



Universidad de Valladolid



**ESCUELA DE INGENIERÍAS
INDUSTRIALES**

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERÍAS INDUSTRIALES

Grado en Ingeniería en Electrónica Industrial y Automática

**Seguimiento de gases quirúrgicas en
imágenes de laparoscopia empleando
redes neuronales convolucionales**

Autor:

Santos Criado, Francisco Javier

Tutor(es):

**De la Fuente López, Eusebio
Departamento de Ingeniería de
Sistemas y Automática**

Valladolid, septiembre de 2020





RESUMEN

El objetivo del presente proyecto es el de realizar un sistema de procesamiento de imágenes basado en técnicas de inteligencia artificial para la detección y seguimiento automático de gasas en operaciones de cirugía laparoscópica con objeto de evitar la retención involuntaria de estas en el interior del paciente.

Este tipo de error médico, aunque es poco frecuente, puede tener consecuencias muy graves para la salud del paciente pudiendo causar incluso la muerte.

En el algoritmo de detección y seguimiento de gasas desarrollado en este TFG se ha intentado buscar un equilibrio entre precisión, sensibilidad y tiempo de procesamiento por lo que se han realizado distintas pruebas empleando tres tipos de redes convolucionales (AlexNet, ResNet y GoogLeNet) analizando las prestaciones de cada una de ellas. Finalmente se ha logrado un algoritmo de detección de gasas con precisión y sensibilidad entorno al 99% y capaz de procesar 3 imágenes por segundo.

PALABRAS CLAVE

Cirugía laparoscópica; Gossypiboma; Redes Neuronales Convolucionales; Detección; Matlab.



ABSTRACT

The aim of this present project is to perform a system of the image processing based on techniques of artificial intelligence for the detection and automatic monitoring of the gauzes in the operation of laparoscopic surgery with the aim to prevent the involuntary retention of gauzes in the inner body of the patient.

Although this type of medical error is a rare event, it could have serious consequences in the patient's health ending even in death.

The algorithm based on the detection and tracking of the gauzes created in this Final Project has pretended to find a balance between precision, sensibility, and time of processing. So different tests have been carried out using three types of convolutional networks (AlexNet, ResNet y GoogLeNet) with a view to analyse its benefits and to develop the algorithm of detection and monitoring of gauzes. Finally, a gauze detection algorithm has been achieved with precision and sensitivity around 99% and it is capable of processing 3 images per second.

KEYWORDS

Laparoscopic surgery; Gossypiboma; Convolutional Neural Networks; Detection; Matlab.



AGRADECIMIENTOS

A mi familia y amigos por apoyarme, a Marta Herráez por ayudarme en estos tiempos difíciles cuando más lo necesité y por supuesto a mi tutor Eusebio.





ÍNDICE GENERAL

1	Introducción	13
1.1	Cirugía laparoscópica	14
1.2	Cirugía robótica	14
1.2.1	AESOP	16
1.2.2	Zeus	16
1.2.3	Sistema Da Vinci	17
1.3	Problemática del material quirúrgico retenido	18
1.4	Objetivos del TFG.....	20
2	Redes neuronales.....	21
2.1	Inspiración biológica	21
2.2	Características de las redes neuronales artificiales	22
2.3	Aprendizaje de las redes neuronales	22
2.4	Redes neuronales convolucionales	24
2.4.1	Arquitectura de una red neuronal convolucional	25
2.4.2	Alexnet	32
2.4.3	ResNet-50	33
2.4.4	GoogLeNet.....	35
3	Desarrollo del sistema del seguimiento de gases quirúrgicas	39
3.1	Software	40
3.1.1	Matlab.....	41
3.1.2	Dataset.....	41
3.2	Hardware.....	42
3.3	Procedimiento.....	43
4	Resultados.....	47
4.1	Alexnet	48
4.2	ResNet50	56
4.3	GoogLeNet	63
4.4	Comparativa	72
5	Conclusiones y líneas futuras	77
6	Bibliografía.....	79



7	ANEXO	83
7.1	Código fuente Red Alexnet	86
7.2	Código fuente red ResNet	90
7.3	Código fuente Red GoogLeNet.....	94
7.4	Código fuente detección en vídeo.....	98
7.4.1	creaSecuenciaVideoTotal.....	98
7.4.2	sacaFrames.m.....	99
7.4.3	GuardaBloques.m.....	99
7.4.4	pintaCuadrados.m	100
7.4.5	creaVideo.m	101



