ACEjeY(j,i-d))
                auxiliar = 0;
            end
        end
```

```
                auxACY =  auxACY+auxiliar;
            end
            picosACY = [picosACY; auxACY];
        end
        %%%%%%%%%%%%%%%%%%%%%%%
        if (Genomas(G,30))
            auxACZ = 0;
            for i=numpicos+1:size(ACEjeZ,2)-(numpicos+1)
                auxiliar = 1;
                for d=1:numpicos
                    if(ACEjeZ(j,i)<=ACEjeZ(j,i+d))
                        auxiliar = 0;
                    end
                    if(ACEjeZ(j,i)<=ACEjeZ(j,i-d))
                        auxiliar = 0;
                    end
                end
                auxACZ =  auxACZ+auxiliar;
            end
            picosACZ = [picosACZ; auxACZ];
        end
    end



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%
    %%Creacion de la Red SOM
    fprintf('Creacion de la Red SOM\n');

    data1 = [];
    names_variables = {};
    numindices = 0;
    nameMAGX = {};
    nameMAGY = {};
    nameMAGZ = {};

    nameACX = {};
    nameACY = {};
    nameACZ = {};

    for i=1:nmax
        nameMAGX{i} = strcat('BARRA MAGNETICO EJE X ',num2str(i));
        nameMAGY{i} = strcat('BARRA MAGNETICO EJE Y ',num2str(i));
        nameMAGZ{i} = strcat('BARRA MAGNETICO EJE Z ',num2str(i));

        nameACX{i} = strcat('BARRA ACELERACION EJE X
',num2str(i));
        nameACY{i} = strcat('BARRA ACELERACION EJE Y
',num2str(i));
```

```
            nameACZ{i} = strcat('BARRA ACELERACION EJE Z
',num2str(i));
        end

        if (Genomas(G,1))
            data1 = [data1; MAXMAGX'];
            names_variables = [names_variables, 'MAX MAGNETICO EJE
X'];
            numindices = numindices+1;
        end
        if (Genomas(G,2))
            data1 = [data1; MAXMAGY'];
            names_variables = [names_variables, 'MAX MAGNETICO EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,3))
            data1 = [data1; MAXMAGZ'];
            names_variables = [names_variables, 'MAX MAGNETICO EJE
Z'];
            numindices = numindices+1;
        end
         if (Genomas(G,4))
            data1 = [data1; MAXACEX'];
            names_variables = [names_variables, 'MAX ACELERACION EJE
X'];
            numindices = numindices+1;
        end
        if (Genomas(G,5))
            data1 = [data1; MAXACEY'];
            names_variables = [names_variables, 'MAX ACELERACION EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,6))
            data1 = [data1; MAXACEZ'];
            names_variables = [names_variables, 'MAX ACELERACION EJE
Z'];
            numindices = numindices+1;
        end



        if (Genomas(G,7))
            data1 = [data1; MINMAGX'];
            names_variables = [names_variables, 'MIN MAGNETICO EJE
X'];
            numindices = numindices+1;
        end
        if (Genomas(G,8))
            data1 = [data1; MINMAGY'];
            names_variables = [names_variables, 'MIN MAGNETICO EJE
Y'];
            numindices = numindices+1;
```

```
        end
        if (Genomas(G,9))
            data1 = [data1; MINMAGZ'];
            names_variables = [names_variables, 'MIN MAGNETICO EJE
Z'];
            numindices = numindices+1;
        end
         if (Genomas(G,10))
            data1 = [data1; MINACEX'];
            names_variables = [names_variables, 'MIN ACELERACION EJE
X'];
            numindices = numindices+1;
        end
        if (Genomas(G,11))
            data1 = [data1; MINACEY'];
            names_variables = [names_variables, 'MIN ACELERACION EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,12))
            data1 = [data1; MINACEZ'];
            names_variables = [names_variables, 'MIN ACELERACION EJE
Z'];
            numindices = numindices+1;
        end



        if (Genomas(G,13))
            data1 = [data1; MEANMAGX'];
            names_variables = [names_variables, 'MEDIA MAGNETICO EJE
X'];
            numindices = numindices+1;
        end
        if (Genomas(G,14))
            data1 = [data1; MEANMAGY'];
            names_variables = [names_variables, 'MEDIA MAGNETICO EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,15))
            data1 = [data1; MEANMAGZ'];
            names_variables = [names_variables, 'MEDIA MAGNETICO EJE
Z'];
            numindices = numindices+1;
        end
         if (Genomas(G,16))
            data1 = [data1; MEANACEX'];
            names_variables = [names_variables, 'MEDIA ACELERACION EJE
X'];
            numindices = numindices+1;
        end
        if (Genomas(G,17))
            data1 = [data1; MEANACEY'];
```

```
            names_variables = [names_variables, 'MEDIA ACELERACION EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,18))
            data1 = [data1; MEANACEZ'];
            names_variables = [names_variables, 'MEDIA ACELERACION EJE
Z'];
            numindices = numindices+1;
        end



        if (Genomas(G,19))
            data1 = [data1; BarraMAGX'];
            names_variables = [names_variables, nameMAGX];
            numindices = numindices + size(BarraMAGX,2);
        end
        if (Genomas(G,20))
            data1 = [data1; BarraMAGY'];
            names_variables = [names_variables, nameMAGY];
            numindices = numindices + size(BarraMAGY,2);
        end
        if (Genomas(G,21))
            data1 = [data1; BarraMAGZ'];
            names_variables = [names_variables, nameMAGZ];
            numindices = numindices + size(BarraMAGZ,2);
        end
         if (Genomas(G,22))
            data1 = [data1; BarraACX'];
            names_variables = [names_variables, nameACX];
            numindices = numindices + size(BarraACX,2);
        end
        if (Genomas(G,23))
            data1 = [data1; BarraACY'];
            names_variables = [names_variables, nameACY];
            numindices = numindices + size(BarraACY,2);
        end
        if (Genomas(G,24))
            data1 = [data1; BarraACZ'];
            names_variables = [names_variables, nameACZ];
            numindices = numindices + size(BarraACZ,2);
        end



        if (Genomas(G,25))
            data1 = [data1; picosMAGX'];
            names_variables = [names_variables, 'PICOS MAGNETICO EJE
X'];
            numindices = numindices+1;
        end
        if (Genomas(G,26))
```

```
            data1 = [data1; picosMAGY'];
            names_variables = [names_variables, 'PICOS MAGNETICO EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,27))
            data1 = [data1; picosMAGZ'];
            names_variables = [names_variables, 'PICOS MAGNETICO EJE
Z'];
            numindices = numindices+1;
        end
         if (Genomas(G,28))
            data1 = [data1; picosACX'];
            names_variables = [names_variables, 'PICOS ACELERACION EJE
X'];
            numindices = numindices+1;
        end
        if (Genomas(G,29))
            data1 = [data1; picosACY'];
            names_variables = [names_variables, 'PICOS ACELERACION EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,30))
            data1 = [data1; picosACZ'];
            names_variables = [names_variables, 'PICOS ACELERACION EJE
Z'];
            numindices = numindices+1;
        end
        if (Genomas(G,31))
            data1 = [data1; Longitudes'];
            names_variables = [names_variables, 'LONGITUD MUESTRA'];
            numindices = numindices+1;
        end

        indices = 1:1:numindices;

        clase = [];
        data = [];
        GG = 0;
        M = 0;
        P = 0;
        MP = 0;
        GM = 0;

        fprintf('Creacion del fichero de detecciones\n');

        for i=1:size(Detecciones,1)

            GM = 0;
            if (Detecciones(i).deteccion.Longitud  < 3600)
                clase = [ 'P'; clase];
                data = [data1(:,i) data];
                P = P+1;
```

```
            else
                if (Detecciones(i).deteccion.Longitud  < 4500)
                    clase = [clase; 'M'];
                    data = [data data1(:,i)];
                    M = M+1;
                else
                    clase = [clase; 'G'];
                    data = [data data1(:,i)];
                    GG = GG+1;
                end
            end
        end
        data = double(data);

        numclases = 2;
        numpatclases = [ M GG];
        name_clases = {  'MEDIANO', 'GRANDE' };

        nombre_mapa = 'SOM ANALISIS DATOS';
        [ numinpnomask , npat ] = size( data );

        sD = som_data_struct( data' , 'name' , nombre_mapa ,
'comp_names' , names_variables , 'labels' ,  clase  );

        fprintf('Normalizamos\n');
        % Normalizamos cada variable de entrada independientemente en
el rango [0 1]
        sD = som_normalize( sD, 'range' );

        lattice = 'hexa';   %% Tipo de mapa  'hexa' = hexagonal,
'rect' = rectangular
        mapsize = [13 8];    %% Tamaño del mapa

        sM = som_make(sD, 'msize', mapsize, 'lattice', lattice);


        [ unit_class_index , unit_class_label] =
som_unit_classification( sM , sD , clase);
        sM.labels = unit_class_label;

        [ data_class_index , data_class_label ] =
som_data_classification( sM , sD , unit_class_index );

        [ conf_mat , Kappa ] = confussion_matrix( data_class_index ,
clase );
        fprintf('\nKappa: %1.10f ' , Kappa  );
        Kappas = [Kappas; Kappa];

    end

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %% Ordenar los Kappas
```

```
    fprintf('Ordenar los Kappas\n');
    KappasOrdenadas = [];
    GenomasOrdenados = [];
    KappasOrdenadas = [KappasOrdenadas;Kappas(1)];
    GenomasOrdenados = [GenomasOrdenados;Genomas(1,1:33)];

    for i=2:size(Kappas,1)
        bucle=1;
        q = 1;
        while (bucle==1)
             if (Kappas(i) >= KappasOrdenadas(q))
                 KappasOrdenadas = [KappasOrdenadas(1:(q-1));
Kappas(i); KappasOrdenadas(q:size(KappasOrdenadas,1))];
                 GenomasOrdenados = [GenomasOrdenados(1:(q-1),1:33);
Genomas(i,1:33); GenomasOrdenados(q:size(GenomasOrdenados,1),1:33)];
                 bucle = 0;
             else
                 q = q+1;
             end
             if (q>size(KappasOrdenadas,1))
                 KappasOrdenadas = [KappasOrdenadas;Kappas(i)];
                 GenomasOrdenados =
[GenomasOrdenados;Genomas(i,1:33)];
                 bucle = 0;
             end
        end
    end
    KappaMax = [KappaMax; KappasOrdenadas(1)];

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %% Generar nueva generacion
    fprintf('Generar nueva generacion\n');
    NuevaGeneracion = [];


    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %  Elitismo, replica de los mejores

    NuevaGeneracion =
GenomasOrdenados(1:20,1:size(GenomasOrdenados,2));

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %  Reproduccion

    q = 1;
    while(size(NuevaGeneracion,1)<80)
        if (randi([0,1000],1,1)<(KappasOrdenadas(q)*1000))
            corte = randi([1,33],1,1);
            auxiliar = [GenomasOrdenados(q,1:corte),
GenomasOrdenados(randi([1,20],1,1),corte+1:33)];
            NuevaGeneracion = [NuevaGeneracion; auxiliar];
        else
            if (q<20)
```

```
                q = q+1;
            else
                q = 1;
            end
        end

    end

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %  Mutacion

    for i=1:20
        mutacion = randi([1,33],1,1);
        if (mutacion>31)
            auxiliar = GenomasOrdenados(i,1:33);
            auxiliar(mutacion) = randi([1,400],1,1);
        else
            auxiliar = GenomasOrdenados(i,1:33);
            auxiliar(mutacion) = randi([0,1],1,1);
        end
        NuevaGeneracion = [NuevaGeneracion; auxiliar];
    end

    if (Ciclos > 1)
        Ciclos = Ciclos -1;
    else
        fprintf('\n');
        fprintf('\n');
        fprintf('\n');
        fprintf('FINAL DE LA GENERACION\n');
        pause();
        t = 1:1:size(KappaMax);
        figure();
        plot(t,KappaMax);
        Ciclos = input('Numero de ciclos a simular: ');
     end
  end
```

2. AG PARA DIRECCIÓN

```
clear all;
close all;
fprintf('Analisis completo de las Features mediante Algoritmos
Geneticos\n');
fprintf('Para muestras capturadas automaticamente\n');
fprintf('Para el DIRECCIÓN\n');


