UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Detección de noticias falsas y caras de personas manipuladas

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Autor: David Barbas Rebollo

Tutor: Jon Ander Gómez Adrián

Curso 2019-2020

Resum

En este treball final de grau s'estudian les tècniques existents per a detectar desinformació en dos àmbits distints: (i) notícies falses i (ii) imatges facials falses generades per xarxes neuronals.

Durant la primera fase del treball, s'analitzaran les tècniques actuals basades en xarxes neuronals i es reproduiran els resultats de l'estat de l'art. La segona fase del treball consistirà a tractar d'innovar a partir de les tècniques de l'estat de l'art que millors resultats hagen obtingut. La tercera fase consistirà a redactar la memòria, on es compararan els resultats de les variants dissenyades en este treball amb els resultats de les tècniques existents a partir de les quals es van dissenyar.

Paraules clau: xarxes neuronals, notícies falses, processament del llenguatge natural, imatges facials, classificació d'imatges

Resumen

En este trabajo final de grado se estudian las técnicas existentes para detectar desinformación en dos ámbitos distintos: (i) noticias falsas e (ii) imágenes faciales falsas generadas por redes neuronales.

Durante la primera fase del trabajo, se analizarán las técnicas actuales basadas en redes neuronales y se reproducirán los resultados del estado del arte. La segunda fase del trabajo consistirá en tratar de innovar a partir de las técnicas del estado del arte que mejores resultados hayan obtenido. La tercera fase consistirá en redactar la memoria, donde se compararán los resultados de las variantes diseñadas en este trabajo con los resultados de las técnicas existentes a partir de las cuales se diseñaron.

Palabras clave: redes neuronales, noticias falsas, procesamiento del lenguaje natural, imágenes faciales, clasificación de imágenes

Abstract

This work consists in the study of existing forgeries detection tecniques in two different areas of application: (i) fake news and (ii) digitally manipulated faces by means of neural networks.

The work starts by analyzing existing techniques based on deep neural networks and reproducing the state-of-the-art results. During the second stage, the aim will be to try to innovate by proposing different variants from those state-of-the-art techniques which obtained the best results. The third and last stage will consist in writing the document describing this work where the results of the designed approaches will be compared to the results obtained by the techniques from which the approaches were designed.

Key words: neural networks, fake news, natural language processing, facial images, image classification

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

La detección de elementos falsos sobre un conjunto consiste en predecir si pertenecen a la categoría etiquetada como real o falso.

Las noticias falsas, también conocidas como *fake news*, son un tipo de bulo que consiste en divulgar un contenido pseudoperiodístico a través de portales de noticias, prensa escrita, radio, televisión y redes sociales. Su objetivo es la desinformación. Se diseñan y difunden deliberadamente para engañar, inducir a error, manipular decisiones personales, desprestigiar o enaltecer a una institución, entidad o persona, u obtener ganancias económicas o rédito político. No están basadas en hechos. Además, no sólo comprometen la credibilidad de los medios de comunicación y de los periodistas profesionales, sino que suponen un reto para los lectores, puesto que han de diferenciarlas de los hechos reales. La detección de *fake news* supone un problema nada despreciable en el procesamiento del lenguaje natural, útil para todos los proveedores de contenido online.

Las imágenes faciales manipuladas son aquellas que han sido modificadas y/o generadas por herramientas de edicción o métodos de *deep learning*. Lamentablemente, estas técnicas pueden causar problemas potencialmente morales, éticos y legales ya que pueden ser usadas para difamar a individuos, crear identidades falsas y escenas ficticias.

Debido a la preocupante velocidad a la que se difunde esta desinformación, además de a la extensa audiencia que puede alcanzar, mediante redes sociales y medios de internet, han aumentado considerablemente las investigaciones en este ámbito.

1.1 Motivación

La elección de centrarse en el tema de la detección de desinformación se debe a la facilidad actual con la que se puede producir y propagar a través de la red.

A día de hoy, existen múltiples herramientas y aplicaciones que permiten editar y generar imágenes faciales falsas de gran calidad y resolución. Los avances en inteligencia artificial para crear, editar y manipular fotografías están cada vez más accesibles para la población general. A su vez, consiguen resultados casi indiscernibles de la realidad. También, las herramientas profesionales de edición como Adobe Photoshop o GIMP se han convertido en instrumentos sencillos de utilizar gracias a los diversos tutoriales disponibles en internet. El uso incorrecto de estas tecnologías permite la creación de perfiles falsos, saltarse mecanismos de reconocimiento facial o comprometer a individuos haciéndoles aparecer en actos en los que no han formado parte.

Las declaraciones y noticias falsas son un problema recurrente para la humanidad, cuya intención es manipular la opinión pública.

En la actualidad, este problema se ha acentuado, puesto que la información está a un clic de distancia y, por tanto, más accesible para todos. A primera vista, los posibles problemas derivados de estas manipulaciones pueden aparentar no ser importantes. Este hecho difiere de la realidad, puesto que unas pocas líneas de texto pueden tener un impacto social considerable, difamando o ensalzando la percepción de una persona entre los lectores. Paralelamente, pueden generar cambios económicos en el mercado, dando lugar a una opinión sesgada de un producto o empresa. De la misma manera, mediante la manipulación de la información pueden modificar la opinión política.

Todos estos problemas crecen exponencialmente gracias a la propagación masiva a través de redes sociales y páginas web. Por esta razón se cree que es importante estudiar y desarrollar técnicas para detectar esta información falsa y prevenir su mal uso.

1.2 Objetivos

El objetivo principal de este proyecto es desarrollar un modelo capaz de detectar si una noticia o imagen facial es falsa. Para su cumplimentación, se plantea realizar una revisión de las técnicas del estado del arte relacionadas con el aprendizaje profundo en estos dos ámbitos. Un objetivo secundario es comprender estas técnicas y aprender a utilizarlas. Por consiguiente, se realizarán experimentos para alcanzar los resultados del estado del arte con técnicas de *deep learning* y, posteriormente, se intentará innovar para mejorar los resultados ya obtenidos.

1.3 Estructura de la memoria

La estructura de la memoria consistirá de cinco partes principales. Primero, se realizará un resumen del estado del arte relacionado con el tema del trabajo con las técnicas que lo conforman. A continuación, se describirán los recursos usados para realizar este proyecto. Se comentarán, tanto las librerías de *software* como los conjuntos de datos utilizados. Más adelante, se describirá cómo se han preparado los datos para su posterior uso en el entrenamiento de los diferentes modelos, en conjunto con las distintas estructuras utilizadas. Seguidamente, se detallarán las pautas seguidas durante los experimentos y los resultados obtenidos según las métricas usadas. Finalmente, se expondrán las conclusiones obtenidas de la experimentación. Terminando con la revisarán de los objetivos del proyecto y analizando los logros conseguidos en este trabajo.

CAPÍTULO 2
Estado del arte

Desde la aparición de la fotografía, la manipulación y creación de imágenes ha dejado de ser una novedad. Además, desde hace un tiempo se dispone de potentes herramientas de edición de imagen digital. En la última decada, ha habido un gran incremento en la creación de este contenido gracias a la inteligencia artificial, en particular, al uso de las redes de *deep learning*. Esta revolución se debe a las redes generativas antagónicas (*GANs*) [10]. Antes de definir las *GANs*, es importante remarcar el significado de las redes generativas. Como su nombre indica, estas redes tienen como objetivo crear un elemento que maximice la similitud con las muestras reales que intenta replicar. Las *GANs* están compuestas por dos elementos clave. El primero es un generador, que intenta aprender la representación de las muestras reales con el fin de producir muestras similares. El segundo elemento es un discriminador, cuya función es intentar identificar si una muestra dada es del conjunto de muestras reales o generadas. Estas redes se denominan antagónicas, ya que estos dos elementos compiten entre sí. Por un lado, el generador intenta engañar al discriminador, mientras que éste intenta evitar ser engañado. Se puede encontrar información más detallada sobre este tipo de redes en [15].

En la actualidad, las técnicas de manipulación facial se dividen en cuatro categorías: generación completa de imagen, cambio de identidad, manipulación de atributos y cambio de expresiones. En lo referente a la generación de imagen, cabe señalar que tanto la famosa red StyleGAN [17] como su sucesora StyleGAN2 [18] logran generar imágenes faciales hiperrealistas. En el ámbito del cambio de identidad, es decir, donde se cambia la cara de una persona por otra, destacan aplicaciones como FaceSwap [19] y DeepFakes [1]. En cuanto a la manipulación de atributos, donde se editan diferentes caracteres de un individuo, como el pelo o el color de la piel, sobresale la red StarGAN [7]. Por último, en el campo del cambio de expresiones, en el cual se reemplazan expresiones faciales por otras, resaltan las aplicaciónes Face2Face [31] y Neural Textures [32].

Respecto a la manipulación facial, existen en la literatura científica múltiples artículos donde utilizan redes neuronales convolucionales (*CNNs*) por su adecuación a problemas enfocados a aprender características discriminatorias en imágenes. Esto es debido a que las *CNNs* son capaces de detectar posibles defectos en la edición de una imagen y marcas de agua hechas por los generadores, imperceptibles para el ojo humano. Algunos trabajos deciden inspirarse o directamente utilizar los modelos creados, con buenos resultados, para la tarea de clasificación de imágenes. Éstos son reentrenados para identificar manipulaciones faciales. Comúnmente, se suelen usar los modelos ya diseñados para la tarea de ImageNet [28], pero también es frecuente inspirarse en ellos para crean nuevos modelos. Normalmente, al reutilizar un modelo creado para la tarea anterior, éste entrena a partir de los pesos ya aprendidos para esa tarea.

Existen otros estudios que, con la finalidad de obtener mejores resultados, utilizan un filtro de paso alto al inicio de la red. De esta forma, consiguen destacar características discriminatorias de una manera más efectiva [22]. Otros, proponen realizar una detección de las áreas manipuladas mediante mecanismos de atención, con la intención de mejorar la decisión de clasificación de una imagen [9]. Habitualmente, en estos artículos se intenta mantener una división equilibrada en los conjuntos de entrenamiento y test en lo referente a la diversidad de las caras. Cabe mencionar, que existen artículos que describen haber obtenido buenos resultados utilizando métodos tradicionales de *machine learning*, como maquinas vectores de soporte, análisis discriminante lineal y *k* vecinos más cercanos, para asistir a la tarea de clasificación [11].

Es importante resaltar los tipos de noticias falsas que existen con el objetivo de poder detectarlas con mayor facilidad. Éstas se dividen en satiras, bulos y propaganda. Una característica común a las tres es el lenguaje que emplean en su redacción, puesto que utilizan una gran cantidad de recursos lingüísticos, como por ejemplo el lenguaje sensacionalista y la subjetividad. Mediante el uso de información generalizada e imprecisa, intentan transmitir al lector un mensaje falso sin que llegue a detectar ambigüedades en la información comunicada.

Cabe destacar, que los hechos expuestos en el párrfo anterior hacen que la tarea de detección humana de *fake news* sea tediosa y costosa. Otro hecho que dificulta esta tarea es la necesidad de contrastar las declaraciones realizadas con la evidencia real disponible. Por ello, en 2016, se organizó una competición para la detección automática de noticias falsas [2] y, por consiguiente, dar más visibilidad a este problema.

En general, el preproceso de las noticias suele ser similar en las distintas investigaciones revisadas, ya que únicamente se han detectado pequeñas modificaciones entre ellas. El procedimiento más comúnmente utilizado consiste, en primer lugar, en dividir el texto en palabras, eliminar las *stopwords* y aplicar *stemming* o *lemmatisation* a las restantes. Después, se generalizan las palabras con varias presentaciones, y por último, se obtiene una representación númerica de las mismas con el fin de poder analizar las noticias posteriormente. Las representaciones más utilizadas sobre las palabras de un texto son *Term Frequency-Inverse Document Frequency* (*TF-IDF*) y *Linguistic Inquiry and Word Count* (*LIWC*). En cuanto a la representación de secuencias de palabras predomina el uso de *word embeddings* preentrenados, en particular el uso de word2vec [21] y GloVe [25].

Hoy en día, el volumen de los conjuntos de datos se ha visto incrementado y, por ello, la mayoría de métodos actuales para dar solución a este problema se basan en *deep learning*. Consecuentemente, se han obtenido mejores resultados. En concreto, los estudios que obtienen mejores resultados utilizando redes convolucionales, redes *Long-Short-Term-Memory* (*LSTM*) o una red híbrida de las dos anteriores, [5]. También, se encuentran soluciones con mecanismos de atención [26]. Cabe mencionar que algunos estudios consiguen buenos resultados mediante el uso de modelos que no pertenecen a las redes neuronales como maquinas vectores de soporte, *random forests* y *logistic regression* [13].

2.1 Crítica del estado del arte

Como se puede comprobar, existe una gran variedad de investigaciones relacionadas con la manipulación de imágenes faciales con el objetivo de entrenar clasificadores que puedan discriminar las caras reales de las falsificadas. En un gran número de artículos, se suele mencionar la resolución a la cual se redimensionan las imágenes, pero no se explica el motivo por el cual se ha elegido esa dimensión ni cómo realizan la transformación. También, se han detectado problemas a la hora de comparar distintos estudios, ya que no todos usan las mismas métricas de evaluación.

Debido al creciente interés en dar solución a la detección de *fake news* se ha mejorado la calidad de los conjuntos de datos. Esto se debe al considerable aumento del volumen de noticias en los *datasets* a lo largo de los años. A pesar de ello, un inconveniente en el cual cabría incidir es la escasa variación en la temática de los artículos, ya que la extensa mayoría están relacionados con la política. Es importante resaltar la ventaja que supondría el disponer de artículos que versaran sobre diferentes temáticas, puesto que permitiría analizar si el lenguaje usado para transmitir información falsa es similar o no. A pesar de que el preprocesamiento de los datos suele ser similar, en algunas investigaciones no se describe, o si lo hacen, se explica de forma superficial. Ello implicaría, que podrían existir dificultades a la hora de reproducir los resultados de un experimento.