ÍNDICE DE TABLAS

Tabla 1 – Comparativa de los efectos de diferentes filtros de convolución	29
Tabla 2 – Explicación de los componentes de la matriz de confusión.....	44
Tabla 3 – Parámetros iniciales de entrenamiento de la red Alexnet.....	48
Tabla 4 – Parámetros finales del entrenamiento utilizando AlexNet	49
Tabla 5 – Resumen de la precisión en la clasificación mediante Alexnet	51
Tabla 6 – Sensibilidad de la red Alexnet	52
Tabla 7 – Cuadro resumen de la clasificación de la red mediante ResNet50 ..	58
Tabla 8 – Sensibilidad de la red ResNet	59
Tabla 9 – Opciones de entrenamiento de la red GoogLeNet	64
Tabla 10 – Sensibilidad en la red GoogLeNet.	66
Tabla 11 – Cuadro resumen de la clasificación de la red mediante ResNet50.	67
Tabla 12 – Precisión de clasificación en las redes analizadas.	72
Tabla 13 – Sensibilidad en la clasificación de AlexNet, ResNet y GoogLeNet.	73
Tabla 14 – Tiempo de entrenamiento y de clasificación de las redes.	75
Tabla 15 – Resumen general de las redes Alexnet, ResNet y GoogLeNet.....	75



ÍNDICE DE FIGURAS

Figura 1 – Comparación entre cirugía laparoscópica y cirugía convencional abierta. [1]	14
Figura 2 – Robot controlado por dos cirujanos de manera remota [5].....	15
Figura 3 - Cirujano operando con técnica de control por voz. [8]	16
Figura 4 – Robot Zeus con los brazos fijos en la consola del cirujano [10]	17
Figura 5 - Robot Da Vinci con 4 brazos [13].....	17
Figura 6 – Gráfico de la sintomatología causada por un Gossypiboma. [Fuente: elaboración propia].....	19
Figura 7 – Ejemplo de detección de un fotograma de un vídeo captado por un laparoscopio y clasificada con una red neuronal. Los cuadros verdes indican dónde el sistema ha detectado la presencia de gasa.....	20
Figura 8 – Diagrama de bloques del sistema nervioso humano.....	21
Figura 9 – Ejemplo de una red neuronal de 3 capas [17].....	22
Figura 10 – Esquema de una red neuronal artificial multicapa.....	23
Figura 11 - Representación de los píxeles de una imagen en blanco y negro. 24	
Figura 12 – Representación de los 3 canales de una imagen a color [20].	24
Figura 13 – Arquitectura de una red neuronal convolucional [21].	25
Figura 14 – Ejemplo de realización de una convolución [22].	27
Figura 15 – Ejemplo de imagen resultante al aplicar un filtro de convolución . 28	
Figura 16 – Imagen de una gasa sobre la que se van a realizar operaciones convolucionales con filtros.....	28
Figura 17 - Ejemplos de aplicar las capas de MaxPool y AvePool [Fuente: elaboración propia].....	30
Figura 18 – Estructura de una red neuronal convolucional simple formada por dos módulos convolucionales y dos capas totalmente conectadas. [24].....	31
Figura 19 – Clasificación de ‘no-gasa’ y ‘gasa’ utilizando GoogLeNet	31
Figura 20- Esquema de reutilización de una red preentrenada [fuente: elaboración propia].....	32
Figura 21 – Dimensiones de la primera capa de la red Alexnet.	33
Figura 22 – Error de degradación en un entrenamiento en la imagen izquierda con 20 y 56 capas. En la imagen derecha, el error acumulado al clasificar las imágenes. [27].....	34
Figura 23 – Diagrama mostrando un salto de capa.....	34
Figura 24 – Tasa de error (%) en la competición ILSVRC de 2014.....	35
Figura 25 – Ejemplo de una convolución sin etapa intermedia.	36



Figura 26 – Ejemplo de una convolución intermedia.....	36
Figura 27 - Módulos de la capa “Inception Layer”.....	37
Figura 28 – Diagrama de flujo del algoritmo empleado en el sistema de detección [Fuente: elaboración propia].....	39
Figura 29 – Ejemplos de clasificación mediante redes neuronales utilizando diferentes tamaños de bloques. Los bloques verdes indican la presencia de gasa. [Fuente: elaboración propia].....	42
Figura 30 – Diagrama del algoritmo de entrenamiento y detección de gasas..	43
Figura 31 – Ejemplos de diferentes tipos de gasas que el sistema es capaz de rastrear. [32].....	47
Figura 32 – Resultados del entrenamiento de Alexnet con 30 iteraciones.....	49
Figura 33 – Entrenamiento de la red AlexNet con 15 iteraciones.....	50
Figura 34 – Matriz de confusión de la clasificación utilizando Alexnet.	51
Figura 35 – Ejemplos de imágenes mal detectadas por Alexnet.....	53
Figura 36 – Ejemplos de imágenes mal clasificadas por Alexnet.....	53
Figura 37 – Gasa clasificada en bloques 100x100 utilizando AlexNet.	55
Figura 38 – Ejemplos de clasificaciones parcialmente erróneas. En la imagen de la izquierda ha detectado una gasa cuando no la había y en la imagen de la derecha aparece por la parte inferior una gasa que no ha detectado probablemente por su escasa presencia en la imagen.	55
Figura 39 – Fotogramas de detecciones en gasas manchadas de sangre.	56
Figura 40 – Entrenamiento de la red mediante ResNet50	57
Figura 41 – Matriz de confusión de la red ResNet50	57
Figura 42 – Ejemplos de imágenes mal clasificadas en la red.....	59
Figura 43 – Imágenes clasificadas cómo gasas a pesar de la escasa presencia de esta.	59
Figura 44 – Error típico de clasificación como si fuera una gasa en presencia de reflejos.....	60
Figura 45 – Imágenes con porcentaje inferior al 60% de confianza en la clasificación.....	60
Figura 46 – Ejemplo de clasificación de gasas limpias en vídeo.....	61
Figura 47 – Ejemplo de clasificación de gasas limpias en vídeo.....	61
Figura 48 – Ejemplo de clasificación de gasas limpias en vídeo.....	62
Figura 49 - Ejemplo de clasificación de gasas manchadas en vídeo.	63
Figura 50 – Esquema de las últimas capas de GoogLeNet	64
Figura 51 – Resultados del reentrenamiento de GoogLeNet.	65
Figura 52 – Matriz de confusión del entrenamiento de GoogLeNet.	65