%% CONFIGURAR
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
nombreArchivo = 'C:\Users\Ruben\Documents\universidad\Master\Trabajo
Final de Master\Base de datos\Datos-Detectados-
Automaticos\DatosAutomaticos.mat';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
Ciclos = 200;
%%%%%%%%%%%%%%%%%%%%%%%%%%

%% CARGA DE DATOS
load(nombreArchivo');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Extraccion de las features
fprintf('Extraccion de las features\n');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Features estadisticasa

MAXMAGX= max(MagEjeX,[],2);
MAXMAGY= max(MagEjeY,[],2);
MAXMAGZ= max(MagEjeZ,[],2);
MAXACEX= max(ACEjeX,[],2);
MAXACEY= max(ACEjeY,[],2);
MAXACEZ= max(ACEjeZ,[],2);

MINMAGX= minimos(MagEjeX,Longitudes);
MINMAGY= minimos(MagEjeY,Longitudes);
MINMAGZ= minimos(MagEjeZ,Longitudes);
MINACEX= minimos(ACEjeX,Longitudes);
MINACEY= minimos(ACEjeY,Longitudes);
MINACEZ= minimos(ACEjeZ,Longitudes);

MEANMAGX= medias(MagEjeX,Longitudes);
MEANMAGY= medias(MagEjeY,Longitudes);
MEANMAGZ= medias(MagEjeZ,Longitudes);
MEANACEX= medias(ACEjeX,Longitudes);
MEANACEY= medias(ACEjeY,Longitudes);
MEANACEZ= medias(ACEjeZ,Longitudes);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Creacion de los genomas iniciales
fprintf('Creacion de los genomas iniciales\n');

NuevaGeneracion = randi([0,1],100,31);
auxi = randi([2,400],100,2);
NuevaGeneracion = [NuevaGeneracion, auxi];
KappaMax = [];

MaxTam = max(Longitudes,[],1)


            %%%%%%%%%%%%%%%%%%%%%%%%%%
%      1     MAXMAGX       MAXMAGY       MAXMAGZ       MAXACEX
MAXACEY       MAXACEZ
%      7     MINMAGX       MINMAGY       MINMAGZ       MINACEX
MINACEY       MINACEZ
```

```
%      13    MEANMAGX      MEANMAGY      MEANMAGZ      MEANACEX
MEANACEY      MEANACEZ
%      19    BarraMAGX     BarraMAGY     BarraMAGZ     BarraACX
BarraACY      BarraACZ
%      25    picosMAGX     picosMAGY     picosMAGZ     picosACX
picosACY      picosACZ
%      31    Longitud      NumBarras     NumPicos

% En caso de querer meter reproductores

 Reproductor1 = [0 0 1 1 1 0 1 0 0 0 0 1 0 1 0 0 0 0 1 1 0 0 0 0 1 1 1
1 1 0 1 23  362];
 Reproductor2 = [0 0 1 1 1 0 1 0 0 0 0 1 0 1 0 0 0 0 1 1 0 0 0 0 1 1 1
1 0 1 1 23  369];
 Reproductor3 = [0 0 1 1 1 0 1 0 0 0 0 1 0 1 0 0 0 0 1 1 0 0 0 0 1 1 1
1 0 1 1 23  369];
 Reproductor4 = [1 1 1 1 1 0 1 0 0 1 1 1 1 1 0 0 1 0 1 1 0 0 1 1 0 1 1
1 1 0 0 143 376];

 NuevaGeneracion = [NuevaGeneracion; Reproductor1; Reproductor2;
Reproductor3; Reproductor4];

while(Ciclos > 0)
  Genomas =   NuevaGeneracion;
  fprintf('Ciclos:\n');
  Ciclos
  Kappas = [];

for G=1:size(Genomas,1)
        % Features de media en barra
        BarraMAGX = [];
        BarraMAGY = [];
        BarraMAGZ = [];
        BarraACX = [];
        BarraACY = [];
        BarraACZ = [];

        numeroMuestras = Genomas(G,32);
        nmax = MaxTam/numeroMuestras;
        nmax=floor(nmax);

        for u=1:size(MagEjeX,1)                    %Para cada
vehículo

            n = Longitudes(u)/numeroMuestras;
            n=floor(n);
            if n==0
                n=1;
            end

            %%%%%%%%%%%%%%%%%%%%%%
            if (Genomas(G,19))
                if n<=1
```

```
                    auxMAGX = [];
                    aux = MagEjeX(u,1:Longitudes(u));
                    auxMAGX = [auxMAGX, mean(aux,2)];
                else
                    auxMAGX = [];
                    aux = MagEjeX(u,1:(numeroMuestras));
                    auxMAGX = [auxMAGX, mean(aux,2)];
                    for i=1:n-2
                        aux =
MagEjeX(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                        auxMAGX = [auxMAGX, mean(aux,2)];
                    end
                    aux = MagEjeX(u,(numeroMuestras*(n-
1)):Longitudes(u));
                    auxMAGX = [auxMAGX, mean(aux,2)];
                end
                 if n<nmax
                     BarraMAGX = [BarraMAGX; auxMAGX zeros(1,nmax-n)];
                 else
                     BarraMAGX = [BarraMAGX; auxMAGX];
                 end
            end

            %%%%%%%%%%%%%%%%%%%%%%%
            if (Genomas(G,20))
               if n<=1
                    auxMAGY = [];
                    aux = MagEjeY(u,1:Longitudes(u));
                    auxMAGY = [auxMAGY, mean(aux,2)];
               else
                    auxMAGY = [];
                    aux = MagEjeY(u,1:(numeroMuestras));
                    auxMAGY = [auxMAGY, mean(aux,2)];
                    for i=1:n-2
                        aux =
MagEjeY(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                        auxMAGY = [auxMAGY, mean(aux,2)];
                    end
                    aux = MagEjeY(u,(numeroMuestras*(n-
1)):Longitudes(u));
                    auxMAGY = [auxMAGY, mean(aux,2)];
               end
                 if n<nmax
                     BarraMAGY = [BarraMAGY; auxMAGY zeros(1,nmax-n)];
                 else
                     BarraMAGY = [BarraMAGY; auxMAGY];
                 end
            end
            %%%%%%%%%%%%%%%%%%%%%%%
            if (Genomas(G,21))
               if n<=1
                    auxMAGZ = [];
                    aux = MagEjeZ(u,1:Longitudes(u));
                    auxMAGZ = [auxMAGZ, mean(aux,2)];
```

```
                else
                     auxMAGZ = [];
                     aux = MagEjeZ(u,1:(numeroMuestras));
                     auxMAGZ = [auxMAGZ, mean(aux,2)];
                     for i=1:n-2
                         aux =
MagEjeZ(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                         auxMAGZ = [auxMAGZ, mean(aux,2)];
                     end
                     aux = MagEjeZ(u,(numeroMuestras*(n-
1)):Longitudes(u));
                     auxMAGZ = [auxMAGZ, mean(aux,2)];
                end
                  if n<nmax
                      BarraMAGZ = [BarraMAGZ; auxMAGZ zeros(1,nmax-n)];
                  else
                      BarraMAGZ = [BarraMAGZ; auxMAGZ];
                  end
            end
            %%%%%%%%%%%%%%%%%%%%%%%%
            if (Genomas(G,22))
               if n<=1
                     auxACX = [];
                     aux = ACEjeX(u,1:Longitudes(u));
                     auxACX = [auxACX, mean(aux,2)];
               else
                     auxACX = [];
                     aux = ACEjeX(u,1:(numeroMuestras));
                     auxACX = [auxACX, mean(aux,2)];
                     for i=1:n-2
                         aux =
ACEjeX(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                         auxACX = [auxACX, mean(aux,2)];
                     end
                     aux = ACEjeX(u,(numeroMuestras*(n-
1)):Longitudes(u));
                     auxACX = [auxACX, mean(aux,2)];
                end
                  if n<nmax
                      BarraACX = [BarraACX; auxACX zeros(1,nmax-n)];
                  else
                      BarraACX = [BarraACX; auxACX];
                  end
            end
            %%%%%%%%%%%%%%%%%%%%%%%%
            if (Genomas(G,23))
               if n<=1
                     auxACY = [];
                     aux = ACEjeY(u,1:Longitudes(u));
                     auxACY = [auxACY, mean(aux,2)];
               else
                     auxACY = [];
                     aux = ACEjeY(u,1:(numeroMuestras));
                     auxACY = [auxACY, mean(aux,2)];
                     for i=1:n-2
```

```
                        aux =
ACEjeY(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                        auxACY = [auxACY, mean(aux,2)];
                    end
                    aux = ACEjeY(u,(numeroMuestras*(n-
1)):Longitudes(u));
                    auxACY = [auxACY, mean(aux,2)];
               end
                 if n<nmax
                     BarraACY = [BarraACY; auxACY zeros(1,nmax-n)];
                 else
                     BarraACY = [BarraACY; auxACY];
                 end
            end
            %%%%%%%%%%%%%%%%%%%%%%%%
            if (Genomas(G,24))
               if n<=1
                    auxACZ = [];
                    aux = ACEjeZ(u,1:Longitudes(u));
                    auxACZ = [auxACZ, mean(aux,2)];
               else
                    auxACZ = [];
                    aux = ACEjeZ(u,1:(numeroMuestras));
                    auxACZ = [auxACZ, mean(aux,2)];
                    for i=1:n-2
                        aux =
ACEjeZ(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                        auxACZ = [auxACZ, mean(aux,2)];
                    end
                    aux = ACEjeZ(u,(numeroMuestras*(n-
1)):Longitudes(u));
                    auxACZ = [auxACZ, mean(aux,2)];
               end
                 if n<nmax
                     BarraACZ = [BarraACZ; auxACZ zeros(1,nmax-n)];
                 else
                     BarraACZ = [BarraACZ; auxACZ];
                 end
            end
        end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%
    % Features de picos

    picosMAGX = [];
    picosMAGY = [];
    picosMAGZ = [];
    picosACX = [];
    picosACY = [];
    picosACZ = [];
    auxiliar = 0;
    numpicos = Genomas(G,33);
```

```
for j=1:size(MagEjeX,1)

    %%%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(G,25))
        auxMAGX = 0;
        for i=numpicos+1:size(MagEjeX,2)-(numpicos+1)
            auxiliar = 1;
            for d=1:numpicos
                if(MagEjeX(j,i)<=MagEjeX(j,i+d))
                    auxiliar = 0;
                end
                if(MagEjeX(j,i)<=MagEjeX(j,i-d))
                    auxiliar = 0;
                end
            end
            auxMAGX =  auxMAGX+auxiliar;
        end
        picosMAGX = [picosMAGX; auxMAGX];
    end
    %%%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(G,26))
        auxMAGY = 0;
        for i=numpicos+1:size(MagEjeY,2)-(numpicos+1)
            auxiliar = 1;
            for d=1:numpicos
                if(MagEjeY(j,i)<=MagEjeY(j,i+d))
                    auxiliar = 0;
                end
                if(MagEjeY(j,i)<=MagEjeY(j,i-d))
                    auxiliar = 0;
                end
            end
            auxMAGY =  auxMAGY+auxiliar;
        end
        picosMAGY = [picosMAGY; auxMAGY];
    end
    %%%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(G,27))
        auxMAGZ = 0;
        for i=numpicos+1:size(MagEjeZ,2)-(numpicos+1)
            auxiliar = 1;
            for d=1:numpicos
                if(MagEjeZ(j,i)<=MagEjeZ(j,i+d))
                    auxiliar = 0;
                end
                if(MagEjeZ(j,i)<=MagEjeZ(j,i-d))
                    auxiliar = 0;
                end
            end
            auxMAGZ =  auxMAGZ+auxiliar;
        end
        picosMAGZ = [picosMAGZ; auxMAGZ];
    end
```

```
        %%%%%%%%%%%%%%%%%%%%%%%%
        if (Genomas(G,28))
            auxACX = 0;
            for i=numpicos+1:size(ACEjeX,2)-(numpicos+1)
                auxiliar = 1;
                for d=1:numpicos
                    if(ACEjeX(j,i)<=ACEjeX(j,i+d))
                        auxiliar = 0;
                    end
                    if(ACEjeX(j,i)<=ACEjeX(j,i-d))
                        auxiliar = 0;
                    end
                end
                auxACX =  auxACX+auxiliar;
            end
            picosACX = [picosACX; auxACX];
        end

