

Grado Universitario en Ingeniería Informática
2022-2023

Trabajo fin de grado. 4ºcurso

Modelado de la actividad de conducción mediante Inteligencia Artificial

Roberto Gutiérrez Hernández

100429007@alumnos.uc3m.es

Tutor

Agapito Ledezma Espino

Campus de Leganés, 2023





RESUMEN

Este documento presenta el proceso de desarrollo del modelado de la actividad de conducción mediante Inteligencia Artificial. La implementación de este modelo se fundamenta en una red de neuronas denominada Long Short-Term Memory (LSTM), que permite la clasificación de las acciones que va a tomar un conductor en un momento determinado. El objetivo del proyecto es representar maniobras de conducción de alto nivel a partir de un conjunto de datos del vehículo y su situación.

Para recoger este conjunto de datos se dispone de un simulador de conducción desarrollado por el grupo de investigación CAOS, mediante el cual se pueden extraer datos sobre varios elementos del vehículo, el entorno que le rodea en la simulación y sobre la zona de visión del conductor. Este conjunto de información la utiliza el modelo para determinar la tarea o acción que está realizando el conductor y que posteriormente permite que el sistema determine la maniobra que se pretende realizar.

Una vez desarrollado el modelo, es necesario evaluar su precisión, ya que se han detectado una pequeña parte de falsos positivos, que dificultan el seguimiento de transiciones entre maniobras. Por ello, aunque la red neuronal ofrece unos resultados prometedores, es necesario seguir trabajando en el modelo para reducir ese número de falsos positivos con el fin de obtener un sistema perfectamente preparado para su implementación en vehículos reales.

Palabras clave: conducción, Inteligencia Artificial, red de neuronas, Long-Short Term Memory, clasificación, conductor, maniobras, vehículo.



ABSTRACT

This paper presents the development process of the modeling of driving activity using Artificial Intelligence. The implementation of this model is based on a neural network called Long Short-Term Memory (LSTM), which allows the classification of the actions to be taken by a driver at a given time. The aim of the project is to represent high-level driving maneuvers from a set of vehicle and situation data.

To collect this data set, a driving simulator developed by the CAOS research group is available, which can be used to extract data on various elements of the vehicle, the environment surrounding the vehicle in the simulation and the driver's zone of vision. This set of information is used by the model to determine the task or action being performed by the driver, which then allows the system to determine the intended maneuver.

Once the model has been developed, it is necessary to evaluate its accuracy, as a small number of false positives have been detected, making it difficult to track transitions between maneuvers. Therefore, although the neural network offers promising results, it is necessary to continue working on the model to reduce the number of false positives in order to obtain a system perfectly prepared for its implementation in real vehicles.

Keywords: driving, Artificial Intelligence, neural network, driving, neural network, Long-Short Term Memory, classification, driver, maneuvers , vehicle.



ÍNDICE

1. Introducción	1
1.1 Motivación y objetivos	1
1.2 Descripción del problema	3
1.3 Alcance del proyecto	6
1.4 Marco regulador	7
1.5 Entorno operacional	7
1.6 Restricciones	8
1.7 Organización del documento	9
1.8 Acrónimos	10
2. Estado del arte	12
2.1 Simuladores de conducción	12
2.1.1 Dynisma DMG-1	12
2.1.2 One & One	13
2.2 Sistema avanzado de asistencia a la conducción (ADAS)	14
2.3 Modelos predictivos del comportamiento de un conductor	16
2.3.1 Factores influyentes en la conducción	16
2.3.1.1 Factores internos y externos	17
2.3.1.2 Evaluación de la atención del conductor al volante	19
2.3.2 Azimut DSM (driver status monitor)	20
2.3.3 Hikvision	21
2.4 Modelos de prevención de accidentes	21
2.4.1 Detección de accidentes mediante redes neuronales convolucionales (CNN)	22
2.4.2 Reducción de accidentes urbanos mediante deep learning	23
2.5 Redes neuronales recurrentes (LSTM)	24
2.6 Conclusión estado del arte	26
3. Análisis del sistema	28
3.1 Especificación de requisitos	28
3.1.1 Descripción de requisitos	28
3.1.2 Requisitos funcionales	29
3.1.3 Requisitos no funcionales	38
3.2 Casos de uso	42
3.2.1 Descripción tabular de los casos de uso	42
3.2.2 Matriz de trazabilidad	51
3.2.3 Descripción gráfica de los casos de uso	52
4. Arquitectura, diseño e implementación del sistema	54
4.1 Arquitectura general del sistema	54
4.2 Red long-short term memory (LSTM)	56
4.3. Implementación del sistema	57
4.3.1 Recopilación de datos simulador ADAS	57
4.3.1.1 Recopilación de información del entorno	57
4.3.1.2 Envío y formato de los datos	59



4.3.2 Procesado de datos	61
4.3.2.1 Limpieza y transformación de los datos	61
4.3.2.2 Creación de la variable objetivo (target)	63
4.3.3 Diseño e implementación de la red LSTM	64
4.3.3.1 Ajuste de hiper parámetros	65
4.3.3.2 Entrenamiento	66
4.3.3.3 Clasificación	67
4.3.3.4 Creación de autómatas	69
5. Resultados y evaluación	72
5.1 Plan de pruebas	72
5.2 Matriz de trazabilidad	76
5.3 Evaluación de los resultados obtenidos	77
5.3.1 Evaluación de la clasificación: primera y segunda fase	78
5.3.2 Evaluación de las maniobras definidas por autómatas	80
6. Gestión del proyecto	83
6.1 Metodología de planificación	83
6.2 Planificación del trabajo	84
6.3 Presupuesto	86
6.3.1 Coste de personal	86
6.3.2 Coste de material	88
6.3.4 Costes totales	89
6.4 Impacto socioeconómico	89
6.5 Impacto social y medioambiental	90
7. Conclusiones y trabajos futuros	92
7.1 Conclusiones técnicas	92
7.2 Conclusiones personales	92
7.3 Líneas futuras	93
Referencias	95
ANEXOS	99
Anexo I - Introduction	99
Anexo II - Implementation	101
Anexo III - Evaluation	107
Anexo IV - Conclusion	108
Anexo V - Future lines	109



ÍNDICE DE FIGURAS

Figura 1.1: Figura 1.1: Accidentes mortales y fallecidos a 24h en vías interurbanas 2011-2021 (España)	1
Figura 1.2: Principales causas de accidentes de tráfico.	2
Figura 1.3: Ejemplo datos en formato JSON recibidos del simulador.	4
Figura 1.4: Entorno de simulación basado en el software STISIM DRIVE.	5
Figura 1.5: Modelo jerárquico de actividades.	5
Figura 2.1: Simulador Dynisma DMG-1.	13
Figura 2.2: Simulador One & One presentado en el salón Internacional de la Seguridad 2018.	14
Figura 2.3: Modelo de un sistema avanzado de asistencia a la conducción.	15
Figura 2.4: Modelo jerárquico de conducción de John A. Michon.	17
Figura 2.5: Gráfico de los principales tipos de atención.	19
Figura 2.6: Sistema de reconocimiento de la fatiga del conductor Azimut DSM (Driver Status Monitor).	20
Figura 2.7: Arquitectura de una SSD MobileNet v2.	22
Figura 2.8: Mapa de la accidentalidad de Madrid.	23
Figura 2.9: Esquema del funcionamiento de una Red LSTM.	25
Figura 3.1: Diagrama de casos de uso.	53
Figura 4.1: Arquitectura del entorno de simulación.	55
Figura 4.2: Arquitectura de la red Long-Short Term Memory (LSTM).	56
Figura 4.3: Simulación del funcionamiento de un sensor LiDAR.	58
Figura 4.4: Simulación del funcionamiento de la cámara frontal.	58
Figura 4.5: Simulación cámara frontal y trasera.	59
Figura 4.6: Gráficas de series temporales de cada variable.	62



Figura 4.7: Mapa de calor de la matriz de correlación.	63
Figura 4.8: Matriz de confusión Primera Fase de Clasificación.	68
Figura 4.9: Matriz de confusión Segunda Fase de Clasificación.	69
Figura 4.10: Subgrafo maniobra “Adelantamiento por la derecha”.	71
Figura 4.11: Subgrafo maniobra “Adelantamiento por la izquierda”.	71
Figura 4.12: Subgrafo maniobra “Ajustar distancia de seguridad”.	71
Figura 5.1: Representación gráfica del error cuadrático medio (mse) del modelo primera fase.	78
Figura 5.2: Representación gráfica de la precisión (accuracy) del modelo primera fase.	79
Figura 5.3: Representación gráfica del error cuadrático medio (mse) del modelo segunda fase.	79
Figura 5.4: Representación gráfica de la precisión (accuracy) del modelo segunda fase.	80
Figura 5.5: Grafo del autómata no determinista del recorrido de la simulación.	81
Figura 5.6: Subgrafo maniobra “Adelantamiento por la derecha”.	81
Figura 5.7: Subgrafo maniobra “Adelantamiento por la izquierda”.	81
Figura 5.8: Subgrafo maniobra “Ajustar distancia de seguridad”.	82
Figura 6.1: Etapas de la metodología en cascada.	83
Figura 6.2: Planificación del proyecto en forma de diagrama de Gantt.	86



ÍNDICE DE TABLAS

Tabla 2.1: Frecuencia de conductas de riesgo en la conducción.	18
Tabla 3.1: Tabla modelo para la especificación de requisitos.	28
Tabla 3.2: Requisito Funcional RF-01.	30
Tabla 3.3: Requisito Funcional RF-02.	30
Tabla 3.4: Requisito Funcional RF-03.	31
Tabla 3.5: Requisito Funcional RF-04.	31
Tabla 3.6: Requisito Funcional RF-05.	32
Tabla 3.7: Requisito Funcional RF-06.	32
Tabla 3.8: Requisito Funcional RF-07.	33
Tabla 3.9: Requisito Funcional RF-08.	33
Tabla 3.10: Requisito Funcional RF-09.	34
Tabla 3.11: Requisito Funcional RF-10.	34
Tabla 3.12: Requisito Funcional RF-11.	35
Tabla 3.13: Requisito Funcional RF-12.	35
Tabla 3.14: Requisito Funcional RF-13.	36
Tabla 3.15: Requisito Funcional RF-14.	36
Tabla 3.16: Requisito Funcional RF-15.	37
Tabla 3.17: Requisito Funcional RF-16.	37
Tabla 3.18: Requisito Funcional RF-17.	38
Tabla 3.19: Requisito Funcional RF-18.	38
Tabla 3.20: Requisito Funcional RNF-01.	39
Tabla 3.21: Requisito Funcional RNF-02.	39
Tabla 3.22: Requisito Funcional RNF-03.	40



Tabla 3.23: Requisito Funcional RNF-04.	40
Tabla 3.24: Requisito Funcional RNF-05.	41
Tabla 3.25: Requisito Funcional RNF-06.	41
Tabla 3.26: Requisito Funcional RNF-07.	42
Tabla 3.27: Tabla modelo para casos de uso.	42
Tabla 3.28: Caso de uso CU-01.	43
Tabla 3.29: Caso de uso CU-02.	44
Tabla 3.30: Caso de uso CU-03.	44
Tabla 3.31: Caso de uso CU-04.	45
Tabla 3.32: Caso de uso CU-05.	45
Tabla 3.33: Caso de uso CU-06.	46
Tabla 3.34: Caso de uso CU-07.	46
Tabla 3.35: Caso de uso CU-08.	47
Tabla 3.36: Caso de uso CU-09.	47
Tabla 3.37: Caso de uso CU-10.	48
Tabla 3.38: Caso de uso CU-11.	48
Tabla 3.39: Caso de uso CU-12.	49
Tabla 3.40: Caso de uso CU-13.	49
Tabla 3.41: Caso de uso CU-14.	50
Tabla 3.42: Caso de uso CU-15.	50
Tabla 3.43: Caso de uso CU-16.	51
Tabla 3.44: Matriz de trazabilidad.	52
Tabla 5.1: Plantilla documentación pruebas.	72
Tabla 5.2: Prueba P-01.	73
Tabla 5.3: Prueba P-02.	73



Tabla 5.4: Prueba P-03.	73
Tabla 5.5: Prueba P-04.	74
Tabla 5.6: Prueba P-05.	74
Tabla 5.7: Prueba P-06.	74
Tabla 5.8: Prueba P-07.	75
Tabla 5.9: Prueba P-08.	75
Tabla 5.10: Prueba P-09.	75
Tabla 5.11: Prueba P-10.	76
Tabla 5.12: Matriz de trazabilidad.	77
Tabla 6.1: Planificación del proyecto por tareas realizadas.	85
Tabla 6.2: Equivalencia entre roles y categorías profesionales según el convenio.	87
Tabla 6.3: Costes de personal asociados al proyecto.	87
Tabla 6.4: Costes de materiales asociados al proyecto.	88
Tabla 6.5: Costes directos e indirectos.	89
Tabla 6.6: Costes totales del proyecto.	89

1. Introducción

En el presente capítulo se aborda la introducción del Trabajo Fin de Grado, en la que se expone el problema que ha motivado el desarrollo de este proyecto, así como sus objetivos y las razones que han impulsado su realización. Además, se incluye una descripción del alcance del proyecto, el marco regulador aplicable, el entorno operacional y las restricciones de software y hardware que deben ser consideradas. Por último, se proporciona una información sobre la organización y estructura del documento.

1.1 Motivación y objetivos

Los vehículos personales, especialmente los automóviles, son uno de los medios de transporte más utilizados para desplazamientos de media y corta distancia. Sin embargo, según los informes de la Organización Mundial de la Salud (OMS), los accidentes de tráfico son una de las principales causas de mortalidad en el mundo [1]. De acuerdo con esta fuente, los accidentes de tráfico causan alrededor de 1,25 millones de muertes anualmente y entre 20 y 50 millones de personas sufren traumatismos no mortales, que en muchos casos pueden derivar en discapacidad (ver Figura 1.1).

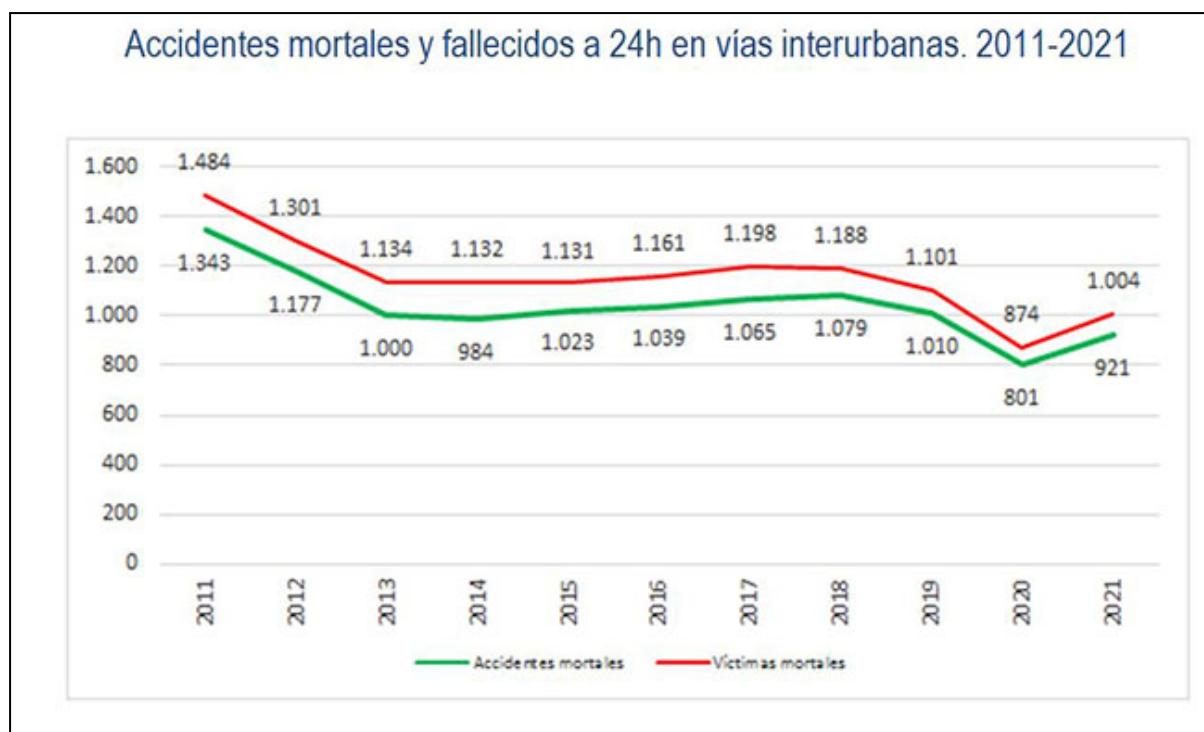


Figura 1.1: Accidentes mortales y fallecidos a 24h en vías interurbanas 2011-2021 (España).[2]

Además, se estima que los accidentes de tráfico son la principal causa de muerte en niños y jóvenes de entre 5 y 29 años, y que en países con ingresos bajos o medios representan el 90% de las causas de defunción. Entre los principales factores de riesgo de estos accidentes se

encuentran la velocidad excesiva, la conducción distraída, la conducción bajo los efectos del alcohol y otras drogas, los vehículos inseguros y las infraestructuras de las vías inseguras (ver Figura 1.2).

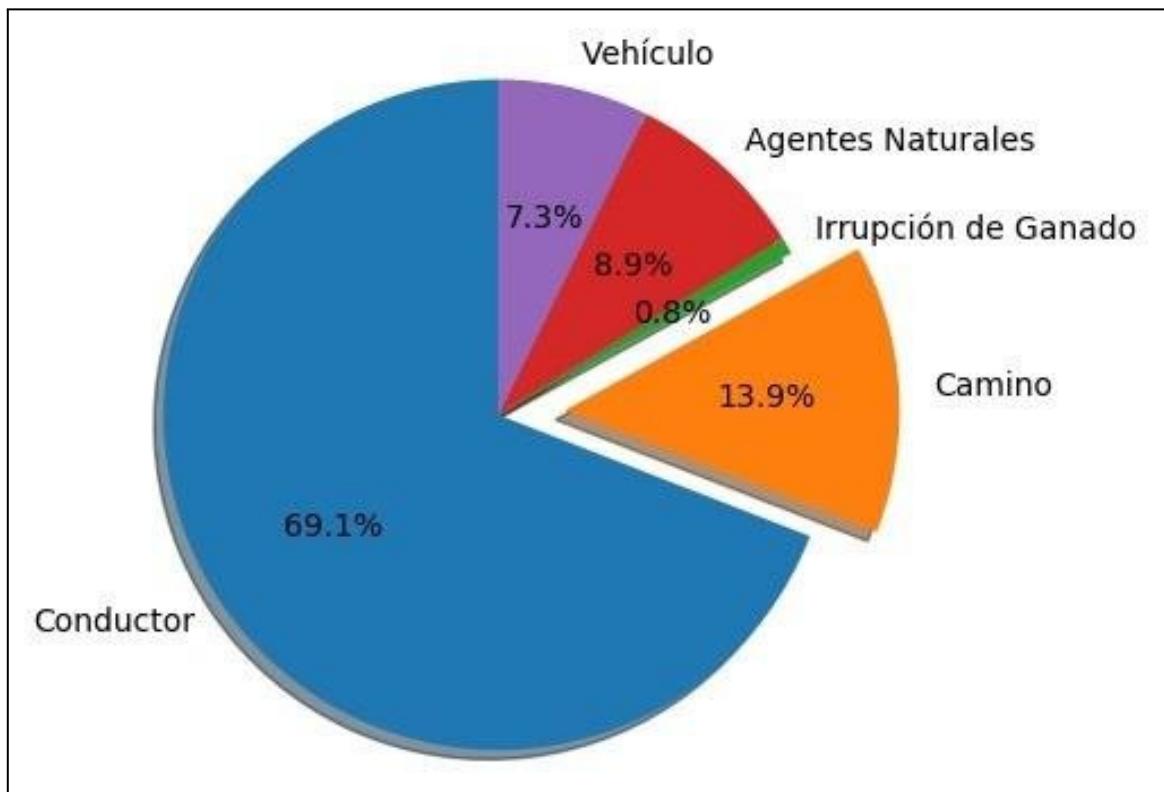


Figura 1.2: Principales causas de accidentes de tráfico [3].

Como se observa en la Figura anterior, la principal causa de accidentes mortales son los conductores de vehículos. En consecuencia, en las últimas décadas se han desarrollado mecanismos para prevenir estos sucesos, como los sistemas de asistencia a la conducción. Estos sistemas utilizan sensores, que pueden funcionar tanto con luz natural durante el día como con sistemas de infrarrojos durante la noche, para localizar el rostro y los ojos del conductor y analizarlos a lo largo del tiempo para generar un índice de somnolencia y uno de distracción [4].

Por otra parte, se están desarrollando simuladores que utilizan realidad mixta¹ para recopilar todo tipo de datos de la simulación y crear modelos que puedan predecir situaciones y, en caso necesario, advertir al conductor de posibles peligros. Una empresa española llamada LANDER SIMULATION & TRAINING SOLUTIONS S.A. [5] trabaja en este campo, desarrollando e implementando dispositivos de simulación comercial de última generación orientados a la formación de personas para ayudarles a manejar vehículos de forma segura y evitar futuros accidentes de tráfico [6].

¹ La realidad mixta es una mezcla de universos físicos y digitales, que permite interacciones 3D naturales e intuitivas entre personas, equipos y el entorno



Por todas estas razones, las instituciones están interesadas en recopilar y clasificar este tipo de datos con el objetivo de minimizar el número de accidentes de tráfico en el futuro [7]. Todos estos sistemas permiten generar modelos y asistentes de conducción que faciliten la vida de los conductores y hagan la infraestructura del vehículo más segura.

El objetivo final de este Trabajo Fin de Grado es contribuir a la recopilación y clasificación de datos provenientes de una simulación de conducción en tiempo real para su posterior análisis y uso en diferentes modelos.

1.2 Descripción del problema

El grupo de investigación CAOS del Departamento de Informática de la Universidad Carlos III de Madrid ha estado a cargo del desarrollo de un sistema multiagente que forma parte de un Sistema Avanzado de Asistencia a la Conducción [8]. En el marco de este estudio, los últimos trabajos realizados con este proyecto se han centrado en predecir las actividades del conductor en el momento de tomar decisiones con el objetivo de mejorar la prevención de accidentes de tráfico [9].

Dentro de este sistema, los datos se dividen en diferentes contextos que forman una ontología en la que se puede representar la situación y entradas del vehículo, el entorno de la simulación y la información del conductor. En consecuencia, se pueden dividir los datos en:

- Información del vehículo: por ejemplo, velocidad, marcha, aceleración o intensidad de frenado, o grado de inclinación del volante.
- Información del entorno: por ejemplo, distancia a cualquier ángulo con el vehículo o los peatones más cercanos.
- Información del conductor: por ejemplo, área de visión o grado de inclinación de la cabeza.

Estos datos se recogen gracias a la comunicación entre sensores y el ordenador principal (ADAS), que se comunican entre sí mediante un protocolo MQTT y forman un archivo JSON con cada uno de los parámetros recogidos en la simulación (ver Figura 1.3).

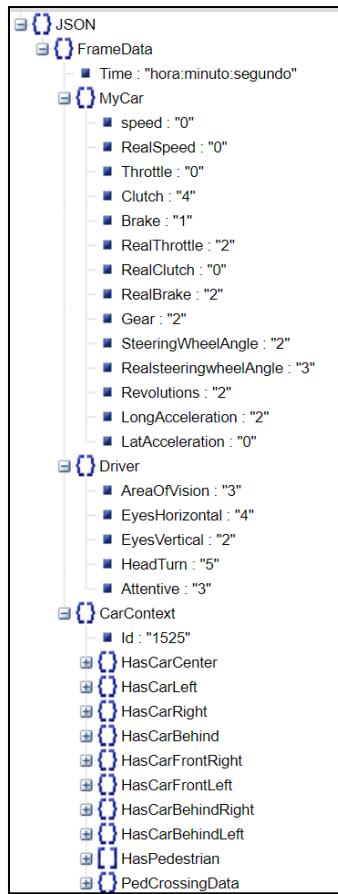


Figura 1.3: Ejemplo datos en formato JSON recibidos del simulador.

Para el desarrollo de estas simulaciones, el grupo cuenta con un simulador de conducción basado en STISIM DRIVE (ver Figura 1.4). Este software permite recrear un entorno de conducción en tiempo real y recopilar los datos en formato JSON. Para el funcionamiento del sistema se dispone de:

- Tres pantallas que visualizan el escenario simulado.
- Un juego de volante, pedales y cambio de marchas, así como un asiento apropiado para la conducción, con el fin de que la simulación sea lo más realista posible.
- Un ordenador en el que se encuentra el ADAS desarrollado.
- Un ordenador en el que se encuentra el simulador basado en STISIM Drive, conectado al resto de sistemas.



Figura 1.4: Entorno de simulación basado en el software STISIM DRIVE. [8]

En este trabajo, los datos referentes al entorno del vehículo y al estado del conductor no serán utilizados, ya que se van a centrar en el escalado de acciones siguiendo el modelo jerárquico de actividades (ver Figura 1.5). De esta manera el proyecto se enfoca en clasificar las actividades del conductor con mayor complejidad a partir de los datos más bajos dentro de la jerarquía (eventos atómicos). Por ejemplo, gracias a datos como la presión del pedal, la inclinación del volante o la marcha en la que se encuentra el vehículo, se puede detectar si un conductor se dispone a realizar un adelantamiento.

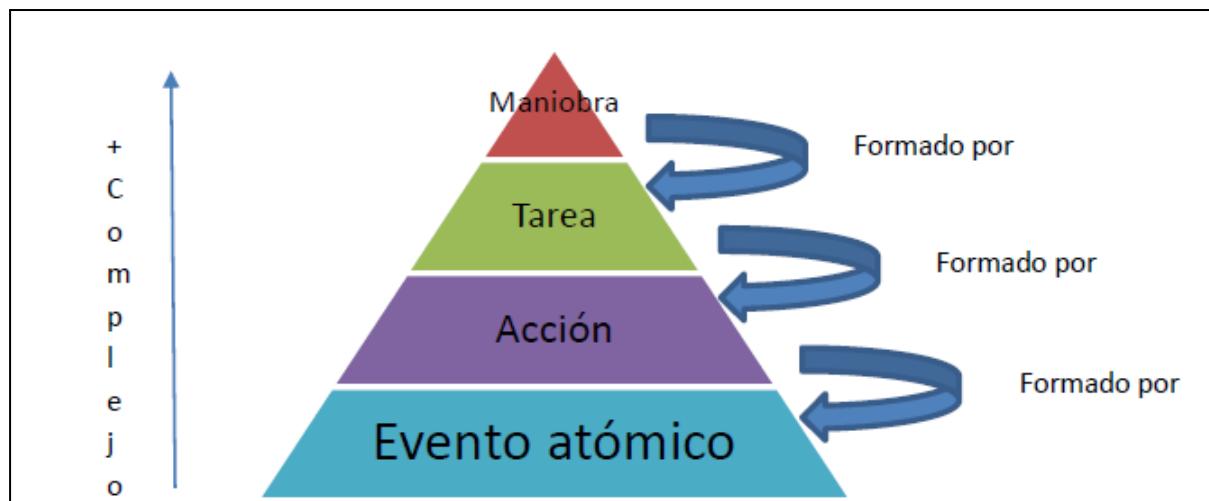


Figura 1.5: Modelo jerárquico de actividades. [38]



Es importante mencionar que el entorno de simulación cuenta con una cámara Microsoft Kinect V2.0 para Windows que permite obtener datos sobre la atención del conductor. Sin embargo, estos datos son irrelevantes para este estudio, por lo que se ha decidido fijarlos para todas las simulaciones.

En la actualidad, se dispone de archivos JSON que resultan complicados de procesar y extraer datos de forma sencilla. No obstante, estos datos se han utilizado para crear ficheros EXCEL que recogen los datos de cada una de las simulaciones en función de la maniobra realizada en el simulador. Por lo tanto, siguiendo los trabajos previos del grupo CAOS, el objetivo de este Trabajo de Fin de Grado es implementar una red neuronal recurrente LSTM que sea capaz de clasificar los datos de estos ficheros en varios niveles de maniobras, y finalmente poder representar el nivel más alto en forma de autómatas.

1.3 Alcance del proyecto

El objetivo principal de este Trabajo Fin de Grado es organizar y clasificar el conjunto de datos recogidos a partir de una simulación de conducción utilizando el sistema de simulación STISIM DRIVE con el objetivo de representar maniobras complejas de conducción. Para ello, el sistema debe procesar los datos de varias simulaciones en las que un conductor determina las acciones a realizar en función del entorno de simulación en el que se encuentra.

Además, el objetivo de este proyecto es desarrollar un modelo de clasificación basado en una red neuronal LSTM para determinar si la actividad de un conductor en un momento dado corresponde a un cambio de dirección hacia la izquierda o la derecha, o si sigue recto. Posteriormente, se llevará a cabo una segunda fase del proyecto en la que se utilizarán los datos obtenidos para clasificar las actividades en maniobras de alto nivel (como adelantamientos) y formar grafos que representan dichas maniobras como un autómata finito determinado.

El alcance del proyecto también incluye los siguientes objetivos específicos:

- Procesado de datos, incluye la limpieza de los mismos y el etiquetado de la variable objetivo a clasificar con la red.
- Diseño e implementación de una red neuronal recurrente Long short-term memory (LSTM).
- Clasificación, en un instante de tiempo concreto, de maniobras de bajo nivel realizadas por un conductor utilizando redes neuronales.
- Escritura de los resultados de la clasificación en un fichero de texto para su posterior procesado y nuevo estudio.
- Ampliación de la clasificación de maniobras de más alto nivel.
- Identificación de maniobras por medio de autómatas finitos determinados definidos por estados.



1.4 Marco regulador

El objetivo del proyecto del cual forma parte este trabajo fin de grado es el desarrollo de un modelo de clasificación de maniobras de conducción por medio de una red neuronal long-short term memory (LSTM), con el fin de conocer el tipo de maniobra de alto nivel que va a realizar un conductor en un instante de tiempo determinado.

El proyecto, hasta el momento, se está desarrollando de la mano de un simulador de conducción, y no de vehículos reales. Esto es debido, a que es una fase temprana de experimentación y sólo con los datos obtenidos de la simulación es suficiente para comenzar el estudio. Lo que supone que no es necesario solicitar ninguna autorización especial.

Sin embargo, el proyecto podría evolucionar hacia la clasificación de maniobras más avanzadas en donde intervenga la posición corporal del conductor o su campo visual. En ese caso, sí sería necesario el cumplimiento de la Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales, recogida en el BOE [32]. Ya que el proyecto necesitará la grabación del conductor en la fase de recogida de datos.

1.5 Entorno operacional

En esta sección se describe el entorno operacional, en el cual ha sido desarrollado el proyecto, especificando las herramientas software y hardware que se han utilizado.

