Universidad de Sevilla

Máster Universitario en Lógica, Computación e Inteligencia Artificial

Trabajo de Fin de Máster

Detección de accidentes de tráfico en vídeo CCTV

Departamento de Ciencias de la Computación e Inteligencia Artificial

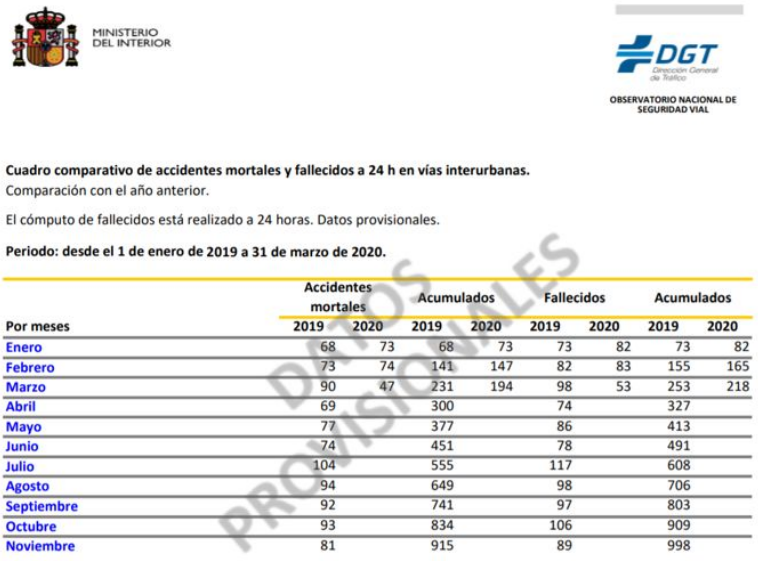
Javier Ivar Advani Aguilar

Tutorizado por
Miguel Ángel Martínez del Amor
Ignacio Pérez Hurtado

Sevilla, Diciembre 2020

Introducción

A pesar de los sistemas de seguridad que incorporan los vehículos en la actualidad, así como la señalización y el buen estado de la mayoría de carreteras y vías estatales, la Dirección General de Tráfico destaca que los accidentes de circulación siguen siendo siendo una de las principales causas de mortalidad entre los jóvenes [1]. Además, durante la crisis sanitaria del SaRs Cov 2, pese a que el país se ha visto sumido en fases de confinamiento, la siniestralidad ha repuntado en algunos períodos más de un 20% por encima de lo usual [2].



MINISTERIO DEL INTERIOR

DGT Dirección General de Tráfico

OBSERVATORIO NACIONAL DE SEGURIDAD VIAL

Cuadro comparativo de accidentes mortales y fallecidos a 24 h en vías interurbanas.
Comparación con el año anterior.

El cómputo de fallecidos está realizado a 24 horas. Datos provisionales.

Periodo: desde el 1 de enero de 2019 a 31 de marzo de 2020.

DATOS PROVISIONALES

	Accidentes mortales		Acumulados		Fallecidos		Acumulados	
Por meses	2019	2020	2019	2020	2019	2020	2019	2020
Enero	68	73	68	73	73	82	73	82
Febrero	73	74	141	147	82	83	155	165
Marzo	90	47	231	194	98	53	253	218
Abril	69		300		74		327	
Mayo	77		377		86		413	
Junio	74		451		78		491	
Julio	104		555		117		608	
Agosto	94		649		98		706	
Septiembre	92		741		97		803	
Octubre	93		834		106		909	
Noviembre	81		915		89		998	

Figura 1. Tal y como se muestra en los índices recogidos por la DGT hasta la fecha, la mortalidad en carretera no ha descendido significativamente durante el mes de marzo a pesar de que el país se encontraba sumido en un estado de confinamiento. Aunque aún no han revelado el balance anual, las noticias citadas anteriormente auguran que el resultado es mucho peor de lo que se esperaba, aún existiendo restricciones de movilidad.