        %%%%%%%%%%%%%%%%%%%%%%%%
        if (Genomas(G,29))
            auxACY = 0;
            for i=numpicos+1:size(ACEjeY,2)-(numpicos+1)
                auxiliar = 1;
                for d=1:numpicos
                    if(ACEjeY(j,i)<=ACEjeY(j,i+d))
                        auxiliar = 0;
                    end
                    if(ACEjeY(j,i)<=ACEjeY(j,i-d))
                        auxiliar = 0;
                    end
                end
                auxACY =  auxACY+auxiliar;
            end
            picosACY = [picosACY; auxACY];
        end
        %%%%%%%%%%%%%%%%%%%%%%%%
        if (Genomas(G,30))
            auxACZ = 0;
            for i=numpicos+1:size(ACEjeZ,2)-(numpicos+1)
                auxiliar = 1;
                for d=1:numpicos
                    if(ACEjeZ(j,i)<=ACEjeZ(j,i+d))
                        auxiliar = 0;
                    end
                    if(ACEjeZ(j,i)<=ACEjeZ(j,i-d))
                        auxiliar = 0;
                    end
                end
                auxACZ =  auxACZ+auxiliar;
            end
            picosACZ = [picosACZ; auxACZ];
        end
    end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%
        %%Creacion de la Red SOM
        fprintf('Creacion de la Red SOM\n');

        data1 = [];
        names_variables = {};
        numindices = 0;
        nameMAGX = {};
        nameMAGY = {};
        nameMAGZ = {};

        nameACX = {};
        nameACY = {};
        nameACZ = {};

        for i=1:nmax
            nameMAGX{i} = strcat('BARRA MAGNETICO EJE X ',num2str(i));
            nameMAGY{i} = strcat('BARRA MAGNETICO EJE Y ',num2str(i));
            nameMAGZ{i} = strcat('BARRA MAGNETICO EJE Z ',num2str(i));

            nameACX{i} = strcat('BARRA ACELERACION EJE X
',num2str(i));
            nameACY{i} = strcat('BARRA ACELERACION EJE Y
',num2str(i));
            nameACZ{i} = strcat('BARRA ACELERACION EJE Z
',num2str(i));
        end

        if (Genomas(G,1))
            data1 = [data1; MAXMAGX'];
            names_variables = [names_variables, 'MAX MAGNETICO EJE
X'];
            numindices = numindices+1;
        end
        if (Genomas(G,2))
            data1 = [data1; MAXMAGY'];
            names_variables = [names_variables, 'MAX MAGNETICO EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,3))
            data1 = [data1; MAXMAGZ'];
            names_variables = [names_variables, 'MAX MAGNETICO EJE
Z'];
            numindices = numindices+1;
        end
         if (Genomas(G,4))
            data1 = [data1; MAXACEX'];
            names_variables = [names_variables, 'MAX ACELERACION EJE
X'];
            numindices = numindices+1;
        end
```

```
        if (Genomas(G,5))
            data1 = [data1; MAXACEY'];
            names_variables = [names_variables, 'MAX ACELERACION EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,6))
            data1 = [data1; MAXACEZ'];
            names_variables = [names_variables, 'MAX ACELERACION EJE
Z'];
            numindices = numindices+1;
        end

        if (Genomas(G,7))
            data1 = [data1; MINMAGX'];
            names_variables = [names_variables, 'MIN MAGNETICO EJE
X'];
            numindices = numindices+1;
        end
        if (Genomas(G,8))
            data1 = [data1; MINMAGY'];
            names_variables = [names_variables, 'MIN MAGNETICO EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,9))
            data1 = [data1; MINMAGZ'];
            names_variables = [names_variables, 'MIN MAGNETICO EJE
Z'];
            numindices = numindices+1;
        end
         if (Genomas(G,10))
            data1 = [data1; MINACEX'];
            names_variables = [names_variables, 'MIN ACELERACION EJE
X'];
            numindices = numindices+1;
        end
        if (Genomas(G,11))
            data1 = [data1; MINACEY'];
            names_variables = [names_variables, 'MIN ACELERACION EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,12))
            data1 = [data1; MINACEZ'];
            names_variables = [names_variables, 'MIN ACELERACION EJE
Z'];
            numindices = numindices+1;
        end

        if (Genomas(G,13))
            data1 = [data1; MEANMAGX'];
            names_variables = [names_variables, 'MEDIA MAGNETICO EJE
X'];
```

```
            numindices = numindices+1;
        end
        if (Genomas(G,14))
            data1 = [data1; MEANMAGY'];
            names_variables = [names_variables, 'MEDIA MAGNETICO EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,15))
            data1 = [data1; MEANMAGZ'];
            names_variables = [names_variables, 'MEDIA MAGNETICO EJE
Z'];
            numindices = numindices+1;
        end
         if (Genomas(G,16))
            data1 = [data1; MEANACEX'];
            names_variables = [names_variables, 'MEDIA ACELERACION EJE
X'];
            numindices = numindices+1;
        end
        if (Genomas(G,17))
            data1 = [data1; MEANACEY'];
            names_variables = [names_variables, 'MEDIA ACELERACION EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,18))
            data1 = [data1; MEANACEZ'];
            names_variables = [names_variables, 'MEDIA ACELERACION EJE
Z'];
            numindices = numindices+1;
        end

        if (Genomas(G,19))
            data1 = [data1; BarraMAGX'];
            names_variables = [names_variables, nameMAGX];
            numindices = numindices + size(BarraMAGX,2);
        end
        if (Genomas(G,20))
            data1 = [data1; BarraMAGY'];
            names_variables = [names_variables, nameMAGY];
            numindices = numindices + size(BarraMAGY,2);
        end
        if (Genomas(G,21))
            data1 = [data1; BarraMAGZ'];
            names_variables = [names_variables, nameMAGZ];
            numindices = numindices + size(BarraMAGZ,2);
        end
         if (Genomas(G,22))
            data1 = [data1; BarraACX'];
            names_variables = [names_variables, nameACX];
            numindices = numindices + size(BarraACX,2);
        end
        if (Genomas(G,23))
```

```
            data1 = [data1; BarraACY'];
            names_variables = [names_variables, nameACY];
            numindices = numindices + size(BarraACY,2);
        end
        if (Genomas(G,24))
            data1 = [data1; BarraACZ'];
            names_variables = [names_variables, nameACZ];
            numindices = numindices + size(BarraACZ,2);
        end

        if (Genomas(G,25))
            data1 = [data1; picosMAGX'];
            names_variables = [names_variables, 'PICOS MAGNETICO EJE
X'];
            numindices = numindices+1;
        end
        if (Genomas(G,26))
            data1 = [data1; picosMAGY'];
            names_variables = [names_variables, 'PICOS MAGNETICO EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,27))
            data1 = [data1; picosMAGZ'];
            names_variables = [names_variables, 'PICOS MAGNETICO EJE
Z'];
            numindices = numindices+1;
        end
         if (Genomas(G,28))
            data1 = [data1; picosACX'];
            names_variables = [names_variables, 'PICOS ACELERACION EJE
X'];
            numindices = numindices+1;
        end
        if (Genomas(G,29))
            data1 = [data1; picosACY'];
            names_variables = [names_variables, 'PICOS ACELERACION EJE
Y'];
            numindices = numindices+1;
        end
        if (Genomas(G,30))
            data1 = [data1; picosACZ'];
            names_variables = [names_variables, 'PICOS ACELERACION EJE
Z'];
            numindices = numindices+1;
        end
        if (Genomas(G,31))
            data1 = [data1; Longitudes'];
            names_variables = [names_variables, 'LONGITUD MUESTRA'];
            numindices = numindices+1;
        end

        indices = 1:1:numindices;
```

```
        clase = [];
        data = [];
        num = 0;
        fprintf('Creacion del fichero de direcciones\n');
        for i=1:size(Detecciones,1)
            if (Detecciones(i).deteccion.Direccion(2) == 'S')
                clase = [ 'S'; clase];
                data = [data1(:,i) data];
                num = num+1;
            else
                clase = [clase; 'E'];
                data = [data data1(:,i)];
            end
        end

        data = double(data);
        numclases = 2;
        numpatclases = [ num 113-num ];
        name_clases = { 'Salida'  ,  'Entrada' };

        nombre_mapa = 'SOM ANALISIS DATOS';
        [ numinpnomask , npat ] = size( data );

        sD = som_data_struct( data' , 'name' , nombre_mapa ,
'comp_names' , names_variables , 'labels' ,  clase  );

        fprintf('Normalizamos\n');
        % Normalizamos cada variable de entrada independientemente en
el rango [0 1]
        sD = som_normalize( sD, 'range' );

        lattice = 'hexa';   %% Tipo de mapa  'hexa' = hexagonal,
'rect' = rectangular
        mapsize = [13 8];    %% Tamaño del mapa

        sM = som_make(sD, 'msize', mapsize, 'lattice', lattice);


        [ unit_class_index , unit_class_label] =
som_unit_classification( sM , sD , clase);
        sM.labels = unit_class_label;

        [ data_class_index , data_class_label ] =
som_data_classification( sM , sD , unit_class_index );

        % Calculo de las Kappas
        [ conf_mat , Kappa ] = confussion_matrix( data_class_index ,
clase );
        fprintf('\nKappa: %1.10f ' , Kappa  );
        Kappas = [Kappas; Kappa];

    end
```

```
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %% Ordenar los Kappas
    fprintf('Ordenar los Kappas\n');
    KappasOrdenadas = [];
    GenomasOrdenados = [];
    KappasOrdenadas = [KappasOrdenadas;Kappas(1)];
    GenomasOrdenados = [GenomasOrdenados;Genomas(1,1:33)];

    for i=2:size(Kappas,1)
        bucle=1;
        q = 1;
        while (bucle==1)
             if (Kappas(i) >= KappasOrdenadas(q))
                 KappasOrdenadas = [KappasOrdenadas(1:(q-1));
Kappas(i); KappasOrdenadas(q:size(KappasOrdenadas,1))];
                 GenomasOrdenados = [GenomasOrdenados(1:(q-1),1:33);
Genomas(i,1:33); GenomasOrdenados(q:size(GenomasOrdenados,1),1:33)];
                 bucle = 0;
             else
                 q = q+1;
             end
             if (q>size(KappasOrdenadas,1))
                 KappasOrdenadas = [KappasOrdenadas;Kappas(i)];
                 GenomasOrdenados =
[GenomasOrdenados;Genomas(i,1:33)];
                 bucle = 0;
             end
        end
    end
    KappaMax = [KappaMax; KappasOrdenadas(1)];

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %% Generar nueva generacion
    fprintf('Generar nueva generacion\n');
    NuevaGeneracion = [];

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %  Elitismo, replica de los mejores

    NuevaGeneracion =
GenomasOrdenados(1:20,1:size(GenomasOrdenados,2));

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %  Reproduccion

    q = 1;
    while(size(NuevaGeneracion,1)<80)
        if (randi([0,1000],1,1)<(KappasOrdenadas(q)*1000))
            corte = randi([1,33],1,1);
            auxiliar = [GenomasOrdenados(q,1:corte),
GenomasOrdenados(randi([1,20],1,1),corte+1:33)];
            NuevaGeneracion = [NuevaGeneracion; auxiliar];
```

```
        else
            if (q<20)
                q = q+1;
            else
                q = 1;
            end
        end

    end

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %  Mutacion

    for i=1:20
        mutacion = randi([1,33],1,1);
        if (mutacion>31)
            auxiliar = GenomasOrdenados(i,1:33);
            auxiliar(mutacion) = randi([1,400],1,1);
        else
            auxiliar = GenomasOrdenados(i,1:33);
            auxiliar(mutacion) = randi([0,1],1,1);
        end
        NuevaGeneracion = [NuevaGeneracion; auxiliar];
    end

    if (Ciclos > 1)
        Ciclos = Ciclos -1;
    else
        fprintf('\n');
        fprintf('\n');
        fprintf('\n');
        fprintf('FINAL DE LA GENERACION\n');
        pause();
        t = 1:1:size(KappaMax);
        figure();
        plot(t,KappaMax);
        Ciclos = input('Numero de ciclos a simular: ');
     end


end
```

ANEXO 6. ALGORITMO ELECCIÓN ARQUITECTURA

1. ALGORITMO PARA TAMAÑO

```
clear all;
close all;
fprintf('MLP \n');
fprintf('Analisis de diferentes arquitecturas de MLP \n');
fprintf('Para muestras capturadas automaticamente\n');
fprintf('Para el TAMAÑO\n');