- Ordenador en el que se encuentra el software de simulación, cuyas características principales son:
 - Procesador Intel Core i7-3770 CPU @ 3.4GHz
 - Memoria RAM 16GB
 - Sistema operativo Windows 7
- Ordenador en el que se encuentra el ADAS desarrollado, cuyas características principales son:
 - Procesador Intel Core i5-3470 CPU @ 3.2GHz
 - Memoria RAM 8GB
 - SO Windows 10
- Ordenador personal para desarrollo del proyecto:
 - Procesador AMD Ryzen 7 4800H con Radeon Graphics 2.90 GHz.
 - Memoria RAM de 16GB.
 - SO Windows 10 Home.
- Juego de volante, palanca de cambios y pedales modelo Logitech G27

Y las herramientas software utilizadas han sido:

- Software de simulación STISIM Drive con su correspondiente módulo de desarrollo
- Entorno de desarrollo Visual Studio Code versión 1.77.1
- Entorno de desarrollo Google Collaboratory
- Paquete ofimático de Microsoft Office Professional Plus 2016:
 - Microsoft Word 2016.

- Microsoft Excel 2016.
- Microsoft Project 2016.
- Software de dibujo de diagramas Diagrams.net
- Librería para TensorFlow de alto nivel, Keras

1.6 Restricciones

En este apartado se identifican las restricciones hardware y software del sistema, teniendo en cuenta las herramientas especificadas en el apartado *1.5 Entorno Operacional*. Del mismo modo se detallan los requisitos mínimos para el correcto desarrollo del proyecto:

- Restricciones de hardware:
 - Se debe contar con tres pantallas. Deberán estar colocadas en línea, y las pantallas de los extremos deberán estar giradas en un ángulo de 45° para que la simulación sea lo más parecida posible a la visión real de un conductor.
 - Se debe disponer de volante, palanca de cambios, pedales y asiento apropiado para la conducción, con el fin de darle más realismo a la simulación.
 - Se debe disponer de un ordenador con el software de simulación.
 - Se debe disponer de un ordenador con el software del ADAS.
 - Ambos ordenadores deben tener conexión a Internet y formar parte de la misma red local para poder intercambiar datos.
 - Todos los dispositivos hardware deberán estar correctamente conectados entre sí.
- Restricciones de software:
 - Respecto al ordenador sobre el que se ejecuta el sistema de visión artificial para la monitorización del conductor y el sistema multiagente:
 - Debe tener instalados los drivers de Microsoft Kinect para Windows Runtime Dev V2.2 o superior, y Microsoft Kinect para Windows SDK V2.0 o superior para poder usar las librerías correspondientes que en ellos se incluyen.
 - Por la naturaleza de estos drivers, debe contar con el SO Windows 10.
 - Debe tener instaladas las librerías de Emgu CV en su versión 3.3.0.2824.
 - Para poder ejecutar el sistema multi-agente y los programas codificados en el lenguaje de programación JAVA se debe tener instalado JAVA en su versión 8 (update 171 o superior), y el JDK correspondiente.
 - Respecto al ordenador en el que se ejecuta el simulador de conducción:
 - Debe contar con el software asociado (STISIM Drive 3.14.13 o superior).
 - Respecto al ordenador usado para el entrenamiento del modelo:
 - Deberá tener instalado el lenguaje de programación Python en su versión 3.9.16 o superior compatible.



- Deberá tener instalado la API de alto nivel Keras en su versión 2.12.0 por motivos de compatibilidad.

1.7 Organización del documento

En este apartado se describen brevemente los contenidos de cada una de las secciones de este documento, que conforman la memoria del Trabajo Fin de Grado.

En el [Capítulo 1](#), se presenta la introducción del proyecto. Esta sección incluye la motivación y los objetivos del proyecto, la descripción y el alcance del problema a resolver, el marco regulador, el entorno operacional y las restricciones (tanto de software como de hardware). Además, se detalla la organización del documento y se incluye una lista de acrónimos utilizados a lo largo del documento.

En el [Capítulo 2](#), se desarrolla el estado del arte. En esta sección se expondrá la situación actual de los simuladores de conducción, junto a sus beneficios, y los modelos predictivos más recientes en cuanto a la prevención de accidentes y del comportamiento del conductor. Además, se define el funcionamiento y uso de las redes neuronales recurrentes, y en especial las redes LSTM.

En el [Capítulo 3](#), se presenta el análisis del sistema. En este se establecen los requisitos para el correcto desarrollo del proyecto, tanto funcionales como no funcionales, y se especifican los casos de uso que se contemplan. Además incluye el diagrama de dichos casos de uso y una matriz de trazabilidad entre estos y los requisitos del proyecto.

En el [Capítulo 4](#), se detalla la arquitectura y el diseño del sistema, tanto de la parte del simulador de conducción, como la arquitectura de la red neuronal recurrente LSTM desarrollada para este proyecto. Además, se describe la implementación del sistema desarrollado en el trabajo de fin de grado, explicando el diseño y desarrollo de cada uno de los componentes en los que se descompone el sistema.

En el [Capítulo 5](#), se exponen los resultados finales del trabajo fin de grado, exponiendo las pruebas realizadas. Con el fin de verificar que el sistema funciona correctamente y si los requisitos establecidos se cumplen.

En el [Capítulo 6](#), se definen la planificación (definiendo su metodología) y los costes del proyecto. Además se incluye el impacto esperado del producto, ya sea en el sector socioeconómico, ambiental o social.

En el [Capítulo 7](#), se analiza y se extraen las conclusiones finales del trabajo fin de grado, tanto técnicas como personales. Además se definen los trabajos futuros que se podrían realizar y mejorar a partir de este trabajo.



Por último, se incluyen las referencias consultadas para la documentación e investigación de este trabajo.

1.8 Acrónimos

ADAS: Advanced Driving Assistance System, en castellano, sistema avanzado de ayuda a la conducción.

AMD: Advanced Micro Devices. En castellano, Micro dispositivos avanzados.

API: Application Programming Interfaces. En castellano, Interfaz de Programación de Aplicaciones.

BOE: Boletín Oficial del Estado.

CAOS: Control Aprendizaje y Optimización de Sistemas.

CNN: Convolutional Neural Network. En castellano, Redes Neuronales Convolucionales.

CPU: Central Processing Unit. En castellano, unidad central de procesamiento.

CSV: Comma Separated Values. En castellano, Valores Separados por Comas.

CU: Casos de Uso.

DGT: Dirección General de Tráfico.

DL: Deep Learning. En castellano, Aprendizaje Profundo.

DSM: Driver Status Monitor. En castellano, Monitor del Estado del Conductor.

GPS: Global Positioning System. En castellano, Sistema de Posicionamiento Global.

HMI: Human-Machine Interface. En castellano, Interfaz Humano-Máquina.

IA: Inteligencia Artificial.

JDK: JAVA Development Kit, en castellano, Kit de Desarrollo de JAVA.

JSON: JavaScript Object Notation.

LSTM: Long Short-Term Memory. En castellano, Memoria Prolongada de Corto Plazo.

MIT: Massachusetts Institute of Technology. En castellano, Instituto de Tecnología de Massachusetts.

ML: Machine Learning. En castellano, Aprendizaje Automático.

MQTT: Message Queuing Telemetry Transport. En castellano, Protocolo de Mensajería Basado en Estándares.

OEM: Original Equipment Manufacturer. En castellano, Fabricante de Equipos Originales.

OMS: Organización Mundial de la Salud.

PNG: Portable Network Graphics. En castellano, Gráficos de Red Portátiles.

RAM: Random Access Memory, en castellano, Memoria de Acceso Aleatorio.



RELU: Rectified Linear Unit, en castellano, Unidad Lineal Rectificada

RF: Requisitos Funcionales.

RNF: Requisitos No Funcionales

RNN: Recurrent Neural Networks. En castellano, Redes Neuronales Recurrentes.

RPM: Revoluciones Por Minuto.

SAFE: Stop Atropellos de Fauna en España.

SDK: Software Development Kit, en castellano, Kit de Desarrollo de Software

SICUR: Salón Internacional de la Seguridad.

SO: Sistema Operativo.

UOC: Universidad Oberta de Catalunya.



2. Estado del arte

En este capítulo se analiza el estado actual de los simuladores de conducción y los modelos utilizados para predecir el comportamiento de un conductor y prevenir accidentes de tráfico. Además, se pretende ofrecer una visión general de las posibles aplicaciones de estos modelos a través de la etiquetación y clasificación de datos proporcionados por los Sistemas Avanzados de Asistencia a la Conducción (ADAS).

2.1 Simuladores de conducción

Los sistemas de simulación de conducción se utilizan con frecuencia para la formación de nuevos conductores mediante prácticas de maniobras o situaciones que puedan darse en la realidad. Por otro lado, se utilizan para concienciar a los conductores noveles o experimentados sobre las consecuencias que pueden producir la ingesta de alcohol y drogas o la conducción temeraria [10]. De la misma manera, estos los emplean conductores profesionales, por ejemplo de Fórmula 1 o MotoGP, para practicar antes de las competiciones y reconocer circuitos.

Además, estos simuladores también se emplean en el estudio de la prevención de accidentes de tráfico. Aunque, la gran parte del riesgo en trayectos largos depende del estado del conductor y de la velocidad de la vía, también influyen otros factores como la meteorología adversa (niebla, lluvias, etc.). Según datos de la revista Tráfico y Seguridad Vial- Invierno 2016-17, de la DGT, en los últimos tres años se han producido 11.655 accidentes de los cuales las cifras más altas de víctimas mortales se han producido con lluvia débil [11].

Por ello, se emplean estos simuladores con el objetivo de recopilar datos de infinidad de escenarios, gracias a la implementación de sensores y multitud de entradas (volante, pedales, caja de cambios, etc.), para su posterior incorporación en los nuevos vehículos con el fin de evitar todo tipo de accidentes en las vías. Por ejemplo, en la Universidad Politécnica de Valencia se realizó un estudio con un simulador de conducción para evaluar las capacidades de conducción de personas con discapacidad [12]. Sin embargo, algunos de los problemas de estos simuladores son la imagen lúdica que pueden llegar a ofrecer de cara al público y a los conductores más noveles o el coste y espacio que son necesarios para disponer de este tipo de herramientas [13].

A continuación, se describen los principales simuladores de conducción en investigación y desarrollo de prevención de accidentes.

2.1.1 Dynisma DMG-1

El Dynisma DMG-1 [14] es un simulador de conducción de alta tecnología desarrollado por la empresa Dynisma con sede en Reino Unido. Se ha definido como "el simulador de conducción más avanzado del mundo para el desarrollo de vehículos a motor y de

"competiciones de motor", y se ha utilizado para realizar pruebas avanzadas para fabricantes de equipos originales (OEM) y proveedores de primer nivel. El simulador tiene una latencia más baja y un ancho de banda más alto que cualquier otro simulador disponible en el mercado, lo que lo hace ideal para pruebas de sistemas ADAS y dinámicas extremas de manera segura.



Figura 2.1: Simulador Dynisma DMG-1.

Además, el simulador permite ahorrar kilómetros de desarrollo y pueden reducir el coste, el impacto de carbono y el tiempo que lleva probar los vehículos, permitiendo a los ingenieros realizar pruebas en un entorno virtual confidencial y 100 % seguro.

2.1.2 One & One

El simulador de conducción "One & One" [15] es un simulador de realidad virtual desarrollado por PONS Seguridad Vial en colaboración con el Instituto Tecnológico de Castilla y León y Drivesim, y es utilizado en la formación de prevención de riesgos laborales en el ámbito de la seguridad vial. El simulador ofrece una formación presencial y totalmente práctica, centrada en la correcta ergonomía, los sistemas de seguridad del vehículo y la recreación de situaciones de conducción en condiciones adversas. Gracias a su avanzado software, el simulador puede reproducir situaciones reales sin riesgo para la persona que está siendo formada. Se presentó como una de las 45 mejores propuestas entre más de 2.000 proyectos presentados en el Salón Internacional de la Seguridad 2018 (SICUR) debido a su alto grado de innovación en seguridad vial laboral.

El simulador ha sido diseñado específicamente para brindar formación en seguridad vial laboral a empresas e instituciones, con el objetivo de reducir el número de empleados fallecidos en accidentes de tráfico durante la jornada laboral. Utiliza una tecnología de realidad virtual avanzada para reproducir situaciones reales de conducción de manera segura para la persona que está siendo formada, sin tener que exponer a sus empleados a situaciones potencialmente peligrosas en la carretera.



Figura 2.2: Simulador One & One presentado en el salón Internacional de la Seguridad 2018.

2.2 Sistema avanzado de asistencia a la conducción (ADAS)

Este punto se enfoca en la concepción y construcción de un Sistema Avanzado de Asistencia a la Conducción (ADAS) dirigido a la prevención de accidentes automovilísticos. Este sistema se centra en optimizar la seguridad vial al integrar datos de diversas fuentes, con el objetivo de lograr mejoras sustanciales en la seguridad, superando los efectos individuales de los datos aislados.

La conducción automovilística implica una toma constante de decisiones que impactan directamente en la seguridad del conductor y de otros usuarios de la vía. No obstante, factores como la interpretación errónea de la información, la percepción incompleta de situaciones de riesgo y la adopción de comportamientos inseguros pueden desencadenar accidentes. Esta problemática se acentúa en conductores mayores o con capacidades disminuidas, sectores que representan una porción considerable de la población conductora.

La proliferación de sistemas ADAS ha introducido nuevas posibilidades para elevar los estándares de seguridad vial. Sin embargo, el diseño de interfaces de usuario (HMI) puede resultar enredado y contraproducente, dada la creciente complejidad de los sistemas. El enfoque se orienta hacia la construcción de interfaces intuitivas y no distractoras. Se establecen directrices para una HMI efectiva [16], incluyendo modos de presentación visual independientes del sonido, combinaciones de colores amigables para usuarios con deficiencias visuales y mensajes concisos que permitan una interpretación y respuesta instantáneas.

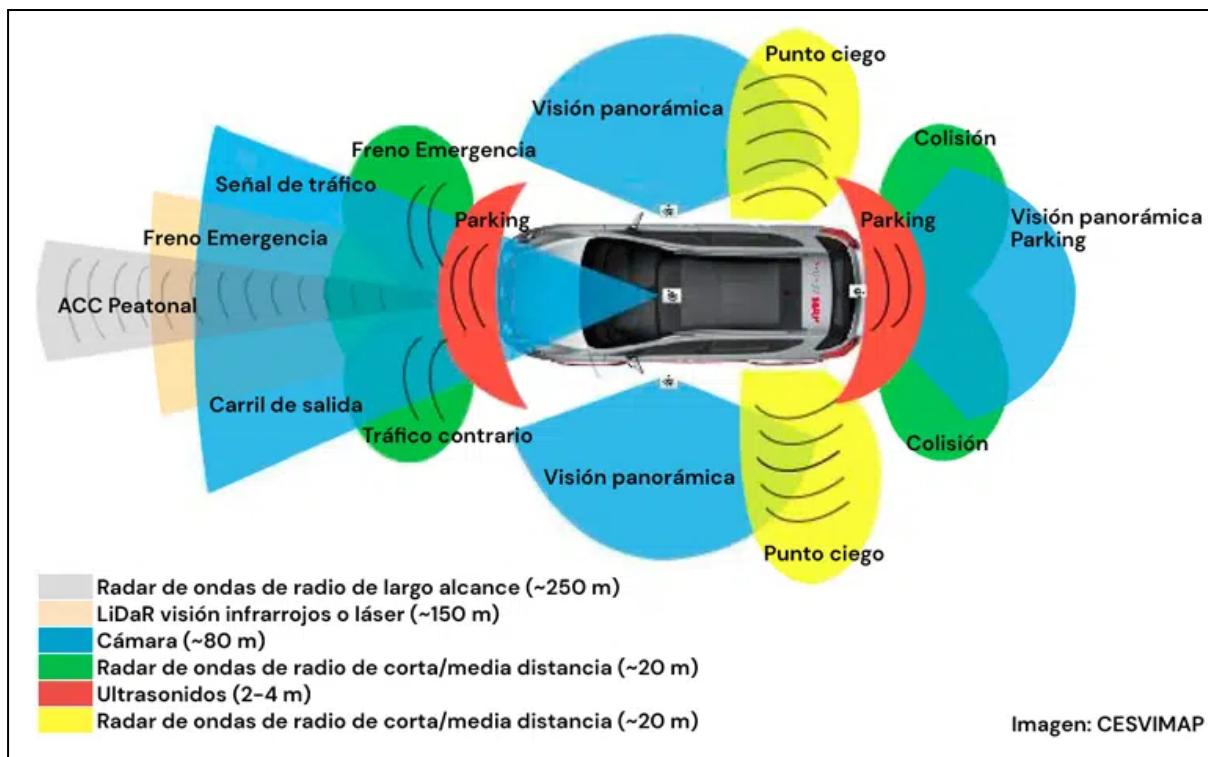


Figura 2.3: Modelo de un sistema avanzado de asistencia a la conducción. [16]

Se reconoce la importancia de adecuar el sistema ADAS a una gama diversa de usuarios. Se han identificado grupos clave, entre ellos conductores generales, jóvenes conductores, personas mayores y conductores con discapacidades. Cada grupo presenta necesidades particulares, demandando una presentación de la información adaptada. Usuarios generales necesitan contextualización, jóvenes deben ser alertados sobre alta velocidad y consumo de alcohol, personas mayores requieren mensajes claros y directos, y conductores con discapacidades necesitan información priorizada conforme a sus limitaciones específicas.

En resumen, el desarrollo de un sistema ADAS efectivo para la prevención de accidentes vehiculares implica la sinergia de información proveniente de diversas fuentes, un diseño de interfaz de usuario inteligente y adaptado, y la consideración de una gama variada de conductores. Estas medidas, tomadas en conjunto, poseen el potencial de elevar los estándares de seguridad vial, reducir accidentes y contribuir a un entorno de conducción más seguro y eficaz en el presente y en el futuro.

2.3 Modelos predictivos del comportamiento de un conductor

El comportamiento o actitud al volante son factores importantes para evitar accidentes y mejorar el tránsito en las carreteras. Según estudios recientes, el 38,1% de los conductores de vehículos ligeros exceden el límite de velocidad permitido y el 67,1% de los vehículos no usan el intermitente al incorporarse a su carril de origen después de realizar un adelantamiento [17]. Este tipo de comportamientos incorrectos no se deben a la falta de conocimientos, sino a una falta de actitud al volante que puede ser provocada por la fatiga o el estrés.

Por esta razón, se han llevado a cabo estudios que, utilizando una serie de sensores y el llamado "tracking", recopilan información sobre el estado de grupos de conductores al volante con el objetivo de elaborar modelos, con diferentes técnicas de inteligencia artificial (como redes neuronales artificiales o lógica difusa), que puedan predecir y, en algunos casos, advertir cuando un conductor actúa de manera incorrecta o se encuentra en peligro.

A continuación, se describen los principales factores que influyen en la conducción y algunos de los modelos que estudian el comportamiento de un conductor utilizando técnicas de Inteligencia Artificial (IA).

2.3.1 Factores influyentes en la conducción

En este trabajo fin de grado, se analiza la influencia de diversos factores en el comportamiento de un conductor al volante. Según el catedrático John A. Michon (1985), la conducción puede dividirse en tres niveles de actividad: el nivel de control, enfocado en mantener el vehículo en su trayectoria; el nivel de táctica o de maniobra; y el nivel estratégico o de planificación de rutas. Estos niveles varían en cuanto a su complejidad y la escala de tiempo en la que se desarrollan. En este apartado, se examinarán los factores internos y externos que pueden afectar a estos niveles de actividad y cómo pueden influir en el comportamiento de un conductor en la carretera [18].

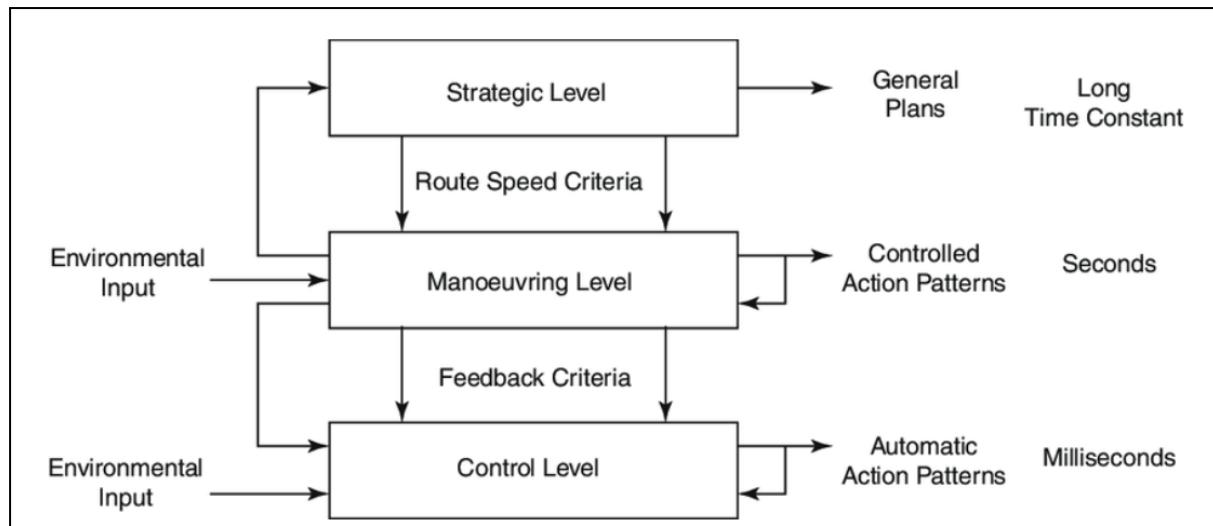


Figura 2.4: Modelo jerárquico de conducción de John A. Michon. [18]

Para profundizar en estos factores, a continuación se muestran algunos de los factores externos e internos que más influyen en la conducción y las diferentes evaluaciones entre los tipos de atención que puede experimentar un conductor.

2.3.1.1 Factores internos y externos

Los factores externos son elementos, naturales o artificiales, que pueden facilitar la pérdida de atención de los conductores. En ocasiones, un simple vistazo fuera de la vía puede desencadenar en una tragedia [19]. Dentro de estos factores externos se pueden encontrar:

- El manejo de la radio o de teléfonos móviles: representan pequeñas distracciones que reciben atención por parte del conductor.
- Comer, beber o fumar dentro del vehículo: esto aparte de ser considerado una distracción, evita que el conductor disponga de las dos manos en caso de peligro.
- La compañía al volante: hablar con los acompañantes del vehículo puede generar distracciones.
- Señalización excesiva: la abundancia de señales despista al conductor de la carretera.
- La publicidad: al igual que las señales, suponen una distracción en la carretera.
- Accidentes de tráfico: otros accidentes pueden distraer al conductor y provocar uno nuevo.
- La climatología: los temporales adversos, como la lluvia o la niebla, pueden provocar distracciones o falta de visión para el conductor.

Por otro lado, existen factores que influyen en el estado físico y mental, son los denominados factores internos. Estos se diferencian de los anteriores en que no pueden ser detectados a simple vista y pueden llegar a ser todavía más peligrosos [20]. dentro de estos factores externos se encuentran:

- Las enfermedades y los medicamentos: muchos de estos pueden provocar somnolencia en el conductor o afectar alguno de sus sentidos.
- La fatiga: se corresponde con un descenso de las capacidades del conductor, manifestándose en cansancio corporal y en la disminución de la concentración.
- El sueño: de la misma manera que la fatiga, genera somnolencia y cansancio en el conductor.
- La vista: las deficiencias visuales no corregidas generan fatiga visual en el conductor, del mismo modo que generan un gran riesgo a la hora de conducir.
- El oído: las deficiencias auditivas que no puedan ser corregidas suponen un riesgo para el conductor. Del mismo modo, otras conductas como llevar la música muy alta también generan riesgo al impedir al conductor percibir su entorno.
- El estado anímico: el estado mental del conductor puede ser una causa de riesgo, ya sea por encontrarse en un estado alterado, con excesiva relajación, etc.

A continuación en la Tabla 2.1, se muestra un resumen de las principales causas de riesgo en las distintas actitudes de conducción, con su porcentaje asociado, recogido en el artículo “*Actitudes y conductas de riesgo en la conducción*²” de la revista Psicología para América Latina.

	Nunca (%)	Alguna vez (%)	Algunas veces (%)	Varias veces (%)	Muchas veces (%)
Exceder velocidad permitida	32.5	24.6	20.9	11.6	10.4
Manejar después de beber alcohol	44.8	35.4	13.1	5.2	1.5
Distracción al conducir	26.5	39.6	22.4	8.2	3.4
Uso de celular al conducir	40.7	28.7	16.4	9.7	4.5
Manejar cansado	43.3	32.5	15.3	6	3
Hacer otras cosas además de conducir	49.6	29.1	15.7	3.7	1.9
Cruzar en Rojo	47	36.2	11.6	4.1	1.1
Maniobra peligrosa	41.4	36.6	13.1	4.5	4.5
No usar Cinturón	66	16.8	6.3	4.1	6.7
No reducir velocidad en las esquinas	49.6	34.3	10.4	3.4	2.2
Violar norma vial	34.3	48.9	13.1	2.6	1.1
Conducción agresiva	73.9	19.4	3.4	1.9	1.5

Tabla 2.1: Frecuencia de conductas de riesgo en la conducción. [21]

² Escrito por Tosi, Jeremías, Trógolo, Mario y Ledesma, Rubén D. Año 2019 [21]

2.3.1.2 Evaluación de la atención del conductor al volante

Aparte de los factores descritos en el apartado anterior, se encuentran la atención del conductor cuando se encuentra al volante. El sistema cognitivo impone una restricción para que solo seamos conscientes de una pequeña parte de esa información, ante la enorme cantidad de estímulos que en cada momento llegan a nuestros sentidos. Esto provoca que los conductores expertos generen automatismos o comportamientos reflejo que aumentan la eficiencia y la fluidez de la conducción. Sin embargo, no deben sustituir a una atención óptima de lo que sucede en la carretera.

En este sentido, existen tres tipos de atención, que se dividen en función de su carácter intensivo o selectivo:

1. La atención selectiva o focalizada se refiere a la capacidad de prestar atención a un determinado estímulo o tarea, mientras se bloquea la información irrelevante.
2. La atención dividida o multitarea implica la atención simultánea a varias fuentes de información o la realización conjunta de dos o más tareas.
3. La atención sostenida o vigilada se refiere a la capacidad de prestar atención durante un período prolongado de tiempo.

Además, la atención se puede entender como un mecanismo de alerta o activación, que implica cambios fisiológicos para procesar la información del entorno. También se puede considerar como una reserva de recursos de procesamiento, es decir, una capacidad o energía mental que se puede utilizar para procesar la información. Por último, la atención puede entenderse como un mecanismo de selección o filtro de la información circundante, lo que permite su selección y procesamiento (ver Figura 2.5).

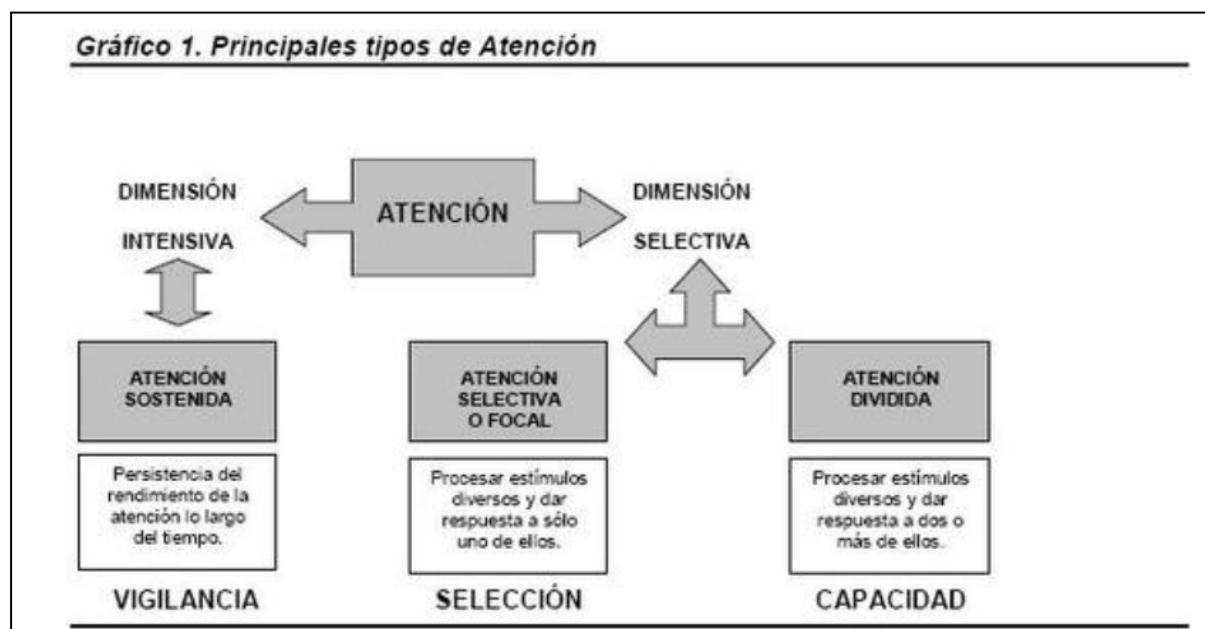


Figura 2.5: Gráfico de los principales tipos de atención. [22]

2.3.2 Azimut DSM (driver status monitor)

Azimut DSM es un sistema de advertencia auxiliar de conducción que utiliza tecnología de visión artificial [23]. Su propósito principal es detectar el estado del conductor con el fin de evitar comportamientos peligrosos durante la conducción. El sistema alerta al conductor a través de dos alertas, visual y sonora.

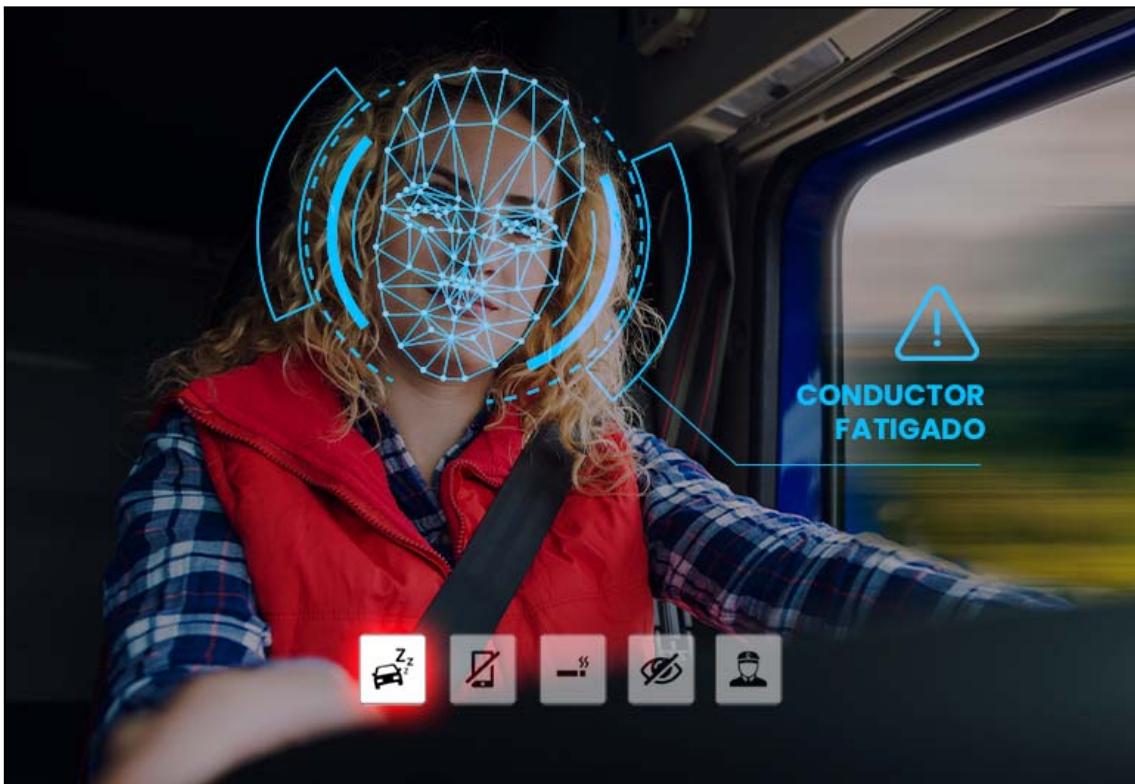


Figura 2.6: Sistema de reconocimiento de la fatiga del conductor Azimut DSM (Driver Status Monitor). [23]

Para su funcionamiento, se necesita una cámara de infrarrojos ultrasensible. Gracias a estos sensores y a un algoritmo basado en Deep Learning, el sistema genera puntos de control que permiten detectar el estado real del conductor. La cámara está conectada al sistema de inteligencia artificial, que es el encargado de leer, interpretar y predecir las situaciones de riesgo y, en consecuencia, es el encargado de generar las alertas correspondientes para el conductor. Además, el sistema está conectado a una plataforma de gestión en la nube, que emite la grabación.