Detectar y asistir a aquellas personas que sufren un siniestro utilizando técnicas de procesado de imágenes por computador (Visión Artificial) e Inteligencia Artificial ante este tipo de circunstancias puede resultar crítico para salvar sus vidas.

Los informes más recientes confirman que hay 225 cámaras ubicadas en la red española de carreteras (pueden consultarse en tiempo real a través de este enlace), en zonas donde se concentra una mayor siniestralidad. Aportar una interpretación *inteligente* de las imágenes que captan haría de las vías interurbanas un entorno aún más seguro.

Una de las primeras limitaciones que se presenta al plantear cualquier solución de este tipo, consiste en tener claro de dónde podríamos captar una cantidad suficiente de datos para que la *máquina* aprenda a detectarlos con una precisión aceptable. De acuerdo con lo indicado por la Agencia Española de Protección de Datos en su Guía para la Videovigilancia:

“Como regla general, la captación de imágenes con fines de seguridad de la vía pública debe realizarse por las Fuerzas y Cuerpos de Seguridad, ya que les corresponde la prevención de hechos delictivos y la garantía de la seguridad en la citada vía pública, de conformidad con lo regulado por Ley Orgánica 4/1997, de 4 de agosto, y su Reglamento de desarrollo”.

A esto se une que, nuestro foco de interés va a encontrarse en secuencias de vídeo de contenido explícito, en ocasiones con heridos de gravedad o fallecidos, y tanto las personas como familiares de éstos pueden reservarse el derecho de no querer que esas imágenes sean divulgadas.

No sin ello, en otros países este tipo de normas no se imponen de manera ortodoxa. Además, la popularización de plataformas de vídeo, como YouTube facilitan, a través de vídeos recopilatorios, la obtención de imágenes de accidentes de tráfico (uno de los conjuntos de datos utilizado provendrá de extractos descargados de esta plataforma).

Así, este trabajo tiene como fin recoger un conjunto de pruebas y técnicas que que aporten una manera efectiva de procesar secuencias de vídeo y detectar si en éstas acontece un accidente de tráfico o no.

Trabajos previos relacionados

El notable avance en el ámbito de los coches autónomos y smart cities durante estos últimos años, ha hecho que se elaboren diversas investigaciones que versan, en parte, en áreas de interés de este proyecto, combinando técnicas de visión e inteligencia artificial:

- Monitorización del flujo de tráfico en carreteras [3]. A través de técnicas de visión artificial, se revisan diversos métodos para detectar elementos en movimiento y seguirlos en escena. La combinación de sistemas de sensorización activa en los vehículos como RADAR o LIDAR permiten obtener resultados muy precisos. En cuanto al procesado de imágenes puro, dejando de lado el contexto, se hace alusión a diversas técnicas de segmentación[1]. En este trabajo se presentará una alternativa más eficaz para procesar largas secuencias de vídeo sin necesidad de analizar cada fotograma minuciosamente, obteniendo todas las ROIs[2] de una secuencia de vídeo sin un coste computacional elevado.

1 La **segmentación** en visión artificial es el proceso de dividir una imagen digital en varias partes (grupos de píxeles) u objetos. El objetivo de la segmentación es simplificar y/o cambiar la representación de una imagen en otra más significativa y más fácil de analizar. La segmentación se usa tanto para localizar objetos como para encontrar los límites de estos dentro de una imagen. Más precisamente, la segmentación de la imagen es el proceso de asignación de una etiqueta a cada píxel de la imagen de forma que los píxeles que compartan la misma etiqueta también tendrán ciertas características visuales similares.
2. ROI son las siglas en inglés para denotar una región de interés (*Region of Interest*) en procesado de imágenes, aquella zona o zonas de la imagen que son objeto de estudio.