%% CONFIGURACION
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
nombreArchivo = 'C:\Users\Ruben\Documents\universidad\Master\Trabajo
Final de Master\Base de datos\Datos-Detectados-
Automaticos\DatosAutomaticos.mat';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
n_pool = 10;  % NÚMERO DE REPETICIONES DE CADA REDv
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Valor del umbral con el que comparar las salidas de la red
umbral = 0.1;  % umbral para seleccionar una salida de red como no
nula
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Número máximo de ciclos de entrenamiento
% (disminuirlo para los metodos cuasi-Newton y de grad.conjugados)
numciclos = 100;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Genomas = [0 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 1 0 1 0 1 1 0 0 1 1
1 362 23];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
NeuroMAX = 22;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
funtrain = 'trainlm'; % por defecto L-M



%% CARGA DE DATOS
load(nombreArchivo');

%% CARGA DE COMBINACIONES DE ARQUITECTURAS

% Unidades Ocultas en las diferentes arquitecturas
    AA = 1;
    AB = 0;
    nhunit = [ AA  AB ];
    nCapas = [];
    nCapas = [nCapas 1];
while AB<=NeuroMAX
```

```
    AA = AA+1;
    if AA > NeuroMAX
        AA = 1;
        AB = AB+1;
    end

    nhunit = [nhunit; AA  AB ];
    if AB == 0
        nCapas = [nCapas 1];
    else
        nCapas = [nCapas 2];
    end
end
% nhunit = [ 20 19 18 17 16; 0 0 0 12 10];
% nCapas = [ 1 1 1 2 2];
nhunit = nhunit';


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% VARIABLES

KappasOrdenadas = [];
IndicesOrdenados = [];
Kappas = [];
Indices = [];
MaxTam = max(Longitudes,[],1)



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Extraccion de las features
fprintf('Extraccion de las features\n');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Features estadisticas

MAXMAGX= max(MagEjeX,[],2);
MAXMAGY= max(MagEjeY,[],2);
MAXMAGZ= max(MagEjeZ,[],2);
MAXACEX= max(ACEjeX,[],2);
MAXACEY= max(ACEjeY,[],2);
MAXACEZ= max(ACEjeZ,[],2);

MINMAGX= minimos(MagEjeX,Longitudes);
MINMAGY= minimos(MagEjeY,Longitudes);
MINMAGZ= minimos(MagEjeZ,Longitudes);
MINACEX= minimos(ACEjeX,Longitudes);
MINACEY= minimos(ACEjeY,Longitudes);
MINACEZ= minimos(ACEjeZ,Longitudes);

MEANMAGX= medias(MagEjeX,Longitudes);
MEANMAGY= medias(MagEjeY,Longitudes);
```

```
MEANMAGZ= medias(MagEjeZ,Longitudes);
MEANACEX= medias(ACEjeX,Longitudes);
MEANACEY= medias(ACEjeY,Longitudes);
MEANACEZ= medias(ACEjeZ,Longitudes);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Features de media en barra

BarraMAGX = [];
BarraMAGY = [];
BarraMAGZ = [];
BarraACX = [];
BarraACY = [];
BarraACZ = [];

numeroMuestras = Genomas(32);
nmax = MaxTam/numeroMuestras;
nmax=floor(nmax);

 for u=1:size(MagEjeX,1)                    %Para cada vehículo

    n = Longitudes(u)/numeroMuestras;
    n=floor(n);
    if n==0
        n=1;
    end

    %%%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(19))
        if n<=1
            auxMAGX = [];
            aux = MagEjeX(u,1:Longitudes(u));
            auxMAGX = [auxMAGX, mean(aux,2)];
        else
            auxMAGX = [];
            aux = MagEjeX(u,1:(numeroMuestras));
            auxMAGX = [auxMAGX, mean(aux,2)];
            for i=1:n-2
                aux =
MagEjeX(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxMAGX = [auxMAGX, mean(aux,2)];
            end
            aux = MagEjeX(u,(numeroMuestras*(n-1)):Longitudes(u));
            auxMAGX = [auxMAGX, mean(aux,2)];
        end
         if n<nmax
             BarraMAGX = [BarraMAGX; auxMAGX zeros(1,nmax-n)];
         else
             BarraMAGX = [BarraMAGX; auxMAGX];
         end
    end

    %%%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(20))
```

```
        if n<=1
             auxMAGY = [];
             aux = MagEjeY(u,1:Longitudes(u));
             auxMAGY = [auxMAGY, mean(aux,2)];
        else
             auxMAGY = [];
             aux = MagEjeY(u,1:(numeroMuestras));
             auxMAGY = [auxMAGY, mean(aux,2)];
             for i=1:n-2
                 aux =
MagEjeY(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                 auxMAGY = [auxMAGY, mean(aux,2)];
             end
             aux = MagEjeY(u,(numeroMuestras*(n-1)):Longitudes(u));
             auxMAGY = [auxMAGY, mean(aux,2)];
        end
          if n<nmax
              BarraMAGY = [BarraMAGY; auxMAGY zeros(1,nmax-n)];
          else
              BarraMAGY = [BarraMAGY; auxMAGY];
          end
    end
    %%%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(21))
        if n<=1
             auxMAGZ = [];
             aux = MagEjeZ(u,1:Longitudes(u));
             auxMAGZ = [auxMAGZ, mean(aux,2)];
        else
             auxMAGZ = [];
             aux = MagEjeZ(u,1:(numeroMuestras));
             auxMAGZ = [auxMAGZ, mean(aux,2)];
             for i=1:n-2
                 aux =
MagEjeZ(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                 auxMAGZ = [auxMAGZ, mean(aux,2)];
             end
             aux = MagEjeZ(u,(numeroMuestras*(n-1)):Longitudes(u));
             auxMAGZ = [auxMAGZ, mean(aux,2)];
        end
          if n<nmax
              BarraMAGZ = [BarraMAGZ; auxMAGZ zeros(1,nmax-n)];
          else
              BarraMAGZ = [BarraMAGZ; auxMAGZ];
          end
    end
    %%%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(22))
        if n<=1
             auxACX = [];
             aux = ACEjeX(u,1:Longitudes(u));
             auxACX = [auxACX, mean(aux,2)];
        else
             auxACX = [];
             aux = ACEjeX(u,1:(numeroMuestras));
```

```
            auxACX = [auxACX, mean(aux,2)];
            for i=1:n-2
                aux =
ACEjeX(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxACX = [auxACX, mean(aux,2)];
            end
            aux = ACEjeX(u,(numeroMuestras*(n-1)):Longitudes(u));
            auxACX = [auxACX, mean(aux,2)];
        end
          if n<nmax
              BarraACX = [BarraACX; auxACX zeros(1,nmax-n)];
          else
              BarraACX = [BarraACX; auxACX];
          end
    end
    %%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(23))
       if n<=1
            auxACY = [];
            aux = ACEjeY(u,1:Longitudes(u));
            auxACY = [auxACY, mean(aux,2)];
       else
            auxACY = [];
            aux = ACEjeY(u,1:(numeroMuestras));
            auxACY = [auxACY, mean(aux,2)];
            for i=1:n-2
                aux =
ACEjeY(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxACY = [auxACY, mean(aux,2)];
            end
            aux = ACEjeY(u,(numeroMuestras*(n-1)):Longitudes(u));
            auxACY = [auxACY, mean(aux,2)];
       end
          if n<nmax
              BarraACY = [BarraACY; auxACY zeros(1,nmax-n)];
          else
              BarraACY = [BarraACY; auxACY];
          end
    end
    %%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(24))
       if n<=1
            auxACZ = [];
            aux = ACEjeZ(u,1:Longitudes(u));
            auxACZ = [auxACZ, mean(aux,2)];
       else
            auxACZ = [];
            aux = ACEjeZ(u,1:(numeroMuestras));
            auxACZ = [auxACZ, mean(aux,2)];
            for i=1:n-2
                aux =
ACEjeZ(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxACZ = [auxACZ, mean(aux,2)];
            end
            aux = ACEjeZ(u,(numeroMuestras*(n-1)):Longitudes(u));
```

```
            auxACZ = [auxACZ, mean(aux,2)];
        end
          if n<nmax
             BarraACZ = [BarraACZ; auxACZ zeros(1,nmax-n)];
          else
             BarraACZ = [BarraACZ; auxACZ];
          end
    end
    end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%
% Features de picos

picosMAGX = [];
picosMAGY = [];
picosMAGZ = [];
picosACX = [];
picosACY = [];
picosACZ = [];
auxiliar = 0;
numpicos = Genomas(33);

for j=1:size(MagEjeX,1)

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(25))
        auxMAGX = 0;
        for i=numpicos+1:size(MagEjeX,2)-(numpicos+1)
            auxiliar = 1;
            for d=1:numpicos
                if(MagEjeX(j,i)<=MagEjeX(j,i+d))
                    auxiliar = 0;
                end
                if(MagEjeX(j,i)<=MagEjeX(j,i-d))
                    auxiliar = 0;
                end
            end
            auxMAGX =  auxMAGX+auxiliar;
        end
        picosMAGX = [picosMAGX; auxMAGX];
    end
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(26))
        auxMAGY = 0;
        for i=numpicos+1:size(MagEjeY,2)-(numpicos+1)
            auxiliar = 1;
            for d=1:numpicos
                if(MagEjeY(j,i)<=MagEjeY(j,i+d))
                    auxiliar = 0;
                end
                if(MagEjeY(j,i)<=MagEjeY(j,i-d))
                    auxiliar = 0;
                end
```

```
        end
        auxMAGY =  auxMAGY+auxiliar;
    end
    picosMAGY = [picosMAGY; auxMAGY];
end
%%%%%%%%%%%%%%%%%%%%%%%%
if (Genomas(27))
    auxMAGZ = 0;
    for i=numpicos+1:size(MagEjeZ,2)-(numpicos+1)
        auxiliar = 1;
        for d=1:numpicos
            if(MagEjeZ(j,i)<=MagEjeZ(j,i+d))
                auxiliar = 0;
            end
            if(MagEjeZ(j,i)<=MagEjeZ(j,i-d))
                auxiliar = 0;
            end
        end
        auxMAGZ =  auxMAGZ+auxiliar;
    end
    picosMAGZ = [picosMAGZ; auxMAGZ];
end

%%%%%%%%%%%%%%%%%%%%%%%%
if (Genomas(28))
    auxACX = 0;
    for i=numpicos+1:size(ACEjeX,2)-(numpicos+1)
        auxiliar = 1;
        for d=1:numpicos
            if(ACEjeX(j,i)<=ACEjeX(j,i+d))
                auxiliar = 0;
            end
            if(ACEjeX(j,i)<=ACEjeX(j,i-d))
                auxiliar = 0;
            end
        end
        auxACX =  auxACX+auxiliar;
    end
    picosACX = [picosACX; auxACX];
end

%%%%%%%%%%%%%%%%%%%%%%%%
if (Genomas(29))
    auxACY = 0;
    for i=numpicos+1:size(ACEjeY,2)-(numpicos+1)
        auxiliar = 1;
        for d=1:numpicos
            if(ACEjeY(j,i)<=ACEjeY(j,i+d))
                auxiliar = 0;
            end
            if(ACEjeY(j,i)<=ACEjeY(j,i-d))
                auxiliar = 0;
            end
        end
```