Figura 53 – Imágenes clasificadas como ‘no gasa’	67
Figura 54 – Gasas clasificadas de manera errónea	67
Figura 55 – Imagen clasificada como gasa erróneamente	68
Figura 56 – Ejemplos de porcentajes de seguridad en la clasificación con GoogLeNet	69
Figura 57 – Imágenes clasificadas correctamente con un porcentaje de seguridad menor del 60%	69
Figura 58 – Fotogramas de ejemplo clasificando un vídeo con GoogLeNet	70
Figura 59 - Fotogramas de ejemplo clasificando un vídeo con GoogLeNet	71
Figura 60 - Fotogramas de ejemplo clasificando un vídeo con GoogLeNet	71
Figura 61 – Comparativa de la precisión al clasificar diferentes tipos de imágenes mediante redes neuronales	73
Figura 62 – Gráfico de la sensibilidad al clasificar ‘gasa’ y ‘no-gasa’ en diferentes redes	74
Figura 63 – Interior de la carpeta “Bloques”	83
Figura 64 – Interior de la carpeta “Imágenes”	84
Figura 65 – Ejemplos de imágenes en la carpeta “ImagenesDeteccion”	84
Figura 66 – Ejemplos de vídeos en la carpeta “Videos”	84
Figura 67 – Ejemplos de vídeos finales en la carpeta VideosDeteccion	85



1 INTRODUCCIÓN

En la actualidad, las técnicas quirúrgicas están en constante evolución. Ejemplo de ello son las denominadas cirugías mínimamente invasivas como la laparoscopia. En esta técnica se opera al paciente realizando pequeñas incisiones con ayuda de un robot. La problemática principal de estas cirugías es que, al abrir incisiones tan pequeñas en el cuerpo, el cirujano no tiene una visión tan clara de lo que ocurre dentro del cuerpo como en las cirugías tradicionales realizando grandes incisiones, es por ello, por lo que, en algunas ocasiones, queda material retenido en el interior del paciente.

Este es un problema poco frecuente, pero con consecuencias muy graves para la salud del paciente. Existen varios protocolos para evitar que esto ocurra, como contar el material que entra y sale y gasas con códigos de barras, pero aun así es un problema que hoy en día sigue ocurriendo debido principalmente a fallos humanos. Un estudio reciente desveló que en 1 de cada 1500 cirugías se quedaba material olvidado en el interior del paciente, representando las gasas el 69% del material que se queda retenido involuntariamente en el interior del cuerpo.

Es un problema evitable, ya que se debe a un error humano, en muchas ocasiones, producido por largas horas de operación, cambio de la plantilla durante la cirugía, cansancio del personal... Es por ello por lo que se ha desarrollado este sistema, con el fin, no tanto de sustituir la revisión manual, si no como la de ayudar a detectar y rastrear en la manera de lo posible las gasas involuntariamente retenidas en el cuerpo del paciente.

En este proyecto se va a realizar el seguimiento y rastreo de gasas quirúrgicas retenidas en cirugías laparoscópicas mediante técnicas de detección por aprendizaje autónomo empleando redes neuronales convolucionales.

Primeramente, se desarrolla y se pone en contexto el porqué es necesario la creación de este sistema en este tipo de cirugías, y posteriormente se expondrá de manera teórica y práctica mediante la creación de un programa, el desarrollo de la red neuronal y del sistema de detección. Finalmente, se elaboran las conclusiones sacadas del proceso de investigación de los algoritmos empleados en esta detección y se indicarán mejoras y posibles líneas de investigación futuras.



1.1 CIRUGÍA LAPAROSCÓPICA

La cirugía laparoscópica es una técnica quirúrgica que ha evolucionado completamente el mundo de la cirugía mínimamente invasiva. Mediante esta cirugía se permite la visión de la cavidad pélvica-torácica a través de una fibra óptica que actúa como lente. Se realizan varias pequeñas incisiones en el cuerpo del paciente y se utiliza un tubo delgado llamado laparoscopio, el cual tiene una luz y una cámara conectada a un monitor. Mediante esta técnica se permite que el cirujano vea el interior del cuerpo realizándole únicamente una pequeña incisión.