- Detección y seguimiento de peatones [4]. Algunos de los fenómenos que acontecen al estudiar grandes cantidades de imágenes de viandantes vuelven a aparecer en el caso de los vehículos: saber hacia dónde se va a dirigir un elemento móvil en el siguiente fotograma, o afrontar impedimentos como el de oclusión, cuando parte del área de interés se encuentra oculta por otros elementos en escena, son revisados en profundidad con resultados muy satisfactorios. La diferencia sustancial que limita comparar estas técnicas desde el comienzo, se encuentra en la falta datos debidamente etiquetados sobre los que contrastar su uso.

- Predicción y detección de Accidentes de Tráfico. En los últimos años, han sido publicados algunos trabajos centrados en el uso de cámaras para detectar accidentes. Por ejemplo, [5] utiliza las cámaras que se colocan en el salpicadero del coche (*dashboard cameras*) para la predicción de accidentes. El trabajo citado, en especial, hace uso de una red recurrente que presta especial atención a las posiciones de los objetos en escena que cambian de posición en los frames consecutivos. A pesar de que los resultados resultan prometedores, es muy necesario que este conjunto de datos y estrategias sean mejorados, obteniendo una mayor precisión y mejor tiempo de reacción de los vehículos autónomos En el documento se afirma que los resultados arrojaron que el método anticipa el acontecimiento de los accidentes entre 1 y 2 segundos antes de que ocurran, con un 80% de *recall* y 56.14% de precisión[3].

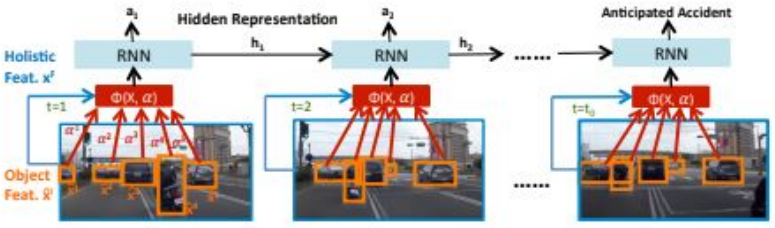


Figura 2. Funcionamiento de RNN para detectar accidentes en vídeos Dashcam [5]

El motivo que justifica la escasa efectividad de esta técnica, reside en el formato en que se reciben los datos: las imágenes provienen de cámaras que cambian de posición constantemente, debido al movimiento del vehículo y vibraciones, aún encontrándose en estado de reposo. Al carecer de buenas técnicas de estabilización, y, en ocasiones, teniendo una resolución insuficiente, referenciar el cambio de posición de los elementos en escena resulta a menudo difícil, y los resultados obtenidos no concuerdan por ello con lo esperado.

3. *Recall* $\frac{VP}{VP+FN}$ (exhaustividad) indica la proporción de verdaderos positivos que han sido identificados de entre todos los positivos. Precisión $\frac{VP}{VP+FP}$ aduce a la cantidad de positivos correctamente identificados

Otro de los trabajos a destacar, y que contribuye en parte con los datos que son utilizados en este estudio, recoge un conjunto de secuencias de vídeo CCTV [6] a los que se aplican diversas técnicas de detección de objetos (R-CNN, Improved faster R-CNN) y predicción de accidentes usando DSA-LSTM, muy similar a la red recurrente utilizada en el estudio anteriormente citado. Los accidentes logran predecirse entre 1.684 segundos y 3.078 segundos antes de que acontezcan. El *recall*, en los resultados, vuelve a rondar el 80%. Sin embargo, la precisión es sorprendentemente menor, de un 47%, a pesar de que ahora en este tipo de escenarios la cámara se encuentra fija.

Uno de los principales impedimentos que, argumentan en el texto, les frena al tratar de mejorar estas cifras, es que las secuencias de vídeo, en ocasiones de una longitud de más de 1000 frames, suelen tener lugar en los primeros 100 fotogramas. Idealmente, si se hiciesen pruebas más exhaustivas en un escenario real, defienden que los resultados mejorarían algo. Al fin y al cabo, la red LSTM (Long-Short Term Memory Network) observa la posición del objeto en los instantes anteriores para obtener una predicción de la siguiente. Careciendo de suficientes frames previos, la red no es capaz de retroalimentarse.