```
            auxACY =  auxACY+auxiliar;
        end
        picosACY = [picosACY; auxACY];
    end
    %%%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(30))
        auxACZ = 0;
        for i=numpicos+1:size(ACEjeZ,2)-(numpicos+1)
            auxiliar = 1;
            for d=1:numpicos
                if(ACEjeZ(j,i)<=ACEjeZ(j,i+d))
                    auxiliar = 0;
                end
                if(ACEjeZ(j,i)<=ACEjeZ(j,i-d))
                    auxiliar = 0;
                end
            end
            auxACZ =  auxACZ+auxiliar;
        end
        picosACZ = [picosACZ; auxACZ];
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Eleccion de las Features a utilizar
fprintf('Eleccion de las Features a utilizar\n');

data1 = [];
names_variables = {};
numindices = 0;
nameMAGX = {};
nameMAGY = {};
nameMAGZ = {};

nameACX = {};
nameACY = {};
nameACZ = {};

for i=1:nmax
    nameMAGX{i} = strcat('BARRA MAGNETICO EJE X ',num2str(i));
    nameMAGY{i} = strcat('BARRA MAGNETICO EJE Y ',num2str(i));
    nameMAGZ{i} = strcat('BARRA MAGNETICO EJE Z ',num2str(i));

    nameACX{i} = strcat('BARRA ACELERACION EJE X ',num2str(i));
    nameACY{i} = strcat('BARRA ACELERACION EJE Y ',num2str(i));
    nameACZ{i} = strcat('BARRA ACELERACION EJE Z ',num2str(i));
end

if (Genomas(1))
    data1 = [data1; MAXMAGX'];
    names_variables = [names_variables, 'MAX MAGNETICO EJE X'];
    numindices = numindices+1;
end
```

```
if (Genomas(2))
    data1 = [data1; MAXMAGY'];
    names_variables = [names_variables, 'MAX MAGNETICO EJE Y'];
    numindices = numindices+1;
end
if (Genomas(3))
    data1 = [data1; MAXMAGZ'];
    names_variables = [names_variables, 'MAX MAGNETICO EJE Z'];
    numindices = numindices+1;
end
 if (Genomas(4))
    data1 = [data1; MAXACEX'];
    names_variables = [names_variables, 'MAX ACELERACION EJE X'];
    numindices = numindices+1;
end
if (Genomas(5))
    data1 = [data1; MAXACEY'];
    names_variables = [names_variables, 'MAX ACELERACION EJE Y'];
    numindices = numindices+1;
end
if (Genomas(6))
    data1 = [data1; MAXACEZ'];
    names_variables = [names_variables, 'MAX ACELERACION EJE Z'];
    numindices = numindices+1;
end



if (Genomas(7))
    data1 = [data1; MINMAGX'];
    names_variables = [names_variables, 'MIN MAGNETICO EJE X'];
    numindices = numindices+1;
end
if (Genomas(8))
    data1 = [data1; MINMAGY'];
    names_variables = [names_variables, 'MIN MAGNETICO EJE Y'];
    numindices = numindices+1;
end
if (Genomas(9))
    data1 = [data1; MINMAGZ'];
    names_variables = [names_variables, 'MIN MAGNETICO EJE Z'];
    numindices = numindices+1;
end
 if (Genomas(10))
    data1 = [data1; MINACEX'];
    names_variables = [names_variables, 'MIN ACELERACION EJE X'];
    numindices = numindices+1;
end
if (Genomas(11))
    data1 = [data1; MINACEY'];
    names_variables = [names_variables, 'MIN ACELERACION EJE Y'];
    numindices = numindices+1;
end
if (Genomas(12))
```

```
    data1 = [data1; MINACEZ'];
    names_variables = [names_variables, 'MIN ACELERACION EJE Z'];
    numindices = numindices+1;
end



if (Genomas(13))
    data1 = [data1; MEANMAGX'];
    names_variables = [names_variables, 'MEDIA MAGNETICO EJE X'];
    numindices = numindices+1;
end
if (Genomas(14))
    data1 = [data1; MEANMAGY'];
    names_variables = [names_variables, 'MEDIA MAGNETICO EJE Y'];
    numindices = numindices+1;
end
if (Genomas(15))
    data1 = [data1; MEANMAGZ'];
    names_variables = [names_variables, 'MEDIA MAGNETICO EJE Z'];
    numindices = numindices+1;
end
 if (Genomas(16))
    data1 = [data1; MEANACEX'];
    names_variables = [names_variables, 'MEDIA ACELERACION EJE X'];
    numindices = numindices+1;
end
if (Genomas(17))
    data1 = [data1; MEANACEY'];
    names_variables = [names_variables, 'MEDIA ACELERACION EJE Y'];
    numindices = numindices+1;
end
if (Genomas(18))
    data1 = [data1; MEANACEZ'];
    names_variables = [names_variables, 'MEDIA ACELERACION EJE Z'];
    numindices = numindices+1;
end



if (Genomas(19))
    data1 = [data1; BarraMAGX'];
    names_variables = [names_variables, nameMAGX];
    numindices = numindices + size(BarraMAGX,2);
end
if (Genomas(20))
    data1 = [data1; BarraMAGY'];
    names_variables = [names_variables, nameMAGY];
    numindices = numindices + size(BarraMAGY,2);
end
if (Genomas(21))
    data1 = [data1; BarraMAGZ'];
    names_variables = [names_variables, nameMAGZ];
```

```
    numindices = numindices + size(BarraMAGZ,2);
end
 if (Genomas(22))
    data1 = [data1; BarraACX'];
    names_variables = [names_variables, nameACX];
    numindices = numindices + size(BarraACX,2);
end
if (Genomas(23))
    data1 = [data1; BarraACY'];
    names_variables = [names_variables, nameACY];
    numindices = numindices + size(BarraACY,2);
end
if (Genomas(24))
    data1 = [data1; BarraACZ'];
    names_variables = [names_variables, nameACZ];
    numindices = numindices + size(BarraACZ,2);
end



if (Genomas(25))
    data1 = [data1; picosMAGX'];
    names_variables = [names_variables, 'PICOS MAGNETICO EJE X'];
    numindices = numindices+1;
end
if (Genomas(26))
    data1 = [data1; picosMAGY'];
    names_variables = [names_variables, 'PICOS MAGNETICO EJE Y'];
    numindices = numindices+1;
end
if (Genomas(27))
    data1 = [data1; picosMAGZ'];
    names_variables = [names_variables, 'PICOS MAGNETICO EJE Z'];
    numindices = numindices+1;
end
 if (Genomas(28))
    data1 = [data1; picosACX'];
    names_variables = [names_variables, 'PICOS ACELERACION EJE X'];
    numindices = numindices+1;
end
if (Genomas(29))
    data1 = [data1; picosACY'];
    names_variables = [names_variables, 'PICOS ACELERACION EJE Y'];
    numindices = numindices+1;
end
if (Genomas(30))
    data1 = [data1; picosACZ'];
    names_variables = [names_variables, 'PICOS ACELERACION EJE Z'];
    numindices = numindices+1;
end
if (Genomas(31))
    data1 = [data1; Longitudes'];
    names_variables = [names_variables, 'LONGITUD MUESTRA'];
    numindices = numindices+1;
```

```
end
indices = 1:1:numindices;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Creacion de las clases
fprintf('Creacion de las clases\n');
clase = [];
data = [];
GG = 0;
M = 0;
P = 0;

for i=1:size(Detecciones,1)

    GM = 0;
    if (Detecciones(i).deteccion.Longitud  < 3600)
        clase = [ 'P'; clase];
        data = [data1(:,i) data];
        P = P+1;
    else
        if (Detecciones(i).deteccion.Longitud  < 4500)
            clase = [clase; 'M'];
            data = [data data1(:,i)];
            M = M+1;
        else
            clase = [clase; 'G'];
            data = [data data1(:,i)];
            GG = GG+1;
        end
    end
end
data = double(data);
numclases = 2;
numpatclases = [ M GG];
name_clases = {  'MEDIANO', 'GRANDE' };

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Creacion del MPL
fprintf('Creacion del MPL\n');

numinp = numindices;                  % numinp = numero de variables
No enmascaradas
npat = size( clase,1 );
nclases = numclases;
outclase = [];
for i=1:size(clase,1)
    if (clase(i) == 'M')
        auxi = [1;0];
    else
        auxi = [0;1];
    end
    outclase = [outclase auxi];
end
```

```
% Calculamos las prob a priori de las clases
prob_priori = numpatclases / npat;

nsalidas = 1:1:nclases;

% Numero de salidas de la red
out = outclase( nsalidas, : );                % seleccionamos las
salidas que deseamos que tenga la red
[ numout dummy ] = size( out );             % Numero de clases de
salida
ind_out = [ 1:1:numout ];                   % vector de indices de
salidas
clase = ind_out * out;                      % matriz de clasificacion
de patrones no-dispersa
prob_priori = prob_priori(numclases);       % seleccionamos las prob a
priori de las clases seleccionadas



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% INTRODUCIENDO DATOS DE ENTRENAMIENTO

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%
%%%%%                INTRODUCIENDO DATOS DE ENTRENAMIENTO
%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%


n_arch = size(nhunit,2); % numero de arquitecturas


% Metodo de entrenamiento
% funtrain = 'trainlm'; % por defecto L-M




% AQUI SE ENTRENAN Y EVALUAN n_eval REDES
% trainmlp = [];
% Error = {};
Toterr = [];  % matriz con los errores medios: n_pool x n_arch
min_error = [];  % vector con los errores medios minimos en cada
arquitectura
max_error = [];  % vector con los errores medios maximos en cada
arquitectura
mlpnet = [];  % matriz con las estructuras de redes: n_pool x n_arch
```

```
indarqmin = nhunit( 1 ); % arquitectura de la red con minimo error,
inicialmente la primera
indpoolmin = 1;
minerrortotal = npat * numout ; % error inicial igual al numero de
muestras x salidas
% netmin = [];  % estructura de red de error minimo
conf_mat_arq = []; % matriz 3D para guardar las matrices de confusion
promedio de cada arquitectura
% warning off MATLAB:nearlySingularMatrix;

for i = 1:n_arch

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%
    %%%%%      CREANDO Y SIMULANDO CADA RED  num_arquitecturas x
num_pool           %%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%
    % Creamos la red MLP nm-esima con una capa oculta con nhunit(i)
bipolares
    % y una capa de salida con 1 unidad unipolar
    datam = [ min(data') ; max(data') ]';
    Ymed = zeros( numout , npat ); % matriz de salidas promedio del
pool de redes
    for j = 1:n_pool

         funcion = {};
         for w=1:nCapas(i)
             funcion = [funcion 'tansig'];
         end
         funcion = [funcion 'logsig'];

         if ( nCapas( i ) ~= 0) % distinguimos red de dos capas de la
red de una capa
            net = newff( datam , [ nhunit(1:nCapas(i),i)'  numout ],
funcion, funtrain ); % dos capas
        else
            net = newff( datam , [ numout ] , { 'logsig' }, funtrain
); % una capa
        end
        net.trainParam.epochs = numciclos;
        % Número de iteraciones de entrenamiento para cada
presentación del error
        net.trainParam.show = 5;
        % Error cuadrado máximo promedio
        net.trainParam.goal = 1e-6;
        % Entrenamos a la Red
        fprintf('\nENTRENANDO LA RED %d EN LA ARQUITECTURA CON %d
NEURONAS OCULTAS:\n' , j , nhunit( i ) );
        net = train( net , data , out );
        % Obtenemos la respuesta para los patrones
        Y = sim( net , data );
        Ymed = Ymed + Y; % acumulamos las salidas
```

```
        % Calculamos el vector de errores en cada patron frente a la
respuesta deseada
        err = sum( ( ( Y - out ) .^ 2 ) , 1 ); % sumamos el error en
todas las salidas
        % Guardamos los datos de la red en un vector de redes y los
errores de las redes
        if ( j == 1 ) % si j==1 creamos el vector de redes y de
errores
            mlpnet = [ net ];
            errnet = [ err ];
        else % en caso contrario, concatenamos las redes y errores
            mlpnet = cat( 1 , mlpnet , net );
            errnet = cat( 1 , errnet , err );  % guardamos los
vectores de error
        end
        errnetmedio = mean( err );         % error medio en todos los
patrones
        errnervar = var( err );           % varianza del error en
todos los patrones
        Toterr( j , i ) = errnetmedio;    % guardamos el error medio
        Totvar( j , i ) = errnervar;      % guardamos la varianza
        if ( errnetmedio < minerrortotal )
            indarqmin = nhunit( i );      % indice de arquitectura de
red con error minimo
            indpoolmin = j;               % indice de la red del pool
con minimo error
            minerrortotal = errnetmedio ; % nuevo error minimo
            netmin = net;                 % estructura de red de error
minimo
        end
    end  % fin de bucle de j para repeticiones de la misma
arquitectura de red