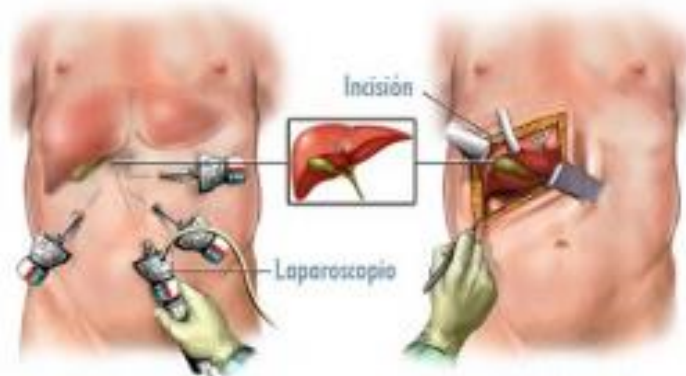


Figura 1 – Comparación entre cirugía laparoscópica y cirugía convencional abierta. [1]

Es a Heinz Kalk a quien se le atribuyen las primeras cirugías de este tipo en la primera mitad del siglo XX. Heinz publicó más de 2000 estudios con muy buenos resultados. A partir de entonces, se empezaron a realizar más técnicas laparoscópicas como biopsias. En España la primera intervención laparoscópica se realizó en 1990 por la Doctora Vincent, extendiéndose por todos los hospitales públicos del país.

El empleo de estas técnicas ha permitido que el tiempo que tiene que pasar una persona en el hospital sea menor, se recupere más rápido, sienta menos dolor, disminuyan las infecciones y las cicatrices sean más pequeñas que con la cirugía tradicional. [2] [3]

1.2 CIRUGÍA ROBÓTICA

Una de las limitaciones de la cirugía laparoscópica es la pérdida de la sensación de profundidad ya que el cirujano percibe una representación bidimensional a través de un monitor. La disminución de la sensación de tacto hace que la operación dependa de la visión. Esta es una de las principales motivaciones para el desarrollo de robots, puesto que pueden mejorar en gran medida el trabajo de los cirujanos eliminando estas limitaciones.



Los robots empleados en cirugías están basados en dos vertientes, la realidad virtual y la cibernética. La primera se produce cuando al cirujano se le hace sentir como si estuviera en un lugar distinto en el que está físicamente. En este ámbito, se genera un entorno en el que el cirujano está inmerso en el denominado entorno virtual. Mientras que la vertiente cibernética engloba la robótica, donde se estudia el diseño y la construcción de robots capaces de realizar tareas programadas por el ser humano ya sea de manera autónoma o controlada por una persona.

Los sistemas robóticos pueden ser autónomos o esclavos. Los primeros necesitan de una inteligencia diseñada para realizar ciertos movimientos, mientras que los esclavos, no tienen capacidad de movimiento autónomo y dependen de la habilidad del cirujano. Suelen tener una estructura semejante a la anatomía de las extremidades superiores de los humanos, para imitar de la mejor manera los movimientos de un médico. [4]



Figura 2 - Robot controlado por dos cirujanos de manera remota [5]

Mediante el uso de robots, los cirujanos pueden llevar a cabo tareas que por complejas o delicadas hubieran sido difíciles o incluso imposible de realizar sin esta asistencia. Es muy frecuente que la cirugía laparoscópica venga ayudada por la utilización de un robot. [6]

Actualmente, los robots quirúrgicos se entienden como un sistema robótico creado para mejorar principalmente la precisión y destreza del cirujano, así como la capacidad de visión 2D que impide al cirujano esa sensación de profundidad al ver a través de una cámara. Este aumento de la precisión en las operaciones laparoscópicas aumenta de manera considerable la seguridad de la operación y la minimización de los posibles errores que puedan derivar de esta. Los sistemas robóticos quirúrgicos eliminan cualquier temblor que pueda tener el cirujano en sus manos, así como añaden estabilidad y comodidad puesto que el cirujano puede



operar sentado con las manos y brazos apoyados. Mediante técnicas de sensorización, es posible convertir grandes movimientos de la mano del cirujano en pequeños movimientos del robot donde se puede ampliar la precisión y control de manera muy eficiente. [4]

1.2.1 AESOP

Se trata del primer robot aprobado por la FDA (Food and Drugs Administration), una de las agencias del gobierno de Estados Unidos más importantes en cuanto a la regulación de aparatos médicos. Fue fundada tras una concesión de la NASA (National Aeronautics and Space Administration) para desarrollar un sistema robótico para uno de los programas espaciales de Estados Unidos. Fue modificado para sujetar un laparoscopio. Se trata un sistema endoscópico para intervenciones quirúrgicas abdominales. Consta de un brazo robótico con una cámara laparoscópica y que puede ser controlado por voz. (Figura 3) [7]



Figura 3 - Cirujano operando con técnica de control por voz. [8]

El cirujano graba los comandos de voz y el ordenador identifica el patrón de voz del médico. El sistema AESOP tenía mecanismos de protección al paciente, de manera que el cirujano marcaba una zona segura del brazo robótico, de manera que previene herir al paciente. [9]

1.2.2 Zeus

El sistema Zeus fue diseñado al igual que AESOP por Computer Motion en Canadá. Amplió el concepto de robótica ampliándolo a telerrobótica en la cirugía robótica. Fue diseñado para ayudar en la cirugía y constaba de tres brazos robóticos controlados por el cirujano a distancia. El primer brazo era activado también por voz, los otros dos brazos restantes imitaban los movimientos del cirujano.