En un primer intento, el objetivo era tratar de replicar la implementación descrita en estas dos publicaciones, poniendo especial atención en CADP [6]. Sin embargo, esto no fue posible desde un primer intento, debido a que los autores originales no habían liberado el dataset[4] original ni las anotaciones hasta mucho después, tras contactar con ellos en numerosas ocasiones. La falta de datos para poder comenzar por esta vía, hizo que tomase la decisión de comenzar con otras líneas de estudio que, posteriormente, pudiesen ser incorporadas o incluso compararse en el proceso de codificación e implementación.

Primeros pasos: clasificación de vehículos dañados.

Tras una búsqueda exhaustiva en diversas fuentes, a fin de encontrar una alternativa con la que comenzar a hacer pruebas, se encontró este dataset de imágenes de vehículos dañados, un campo de amplio interés para compañías aseguradoras. Este dataset de imágenes (en realidad dos datasets), por una parte, tenía carpetas para determinar la severidad de un vehículo siniestrado, y además, otro conjunto de imágenes para determinar si un vehículo está accidentado o no, que fue el que centró toda atención en los primeros intentos por obtener un clasificador binario de accidentes.

Figura 3. El dataset contiene 2500 imágenes de imágenes que no son un accidente, y 2398 de vehículos siniestrados (hasta donde se ha revisado, son sólo coches lo que aparecen en el dataset).

4. Un *dataset* es una colección de datos, que es procesada aplicando técnicas computacionales. En un sentido amplio, dentro del ámbito de la Inteligencia Artificial, podría entenderse como una gran tabla de datos, en que cada fila corresponde a cada uno de los registros que se tienen, y las columnas son cada uno de los campos que los define (si por ejemplo, tenemos un conjunto de datos que defina personas, dos columnas podrían ser, por ejemplo, su edad o su sexo, y sus filas, cada uno de los individuos). En los casos que vamos a cubrir, nuestro conjunto de datos lo conforma un grupo de imágenes, y los campos que definen a cada imagen son sus píxeles.

La implementación escogida para esbozar los primeros resultados de la clasificación de vehículos, será una red neuronal convolucional (en adelante CNN, siglas del inglés *Convolutional Neural Network*). Una CNN es un tipo de Red Neuronal Artificial con aprendizaje supervisado que procesa sus capas imitando al córtex visual del ojo humano, con el fin de identificar distintas características de las imágenes de entrada que puedan identificar lo más unívocamente posible los tipos de objetos o escenas. Para ello, la CNN contiene varias capas (*layers*) que atañen a un propósito específico: esto quiere decir que ciertas capas, por ejemplo, se dedican enteramente a la detección líneas rectas, otras se encargan de procesar las curvas, etc, ... y se van especializando hasta llegar a capas más profundas que reconocen formas complejas, como un rostro, la silueta de un animal o, en el contexto actual, las ruedas de un coche.

Al ser las imágenes del dataset de una resolución de 28x28, la arquitectura de la red fue establecida de la siguiente manera:

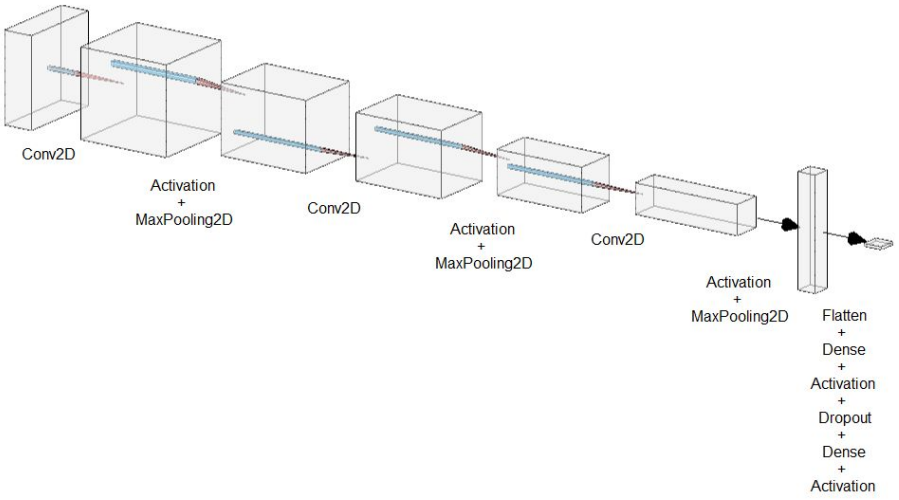


Figura 4. Arquitectura de la CNN utilizada para clasificar imágenes de vehículos dañados.

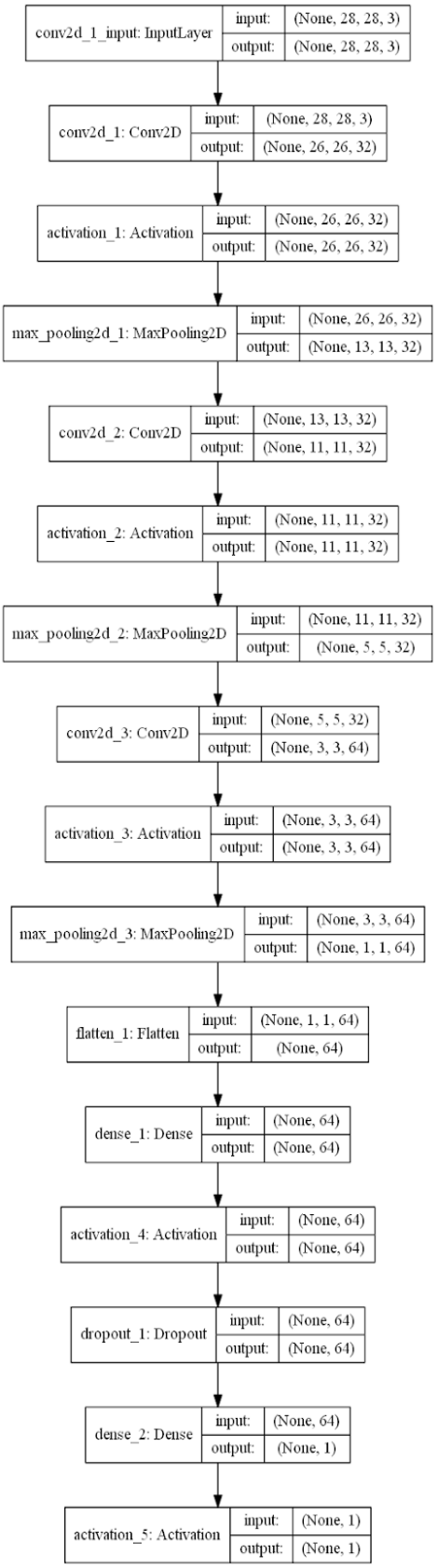


Figura 5. Configuración detallada de la arquitectura inicial utilizada, indicando el tamaño de la entrada y salida en cada una de las capas.

La densidad de cada una de las capas, así como las funciones de activación utilizada, fueron modificadas hasta obtener unos resultados aceptables con la configuración anteriormente mostrada. La función de Unidad Linear Rectificada (ReLU), a pesar de ser la más frecuentemente utilizada en arquitecturas convolucionales, debido a su sencillez computacional, tiene la desventaja de que descarta todos aquellos valores menores que 0 (matamos neuronas de la red, que al procesarse durante el proceso de *back-propagation*, van a ser pesos desactualizados: este fenómeno se conoce como el *dying-effect*) fue sustituida por su variante "sineReLU", que se caracteriza por ser una función sinusoidal para aquellos valores menores que 0.