CAPÍTULO 3
Recursos utilizados

3.1 Herramientas utilizadas

En este apartado se exponen las herramientas utilizadas para realizar la parte experimental del proyecto.

3.1.1. Tensorflow

Tensorflow [3] es una librería de software de código abierto de aprendizaje automático. Esta herramienta se creó para el uso interno de equipos de la compañía Google, pero desde 2015 su uso se hizo público a través de la plataforma de Github. De esta manera, se ha elegido dicho *backend* de *machine learning*, ya que ofrece gran flexibilidad y un amplio abanico de posibilidades para la creación de redes neuronales con topologías complejas en distintos lenguajes. A su vez, ofrece soporte para realizar operaciones de cálculo en *CPUs*, *GPUs* y *TPUs*, tanto en equipos de escritorio como en servicios en la nube.

En consecuencia, esta librería se encuentra entre las más utilizadas en el área de investigación y en la creación de productos de diversas multinacionales.

3.1.2. Keras

Keras [8] es una *API* de redes neuronales de código abierto diseñada para facilitar el desarrollo y experimentación de modelos de *deep learning*. Tiene un enfoque para actuar como interfaz con una abstracción más intuitiva y de más alto nivel. De esta manera, facilita el desarrollo de modelos de *deep learning* independientemente del *backend* computacional utilizado. Así pues, agiliza el proceso de desarrollo e implementación y ofrece a los usuarios una curva de aprendizaje más gradual. Por estas razones, hemos decidido utilizarlo para la creación de nuestros modelos. Es posible ejecutarlo sobre distintos *backends* como Tensorflow, The Microsoft Cognitive Toolkit (CNTK) y Theano.

3.1.3. Scikit-learn

Scikit-learn [24] comenzó como un proyecto de Google Summer of Code y, actualmente, se ha convertido en una librería de *machine learning* de software libre para el lenguaje de programación Python. Este incluye una gran variedad de algoritmos de aprendizaje supervisado y no supervisado. Además, está diseñada para interoperar con las bibliotecas numéricas y científicas NumPy y SciPy.

3.2 Conjuntos de datos utilizados

3.2.1. Caras de personas manipuladas

140k Real and Fake Faces

Este conjunto de datos [34] se compone de 70,000 imágenes faciales reales recolectadas por Nvidia de la página web de Flickr, una red social, para usar como *benchmark* para GANs. Estas muestras tienen una considerable variación en términos de edad, origen étnico y el fondo de la imagen. También, encontramos varios ejemplos con accesorios de ropa como gafas, sombreros, bufandas, etc. Debido a esto, es interesante completar el dataset con 70,000 imágenes faciales falsas generadas con StyleGAN [17] por un usuario de Kaggle. Kaggle es una plataforma donde hay competiciones, *datasets* y soluciones relacionadas con *machine learning*. Asimismo, este conjunto de datos está disponible en Kaggle.

Real and Fake Face Detection

Este conjunto de datos [16] es menos extenso que el mencionado anteriormente, con 1081 imágenes faciales reales y 960 imágenes de caras modificadas profesionalmente con Adobe Photoshop. Ha sido creado por el laboratorio de inteligencia computacional y fotografía del departamento de computación de la Universidad de Yonsei, Seúl. El objetivo de este dataset es comparar si un modelo capaz de detectar las características de una imagen generada por una *generative adversial network* también puede discriminar falsificaciones realizadas por expertos. A su vez, este conjunto de datos está disponible en Kaggle.

Las modificaciones han sido realizadas sobre una combinación de los siguientes elementos: uno o dos ojos, la nariz, la boca o la cara en su conjunto. En paralelo, estas ediciones han sido clasificadas según el grado de dificultad para identificarlas, es decir, fácil, intermedio o difícil.

Tabla 3.1: Distribución de las clases del conjunto de datos *Real and Fake Face Detection*.

True	False		
	Fácil	Intermedio	Difícil
1081	240	480	240

3.2.2. Noticias Falsas

Fake and real news dataset

Este conjunto de datos [6] está compuesto por noticias de distintos periódicos online como Reuters, 21st Century Wire, Addicting Info, Fox News, etc. y *tweets* de Rogue POTUS Staff, Donald Trump, Lou Dobbs, entre otros. Está compuesto por un total de 44898 noticias divididas en 21417 verdaderas y 23481 falsas. Este extenso *dataset* incluye el título, el cuerpo, el tema y la fecha de publicación de cada noticia. De la misma manera que los conjuntos mencionados anteriormente, también se encuentra disponible en Kaggle.

Liar dataset

Este conjunto de datos [33] combina 12.8k declaraciones recopiladas de *politifact.com* de la última década. Esta página web publica declaraciones etiquedas, de forma manual, con su grado de veracidad. Además, incluye datos complementarios a las declaraciones, tales como quién la hizo, a que se dedica, en qué contexto se hizo, el tema sobre el que habla, dónde se dijo y a qué partido está afiliado. De igual manera, incluye un recuento de las afirmaciones realizadas por el individuo, clasificadas según su grado de veracidad. Asimismo, la página web incluye una justificación de la clasificación de cada declaración.

La Tabla 3.2 muestra la distribución de las clases del conjunto de datos sobre las particiones de entrenamiento, validación y test con un ratio de 8:1:1, respectivamente. Es importante destacar que el número de muestras etiquedas en la categoría 'pants-fire' es considerablemente menor que el resto.

Tabla 3.2: Distribución de las 6 clases del conjunto de datos *Liar dataset*.

	pants-fire	false	barely-true	half-true	mostly-true	true	total
Train	842	1,998	1,657	2,213	1,996	1,683	10,269
Valid	116	263	237	248	251	169	1,284
Test	92	250	214	267	249	211	1,283

CAPÍTULO 4
Descripciones

4.1 Preprocesamiento de los datos

4.1.1. Imágenes

En lo referente a las imágenes, se dispone de un gran volumen de muestras con distintas resoluciones. Por ello, se ha utilizado un *Image Data Generator*, un objeto de Keras, para, convenientemente, cargar las fotografías en batches a una dimensión concreta, siendo esta de 256x256 pixeles, y separar los conjuntos de entrenamiento, validación y test. Hemos decidido usar esta resolución, ya que es lo suficientemente alta como para mantener muchas características pero, a la vez, no muy alta para poder agilizar el entrenamiento de los modelos.

4.1.2. Textos

Antes de suministrar las noticias a los modelos, se han tenido que realizar varias operaciones para homogeneizar las muestras. Paralelamente, se han intentado eliminar todas las subpartes que de forma directa catalogarían el artículo en su clase.

Después de examinar las noticias del conjunto *Fake and real news dataset* se puso de manifiesto de que la mayoría de las noticias verdaderas citan al editor al inicio del artículo, mientras que las falsas no. Por ello, el primer paso fue suprimir esta porción del texto. También, se detectó que muchas muestras, al ser *tweets* o declaraciones cortas, toda la información se había guardado en el atributo del título. Debido a esto, se decidió juntar el título y el cuerpo del artículo, convirtiendo el título en la primera frase de la noticia.

El preproceso compartido por ambos *datasets* de noticias fue expandir una serie de acrónimos, como US, U.S., U.S.A., U.K. y U.N., para que tuvieran la misma representación. A continuación, se eliminaron las *stopwords* y los signos de puntuación. De forma simultánea, las noticias se dividieron en palabras y éstas se lematizaron, es decir, se cambió la forma flexionada de las palabras por su lema.

Para el conjunto *Fake and real news dataset*, se realizó un segundo preprocesamiento. Para normalizar la diferencia de longitud de las noticias, se subdividieron en frases y se formaron subtextos de hasta 100 palabras. En cuanto a la última frase, y para no perder información, ésta solo se añadía si podía conservar el 25 % de sus palabras. Si no, formaría parte del siguiente subtexto. Por tanto, cada noticia estará formada por uno o más grupos de hasta 100 palabras.

Para el conjunto *Liar dataset*, se agruparon las categorías de las muestras en dos clases con la finalidad de realizar una clasificación binaria. De esta manera, las categorías *true*,

mostly-true y *half-true* se unieron para formar la clase *true*, mientras que las categorías *pants-fire*, *false* y *barely-true* conforman la clase *false*. En la Tabla 4.1 se puede observar la nueva distribución.

Tabla 4.1: Distribución del conjunto de datos *Liar dataset* en 2 clases.

	True	False
Train	4,497	6,072
Valid	616	668
Test	556	727

Finalmente, dado que no es posible introducir nuestros textos directamente a los modelos, los textos deben tener una representación numérica. Por ello, se extraerá el vocabulario que conforman todas las palabras de las muestras de entrenamiento. Cada palabra tendrá un identificador único. Después, se formará un vocabulario de máximo 10,000 palabras. Este se compondrá por las 9,999 palabras más frecuentes. La última palabra del vocabulario, simbolizada por el token <OOV>, servirá para representar las palabras que no se encuentran entre las 9,999 primeras.

Los textos se representarán con un número fijo de palabras. Entonces, si un texto supera el máximo de palabras solamente se considerarán las N primeras palabras y se truncará el texto. En cambio, si no se alcanza este número, se rellenará el vector con ceros hasta la longitud máxima de palabras.

4.2 Tipos de capas utilizadas

Convolucional

Este es un elemento clave de las redes convolucionales que asigna pesos a patrones que encuentra en una muestra. A su vez, reduce el tamaño de la muestra con el fin de capturar las dependencias que están presentes en los filtros más relevantes.

La capa convolucional realiza operaciones denominadas convoluciones definiendo un filtro o kernel sobre una subregión de la muestra, haciendo su producto. El kernel determina sobre cuántos elementos se realiza la operación mientras que el *stride* rige el desplazamiento en las dimensiones de la muestra. La unión de estos filtros se denomina *feature map* que, en esencia, es una característica resumida de la muestra. Normalmente, se aplica una función de activación *Rectified Linear Unit* (*ReLU*) para introducir no liniaridades en el modelo.

Pooling

Las *pooling layers* son una forma de resumir los datos de las capas convolucionales en agrupaciones locales o globales. La agrupación local combina pequeñas subregiones, usualmente de 2x2, mientras que la agrupación global actúa sobre todas las neuronas de la capa convolucional. Estas capas, al reducir la dimensionalidad de los datos, reducen el tiempo de computo de las siguientes capas. Normalmente, se calcula el máximo de la agrupación aunque, también, se puede calcular el promedio.

En nuestros modelos utilizamos principalmente *Max-pooling layers* locales y globales.

Dense

La *dense layer* o también conocida como *fully connected layer* es una capa donde todas las neuronas de la capa actual están conectadas a todas las neuronas de la siguiente capa mediante un peso. Esta estructura es la misma que en una red percetrón multicapa. Por lo tanto, cuantas más neuronas contenga esta capa más aumentará su capacidad de aprendizaje a coste de aumentar el número de parámetros a entrenar. Esto quiere decir que el coste de computación es proporcional a la anchura de la capa.

En los modelos propuestos se ha usado usado este tipo de capa para producir el output de los datos preprocesados en las capas anteriores. Además, se ha utilizado el concepto *fully connected* con la capa *global pooling*. Con ello, no sólo se simula la función de esta capa sino que, además, se aprovecha para realizar una gran reducción de la dimensionalidad de los datos.

Concatenate

Esta capa se emplea para combinar la salida de diferentes capas. Consecuentemente, junta 2 o más tensores a lo largo de una sola dimensión. En nuestro caso, la se usará para unir varias características procesadas de una imagen para ser, posteriormente, todas procesadas por una capa posterior.

Embedding

Esta capa se utiliza para aprender una representación vectorial, de tamaño fijo, de cada palabra. Intenta que las palabras con significado similar tengan una represenatción parecida y, por lo tanto, estarán más proximas entre sí en el espacio de representación. Se sitúa al principio de una red neuronal y aprende de forma supervisada de dos maneras. La primera, es mediante el modelo de *Continuous Bag-of-Words*, en el cual dado un contexto se intenta predecir la palabra correcta para el mismo. La segunda forma es a través del modelo *Continuous Skip-Gram*, donde dada una palabra se predicen cuáles formarían el mejor contexto. En los experimentos realizados se ha utilizado el modelo de *Continuous Bag-of-Words*, aunque es muy popular usar una matriz de *embeddings* preentrenados y deshabilitar el entrenamiento de esta capa.

LSTM

Para introducir el concepto de las redes *Long Short-Term Memory* (*LSTM*), primero, se describirán las redes recurrentes. Las redes recurrentes nacen de la necesidad de recordar, a largo plazo, la información ya procesada, concretamente, de secuencias. De esta manera, a la hora de procesar nueva información se obtiene una mayor cantidad de datos relevantes, en cuanto a su contextualización. En consecuencia, se introdujeron ciclos dentro de las redes para que la información pudiera persistir en la red. Una forma más intuitiva de verlo sería interpretar una red neuronal recurrente como múltiples instancias de la misma red conectadas entre sí de una copia a la siguiente. La siguiente figura muestra cómo se desplegaría una red neuronal recurrente.

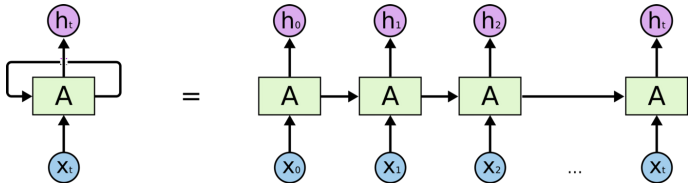

Figura 4.1: Representación de una red neuronal recurrente.

Sin embargo, trabajar con grandes secuencias de datos supone un riesgo. Esto se debe a que al procesar nueva información el riesgo de perder las dependencias ya guardadas es mayor. Este problema se denomina desvanecimiento del gradiente. En respuesta a esta problemática, nacen las redes *LSTM*. Estas redes están formadas por una puerta de entrada, una puerta de salida, una puerta de olvido, las cuales tienen la función de controlar el flujo de información de cada unidad *LSTM*, y una célula de memoria, la cual retiene la información. Se puede encontrar información más detallada sobre este tipo de redes recurrentes en [23].