    %% Buscamos una matriz 'clasificacion' con 1 y 0, donde el 1
indica
    %% las salidas de red maximas
    Ymed = Ymed / n_pool; % promediamos las salidas de la red en el
pool de redes

    % Obtenemos una matriz 'clasificacion' de 1 y 0 para las salidas
seleccionadas como
    % maximas que esten por encima del umbral, si no seran todas cero
    clasificacion = clasificacion_MLP( Y , prob_priori' , umbral );

    % Calculamos la matriz de confusion. Si una columna de
clasificacion es todo ceros, aparece una clase de rechazo
    [ conf_mat , Kappa ] = confussion_matrix( ind_out * clasificacion
, clase );

    % Guardamos el vector de redes en una matriz de redes y las mat.
conf. promedios
    if ( i==1 )
        mlpnet2 = [ mlpnet ];
        conf_mat_arq = [ conf_mat ];
```

```
    else
        mlpnet2 = cat( 2 , mlpnet2 , mlpnet );
        conf_mat_arq = cat( 3 , conf_mat_arq , conf_mat );
    end
    Kappa_arq( i ) = Kappa;  % guardamos el Kappa medio obtenido de
las redes del pool en esta arquitectura
    min_error( i ) = min( Toterr( : , i) ); % guardamos el minimo
error de las redes del pool en esta arquitectura
    max_error( i ) = max( Toterr( : , i) ); % guardamos el maximo
error de las redes del pool en esta arquitectura
    Toterr2(i) = sum( Toterr( : , i) ) / n_pool; % promediamos los
errores de todas las redes del pool
    Totvar2(i) = sum( Totvar( : , i) ) / n_pool; % promediamos las
varianzas de todas las redes del pool

    %%%% Almacenamos los indices Kappa y las posiciones de cada indice
    Kappas = [Kappas; Kappa];
    Indices = [Indices i];

end % fin del bucle i para nhunit

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %% Ordenar los Kappas
    fprintf('Ordenar los Kappas\n');
    KappasOrdenadas = [KappasOrdenadas;Kappas(1)];
    IndicesOrdenados = [IndicesOrdenados;Indices(1)];

    for i=2:size(Kappas,1)
        i
        bucle=1;
        q = 1;
        while (bucle==1)
             if (Kappas(i) >= KappasOrdenadas(q))
                 KappasOrdenadas = [KappasOrdenadas(1:(q-1));
Kappas(i); KappasOrdenadas(q:size(KappasOrdenadas,1))];
                 IndicesOrdenados = [IndicesOrdenados(1:(q-1));
Indices(i); IndicesOrdenados(q:size(IndicesOrdenados,1))];
                 bucle = 0;
             else
                 q = q+1;
             end
             if (q>size(KappasOrdenadas,1))
                 KappasOrdenadas = [KappasOrdenadas;Kappas(i)];
                 IndicesOrdenados = [IndicesOrdenados;Indices(i)];
                 bucle = 0;
             end
        end
    end
 fprintf(' FINAL DE EJECUCCION\n');

 fprintf(' Indice: '); IndicesOrdenados(1)
 fprintf(' capa1: '); nhunit(1,IndicesOrdenados(1))
 fprintf(' capa2: '); nhunit(2,IndicesOrdenados(1))
 fprintf(' Kappa: '); KappasOrdenadas(1)
```

2. ALGORITMO PARA DIRECCIÓN

```
clear all;
close all;
fprintf('MLP \n');
fprintf('Analisis de diferentes arquitecturas de MLP \n');
fprintf('Para muestras capturadas automaticamente\n');
fprintf('Para el DIRECCIÓN\n');



%% CONFIGURACION
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
nombreArchivo = 'C:\Users\Ruben\Documents\universidad\Master\Trabajo
Final de Master\Base de datos\Datos-Detectados-
Automaticos\DatosAutomaticos.mat';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
n_pool = 10;  % NÚMERO DE REPETICIONES DE CADA REDv
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Valor del umbral con el que comparar las salidas de la red
umbral = 0.1;  % umbral para seleccionar una salida de red como no
nula
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Número máximo de ciclos de entrenamiento
% (disminuirlo para los metodos cuasi-Newton y de grad.conjugados)
numciclos = 100;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Genomas = [0 0 1 1 1 0 1 0 1 0 0 1 0 1 0 0 0 0 1 1 0 0 0 0 1 0 0 1 1 1
1 23 362];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
NeuroMAX = 25;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
funtrain = 'trainlm'; % por defecto L-M


%% CARGA DE DATOS
load(nombreArchivo');

%% CARGA DE COMBINACIONES DE ARQUITECTURAS

% Unidades Ocultas en las diferentes arquitecturas
    AA = 1;
    AB = 0;
    nhunit = [ AA  AB ];
    nCapas = [];
    nCapas = [nCapas 1];
while AB<=NeuroMAX

    AA = AA+1;
    if AA > NeuroMAX
        AA = 1;
        AB = AB+1;
    end
```

```
    nhunit = [nhunit; AA  AB ];
    if AB == 0
        nCapas = [nCapas 1];
    else
        nCapas = [nCapas 2];
    end
end
% nhunit = [ 20 19 18 17 16; 0 0 0 12 10];
% nCapas = [ 1 1 1 2 2];
nhunit = nhunit';


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% VARIABLES

KappasOrdenadas = [];
IndicesOrdenados = [];
Kappas = [];
Indices = [];
MaxTam = max(Longitudes,[],1)



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Extraccion de las features
fprintf('Extraccion de las features\n');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Features estadisticas

MAXMAGX= max(MagEjeX,[],2);
MAXMAGY= max(MagEjeY,[],2);
MAXMAGZ= max(MagEjeZ,[],2);
MAXACEX= max(ACEjeX,[],2);
MAXACEY= max(ACEjeY,[],2);
MAXACEZ= max(ACEjeZ,[],2);

MINMAGX= minimos(MagEjeX,Longitudes);
MINMAGY= minimos(MagEjeY,Longitudes);
MINMAGZ= minimos(MagEjeZ,Longitudes);
MINACEX= minimos(ACEjeX,Longitudes);
MINACEY= minimos(ACEjeY,Longitudes);
MINACEZ= minimos(ACEjeZ,Longitudes);

MEANMAGX= medias(MagEjeX,Longitudes);
MEANMAGY= medias(MagEjeY,Longitudes);
MEANMAGZ= medias(MagEjeZ,Longitudes);
MEANACEX= medias(ACEjeX,Longitudes);
MEANACEY= medias(ACEjeY,Longitudes);
MEANACEZ= medias(ACEjeZ,Longitudes);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Features de media en barra

BarraMAGX = [];
BarraMAGY = [];
BarraMAGZ = [];
BarraACX = [];
BarraACY = [];
BarraACZ = [];

numeroMuestras = Genomas(32);
nmax = MaxTam/numeroMuestras;
nmax=floor(nmax);

 for u=1:size(MagEjeX,1)                        %Para cada vehículo

    n = Longitudes(u)/numeroMuestras;
    n=floor(n);
    if n==0
        n=1;
    end

    %%%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(19))
        if n<=1
            auxMAGX = [];
            aux = MagEjeX(u,1:Longitudes(u));
            auxMAGX = [auxMAGX, mean(aux,2)];
        else
            auxMAGX = [];
            aux = MagEjeX(u,1:(numeroMuestras));
            auxMAGX = [auxMAGX, mean(aux,2)];
            for i=1:n-2
                aux =
MagEjeX(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxMAGX = [auxMAGX, mean(aux,2)];
            end
            aux = MagEjeX(u,(numeroMuestras*(n-1)):Longitudes(u));
            auxMAGX = [auxMAGX, mean(aux,2)];
        end
         if n<nmax
             BarraMAGX = [BarraMAGX; auxMAGX zeros(1,nmax-n)];
         else
             BarraMAGX = [BarraMAGX; auxMAGX];
         end
    end

    %%%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(20))
       if n<=1
            auxMAGY = [];
            aux = MagEjeY(u,1:Longitudes(u));
            auxMAGY = [auxMAGY, mean(aux,2)];
       else
```

```
            auxMAGY = [];
            aux = MagEjeY(u,1:(numeroMuestras));
            auxMAGY = [auxMAGY, mean(aux,2)];
            for i=1:n-2
                aux =
MagEjeY(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxMAGY = [auxMAGY, mean(aux,2)];
            end
            aux = MagEjeY(u,(numeroMuestras*(n-1)):Longitudes(u));
            auxMAGY = [auxMAGY, mean(aux,2)];
        end
          if n<nmax
              BarraMAGY = [BarraMAGY; auxMAGY zeros(1,nmax-n)];
          else
              BarraMAGY = [BarraMAGY; auxMAGY];
          end
    end
    %%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(21))
        if n<=1
            auxMAGZ = [];
            aux = MagEjeZ(u,1:Longitudes(u));
            auxMAGZ = [auxMAGZ, mean(aux,2)];
        else
            auxMAGZ = [];
            aux = MagEjeZ(u,1:(numeroMuestras));
            auxMAGZ = [auxMAGZ, mean(aux,2)];
            for i=1:n-2
                aux =
MagEjeZ(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxMAGZ = [auxMAGZ, mean(aux,2)];
            end
            aux = MagEjeZ(u,(numeroMuestras*(n-1)):Longitudes(u));
            auxMAGZ = [auxMAGZ, mean(aux,2)];
        end
          if n<nmax
              BarraMAGZ = [BarraMAGZ; auxMAGZ zeros(1,nmax-n)];
          else
              BarraMAGZ = [BarraMAGZ; auxMAGZ];
          end
    end
    %%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(22))
        if n<=1
            auxACX = [];
            aux = ACEjeX(u,1:Longitudes(u));
            auxACX = [auxACX, mean(aux,2)];
        else
            auxACX = [];
            aux = ACEjeX(u,1:(numeroMuestras));
            auxACX = [auxACX, mean(aux,2)];
            for i=1:n-2
                aux =
ACEjeX(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxACX = [auxACX, mean(aux,2)];
```

```
            end
            aux = ACEjeX(u,(numeroMuestras*(n-1)):Longitudes(u));
            auxACX = [auxACX, mean(aux,2)];
        end
          if n<nmax
              BarraACX = [BarraACX; auxACX zeros(1,nmax-n)];
          else
              BarraACX = [BarraACX; auxACX];
          end
    end
    %%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(23))
       if n<=1
            auxACY = [];
            aux = ACEjeY(u,1:Longitudes(u));
            auxACY = [auxACY, mean(aux,2)];
       else
            auxACY = [];
            aux = ACEjeY(u,1:(numeroMuestras));
            auxACY = [auxACY, mean(aux,2)];
            for i=1:n-2
                aux =
ACEjeY(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxACY = [auxACY, mean(aux,2)];
            end
            aux = ACEjeY(u,(numeroMuestras*(n-1)):Longitudes(u));
            auxACY = [auxACY, mean(aux,2)];
       end
          if n<nmax
              BarraACY = [BarraACY; auxACY zeros(1,nmax-n)];
          else
              BarraACY = [BarraACY; auxACY];
          end
    end
    %%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(24))
       if n<=1
            auxACZ = [];
            aux = ACEjeZ(u,1:Longitudes(u));
            auxACZ = [auxACZ, mean(aux,2)];
       else
            auxACZ = [];
            aux = ACEjeZ(u,1:(numeroMuestras));
            auxACZ = [auxACZ, mean(aux,2)];
            for i=1:n-2
                aux =
ACEjeZ(u,(numeroMuestras*i):(numeroMuestras*(i+1)));
                auxACZ = [auxACZ, mean(aux,2)];
            end
            aux = ACEjeZ(u,(numeroMuestras*(n-1)):Longitudes(u));
            auxACZ = [auxACZ, mean(aux,2)];
       end
          if n<nmax
              BarraACZ = [BarraACZ; auxACZ zeros(1,nmax-n)];
          else
```

```
                BarraACZ = [BarraACZ; auxACZ];
            end
        end
    end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%
% Features de picos

picosMAGX = [];
picosMAGY = [];
picosMAGZ = [];
picosACX = [];
picosACY = [];
picosACZ = [];
auxiliar = 0;
numpicos = Genomas(33);

for j=1:size(MagEjeX,1)