Algunas de las ventajas que ofrece el sistema Azimut DSM se basan en emplear una tecnología de alto rendimiento. La cámara infrarroja del dispositivo es ultrasensible y detecta todos los aspectos de la cara del conductor durante el día y también durante la noche. Por ejemplo, el uso de gafas no disminuye su sensibilidad. Además, La plataforma está diseñada



para mejorar la eficiencia del servicio y mejorar la seguridad a bordo, de acuerdo con las demandas reales del servicio y dispone de ubicación vídeo en tiempo real.

2.3.3 Hikvision

Hikvision es una empresa de tecnología de vídeo que se dedica al desarrollo de soluciones de vigilancia y seguridad a través de cámaras y sistemas de vídeo inteligentes [24]. Estos sistemas utilizan tecnología de inteligencia artificial para analizar y procesar en tiempo real la información recogida por las cámaras, permitiendo detectar patrones y eventos específicos que puedan ser de interés para la seguridad y el control del tráfico.

La estructura de Hikvision se basa en la utilización de algoritmos de Deep Learning y otras técnicas de inteligencia artificial para analizar y procesar la información recogida por las cámaras. Esto permite detectar patrones y eventos específicos, como comportamientos peligrosos o anormales en la conducción, y enviar alertas en tiempo real a los equipos de emergencia y a las autoridades competentes para que puedan tomar medidas rápidas y adecuadas. Además, estos sistemas también permiten grabar y guardar las imágenes recogidas por las cámaras para su posterior revisión y análisis.

Entre las ventajas de elegir Hikvision, destaca su aplicación prácticamente ilimitada, ya que puede ayudar a distintos tipos de organizaciones en diferentes campos. Por ejemplo, puede ayudar a los bancos a proteger a sus empleados, clientes, sucursales y cajeros automáticos, a los minoristas a comprender la afluencia de público a sus tiendas y a optimizar sus estrategias de comercialización, y a las autoridades municipales a reducir la congestión y la contaminación con soluciones inteligentes de gestión del tráfico.

2.4 Modelos de prevención de accidentes

Los accidentes de tráfico son una de las principales causas de muerte en la población adulta. Muchos de los factores que influyen en estos accidentes son repetitivos, por lo que se han desarrollado proyectos para analizar dichos factores y buscar patrones o correlaciones con el fin de evitar nuevos accidentes. Para lograr este objetivo, se emplean técnicas de aprendizaje automático (ML, machine learning) que generan y entrena diversos modelos a partir de un conjunto de datos.

La idea general de estos modelos es aportar una valiosa información a las empresas automovilísticas para las mejoras de los vehículos y proporcionar datos con los que los gobiernos puedan priorizar los gastos en los principales impulsores de accidentes, como la iluminación o la condición de la carretera [25].

En los siguientes apartados de este proyecto de fin de grado, se describirán algunas de las técnicas empleadas en este tipo de proyectos enfocados en la creación de modelos predictivos de accidentes de tráfico, así como su futuro impacto en la sociedad.

2.4.1 Detección de accidentes mediante redes neuronales convolucionales (CNN)

En este proyecto, desarrollado por un estudiante de la Universidad Europea de Madrid, se utilizaron redes neuronales convolucionales para detectar situaciones de peligro en un conjunto de videos. Para su implementación fue necesario un correcto tratamiento del conjunto de datos, en especial las imágenes (frames) en los que se separaron los vídeos y el entrenamiento y evaluación del modelo con la correspondiente CNN (Convolutional Neural Network) [26].

Los frames resultantes del preprocesado de datos se etiquetan utilizando la herramienta “LabelIMG” obteniendo información sobre la posición, en sus coordenadas x e y, dónde se encuentra el vehículo siniestrado. Con todo el conjunto de imágenes etiquetados, se emplea una red neuronal convolucional pre-entrenada, creada a partir de la API de detección de objetos de “TensorFlow”, como *SSD MobileNet v2*, que emplea hasta seis filtros de capas de convolución separables por profundidad.

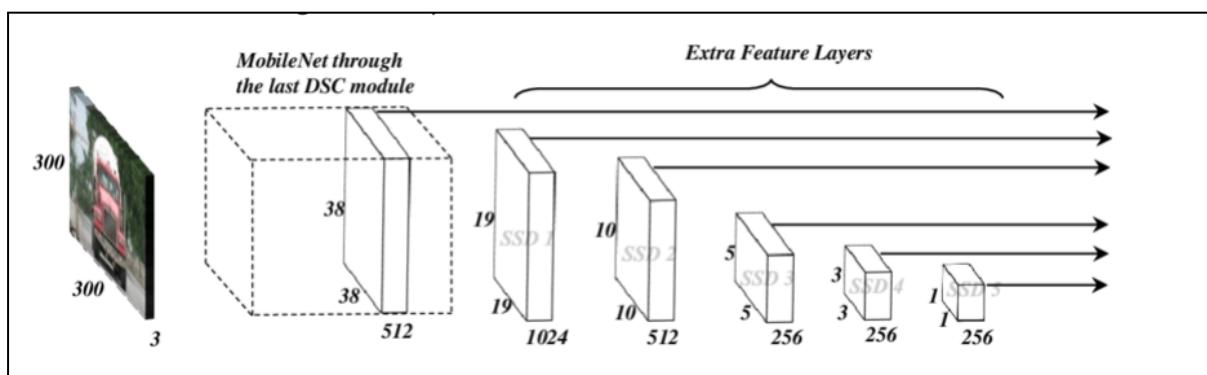


Figura 2.7: Arquitectura de una SSD MobileNet v2. [26]

Una vez realizado el entrenamiento de este proyecto, los resultados obtenidos permitieron crear gráficos que representan la cantidad de accidentes ocurridos según el estado meteorológico de la situación, la hora del día en el que ocurren o cuales son los distritos donde se originan más siniestros. Además, se representó un mapa de la ciudad de Madrid con los puntos donde se han detectado la mayor parte de los accidentes de tráfico.

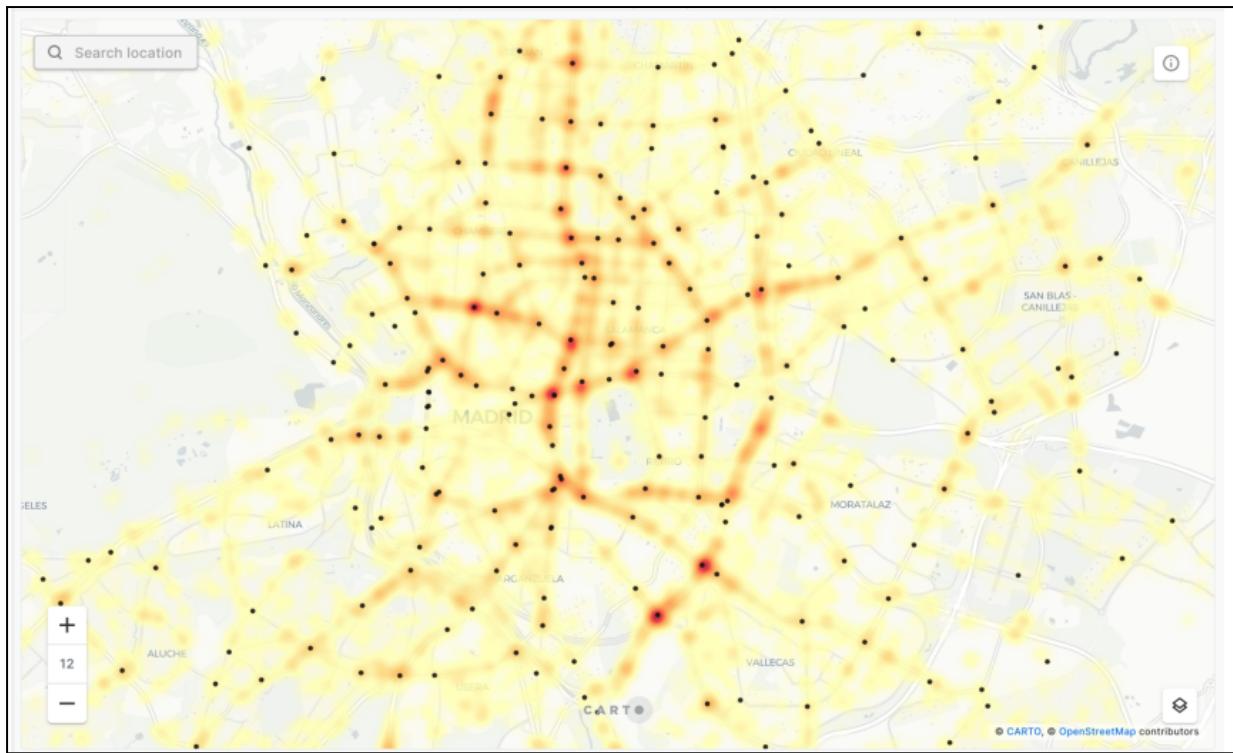


Figura 2.8: Mapa de la accidentalidad de Madrid. [26]

2.4.2 Reducción de accidentes urbanos mediante deep learning

Un grupo interdisciplinario de investigadores de la Universitat Oberta de Catalunya (UOC), el Instituto Tecnológico de Massachusetts (MIT) y la Universitat Rovira i Virgili [27], en colaboración con la Dirección General de Tráfico (DGT) y los ayuntamientos de Madrid y Barcelona, está utilizando inteligencia artificial (IA) para ayudar en la toma de decisiones que conviertan las ciudades en espacios más seguros en términos de seguridad vial. La investigación ha demostrado la relación entre la complejidad urbana y la probabilidad de sufrir un accidente en la carretera.

Los datos recogidos se utilizan para entrenar redes neuronales que pueden detectar posibles peligros dentro de un espacio y los patrones asociados a estos obstáculos. Los investigadores descubrieron que la configuración visual de la "escena urbana", como la ubicación del mobiliario urbano, la situación de los coches aparcados, los anuncios o las fachadas, puede afectar la probabilidad de que ocurra un accidente de tráfico.

Para este proyecto, se utilizaron técnicas de Deep Learning (DL), ya que disponen de la capacidad de generar modelos que aprendan de grandes cantidades de datos y generalicen patrones. Permitiendo a las compañías y agencias gubernamentales desarrollar sistemas de seguridad avanzados, como la detección de obstáculos, la asistencia a la conducción y la conducción autónoma, que están ayudando a reducir el número de accidentes y a salvar vidas en todo el mundo.



La hipótesis resultante de este estudio, indica que las limitaciones cognitivas humanas se ven afectadas por la complejidad de la escena urbana y que la IA puede ayudar a reducir los accidentes de tráfico gracias a la generalización de patrones y a la identificación de situaciones peligrosas. La idea es poder diseñar ciudades más seguras en cuanto al tráfico y la disposición de las propias calles en un futuro no muy lejano.

2.5 Redes neuronales recurrentes (LSTM)

Para comprender el trabajo que se va a realizar en este proyecto es necesario definir el concepto y funcionamiento de una red neuronal recurrente (RNN), y especialmente las Redes Neuronales de Memoria Prolongada de Corto Plazo (Long Short-Term Memory, LSTM).

En primer lugar, las redes neuronales recurrentes [28] son un tipo de modelo de aprendizaje automático que se utiliza para procesar secuencias de datos, como texto, audio y vídeo. A diferencia de las redes neuronales convencionales, que procesan cada entrada de manera independiente, las redes neuronales recurrentes utilizan la información contextual previa para hacer predicciones y tomar decisiones.

El funcionamiento de las redes neuronales recurrentes se basa en el uso de unidades de memoria llamadas celdas de memoria. Estas celdas de memoria se actualizan continuamente a medida que la red procesa una secuencia de datos, permitiendo que la red retenga información contextual y utilice esta información para hacer predicciones en el futuro.

Cada celda de memoria en una red neuronal recurrente contiene una serie de pesos y conexiones que se utilizan para procesar la entrada actual y actualizar la información contextual. La entrada actual se combina con la información contextual anterior y se utiliza para calcular una salida y actualizar la información contextual actualizada. Esta información contextual actualizada se transmite a la siguiente celda de memoria en la secuencia, lo que permite a la red procesar la entrada actual y mantener una representación contextual de toda la secuencia.

A medida que la red procesa más datos, las celdas de memoria en la red pueden aprender patrones en los datos y utilizar estos patrones para hacer predicciones precisas en el futuro. Además, las redes neuronales recurrentes también pueden ser entrenadas para realizar tareas específicas, como reconocimiento de voz y análisis de texto, mediante el uso de técnicas de aprendizaje supervisado.

Por otro lado, las redes neuronales LSTM [29] son una variante de RNN diseñada para resolver el problema del desvanecimiento del gradiente, que puede ocurrir en RNN convencionales al entrenar con secuencias largas. Los LSTM utilizan una estructura de celdas de memoria que puede mantener información a largo plazo. Las unidades LSTM abordan este problema permitiendo que la información fluya sin cambios a través de la red en forma de una "célula de memoria", que se actualiza y se olvida selectivamente.

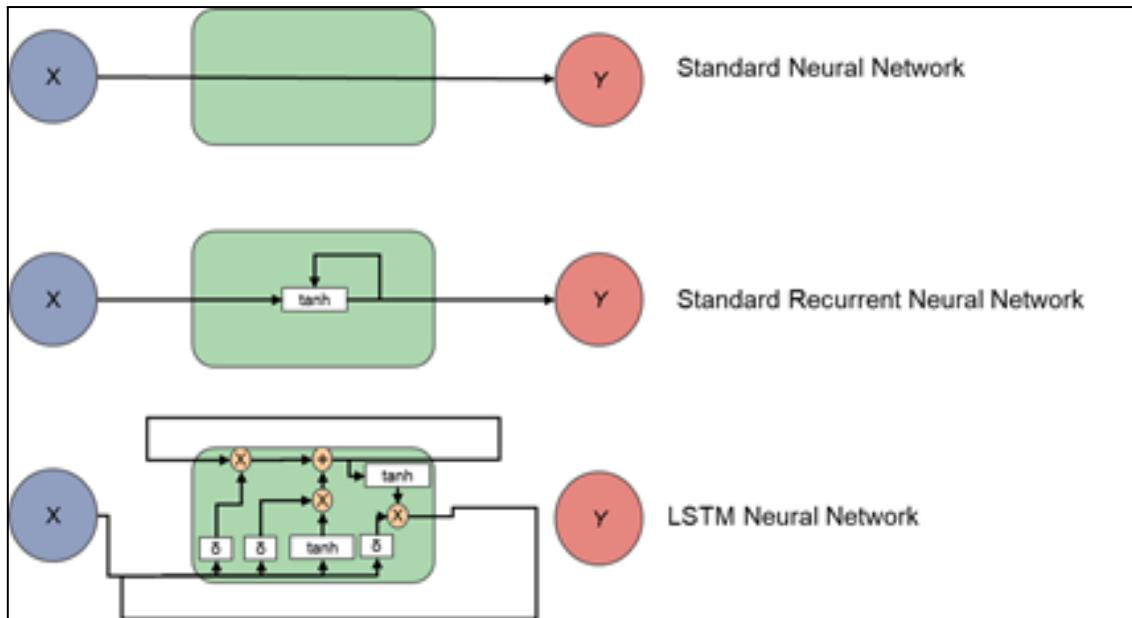


Figura 2.9: Esquema del funcionamiento de una Red LSTM. [29]

Cada unidad LSTM consta de varias compuertas, que son módulos que controlan el flujo de información en la red. Las compuertas principales son la compuerta de entrada, que decide qué información se va a agregar a la memoria, la compuerta de olvido, que decide qué información se va a olvidar de la memoria y la compuerta de salida, que decide qué información se va a enviar a la salida de la unidad.

La arquitectura de la red LSTM permite que la red aprenda patrones de larga duración en los datos secuenciales, lo que la hace especialmente útil para aplicaciones de procesamiento de lenguaje natural, como la traducción automática, la generación de texto y la clasificación de texto.

De esta forma, los pasos necesarios para desarrollar adecuadamente una red LSTM son los siguientes [29]:

- 1. Recopilar y preparar los datos:** Una vez definido el problema, se debe recopilar y preparar los datos. Es importante tener una cantidad suficiente de datos y que estos estén limpios y bien estructurados. También es importante dividir los datos en conjuntos de entrenamiento, validación y prueba.
- 2. Diseñar la arquitectura de la red:** En este paso se decide cómo será la arquitectura de la red LSTM. Esto implica definir la cantidad de capas que tendrá la red, la cantidad de neuronas por capa, qué tipo de funciones de activación se utilizarán, etc. En este paso también se debe decidir si se utilizarán técnicas de regularización para evitar el sobreajuste.

3. **Preparar los datos para la red:** Antes de entrenar la red LSTM, los datos deben ser preparados para ser ingresados a la red. Esto implica normalizar los datos, cambiar su estructura para que puedan ser ingresados a la red, etc.
4. **Entrenar la red:** En este paso se entrena la red LSTM utilizando el conjunto de entrenamiento. Esto implica definir la función de pérdida que se utilizará para evaluar el desempeño de la red, el optimizador que se utilizará para minimizar la función de pérdida y la cantidad de épocas que se utilizarán para el entrenamiento.
5. **Evaluar la red:** Una vez entrenada la red, se debe evaluar su desempeño utilizando el conjunto de validación. Esto permite verificar que la red no se está sobreajustando y que su desempeño generaliza bien a datos que no ha visto antes.
6. **Ajustar la red:** Si el desempeño de la red en el conjunto de validación no es el esperado, se pueden ajustar los hiperparámetros de la red (como la cantidad de neuronas por capa, la tasa de aprendizaje, etc.) para mejorar su desempeño.
7. **Evaluar la red final:** Una vez que se ha ajustado la red LSTM, se debe evaluar su desempeño final utilizando el conjunto de prueba. Esto permite verificar que la red funciona bien en datos que no ha visto antes y que cumple con el objetivo inicial del proyecto.
8. **Implementar la red:** Si la red LSTM ha demostrado ser efectiva en la resolución del problema, se puede implementar en un ambiente de producción para ser utilizada en situaciones reales.

2.6 Conclusión estado del arte

Después de realizar un estudio previo, se puede observar que se han encontrado multitud de ideas que buscan el mismo objetivo que este trabajo fin de grado, disminuir el número de accidentes en carretera.

Se ha profundizado en temas relacionados con los procesos que se van a recoger en este documento, como el avance y desarrollo de simuladores de conducción, los sistemas avanzados de asistencia a la conducción (ADAS), modelos predictivos de prevención de accidentes y otros modelos centrados en el empleo de redes neuronales recurrentes.

Este enfoque se sustenta en el objetivo general de este proyecto, que es la instauración de un modelo utilizando una red neuronal recurrente para clasificar maniobras de conducción de alto nivel. Esto se realiza a partir de datos de menor escala recopilados en diversas situaciones dentro de un simulador de conducción.

En resumen, se ha llevado a cabo una investigación preliminar exhaustiva sobre los principales elementos considerados relevantes para la elaboración de este trabajo fin de



grado. Poniendo énfasis en los cambios generados y los resultados introducidos por investigaciones previas, y en el momento en el que nos encontramos actualmente.



3. Análisis del sistema

En este capítulo se describe la etapa de análisis del sistema, donde se especificarán los requisitos, tanto funcionales como no funcionales, que deben cumplirse para considerar que los objetivos del proyecto han sido alcanzados. Y, finalmente, se detallan los casos de uso del sistema, primero de forma tabular y después de forma gráfica, mediante los cuales se especificarán las situaciones en las que debe funcionar el sistema.

3.1 Especificación de requisitos

En esta sección del capítulo se establecen los requisitos del proyecto, tanto los requisitos funcionales como los requisitos no funcionales. Para ello, antes de la especificación de los requisitos se realiza una descripción de cómo éstos van a ser definidos.

3.1.1 Descripción de requisitos

En este punto se llevará a cabo un análisis exhaustivo para determinar las condiciones mínimas que el sistema debe cumplir para considerarlo exitoso en términos de los objetivos previstos. Para ello, se establecerán una serie de requisitos que deben ser satisfechos para garantizar el cumplimiento de los objetivos del proyecto.

En la **Tabla 3.1** se muestra el formato con el que se definirá cada uno de los requisitos del trabajo.

Tabla 3.1: Tabla modelo para la especificación de requisitos.

Identificador: RX-YY			
Nombre			
Prioridad	Baja	Media	Alta
Versión		Dependencias	
Descripción			
Necesidad	Opcional	Deseable	Esencial
Estabilidad	Baja	Media	Alta
Verificabilidad	Baja	Media	Alta

A continuación, se explica cada uno de los valores de la tabla modelo para la especificación de requisitos:

- **Identificador:** Código único para identificar de forma única cada requisito, y para facilitar la trazabilidad. Los caracteres indicados en la tabla representan:

- **R.** Requisito.
- **X.** Puede tomar los valores *F* y *NF* para indicar que se está especificando un requisito funcional o no funcional, respectivamente.
- **Y.** Toma valores numéricos únicos, para requisitos funcionales y no funcionales de forma independiente, para identificar únicamente el requisito en cuestión.
- **Nombre.** Nombre descriptivo para la identificación de requisitos.
- **Prioridad.** Conjunto de tres valores que indican la prioridad del cumplimiento del requisito en tres niveles: baja, media y alta. La prioridad es útil a la hora de decidir la planificación global del proyecto.
- **Versión.** Versión actual de la definición del requisito, es decir, número de veces que ha sido revisado durante la realización del proyecto.
- **Dependencias.** Requisitos con los que el requisito en cuestión tiene dependencias. Se indican mediante los identificadores de requisito correspondientes.
- **Descripción.** Breve explicación que detalla el requisito en cuestión.
- **Necesidad.** Indica la importancia del requisito para el desarrollo del proyecto en tres niveles: opcional, deseable y esencial. Que un requisito tenga necesidad opcional quiere decir que puede o no ser cumplido sin afectar en gran medida al proyecto en su totalidad. Que un requisito sea deseable quiere decir que su cumplimiento sería bueno para el proyecto, pero no se asegura su cumplimiento por algún motivo. Y, por último, que un requisito tenga necesidad esencial quiere decir que su cumplimiento es obligatorio, y en caso de no cumplirse el proyecto no podría ser llevado a cabo.
- **Escalabilidad.** Conjunto de tres valores que indican la probabilidad de cambio del requisito durante el desarrollo del proyecto en tres niveles: baja, media y alta.
- **Verificabilidad.** Conjunto de tres valores que indican la facilidad que tiene el requisito en cuestión de ser verificado mediante las pruebas en tres niveles: baja, media y alta.

Para la selección en los campos que requieran indicar uno de los valores indicados en la tabla, se colorea y se subraya (para el caso de la impresión o visualización del documento en escala de grises) la opción correspondiente.

3.1.2 Requisitos funcionales

Los requisitos funcionales de un proyecto software especifican qué es lo que debe hacer dicho software. Cada uno de estos requisitos describe una funcionalidad del sistema, junto con la información que se incluye en cada tabla. Los requisitos funcionales del presente trabajo de fin de grado se describen a continuación (desde la Tabla 3.2 hasta la Tabla 3.19).

**Tabla 3.2: Requisito Funcional RF-01.**

Identificador: RF-01			
Nombre	Cargar datos		
Prioridad	Baja	Media	<u>Alta</u>
Versión	1,0	Dependencias	
Descripción	El sistema debe permitir cargar los datos en formato de Data Frames.		
Necesidad	Opcional	Deseable	<u>Esencial</u>
Estabilidad	Baja	Media	<u>Alta</u>
Verificabilidad	Baja	Media	<u>Alta</u>

Tabla 3.3: Requisito Funcional RF-02.

Identificador: RF-02			
Nombre	Visualizar Data Frames		
Prioridad	Baja	<u>Media</u>	Alta
Versión	1.0	Dependencias	RF-01
Descripción	El sistema debe permitir visualizar previamente los data frames que utilizará el entrenamiento del modelo.		
Necesidad	Opcional	<u>Deseable</u>	Esencial
Estabilidad	<u>Baja</u>	Media	Alta
Verificabilidad	Baja	Media	<u>Alta</u>

Tabla 3.4: Requisito Funcional RF-03.

Identificador: RF-03			
Nombre	Visualizar hiper parámetros		
Prioridad	Baja	Media	<u>Alta</u>
Versión	1.0	Dependencias	
Descripción	El sistema debe permitir visualizar los hiper parámetros que utiliza el modelo.		
Necesidad	Opcional	Deseable	<u>Esencial</u>
Estabilidad	Baja	<u>Media</u>	Alta
Verificabilidad	Baja	Media	<u>Alta</u>

Tabla 3.5: Requisito Funcional RF-04.

Identificador: RF-04			
Nombre	Ajustar hiper parámetros		
Prioridad	Baja	Media	<u>Alta</u>
Versión	1.0	Dependencias	RF-01, RF-03
Descripción	El sistema debe permitir ajustar los valores de los hiper parámetros del modelo.		
Necesidad	Opcional	Deseable	<u>Esencial</u>
Estabilidad	Baja	<u>Media</u>	Alta
Verificabilidad	Baja	Media	<u>Alta</u>

**Tabla 3.6: Requisito Funcional RF-05.**

Identificador: RF-05			
Nombre	Visualizar gráficas (mse, accuracy)		
Prioridad	Baja	<u>Media</u>	Alta
Versión	1.0	Dependencias	RF-04
Descripción	El sistema debe permitir visualizar las gráficas correspondientes al error cuadrático medio y la precisión del modelo.		
Necesidad	Opcional	<u>Deseable</u>	Esencial
Estabilidad	Baja	<u>Media</u>	Alta
Verificabilidad	Baja	Media	<u>Alta</u>

Tabla 3.7: Requisito Funcional RF-06.

Identificador: RF-06			
Nombre	Visualizar series temporales		
Prioridad	<u>Baja</u>	Media	Alta
Versión	1.0	Dependencias	RF-01
Descripción	El sistema debe permitir visualizar las gráficas correspondientes para enfrentar cada una de las variables con la serie temporal del modelo.		
Necesidad	<u>Opcional</u>	Deseable	Esencial
Estabilidad	<u>Baja</u>	Media	Alta
Verificabilidad	Baja	Media	<u>Alta</u>



Tabla 3.8: Requisito Funcional RF-07.

Identificador: RF-07			
Nombre	Visualizar mapa de calor		
Prioridad	<u>Baja</u>	Media	Alta
Versión	1.0	Dependencias	RF-01
Descripción	El sistema debe permitir visualizar las gráficas correspondientes al mapa de calor de cada una de las variables que utiliza el modelo.		
Necesidad	<u>Opcional</u>	Deseable	Esencial
Estabilidad	<u>Baja</u>	Media	Alta
Verificabilidad	Baja	Media	<u>Alta</u>

Tabla 3.9: Requisito Funcional RF-08.

Identificador: RF-08			
Nombre	Visualizar matriz de confusión		
Prioridad	Baja	Media	<u>Alta</u>
Versión	1,.0	Dependencias	RF-04
Descripción	El sistema debe permitir visualizar la matriz de confusión, la cual contiene los datos bien clasificados y los falsos positivos, que genera el modelo al ser evaluado.		
Necesidad	Opcional	Deseable	<u>Esencial</u>
Estabilidad	Baja	<u>Media</u>	Alta
Verificabilidad	Baja	Media	<u>Alta</u>

**Tabla 3.10: Requisito Funcional RF-09.**

Identificador: RF-09			
Nombre	Crear un checkpoint que almacene el mejor resultado		
Prioridad	Baja	<u>Media</u>	Alta
Versión	1.0	Dependencias	
Descripción	El sistema debe almacenar el mejor resultado de la evaluación del modelo (dependiendo del menor valor de error cuadrático medio).		
Necesidad	Opcional	<u>Deseable</u>	Esencial
Estabilidad	Baja	<u>Media</u>	Alta
Verificabilidad	Baja	<u>Media</u>	Alta

Tabla 3.11: Requisito Funcional RF-10.

Identificador: RF-10			
Nombre	Ajustar pesos a las clases		
Prioridad	Baja	Media	<u>Alta</u>
Versión	1.0	Dependencias	RF-04
Descripción	El sistema debe permitir ajustar los pesos de las clases a clasificar con el objetivo de obtener el mayor número de datos bien clasificados.		
Necesidad	Opcional	Deseable	<u>Esencial</u>
Estabilidad	Baja	Media	<u>Alta</u>
Verificabilidad	Baja	Media	<u>Alta</u>

**Tabla 3.12: Requisito Funcional RF-11.**

Identificador: RF-11			
Nombre	Preprocesar datos		
Prioridad	Baja	Media	<u>Alta</u>
Versión	1,0	Dependencias	
Descripción	El sistema debe permitir el preprocesado de datos y la definición de una nueva variable objetivo.		
Necesidad	Opcional	Deseable	<u>Esencial</u>
Estabilidad	Baja	Media	<u>Alta</u>
Verificabilidad	Baja	<u>Media</u>	Alta

Tabla 3.13: Requisito Funcional RF-12.

Identificador: RF-12			
Nombre	Actualizar resultados de clasificación		
Prioridad	Baja	Media	Alta
Versión	1,0	Dependencias	RF-11
Descripción	El sistema debe permitir actualizar los resultados de la clasificación dentro del Data Frame.		
Necesidad	Opcional	Deseable	<u>Esencial</u>
Estabilidad	Baja	Media	<u>Alta</u>
Verificabilidad	Baja	Media	<u>Alta</u>



Tabla 3.14: Requisito Funcional RF-13.