En este proyecto, se ha elegido trabajar con este tipo de redes, puesto que no sólo son unas de las más utilizadas en el lenguaje natural, sino que, también, lo son en el campo de la detección de noticias falsas.

Dropout

Es una técnica de regularización sencilla y, a la vez, efectiva para mejorar los modelos de redes neuronales. Consiste en ignorar un porcentaje de las neuronas durante el entrenamiento de forma aleatoria. Esta técnica se puede aplicar a cada capa por separado. De esta forma, se evita que las neuronas dependan unas de otras para captar características y evitar un sobreaprendizaje de la red sobre los datos de entrenamiento.

Softmax

Cuando se hace referencia a una capa *softmax*, se alude a una *dense layer* compuesta por N neuronas, igual al número de clases presentes en el problema, cuya función de activación es la función *softmax*. Es considerado un elemento muy común en métodos de clasificación, sobre todo para representar la distribución de probabilidad sobre N diferentes clases cuyas probabilidades suman 1. En nuestro caso, lo se usará para calcular la salida de nuestros modelos.

CAPÍTULO 5
Experimentos y resultados

5.1 Métricas utilizadas

Para evaluar y comparar los resultados obtenidos en los experimentos realizados no sólo se usa la *accuracy* sino también se calculan otras métricas como *precision*, *recall*, *f1-score* y el área bajo la curva *ROC*, *Receiver Operating Characteristic*. Mediante el uso de estas métricas, el grado de información obtenido a partir de los resultados es mayor. Ello nos permite determinar el mejor modelo para dar solución a los problemas encontrados.

Para poder definir las métricas mencionadas hablaremos de *true positives TP*, *true negatives TN*, *false positives FP* y *false negatives FN*. *True positives* son las muestras clasificadas en la clase objetivo correctamente, *true negatives* son las muestras no clasificadas en la clase objetivo correctamente, *false positives* son las muestras clasificadas en la clase objetivo incorrectamente y *false negatives* son las muestras no clasificadas en la clase objetivo incorrectamente.

La medida de *accuracy* es el ratio de las predicciones correctas sobre todas las predicciones realizadas.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

La *precision* es la proporción clasificada en la clase objetivo correctamente.

$$Precision = \frac{TP}{TP + FP}$$

El *recall* es el porcentaje del conjunto de muestras clasificadas en la clase objetivo que realmente pertenece a la clase objetivo.

$$Recall = \frac{TP}{TP + FN}$$

El valor *f1-score* se calcula como la media harmónica entre la *precision* y el *recall*. De esta forma, al combinar estos dos valores es más fácil comparar el rendimiento de los modelos.

$$F1 - score = 2 \times \frac{precision \times recall}{precision + recall}$$

La curva *ROC*, es la representación del ratio de *TP* y el ratio de *FP*. El análisis de la curva *ROC* ayuda a discriminar los posibles modelos óptimos y descartar los subóptimos sin tener en cuenta la distribución de las clases en el conjunto de datos. Entonces, con la métrica del área bajo la curva *ROC* (*AUC*) podemos comparar el rendimiento de los modelos.

5.2 Evolución de los experimentos

En este capítulo, se presentarán los modelos que han sido entrenados a lo largo de los distintos experimentos realizados. Además, se explicarán las modificaciones que se han ido realizando y los problemas que se han resuelto. Posteriormente, se compararán los resultados obtenidos con los resultados publicados para cada conjunto de datos con los que se ha trabajado.

5.2.1. 140k Real and Fake Faces

Primer modelo

Antes de profundizar en los resultados obtenidos en los diferentes modelos, resulta importante explicar, de forma simplificada, el módulo empleado para la detección de caras falsas. Es un bloque formado por una capa convolucional sin una función de activación especificada seguido por un batch normalization layer que finaliza con la función de activación de tipo *rectified linear unit* (*ReLU*).

El motivo por el cual se usa este bloque es ajustar los valores obtenidos de las convoluciones con la *batch normalization layer* y, así, evitar el problema de desvanecimiento de gradiente. De esta forma, se pretende que la red siga aprendiendo a lo largo del entrenamiento. Además, así, se podrán representar los modelos de una forma más compacta.

A partir de este momento, cuando se haga referencia a una capa convolucional se estará haciendo referencia a este módulo.

De la misma manera, se propone la utilización de una red convolucional poco profunda compuesta por convoluciones con pocos filtros como método para abordar la tarea de identificación de caras manipuladas. A través de su uso, se quiere investigar la capacidad que posee para detectar las características clave de cada subconjunto. En la siguiente figura se muestra la topología de la red.

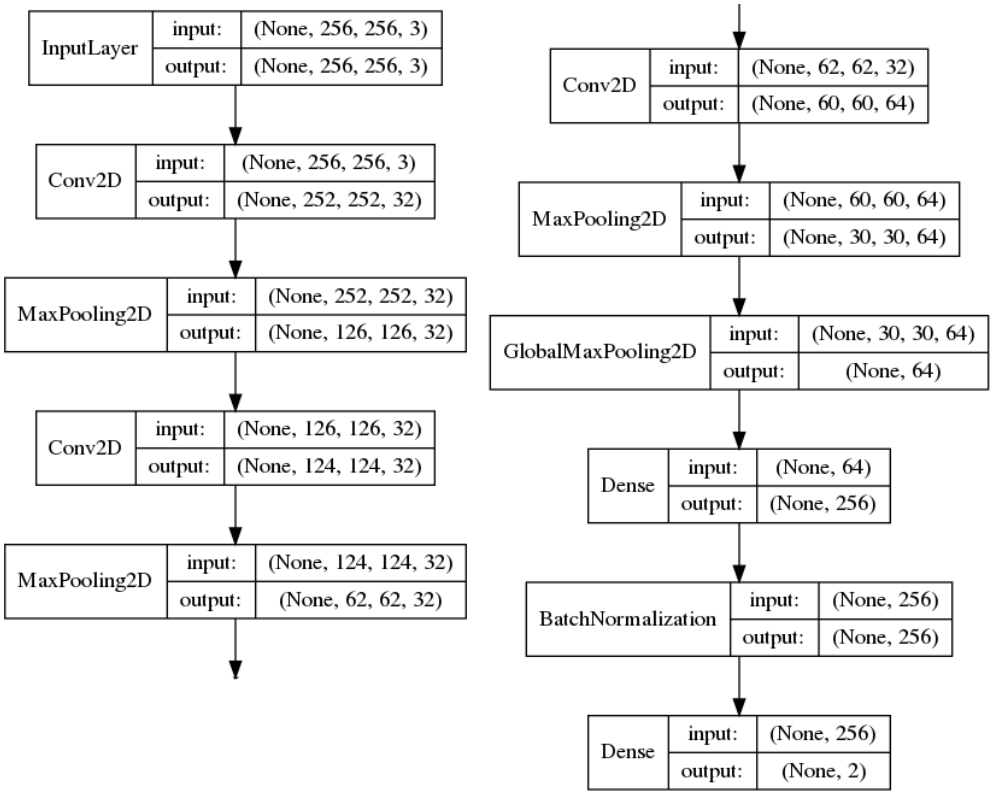

Figura 5.1: Topología del primer modelo.

A continuación, a través de dos gráficas, se muestran los resultados obtenidos por el primer modelo. En ambas gráficas, se puede apreciar cómo los picos de *loss* coinciden con los de *accuracy*. Aún así, se puede asegurar que los resultados obtenidos tanto en entrenamiento como test, son buenos. Concretamente, en test se ha conseguido alcanzar un valor de 96.74 % de accuracy con los pesos de la red calculados en la epoch 56.

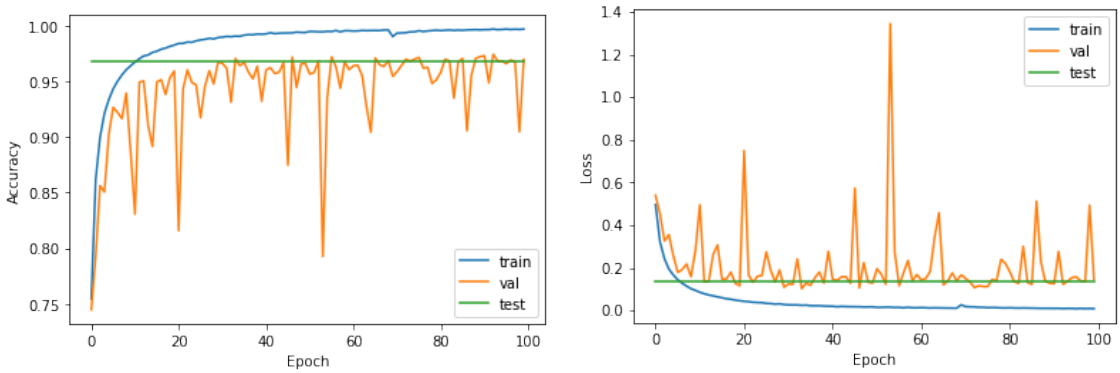

Figura 5.2: Gráficas mostrando la evolución del *accuracy* y *loss* del primer modelo. También se muestra el valor de *accuracy*, 96.74 %, y *loss*, 0.134, obtenido en test con la red entrenada en la epoch 56.

Segundo modelo

El segundo modelo se obtiene a partir de la adición de tres capas convolucionales. El tamaño de la totalidad de los kernels usados es 3, con el fin de conseguir una reducción progresiva de la imagen. De esta manera, se consigue aumentar, de forma progresiva, el número de filtros, hasta llegar a la cifra de 256. La razón por la cual se ha decidido realizar dicha modificación es mejorar el acierto de clasificación a partir de un mayor número de características discriminatorias de las imágenes. La siguiente figura muestra la topología de la red.

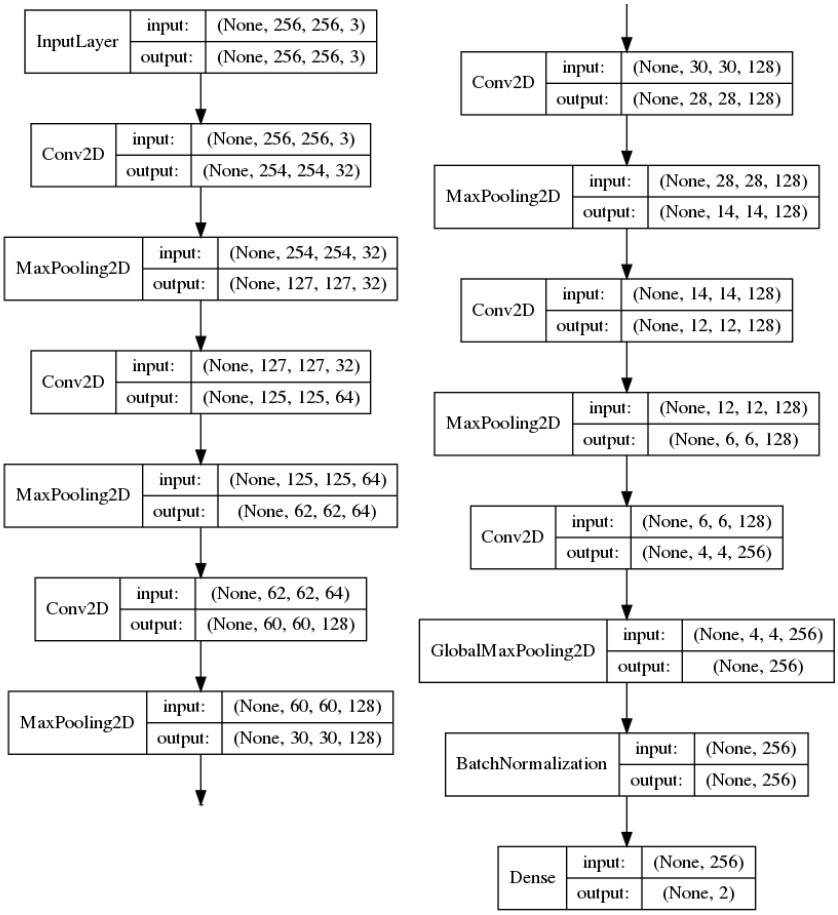

Figura 5.3: Topología del segundo modelo.

Estas dos gráficas ponen de manifiesto que el segundo modelo tiene la capacidad de realizar un mejor aprendizaje que el anterior. Dicha afirmación se obtiene tras la interpretación de las gráficas, puesto que al compararlas con las del modelo previo se observa que los valores de accuracy obtenidos convergen de una manera menos acentuada y más progresiva. A su vez, conviene reseñar que los resultados obtenidos por este modelo mejoran a un 99.54 % de *accuracy* en test con los pesos de la red calculados en la epoch 87.

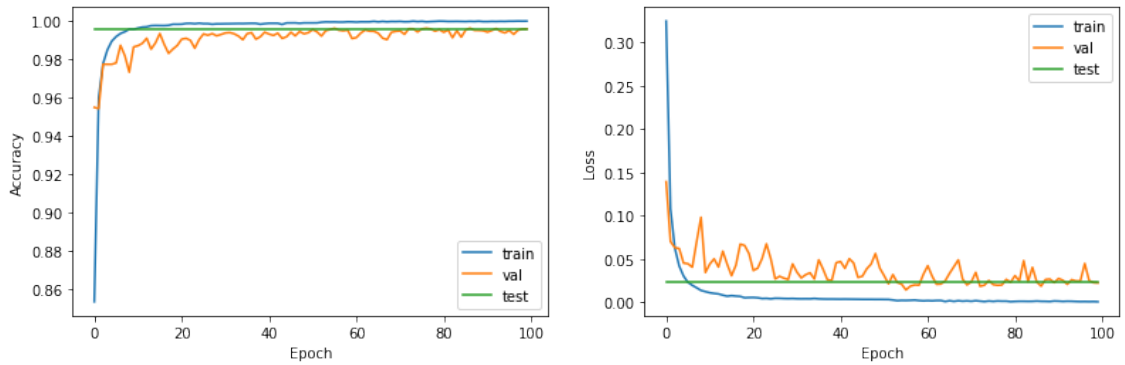

Figura 5.4: Gráficas mostrando la evolución del *accuracy* y *loss* del segundo modelo. También se muestra el valor de *accuracy*, 99.54 %, y *loss*, 0.023, obtenido en test con la red entrenada en la epoch 87.