$$\begin{cases} Z & Z > 0 \\ \varepsilon(sin(Z) - cos(Z)) & Z \leq 0 \end{cases}$$

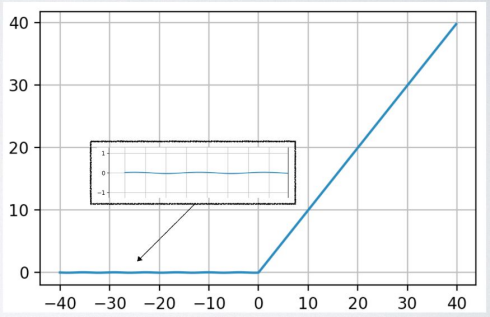


Figura 6. Expresión y representación gráfica de la función de activación sineReLU()

Como keras no trae por defecto esta variante de la función ReLU(), se efectuó su implementación en forma de función auxiliar, que es posteriormente importada al script en que se describe la red completa (puede consultarse el código completo en el anexo).

El dataset, para comenzar a operar y entrenar el modelo, fue dividido aleatoriamente en tres grupos de imágenes, *train, validation* y *test*. La calidad de los resultados, fue contabilizada con los siguientes indicadores:

Entropía cruzada binaria (*binary cross entropy*) como función de pérdidas.

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

Donde y_i representa la etiqueta de la imagen i (1 para dañado y 0 para vehículo no dañado) y $p(y_i)$ representa la probabilidad predecida de que la imagen sea de tipo 1 en las N imágenes. Los valores que intervienen en la expresión anterior se trabajan en escala logarítmica para que el resultado obtenido, que podría resultar negativo debido al signo, dividido entre las N imágenes utilizadas, sea un valor comprendido entre 0 y 1, a modo de probabilidad.

La exactitud (accuracy) mide el porcentaje de casos que el modelo ha acertado.

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

Como esta medición puede no aportar información suficiente, se incluyeron el recall, la precisión (anteriormente explicadas) y F1, que combina estos dos valores anteriores. Este último medidor permite comparar el rendimiento entre varias soluciones:

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

Los resultados obtenidos, después de numerosas pruebas, pueden verse reflejados aquí:

```
104/104 [==============================] - 1s 12ms/step - loss: 0.2526 - accuracy:
0.9080 - precision_m: 0.9197 - f1_m: 0.8995 - recall_m: 0.9016 - val_loss: 0.3210
- val_accuracy: 0.8718 - val_precision_m: 0.8583 - val_f1_m: 0.8760 -
val_recall_m: 0.9100
```

A pesar de obtener una validation accuracy de entorno al 0.87 no es un mal resultado, continuar indagando con otras posibles configuraciones resultaba necesario, al menos para descartar una arquitectura potencialmente mejor. Es por ello que, a partir de este momento, se recurrió a técnicas de *transfer learning*.

El *Transfer Learning* consiste en tomar las características aprendidas de un problema y aprovecharlas en otro problema nuevo, y similar. Por ejemplo, las características de un modelo que ha aprendido a identificar gatos pueden ser útiles para poner en marcha un modelo destinado a identificar tigres.

El *Transfer Learning* se realiza generalmente para tareas en las que el conjunto de datos tiene muy pocos datos para entrenar un modelo a escala completa desde cero.

Una de las prácticas habituales más interesantes, consiste en tomar parte de las capas que componen la red del problema original e insertarlas, a modo de “injerto” en la arquitectura actual. Esto mismo es lo que se aplica en la red para identificar vehículos, a partir de **InceptionV3**, una red que originalmente se entrenó con ImageNet[5] a fin de obtener una red que asista en el proceso de detección y clasificación de objetos. La red original contiene hasta 7 convoluciones diferentes, pero hay un gran inconveniente: el tamaño mínimo de las imágenes de entrada ha de ser de 75x75.