Identificador: RF-13			
Nombre	Estratificación de los datos		
Prioridad	Baja	Media	<u>Alta</u>
Versión	1,0	Dependencias	
Descripción	El sistema debe permitir realizar una estratificación de los datos para obtener un conjunto de validación de cara a la evaluación del modelo.		
Necesidad	Opcional	Deseable	<u>Esencial</u>
Estabilidad	Baja	<u>Media</u>	Alta
Verificabilidad	Baja	<u>Media</u>	Alta

Tabla 3.15: Requisito Funcional RF-14.

Identificador: RF-14			
Nombre	Fijar semilla aleatoria		
Prioridad	Baja	<u>Media</u>	Alta
Versión	1,0	Dependencias	
Descripción	El sistema debe permitir fijar una semilla aleatoria para obtener siempre los mismos resultados de evaluación del modelo.		
Necesidad	Opcional	<u>Deseable</u>	Esencial
Estabilidad	Baja	<u>Media</u>	Alta
Verificabilidad	Baja	Media	<u>Alta</u>



Tabla 3.16: Requisito Funcional RF-15.

Identificador: RF-15			
Nombre	Guardar las gráficas		
Prioridad	Baja	<u>Media</u>	Alta
Versión	1,0	Dependencias	
Descripción	El sistema debe permitir guardar las gráficas correspondientes a la evaluación del modelo y la previsualización de datos.		
Necesidad	Opcional	<u>Deseable</u>	Esencial
Estabilidad	Baja	Media	<u>Alta</u>
Verificabilidad	Baja	Media	<u>Alta</u>

Tabla 3.17: Requisito Funcional RF-16.

Identificador: RF-16			
Nombre	Guardar resultados de clasificación		
Prioridad	Baja	Media	<u>Alta</u>
Versión	1,0	Dependencias	
Descripción	El sistema debe permitir guardar los resultados de clasificación correspondientes a la evaluación del modelo.		
Necesidad	Opcional	Deseable	<u>Esencial</u>
Estabilidad	Baja	Media	<u>Alta</u>
Verificabilidad	Baja	Media	<u>Alta</u>



Tabla 3.18: Requisito Funcional RF-17.

Identificador: RF-17			
Nombre	Establecer la variable clasificada en estados		
Prioridad	Baja	Media	<u>Alta</u>
Versión	1,0	Dependencias	
Descripción	El sistema debe permitir transformar las variables clasificadas en estados para generar autómatas.		
Necesidad	Opcional	Deseable	<u>Esencial</u>
Estabilidad	Baja	Media	<u>Alta</u>
Verificabilidad	Baja	Media	<u>Alta</u>

Tabla 3.19: Requisito Funcional RF-18.

Identificador: RF-18			
Nombre	Guardar los autómatas		
Prioridad	Baja	Media	<u>Alta</u>
Versión	1,0	Dependencias	
Descripción	El sistema debe permitir guardar los subgrafos correspondientes a los autómatas que describen maniobras.		
Necesidad	Opcional	Deseable	<u>Esencial</u>
Estabilidad	Baja	Media	<u>Alta</u>
Verificabilidad	Baja	Media	<u>Alta</u>

3.1.3 Requisitos no funcionales

Los requisitos no funcionales de un proyecto software, en vez de especificar qué es lo que debe hacer dicho software, describen características de funcionamiento del mismo, es decir, especifican cómo se cumplen las funcionalidades. Los requisitos no funcionales del presente trabajo de fin de grado se describen a continuación (desde la Tabla 3.20 hasta la Tabla 3.26).



Tabla 3.20: Requisito Funcional RNF-01.

Identificador: RNF-01			
Nombre	Sistema Internacional		
Prioridad	Baja	Media	<u>Alta</u>
Versión	1.0	Dependencias	
Descripción	El sistema debe trabajar con las unidades del Sistema Internacional.		
Necesidad	Opcional	<u>Deseable</u>	Esencial
Estabilidad	Baja	Media	<u>Alta</u>
Verificabilidad	<u>Baja</u>	Media	Alta

Tabla 3.21: Requisito Funcional RNF-02.

Identificador: RNF-02			
Nombre	Lenguaje de programación de la red neuronal LSTM		
Prioridad	Baja	Media	<u>Alta</u>
Versión	1.0	Dependencias	
Descripción	La red neuronal recurrente LSTM debe estar programada en el lenguaje Python.		
Necesidad	Opcional	Deseable	<u>Esencial</u>
Estabilidad	Baja	Media	<u>Alta</u>
Verificabilidad	Baja	Media	<u>Alta</u>



Tabla 3.22: Requisito Funcional RNF-03.

Identificador: RNF-03			
Nombre	Etiquetado binario de los datos		
Prioridad	Baja	<u>Media</u>	Alta
Versión	1.0	Dependencias	
Descripción	El sistema utiliza la librería LabelBinarizer de Python para etiquetar los datos en formato binario antes de evaluar el modelo.		
Necesidad	Opcional	<u>Deseable</u>	Esencial
Estabilidad	Baja	<u>Media</u>	Alta
Verificabilidad	<u>Baja</u>	Media	Alta

Tabla 3.23: Requisito Funcional RNF-04.

Identificador: RNF-04			
Nombre	Compatibilidad con distintos sistemas operativos		
Prioridad	<u>Baja</u>	Media	Alta
Versión	1.0	Dependencias	
Descripción	El sistema debe poder ejecutarse en diferentes sistemas operativos: Windows, Mac, Linux.		
Necesidad	<u>Opcional</u>	Deseable	Esencial
Estabilidad	Baja	<u>Media</u>	Alta
Verificabilidad	Baja	<u>Media</u>	Alta



Tabla 3.24: Requisito Funcional RNF-05.

Identificador: RNF-05			
Nombre	Capacidad de respuesta		
Prioridad	Baja	<u>Media</u>	Alta
Versión	1.0	Dependencias	
Descripción	El sistema debe ser capaz de evaluar y entrenar el modelo en el menor tiempo posible.		
Necesidad	Opcional	<u>Deseable</u>	Esencial
Estabilidad	Baja	<u>Media</u>	Alta
Verificabilidad	<u>Baja</u>	Media	Alta

Tabla 3.25: Requisito Funcional RNF-06.

Identificador: RNF-06			
Nombre	Escalabilidad		
Prioridad	Baja	<u>Media</u>	Alta
Versión	1.0	Dependencias	
Descripción	El sistema debe ser capaz de manejar grandes volúmenes de datos sin disminuir su rendimiento.		
Necesidad	Opcional	<u>Deseable</u>	Esencial
Estabilidad	Baja	<u>Media</u>	Alta
Verificabilidad	Baja	<u>Media</u>	Alta

Tabla 3.26: Requisito Funcional RNF-07.

Identificador: RNF-07			
Nombre	Mantenibilidad		
Prioridad	<u>Baja</u>	Media	Alta
Versión	1.0	Dependencias	
Descripción	El sistema debe ser fácil de mantener y actualizar, con un código bien estructurado y comentado.		
Necesidad	<u>Opcional</u>	<u>Deseable</u>	Esencial
Estabilidad	Baja	<u>Media</u>	<u>Alta</u>
Verificabilidad	Baja	<u>Media</u>	<u>Alta</u>

3.2 Casos de uso

A continuación, se procederá a la definición de los casos de uso. Un caso de uso es una descripción detallada de los pasos que debe seguir un actor (es decir, uno de los usuarios identificados en el sistema que es responsable de realizar una acción) para completar una tarea específica en el sistema. Los casos de uso se pueden definir de dos maneras: de manera tabular, que consiste en proporcionar una descripción detallada utilizando una tabla, o de manera gráfica, mediante la creación de un diagrama de casos de uso que relacione estos casos con los usuarios del sistema. Ambas formas de representación se mostrarán en los siguientes apartados.

3.2.1 Descripción tabular de los casos de uso

Para la descripción tabular de los casos de uso se emplean tablas que siguen el siguiente formato (Tabla 3.27):

Tabla 3.27: Tabla modelo para casos de uso.

Identificador: CU-XX	
Título	
Actores	
Objetivo	
Precondiciones	
Postcondiciones	
Requisitos implicados	

A continuación, se explica qué significa cada una de las filas de la tabla modelo para la definición de casos de uso:

- **Identificador:** Código único para identificar de forma unívoca cada requisito, y para facilitar la trazabilidad. Los caracteres indicados en la tabla representan.
 - **CU.** Caso de uso.
 - **X.** Toma valores numéricos únicos para la identificación de cada caso de uso.
- **Título:** Nombre, conciso y descriptivo, para la identificación del caso de uso.
- **Actores:** Usuarios o sistemas identificados e involucrados en los casos de uso.
- **Objetivo:** Finalidad de la acción del caso de uso que se está definiendo.
- **Precondiciones:** Condiciones previas al caso de uso que han de cumplirse para su correcta realización.
- **Postcondiciones:** Estado del sistema tras la correcta realización del caso de uso que se está definiendo.
- **Requisitos implicados:** Enumera los requisitos funcionales relacionados con el caso de uso en cuestión

Se procede a especificar los casos de uso del presente trabajo de fin de grado en su descripción tabular (desde la Tabla 3.26 hasta la Tabla 3.39).

Tabla 3.28: Caso de uso CU-01.

Identificador: CU-01	
Título	Proporcionar conjunto de datos
Actores	Simulador
Objetivo	Proporcionar al investigador los datos recopilados por el simulador.
Precondiciones	<ul style="list-style-type: none">● Ejecutar el simulador con un conductor de pruebas● Comunicar los datos resultantes del simulador mediante el protocolo MQTT● Almacenar los datos en formato JSON y posteriormente convertirlos a formato xlsx
Postcondiciones	<ul style="list-style-type: none">● Datos preparados para su manipulación previa a la evaluación del modelo
Requisitos implicados	RF-01, RF-02

**Tabla 3.29: Caso de uso CU-02.**

Identificador: CU-02	
Título	Limpieza de datos
Actores	Usuario/Investigador
Objetivo	Limpiar el conjunto de datos eliminando variables y datos vacíos que no aportan valor al modelo.
Precondiciones	<ul style="list-style-type: none">• Obtener los datos provenientes del simulador
Postcondiciones	<ul style="list-style-type: none">• Datos limpios y sin valores nulos
Requisitos implicados	RF-01, RF-02, RF-11

Tabla 3.30: Caso de uso CU-03.

Identificador: CU-03	
Título	Crear objetivo
Actores	Usuario/Investigador
Objetivo	Crear una variable objetivo (Target) que se corresponda con el valor a clasificar y evaluar por el modelo.
Precondiciones	<ul style="list-style-type: none">• Disponer de los datos limpios y sin valores nulos• Distinguir las variables que se van a emplear para el estudio
Postcondiciones	<ul style="list-style-type: none">• Datos actualizados con una nueva variable objetivo
Requisitos implicados	RF-01, RF-02, RF-11, RF-12

**Tabla 3.31: Caso de uso CU-04.**

Identificador: CU-04	
Título	Actualizar datos
Actores	Usuario/Investigador
Objetivo	Actualizar los datos con los resultados de la primera fase de clasificación del modelo.
Precondiciones	<ul style="list-style-type: none">● Entrenamiento del modelo● Valoración de resultados y exposición de un nuevo objetivo a clasificar por el modelo
Postcondiciones	<ul style="list-style-type: none">● Datos actualizados
Requisitos implicados	RF-01, RF-02, RF-05, RF-08, RF-12, RF-16

Tabla 3.32: Caso de uso CU-05.

Identificador: CU-05	
Título	Cargar conjunto de datos
Actores	Simulador
Objetivo	Cargar el conjunto de los datos en el modelo para su transformación y evaluación.
Precondiciones	<ul style="list-style-type: none">● Disponer de los datos proporcionados por el simulador de conducción
Postcondiciones	<ul style="list-style-type: none">● Conjunto de datos preparados para comenzar su estudio
Requisitos implicados	RF-01, RF-02

**Tabla 3.33: Caso de uso CU-06**

Identificador: CU-06	
Título	Ajustar hiper parámetros
Actores	Usuario/Investigador
Objetivo	Ajustar los valores de los hiper parámetros de la red neuronal para su entrenamiento.
Precondiciones	<ul style="list-style-type: none">● Disponer de los datos actualizados, preprocesados y cargados● Disponer del código correspondiente a la red neuronal para entrenar
Postcondiciones	<ul style="list-style-type: none">● Modelo preparado para su entrenamiento y posterior evaluación
Requisitos implicados	RF-03, RF-04, RF-10, Rf-14

Tabla 3.34: Caso de uso CU-07.

Identificador: CU-07	
Título	Fijar semilla
Actores	Usuario/Investigador
Objetivo	Ajustar el valor de la semilla para todos los entrenamientos del modelo.
Precondiciones	<ul style="list-style-type: none">● Disponer de los datos actualizados, preprocesados y cargados● Disponer del código correspondiente a la red neuronal para entrenar
Postcondiciones	<ul style="list-style-type: none">● Semilla fijada para los entrenamientos y evaluación del modelo
Requisitos implicados	RF-03, RF-04, RF-14



Tabla 3.35: Caso de uso CU-08.

Identificador: CU-08	
Título	Estratificación de datos
Actores	Usuario/Investigador
Objetivo	Separar los datos de entrenamiento en un nuevo conjunto de validación.
Precondiciones	<ul style="list-style-type: none">• Disponer de los datos actualizados, preprocesados y cargados• Disponer del código correspondiente a la red neuronal para entrenar
Postcondiciones	<ul style="list-style-type: none">• Nuevo conjunto de validación.
Requisitos implicados	RF-01, RF-02, RF-13

Tabla 3.36: Caso de uso CU-09.

Identificador: CU-09	
Título	Crear checkpoint
Actores	Usuario/Investigador
Objetivo	Guardar los parámetros de los mejores resultados del modelo.
Precondiciones	<ul style="list-style-type: none">• Disponer de los datos actualizados, preprocesados y cargados• Disponer del código correspondiente a la red neuronal para entrenar• Entrenar el modelo con la red neuronal
Postcondiciones	<ul style="list-style-type: none">• Modelo entrenado• Guardar los parámetros y resultados del mejor modelo
Requisitos implicados	RF-03, RF-04, RF-09

Tabla 3.37: Caso de uso CU-10.

Identificador: CU-10	
Título	Balanceo de clases
Actores	Usuario/Investigador
Objetivo	Utilizar pesos para balancear las respectivas clases de cara a su entrenamiento y evaluación.
Precondiciones	<ul style="list-style-type: none"> ● Disponer de los datos actualizados, preprocesados y cargados ● Disponer del código correspondiente a la red neuronal para entrenar ● Conocer los pesos iniciales de las clases
Postcondiciones	<ul style="list-style-type: none"> ● Modelo entrenado ● Establecer los pesos de las clases para obtener el balance óptimo
Requisitos implicados	RF-03, RF-04, RF-10

Tabla 3.38: Caso de uso CU-11.

Identificador: CU-11	
Título	Entrenar la red
Actores	Usuario/Investigador
Objetivo	Realizar el entrenamiento de la red neuronal recurrente LSTM y la posterior evaluación del modelo.
Precondiciones	<ul style="list-style-type: none"> ● Disponer de los datos actualizados, preprocesados y cargados ● Disponer del código correspondiente a la red neuronal para entrenar ● Disponer del ajuste de los hiper parámetros del modelo
Postcondiciones	<ul style="list-style-type: none"> ● Modelo entrenado ● Resultados de clasificación y evaluación
Requisitos implicados	RF-01, RF-02, RF-03, RF-04, RF-09, RF-10, RF-11, RF-13, RF-14, RF-15, RF-16

Tabla 3.39: Caso de uso CU-12.

Identificador: CU-12	
Título	Visualizar gráficas
Actores	Usuario/Investigador
Objetivo	Se podrá visualizar los resultados y estadísticas mediante el uso de gráficas.
Precondiciones	<ul style="list-style-type: none"> ● Disponer de los datos actualizados, preprocesados y cargados ● Disponer del código correspondiente a la red neuronal para entrenar
Postcondiciones	<ul style="list-style-type: none"> ● Modelo entrenado ● Visualización de resultados y estadísticas
Requisitos implicados	RF-01, RF-05, RF-06, R-15

Tabla 3.40: Caso de uso CU-13.

Identificador: CU-13	
Título	Visualizar mapa de calor
Actores	Usuario/Investigador
Objetivo	Se podrá visualizar los resultados y estadísticas mediante el uso de mapas de calor o matrices de confusión.
Precondiciones	<ul style="list-style-type: none"> ● Disponer de los datos actualizados, preprocesados y cargados ● Disponer del código correspondiente a la red neuronal para entrenar
Postcondiciones	<ul style="list-style-type: none"> ● Modelo entrenado ● Visualización de resultados, mapas de calor y matriz de confusión
Requisitos implicados	RF-01, RF-07, RF-08, RF-15

Tabla 3.41: Caso de uso CU-14.

Identificador: CU-14	
Título	Guardar resultados
Actores	Usuario/Investigador
Objetivo	Guardar los resultados de clasificación, así como las gráficas correspondientes.
Precondiciones	<ul style="list-style-type: none"> ● Disponer de los datos actualizados, preprocesados y cargados ● Disponer del código correspondiente a la red neuronal para entrenar ● Entrenar el modelo
Postcondiciones	<ul style="list-style-type: none"> ● Evaluación de los resultados ● Visualización de gráficas ● Visualización de matriz de confusión y mapa de calor
Requisitos implicados	RF-15, RF-16

Tabla 3.42: Caso de uso CU-15.

Identificador: CU-15	
Título	Transformar datos en estados
Actores	Usuario/Investigador
Objetivo	Transformar los datos correspondientes a los resultados de clasificación en estados para generar autómatas.
Precondiciones	<ul style="list-style-type: none"> ● Disponer de los datos actualizados, preprocesados, cargados y entrenados ● Disponer de los datos actualizados ● Entrenar el modelo
Postcondiciones	<ul style="list-style-type: none"> ● valores modificados en forma de estados de transición ● Generación de autómatas
Requisitos implicados	RF-17

Tabla 3.43: Caso de uso CU-16.

Identificador: CU-16	
Título	Guardar subgrafos
Actores	Usuario/Investigador
Objetivo	Guardar los subgrafos correspondientes a la generación de autómatas que describen maniobras de alto nivel.
Precondiciones	<ul style="list-style-type: none">● Disponer de los datos actualizados, preprocesados, cargados y entrenados● Disponer de los grafos completos● Definir el conjunto de transiciones de estado
Postcondiciones	<ul style="list-style-type: none">● Visualización de subgrafos● Almacenamiento de subgrafos
Requisitos implicados	RF-18

3.2.2 Matriz de trazabilidad

En este apartado se presentará una descripción detallada de los aspectos relacionados con la escalabilidad del proyecto, que incluye la capacidad del sistema para crecer y adaptarse a los cambios en las necesidades y requisitos del usuario.

Tabla 3.44: Matriz de trazabilidad.

		REQUISITOS FUNCIONALES																	
		RF-01	RF-02	R-F-03	R-F-04	R-F-05	R-F-06	R-F-07	R-F-08	R-F-09	R-F-10	RF-11	RF-12	RF-13	RF-14	RF-15	RF-16	RF-17	RF-18
CASOS DE USO	CU-01																		
	CU-02																		
	CU-03																		
	CU-04																		
	CU-05																		
	CU-06																		
	CU-07																		
	CU-08																		
	CU-09																		
	CU-10																		
	CU-11																		
	CU-12																		
	CU-13																		
	CU-14																		
	CU-15																		
	CU-16																		

3.2.3 Descripción gráfica de los casos de uso

En este punto se presentan los casos de uso en su descripción gráfica. El objetivo de esta representación es observar las relaciones entre los casos de uso y los diferentes actores. Para un nivel mayor de detalle, se tiene la descripción tabular de los casos de uso del apartado anterior. Se muestra la descripción gráfica de los casos de uso en la Figura 3.1.

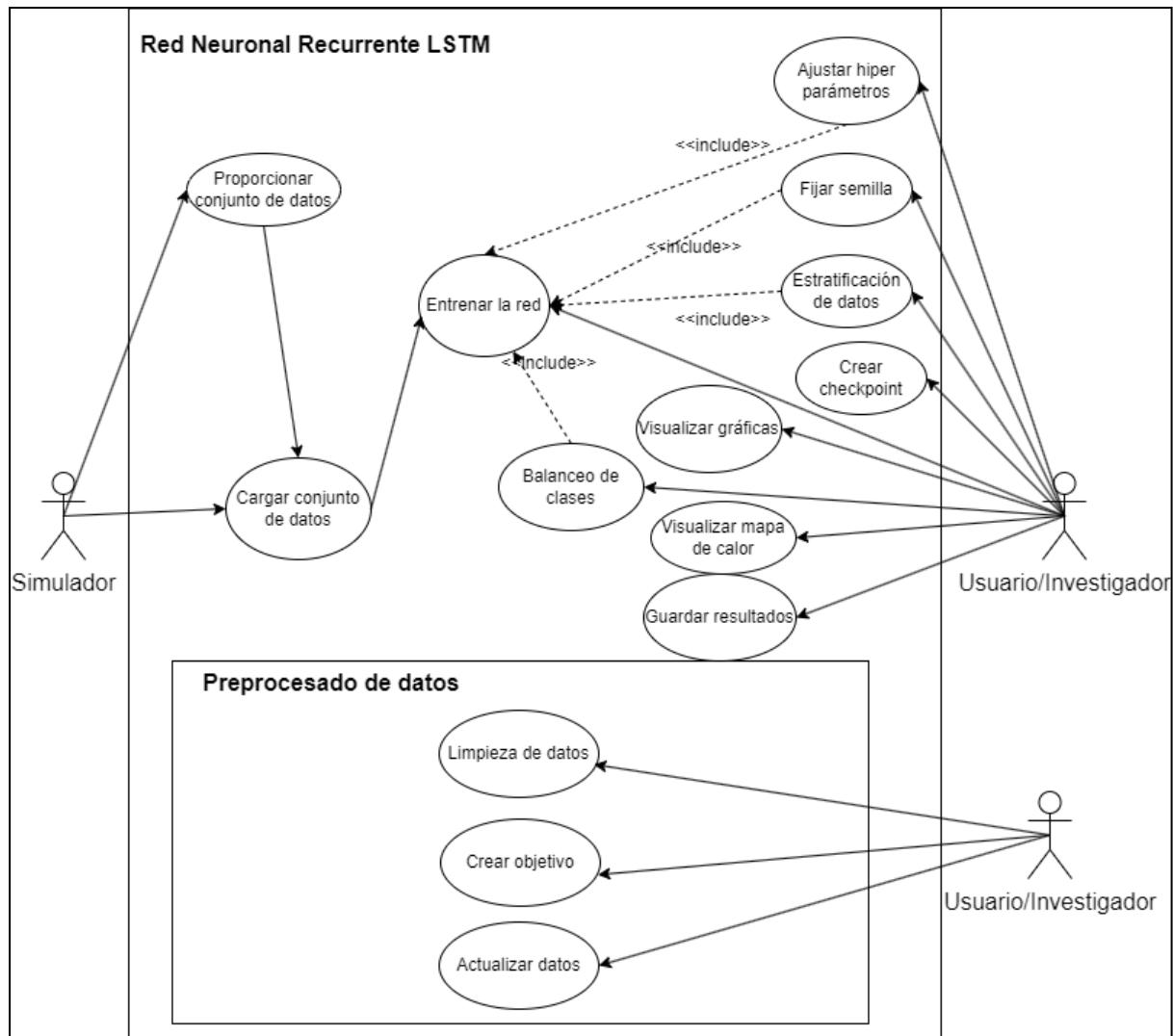


Figura 3.1: Diagrama de casos de uso.

4. Arquitectura, diseño e implementación del sistema

En este apartado se presenta la descripción de la arquitectura del sistema y el entorno tecnológico que la soporta, incluyendo sus componentes, organización, funciones y cómo interactúan entre sí.

Para ello, se irá desde un nivel mayor de abstracción, ubicando el sistema dentro de la arquitectura general del ADAS, a un nivel más detallado con menor abstracción.

Además se presenta la descripción de la arquitectura de la red neuronal recurrente Long-Short Term Memory (LSTM).

4.1 Arquitectura general del sistema

Es relevante destacar que el Sistema de Asistencia Avanzada a la Conducción se ha construido sobre un proyecto anterior, lo que implicó la realización previa de las tareas de diseño e integración de la arquitectura del sistema.

En la actualidad, el entorno operativo del sistema de simulación y asistencia al conductor está estructurado de la siguiente manera (Figura 4.1):

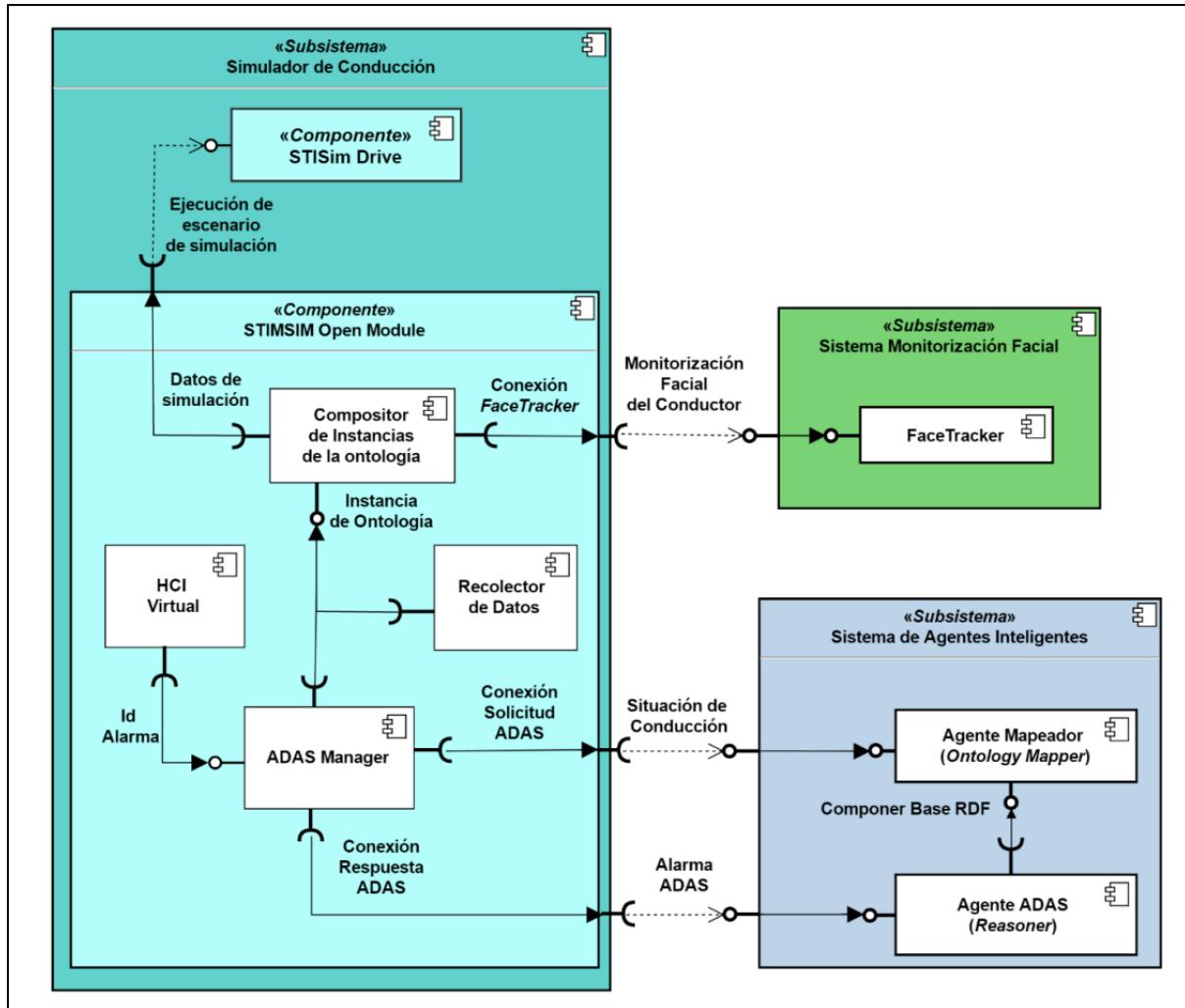


Figura 4.1: Arquitectura del entorno de simulación. [9]

Como se puede observar en la Figura 4.1, el entorno de simulación está dividido en tres grandes subsistemas. Estos subsistemas se definen de la siguiente forma:

- **Subsistema Simulador de Conducción:** es el subsistema encargado de la simulación de la conducción. Está compuesto por dos componentes principales: el STISim Drive, que es el software encargado de llevar a cabo la simulación de conducción, y el STIMSIM Open Module, que es un módulo que permite agregar otros componentes al simulador a través de código en lenguaje de programación Visual Basic. De esta manera, se puede mejorar y personalizar la simulación de conducción según las necesidades y requerimientos del proyecto.
- **Subsistema de Monitorización Facial:** este subsistema incluye los componentes relacionados con la supervisión del estado del conductor mediante la monitorización del conductor a través de su mirada en formato de imágenes.
- **Subsistema de Agentes Inteligentes:** este subsistema se encarga de procesar y analizar los datos que provienen del simulador de conducción, evaluando las diferentes situaciones de manejo que pueden presentarse. Su principal función es la de determinar si existe una situación de peligro y, en caso afirmativo, enviar una alarma al simulador para alertar al conductor. Esta arquitectura se basa en un sistema

multi-agente, compuesto por dos componentes principales: un agente que se encarga de mapear la situación de conducción en relación a la ontología, y otro agente que se dedica al razonamiento.

4.2 Red long-short term memory (LSTM)

Además de comprender la arquitectura general del sistema de simulación, se va a definir la arquitectura de la red Long-Short Term Memory (LSTM), ya que es el principal objetivo de desarrollo de este trabajo. A continuación se muestra un diagrama de la implementación de dicha arquitectura (Figura 4.2).

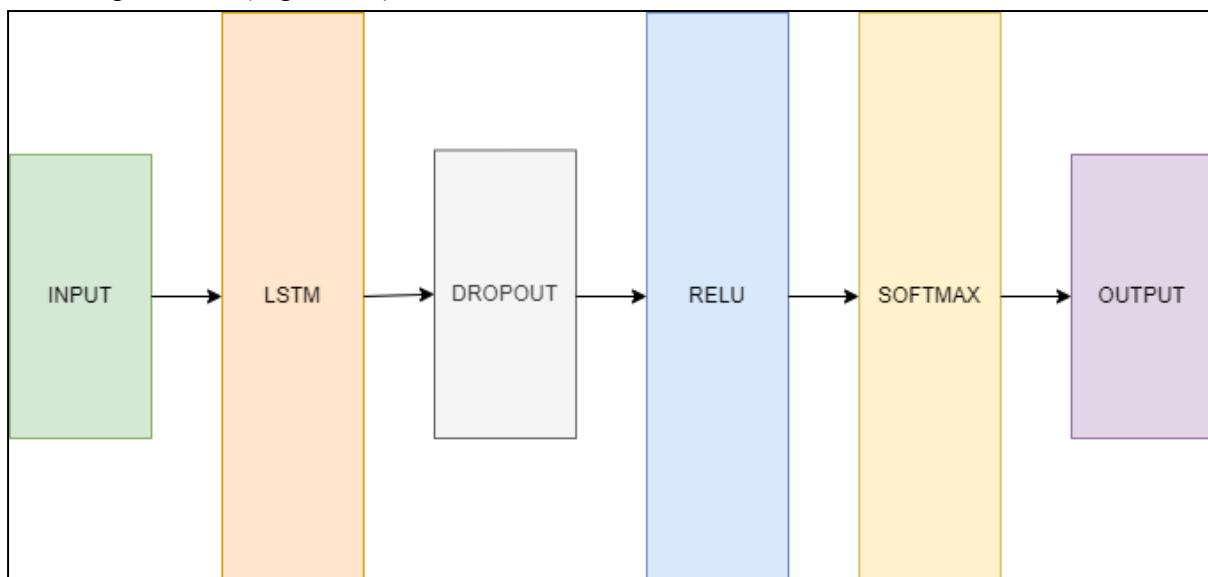


Figura 4.2: Arquitectura de la red Long-Short Term Memory (LSTM).