Tercer y cuarto modelo

Estos modelos se crearon gracias a la realización de pequeñas modificaciones sobre el segundo modelo. Las modificaciones se centraron en efectuar cambios sobre el tamaño del kernel, y el número de filtros calculados por las convoluciones. También, se realizó un crecimiento más progresivo de los filtros usados. Como ambos modelos obtuvieron resultados similares con respecto al modelo anterior, no se considera relevante incidir más sobre ellos.

Quinto y sexto modelo

Antes de describir los modelos cinco y seis conviene definir el concepto de *transfer learning*, utilizado de forma frecuente en este ámbito. Se podría definir como la habilidad de aplicar lo que ha sido aprendido en un determinado contexto a nuevos contextos.

Así pues, para la creación de los modelos cinco y seis reutilizaremos redes que hayan obtenido buenos resultados en la tarea de ImageNet [28]. En este sentido, se utilizarán los pesos entrenados para esta tarea. Además, se deberá adecuar la entrada de la red con el objetivo de poder utilizar imágenes de 256 pixeles. La segunda modificación realizada se basa en sustituir las últimas *fully connected layers* de la red. Éstas se sustituirán por una *Global-avergae-pooling layer*, una *batch normalization layer*, una *dropout layer*, en la cual se omitirán el 30 % de las neuronas, y una *softmax layer*.

En lo referente al diseño del quinto modelo se ha decidido usar la red VGG16 [30] aplicando el concepto de transfer learning. Se ha elegido esta topología, ya que no es frecuente su uso por el estado del arte. Asimismo, se introdujo el concepto de acumular convoluciones para aumentar la efectividad del campo receptivo de las convoluciones manteniendo las ventajas de las convoluciones con un kernel pequeño. Al realizar 3 convoluciones seguidas con kernel de 3x3, se obtiene el mismo campo receptivo que una convolución con kernel 7x7 asociado a un menor coste computacional. En la Figura 5.5 se puede ver la topología original de este modelo.

Una vez procesados los datos se puede concluir que los resultados obtenidos son muy satisfactorios, puesto que se ha conseguido una clasificación casi perfecta, concretamente de 99.93 % de *accuracy* en test con los pesos de la red calculados en la epoch 97. Estos resultados se muestran en la Figura 5.7, donde se evidencia que la red ha logrado aprender las características discriminatorias de los datos. Asimismo, aunque lo descrito anteriormente haya sido un evento muy satisfactorio, resulta interesante resaltar que durante la fase de entrenamiento el aprendizaje no fue ni lineal ni progresivo.

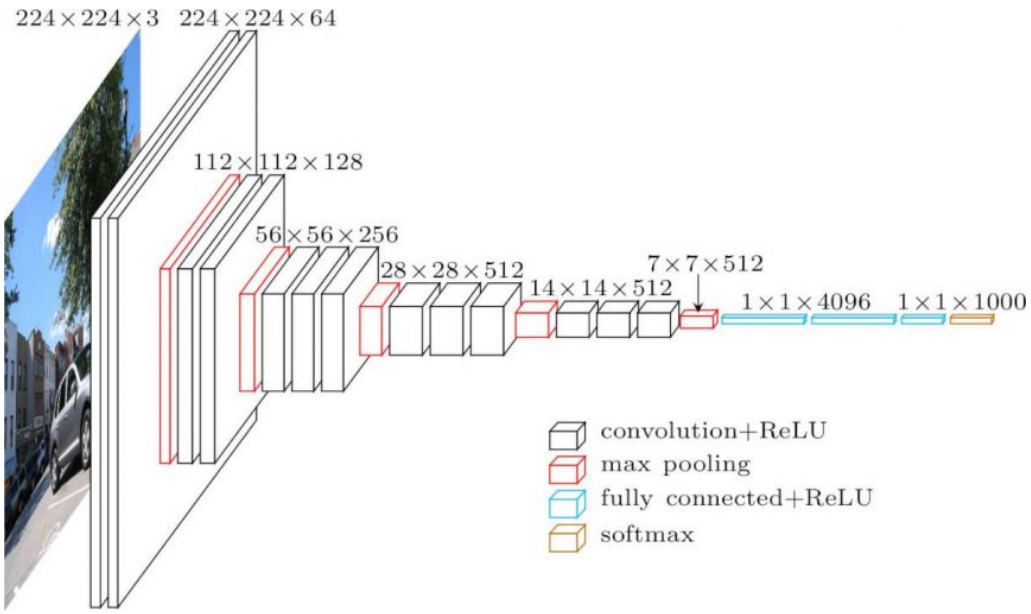

Figura 5.5: Topología original de la red VGG16.

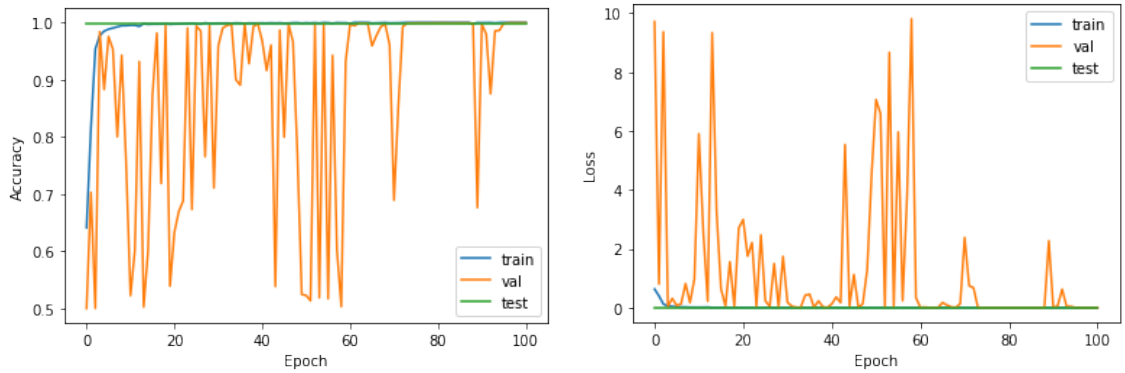

Figura 5.6: Gráficas mostrando la evolución del *accuracy* y *loss* del quinto modelo. También se muestra el valor de *accuracy*, 99.93 %, y *loss*, 0.003, obtenido en test con la red entrenada en la epoch 97.

Con respecto al sexto modelo, y en consonancia con lo explicado al inicio del apartado, se ha optado por implementar el concepto de *transfer learning* a la red ResNet50V2 [14]. En relación a la elección de esta red, es importante indicar que, al igual que en el quinto modelo, se ha elegido dicha red por su escasa participación en el estado del arte. Además, permite introducir modelos residuales.

En paralelo, con la finalidad de evitar el sobreaprendizaje de las redes se aumentó la profundidad de las redes convolucionales. Derivado de esta acción, se incrementó el problema de desvanecimiento del gradiente. En consecuencia, con el objetivo de solucionar dicho problema se introdujo un salto de una o más capas en las redes. Posteriormente, las ramas formadas por los saltos se unen mediante una operación de suma. Todo ello, conlleva la posibilidad de crear redes de mayor profundidad.

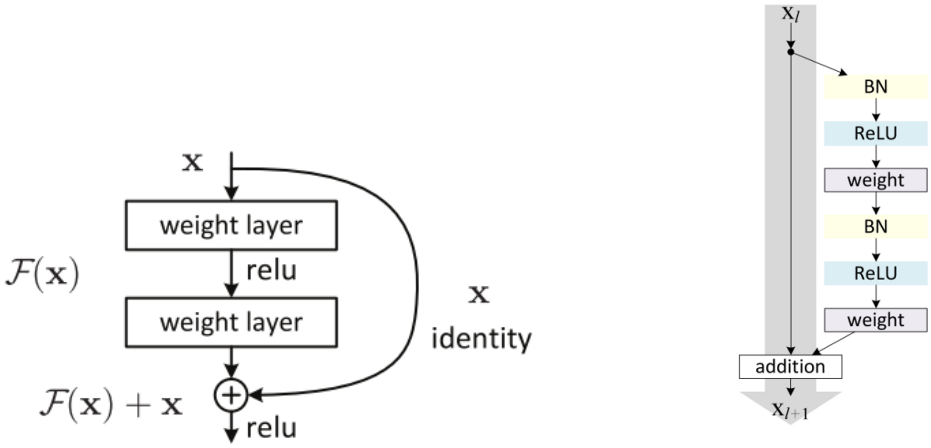

Figura 5.7: Representación básica de un modulo residual y la representación de los modulos residuales usados por ResNet50V2.

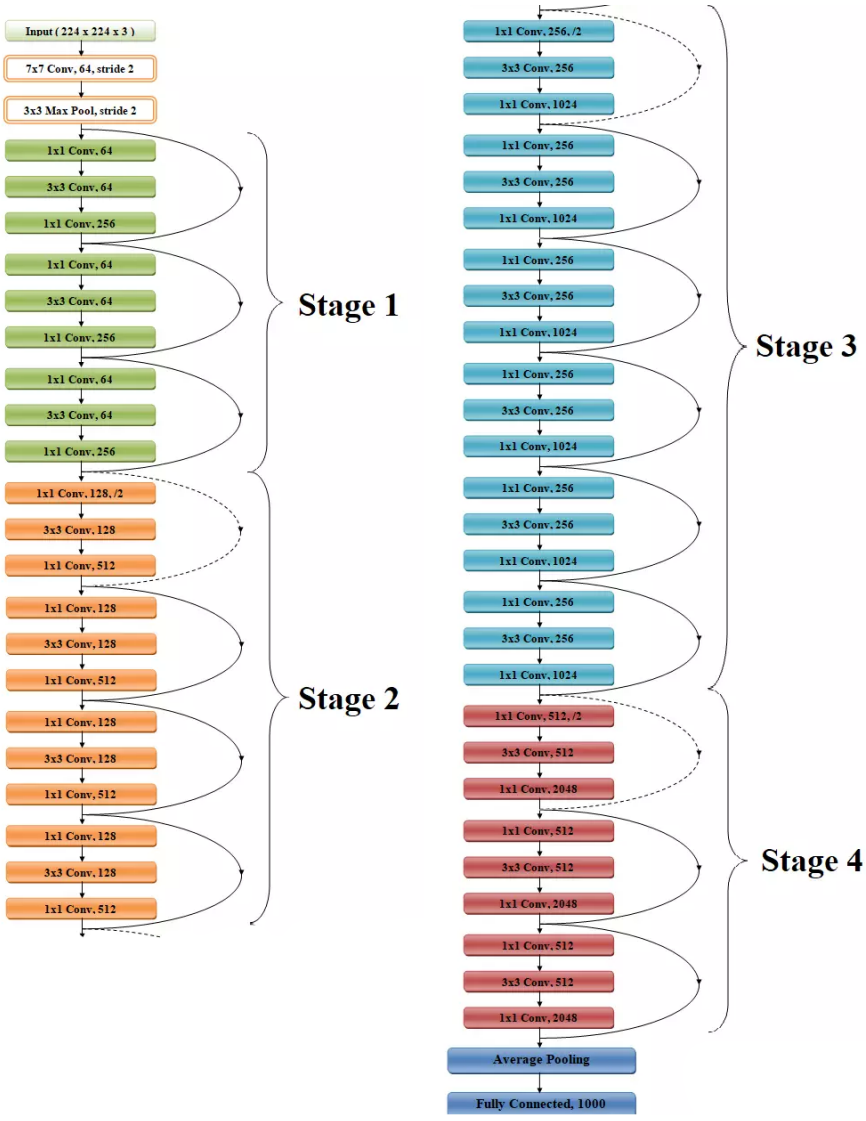

Figura 5.8: Topología original de la red ResNet50V2.

En este caso, los resultados obtenidos no son tan satisfactorios como los obtenidos en el quinto modelo, siendo el valor obtenido de *accuracy* en test en la epoch 89 de 96.86 %. A pesar de tener una estructura para combatir el sobreaprendizaje, en las gráficas representadas, se observa como aprende muy rápido las características de las muestras de entrenamiento. Describiendo de forma más detallada las gráficas, se da uno cuenta de que en la epoch 20 el valor de *accuracy* y *loss* del entrenamiento convergen a 1 y 0, respectivamente, hecho que pone de manifiesto un sobreaprendizaje. En consecuencia, se observan cambios abruptos en validación.

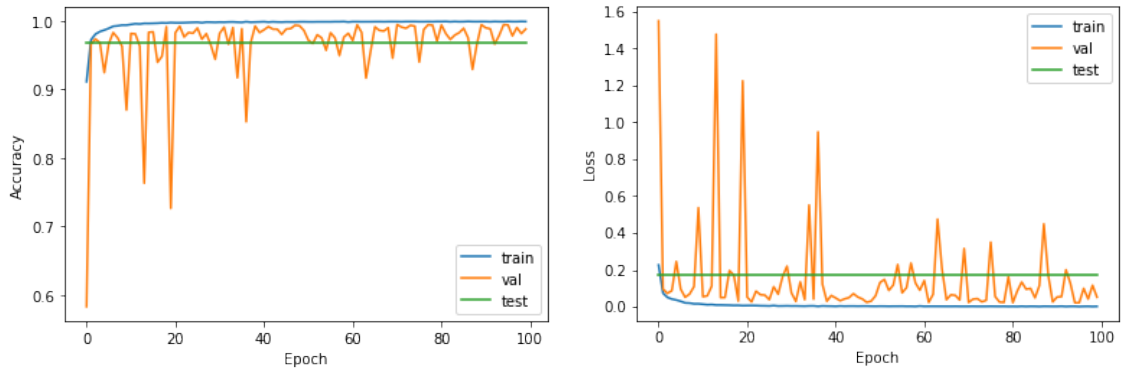

Figura 5.9: Gráficas mostrando la evolución del *accuracy* y *loss* del sexto modelo. También se muestra el valor de *accuracy*, 96.83 %, y *loss*, 0.170, obtenido en test con la red entrenada en la epoch 89.