5.El proyecto ImageNet es una gran base de datos visual diseñada para su uso en la investigación de software de reconocimiento de objetos visuales. El proyecto ha anotado a mano más de 14 millones de imágenes para indicar qué objetos se han fotografiado. ImageNet contiene más de 20.000 categorías con una categoría típica, como "globo" o "fresa", que consiste en varios cientos de imágenes.

Para solucionar este problema, se recurrió a un pequeño script de redimensionado del dataset completo, aplicando interpolación cúbica con openCV. La idea inicial partió de poder insertar las capas iniciales de InceptionV3 unidas a la arquitectura previa, y los resultados, como puede verse más abajo, mejoraron ligeramente.

```
500/500 - 38s - loss: 0.4550 - tp: 234.0000 - fp:
28.0000 - tn: 213.0000 - fn: 25.0000 - accuracy:
0.8940 - precision_m: 0.4680 - recall_m: 0.4680 -
f1_m: 0.4680 - auc: 0.9461 - val_loss: 0.0559 -
val_tp: 8.0000 - val_fp: 0.0000e+00 - val_tn: 11.0000
- val_fn: 1.0000 - val_accuracy: 0.9500 -
val_precision_m: 0.4000 - val_recall_m: 0.4000 -
val_f1_m: 0.4000 - val_auc: 1.0000
```

Nota importante: Debido a una reciente actualización de keras, parece que la forma en que se codificó el cálculo de la precisión, recall, y F1, dan un resultado equivocado (era imposible obtener estos tres parámetros con el mismo valor en cada iteración durante el entrenamiento). Para solucionarlo, se han mostrado en los logs el número de tp, fp, tn y fn que se tienen en cada momento, obtenidos a través de las funciones nativas de keras.metrics.

Así, los resultados verdaderamente obtenidos son:

$$recall = \frac{TP}{TP + FN} = \frac{234}{234 + 25} = 0.903407$$

$$precision = \frac{TP}{TP + FP} = \frac{234}{234 + 28} = 0.89313$$

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision \cdot recall} = 0.89824$$

Como puede apreciarse, los índices no han variado significativamente, pero la validation accuracy sí ha aumentado considerablemente (hemos pasado de un 87 a un 95%).

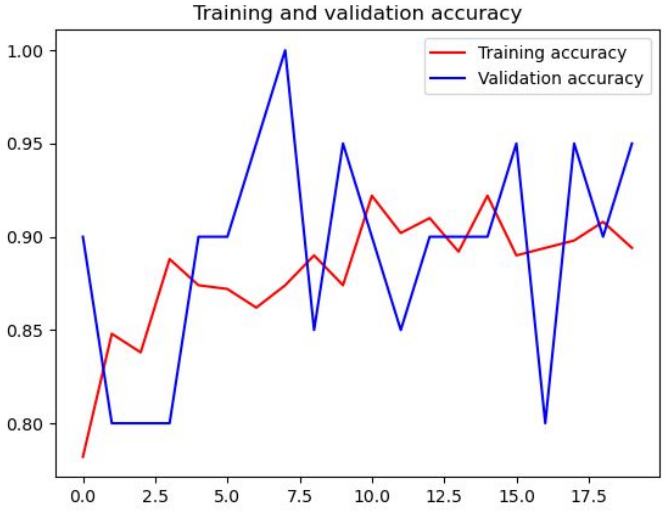


Figura 7. Variación obtenida del accuracy con los datos de entrenamiento y validación durante el proceso de entrenamiento, aplicando esta vez transfer learning.

En este punto del trabajo, se logró contactar con uno de los autores de CADP [6], por lo que se continuó indagando la forma en que podrían procesarse las secuencias de vídeo de manera efectiva, extrayendo secciones de fotogramas para ser pasados por esta primera aproximación de clasificación binaria.

Procesado de vídeo CCTV.