    %%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(25))
        auxMAGX = 0;
        for i=numpicos+1:size(MagEjeX,2)-(numpicos+1)
            auxiliar = 1;
            for d=1:numpicos
                if(MagEjeX(j,i)<=MagEjeX(j,i+d))
                    auxiliar = 0;
                end
                if(MagEjeX(j,i)<=MagEjeX(j,i-d))
                    auxiliar = 0;
                end
            end
            auxMAGX =  auxMAGX+auxiliar;
        end
        picosMAGX = [picosMAGX; auxMAGX];
    end
    %%%%%%%%%%%%%%%%%%%%%%%
    if (Genomas(26))
        auxMAGY = 0;
        for i=numpicos+1:size(MagEjeY,2)-(numpicos+1)
            auxiliar = 1;
            for d=1:numpicos
                if(MagEjeY(j,i)<=MagEjeY(j,i+d))
                    auxiliar = 0;
                end
                if(MagEjeY(j,i)<=MagEjeY(j,i-d))
                    auxiliar = 0;
                end
            end
            auxMAGY =  auxMAGY+auxiliar;
        end
        picosMAGY = [picosMAGY; auxMAGY];
    end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%
if (Genomas(27))
    auxMAGZ = 0;
    for i=numpicos+1:size(MagEjeZ,2)-(numpicos+1)
        auxiliar = 1;
        for d=1:numpicos
            if(MagEjeZ(j,i)<=MagEjeZ(j,i+d))
                auxiliar = 0;
            end
            if(MagEjeZ(j,i)<=MagEjeZ(j,i-d))
                auxiliar = 0;
            end
        end
        auxMAGZ =  auxMAGZ+auxiliar;
    end
    picosMAGZ = [picosMAGZ; auxMAGZ];
end

%%%%%%%%%%%%%%%%%%%%%%%%
if (Genomas(28))
    auxACX = 0;
    for i=numpicos+1:size(ACEjeX,2)-(numpicos+1)
        auxiliar = 1;
        for d=1:numpicos
            if(ACEjeX(j,i)<=ACEjeX(j,i+d))
                auxiliar = 0;
            end
            if(ACEjeX(j,i)<=ACEjeX(j,i-d))
                auxiliar = 0;
            end
        end
        auxACX =  auxACX+auxiliar;
    end
    picosACX = [picosACX; auxACX];
end

%%%%%%%%%%%%%%%%%%%%%%%%
if (Genomas(29))
    auxACY = 0;
    for i=numpicos+1:size(ACEjeY,2)-(numpicos+1)
        auxiliar = 1;
        for d=1:numpicos
            if(ACEjeY(j,i)<=ACEjeY(j,i+d))
                auxiliar = 0;
            end
            if(ACEjeY(j,i)<=ACEjeY(j,i-d))
                auxiliar = 0;
            end
        end
        auxACY =  auxACY+auxiliar;
    end
    picosACY = [picosACY; auxACY];
end
%%%%%%%%%%%%%%%%%%%%%%%%
```

```
    if (Genomas(30))
        auxACZ = 0;
        for i=numpicos+1:size(ACEjeZ,2)-(numpicos+1)
            auxiliar = 1;
            for d=1:numpicos
                if(ACEjeZ(j,i)<=ACEjeZ(j,i+d))
                    auxiliar = 0;
                end
                if(ACEjeZ(j,i)<=ACEjeZ(j,i-d))
                    auxiliar = 0;
                end
            end
            auxACZ =  auxACZ+auxiliar;
        end
        picosACZ = [picosACZ; auxACZ];
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Eleccion de las Features a utilizar
fprintf('Eleccion de las Features a utilizar\n');

data1 = [];
names_variables = {};
numindices = 0;
nameMAGX = {};
nameMAGY = {};
nameMAGZ = {};

nameACX = {};
nameACY = {};
nameACZ = {};

for i=1:nmax
    nameMAGX{i} = strcat('BARRA MAGNETICO EJE X ',num2str(i));
    nameMAGY{i} = strcat('BARRA MAGNETICO EJE Y ',num2str(i));
    nameMAGZ{i} = strcat('BARRA MAGNETICO EJE Z ',num2str(i));

    nameACX{i} = strcat('BARRA ACELERACION EJE X ',num2str(i));
    nameACY{i} = strcat('BARRA ACELERACION EJE Y ',num2str(i));
    nameACZ{i} = strcat('BARRA ACELERACION EJE Z ',num2str(i));
end

if (Genomas(1))
    data1 = [data1; MAXMAGX'];
    names_variables = [names_variables, 'MAX MAGNETICO EJE X'];
    numindices = numindices+1;
end
if (Genomas(2))
    data1 = [data1; MAXMAGY'];
    names_variables = [names_variables, 'MAX MAGNETICO EJE Y'];
    numindices = numindices+1;
end
```

```
if (Genomas(3))
    data1 = [data1; MAXMAGZ'];
    names_variables = [names_variables, 'MAX MAGNETICO EJE Z'];
    numindices = numindices+1;
end
 if (Genomas(4))
    data1 = [data1; MAXACEX'];
    names_variables = [names_variables, 'MAX ACELERACION EJE X'];
    numindices = numindices+1;
end
if (Genomas(5))
    data1 = [data1; MAXACEY'];
    names_variables = [names_variables, 'MAX ACELERACION EJE Y'];
    numindices = numindices+1;
end
if (Genomas(6))
    data1 = [data1; MAXACEZ'];
    names_variables = [names_variables, 'MAX ACELERACION EJE Z'];
    numindices = numindices+1;
end



if (Genomas(7))
    data1 = [data1; MINMAGX'];
    names_variables = [names_variables, 'MIN MAGNETICO EJE X'];
    numindices = numindices+1;
end
if (Genomas(8))
    data1 = [data1; MINMAGY'];
    names_variables = [names_variables, 'MIN MAGNETICO EJE Y'];
    numindices = numindices+1;
end
if (Genomas(9))
    data1 = [data1; MINMAGZ'];
    names_variables = [names_variables, 'MIN MAGNETICO EJE Z'];
    numindices = numindices+1;
end
 if (Genomas(10))
    data1 = [data1; MINACEX'];
    names_variables = [names_variables, 'MIN ACELERACION EJE X'];
    numindices = numindices+1;
end
if (Genomas(11))
    data1 = [data1; MINACEY'];
    names_variables = [names_variables, 'MIN ACELERACION EJE Y'];
    numindices = numindices+1;
end
if (Genomas(12))
    data1 = [data1; MINACEZ'];
    names_variables = [names_variables, 'MIN ACELERACION EJE Z'];
    numindices = numindices+1;
end
```

```
if (Genomas(13))
    data1 = [data1; MEANMAGX'];
    names_variables = [names_variables, 'MEDIA MAGNETICO EJE X'];
    numindices = numindices+1;
end
if (Genomas(14))
    data1 = [data1; MEANMAGY'];
    names_variables = [names_variables, 'MEDIA MAGNETICO EJE Y'];
    numindices = numindices+1;
end
if (Genomas(15))
    data1 = [data1; MEANMAGZ'];
    names_variables = [names_variables, 'MEDIA MAGNETICO EJE Z'];
    numindices = numindices+1;
end
 if (Genomas(16))
    data1 = [data1; MEANACEX'];
    names_variables = [names_variables, 'MEDIA ACELERACION EJE X'];
    numindices = numindices+1;
end
if (Genomas(17))
    data1 = [data1; MEANACEY'];
    names_variables = [names_variables, 'MEDIA ACELERACION EJE Y'];
    numindices = numindices+1;
end
if (Genomas(18))
    data1 = [data1; MEANACEZ'];
    names_variables = [names_variables, 'MEDIA ACELERACION EJE Z'];
    numindices = numindices+1;
end



if (Genomas(19))
    data1 = [data1; BarraMAGX'];
    names_variables = [names_variables, nameMAGX];
    numindices = numindices + size(BarraMAGX,2);
end
if (Genomas(20))
    data1 = [data1; BarraMAGY'];
    names_variables = [names_variables, nameMAGY];
    numindices = numindices + size(BarraMAGY,2);
end
if (Genomas(21))
    data1 = [data1; BarraMAGZ'];
    names_variables = [names_variables, nameMAGZ];
    numindices = numindices + size(BarraMAGZ,2);
end
 if (Genomas(22))
    data1 = [data1; BarraACX'];
    names_variables = [names_variables, nameACX];
    numindices = numindices + size(BarraACX,2);
```

```
end
if (Genomas(23))
    data1 = [data1; BarraACY'];
    names_variables = [names_variables, nameACY];
    numindices = numindices + size(BarraACY,2);
end
if (Genomas(24))
    data1 = [data1; BarraACZ'];
    names_variables = [names_variables, nameACZ];
    numindices = numindices + size(BarraACZ,2);
end



if (Genomas(25))
    data1 = [data1; picosMAGX'];
    names_variables = [names_variables, 'PICOS MAGNETICO EJE X'];
    numindices = numindices+1;
end
if (Genomas(26))
    data1 = [data1; picosMAGY'];
    names_variables = [names_variables, 'PICOS MAGNETICO EJE Y'];
    numindices = numindices+1;
end
if (Genomas(27))
    data1 = [data1; picosMAGZ'];
    names_variables = [names_variables, 'PICOS MAGNETICO EJE Z'];
    numindices = numindices+1;
end
 if (Genomas(28))
    data1 = [data1; picosACX'];
    names_variables = [names_variables, 'PICOS ACELERACION EJE X'];
    numindices = numindices+1;
end
if (Genomas(29))
    data1 = [data1; picosACY'];
    names_variables = [names_variables, 'PICOS ACELERACION EJE Y'];
    numindices = numindices+1;
end
if (Genomas(30))
    data1 = [data1; picosACZ'];
    names_variables = [names_variables, 'PICOS ACELERACION EJE Z'];
    numindices = numindices+1;
end
if (Genomas(31))
    data1 = [data1; Longitudes'];
    names_variables = [names_variables, 'LONGITUD MUESTRA'];
    numindices = numindices+1;
end
indices = 1:1:numindices;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Creacion de las clases
fprintf('Creacion de las clases\n');
```

```
clase = [];
data = [];
SS = 0;
EE = 0;
fprintf('Creacion del fichero de direcciones\n');
for i=1:size(Detecciones,1)
    if (Detecciones(i).deteccion.Direccion(2) == 'S')
        clase = [ 'S'; clase];
        data = [data1(:,i) data];
        SS = SS+1;
    else
        clase = [clase; 'E'];
        data = [data data1(:,i)];
        EE = EE+1;
    end
end


data = double(data);
numclases = 2;
numpatclases = [ SS EE ];
name_clases = { 'Salida'  ,  'Entrada' };

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Creacion del MPL
fprintf('Creacion del MPL\n');

numinp = numindices;                    % numinp = numero de variables
No enmascaradas
npat = size( clase,1 );
nclases = numclases;
outclase = [];
for i=1:size(clase,1)
    if (clase(i) == 'S')
        auxi = [1;0];
    else
        auxi = [0;1];
    end
    outclase = [outclase auxi];
end


% Calculamos las prob a priori de las clases
prob_priori = numpatclases / npat;

nsalidas = 1:1:nclases;

% Numero de salidas de la red
out = outclase( nsalidas, : );                  % seleccionamos las
salidas que deseamos que tenga la red
[ numout dummy ] = size( out );             % Numero de clases de
salida
```

```
ind_out = [ 1:1:numout ];                    % vector de indices de
salidas
clase = ind_out * out;                       % matriz de clasificacion
de patrones no-dispersa
prob_priori = prob_priori(numclases);        % seleccionamos las prob a
priori de las clases seleccionadas



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% INTRODUCIENDO DATOS DE ENTRENAMIENTO