Como se puede observar en la Figura 4.2, la red neuronal recurrente está dividida en tres capas principales con una función de dropout intermedia. El conjunto de datos que forman el modelo se entrena siguiendo esta estructura, que en función de los hiperparámetros elegidos varían el objetivo del entrenamiento del modelo. De esta manera, las capas de la red se definen de la siguiente forma:

- **Capa Long-Short Term Memory (LSTM):** Esta capa es especialmente útil para procesar datos secuenciales, ya que es capaz de recordar información relevante de las entradas anteriores y utilizarla para tomar decisiones en la entrada actual. La capa LSTM tiene una serie de parámetros que se pueden ajustar para optimizar el rendimiento del modelo, incluyendo el número de neuronas ocultas y la forma de la entrada (input shape).
- **Capa Dropout:** Se utiliza una capa de dropout como una técnica de regularización que ayuda a reducir el sobreajuste o overfitting. En este proyecto se emplea un valor de 0.5, lo que significa que el 50% de las neuronas de la capa anterior serán seleccionadas al azar y eliminadas temporalmente de la red, lo que obliga a las otras neuronas a aprender de manera más independiente y a no depender demasiado de ninguna neurona en particular.

- **Capa Relu:** La función relu es una función no lineal que se define como $f(x) = \max(0, x)$ [31], lo que significa que si el valor de entrada es mayor que cero, la salida será el mismo valor de entrada y si el valor de entrada es menor o igual a cero, la salida será cero. La función relu se utiliza porque puede ayudar a superar el problema de la desaparición del gradiente y proporcionar una mejor representación de los datos de entrada.
- **Capa Softmax:** La capa softmax es una función de activación que se utiliza para el procesamiento de clasificación multiclase. La capa tiene un número de neuronas igual al número de clases en el problema de clasificación y transforma las salidas de cada una de estas neuronas en una distribución de probabilidad, es decir, que la suma de todas las salidas será igual a 1.

4.3. Implementación del sistema

Aquí se describen detalladamente los procesos de recogida de datos, procesado de datos y el diseño y la implementación de la red neuronal long-short term memory (LSTM), siguiendo el esquema del diseño de la arquitectura de la red (ver Figura 4.2: Arquitectura de la red LSTM).

4.3.1 Recopilación de datos simulador ADAS

Para la primera parte de este proyecto, es necesario utilizar el simulador de conducción ADAS para la extracción de datos de diferentes pruebas utilizando una selección de conductores de pruebas.

A continuación, se explicará el funcionamiento de recogida de datos desarrollado previamente por el grupo CAOS. Además del formato recibido y empleado para el desarrollo de este proyecto.

4.3.1.1 Recopilación de información del entorno

Cuando se recolecta información sobre el entorno en el simulador de conducción STISIM Drive, es importante tener en cuenta que se puede acceder a cualquier dato de la simulación. Sin embargo, es necesario procesar los datos para ajustarlos a los que un vehículo real podría recibir.

Para detectar vehículos en el entorno, el simulador utiliza un módulo que simula un sensor LiDAR [33], el cual emite rayos láser para detectar objetos alrededor del vehículo y medir la distancia entre ellos midiendo el tiempo de retraso entre la emisión y el reflejo del rayo en el objeto. A partir de la distancia entre el vehículo y otros objetos, es posible obtener más información sobre el objeto, como si está detenido o en movimiento, su velocidad relativa al vehículo y su trayectoria, siempre que se tenga una medición continua a lo largo del tiempo (ver Figura 4.3).

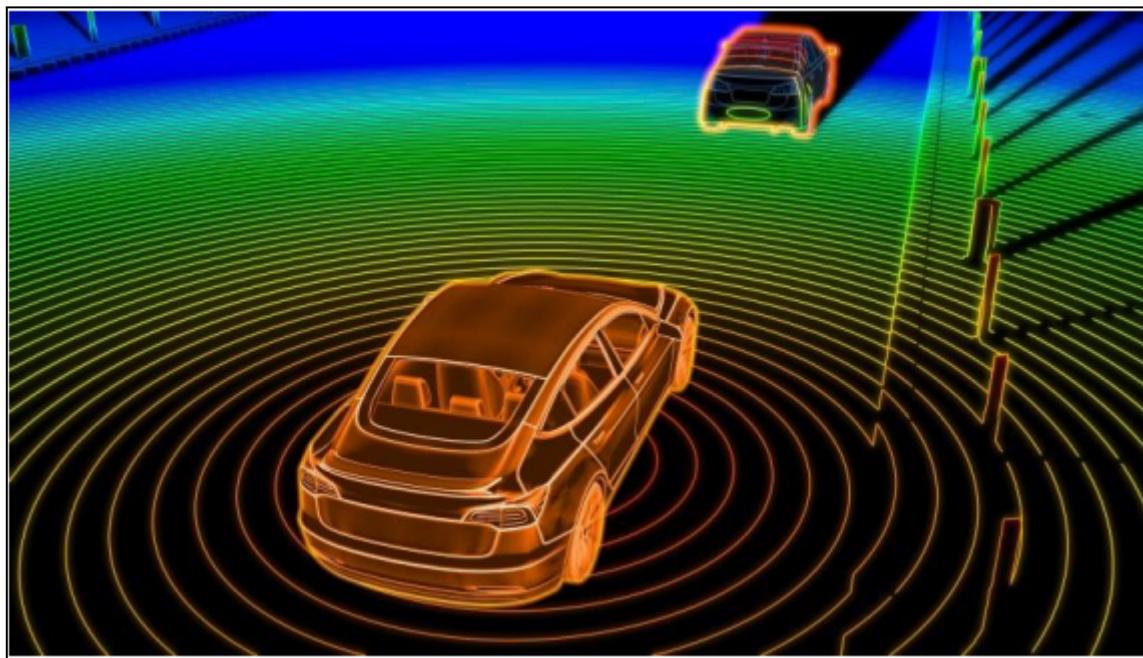


Figura 4.3: Simulación del funcionamiento de un sensor LiDAR. [33]

Utilizando esta tecnología, el simulador puede recibir información de vehículos que se encuentren en sus inmediaciones. E incluso se pueden reconocer peatones gracias a que se simuló una cámara frontal, que sí sería capaz de distinguir cuáles de los objetos que tiene delante (ver Figura 4.4).

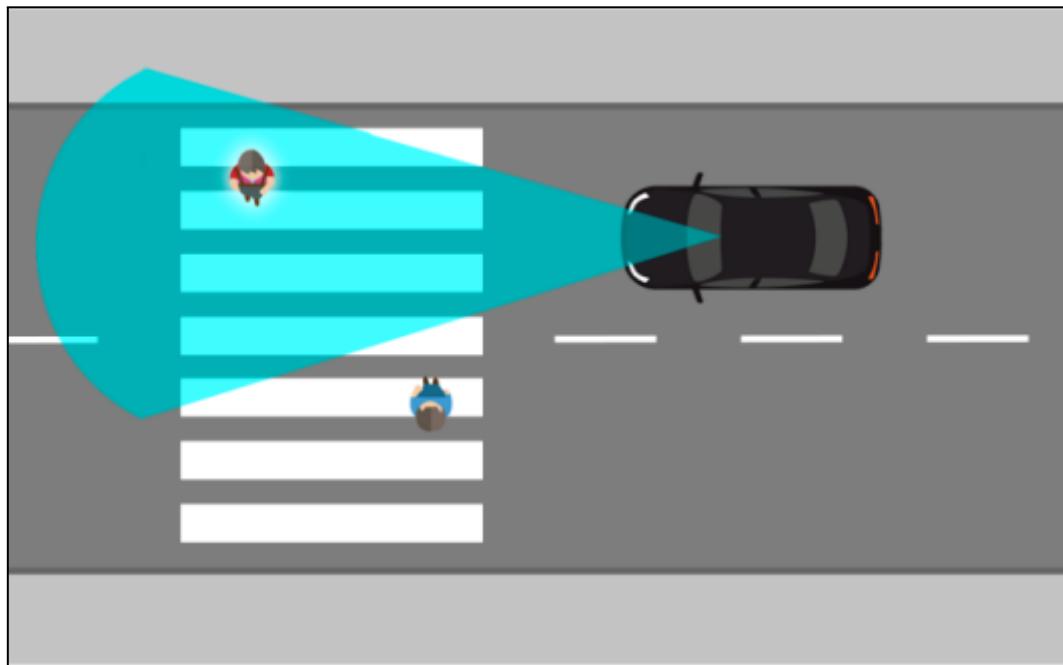


Figura 4.4: Simulación del funcionamiento de la cámara frontal. [33]

Concretamente, la implementación de esta cámara depende de tres valores: el punto de visión en el que está localizada la cámara (POV), del ángulo de visión (α) y el alcance (dist).

Mediante estos tres valores se es capaz de definir el área triangular sobre el que el vehículo sería capaz de reconocer a los peatones. Además, la unión del sensor LiDAR con una cámara frontal y una cámara trasera permite conocer todo el entorno que rodea al vehículo (ver Figura 4.5).

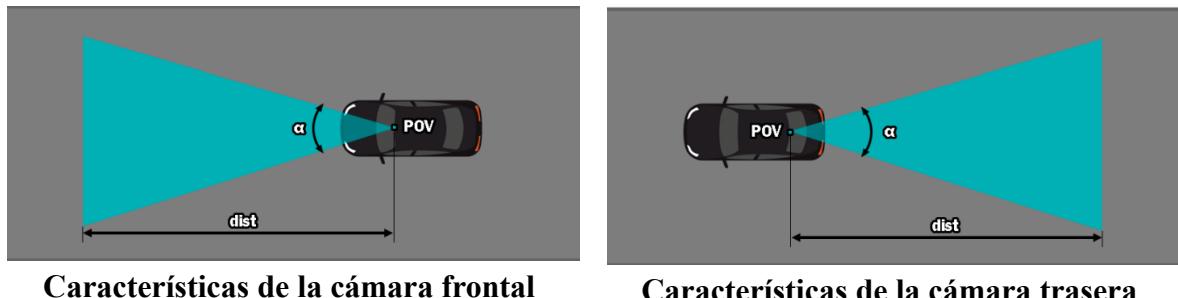


Figura 4.5: Simulación cámara frontal y trasera. [33]

Por último, comentar que el simulador proporciona información acerca de los periféricos que forman parte de la experiencia de conducción de un vehículo. Entre estos, se encuentran los siguientes:

- Giro del volante, velocidad, marcha y valores de activación de los pedales del vehículo propio, como valores numéricos.
- Ángulo, velocidad y trayectoria de los vehículos del contexto.

4.3.1.2 Envío y formato de los datos

Como ya se ha mencionado en el punto [1.2 Descripción del Problema](#) de este documento, el simulador recoge los datos referentes a los valores de los periféricos, a la situación del entorno del vehículo y la situación del conductor gracias a sus sensores. Estos datos se recopilan en formato JSON (ver Figura 1.3: Ejemplo datos en formato JSON recibidos del simulador) y se comunican mediante un protocolo MQTT³.

Una vez recogidos los datos del simulador, estos son transformados a un fichero xlsx. En este proyecto se han recogido datos de cinco conductores distintos realizando cinco maniobras específicas:

- **3Step-Turnings:** hace referencia a un proceso de tres pasos para completar una maniobra de giro.
- **Overtaking:** hace referencia a un proceso para completar una maniobra de adelantamiento a otro vehículo.

³ El protocolo MQTT (Message Queuing Telemetry Transport) es un protocolo de mensajería basado en MQTT diseñado para la transferencia de datos en redes de dispositivos IoT (Internet de las cosas) y entornos M2M (Machine to Machine) con alta frecuencia y baja latencia de transmisión. Este protocolo permite la comunicación bidireccional y en tiempo real entre dispositivos y servidores, lo que facilita el desarrollo de aplicaciones IoT en diferentes sectores

- **Stopping:** hace referencia a un proceso para completar una maniobra de frenado del vehículo.
- **Turnings:** hace referencia a un proceso para completar una maniobra de un único giro.
- **U-Turnings:** hace referencia a un proceso para completar una maniobra de cambio de sentido.

Cada fichero contiene un conjunto de datos que representan los siguientes valores (todos se representan en el formato internacional):

- **Elapsed Time:** representa el momento del tiempo en el que se encuentra el vehículo. Es un valor ascendente con el que se evaluará la serie temporal.
- **Long Dist:** representa la distancia longitudinal que ha recorrido el vehículo en un instante determinado.
- **Lat Pos:** representa la distancia lateral entre el vehículo y otro objeto que se encuentre a su lado.
- **Steering wheel angle:** representa el ángulo de rotación del volante del vehículo.
- **Throttle input:** representa la cantidad de aceleración aplicada por el conductor al vehículo mediante el pedal del acelerador.
- **Brake pedal force:** representa la fuerza que se aplica en el pedal de freno del vehículo.
- **Gas pedal:** representa la potencia del pedal de aceleración del vehículo.
- **Brake pedal:** representa la potencia del pedal de freno del vehículo.
- **Clutch pedal:** representa la potencia del pedal de embrague del vehículo.
- **Left turn:** representa con un valor binario si el vehículo se encuentra girando a la izquierda.
- **Right turn:** representa con un valor binario si el vehículo se encuentra girando a la derecha.
- **Gear:** representa la marcha en la que se encuentra el vehículo.
- **Speed:** representa la velocidad que adquiere el vehículo.
- **RPM:** representa las revoluciones por minuto que alcanza el vehículo.
- **Hand wheel torque:** representa la cantidad de fuerza o momento que se aplica al volante del vehículo para realizar una maniobra de giro o dirección.
- **Maneuver marker flag:** representa la señalización o bandera que se utiliza para indicar que se ha detectado un cierto tipo de maniobra o situación específica.
- **Accidents:** representación del número de accidentes producidos en la simulación.
- **Collisions:** representación del número de colisiones producidas en la simulación.
- **Peds Hit:** representación del número de peatones atropellados durante la simulación.
- **Speeding Tics:** representación del número de veces que el vehículo ha superado el límite de velocidad durante la simulación.
- **Red Lgt Tics:** representación del número de veces que el vehículo ha causado una infracción de tráfico por no respetar la señal de semáforo en rojo durante la simulación.
- **Speed Exceed:** representación del máximo de velocidad que ha excedido el vehículo.

- **Stop Sign Ticks:** representación del número de veces que el vehículo ha causado una infracción de tráfico por no respetar la señal de alto (stop) durante la simulación.

4.3.2 Procesado de datos

Una vez que se han recopilado los datos del simulador, se realiza un preprocesado de los datos utilizando el lenguaje de programación de *Python*. Se escoge este lenguaje de programación por encima de otros lenguajes debido a su facilidad y rapidez en la implementación, además de que ofrece numerosas librerías de aprendizaje automático.

Por otro lado, se aborda la definición de una variable objetivo a partir del conjunto de datos. Se examinará la importancia de una definición clara de la variable objetivo y se explorarán técnicas para seleccionar la variable más adecuada para el análisis.

4.3.2.1 Limpieza y transformación de los datos

En primer lugar, se utiliza un código que se encarga de procesar datos de un conjunto de archivos en formato Excel, eliminando las columnas: Maneuver marker flag, Accidents, Collisions, Peds Hit, Speeding Tics, Red Lgt Tics, Speed Exceed y Stop Sign Ticks, que se consideran irrelevantes dentro de este estudio. El código utiliza las librerías de Pandas y NumPy para trabajar con los datos, y luego, guarda el dataframe modificado en un archivo CSV.

Una vez cargados los nuevos archivos CSV, se asegura que ninguna columna contiene valores nulos. Además, como se trata de una clasificación con redes LSTM, se separa la columna Elapsed Time, la cual, debe ser tratada como una fecha, y se utiliza un diccionario para especificar que esta columna será asignada a una nueva columna llamada 'dt' encargada de representar la serie temporal. Y se ordena el dataframe para establecer las primeras columnas según la importancia dentro del estudio.

De este modo, se tiene un conjunto de datos de 15 columnas (con la variable Target) que representa los siguientes valores:

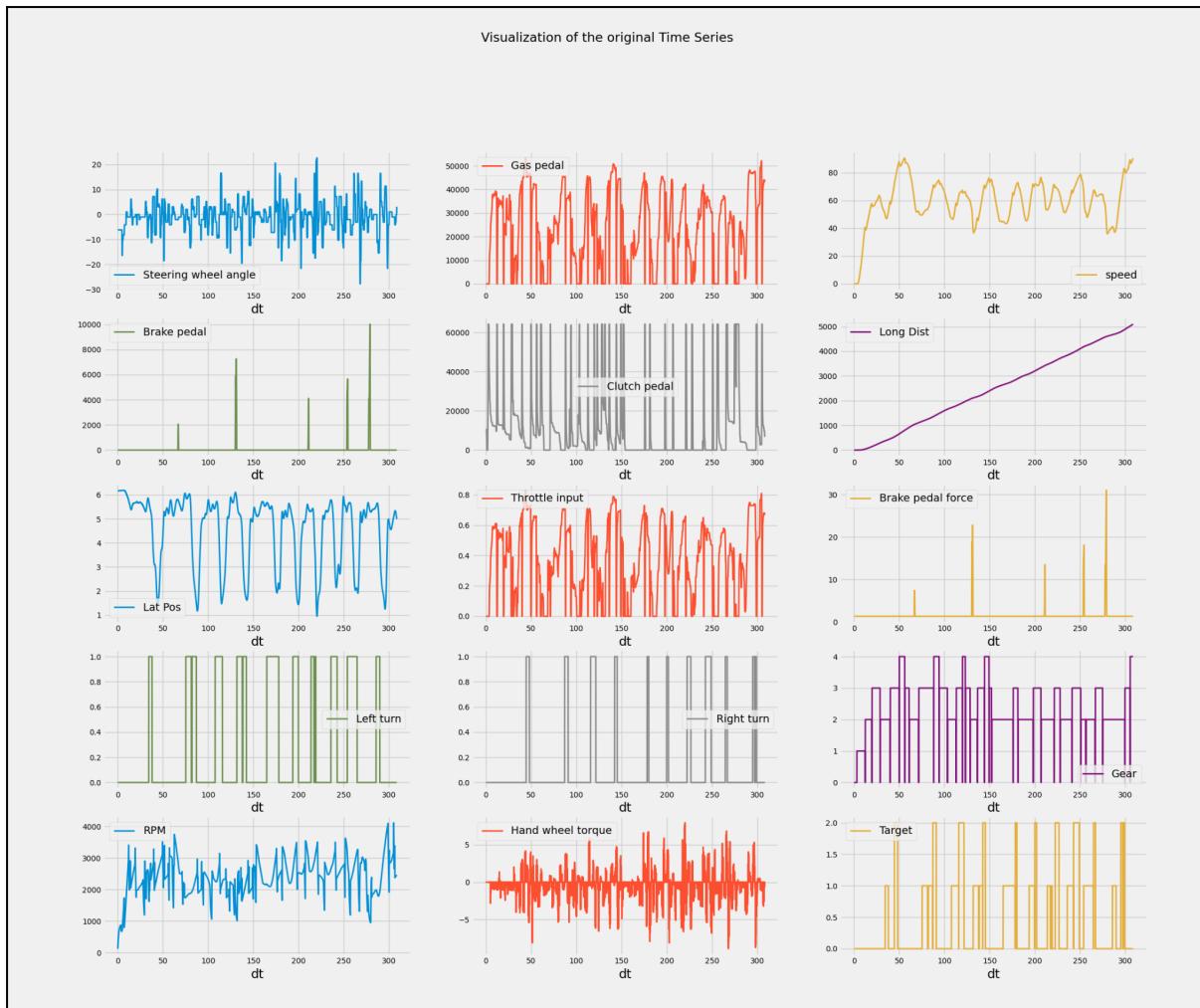


Figura 4.6: Gráficas de series temporales de cada variable.

También se incluye una representación gráfica del mapa de calor de cada una de las variables con la siguiente matriz de correlación:

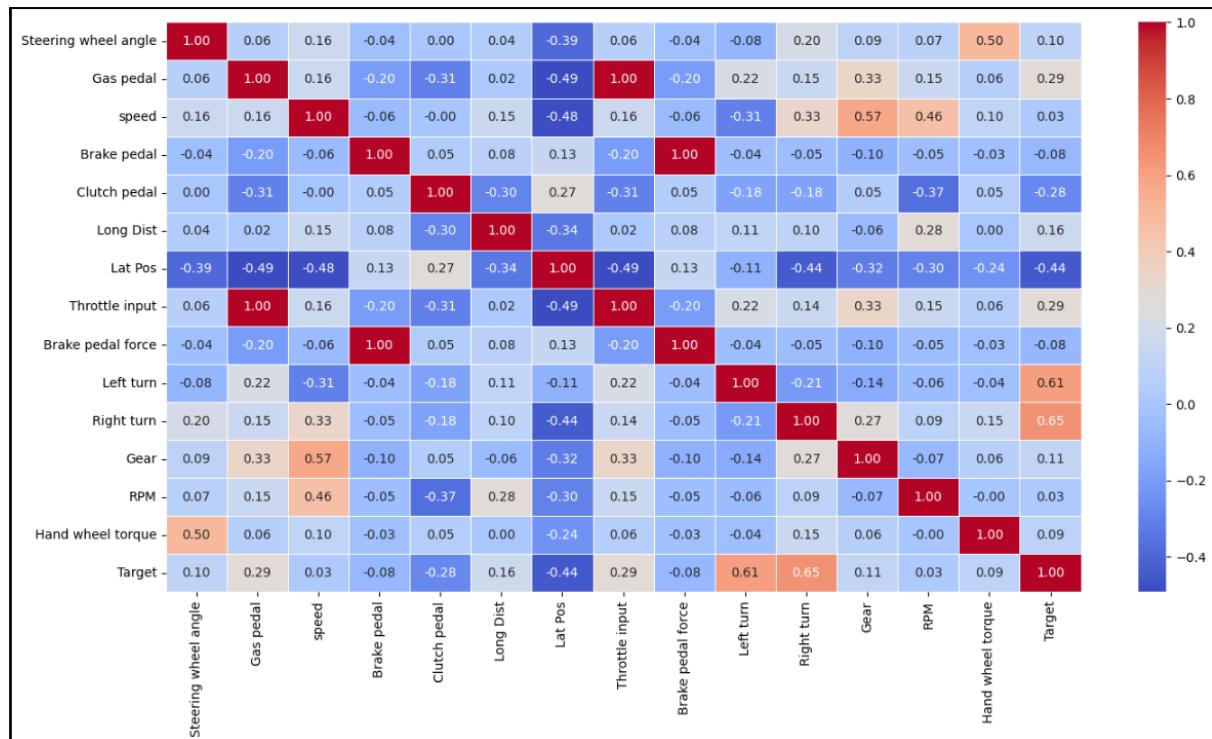


Figura 4.7: Mapa de calor de la matriz de correlación.

Por último, se normalizan los datos en un rango entre 0 y 1 utilizando la librería *scikit-learn* de *Python*, mediante la clase MinMaxScaler del módulo *preprocessing*. Esto se realiza con el fin de escalar los valores de diferentes variables a un rango común, de manera que las variables tengan una escala comparable entre sí y se eliminen las diferencias de magnitud que puedan existir entre ellas. Y utilizando la clase LabelBinarizer del mismo módulo, se convierten las etiquetas de clasificación en forma binaria para que puedan ser utilizadas en el entrenamiento y prueba del modelo de aprendizaje automático.

4.3.2.2 Creación de la variable objetivo (target)

Para el desarrollo de este trabajo es necesario establecer una variable objetivo, llamada "Target". Esta variable se basa en la combinación de los valores binarios de las columnas "Left Turn" y "Right Turn". Para crearla se utiliza la función "np.select" de *NumPy*, y los posibles valores que pueden tener son los siguientes:

- #0: Continuar recto (0,0).
- #1: Girar a la izquierda (1,0).
- #2: Girar a la derecha (0,1).

Tras establecer la variable objetivo se comienza el entrenamiento del modelo. Una vez finalizado este proceso se obtiene una columna que representa la clasificación del "Target" (en función del tiempo en segundos de la simulación), definido anteriormente. Esta nueva columna se añade al conjunto de datos para comenzar con la segunda fase de clasificación.

Antes de comenzar la segunda fase de clasificación se decidieron las nuevas tareas a clasificar, representadas en un nivel superior (*ver figura 1.5 del Capítulo 1*). Para ello se expusieron las siguientes opciones:

- **Tarea Aumentar/Reducir Velocidad:** Sabiendo que el conductor continúa recto (empleando la anterior clasificación), presiona el acelerador/freno y el cambio de marcha. Se puede clasificar si un conductor está acelerando o frenando en una recta.
- **Tarea Detener Vehículo:** Caso en el que un conductor presiona el pedal del freno hasta que su velocidad disminuye a 0 y la marcha se reduce. (En este caso no interviene la clasificación previamente realizada).
- **Tarea Arrancar Vehículo:** Caso en el que un conductor presiona el pedal del acelerador hasta que su velocidad aumente de 0 y la marcha se aumente. (En este caso no interviene la clasificación previamente realizada).
- **Tarea Marcha Atrás:** Caso en el que un conductor presiona el acelerador con una marcha específica sin aumentar ni reducir (marcha atrás). (En este caso no interviene la clasificación previamente realizada).
- **Tarea Cambiar de Carril (izq/dch):** En este caso un conductor gira el volante hacia uno de los lados durante un corto periodo de tiempo y vuelve a girarlo en sentido contrario para colocarse en el carril. En este caso se podría utilizar la clasificación realizada previamente. En este caso no se dispone de datos acerca de la vista del conductor.

Después de valorar todas las opciones se ha decidido definir una nueva variable objetivo llamada “New_Target”, que pretende clasificar una tarea de aumentar/reducir la velocidad del vehículo. Esta variable se basa en los resultados de la clasificación realizada en la primera fase, el valor de la columna “Gear” y la columna “Gas Pedal” (ambas comparan el valor en el tiempo actual y en el posterior). Los posibles valores que pueden tener son los siguientes:

- **#0:** Continuar (sin aumentar o reducir la velocidad)
- **#1:** Acelerar recto (aumentar la velocidad cuando clasificación=0 y marcha >=marcha-1)
- **#2:** Reducir recto (disminuir la velocidad cuando clasificación=0 y marcha <=marcha-1)
- **#3:** Acelerar derecha(aumentar la velocidad cuando clasificación=1 y marcha >=marcha-1)
- **#4:** Reducir derecha(disminuir la velocidad cuando clasificación=1 y marcha <=marcha-1)
- **#5:** Acelerar izquierda (aumentar la velocidad cuando clasificación=2 y marcha >=marcha-1)
- **#6:** Reducir izquierda (disminuir la velocidad cuando clasificación=2 y marcha <=marcha-1)

4.3.3 Diseño e implementación de la red LSTM

En este apartado se describen los procesos de diseño e implementación de la red neuronal recurrente Long-Short Term Memory (LSTM), desarrollada con el lenguaje de programación

Python, elegido por su compatibilidad con los frameworks de deep learning Tensorflow y Keras.

Del mismo modo, se van a describir los hiper parámetros elegidos, el entrenamiento del modelo de clasificación y la evaluación de sus resultados.

4.3.3.1 Ajuste de hiper parámetros

El ajuste de hiper parámetros es una etapa crítica en el desarrollo de modelos de redes neuronales, ya que la elección adecuada de los valores de los parámetros puede mejorar significativamente la precisión y el rendimiento del modelo. Los hiper parámetros son valores que se establecen antes del entrenamiento del modelo y que afectan directamente al proceso de aprendizaje.

En este modelo se emplean diversas técnicas como la validación cruzada, estratificación de datos, el uso de un checkpoint, balanceo de clases (ajuste de pesos), control de sobreajuste (Early Stopping) y fijación de la semilla. A continuación, se realiza una descripción de cada una de las técnicas:

- **Validación cruzada:** el uso de la validación cruzada tiene como objetivo obtener una estimación más precisa del rendimiento del modelo en datos no vistos, reduciendo así la posibilidad de sobreajuste (overfitting) y aumentando la confianza en las métricas de evaluación obtenidas.
- **Estratificación de los datos:** se refiere al proceso de dividir los datos en conjuntos de entrenamiento y validación de tal manera que las proporciones de las clases se mantengan en ambas particiones. La finalidad de la estratificación es garantizar que el modelo se entrene y evalúe de manera equilibrada en todas las clases y se evite el sobreajuste en una clase particular.
- **Checkpoint:** El uso de un Checkpoint tiene como objetivo prevenir el sobreajuste del modelo y garantizar que se guarden los pesos que producen el mejor desempeño en el conjunto de datos de prueba.
- **Balanceo de clases:** esta técnica es esencial para este trabajo porque el conjunto de datos de entrada tiene una distribución desigual de clases, lo que significa que hay más instancias de una clase que de otra. Como el modelo no está diseñado para manejar esta situación, es posible que se le dé demasiado peso a la clase mayoritaria y que la clase minoritaria no reciba suficiente atención durante el entrenamiento, lo que resultaría en un modelo sesgado y poco preciso. Para resolver este problema, se puede utilizar la técnica de submuestreo aleatorio, que elimina aleatoriamente instancias de la clase mayoritaria para equilibrar las clases. Además, para asegurar que el modelo preste atención adecuada a ambas clases, se utiliza el ajuste de pesos que asigna un peso mayor a la clase minoritaria para que reciba más atención durante el entrenamiento.
- **Control de sobreajuste (Early Stopping):** El Control de sobreajuste, también conocido como Early Stopping, es una técnica utilizada para prevenir el sobreajuste o sobreentrenamiento en un modelo de aprendizaje automático. En este caso, se



monitorea la pérdida en el conjunto de validación durante el entrenamiento del modelo y detiene el entrenamiento si la pérdida no mejora después de un cierto número de épocas.

- **Fijación de la semilla:** La fijación de la semilla aleatoria se refiere a establecer una semilla para el generador de números aleatorios utilizado por el modelo de red neuronal LSTM. La finalidad de la fijación de la semilla es garantizar la reproducibilidad de los resultados, garantizando que se producirán los mismos resultados cada vez que se ejecute el modelo.

Por último, se establecen los hiper parámetros que utiliza la red neuronal y que afectan directamente al proceso de aprendizaje, como el número de épocas de entrenamiento (100 en la fase 1 y 120 en la fase 2), el número de time steps⁴ (5 en la fase 1 y 4 en la fase 2), el número de neuronas ocultas (128 en ambas fases), el valor del dropout (0.5 en la fase 1 y 0.2 en la fase 2), el tamaño del lote (72 en ambos casos) y la función de activación (función relu en ambos casos).