Aunque este trabajo se trata de una revisión académica de posibles técnicas para abordar la detección de accidentes, pensando en una hipotética implementación de esta solución, es interesante poder ofertar un cómputo lo más simple posible, a fin de que los dispositivos sean reducidos también en coste. Con esta premisa en mente, se ha descartado cualquier opción que analizara el fotograma de manera íntegra, buscando un posible accidente. Aplicar alguna estrategia que descarte grandes regiones en las que podemos tener la certeza de que no ha habido un accidente resulta crucial para una optimización íntegra.

Si alguna vez jugaron al juego de las diferencias, aquel en que se tienen dos imágenes muy parecidas, y ha de identificarse qué ha cambiado de una escena a otra, la táctica es análoga a ello: entre dos fotogramas consecutivos, vamos a fijarnos en aquellos elementos que han sufrido cambios de uno a otro: así, tan sólo los vehículos y elementos que se desplazan en ese instante van a ser los que resultan de interés. Esto va un paso más allá, y es que vamos a escoger además aquellos que se han desplazado de manera lo suficientemente brusca. Veamos con más detalle su funcionamiento con un ejemplo.

Detección de cambios en escena con openCV.
Regiones de interés con YOLO
Solución completa con clasificación binaria.

Mejoras sobre la red neuronal.
Limitaciones del dataset original. ¿Cómo validar todo lo hecho?
Reetiquetado de CADP.
Resultados obtenidos.

Conclusiones. Futuras líneas de trabajo.

Haber probado otras variaciones de transfer learning en el reentreno.
Etiquetar más vídeos, y obtener más recortes de accidentes. Probablemente se conseguirían resultados de mayor accuracy.
Clasificación multiclase por tipos de accidente, o tipos de vehículos implicados (peatones, motos, coches, 2 vehículos, múltiples vehículos, 1 solo implicado).
Lenguaje natural para describir accidente (“moto choca con coche en avenida de...”)
Haber experimentado con LTSM al igual que en las publicaciones originales, probando variantes o incluso incorporándose en el framework.
Los algoritmos de tracking funcionaban mal, pero quizá podemos indagar algo a partir de los encodings con algún algoritmo de detección (si puedo identificar el coche N a lo largo de toda la escena, puedo trazar su trayectoria)

Bibliografía

[3] S. R. E. Datondji, Y. Dupuis, P. Subirats, and P. Vasseur. A Survey of Vision-Based Traffic Monitoring of Road Intersections. IEEE Transactions on Intelligent Transportation Systems, 17(10):2681–2698, oct 2016.
[4] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian Detection: An Evaluation of the State of the Art. IEEE Transactions on Pattern Analysis and Machine Intelligence, 34(4):743–761, apr 2012.
[5] F. H. Chan, Y. T. Chen, Y. Xiang, and M. Sun. Anticipating accidents in dashcam videos. In Asian Conference on Computer Vision, 2016.
[6] Ankit Shah, Jean Baptiste Lamare, Tuan Nyugen Anh, Alexander Hauptmann CADP: A Novel Dataset for CCTV Traffic Camera based Accident Analysis International Workshop on Traffic and Street Surveillance for Safety and Security, Nov 2018.

http://norc.aut.ac.ir/convolution-neural-network-joint-with-mixture-of-extreme-learning-machines-for-feature-extraction-and-classification-of-accident-images/

https://link.springer.com/epdf/10.1007/s11554-019-00852-3?author_access_token=Jq5jLsQkUAqfTNK4monbA_e4RwlQNchNByi7wbcMAY4MnzuRzONgB_10L1jmLJYLDFDj-oU6_eZYzttRvAQ0Q05ngHjjgDkVzarJmlHJRwKcTKM3Aq0C_knmOg5b9Jij5NpIn9Iw1I1pUVpk2IWtDA%3D%3D

pintar arquitectura red
http://alexlenail.me/NN-SVG/AlexNet.html

```
In [9]: keras.utils.plot_model(model,
to_file='red_vieja.png', show_shapes=True)
```