5.2.2. Real and Fake Face Detection

Modelos anteriores

Dado los buenos resultados alcanzados en los modelos propuestos anteriormente, se ha decidido reutilizarlos en este conjunto de datos con la intención de averiguar si alguna de las topologías previamente propuestas sería útil en el desarrollo de este problema. Además, dado que el *transfer learning* es un recurso muy utilizado en este ámbito, se investiga si la utilización de pesos preentrenados en la tarea anterior ayudaría a mejorar los resultados.

En este caso, los resultados obtenidos utilizando el recurso *transfer learning* no son tan favorables como los conseguidos realizando el entrenamiento desde cero. A pesar de que se consideren tareas con propiedades similares, el hecho que el *dataset* esté compuesto por fotografías editadas por humanos hace que las características aprendidas no sean de tanta utilidad. Este fue el motivo por el cual se desestimó la posibilidad de continuar esta línea de investigación.

Por otro lado, se decidió entrenar los modelos desde cero. Únicamente el segundo y el cuarto modelo consiguieron alcanzar un *accuracy* en test superior al 60 %. De entre ellos, fue el cuarto modelo el que obtuvo el mejor resultado, un 65.16 % de *accuracy* en test con los pesos de la red calculados en la epoch 60. La siguiente figura muestra su topología.

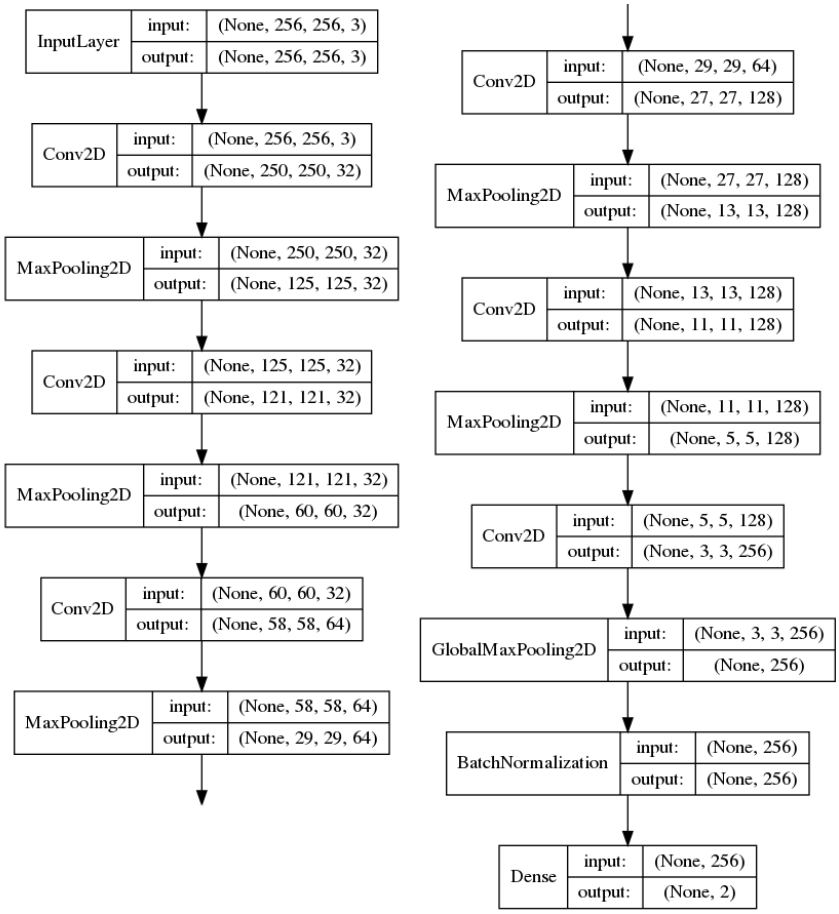

Figura 5.10: Topología del cuarto modelo.

En las gráficas expuestas a continuación, se puede observar como este modelo en torno al epoch 10 ya ha sido capaz de aprender la totalidad de las muestras de entrenamiento. Debido a ello, se evidencia un rápido crecimiento del *loss* en validación y una meseta de la *accuracy* en validación alrededor de la epoch 25.

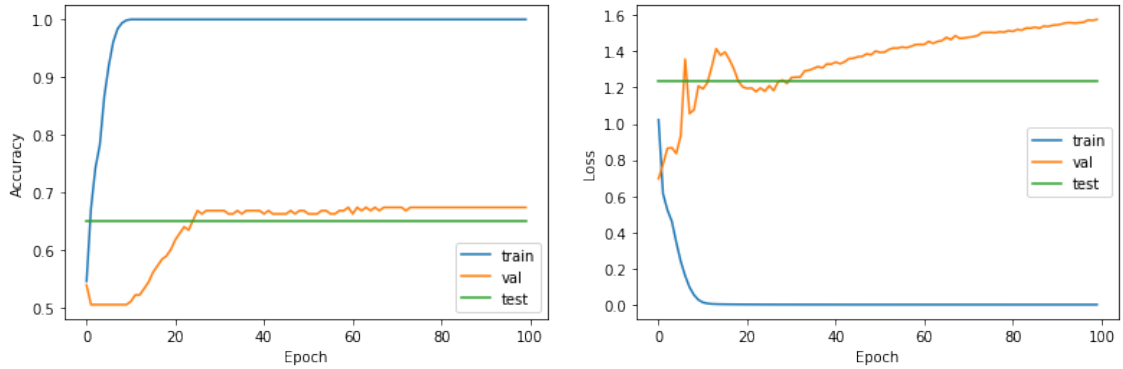

Figura 5.11: Gráficas mostrando la evolución del *accuracy* y *loss* del cuarto modelo. También se muestra el valor de *accuracy*, 65.16 %, y *loss*, 1.234, obtenido en test con la red entrenada en la epoch 60.

Séptimo modelo

El séptimo modelo presenta una característica diferente a los descritos previamente en lo referente a su estructura. En este sentido, su topología difiere del resto en el hecho de que se usan módulos con la finalidad de agrupar las características calculadas al inicio con aquellas calculadas al final. El objetivo de este análisis conjunto de características es mejorar el aprendizaje de las mismas.

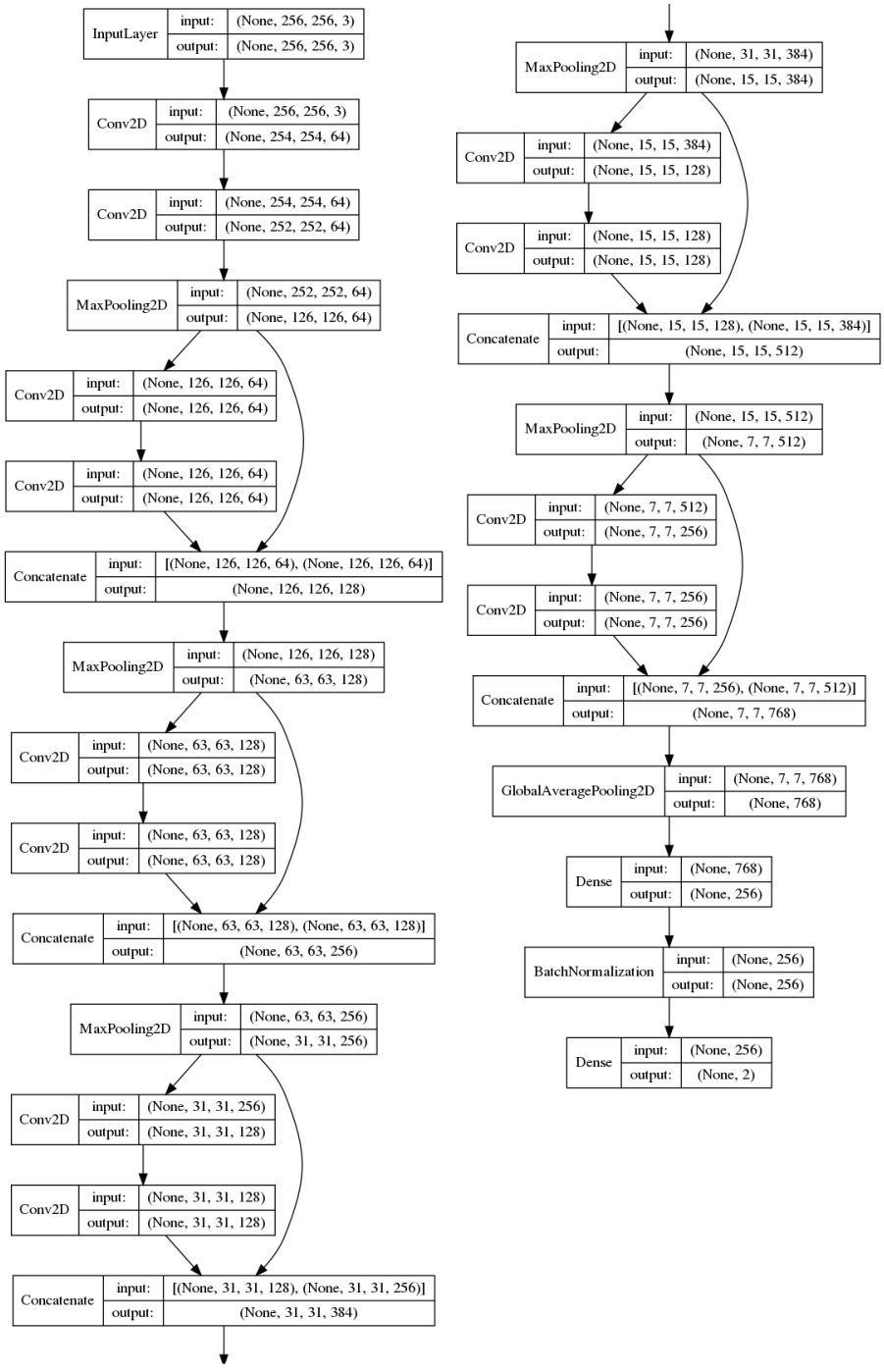

Figura 5.12: Topología del séptimo modelo.

De las siguientes gráficas se puede concluir que existe un aprendizaje progresivo en lo referente a las muestras de entrenamiento. De forma más detallada, resulta interesante destacar que a pesar de existir una gran variabilidad en el desarrollo del valor de validación éste incrementa con el tiempo. Lamentablemente, se obtiene un valor no satisfactorio de *loss* en validación.

A pesar de que se puede suponer que este modelo no se considere una solución aceptable, ya que consigue un *accuracy* en test de 62.53 % en la epoch 46, consigue un valor de *AUC* comparable al del cuarto modelo.

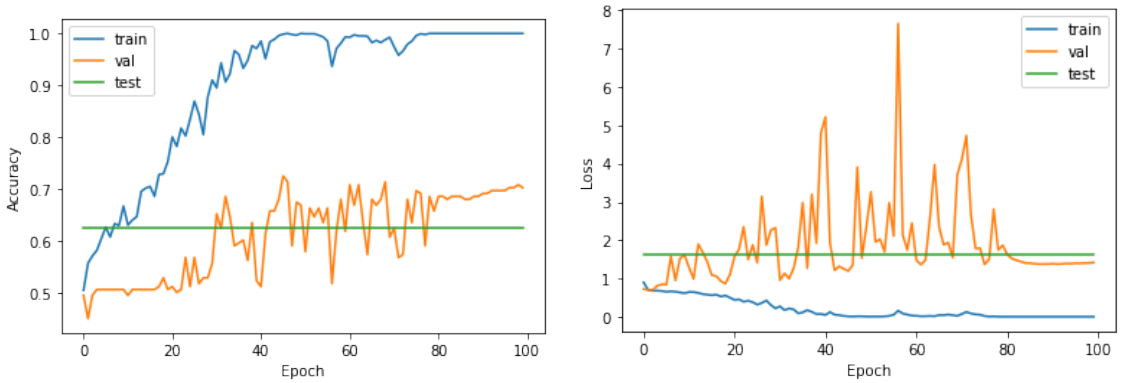

Figura 5.13: Gráficas mostrando la evolución del *accuracy* y *loss* del séptimo modelo. También se muestra el valor de *accuracy*, 62.53 %, y *loss*, 1.639, obtenido en test con la red entrenada en la epoch 46.

Octavo modelo

Para la creación de este último modelo se ha tomado como referencia el cuarto modelo. A éste se le ha añadido un cúmulo de convoluciones a las últimas capas con la intención de conseguir un mayor aprendizaje de características discriminatorias. En la Figura 5.14 se muestra la topología de este modelo.

Dada su similitud con el cuarto modelo, en las gráficas se evidencia que tanto el comportamiento como los valores son similares. El valor obtenido es 65.87 % de *accuracy* en test en la epoch 32.