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%                 INTRODUCIENDO DATOS DE ENTRENAMIENTO
%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


n_arch = size(nhunit,2); % numero de arquitecturas


% Metodo de entrenamiento
% funtrain = 'trainlm'; % por defecto L-M




% AQUI SE ENTRENAN Y EVALUAN n_eval REDES
% trainmlp = [];
% Error = {};
Toterr = [];  % matriz con los errores medios: n_pool x n_arch
min_error = [];  % vector con los errores medios minimos en cada
arquitectura
max_error = [];  % vector con los errores medios maximos en cada
arquitectura
mlpnet = [];  % matriz con las estructuras de redes: n_pool x n_arch
indarqmin = nhunit( 1 ); % arquitectura de la red con minimo error,
inicialmente la primera
indpoolmin = 1;
minerrortotal = npat * numout ; % error inicial igual al numero de
muestras x salidas
% netmin = [];  % estructura de red de error minimo
conf_mat_arq = []; % matriz 3D para guardar las matrices de confusion
promedio de cada arquitectura
% warning off MATLAB:nearlySingularMatrix;

for i = 1:n_arch

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
    %%%%%      CREANDO Y SIMULANDO CADA RED  num_arquitecturas x
num_pool           %%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%
    % Creamos la red MLP nm-esima con una capa oculta con nhunit(i)
bipolares
    % y una capa de salida con 1 unidad unipolar
    datam = [ min(data') ; max(data') ]';
    Ymed = zeros( numout , npat ); % matriz de salidas promedio del
pool de redes
    for j = 1:n_pool

         funcion = {};
         for w=1:nCapas(i)
             funcion = [funcion 'tansig'];
         end
         funcion = [funcion 'logsig'];

         if ( nCapas( i ) ~= 0) % distinguimos red de dos capas de la
red de una capa
            net = newff( datam , [ nhunit(1:nCapas(i),i)'  numout ],
funcion, funtrain ); % dos capas
        else
            net = newff( datam , [ numout ] , { 'logsig' }, funtrain
); % una capa
        end
        net.trainParam.epochs = numciclos;
        % Número de iteraciones de entrenamiento para cada
presentación del error
        net.trainParam.show = 5;
        % Error cuadrado máximo promedio
        net.trainParam.goal = 1e-6;
        % Entrenamos a la Red
        fprintf('\nENTRENANDO LA RED %d EN LA ARQUITECTURA CON %d
NEURONAS OCULTAS:\n' , j , nhunit( i ) );
        net = train( net , data , out );
        % Obtenemos la respuesta para los patrones
        Y = sim( net , data );
        Ymed = Ymed + Y; % acumulamos las salidas
        % Calculamos el vector de errores en cada patron frente a la
respuesta deseada
        err = sum( ( ( Y - out ) .^ 2 ) , 1 ); % sumamos el error en
todas las salidas
        % Guardamos los datos de la red en un vector de redes y los
errores de las redes
        if ( j == 1 ) % si j==1 creamos el vector de redes y de
errores
            mlpnet = [ net ];
            errnet = [ err ];
        else % en caso contrario, concatenamos las redes y errores
            mlpnet = cat( 1 , mlpnet , net );
            errnet = cat( 1 , errnet , err );  % guardamos los
vectores de error
```

```
        end
        errnetmedio = mean( err );          % error medio en todos los
patrones
        errnervar = var( err );            % varianza del error en
todos los patrones
        Toterr( j , i ) = errnetmedio;     % guardamos el error medio
        Totvar( j , i ) = errnervar;       % guardamos la varianza
        if ( errnetmedio < minerrortotal )
            indarqmin = nhunit( i );       % indice de arquitectura de
red con error minimo
            indpoolmin = j;                % indice de la red del pool
con minimo error
            minerrortotal = errnetmedio ; % nuevo error minimo
            netmin = net;                  % estructura de red de error
minimo
        end
    end  % fin de bucle de j para repeticiones de la misma
arquitectura de red

    %% Buscamos una matriz 'clasificacion' con 1 y 0, donde el 1
indica
    %% las salidas de red maximas
    Ymed = Ymed / n_pool; % promediamos las salidas de la red en el
pool de redes

    % Obtenemos una matriz 'clasificacion' de 1 y 0 para las salidas
seleccionadas como
    % maximas que esten por encima del umbral, si no seran todas cero
    clasificacion = clasificacion_MLP( Y , prob_priori' , umbral );

    % Calculamos la matriz de confusion. Si una columna de
clasificacion es todo ceros, aparece una clase de rechazo
    [ conf_mat , Kappa ] = confussion_matrix( ind_out * clasificacion
, clase );

    % Guardamos el vector de redes en una matriz de redes y las mat.
conf. promedios
    if ( i==1 )
        mlpnet2 = [ mlpnet ];
        conf_mat_arq = [ conf_mat ];
    else
        mlpnet2 = cat( 2 , mlpnet2 , mlpnet );
        conf_mat_arq = cat( 3 , conf_mat_arq , conf_mat );
    end
    Kappa_arq( i ) = Kappa;  % guardamos el Kappa medio obtenido de
las redes del pool en esta arquitectura
    min_error( i ) = min( Toterr( : , i) ); % guardamos el minimo
error de las redes del pool en esta arquitectura
    max_error( i ) = max( Toterr( : , i) ); % guardamos el maximo
error de las redes del pool en esta arquitectura
    Toterr2(i) = sum( Toterr( : , i) ) / n_pool; % promediamos los
errores de todas las redes del pool
    Totvar2(i) = sum( Totvar( : , i) ) / n_pool; % promediamos las
varianzas de todas las redes del pool
```

```
    %%%% Almacenamos los indices Kappa y las posiciones de cada indice
    Kappas = [Kappas; Kappa];
    Indices = [Indices i];

end % fin del bucle i para nhunit

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %% Ordenar los Kappas
    fprintf('Ordenar los Kappas\n');
    KappasOrdenadas = [KappasOrdenadas;Kappas(1)];
    IndicesOrdenados = [IndicesOrdenados;Indices(1)];

    for i=2:size(Kappas,1)
        i
        bucle=1;
        q = 1;
        while (bucle==1)
            if (Kappas(i) >= KappasOrdenadas(q))
                KappasOrdenadas = [KappasOrdenadas(1:(q-1));
Kappas(i); KappasOrdenadas(q:size(KappasOrdenadas,1))];
                IndicesOrdenados = [IndicesOrdenados(1:(q-1));
Indices(i); IndicesOrdenados(q:size(IndicesOrdenados,1))];
                bucle = 0;
            else
                q = q+1;
            end
            if (q>size(KappasOrdenadas,1))
                KappasOrdenadas = [KappasOrdenadas;Kappas(i)];
                IndicesOrdenados = [IndicesOrdenados;Indices(i)];
                bucle = 0;
            end
        end
    end
 fprintf(' FINAL DE EJECUCCION\n');

 fprintf(' Indice: '); IndicesOrdenados(1)
 fprintf(' capa1: '); nhunit(1,IndicesOrdenados(1))
 fprintf(' capa2: '); nhunit(2,IndicesOrdenados(1))
 fprintf(' Kappa: '); KappasOrdenadas(1)
```

ANEXO 7. FUNCIONES FEATURES AMPLIADAS

1. DESVIACIÓN ESTÁNDAR

```
function Salida = stdesv(matriz, longitud)

    Salida = [];
    for i=1:1:size(matriz,1)
        vector = matriz(i,1:longitud(i));
        vectorSalida = std(vector,0,2);
        Salida = [Salida; vectorSalida];
    end

end
```

2. DESVIACIÓN MEDIA ABSOLUTA

```
function Salida = MedAbsDev(matriz, longitud)

    Salida = [];
    for i=1:1:size(matriz,1)
        vector = matriz(i,1:longitud(i));
        vectorSalida = mad(vector,0,2);
        Salida = [Salida; vectorSalida];
    end

end
```

3. RANGO INTERCUARTIL

```
function Salida = IntRang(matriz, longitud)

    Salida = [];
    for i=1:1:size(matriz,1)
        vector = matriz(i,1:longitud(i));
        vectorSalida = iqr(vector,2);
        Salida = [Salida; vectorSalida];
    end

end
```

4. ENTROPÍA ESPECTRAL

```
function Salida = SigEnt(matriz, longitud)

    Salida = [];
    for i=1:1:size(matriz,1)
        AUX = [];
        AUX = matriz(i,1:longitud(i));
        AUX = double(AUX);
        Salida= [Salida; entropy(AUX)];
    end
end
```

5. MEDIDA ENERGÉTICA

```
function Salida = EneMea(matriz, longitud)

    Salida = [];
    Suma = 0;
    for i=1:1:size(matriz,1)
        for u=1:1:longitud(i)
            Suma = Suma + (matriz(i,u))^2;
        end
        Suma = sqrt(Suma);
        Salida= [Salida; Suma];
    end
end
```

6. ASIMETRÍA DE LA SEÑAL EN EL DOMINIO DE LA FRECUENCIA

```
function Salida = skew(matriz, longitud)

    Salida = [];
    for i=1:1:size(matriz,1)
        vector = matriz(i,1:longitud(i));
        vectorSalida = skewness(vector,1,2);
        Salida = [Salida; vectorSalida];
    end

end
```

7. CURTOSIS

```
function Salida = Kurt(matriz, longitud)

    Salida = [];
    for i=1:1:size(matriz,1)
        vector = matriz(i,1:longitud(i));
        vectorSalida = kurtosis(vector,1,2);
        Salida = [Salida; vectorSalida];
    end

end
```

8. TRANSFORMADA RÁPIDA DE FOURIER

```
function [ff1, mff1, ff2, mff2, ff3, mff3, ff4, mff4, MP] =
ff4ffm(matriz, longitud, Fs)

    ff1 = [];
    mff1 = [];
    ff2 = [];
    mff2 = [];
    ff3 = [];
    mff3 = [];
    ff4 = [];
    mff4 = [];
    MP = [];

    f1 = 0;
    mf1 = 0;
    f2 = 0;
    mf2 = 0;
    f3 = 0;
    mf3 = 0;
    f4 = 0;
    mf4 = 0;
    SumPesos = 0;
    MediaPonderada = 0;
    Salida_my = [];
    Salida_f = [];

    [Salida_my, Salida_f] = myfft(matriz, longitud, Fs);
    for i=1:1:size(Salida_my,1)
        for u=1:1:size(Salida_my,2)
            if Salida_my(i,u) > mf1
                mf4 = mf3;
                f4 = f3;
                mf3 = mf2;
                f3 = f2;
                mf2 = mf1;
                f2 = f1;
                mf1 = Salida_my(i,u);
                f1 = Salida_f(i,u);
            else
                if Salida_my(i,u) > mf2
                    mf4 = mf3;
                    f4 = f3;
                    mf3 = mf2;
                    f3 = f2;
                    mf2 = Salida_my(i,u);
                    f2 = Salida_f(i,u);
                else
                    if Salida_my(i,u) > mf3
                        mf4 = mf3;
                        f4 = f3;
                        mf3 = Salida_my(i,u);
                        f3 = Salida_f(i,u);
```

```
                    else
                        if Salida_my(i,u) > mf4
                            mf4 = Salida_my(i,u);
                            f4 = Salida_f(i,u);
                        end
                    end
                end
            end
            SumPesos = SumPesos + Salida_my(i,u);
            MediaPonderada = MediaPonderada +
(Salida_f(i,u)*Salida_my(i,u));
        end
        ff1 = [ff1; f1];
        mff1 = [mff1; mf1];
        ff2 = [ff2; f2];
        mff2 = [mff2; mf2];
        ff3 = [ff3; f3];
        mff3 = [mff3; mf3];
        ff4 = [ff4; f4];
        mff4 = [mff4; mf4];
        MP = [MP; MediaPonderada/SumPesos];
        MediaPonderada = 0;
        SumPesos = 0;
    end
end

function [Salida_my, Salida_f] = myfft(matriz, longitud, Fs)
    nfft=1024;%el numero de puntos de la fft
    Salida_my = [];
    Salida_f = [];

    for i=1:1:size(matriz,1)
        Y=fft(matriz(i,:),nfft);% tomar la FFT, y llenando con ceros,
de manera que el largo de la FFT sea nfft
        Y = Y(1:nfft/2); % la FFT es simétrica, así que se tira la
mitad
        my = abs(Y).^2;% tomar la potencia espectral, módulo
alcuadrado de la FFT
        f = (0:nfft/2-1)*Fs/nfft; %construccion del vector de
frecuencias
        Salida_my = [Salida_my; my];
        Salida_f = [Salida_f; f];
    end
end
```