Figura 4 – Robot Zeus con los brazos fijos en la consola del cirujano [10]

El principal inconveniente que tuvo el robot es que los brazos robóticos eran muy grandes, por lo que limitaba mucho el espacio del quirófano. El robot se dejó de fabricar y se retiró en 2003 tras la fusión de la empresa Computer Motion con Intuitive Surgical, quien era su rival hasta entonces. [11]

1.2.3 Sistema Da Vinci

El sistema quirúrgico robótico Da Vinci amplía las posibilidades de operaciones en el interior del cuerpo humano de la manera menos invasiva posible hasta la fecha. El sistema transforma los movimientos del cirujano en la consola del robot de manera intuitiva en pequeños movimientos de los brazos del robot. El cirujano al operar entra en un entorno tridimensional donde está completamente inmerso en la operación para “vivir” la operación quirúrgica casi desde el interior del paciente. [12]

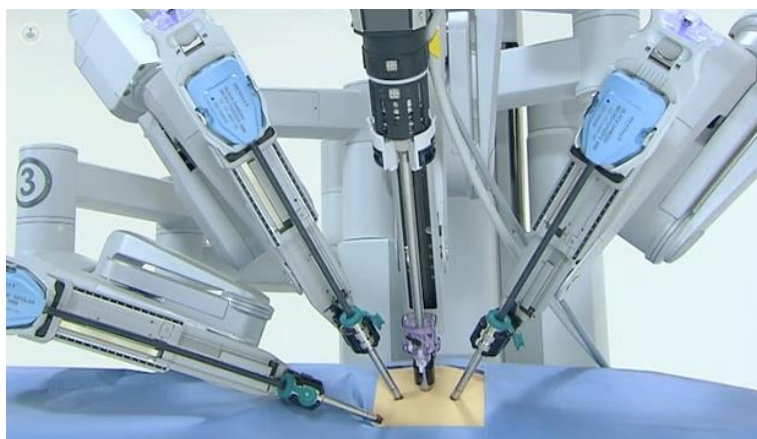


Figura 5 - Robot Da Vinci con 4 brazos [13]



Consta de dos componentes principales:

- La consola de control del cirujano: controlada por él mismo, consta de una visión estereoscópica con visión tridimensional y de dos mandos que controlan los brazos del robot.
- El robot quirúrgico con tres o cuatro brazos articulados. Dos son quirúrgicos y el resto portan las luces y las cámaras desde varios ángulos.

El cirujano opera sentado mirando la consola y con una imagen tridimensional. El robot recibe las órdenes en tiempo real y consta de un sistema de infrarrojos que detecta cuándo el cirujano está operando desde su asiento, para evitar toques accidentales a los mandos. [14]

1.3 PROBLEMÁTICA DEL MATERIAL QUIRÚRGICO RETENIDO

Cualquier error quirúrgico puede conllevar mucho riesgo para el paciente, de hecho, es la octava causa de muerte en Estados Unidos. Los errores humanos son evitables adoptando una serie de patrones preventivos. Pero esto no siempre ocurre así, y siempre existe un riesgo de error médico a pesar de las medidas que toman hoy en día en un quirófano. La AORN (Association of periOperative Registered Nurses) recomienda para el conteo correcto de gasas que se realicen las siguientes medidas:

- Explorar la cavidad antes del cierre.
- Recuento de todo el material quirúrgico antes y después, se deben tener en cuenta todo tipo de gasas, compresas e instrumentos de operación.
- Los objetos punzantes como las agujas se deben contar en todos los procedimientos.
- Todas las guías y procedimientos de conteo se tienen que revisar anualmente y siempre estar accesibles a todo el personal del quirófano.

En estudios realizados recientemente, se ha estimado que el número de incidencias por material quirúrgico retenido no intencional se estima entre 1 de 1500 cirugías. Siendo el error más común en el acto quirúrgico, constituyendo un 4% de todos los reportes.

Gossypiboma o gasoma es el término que se acuña al residuo olvidado accidentalmente en el interior del abdomen durante una intervención. Pueden ser esponjas, gasas, agujas, tubos, entre otras cosas. Un estudio realizado a lo largo de 16 años determinó que las gasas accidentalmente retenidas representan el 69% del total de los casos analizados.



El material quirúrgico erróneamente retenido es muy dañino, aún con los protocolos que se siguen para evitarlo ocurren fallos, fruto de operaciones muy largas, cambios del personal durante la intervención da como resultado final el factor humano.