4.3.3.2 Entrenamiento

Una vez diseñado el modelo y ajustado cada uno de sus hiper parámetros, se procede a entrenar el modelo para cada una de las clasificaciones de las maniobras según su nivel de complejidad. Este entrenamiento se realiza utilizando los logs que proporciona el simulador de conducción.

Para ello, dentro del aprendizaje automático, se emplea deep learning, y, dentro de éste, las redes de neuronas LSTM (descritas en el [Apartado 2.4. Redes de Neuronas Recurrentes LSTM](#)).

Para entrenar esta red LSTM se utilizan cuatro capas distintas, cada una con su propio propósito y configuración (mostradas en el [Apartado 4.2 Arquitectura Red Long-Short Term Memory LSTM](#)). A continuación, se describen cada una de estas capas con detalle:

- **Capa LSTM (Long Short-Term Memory):** La primera capa de la red es la capa LSTM en sí misma. En este caso, se utiliza dicha capa con 128 neuronas ocultas. La capa LSTM permite que la red aprenda y recuerde patrones a largo plazo en secuencias de datos de entrada. En este caso, el primer elemento de la forma de entrada es el total de datos que entran en la red. El segundo elemento es el número de pasos de tiempo, el cual se ajusta recordando las últimas 5 secuencias. Y el tercer elemento es el número de características en cada paso, que se ajusta a las 3 primeras columnas del conjunto de datos.
- **Capa Dropout:** La segunda capa es la capa Dropout, que se utiliza para evitar el sobreajuste. En este caso, se establece el valor de dropout en 0.5, lo que significa que se elimina aleatoriamente la mitad de las unidades de la capa durante el entrenamiento para reducir la correlación entre ellas.

⁴ El time step en un modelo LSTM es la unidad básica de procesamiento que permite a la red neuronal modelar patrones complejos en secuencias de datos temporales.

- **Capa ReLU (Rectified Linear Unit):** La tercera capa es la capa ReLU. Esta capa es una capa completamente conectada con 128 neuronas y utiliza la función de activación 'relu'. Esta capa ayuda a reducir la dimensionalidad de los datos de entrada y extraer características relevantes para la tarea de clasificación.
- **Capa Softmax (Salida):** La última capa es la capa de salida, también conocida como capa de clasificación. Esta capa es otra capa Densa con 3 neuronas, una para cada clase en la tarea de clasificación. La función de activación utilizada en esta capa es 'softmax', que normaliza la salida de la capa anterior en una distribución de probabilidad sobre las clases. La capa de salida se utiliza para realizar la clasificación final de la secuencia de entrada.

La red LSTM utiliza la función de pérdida *Categorical_Crossentropy*⁵, puesto que el problema que se aborda es de clasificación multiclase. Además, se utiliza el optimizador *Adam*⁶, y se miden las métricas de error cuadrático medio (mse) y precisión (accuracy) durante el entrenamiento.

Como ya se ha mencionado, los datos de entrenamiento se obtienen de los ficheros transformados a CSV y, a su vez, se dividen en tres conjuntos de datos diferentes. Un primer conjunto de entrenamiento, un segundo conjunto de validación, a partir de la estratificación de datos del conjunto de entrenamiento, y un tercer conjunto de test, que representará los datos a clasificar por el modelo.

Por último, se realiza el entrenamiento, indicando que se deben hacer 100 ciclos y guardando los resultados del error cuadrático medio (mse), precisión (accuracy), las gráficas correspondientes a cada uno, la matriz de confusión de los resultados y los resultados de la clasificación en un nuevo fichero CSV.

Del mismo modo, como este estudio se basa en dos fases de clasificación, se repite el proceso de entrenamiento ajustando los hiper parámetros del modelo y cargando los datos de entrenamiento actualizados con la nueva variable objetivo, como ya se ha mencionado en el [punto 4.3.2.1](#).

4.3.3.3 Clasificación

Una vez realizado el entrenamiento del modelo, se pasa a su evaluación. Se obtienen métricas de evaluación, como el valor de pérdida, la precisión y el error cuadrático medio tanto para el conjunto de entrenamiento como para el conjunto de validación. Además, se generan gráficos para visualizar el progreso del entrenamiento. Se muestra la precisión y la pérdida del modelo en función del número de épocas. Estos gráficos pueden proporcionar información sobre el rendimiento del modelo y la presencia de sobreajuste o subajuste.

⁵ Esta función compara las distribuciones de probabilidad predichas por el modelo con las distribuciones de probabilidad reales de las etiquetas de clase. La entropía cruzada categórica mide la discrepancia entre estas dos distribuciones y produce un valor de pérdida que indica cuánto se aleja la distribución predicha de la distribución real.

⁶ El optimizador Adam es un algoritmo de optimización adaptativo utilizado en el aprendizaje profundo. Es una combinación del método del descenso del gradiente estocástico (SGD) con la adaptación de la tasa de aprendizaje basada en la estimación de momentos de primer y segundo orden de los gradientes.

También se genera una matriz de confusión y un informe de clasificación para evaluar el desempeño del modelo al clasificar cada una de las clases en sus fases correspondientes. La matriz de confusión muestra la cantidad de predicciones correctas e incorrectas para cada clase, mientras que el informe de clasificación proporciona métricas más detalladas, como precisión, recall y puntuación F1, para cada clase.

A continuación, se muestran las matrices de confusión correspondientes a cada una de las fases de clasificación de este proyecto. En la primera fase (ver Figura 5.6) se puede observar, visualizando la diagonal principal de la matriz, que se obtiene una precisión del 100% a la hora de clasificar las instancias. Mientras que en la segunda fase de clasificación (ver Figura 5.7) se puede ver que la precisión de la matriz es superior al 75% y que la variable que más falsos positivos contiene es la que no representa ninguna variación sobre la maniobra que ejecuta el conductor.

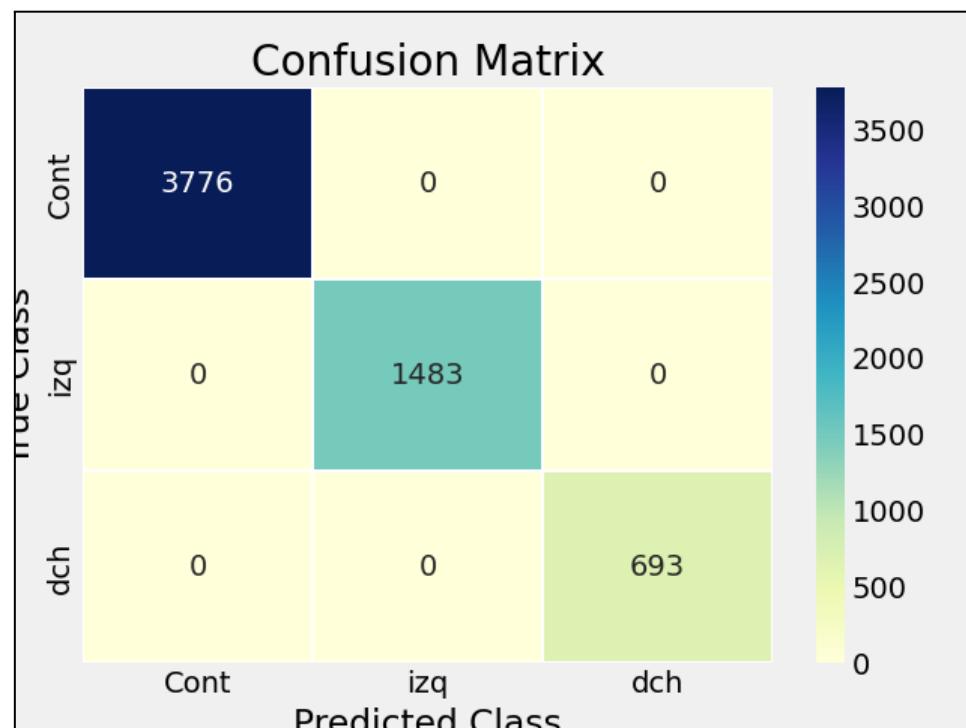


Figura 4.8: Matriz de confusión Primera Fase de Clasificación.

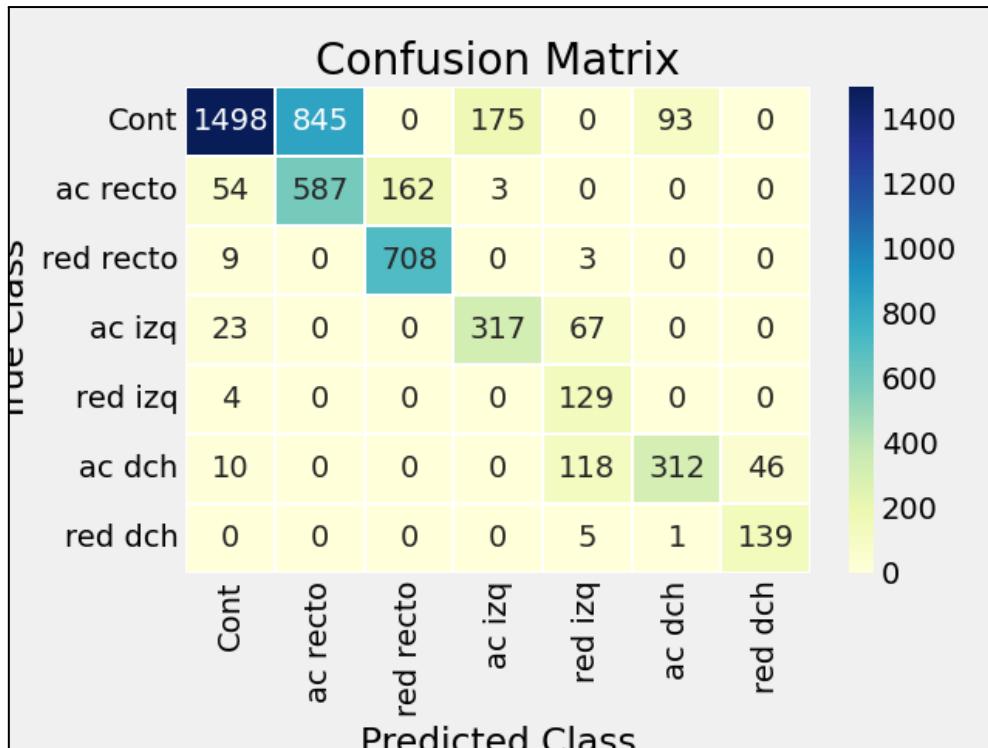


Figura 4.9: Matriz de confusión Segunda Fase de Clasificación.

4.3.3.4 Creación de autómatas

Finalmente, tras las dos fases de clasificación se desarrolla un código para crear grafos que representan maniobras de conducción complejas en un contexto específico. Se toma el archivo CSV que contiene los datos finales de la clasificación, y genera y guarda los grafos correspondientes a las secuencias que cumplen con el patrón deseado.

En este proceso, se analiza la secuencia de estados para identificar las transiciones de estado. Estas transiciones se almacenan en una lista de secuencias, donde cada elemento es una tupla que representa una transición desde un estado actual a un estado siguiente.

Después, se busca en la secuencia todas las subsecuencias que cumplen con el patrón deseado. Esto se realiza mediante un bucle que recorre todas las subsecuencias posibles de longitud igual a la del patrón deseado. Si una subsecuencia cumple con el patrón se generan los subgrafos correspondientes a las secuencias que cumplen con el patrón deseado. Cada subsecuencia que cumple con el patrón se convierte en un subgrafo direccional utilizando la clase DiGraph de la biblioteca NetworkX. Todos estos subgrafos se verifican si cada subgrafo es isomorfo a alguno de los subgrafos existentes para evitar duplicados.

Si se encuentran subgrafos que cumplen con el patrón deseado, se genera una representación visual de cada subgrafo utilizando la biblioteca Matplotlib y se guarda en un archivo PNG.

La representación de estados sigue la misma estructura que la clasificación de la segunda fase (ver [punto 4.3.2.2](#)). Los posibles estados que se pueden representar son los siguientes:

- Q1: Continuar.
- Q2: Acelerar recto.
- Q3: Reducir recto.
- Q4: Acelerar derecha.
- Q5: Reducir derecha.
- Q6: Acelerar izquierda.
- Q7: Reducir izquierda.

Utilizando este sistema se han encontrado secuencias de estados que describen tres tipos de maniobras de alto nivel: adelantamiento por la derecha, adelantamiento por la izquierda y ajustar la distancia de seguridad con el vehículo de delante (ver Figuras 4.10, 4.11, 4.12).

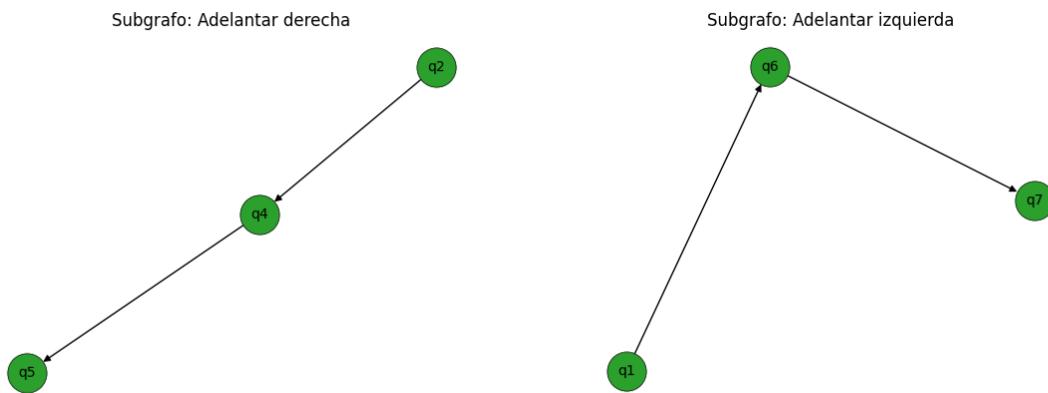


Figura 4.10: Subgrafo maniobra “Adelantamiento por la derecha”.

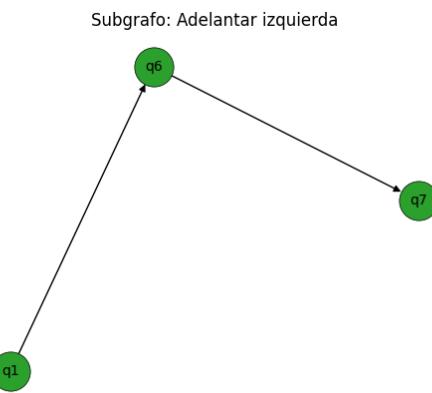


Figura 4.11: Subgrafo maniobra “Adelantamiento por la izquierda”.

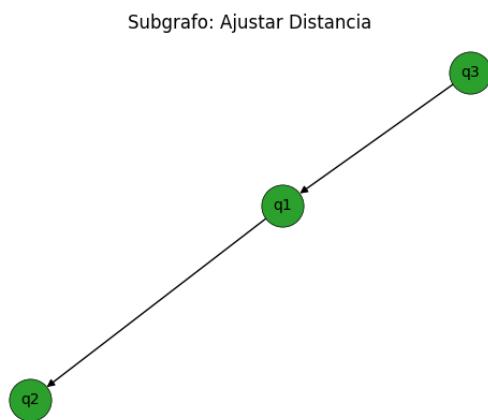


Figura 4.12: Subgrafo maniobra “Ajustar distancia de seguridad”.

Como se puede observar, estos tres autómatas finitos deterministas permiten visualizar la transición entre estados definida anteriormente. Del mismo modo, siguiendo la normativa vigente de conducción, gracias a estos grafos, es posible apreciar el desarrollo de una maniobra compleja en carretera. Estos autómatas ayudan a comprender de una manera más óptima el conjunto de datos iniciales, y crean nuevas oportunidades a la hora de desarrollar sistemas de alarmas para evitar incidentes automovilísticos.

En resumen, el conjunto de procesos que engloba este proyecto ha permitido elaborar estos autómatas a partir de un gran número de pequeños valores recogidos de un simulador (inclinación del volante, marcha del vehículo, presión del acelerador, etc) y que a posteriori se pretenden utilizar en el software de vehículos reales.



5. Resultados y evaluación

En este capítulo se recogen las pruebas realizadas sobre el sistema, con el fin de cumplimentar los requisitos especificados en el [Apartado 3.1 Especificación de Requisitos](#), así como su correcto funcionamiento.

Para ello, se efectuarán pruebas suficientes para comprobar el cumplimiento de todos los requisitos, documentando en este apartado las características de la prueba y evaluando los resultados obtenidos.

5.1 Plan de pruebas

Para comprobar el cumplimiento de todos los requisitos, se seguirá una descripción de las pruebas utilizando la plantilla correspondiente a la Tabla 5.1:

Tabla 5.1: Plantilla documentación pruebas.

Identificador: P-XX	
Descripción	
Resultado esperado	
Resultado obtenido	
RF cubiertos	

A continuación, se explica cada uno de los valores de la plantilla para la descripción de las pruebas:

- **Identificador:** Código único para identificar de forma única cada requisito, y para facilitar la trazabilidad. Los caracteres indicados en la tabla representan:
 - **P:** Prueba
 - **XX:** Toma valores numéricos únicos para identificar únicamente el requisito en cuestión.
- **Descripción:** Breve explicación que detalla la prueba en cuestión.
- **Resultados esperados:** Describe cuál es el resultado necesario para dar la prueba como válida.
- **Resultados obtenidos:** Describe y evalúa el resultado obtenido al realizar la prueba con el sistema implementado
- **RF cubiertos:** Requisitos funcionales que se cubren con la correcta realización de la prueba.

Se procede a presentar el plan de pruebas ejecutado correspondiente a este trabajo fin de grado:

**Tabla 5.2: Prueba P-01.**

Identificador: P-01	
Descripción	Verificar la división de los datos en el conjuntos de entrenamiento y el conjunto de test
Resultado esperado	El sistema muestra la forma de los dos conjuntos en 3 dimensiones (samples, timesteps, features)
Resultado obtenido	Los datos se dividen correctamente y se reestructuran de acuerdo a la forma esperada
RF cubiertos	RF-01, RF-02

Tabla 5.3: Prueba P-02.

Identificador: P-02	
Descripción	Especificar el ajuste de pesos de cada una de las clases a entrenar y clasificar
Resultado esperado	El sistema muestra el peso que corresponde a cada una de las clases con un valor tipo <i>float</i> de acuerdo a las operaciones previas
Resultado obtenido	Los pesos de cada una de las clases se muestran en formato <i>float</i> según la distribución definida
RF cubiertos	RF-10

Tabla 5.4: Prueba P-03.

Identificador: P-03	
Descripción	Especificar el ajuste de hiper parámetros del proceso de entrenamiento del modelo
Resultado esperado	El sistema muestra los hiper parámetros previamente definidos al final del entrenamiento del modelo
Resultado obtenido	El sistema muestra el número de épocas, el dropout y el número de neuronas ocultas
RF cubiertos	RF-03, RF-04, RF-09, RF-13, RF-14

**Tabla 5.5: Prueba P-04.**

Identificador: P-04	
Descripción	Verificar que el sistema muestra las series temporales y el mapa de calor correspondientes a cada una de las variables en función del tiempo
Resultado esperado	Se visualizan las gráficas correspondientes a las series temporales y al mapa de calor
Resultado obtenido	El sistema muestra y almacena las gráficas correspondientes a las series temporales y al mapa de calor
RF cubiertos	RF-06, RF-07

Tabla 5.6: Prueba P-05.

Identificador: P-05	
Descripción	Verificar que el sistema muestra las gráficas correspondientes a las métricas del error cuadrático medio (mse) y la precisión (accuracy)
Resultado esperado	Se visualizan y almacenan las gráficas del error cuadrático medio y de la precisión respectivamente, siguiendo un orden de magnitud respecto al tamaño de los datos
Resultado obtenido	El sistema muestra y almacena las gráficas correctamente
RF cubiertos	RF-05, RF-15

Tabla 5.7: Prueba P-06.

Identificador: P-06	
Descripción	Verificar que el sistema muestra la matriz de confusión correspondiente al entrenamiento del modelo
Resultado esperado	Se visualiza y se almacena la matriz de confusión con el mayor porcentaje de datos dentro de la diagonal principal
Resultado obtenido	El sistema muestra y almacena la matriz correctamente
RF cubiertos	RF-08

Tabla 5.8: Prueba P-07.

Identificador: P-07	
Descripción	Comprobar la actualización de los datos de entrada para cada una de las fases de clasificación y la posterior generación de subgrafos
Resultado esperado	El conjunto de datos almacena la nueva variable objetivo y se muestra procesado y normalizado
Resultado obtenido	Los datos han sido procesados y normalizados correctamente y la nueva variable objetivo (o <i>target</i>) ha sido generada en una nueva columna
RF cubiertos	RF-11, RF-12

Tabla 5.9: Prueba P-08.

Identificador: P-08	
Descripción	Se almacenan los resultados obtenidos después de cada una de las fases de clasificación
Resultado esperado	Se genera un nuevo archivo con los resultados obtenidos del entrenamiento del modelo
Resultado obtenido	El fichero se ha generado correctamente y se almacena
RF cubiertos	RF-16

Tabla 5.10: Prueba P-09.

Identificador: P-09	
Descripción	Verificar que el sistema genera autómatas que representan maniobras de alto nivel
Resultado esperado	Se generan subgrafos correspondientes a la transición de estados definida previamente
Resultado obtenido	Para cada maniobra de alto nivel se generan correctamente los subgrafos
RF cubiertos	RF-17

**Tabla 5.11: Prueba P-10.**

Identificador: P-10	
Descripción	Verificar que el sistema muestra los subgrafos correspondientes a los autómatas que representan maniobras de alto nivel
Resultado esperado	Se visualizan y almacenan los subgrafos de cada maniobra
Resultado obtenido	Los subgrafos se muestran y se almacenan correctamente
RF cubiertos	RF-18

5.2 Matriz de trazabilidad

La matriz de trazabilidad del sistema consiste en una matriz que relaciona las pruebas unitarias del proyecto con los requisitos funcionales del mismo. De esta manera, se puede asegurar el cumplimiento de todos los requisitos funcionales definidos.

Tabla 5.12: Matriz de trazabilidad.

		PRUEBAS UNITARIAS									
		P-01	P-02	P-03	P-04	P-05	P-06	P-07	P-08	P-09	P-10
REQUISITOS FUNCIONALES	RF-01										
	RF-02										
	RF-03										
	RF-04										
	RF-05										
	RF-06										
	RF-07										
	RF-08										
	RF-09										
	RF-10										
	RF-11										
	RF-12										
	RF-13										
	RF-14										
	RF-15										
	RF-16										
	RF-17										
	RF-18										

5.3 Evaluación de los resultados obtenidos

Tras realizar las pruebas y estudiar los resultados obtenidos, se puede concluir que la solución implementada cumple con el objetivo del proyecto. Siendo capaces de distinguir maniobras de conducción de alto nivel a partir de un conjunto de datos de más bajo nivel (eventos atómicos).

A continuación, se presentan las evaluaciones individuales de cada una de las fases del proyecto, las dos fases de clasificación y la generación de autómatas que representan

maniobras complejas. En cada una de ellas se especifica el error cuadrático medio (mse) y la precisión (accuracy) del modelo.

5.3.1 Evaluación de la clasificación: primera y segunda fase

Como se definió en el [punto 1.2](#), el objetivo de cada una de las clasificaciones es escalar un nivel dentro del modelo jerárquico de actividades del conductor (ver Figura 1.5). En la primera fase se parte de un conjunto de eventos atómicos y se busca, con el entrenamiento del modelo, clasificar correctamente las entradas en acciones.

Tras entrenar el modelo en la primera fase, se puede observar que los valores de la clasificación se concentran en la diagonal principal de la matriz de confusión (ver Figura 4.8). Esto nos indica que los datos se han clasificado de forma correcta. Además, para verificar esta premisa, el modelo representa el valores del error cuadrático medio (mse, ver Figura 5.1) y el valor de la precisión del entrenamiento (accuracy, ver Figura 5.2). En ellas, también se puede comprobar como el conjunto de entrenamiento se ajusta perfectamente al conjunto de validación en el número de ciclos marcados.

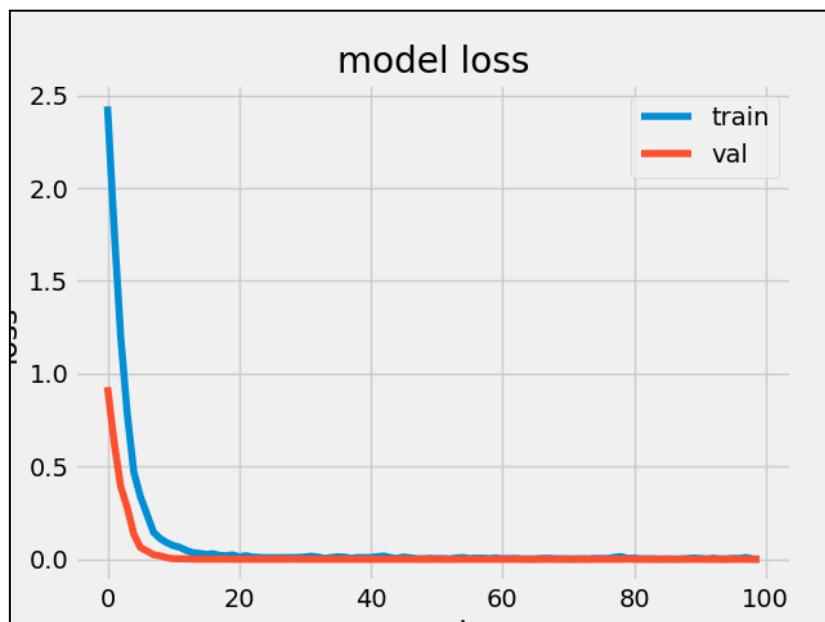


Figura 5.1: Representación gráfica del error cuadrático medio (mse) del modelo primera fase.

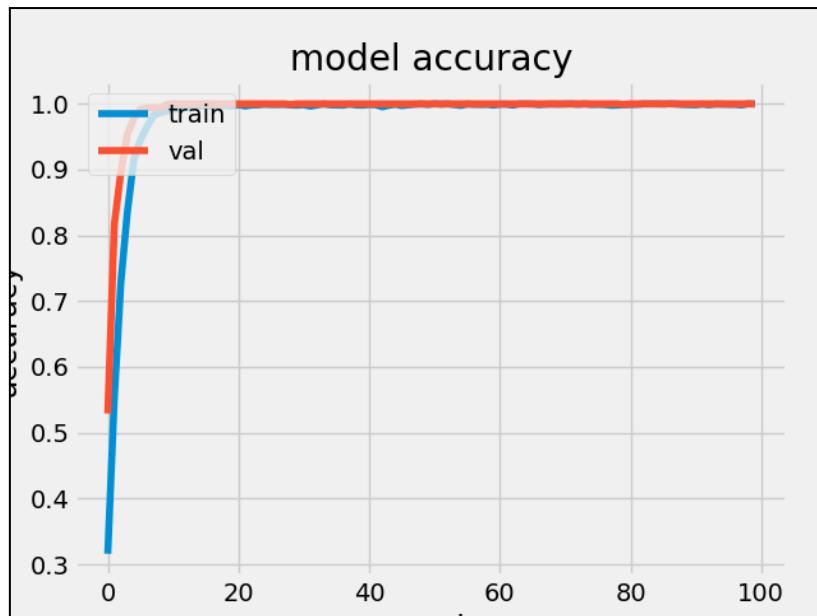


Figura 5.2: Representación gráfica de la precisión (accuracy) del modelo primera fase.

Para la segunda fase se parte de las acciones ya clasificadas en busca de clasificar las nuevas entradas en tareas. Al igual que en la primera fase, tras entrenar el modelo se obtiene una matriz de confusión con el número de entradas clasificadas correctamente (ver Figura 4.9). En esta ocasión la matriz no clasifica correctamente todos los valores, lo que indica que existe un pequeño porcentaje de error en el modelo, el cual se esperaba al escalar en la jerarquía de actividades y al aumentar el número de clases a clasificar. Además, también se representan el error cuadrático medio (ver Figura 5.3) y la precisión (ver Figura 5.4) del modelo para evaluar los resultados.

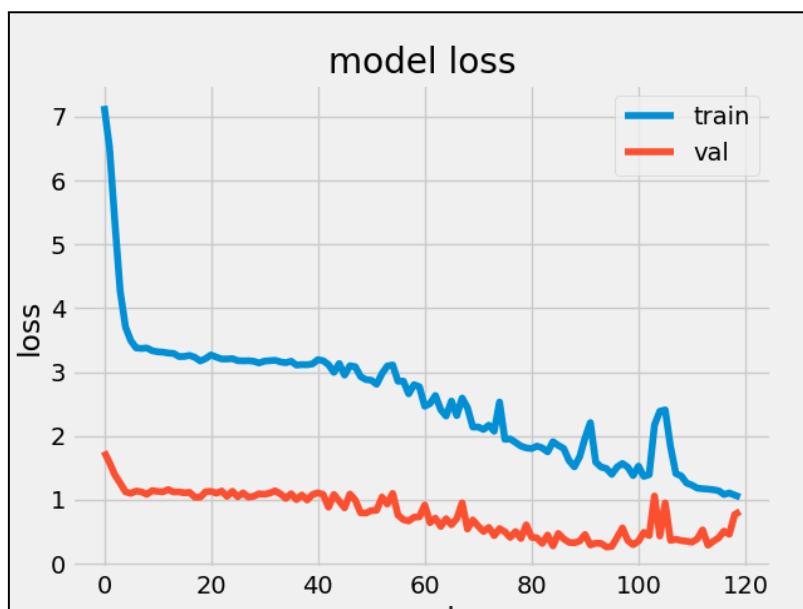


Figura 5.3: Representación gráfica del error cuadrático medio (mse) del modelo segunda fase.

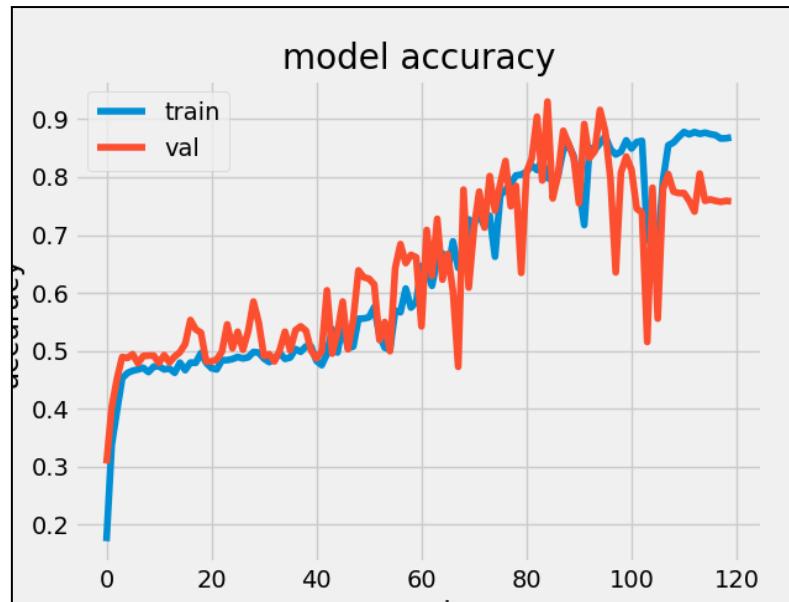


Figura 5.4: Representación gráfica de la precisión (accuracy) del modelo segunda fase.