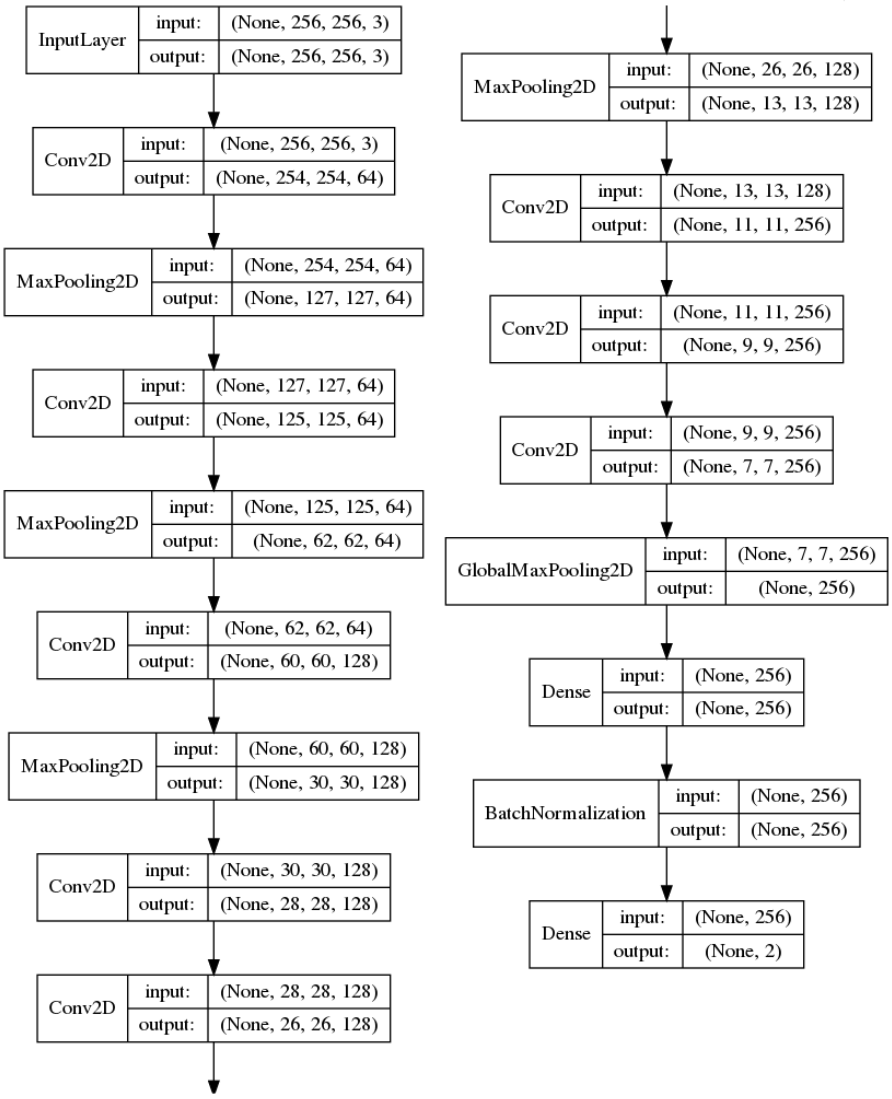

Figura 5.14: Topología del octavo modelo.

En lo referente al aprendizaje, cabe reseñar que en este modelo la curva aumenta de una forma más lenta. En este sentido, podría no considerarse relevante su aplicación, pero, como se observará en apartados posteriores, su rentabilidad se ve aumentada cuando se realiza una pequeña modificación.

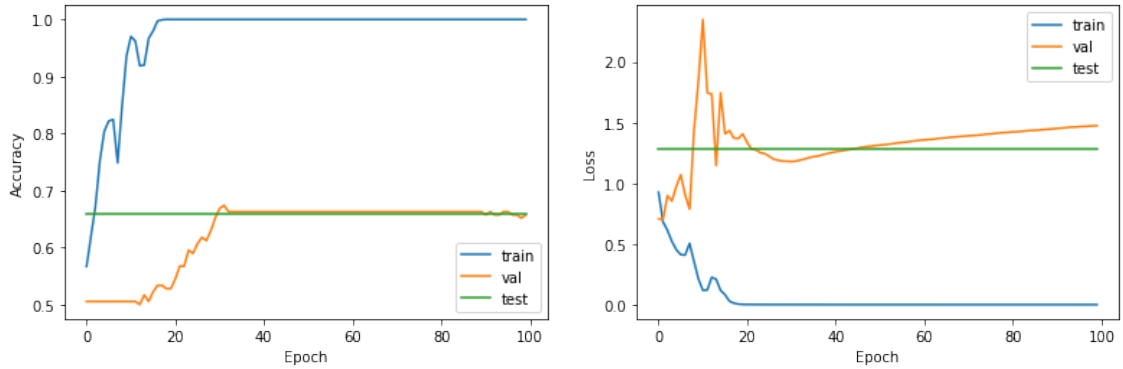

Figura 5.15: Gráficas mostrando la evolución del *accuracy* y *loss* del octavo modelo. También se muestra el valor de *accuracy*, 65.87 %, y *loss*, 1.279, obtenido en test con la red entrenada en la epoch 32.

Regularización mediante dropout

Después de realizar los distintos experimentos comentados previamente, se intentó reducir el sobre aprendizaje de los modelos observado en los experimentos anteriores. En este caso, se decidió añadir una *dropout layer* previa a la función *softmax*.

En consecuencia, los resultados obtenidos fueron favorables puesto que se consiguió aumentar su *accuracy*.

Según lo representado en las gráficas, y a pesar de encontrarse un pico aislado tan marcado, se observa un aprendizaje menos acentuado durante el entrenamiento. Es importante reseñar que dicha regularización dio lugar al mejor resultado obtenido durante los experimentos realizados en este conjunto. Se verificó que esta solución no era subóptima, ya que, también, obtuvo el resultado más alto de *AUC*.

En concreto, el octavo modelo incrementó su *accuracy* a 67.30 % en test con los pesos de la red calculados en la epoch 45.

Después de determinar que este modelo ofrece la mejor solución al problema, a continuación, comprobamos el porcentaje de muestras fáciles, intermedias y difíciles que consigue clasificar correctamente en el conjunto test. Los valores son 54.17 %, 66.67 % y 62.50 %, respectivamente. De forma sorprendente, el clasificador tiene más dificultades para reconocer las imágenes con peor calidad de edición.

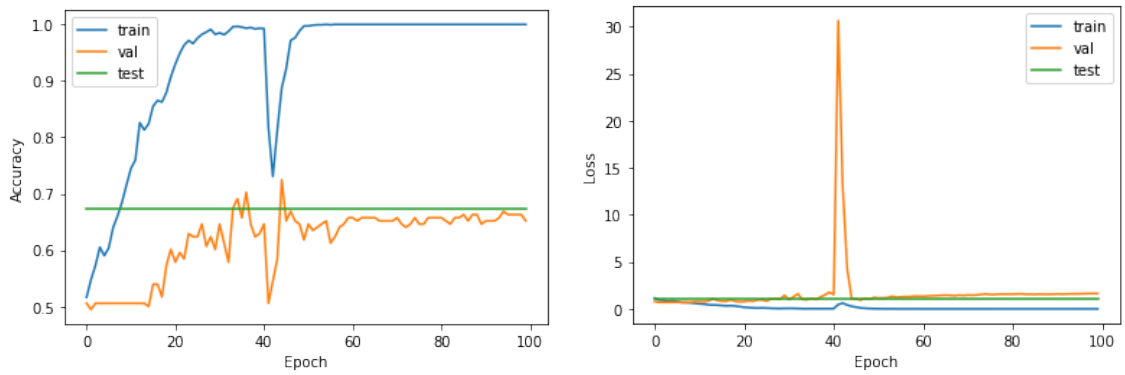

Figura 5.16: Gráficas mostrando la evolución del *accuracy* y *loss* del octavo modelo con *dropout*. También se muestra el valor de *accuracy*, 67.30 %, y *loss*, 1.116, obtenido en test con la red entrenada en la epoch 45.

5.2.3. Fakes and real news dataset

Antes de explicar de forma detallada los experimentos llevados a cabo con este conjunto de datos, cabe destacar que la longitud máxima de las noticias se estableció en 700 palabras. Se decidió optar por dicha longitud, puesto que, por un lado, permitía no perder demasiada información en las noticias más extensas y, por otro, evitaría que las noticias más cortas no fueran representadas en su mayoría por un vector de ceros. También, se escogió 150 como talla de los vectores de los *embeddings*.

Modelo A

En esta primera aproximación, se utiliza una red recurrente simplista, es decir, solamente una *embedding layer* y una *LSTM layer* con 512 unidades. La finalidad de este modelo es tanto analizar su comportamiento ante el conjunto de datos como investigar su capacidad de aprendizaje. Ante el hecho de que, normalmente, se utilizan embeddings preentrenados se ha optado por entrenarlos con el objetivo de averiguar si esto ayudaría en la tarea de clasificación. La topología del modelo A se muestra en la siguiente figura.

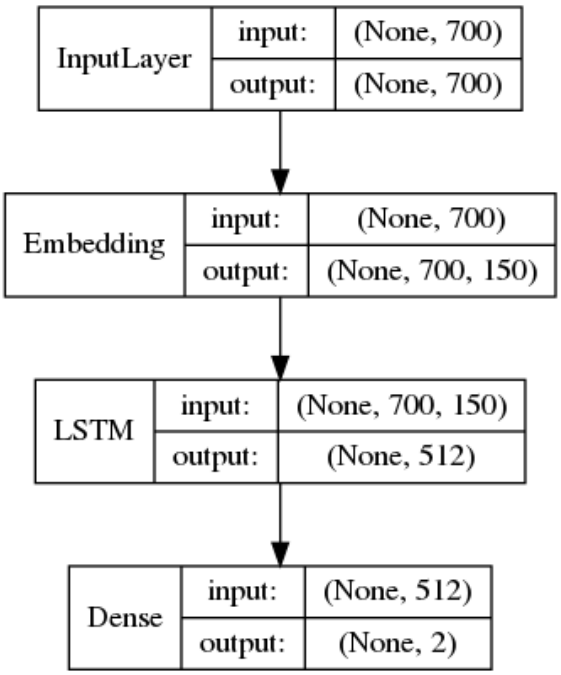

Figura 5.17: Topología del modelo A.

En la Figura 5.18, que se muestra a continuación, se puede observar que entorno a la epoch 10 la red ya ha aprendido a clasificar las muestras de entrenamiento. A partir de este momento, podemos afirmar que empieza a sobreaprender, debido a que el *loss* en validación comienza a incrementar de forma logarítmica. A pesar de ello, se puede comprobar que alcanza buenos resultados, puesto que tiene un *accuracy* en test del 99.29 % con los pesos de la red en la epoch 25.

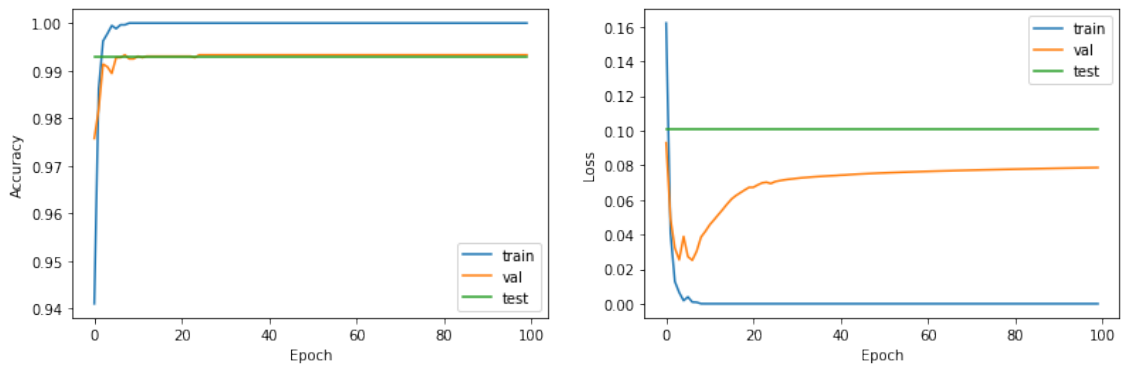

Figura 5.18: Gráficas mostrando la evolución del *accuracy* y *loss* del modelo A con una *LSTM layer* con 512 unidades. También se muestra el valor de *accuracy*, 99.29 %, y *loss*, 0.101, obtenido en test con la red entrenada en la epoch 25.

Variaciones del modelo A

Dados los resultados tan sorprendentes obtenidos en el modelo expuesto en el anterior apartado, a continuación, se investigará si simplificando dicho modelo se consiguen mantener buenos resultados. Se procederá a realizar una reducción progresiva de las unidades de la *LSTM layer* a 256, 128 y 64.

Para las reducciones de 128 y 64 unidades, las gráficas de aprendizaje son muy similares a la obtenida con la configuración inicial. Se vuelve a poner de manifiesto que las muestras de entrenamiento aprenden prematuramente. Esto se traduce en que, en validación se congela el crecimiento del *accuracy* y el *loss* tienda a aumentar su valor a lo largo de las epochs. Resulta interesante resaltar el hecho de que a pesar de que los resultados conseguidos se consideran subóptimos, no son malos. Dados los resultados obtenidos, se decide no profundizar más en estos modelos.

Por otro lado, cabe resaltar la configuración con 256 unidades. Se puede observar a partir de las siguientes gráficas que se ha obtenido un aprendizaje más progresivo. Aun así, alrededor de la epoch 50 se evidencia que la red empieza a sobreajustar los pesos. Este hecho queda reflejado con el aumento de loss en validación.

Considerando los datos mostrados para esta topología, se puede concluir que el valor de unidades más adecuado para la *LSTM layer* es 256. Ha obtenido un *accuracy* de 99.20 % en test con los pesos de la red calculados en la epoch 42. En términos de *accuracy* y *AUC*, la diferencia con el modelo A con 512 unidades es despreciable.