Otro estudio realizado en Massachusetts abarcando 22 Años, los síntomas en los pacientes con material olvidado en su interior:

- Un 10% son asintomáticos.
- Con sospechas de lesiones con un dolor crónico secundario derivado de la intervención, un 30%.
- Con síntomas febriles e íleo postoperatorio un 30%.
- Únicamente fiebre en el inmediato postoperatorio un 10%.
- Síndrome febril y orificio fistuloso en el sitio de la herida quirúrgica un 10% de los observados.
- El 10% fueron diagnosticados con síndrome febril y con una masa intraabdominal sospechosa.

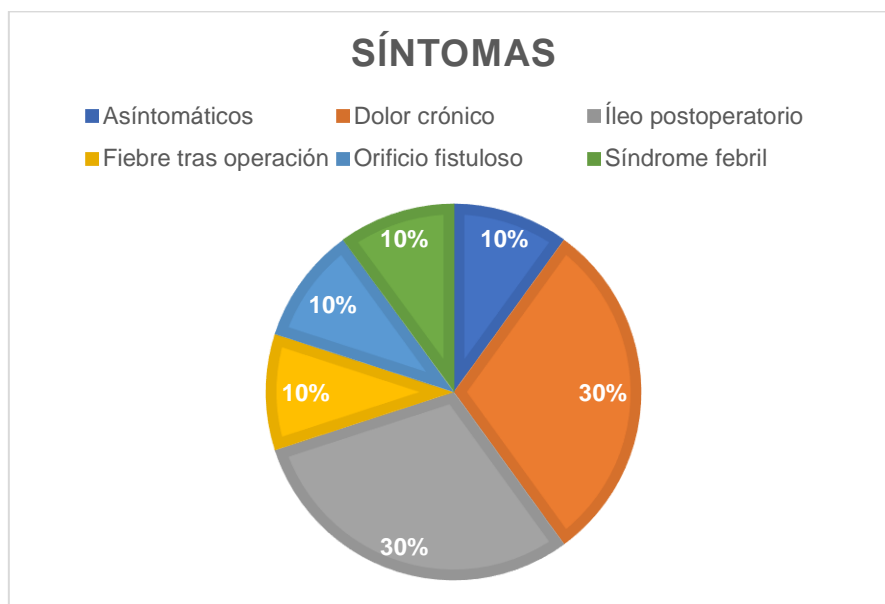


Figura 6 – Gráfico de la sintomatología causada por un Gossypiboma. [Fuente: elaboración propia]

Los objetos retenidos en el interior del cuerpo pueden causar numerosos problemas, y habitualmente para la retirada de ellos se precisa una nueva cirugía, incluso aunque el paciente sea asintomático. Para recuperarlo, el método más común suele ser el de realizar una operación mediante cirugía convencional, no obstante, resolverlo mediante la cirugía mínimamente invasiva puede reducir considerablemente las complicaciones. [15]



1.4 OBJETIVOS DEL TFG

El material quirúrgico retenido expone un gran problema para el paciente tal y como se ha desarrollado en el apartado anterior. En este Proyecto de Fin de Grado se pretende desarrollar un sistema de detección de las gasas quirúrgicas que han sido involuntariamente retenidas dentro del cuerpo del paciente.

El rastreo de estas gasas se realizará a través de la imagen procedente del laparoscopia y un sistema de redes neuronales entrenado exclusivamente para la detección de gasas quirúrgicas. Para ello, las imágenes del laparoscopia se dividirán en bloques que se analizarán de manera independiente y se clasificarán conforme haya gasa o no la haya. Finalmente se obtendrá la imagen recompuesta de los bloques anteriores recuadrados en verde en caso de haber gasa o sin recuadrar en caso de no haberla.

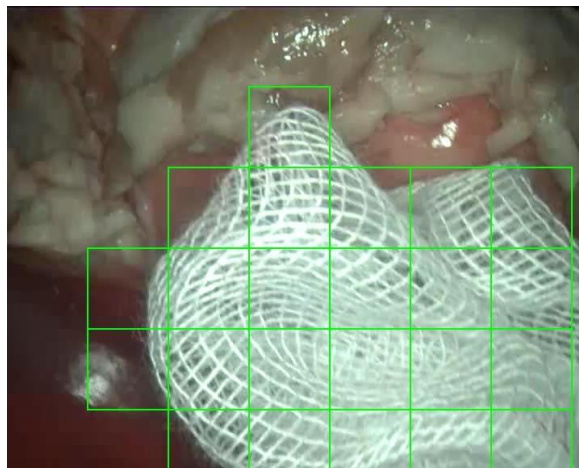


Figura 7 – Ejemplo de detección de un fotograma de un vídeo captado por un laparoscopia y clasificada con una red neuronal. Los cuadros verdes indican dónde el sistema ha detectado la presencia de gasa.

Además, se pretende recoger una comparativa de las diferentes redes neuronales principales para realizar un estudio de las mejores opciones para la detección eficaz de estas gasas. La detección está pensada que sea en tiempo real, por lo que se han desarrollado varios programas que procesan los vídeos para recuadrar las gasas como en el ejemplo de la figura 7.