5.3.2 Evaluación de las maniobras definidas por autómatas

Una vez finalizadas las dos fases de clasificación, el siguiente objetivo es pasar de tareas a poder representar maniobras complejas por medio de autómatas.

Para ello, los valores obtenidos en la segunda fase de este proyecto se deben transformar en estados de transición, dando lugar a un autómata no determinista. De este modo los valores de continuar, acelerar y reducir hacia izquierda o derecha (valores numéricos de 0 a 6) se convierten en estados que van desde “Q1” a “Q7” (ver [punto 4.3.3.4](#)).

Una vez definidos los estados, se forma un grafo (ver Figura 5.5) donde se representan todas las transiciones posibles dentro del conjunto de datos que se ha utilizado en esta experimentación. Cada uno de los estados define una actividad elegida por el conductor en un momento determinado.

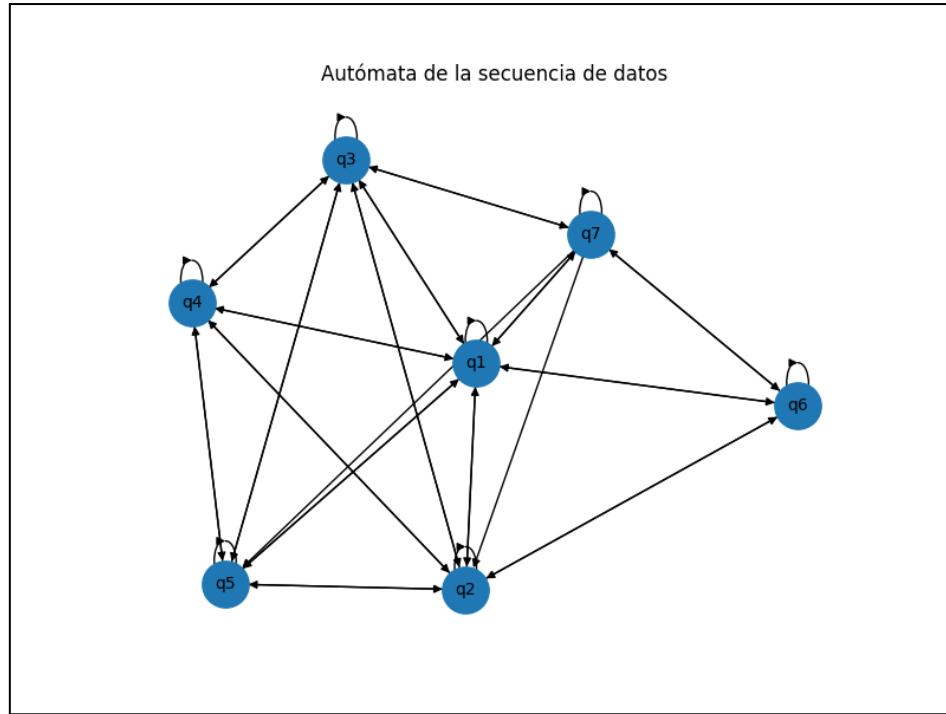
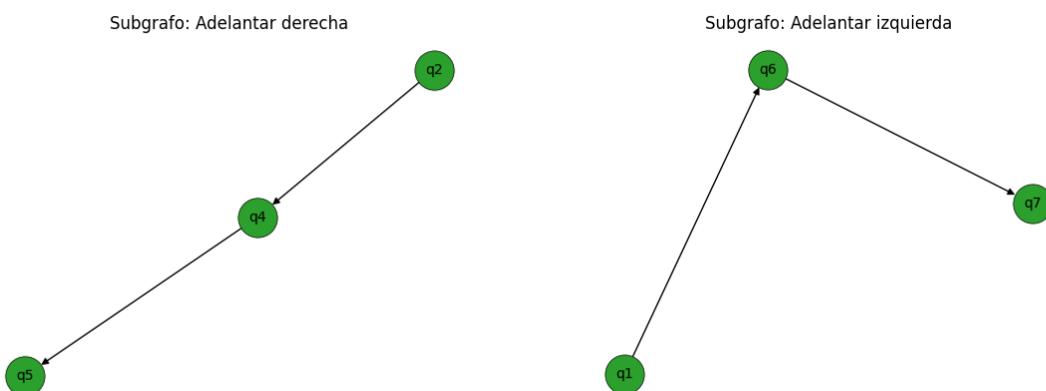


Figura 5.5: Grafo del autómata no determinista del recorrido de la simulación.

Antes de obtener los subgrafos que representan cada una de las maniobras de alto nivel, se deben definir las transiciones entre estados. Estas transiciones definen las tres maniobras representadas en este estudio:

- **La primera maniobra** se corresponde con un adelantamiento por la derecha (ver Figura 6.6, se entiende que ocurre en terreno urbano), donde se tiene la secuencia “(“Q2 ”,”Q4 ”), (“Q4”, “Q5”)”.
- **La segunda maniobra** se trata de un adelantamiento por la izquierda (ver Figura 6.7), donde se tiene la secuencia “(“Q1 ”,”Q6 ”), (“Q6”, “Q7”)”.
- **La tercera maniobra** que se corresponde con ajustar la distancia con el vehículo de delante (ver Figura 6.8), donde se tiene la secuencia “(“Q3 ”,”Q1 ”), (“Q1”, “Q2”)”.



**Figura 5.6: Subgrafo maniobra
“Adelantamiento por la derecha”.**

**Figura 5.7: Subgrafo maniobra
“Adelantamiento por la izquierda”.**

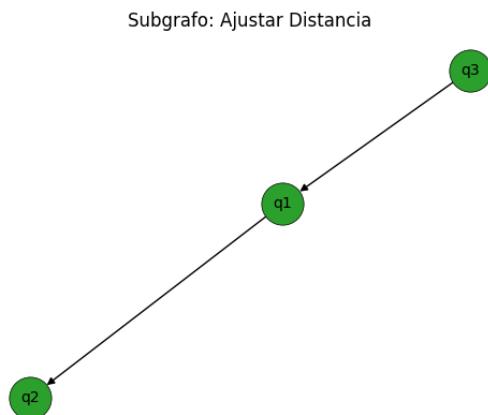


Figura 5.8: Subgrafo maniobra “Ajustar distancia de seguridad”.

Al encontrar coincidencias entre la transición de estados definidos y la sucesión de estados dentro del conjunto de datos se puede concluir que el resultado obtenido del proyecto ha sido satisfactorio.

Gracias a estos subgrafos es posible visualizar de una forma más sencilla y clara el conjunto de datos inicial. Incluso permiten prever la próxima acción del conductor en un momento de tiempo determinado, lo que supone un gran avance para cualquier sistema de creación de alertas o avisos en la conducción. Y además, define el primer contacto del sistema con una posible evolución hacia la implementación modelo en vehículos reales.

6. Gestión del proyecto

En este apartado se describe la gestión del proyecto, desde describir su planificación hasta especificar el presupuesto que costaría implementar este proyecto en una empresa estimando sus costes.

6.1 Metodología de planificación

Para el desarrollo de este proyecto se ha decidido utilizar una metodología basada en el modelo en cascada, definida por Winston W. Royce en 1970 [34]. Esta metodología propone un enfoque secuencial y sistemático para el desarrollo software, basándose en una serie de etapas que deben realizarse de manera descendente. De tal modo que hasta que no se haya completado la etapa anterior, no se podrá iniciar una nueva etapa. En el caso de detectar un error en una etapa previa, se deberá regresar a dicha etapa y, tras efectuar los cambios, repetir el proceso.

Se ha decidido utilizar esta metodología, ya que era la que mejor se ajustaba a este proyecto debido a las características del trabajo a realizar y su facilidad de entendimiento e implementación. Al trabajar con redes de neuronas, agiliza la revisión de los requisitos y del algoritmo desarrollado.

Del mismo modo este tipo de metodologías están orientadas a la documentación, lo que hace muy práctico revisiones dentro del propio documento. El modelo en cascada se adapta de la mejor manera posible a las necesidades de este proyecto. A continuación se muestra el diagrama del proyecto con la metodología en cascada (ver Figura 6.1):

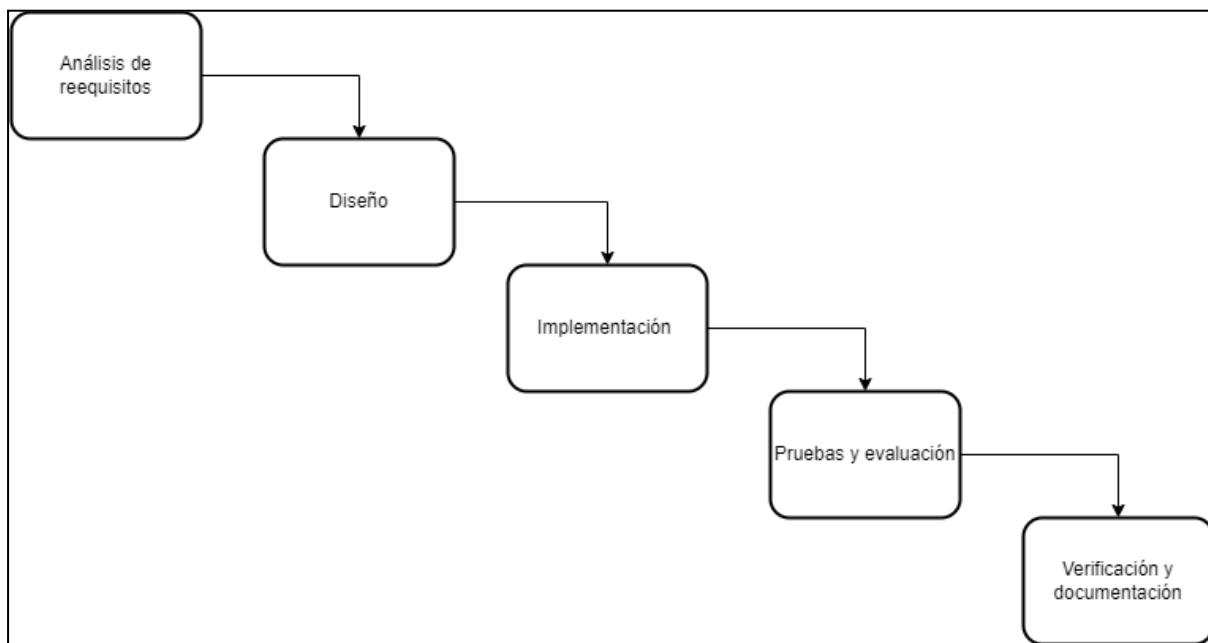


Figura 6.1: Etapas de la metodología en cascada.



Cada una de las etapas se describen de la siguiente manera:

- **Análisis de requisitos:** durante esta etapa se analizan las necesidades del software y se realiza un estudio del problema. Para ello se establecen cuales son los objetivos y los requisitos que se quieren alcanzar, detallando qué tiene que hacer el sistema.
- **Diseño:** en esta etapa se definen los componentes y módulos del sistema en base a los requisitos obtenidos en la etapa anterior, con el objetivo de su posterior implementación.
- **Implementación:** en esta etapa se desarrolla e implementa el algoritmo que cumpla con las necesidades establecidas.
- **Pruebas y evaluación:** durante esta etapa se diseña un plan de pruebas para comprobar que el sistema funciona correctamente y cumple con los requisitos definidos en la primera etapa.
- **Documentación:** en esta etapa se recoge todo el trabajo realizado durante el proyecto en un documento.

Por lo general, la última etapa de la metodología en cascada se describe como una etapa de “Verificación y mantenimiento” del sistema, pero dado que este trabajo no incluye labores de mantenimiento se ha omitido. En su lugar, se ha decidido introducir la etapa de “Documentación” porque es una parte importante en la realización de este proyecto.

6.2 Planificación del trabajo

Este proyecto se ha desarrollado desde el día 16 de septiembre y se ha finalizado el día 11 de agosto, con una duración de 11 meses. Durante el proceso se han realizado unas 14 horas semanales, teniendo en cuenta que se ha disminuido la carga de trabajo durante los períodos vacacionales y de exámenes.

La planificación del proyecto se ha dividido según la metodología en cascada. De esta manera, se han distribuido las tareas en las etapas de análisis de requisitos, diseño del sistema, implementación del sistema, pruebas y evaluación del sistema y documentación. Además, se ha añadido una tarea “Reuniones con los tutores” que se ha llevado a cabo a lo largo de todo el trabajo, por lo que no se ha incluido dentro de ninguna etapa, ni en la planificación Gantt.

De esta forma, se han establecido las siguientes fechas de inicio y fin para cada una de las tareas del proyecto (ver tabla 6.1):

Tabla 6.1: Planificación del proyecto por tareas realizadas.

TAREA	DURACIÓN (DÍAS)	FECHA INICIO	FECHA FIN
Reuniones con los tutores	236	16/09/22	11/08/23
Análisis de requisitos	35	21/09/22	08/11/22
Análisis del sistema anterior	9	21/09/22	03/10/22
Investigación previa	18	21/09/22	14/10/22
Especificación de requisitos	6	17/10/22	24/10/22
Definición de casos de uso	11	25/10/22	08/11/22
Diseño del sistema	46	08/11/22	10/01/23
Diseño de la arquitectura del sistema	40	09/11/22	03/01/23
Definición de redes	3	04/01/23	06/01/23
Interacción entre redes	5	04/01/23	10/01/23
Implementación del sistema	115	10/01/23	19/06/23
Recopilación de datos	9	10/01/23	20/01/23
Procesado de datos	11	23/01/23	06/02/23
Entrenamiento del modelo	94	07/02/23	16/06/23
Clasificación: Fase 1	30	07/02/23	20/03/23
Establecimiento de variable objetivo	3	21/03/23	23/03/23
Clasificación: Fase 2	28	24/03/23	02/05/23
Establecimiento de variable objetivo	3	03/05/23	05/05/23
Generación de autómatas	30	08/05/23	16/06/23
Pruebas y evaluación del sistema	40	19/06/23	11/08/23
Diseño de las pruebas	9	19/06/23	29/06/23
Implementación de las pruebas	21	04/07/23	01/08/23
Evaluación de los resultados	3	08/08/23	11/08/23
Documentación	236	16/09/22	11/08/23
Redacción de la memoria	236	16/09/22	11/08/23

A continuación, se presenta de forma gráfica la planificación en forma de diagrama de Gantt (ver Figura 6.2):

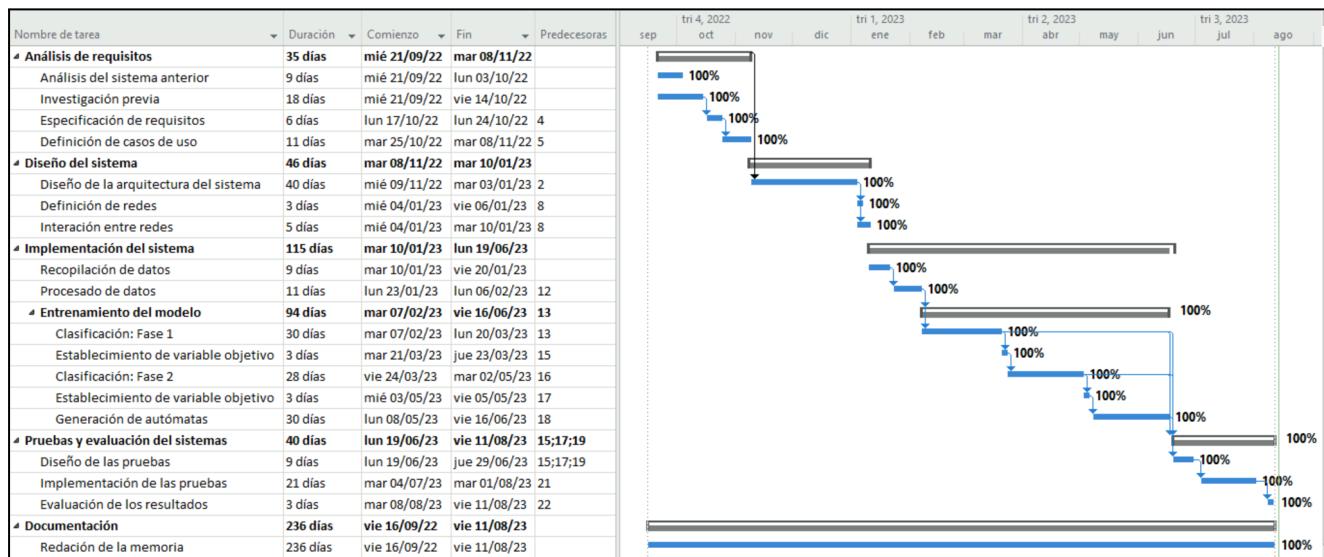


Figura 6.2: Planificación del proyecto en forma de diagrama de Gantt.

6.3 Presupuesto

En este punto se presenta el presupuesto del proyecto, identificando los roles del personal implicado e incluyendo los sueldos de cada uno de ellos. Además se recogerá la estimación del coste del material, ya sea hardware o software, y finalmente, se estimará el coste general del proyecto.

6.3.1 Coste de personal

Antes de desglosar el sueldo de cada uno de los trabajadores, es importante definir cuál es el rol que desempeñan dentro del proyecto y al que se le asociará un coste en función de su implicación.

Para facilitar la división del trabajo, debe existir un rol por cada una de las fases de la metodología, a los que se le añadirá el rol de jefe de proyecto que se encargará de la gestión y supervisión. De este modo, quedarían los siguientes seis roles:

- **Jefe de proyecto:** persona encargada de dirigir, coordinar y gestionar el proyecto.
- **Analista:** persona encargada de la investigación previa y de la obtención y análisis de requisitos.
- **Diseñador:** persona encargada del diseño y de la estructura del sistema.
- **Programador:** persona encargada de la implementación del sistema.
- **Técnico de pruebas:** persona encargada de asegurar el funcionamiento del sistema y probarlo.
- **Investigador:** persona encargada de recoger y documentar la información del proyecto.

Los salarios para cada trabajador se han establecido en función de la tabla salarial publicada en la resolución “«BOE» núm. 59, de 10 de marzo de 2023” [36], tomando como referencia la tabla salarial del área de empresas de ingeniería, oficinas de estudios técnicos, inspección, supervisión y control técnico y de calidad publicada en el “«BOE» núm. 118, de 18 de mayo de 2023”[37]. Además, está establecido en el primer convenio que el número de horas máximo de la jornada ordinaria de trabajo efectivo será de 1.792.

Tabla 6.2: Equivalencia entre roles y categorías profesionales según el convenio.

Rol	Categoría del convenio	Salario anual (€)	Coste por hora (€)
Jefe de proyecto	Grupo profesional I, nivel salarial 1	28.026,81	15,64
Analista	Grupo profesional I, nivel salarial 1	28.026,81	15,64
Diseñador	Grupo profesional II, nivel salarial 3	21.042,77	11,74
Programador	Grupo profesional II, nivel salarial 3	21.042,77	11,74
Técnico de pruebas	Grupo profesional III, nivel salarial 6	15.570,73	8,69
Investigador	Grupo profesional IV, nivel salarial 7	15.130,29	8,44

Teniendo en cuenta estos salarios, así como las horas trabajadas por cada rol, los costes de personal asociados al proyecto serían los siguientes (ver Tabla 6.2):

Tabla 6.3: Costes de personal asociados al proyecto.

Rol	Horas de trabajo	Coste por hora (€)	Coste total (€/h)
Jefe de proyecto	90 h	15,64	1.407,6
Analista	56 h	15,64	875,84
Diseñador	134 h	11,74	1.573,16
Programador	190 h	11,74	2.230,6
Técnico de pruebas	110 h	8,69	955,9
Investigador	58 h	8,44	489,52
Total	638 h	-	7.533,06



6.3.2 Coste de material

En esta sección se especifican los costes de los recursos hardware, como los equipos utilizados, y los recursos software, como las licencias que se han utilizado en las distintas fases del proyecto. Para calcular los costes de cada uno de estos materiales se ha empleado la siguiente fórmula que utiliza el precio del recurso, la duración del proyecto y su tiempo de vida estimado:

$$\text{Coste} = \text{Precio} * \frac{\text{duración del proyecto}}{\text{periodo de vida útil}}$$

Sabiendo que para aplicar la fórmula se estima que la duración del proyecto es aproximadamente de 9 meses, y que la amortización de los recursos se estima en 3 años (36 meses) para los recursos hardware y de 6 años (72 meses) para los recursos software, como las licencias de desarrollo. El resto de recursos que no están compuestos de materiales electrónicos, como el asiento, se estima que será amortizado en el mismo periodo que las licencias de desarrollo (6 años).

Tabla 6.4: Costes de materiales asociados al proyecto.

Recurso	Precio (€)	Periodo de amortización (meses)	Coste en el proyecto (€)
PC Simulador	1771,60	36	442,9
PC Entrenamiento del modelo	949	36	237,25
STISIM Drive + Módulo de Desarrollo	20.000	72	2.500
Televisor LCD Toshiba 32" (3 unidades)	600 (1.800)	36	150 (450)
Juego de volante, palanca de cambios y pedales modelo Logitech G27	229	36	57,25
Asiento Playseat	279	72	34,9
Licencia Microsoft 365	594	72	74,25
Licencia Microsoft Project 2019	151,8	72	19
TOTAL			3.815,55

6.3.4 Costes totales

Para definir los costes totales, primero se debe definir los costes indirectos, como pueden ser la conexión a internet, la factura de la luz, etc. Para ello se establece que estos costes indirectos equivalen a un 10% de los costes directos, los cuales son el conjunto de costes de personal y material.

Tabla 6.5: Costes directos e indirectos.

TIPO DE COSTES	TOTAL (€)
PERSONAL	7.533,06
MATERIAL	3.815,55
DIRECTOS	11.348,61
INDIRECTOS	1.134,86
TOTAL:	12.483,47

Además, de cara a rentabilizar el proyecto se establecen un margen de riesgo y otro de beneficios. Ambos representan un 15% del total de los costes (incluyendo costes directos e indirectos). Finalmente se aplicará el 21% de IVA sobre el total:

Tabla 6.6: Costes totales del proyecto.

CONCEPTO	CANTIDAD (€)
COSTES TOTALES	12.483,47
MARGEN DE RIESGOS (15%)	1.872,52
MARGEN DE BENEFICIOS (15%)	1.872,52
BASE IMPONIBLE	16.228,51
IVA (21%)	6.366,56
PRECIO DEL PROYECTO	22.595,07

6.4 Impacto socioeconómico

En este apartado se analiza el impacto socioeconómico esperado del proyecto. A lo largo de este trabajo se ha desarrollado una red neuronal capaz de clasificar las acciones que realiza un conductor con el fin de identificar con antelación qué tipo de maniobra se pretende realizar. Con esto se pretende poder advertir al conductor si se encuentra en una situación de peligro, con el fin de evitar accidentes de tráfico, ayudados de un gran número de sensores que captan la información del entorno del vehículo y el estado del propio conductor.



Por un lado, la reducción de accidentes de tráfico tendría un impacto positivo sobre los costes de los servicios del Estado, ya sea por la movilización de los servicios de emergencia (tanto sanitaria como fuerzas de seguridad) o por el deterioro de las vías de circulación y su posterior reparación. Gracias a la reducción de accidentes se reducirían los costes asociados y se podrían invertir en otras áreas. Además, la reducción de vehículos que acaban en siniestro supondría un ahorro para los propietarios de vehículos, que podrían invertir ese dinero en otros sectores.

Por otro lado, el sector que podría verse más afectado por la disminución de accidentes serían los talleres mecánicos, que realizarían un menor número de reparaciones, y los fabricantes y vendedores de automóviles, que con un menor número de vehículos siniestros reducirían sus ventas. Sin embargo, este sistema no evita completamente los accidentes, por lo que muchos de estos últimos vehículos que acabarían en siniestro tendrían la posibilidad de únicamente necesitar reparaciones básicas en los talleres, recuperando las pérdidas de este sector.

Finalmente, los fabricantes de vehículos podrían compensar las pérdidas gracias a la incorporación de este nuevo software que permitiría aumentar el precio de salida de los automóviles.

6.5 Impacto social y medioambiental

En este apartado se analiza el impacto social y ambiental relacionado con este trabajo. Los accidentes de tráfico suponen un gran impacto social sobre la sociedad moderna, ya que millones de personas fallecen o sufren lesiones graves cada año en las carreteras de todo el mundo. La prevención de accidentes supone un gran avance positivo para la sociedad, no solo por las víctimas sino también por los daños materiales.

Además, centrándose en el impacto medioambiental se tienen varios impactos negativos: las sustancias tóxicas que se liberan al medio, los recursos que se gastan al reparar y fabricar los vehículos y las consecuencias derivadas del propio accidente, como la alteración del ecosistema o la pérdida de la biodiversidad.

En primer lugar, se debe tener en cuenta que los vehículos utilizan sustancias altamente contaminantes. En numerosas ocasiones cuando se produce un accidente de tráfico se pueden llegar a liberar dichas sustancias, en forma de gases, aceites, anticongelantes, etc. Reducir el vertido de estas sustancias sería muy beneficioso para el medio ambiente, evitando que se puedan llegar a verter cerca de cultivos o llegar a contaminar el agua de los ríos entre otros.

Por otro lado, como ya se mencionó en el punto anterior, la reducción de accidentes de tráfico conllevaría pérdidas para el sector de ventas de automóviles, pero también reduciría el gasto de recursos a la hora de fabricar nuevos vehículos para reemplazar los antiguos siniestrados. Esto tendría un impacto positivo por la parte medioambiental.



Finalmente, cuando se habla de las consecuencias derivadas de un accidente se enfoca en los problemas que causan los vehículos al ecosistema y a su biodiversidad. Muchas veces, además de deteriorar la vía por la que estaban circulando, salen de esta ocasionando daños en el entorno. Algunos de estos daños pueden afectar al hábitat de diversas especies destruyendo sus nidos o acabando con el atropello de algunos animales (por ello se ha desarrollado la iniciativa SAFE, Stop Atropellos de Fauna en España [35]). Otros, sin embargo, afectan al terreno alterando su composición y estructura. Dificultando el desarrollo de la flora o destruyéndola.

Por ello, la reducción de este tipo de accidentes generaría un impacto social positivo tanto en el ámbito social, como en el medioambiental. Y por encima de estos factores, generaría un estado de confianza y seguridad sobre los conductores, lo que reduciría su estrés en carretera y mejoraría el estado de ánimo.



7. Conclusiones y trabajos futuros

En el siguiente apartado se exponen las conclusiones técnicas y personales de este proyecto. De este modo se incluyen las competencias adquiridas y las líneas futuras que se pueden seguir partiendo del presente proyecto.

7.1 Conclusiones técnicas

Tras haber realizado de forma completa el proyecto de modelado de la actividad de conducción mediante inteligencia artificial, se concluye que los objetivos a cubrir definidos al comienzo del documento han sido cumplidos. Se ha logrado diseñar una red neuronal Long-Short Term Memory (LSTM) que permite la clasificación de un conjunto de datos de una simulación en acciones y tareas respectivamente. Y además, se ha conseguido representar los valores obtenidos como un autómata que determina diferentes maniobras de conducción de alto nivel.

Gracias a los resultados obtenidos por el sistema, se puede decir que el modelo podría ser aplicado dentro del software de futuros vehículos. Sin embargo, las restricciones que existen dentro de la simulación indican que es necesario un periodo más largo de evaluación y el desarrollo de pruebas más específicas. Aun así, se ha podido demostrar que el sistema permite escalar la información facilitada por el simulador dentro del modelo jerárquico de actividades.

Por último, cabe destacar que el modelo permite una gran flexibilidad en el momento de realizar el entrenamiento. Pudiendo modificar en cada ejecución un gran número de hiper parámetros, como la propia capa de activación, y permitiendo realizar una gran evaluación al combinar gráficas de precisión, con matrices de confusión, series temporales o mapas de calor.

7.2 Conclusiones personales

Como conclusiones personales, este proyecto ha supuesto una primera toma de contacto con el desarrollo de un software real que puede ser aplicado en un sector determinado. Gracias a ello se han adquirido varias competencias y considerado las características necesarias para alcanzar un proyecto de esta relevancia.

Por un lado, este trabajo ha permitido comprobar la importancia de gestionar y dirigir un proyecto, ya sea con la división de tareas para no solapar unas con otras y permitir el desarrollo sucesivo del sistema, como la importancia de la recogida de información y la documentación del proyecto que se encuentra siempre en constante cambio y evolución.

Además, al igual que el proyecto va evolucionando, el trabajo de un ingeniero informático se encuentra en constante transformación, por lo que es extremadamente importante ajustarse a



las novedades y continuar formándose sin importar el puesto que se desempeñe dentro del proyecto.

Dentro del desarrollo del trabajo fin de grado, una de las capacidades que más se han desarrollado es el manejo de una gran cantidad de información, lo que se conoce como *Big Data*. En primer lugar, mediante el procesamiento del conjunto de datos, desde su limpieza hasta su normalización, y posteriormente en cada uno de los entrenamientos del modelo ha sido necesario actualizar la base de datos, además de preparar el conjunto de cara al siguiente proceso dentro de la cadena sucesiva que forma el sistema.

Por otro lado, el proyecto ha permitido la oportunidad de poner en práctica técnicas desarrolladas a lo largo del ciclo formativo, como el desarrollo de redes neuronales o la planificación *agile*. Esto ha creado la posibilidad de asentar gran parte de los conocimientos informáticos alcanzados durante el transcurso de la carrera.

Por último, como conclusión general, se puede destacar que se ha sido capaz de aplicar todos estos conocimientos informáticos adquiridos de una forma profesional dentro de un problema relacionado con el ámbito de la ingeniería informática. Pudiendo recoger información e interpretar los datos y resultados para emitir un razonamiento lógico en un campo relevante como es la seguridad vial.

7.3 Líneas futuras

En este trabajo se ha desarrollado un sistema que permita, a partir de una mínima información del vehículo, clasificar y representar las maniobras que realiza un conductor con el fin de mejorar la seguridad vial. Para mejorar este modelo, en trabajos futuros se dispone de un amplio abanico de posibilidades.

En primer lugar, en el sistema se han detectado, en la fase de evaluación, ciertos falsos positivos a la hora de clasificar instancias. Para ello, se debe revisar la fase de entrenamiento del modelo y reducir el margen de errores buscando la mejora del rendimiento. Además, para refinar los resultados, conviene probar nuevas arquitecturas del modelo, con los que se podría encontrar una nueva configuración que tenga mejores resultados en el problema de este trabajo.

Sumado a esto, el sistema puede evolucionar para intentar predecir la próxima acción del conductor con el objetivo de prevenir en caso de que esa acción implique cualquier peligro de accidente. El modelo se basa en una clasificación de instancias, sin embargo, se conoce que este tipo de redes neuronales tienen una gran capacidad de predicción sobre series temporales.