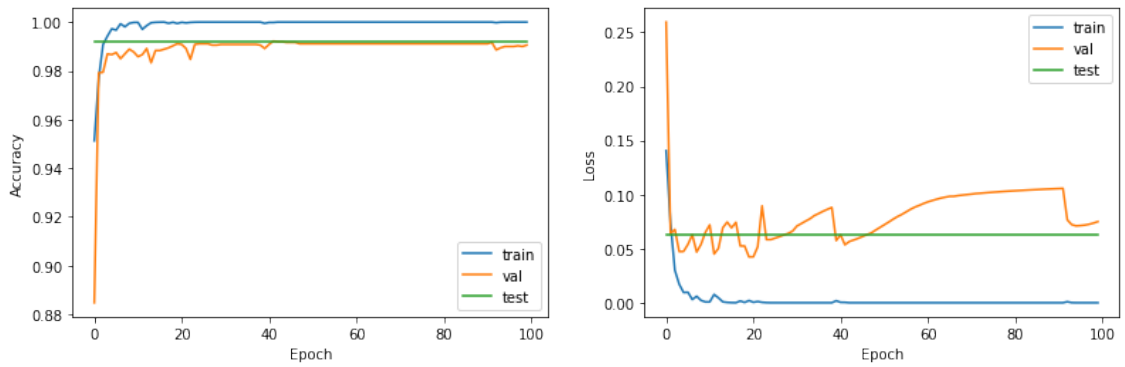

Figura 5.19: Gráficas mostrando la evolución del *accuracy* y *loss* del modelo A con una *LSTM layer* con 256 unidades. También se muestra el valor de *accuracy*, 99.20 %, y *loss*, 0.063, obtenido en test con la red entrenada en la epoch 42.

Subdivisión de noticias

En este experimento, además de aplicarse el preprocesamiento detallado en el apartado 4.1.2 se reduce la entrada de la red a muestras de un máximo de 100 palabras. El objetivo que se pretende alcanzar es conocer si analizando las distintas partes de una noticia obtenemos más información sobre su veracidad que analizando la mayor parte o la totalidad de la noticia como una sola muestra.

En las siguientes gráficas se muestran los resultados tanto de *accuracy* como de *loss*. En este caso, la red no memoriza de forma tan rápida las muestras si se compara con el preprocesado inicial, puesto que consiguen memorizar el conjunto de entrenamiento en la epoch 30 y epoch 10, respectivamente. En cambio, la *accuracy* de validación y test se queda alrededor de 79 % a partir de la epoch 20. En lo referente al valor de *loss*, su evolución en entrenamiento se aproxima a 0 a medida que éste avanza. Por otro lado, el valor de *loss* en validación continúa aumentando según aumenta el número de epochs.

Con lo expuesto anteriormente, se puede concluir que se produce un sobreentrenamiento en la red. Por este motivo, sería interesante realizar experimentos con el *learning rate* y con la introducción de *dropout layers*.

En una fase posterior, se unen las predicciones realizados sobre los subtextos con la intención de clasificar una noticia como verdadera o falsa. Para poder efectuar dicha distinción, se ha de calcular el valor promedio de las predicciones de los subtextos. Una vez realizado este cálculo, el resultado obtenido es un *accuracy* de 61.03 % para clasificar las noticias correctamente.

Tras haber obtenido estos resultados se considera que no es rentable continuar esta línea de investigación.

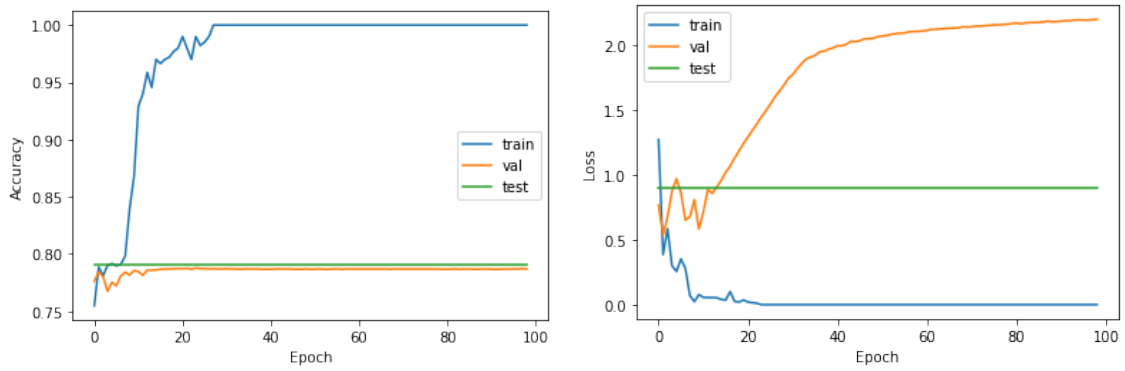

Figura 5.20: Gráficas mostrando la evolución del *accuracy* y *loss* del modelo A subdividiendo las noticias. También se muestra el valor de *accuracy*, 79.1 %, y *loss*, 0.927, obtenido en test con la red entrenada en la epoch 24.

Modelo B

En este tipo de redes se ha observado que el coste computacional es muy alto. Como consecuencia de ello, estas redes tardan mucho tiempo en entrenar. Dado este inconveniente, se opta por aumentar la complejidad de la red, apilando *LSTM layers* y reduciendo las unidades de las mismas. Mediante esta modificación, se consigue captar características más complejas para facilitar la clasificación. En la figura mostrada a continuación se refleja la topología de esta red.

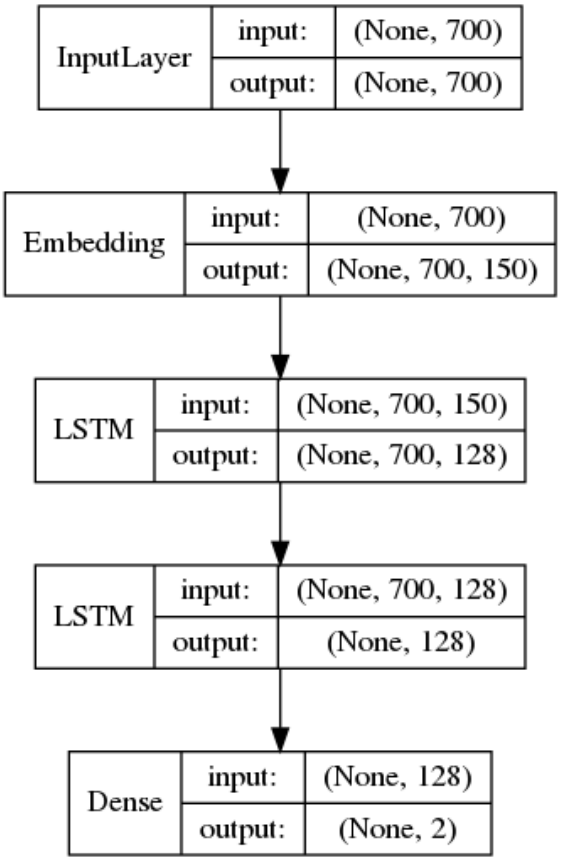

Figura 5.21: Topología del modelo B.

En las siguientes dos gráficas se pone de manifiesto que el modelo B supera al modelo A en lo referente a la capacidad de aprendizaje de características más complejas. Asimismo, a través de los valores de *accuracy* y *loss* se demuestra que esta red realiza un mejor aprendizaje. Concretamente, cabe destacar que los valores de *accuracy* y *loss* en entrenamiento convergen de una manera más lenta a 1 y 0, respectivamente. Es importante mencionar que la red obtiene un valor de 99.64 % de *accuracy* en test con los pesos de la red calculados en la epoch 87.

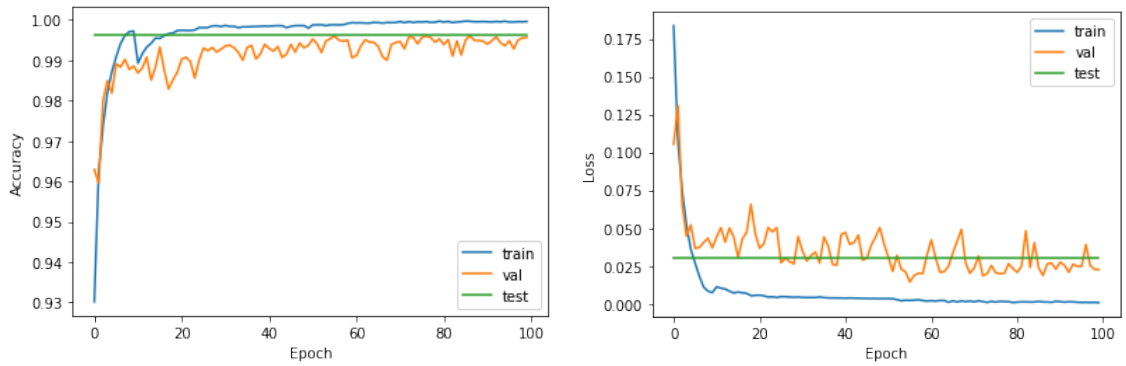

Figura 5.22: Gráficas mostrando la evolución del *accuracy* y *loss* del modelo B. También se muestra el valor de *accuracy*, 99.64 %, y *loss*, 0.033, obtenido en test con la red entrenada en la epoch 57.

Modelos C y D

Con el objetivo de mejorar los resultados obtenidos por los modelos anteriores, se decide introducir un nuevo cambio. Este consiste en añadir, en los modelos A y B, una *fully-connected layer* entre las *LSTM layers* y la *softmax layer*. Los resultados obtenidos no mejoraron los alcanzados con los modelos originales y sólo añadían más parámetros a los modelos, motivo por el cual se decidió no continuar con esta línea de investigación.

5.2.4. Liar-dataset

Previo a profundizar en los experimentos realizados sobre este conjunto de datos, es importante tener en cuenta que la longitud máxima de las declaraciones se estableció en 50 palabras. Esta fue la longitud elegida, debido a que evitaría que las declaraciones de corta extensión no fueran representadas en su mayoría por un vector de ceros. También, evitaría perder demasiada información en las declaraciones más extensas. Además, se eligió 150 como talla de los vectores de los *embeddings*.

Modelos anteriores

Dado los buenos resultados obtenidos por los modelos propuestos en los aparatados anteriores, se ha decidido optar por un modelo simplista e iniciar la experimentación con el modelo A y sus distintas variaciones.

Las gráficas representadas a continuación ponen de manifiesto la evolución del entrenamiento de las muestras, en concreto del modelo A con 128 unidades, puesto que, de todos los experimentos realizados, es el que mejor resultado obtiene en test. En ellas, destaca como la red sufre un enorme sobreajuste de sus parámetros a lo largo de las epochs, impidiendo el aprendizaje de la red. A su vez, el valor de *accuracy* en validación se mantiene constante por debajo del 60 % e incluso llega a disminuir, de forma muy lenta, conforme avanza el entrenamiento. Además, el valor de *loss* en validación es muy alto durante la totalidad del entrenamiento.

Desafortunadamente, los resultados obtenidos en la totalidad de los experimentos realizados obtienen resultados muy similares e insatisfactorios.

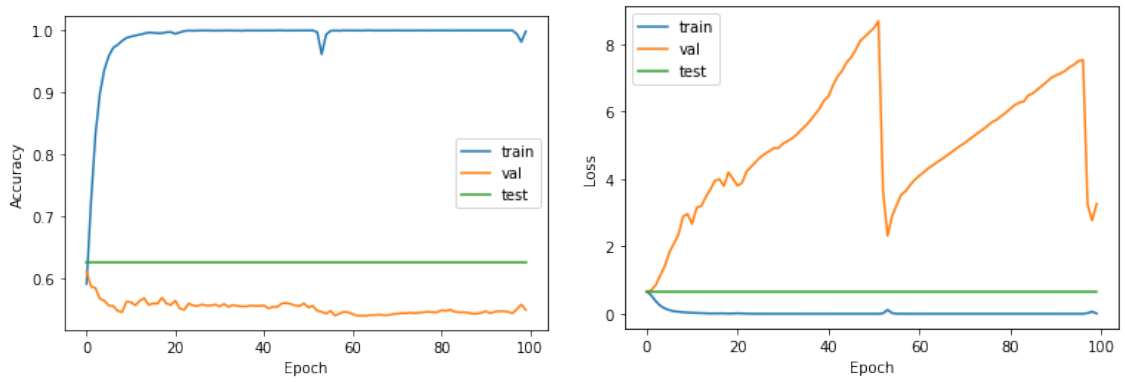

Figura 5.23: Gráficas mostrando la evolución del *accuracy* y *loss* del modelo A con 128 unidades. También se muestra el valor de *accuracy*, 62.51 %, y *loss*, 0.6519, obtenido en test con la red entrenada en la epoch 1.

Modificaciones realizadas

Debido al comportamiento de la red, se deciden ajustar los hiperparámetros de la misma e introducir una regulación mediante *dropout*.

En la configuración inicial se utiliza el optimizador Adam con un learning rate de 0.001. Dado que no se consiguen mejorar los resultados, se reduce el *learning rate* a 0.00001 con el fin de mejorar el aprendizaje. Con esta modificación, se observa una disminución del aprendizaje en entrenamiento, ya que el valor de *accuracy* tanto en entrenamiento como en validación se mantiene constante a lo largo del mismo. Debido al hecho de que no es recomendable utilizar *dropout* con *LSTM layers*, se decide utilizar un valor muy bajo de *dropout*, en concreto del 15 %. Aun así, esta solución no obtiene resultados satisfactorios.

A pesar de que el optimizador Adam posee un alto rendimiento, ante los resultados obtenidos, se decide cambiarlo por el optimizador SGD y probar distintos *learning rates*. A través de esta modificicación, se pretende conseguir un aprendizaje más lento y gradual. Lamentablemente, los resultados obtenidos son similares a los comentados anteriormente.

A continuación, con la intención de averiguar el motivo de este comportamiento se decide realizar una revisión de los datos de entrada. Se realizan pequeñas modificaciones que tienen por objetivo homogeneizar más el texto evitando que se pierda información. Además, se realiza una corrección ortográfica de las palabras y se decide reducir la longitud máxima de las declaraciones a 30 palabras.

La Figura 5.24 muestra el resultado obtenido tras haber realizado las modificaciones explicadas anteriormente en la variante de 256 unidades del modelo A. Ponen de manifiesto un mejor comportamiento de la red. Asimismo, se consigue un mayor aprendizaje durante las epochs, puesto que el valor de *accuracy* y de *loss* en validación van en aumento y disminución, respectivamente. A pesar de ello, los resultados conseguidos no han sido satisfactorios puesto que el valor de recall obtenido es de 0.06. Se puede concluir que esta red no hace bien su tarea, por lo que se decide descartar como una posible solución factible.