En el capítulo siguiente se introducirá el marco teórico del TFG presentando los conceptos básicos referentes a redes neuronales convolucionales y el funcionamiento de las redes entrenadas, así como su composición en capas y tiempos experimentales de procesamiento para su comparación, y posible implementación en el quirófano.



2 REDES NEURONALES

El sistema de cálculo más complejo conocido por el hombre es el cerebro humano. Los ordenadores y los humanos ejecutan tareas de maneras bien diferentes, así como un humano tiene la capacidad de reconocer un objeto de manera sencilla, para un ordenador es una tarea difícil. Sin embargo, por otro lado, para un experto en el ámbito de la contabilidad, tareas que a un humano llevaría mucho tiempo y dificultad, para un ordenador es relativamente sencillo realizar.

La inteligencia artificial ha sido testigo de un crecimiento monumental al cerrar la brecha entre las capacidades de los humanos y las máquinas. El propósito de este campo es permitir que los ordenadores vean el mundo de la misma manera que lo vemos los humanos, puedan percibirlo de manera similar e incluso usen el conocimiento de imágenes, análisis y clasificación de imágenes o procedimientos del lenguaje.

2.1 INSPIRACIÓN BIOLÓGICA

El sistema nervioso de los humanos puede ser representado como un sistema de tres capas tal y como se describe en la figura 8. En el centro del sistema se localiza el cerebro, el cual recibe información, la procesa y toma decisiones. Los receptores son los encargados de convertir los estímulos que reciben del cuerpo en impulsos eléctricos dirigidos hacia la red neuronal (cerebro). Los efectores, convierten los impulsos que genera el cerebro en respuestas del cuerpo humano.

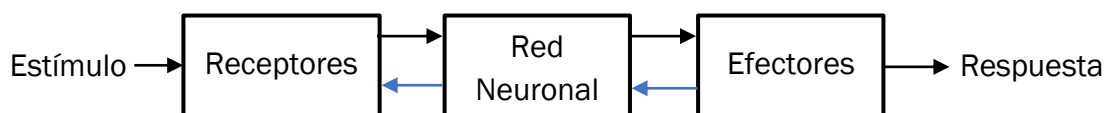


Figura 8 - Diagrama de bloques del sistema nervioso humano.

Ramón y Cajal fue el pionero en descubrir y poner nombre a toda esta estructura del cerebro y del sistema nervioso. Las puertas lógicas de silicio son del orden de cinco o seis órdenes de magnitud más rápidas que las neuronas del cerebro, los procesos en un chip de silicio rondan los nanosegundos mientras que los procesos en el cerebro rondan los milisegundos. No obstante, el cerebro humano tiene una gran cantidad de neuronas conectadas entre ellas de manera masiva, lo que permite compensar esa velocidad tan baja. En la corteza humana, se estima que hay más de 10 mil millones de neuronas y 6 trillones de conexiones. [16]



2.2 CARACTERÍSTICAS DE LAS REDES NEURONALES ARTIFICIALES

Las Redes Neuronales Artificiales (RNA) están fundamentadas en las redes neuronales biológicas que ocurren en el cerebro de los humanos. Se trata de un sistema de entramado de neuronas que interactúan entre sí y colaboran para producir una salida. La interacción es entre capas, cada capa es tiene unos pesos o parámetros, que se van formando durante el entrenamiento de la red. El principal objetivo es el de hacer que esos parámetros se calculen de manera que, a través de una entrada, en este caso, una imagen, se produzca la salida deseada, la clasificación de esta. De este modo, las redes se adaptan y son capaces de aprender, para conseguir esos pesos de manera eficaz, se necesita un buen y amplio conjunto de datos.

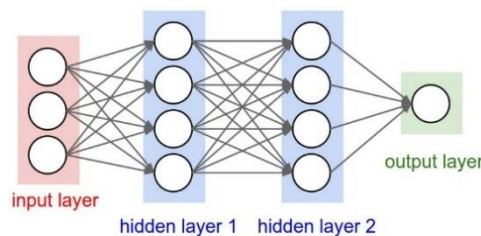


Figura 9 – Ejemplo de una red neuronal de 3 capas [17].

Existen tres aspectos que caracterizan a las redes neuronales artificiales:

- **Aprendizaje:** Las redes pueden modificar su comportamiento por medio del estudio, ejercicio o experiencia. Son capaces de comportarse de diferente manera como respuesta a su entorno. A través de un conjunto de entradas son capaces de ajustarse para producir salidas consistentes.
- **Generalización a través de ejemplos anteriores:** Las RNA lo hacen de manera automática debido a su propia naturaleza. Una vez que está entrenada, la red puede responder de manera correcta incluso a entradas con ruido o distorsión.
- **Abstracción** de la esencia de las entradas: las redes artificiales son capaces de obtener información de las entradas reconociendo patrones.