Por otra parte, el sistema está reducido a la información procedente del interior del vehículo. Por ello, es posible incorporar al modelo los datos referentes al entorno de la simulación y a los datos del conductor recogidos por sensores. Con ellos es posible determinar nuevas



maniobras que no han podido ser valoradas en este proyecto, como cualquier maniobra de estacionar el vehículo por ejemplo.

Por último, el sistema se ha desarrollado con el fin de ser incorporado dentro del software de vehículos reales para mejorar la seguridad y prevenir accidentes. De esta forma, el siguiente paso del proyecto sería ser trasladado a un vehículo y realizar una nueva evaluación en un entorno real. Además, en este punto sería interesante desarrollar un sistema de alarmas asociado a los resultados obtenidos por el modelo, que sea capaz de advertir en caso de predecir una acción imprudente o que conlleve peligro por parte del conductor del vehículo.



Referencias

- [1] A. Gutiérrez, "Los accidentes de tráfico, principal causa de muerte en jóvenes," 2018. Disponible en:
<https://revista.dgt.es/es/noticias/internacional/2018/1218oms-informe-mundial-accidentes-trafico.shtml>.
- [2] Anonymous "Los accidentes de tráfico se cobraron la vida de 1.004 personas el pasado año," 7-01-. 2022.
- [3] A. Montalvo-Fernandez and A. Barranco Gutiérrez, "Sistema Electrónico De Detección De Baches Y Topes Para Automóviles." , 2021.
- [4] M. J. Flores, J. M. Armingol and A. de la Escalera, "Sistema Avanzado de Asistencia a la Conducción para la Detección de la Somnolencia," Revista Iberoamericana De Automática E Informática Industrial RIAI, vol. 8, (3), pp. 216-228, 2011.
- [5] Simuladores adaptados a cada realidad. Disponible en:
<https://www.landersimulation.com/es>.
- [6] J. H. Caro Castrillón y D. A. Gallego Rodríguez, "Diseño de prototipo de simulador de conducción," 2017.
- [7] E. Regidor et al, "Fracaso en el control del número de víctimas por accidentes de tráfico en España: ¿La respuesta correcta a la pregunta equivocada?" Revista Española De Salud Pública, vol. 76, pp. 105-113, 2002.
- [8] V. M. Zamora España, "Seguimiento Activo De La Mirada Para La Monitorización Del Conductor." , Universidad Carlos III de Madrid, 2018.
- [9] E. Magán López, "Sistema De Alarmas De Ayuda a La Conducción Basado En Lógica Difusa." , Universidad Carlos III de Madrid.
- [10] Anónimo, "¿Cómo funcionan los simuladores de conducción?" 2017. Disponible en:
<https://www.eurotaller.com/noticia/como-funcionan-los-simuladores-de-conduccion>.
- [11] S. Delgado, "¿Son los simuladores de conducción un elemento preventivo de accidentes?" 2017.
Available: <https://fundtrafic.org/son-los-simuladores-de-conduccion-un-elemento-preventivo-de-accidentes/>.
- [12] F. J. Bruna Remiro, "Análisis experimental del simulador de conducción UPV-DGT para la evaluación de conductores con discapacidades severas que conducen con Joysticks de



4 vías," Análisis Experimental Del Simulador De Conducción UPV-DGT Para La Evaluación De Conductores Con Discapacidades Severas Que Conducen Con Joysticks De 4 Vías, 2016.

[13] Los simuladores. ¿realmente ayudan a la hora de conducir?. Disponible en: <https://doncar.es/blog/los-simuladores-realmente-ayudan-a-la-hora-de-conducir/>.

[14] L. Blázquez, "Dynisma DMG-1, "el simulador de conducción más avanzado del mundo","2021.

Available:<https://noticias.coches.com/noticias-motor/dynisma-dmg-1-simulador-de-conduccion/431509>.

[15] PONS, "EL SIMULADOR "ONE & ONE", PRINCIPAL NOVEDAD EN SEGURIDAD VIAL LABORAL DE SICUR 2018," 20/02/, 2018.

[16] F. J. Alonso et al, "No title," Integración De Diferentes Sistemas ADAS En Un Interfaz De Usuario Adaptado a Las Características Del Conductor, .

[17] Anónimo, "¿Cuál es tu actitud al volante?" .

[18] A. E. Caparrós, "El comportamiento humano en conducción: factores perceptivos, cognitivos y de respuesta," Cognición Y Psicología Aplicada a La Conducción De Vehículos, 1999. Available: <https://www.um.es/docencia/agustinx/pca/textos/cogniconduc.pdf>.

[19]Distracciones al volante. Disponible en:

<https://www.dgt.es/muevete-con-seguridad/evita-conductas-de-riesgo/distracciones-al-conducir/>

[20] Anónimo, "Los factores que influyen en el estado físico del conductor," 2017. Disponible en: <https://www.sport.es/es/noticias/automocion/los-factores-que-influyen-en-el-estado-fisico-del-conductor-6223880>.

[21] J. Tosi, M. Tróbolo and R. D. Ledesma, "Actitudes y conductas de riesgo en la conducción," Psicología Para América Latina, (31), pp. 39-52, 2019.

[22]Disfunción de la atención en niños y adolescentes. Available: http://www.mailxmail.com/curso-disfuncion-atencion-ninos-adolescentes/tipos-atencion_

[23] DSM Sistema de monitorización de estado del conductor asistido con Inteligencia Artificial.

Available:[https://azimut.es/fleet/soluciones-azimut/dsm/#:~:text=AZIMUT%20DSM%20\(Driver%20Status%20Monitor,comportamientos%20inseguros%20en%20la%20conducci%C3%B3n.](https://azimut.es/fleet/soluciones-azimut/dsm/#:~:text=AZIMUT%20DSM%20(Driver%20Status%20Monitor,comportamientos%20inseguros%20en%20la%20conducci%C3%B3n.)

[24] (.). Maximice la seguridad vial y del conductor con la tecnología de vídeo inteligente.

Available:

<https://hikvision.com/es/newsroom/blog/maximice-la-seguridad-vial-y-del-conductor-con-la-tecnologia-de-video-inteligente/>.

[25] (diciembre 14.). El aprendizaje automático será invaluable a medida que abordemos la creciente amenaza de los riesgos de seguridad del tráfico en los próximos años. Available: <https://simgular.com/2020/12/14/el-aprendizaje-automatico-podria-reducir-las-victimas-de-accidentes-automovilisticos-en-los-proximos-anos/>.

[26] P. Neira Voces, "Detección de accidentes de tráfico en tiempo real mediante redes neuronales convolucionales," 2022.

Available:

<https://titula.universidadeuropea.com/handle/20.500.12880/3419>.

[27] (Martes, 16 de marzo de). La inteligencia artificial ayuda a reducir los accidentes de tráfico en las ciudades. Disponible en: <https://www.dicyt.com/viewNews.php?newsId=43679>.

[28] I. B. Cruz et al, "Redes neuronales recurrentes para el análisis de secuencias," Revista Cubana De Ciencias Informáticas, vol. 1, (4), pp. 48-57, 2007.

[29] (06/11/). Cómo usar redes neuronales (LSTM) en la predicción de averías en las máquinas.

Available: <https://blog.gft.com/es/2018/11/06/como-usar-redes-neuronales-lstm-en-la-prediccion-de-averias-en-las-maquinas/#:~:text=Las%20LSTM%20son%20un%20tipo,decidir%20cu%C3%A1l%20ser%C3%A1%20el%20siguiente.>

[30] Anónimo, "PREDICCIÓN CON SERIES TEMPORALES CON LSTM, REDES NEURONALES RECURRENTES," .

[31] (Junio 22,). ReLU: Funciones de activación.

Available: <https://sitiobigdata.com/2019/06/22/relu-funciones-activacion/#.>

[32] Anonymous "Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales." vol. I. Disposiciones generales, («BOE» núm. 294, de 06/12/2018.), 06/12/, 2018.

[33] R. Changalvala and H. Malik, "LiDAR data integrity verification for autonomous vehicles," IEEE Access, vol. 7, pp. 138018-138031, 2019.

[34] B. M. Montero, H. V. Cevallos y J. D. Cuesta, "Metodologías ágiles frente a las tradicionales en el proceso de desarrollo de software," Espirales Revista Multidisciplinaria De Investigación, vol. 2, (17), pp. 114-121, 2018.



[35]SAFE, Stop Atropellos de Fauna en España: evaluar la mortalidad de fauna por atropellos.

[36] Gobierno de España, "Resolución de 27 de febrero de 2023, de la Dirección General de Trabajo, por la que se registra y publica el XX Convenio colectivo nacional de empresas de ingeniería; oficinas de estudios técnicos; inspección, supervisión y control técnico y de calidad." vol. III. Otras disposiciones, 10 de Marzo de, 2023.

[37] Gobierno de España, "Resolución de 8 de mayo de 2023, de la Dirección General de Trabajo, por la que se registra y publica el Acta del acuerdo de revisión de las tablas salariales para el año 2023 del XX Convenio colectivo nacional de empresas de ingeniería; oficinas de estudios técnicos; inspección, supervisión y control técnico y de calidad." vol. III. OTRAS DISPOSICIONES, 18 de Mayo de 2023.

[38] Á Maroto Güendián, "No title," Detector Patrones De Comportamiento Del Conductor, 2015.

ANEXOS

Anexo I - Introduction

This chapter deals with the introduction of the Final Degree Project, in which the objectives and the reasons that have driven its realization are set out. It also includes a description of the scope of the project.

The problem of this project starts with the CAOS research group, from the Computer Science Department of the Universidad Carlos III de Madrid, which has been in charge of the development of a multi-agent system that is part of an Advanced Driving Assistance System [8]. Within the framework of this study, the latest work carried out with this project has focused on predicting the driver's decision-making activities with the aim of improving the prevention of traffic accidents [9].

Within this system, the data is divided into different contexts that form an ontology in which the vehicle situation and inputs, the simulation environment and driver information can be represented. Accordingly, we can divide the data into:

- Vehicle information: e.g. speed, gear, acceleration or braking intensity, or degree of steering wheel tilt.
- Environment information: e.g. distance at any angle to the nearest vehicle or pedestrians.
- Driver information: e.g. area of vision or degree of head tilt.

These data are collected thanks to the communication between sensors and the host computer (ADAS), which communicate with each other via an MQTT protocol and form a JSON file with each of the parameters collected in the simulation (see Figure 1.3).

For the development of these simulations, the group has a driving simulator based on STISIM DRIVE (see Figure 1.4). This software makes it possible to recreate a driving environment in real time and collect data in JSON format. For the operation of the system, there are:

- Three screens that visualize the simulated scenario.
- A set of steering wheel, pedals and gears, as well as a seat suitable for driving, in order to make the simulation as realistic as possible.
- A computer on which the developed ADAS is located.
- A computer housing the simulator based on STISIM Drive, connected to the rest of the systems.

In this work, data concerning the vehicle's environment and the driver's state will not be used, as we will focus on the scaling of actions following the hierarchical activity model (see

Figure 1.5). In this way, the project focuses on classifying driver activities with higher complexity from the lowest data in the hierarchy (atomic events). For example, thanks to data such as pedal pressure, steering wheel tilt or the gear the vehicle is in, we can detect whether a driver is about to overtake.

It is important to mention that the simulation environment is equipped with a Microsoft Kinect V2.0 for Windows camera, which allows us to obtain data on the driver's attention. However, this data is irrelevant for this study, so it has been decided to fix it for all simulations.

At present, JSON files are available, which are complicated to process and extract data in a simple way. However, this data has been used to create EXCEL files that collect the data from each of the simulations according to the maneuver carried out in the simulator. Therefore, following the previous work of the CAOS group, the aim of this Final Degree Project is to implement a LSTM recurrent neural network capable of classifying the data in these files into various levels of maneuvers, and finally to represent the highest level in the form of automata.

The scope of this project is to organize and classify the data set collected from a driving simulation using the STISIM DRIVE simulation system in order to represent complex driving maneuvers. To do this, the system must process the data from several simulations in which a driver determines the actions to be performed depending on the simulation environment in which he/she finds him/herself.

Furthermore, the objective of this Final Degree Project is to develop a classification model based on an LSTM network to determine whether a driver's activity at a given moment corresponds to a change of direction to the left or right, or whether he/she continues straight ahead. Subsequently, a second phase of the project will be carried out in which the data obtained in the first phase will be used to classify on a higher scale of maneuvers.

The scope of the project includes the following specific objectives:

- Data processing, including data cleaning and labeling of the target variable to be classified with the network.
- Design and implementation of a Long short-term memory (LSTM) recurrent neural network.
- Classification, at a specific instant of time, of low-level maneuvers performed by a driver using neural networks.
- Writing the classification results in a text file for subsequent processing and further study.
- Extending the classification of higher level maneuvers.
- Identification of maneuvers by means of state-defined automata.

Anexo II - Implementation

This chapter describes in detail the processes of data collection, data processing and the design and implementation of the neural network long-short term memory (LSTM), following the network architecture design scheme (see Figure 4.2: LSTM network architecture).

For the first part of this project, it is necessary to use the ADAS driving simulator for the extraction of data from different tests using a selection of test drivers.

When collecting information about the environment in the STISIM Drive driving simulator, it is important to note that any data from the simulation can be accessed. However, it is necessary to process the data to match the data that a real vehicle would receive.

To detect vehicles in the environment, the simulator uses a module that simulates a LiDAR sensor [33], which emits laser beams to detect objects around the vehicle and measure the distance between them by measuring the time delay between the emission and the reflection of the beam on the object. From the distance between the vehicle and other objects, it is possible to obtain more information about the object, such as whether it is stationary or moving, its speed relative to the vehicle and its trajectory, provided there is a continuous measurement over time (see Figure 4.3).

Using this technology, the simulator can receive information from vehicles in its vicinity. Pedestrians can even be recognised by simulating a front camera, which would be able to distinguish which of the objects in front of it (see Figure 4.4).

Finally, the simulator provides information about the peripherals that are part of the driving experience of a vehicle. These include the following:

- Steering wheel rotation, speed, gear and pedal activation values of the vehicle itself, as numerical values.
- Angle, speed and trajectory of the vehicles in the context.

This data is collected in JSON format (see Figure 1.3: Example of JSON data received from the simulator) and communicated via an MQTT protocol.

Once the data is collected from the simulator, it is transformed into an xlsx file. In this project, data has been collected from five different drivers performing five specific maneuvers:

- **3Step-Turnings:** refers to a three-step process to complete a turning maneuver.
- **Overtaking:** refers to a process of completing a maneuver to overtake another vehicle.
- **Stopping:** refers to a process to complete a vehicle braking maneuver.
- **Turnings:** refers to a process to complete a single turn maneuver.
- **U-Turnings:** refers to a process to complete a U-turn maneuver.

Each file contains a set of data representing the following values (all are represented in the international format):

- **Elapsed Time:** represents the point in time at which the vehicle is located. It is an ascending value against which the time series will be evaluated.
- **Long Dist:** represents the longitudinal distance travelled by the vehicle at a given instant.
- **Lat Pos:** represents the lateral distance between the vehicle and another object next to it.
- **Steering wheel angle:** represents the angle of rotation of the vehicle's steering wheel.
- **Throttle input:** represents the amount of acceleration applied by the driver to the vehicle via the accelerator pedal.
- **Brake pedal force:** represents the force applied to the vehicle brake pedal.
- **Gas pedal:** represents the power of the vehicle's accelerator pedal.
- **Brake pedal:** represents the power of the vehicle's brake pedal.
- **Clutch pedal:** represents the power of the clutch pedal of the vehicle.
- **Left turn:** represents with a binary value if the vehicle is turning left.
- **Right turn:** represents with a binary value if the vehicle is turning right.
- **Gear:** represents the gear the vehicle is in.
- **Speed:** represents the speed of the vehicle.
- **RPM:** represents the revolutions per minute achieved by the vehicle.
- **Hand wheel torque:** represents the amount of force or moment applied to the steering wheel of the vehicle to perform a turning or steering maneuver.
- **Maneuver marker flag:** represents the marker or flag used to indicate that a certain type of maneuver or specific situation has been detected.
- **Accidents:** representation of the number of accidents occurring in the simulation.
- **Collisions:** representation of the number of collisions occurring in the simulation.
- **Peds Hit:** representation of the number of pedestrians hit during the simulation.
- **Speeding Tics:** representation of the number of times the vehicle has exceeded the speed limit during the simulation.
- **Red Lgt Tics:** representation of the number of times the vehicle has caused a traffic violation by not respecting the red light signal during the simulation.
- **Speed Exceeded:** representation of the maximum speed the vehicle has exceeded.
- **Stop Sign Ticks:** representation of the number of times the vehicle has caused a traffic offence by not respecting the stop sign during the simulation.

Once the simulator data has been collected, the data is preprocessed using the Python programming language. This programming language was chosen over other languages due to its ease and speed of implementation, as well as the fact that it offers numerous machine learning libraries.

On the other hand, the definition of a target variable from the dataset is addressed. The importance of a clear definition of the target variable will be examined and techniques for selecting the most appropriate variable for analysis will be explored.

To clean and transform the data, firstly, a code is used to process data from a set of Excel files, eliminating the columns: Maneuver marker flag, Accidents, Collisions, Peds Hit, Speeding Tics, Red Lgt Tics, Speed Exceed and Stop Sign Ticks, which are considered irrelevant to this study. The code uses the Pandas and NumPy libraries to work with the data, and then saves the modified dataframe to a CSV file.

Once the new CSV files are loaded, it is ensured that no columns contain null values. In addition, as this is a classification with LSTM networks, we separate the Elapsed Time column, which is to be treated as a date, and use a dictionary to specify that this column will be assigned to a new column called 'dt' to represent the time series. And the data frame is sorted to set the first columns according to importance within the study.

Finally, we normalize the data to a range between 0 and 1 using the scikit-learn Python library, using the MinMaxScaler class of the preprocessing module. This is done in order to scale the values of different variables to a common range, so that the variables have a comparable scale and the differences in magnitude that may exist between them are eliminated. And using the LabelBinarizer class of the same module, the classification labels are converted into binary form so that they can be used in the training and testing of the machine learning model.

For the development of this work it is necessary to establish a target variable, called "Target". This variable is based on the combination of the binary values of the columns "Left Turn" and "Right Turn". The NumPy function "np.select" is used to create it, and the possible values that it can have are the following:

- #0: Continue straight (0,0).
- #1: Turn left (1,0).
- #2: Turn right (0,1).

After setting the target variable, the training of the model is started. Once this process is finished, a column is obtained that represents the classification of the "Target" (according to the time in seconds of the simulation), previously defined. This new column is added to the dataset to start the second classification phase.

Before starting the second ranking phase, the new tasks to be ranked, represented at a higher level, were decided upon. For this purpose, the following options were presented:

- **Task Increase/Decrease Speed:** Knowing that the driver continues straight ahead (using the previous classification), he/she presses the accelerator/brake and the gear shift. It is possible to classify whether a driver is accelerating or braking on a straight line.
- **Task Stop Vehicle:** Case where a driver presses the brake pedal until his speed decreases to 0 and the gear is downshifted. (The above classification does not apply in this case).

- **Task Start Vehicle:** Case where a driver presses the accelerator pedal until his speed increases from 0 and the gear is upshifted. (In this case the previously performed classification does not intervene).
- **Task Reverse:** Case where a driver presses the accelerator pedal in a specific gear without upshifting or downshifting (reverse gear). (In this case the previously performed classification does not intervene).
- **Lane Change Task (left/right):** In this case a driver turns the steering wheel to one side for a short period of time and turns it back in the opposite direction to move into the lane. In this case the classification could be used previously. In this case, no data on the driver's eyesight is available.

After evaluating all options, it has been decided to define a new target variable called "New_Target", which aims to classify a task of increasing/decreasing the vehicle speed. This variable is based on the results of the classification made in the first phase, the value of the "Gear" column and the "Gas Pedal" column (both compare the value in the current and later time). The possible values they can have are as follows:

- #0: Continue (without increasing or decreasing speed).
- #1: Accelerate straight (increase speed when rating=0 and gear >=gear-1)
- #2: Decrease straight (decrease speed when rating=0 and gear <=gear-1)
- #3: Accelerate right (increase speed when rating=1 and gear >=gear-1)
- #4: Downshift right (decrease speed when rating=1 and gear <=gear-1)
- #5: Accelerate left (increase speed when rating=2 and gear >=gear-1)
- #6: Downshift left (decrease speed when rating=2 and gear <=gear-1)

For the design and implementation processes of the Long-Short Term Memory (LSTM) recurrent neural network, developed with the Python programming language, chosen for its compatibility with the deep learning frameworks Tensorflow and Keras.

Hyper-parameter tuning is a critical stage in the development of neural network models, as the appropriate choice of parameter values can significantly improve the accuracy and performance of the model. Hyper parameters are values that are set prior to model training and directly affect the learning process.

Several techniques are employed in this model such as cross-validation, data stratification, the use of a checkpoint, class balancing (weight adjustment), Early Stopping and seed setting. A description of each technique is given below:

- **Cross-validation:** the use of cross-validation aims to obtain a more accurate estimate of model performance on unseen data, thereby reducing the possibility of overfitting and increasing confidence in the evaluation metrics obtained.
- **Data stratification:** refers to the process of partitioning the data into training and validation sets in such a way that the proportions of the classes are maintained in both partitions. The purpose of stratification is to ensure that the model is trained and evaluated in a balanced way across all classes and that overfitting in a particular class is avoided.

- **Checkpoint:** The use of a Checkpoint aims to prevent overfitting of the model and to ensure that the weights that produce the best performance on the test dataset are saved.
- **Class balancing:** This technique is essential for this work because the input dataset has an uneven distribution of classes, meaning that there are more instances of one class than another. As the model is not designed to handle this situation, it is possible that the majority class is given too much weight and the minority class does not receive enough attention during training, resulting in a biased and inaccurate model. To solve this problem, the random subsampling technique, which randomly removes instances of the majority class to balance the classes, can be used. In addition, to ensure that the model pays adequate attention to both classes, weight adjustment is used, which assigns a higher weight to the minority class so that it receives more attention during training.
- **Early Stopping:** Early Stopping is a technique used to prevent overfitting or overtraining in a machine learning model. In this case, it monitors the loss in the validation set during model training and stops training if the loss does not improve after a certain number of epochs.
- **Seed fixation:** Random seed fixation refers to setting a seed for the random number generator used by the LSTM neural network model. The purpose of seed setting is to ensure reproducibility of the results, guaranteeing that the same results will be produced each time the model is run.

Finally, the hyper parameters used by the neural network that directly affect the learning process are established, such as the number of training epochs (100 in phase 1 and 120 in phase 2), the number of time steps (5 in phase 1 and 4 in phase 2), the number of hidden neurons (128 in both phases), the dropout value (0.5 in phase 1 and 0.2 in phase 2), the batch size (72 in both cases) and the activation function (Relu function in both cases).

Once the model has been designed and each of its hyper parameters has been adjusted, the model is trained for each of the maneuver classifications according to their level of complexity.

For this purpose, deep learning is used within the automatic learning, and, within this, the LSTM neural networks are used.

Four different layers are used to train this LSTM network, each with its own purpose and configuration. In the following, we will describe each of these layers in detail:

- **LSTM (Long Short-Term Memory) layer:** The first layer of the network is the LSTM layer itself. In this case, the LSTM layer is used with 128 hidden neurons. The LSTM layer allows the network to learn and remember long-term patterns in sequences of input data. In this case, the first element of the input form is the total amount of data entering the network. The second element is the number of time steps, which is adjusted by remembering the last 5 sequences. And the third element is the number of features in each step, which is set to the first 3 columns of our dataset.

- **Dropout layer:** The second layer is the Dropout layer, which is used to avoid overfitting. In this case, the dropout value is set to 0.5, which means that half of the units in the layer are randomly removed during training to reduce the correlation between them.
- **ReLU (Rectified Linear Unit) layer:** The third layer is the ReLU layer. This layer is a fully connected layer with 128 neurons and uses the 'relu' activation function. This layer helps to reduce the dimensionality of the input data and extract features relevant to the classification task.
- **Softmax (Output) layer:** The last layer is the output layer, also known as the classification layer. This layer is another Dense layer with 3 neurons, one for each class in the classification task. The activation function used in this layer is 'softmax', which normalizes the output of the previous layer into a probability distribution over the classes. The output layer is used to perform the final classification of the input sequence.

The LSTM network uses the loss function Categorical_Crossentropy, since the problem being addressed is a multi-class classification problem. In addition, the Adam optimiser is used, and the mean square error (mse) and accuracy metrics are measured during training.

Finally, the training is performed, indicating that 100 cycles should be done and saving the results of the mean square error (mse), accuracy, the graphs corresponding to each one, the confusion matrix of the results and the classification results in a new CSV file.

Similarly, as this study is based on two classification phases, we repeat the training process by adjusting the hyper parameters of the model and loading the updated training data with the new target variable, as already mentioned in section 4.3.2.1.

Once the model has been trained, it is evaluated. Evaluation metrics such as loss value, accuracy and mean square error are obtained for both the training set and the validation set. In addition, graphs are generated to visualize the training progress. The accuracy and loss of the model as a function of the number of epochs is shown. These plots can provide information on model performance and the presence of overfitting or underfitting.

A confusion matrix and a classification report are also generated to evaluate the performance of the model by classifying each of the classes into their corresponding phases. The confusion matrix shows the number of correct and incorrect predictions for each class, while the classification report provides more detailed metrics, such as accuracy, recall and F1 score, for each class.

Finally, after the two classification phases, a code is developed to create graphs representing complex driving maneuvers in a specific context. The CSV file containing the final classification data is taken, and the graphs corresponding to the sequences that meet the desired pattern are generated and saved.

In this process, the sequence of states is analyzed to identify state transitions. These transitions are stored in a list of sequences, where each element is a tuple representing a transition from a current state to a next state.

The sequence is then searched for all sub-sequences that match the desired pattern. This is done by looping through all possible subsequences of length equal to the length of the desired pattern. If a subsequence meets the pattern, the subgraphs corresponding to the sequences that meet the desired pattern are generated. Each substring matching the pattern is converted into a directional subgraph using the DiGraph class of the NetworkX library. All these subgraphs are checked if each subgraph is isomorphic to any of the existing subgraphs to avoid duplicates.

If subgraphs matching the desired pattern are found, a visual representation of each subgraph is generated using the Matplotlib library and saved as a PNG file.

The state representation follows the same structure as the second stage classification (see section 4.3.2.2). The possible states that can be represented are as follows:

- **Q1:** Continue.
- **Q2:** Accelerate straight ahead.
- **Q3:** Reduce straight.
- **Q4:** Accelerate right.
- **Q5:** Reduce right.
- **Q6:** Accelerate left.
- **Q7:** Reduce left.

Using this system, sequences of states have been found that describe three types of high-level maneuvers: overtaking on the right, overtaking on the left and adjusting the safety distance to the vehicle in front.

Anexo III - Evaluation

Once the two classification phases have been completed, the next objective is to move from tasks to being able to represent complex maneuvers by means of automata.

To do this, all values must first be transformed into transition states, resulting in a non-deterministic automaton (see Figure 5.5). This graph represents all possible transitions within the data set used in this experimentation.

Before obtaining the subgraphs representing each of the high-level maneuvers, the transitions between states must be defined. In the following, the three maneuvers represented in this study and the representation of the subgraphs that generate each of them are defined:

- **The first maneuver** corresponds to an overtaking from the right (see Figure 5.6, it is understood to occur in urban terrain), where we have the sequence ("Q2", "Q4"), ("Q4", "Q5").

- **The second maneuver** is an overtaking maneuver on the left (see Figure 5.7), where the sequence "(Q1", "Q6"), ("Q6 ", "Q7 ")" is given.
- **The third maneuver** corresponds to adjusting the distance to the vehicle in front (see Figure 5.8), where the sequence "(Q3", "Q1"), ("Q1 ", "Q2 ")" is given.

Anexo IV - Conclusion

After completing the project of modeling the driving activity by means of artificial intelligence, it is concluded that the objectives defined at the beginning of the document have been fulfilled. It has been possible to design a Long-Short Term Memory (LSTM) neural network that allows the classification of a set of simulation data into actions and tasks respectively. Furthermore, it has been possible to represent the values obtained as an automaton that determines different high-level driving maneuvers.

Thanks to the results obtained by the system, it can be said that the model could be applied within the software of future vehicles. However, the restrictions that exist within the simulation indicate that a longer period of evaluation and the development of more specific tests is necessary. Nevertheless, it has been demonstrated that the system allows the information provided by the simulator to be scaled within the hierarchical activity model.

Finally, it should be noted that the model allows great flexibility in the moment of carrying out the training. A large number of hyper-parameters can be modified in each execution, such as the activation layer itself, and a great deal of evaluation can be carried out by combining precision graphs with confusion matrices, time series or heat maps.

As personal conclusions, this project has been a first contact with the development of real software that can be applied in a specific sector. Thanks to this, several competences have been acquired and the necessary characteristics to achieve a project of this relevance have been considered.

On the one hand, this work has made it possible to see the importance of managing and directing a project, whether it be the division of tasks so as not to overlap one with another and to allow the successive development of the system, or the importance of collecting information and the documentation of the project, which is always in constant change and evolution.

Moreover, just as the project is evolving, the work of a computer engineer is constantly changing, so it is extremely important to adjust to new developments and to continue training, regardless of the position held within the project.

Within the development of the final degree project, one of the skills that has been developed the most is the handling of a large amount of information, known as Big Data. Firstly, by processing the data set, from its cleaning to its normalization, and subsequently, in each of the



model training sessions, it has been necessary to update the database, as well as preparing the set for the next process within the successive chain that forms the system.

On the other hand, the project has provided the opportunity to put into practice techniques developed throughout the training cycle, such as the development of neural networks or agile planning. This has created the possibility of consolidating a large part of the IT knowledge acquired during the course of the degree.

Finally, as a general conclusion, it can be highlighted that the student has been able to apply all this acquired computer knowledge in a professional way within a problem related to the field of computer engineering. Being able to collect information and interpret the data and results to issue a logical reasoning in a relevant field such as road safety.

Anexo V - Future lines

In this work, a system has been developed to classify and represent the maneuvers performed by a driver on the basis of minimal vehicle information in order to improve road safety. In order to improve this model, a wide range of possibilities are available for future work.

Firstly, some false positives have been detected in the system during the evaluation phase when classifying instances. For this, the training phase of the model should be revised and the margin of error reduced in order to improve performance. Furthermore, in order to refine the results, new architectures of the model should be tested, with which a new configuration could be found that has better results in the problem of this work.

In addition to this, the system can evolve to try to predict the next action of the driver in order to prevent in case that action involves any danger of an accident. The model is based on a classification of instances, however, it is known that this type of neural networks have a high prediction capacity over time series.

On the other hand, the system is reduced to information from inside the vehicle. Therefore, it is possible to incorporate data from the simulation environment and driver data collected by sensors into the model. With them, it is possible to determine new maneuvers that could not be evaluated in this project, such as any parking maneuver, for example.

Finally, the system has been developed with the aim of being incorporated into the software of real vehicles to improve safety and prevent accidents. Thus, the next step of the project would be to transfer it to a vehicle and perform a new evaluation in a real environment. Furthermore, at this point it would be interesting to develop an alarm system associated with the results obtained by the model, which would be able to warn in case of predicting a reckless or dangerous action by the driver of the vehicle.