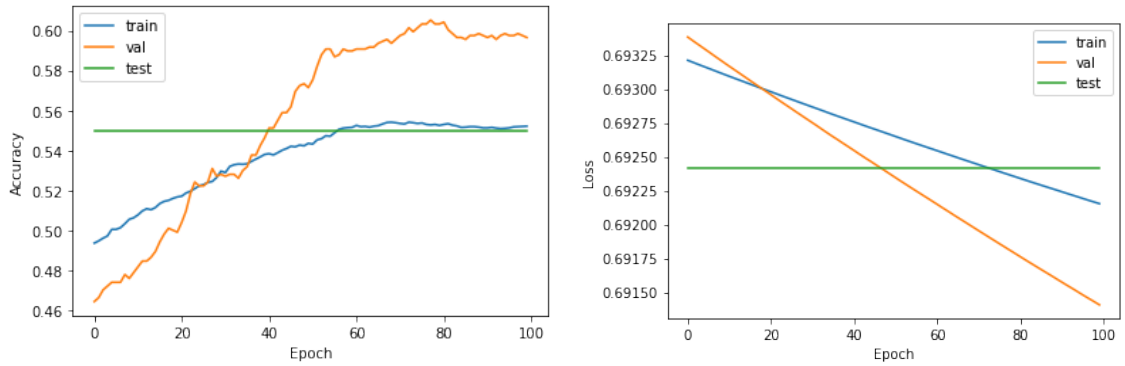

Figura 5.24: Gráficas mostrando la evolución del *accuracy* y *loss* del modelo A con 256 unidades. También se muestra el valor de *accuracy*, 54.99 %, y *loss*, 0.69241, obtenido en test con la red entrenada en la epoch 78.

En definitiva, tras haber desarrollado varios experimentos, sin obtener un resultado factible en ninguna de ellas, se puede indicar que el modelo A de 128 unidades es que el mejor resultado ha obtenido en esta tarea.

5.3 Comparación con el estado del arte

A continuación, se realizará una comparación de los mejores modelos de la experimentación con los resultados del estado del arte. Los datos de comparación se exponen en las siguientes tablas

Tabla 5.1: Comparación de resultados del conjunto *140k Real and Fake Faces* con el estado del arte.

	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	AUC (%)
modelo 2	99.54	99.00	100.00	100.00	99.54
modelo 6	99.93	99.00	100.00	99.00	99.93
Vidhi Gupta [12]	95.00	99.00	91.00	95.00	99.74

Como se puede observar en la Tabla 5.1, los resultados obtenidos a partir de los experimentos realizados han demostrado ser superiores a los resultados disponibles en la actualidad. En términos de *accuracy*, es importante resaltar que se ha conseguido un incremento de casi 5 puntos porcentuales, llegando a 99.93 % dede 95 %.

No obstante, si evaluamos los modelos utilizando la métrica *AUC*, se comprueba que el rendimiento de los tres modelos es similar.

Tabla 5.2: Comparación de resultados del conjunto *Real and Fake Face Detection* con el estado del arte.

	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	AUC (%)
modelo 4	65.16	67.00	70.00	68.00	64.75
modelo 8	65.87	69.00	71.00	70.00	64.73
modelo 8 (Dropout)	67.30	69.00	71.00	70.00	66.93
Anastasia S [29]	61.00	61.00	51.00	55.00	-

Como se puede observar en la Tabla 5.2, los mejores resultados alcanzados en este *dataset* son superiores a los resultados obtenidos en el estado del arte. Comparando los valores de *accuracy* obtenidos, el modelo 8 con *dropout* mejora en 6 puntos porcentuales los resultados actuales.

A pesar de que los resultados obtenidos no consiguen el *accuracy* esperado, se podría considerar un resultado satisfactorio debido al hecho de que la investigación en esta tarea es escasa.

Tabla 5.3: Comparación de resultados del conjunto *Fake and real news dataset* con el estado del arte.

	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	AUC (%)
modelo A (256)	99.20	99.00	99.00	99.00	99.21
modelo B	99.64	99.00	100.00	100.00	99.49
Aditya Mahaddalkar [20]	99.84	-	-	-	-
Rahul Bana [4]	99.80	100.00	100.00	100.00	-

Como se puede observar en la Tabla 5.3, los resultados del estado del arte, en términos de *accuracy*, también son casi perfectos. Cabe destacar que los resultados del estado del arte son superiores, pero esta diferencia es menor al 1 %. Ahora bien, es importante recalcar el hecho de que los modelos creados durante la experimentación tienen una menor complejidad. Esto resulta en que el coste computacional al entrenar es menor. Por ello, se pueden considerar satisfactorios los resultados obtenidos, ya que ofrecen unos resultados muy similares a un menor coste.

Tabla 5.4: Comparación de resultados del conjunto *Liar dataset* con el estado del arte.

	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	AUC (%)
Modelo A (128)	62.19	64.00	77.00	70.00	60.46
Rashkin [27]	56.00	-	-	-	-
Pham [26]	74.40	74.0	73.7	73.8	-
Bhattacharjee [5]	96.17	-	-	-	-

En esta última tabla, la Tabla 5.4, se comparan los resultados del conjunto *Liar dataset* con el estado del arte.

En primer lugar, cabe reseñar que ha sido difícil encontrar experimentación que afronte el problema únicamente subdividido en dos clases, puesto que esto no es la tarea original del conjunto. Aun así, se han encontrado varios trabajos con los que comparar los resultados conseguidos.

De esta manera, comparando los resultados obtenidos con los alcanzados por [27], se han conseguido incrementar los resultados en un 6 %. Lamentablemente, estos resultados no se consideran exitosos, puesto que en otros trabajos se consiguen mejores resultados utilizando técnicas más sofisticadas.

CAPÍTULO 6
Conclusiones

Tras el análisis de los resultados obtenidos en los diferentes experimentos realizados es importante destacar que se han conseguido alcanzar los objetivos propuestos.

En primer lugar, en 3 de los 4 datasets se han alcanzado resultados satisfactorios. De esta manera, es importante indicar que los modelos diseñados han sido capaces de detectar si las noticias o las imágenes son falsas con resultados similares o superiores a los que se obtienen con el estado del arte.

En segundo lugar, al profundizar en las técnicas utilizadas para la detección de información falsa, se ha visto incrementada, de forma exponencial, tanto la compresión de estas técnicas como la manera de utilizarlas de una forma efectiva. De forma más concreta, se ha aprendido a trabajar con redes neuronales utilizando el *framework* de Keras junto a Tensorflow.

En paralelo, el hecho de haber elegido este tema me ha permitido profundizar en el campo del *deep learning*, tecnología en auge y con un gran potencial, puesto que durante el curso académico únicamente se da una visión muy general de la misma.

Por otro lado, una de las dificultades contempladas durante la fase de experimentación ha sido trabajar con conjuntos de datos con pocas muestras. Si comparamos los resultados obtenidos, éste ha sido un factor clave el cual ha permitido un buen aprendizaje en nuestros modelos.

6.1 Imágenes Faciales Falsas

Resulta interesante incidir en el hecho de que el *transfer learning* es un recurso muy útil en lo referente a la detección de imágenes falsas. En consecuencia, permite que se obtengan buenos resultados de una forma rápida y sencilla cuando se utiliza una red potente para el desarrollo de los experimentos.

Cabe destacar que cuando se emplean conjuntos de datos de menor tamaño se recomienda ofrecer una solución adaptada para ese problema. Además, es importante resaltar que es necesario introducir métodos de regularización, tales como *data augmentation* o *dropout*, con el objetivo de evitar el sobreentrenamiento de la red.

A raíz de los experimentos realizados y los resultados obtenidos en el ámbito de la detección de imágenes falsas, se proponen diferentes situaciones, que podría resultar interesante desarrollarlas en futuros proyectos, con la intención de mejorar los resultados actuales.

En primer lugar, se debería investigar el motivo por el cual se obtiene una baja tasa de acierto cuando las imágenes editadas por humanos son fáciles de reconocer.

En segundo lugar, se podría considerar ampliar el conjunto de *Real and Fake Face Detection* con otro conjunto de imágenes editadas de forma profesional o a través de técnicas de *data augmentation*.

En tercer lugar, se propone comprobar el rendimiento de nuestras soluciones para la detección de caras de GANs con un nuevo conjunto de caras generadas por StyleGAN2 [18]. El objetivo de ello sería confirmar si se mantienen los buenos resultados alcanzados.

Por último, se sugiere la posibilidad de abrir otras líneas de investigación con la intención de generalizar la detección de las imágenes faciales falsas. Se realiza dicha propuesta ante la dificultad de mantener una tasa de resultado exitoso. Esto es debido a que cada año se publican nuevos GANs, los cuales generan imágenes capaces de engañar a las técnicas actuales. Por tanto, los métodos de detección tienen una vida media corta en lo referente a su obsolescencia. Además, se ha visto que no resulta factible generar muestras de un nuevo generador y reentrenar la red cada vez que esto suceda.

6.2 Noticias Falsas

Dentro del ámbito de la detección de noticias falsas, es importante incidir en el hecho de que disponer de un conjunto de datos lo suficientemente extenso permite ofrecer mejores soluciones, de una forma sencilla y rápida. También, se ha observado que entrenar los *embeddings* no ofrece mejores resultados, solo incluye un mayor número de parámetros a entrenar. Debido a esto, se recomienda utilizar los modelos preentrenados disponibles.

En paralelo, durante los experimentos realizados sobre el conjunto *Fake and real news dataset*, también se ha podido comprobar que, en cuanto a la detección de características discriminatorias, se obtienen resultados más satisfactorios cuando las secuencias son de mayor longitud. Esto evita la pérdida de contextualización de la información y facilita la detección de información falsa.

Por otro lado, aunque sólo se han conseguido igualar los resultados obtenidos a partir de los modelos expresados, para el conjunto *Fake and real news dataset*, en comparación con los del estado del arte, se considera fundamental resaltar que se han obtenido con una solución menos compleja y con un menor coste computacional.

Por otra parte, se ha intentado investigar el rendimiento del *Liar dataset* analizando únicamente el texto de las noticias. Tras la experimentación y la comparación de los resultados conseguidos con los del estado del arte, se puede concluir que éste es un enfoque que no ha sido capaz de ofrecer buenos resultados. No obstante, revisando la literatura disponible del estado del arte se pone de manifiesto que, utilizando metadatos de las noticias, técnicas más sofisticadas y ampliando el conjunto de datos se consigue un mejor rendimiento.

En último lugar, y dado la escasa efectividad de los resultados conseguidos, se plantea, para futuras investigaciones, la posibilidad de crear un nuevo *dataset* con un número mayor de muestras. Éstas se podrían extraer de la página web *politifact.com*. En este sentido, podría suponer una mejora en los resultados.

Bibliografía

[1] Deepfakes github. https://github.com/deepfakes/faceswap.

[2] Fake news challenge. http://www.fakenewschallenge.org/, 2016.

[3] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[4] Rahul Bana. Fake, true news classification glove embedding. https://www.kaggle.com/rahulbana/fake-true-news-classification-glove-embedding, 2020.

[5] Sreyasee Das Bhattacharjee, Ashit Talukder, and Bala Venkatram Balantrapu. Active learning based news veracity detection with feature weighting and deep-shallow fusion. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 556–565. IEEE, 2017.

[6] Clément Bisaillon. Fake and real news dataset. https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset, 2015.

[7] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation, 2017.

[8] François Chollet et al. Keras. https://keras.io, 2015.

[9] Hao Dang, Feng Liu, Joel Stehouwer, Xiaoming Liu, and Anil Jain. On the detection of digital face manipulation, 2019.

[10] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[11] Luca Guarnera, Oliver Giudice, and Sebastiano Battiato. Deepfake detection by analyzing convolutional traces, 2020.

[12] Vidhi Gupta. Grayscale-densenet. https://www.kaggle.com/vidhi27/grayscale-densenet, 2020.

[13] Naeemul Hassan, Fatma Arslan, Chengkai Li, and Mark Tremayne. Toward automated fact-checking: Detecting check-worthy factual claims by claimbuster. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1803–1812, 2017.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks, 2016.

[15] Hunter Heidenreich. What is a generative adversarial network? https://towardsdatascience.com/what-is-a-generative-adversarial-network-76898dd7ea65, 2018.

[16] Computation Intelligence and Yonsei University Photography Lab. Real and fake face detection. https://www.kaggle.com/ciplab/real-and-fake-face-detection, 2015.

[17] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2018.

[18] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan, 2019.

[19] Marek Kowalski. Faceswap github. https://github.com/MarekKowalski/FaceSwap/.

[20] Aditya Mahaddalkar. Fake news predictor based on simple lstm model. https://www.kaggle.com/adityam1311/fake-news-predictor-based-on-simple-lstm-model, 2020.

[21] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013.

[22] Huaxiao Mo, Bolin Chen, and Weiqi Luo. Fake faces identification via convolutional neural network. In *Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec '18, page 43–47, New York, NY, USA, 2018. Association for Computing Machinery.

[23] Christopher Olah. Understanding lstm networks. https://colah.github.io/posts/2015-08-Understanding-LSTMs, 2015.

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[25] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[26] Trung Tin Pham. A study on deep learning for fake news detection. http://hdl.handle.net/10119/15196, 2018.

[27] Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. Truth of varying shades: Analyzing language in fake news and political fact-checking. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 2931–2937, 2017.

[28] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge, 2014.

[29] Anastasia S. Face anti-spoofing. https://www.kaggle.com/anastasia484/face-anti-spoofing, 2019.

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.

[31] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2387–2395, 2016.

[32] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures, 2019.

[33] William Yang Wang. "liar, liar pants on fire": A new benchmark dataset for fake news detection. *CoRR*, abs/1705.00648, 2017.

[34] xhlulu. 140k real and fake faces. https://www.kaggle.com/xhlulu/140k-real-and-fake-faces, 2015.