2.3 APRENDIZAJE DE LAS REDES NEURONALES

Las Redes neuronales están formadas por diferentes capas, cada una está encargada de una tarea específica. Hay tres tipos de capas, la capa de entrada, las capas ocultas y la capa de salida. La función principal de las redes neuronales es la



de clasificar imágenes correctamente respecto a las categorías de las cuales esté entrenada. El método de entrenamiento de la red neuronal se realiza observando muchos ejemplos de esa categoría, y asignando parámetros a las capas para poder a través de una entrada (imagen) obtener la salida deseada ('gasa' o 'no gasa'). Estos parámetros son los denominados pesos de una capa. Por tanto, el entrenamiento en sí consiste en dar los valores de esos pesos para conseguir el objetivo.

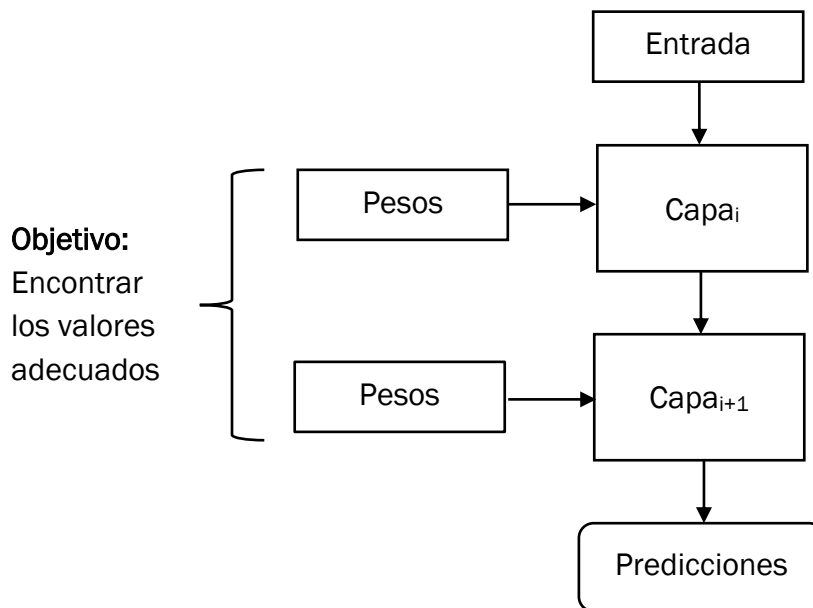


Figura 10 – Esquema de una red neuronal artificial multicapa.

Las redes neuronales pueden contener millones de parámetros, encontrar el parámetro correcto, es una tarea compleja, especialmente al modificar un único parámetro que provoca que cambien los pesos del resto de la red. [18]

Al principio del entrenamiento, todos los parámetros de la red son asignados de manera aleatoria, por tanto, al empezar a entrenar con los primeros datos ya se produce una salida. Esta salida, es comparada con la salida que se desea y que es conocida, y ese error se propaga hacia la capa anterior de manera sucesiva.

La propagación de errores o *Back-propagation* es un algoritmo de entrenamiento supervisado utilizado en redes neuronales artificiales. Es un tipo de entrenamiento que consiste en aprender de un conjunto de datos con sus etiquetas, por eso se dice que es supervisado. El supervisor es el encargado de que aprenda de los errores y corrige para conseguir la salida deseada. Este proceso se repite constantemente y las capas se van ajustando automáticamente para minimizar el error. Cuando se han



empleado todas las imágenes en el algoritmo, se ha terminado una época o *epoch* y el momento de poder utilizar las imágenes de validación, diferentes a las de la prueba para poder determinar la precisión. [19]

2.4 REDES NEURONALES CONVOLUCIONALES

Las Redes Neuronales Convolucionales (CNN) son muy similares a las redes neuronales artificiales explicadas anteriormente. Procesan las capas de manera similar que el córtex del ojo humano para poder identificar objetos. La red contiene varias capas, cada una está especializada en detectar un tipo de forma. Por ejemplo, las primeras capas pueden identificar líneas, curvas y poco a poco se van especializando hasta las capas más profundas que pueden identificar y reconocer, como en el caso del proyecto, gasas.

Las redes neuronales sólo pueden trabajar con números. Esto no es un problema puesto que para un ordenador una imagen no es más que una matriz de números o píxeles que representan como de oscuro es cada píxel. Los valores conviene normalizarlos, los colores de píxeles se representan con valores de 0 a 255, mediante la normalización se convertirán a valores entre 0 y 1.

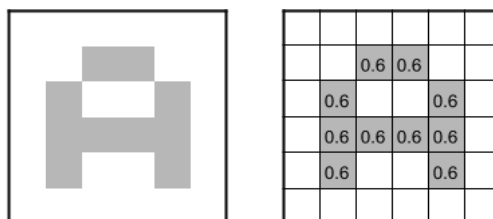


Figura 11 - Representación de los píxeles de una imagen en blanco y negro [20].

Una imagen a color tiene tres canales, rojo verde y azul. Cada canal tiene una matriz relacionada con valores de 0 a 255. Por lo tanto, la representación de nuestra foto a color es a través de tres matrices apiladas.

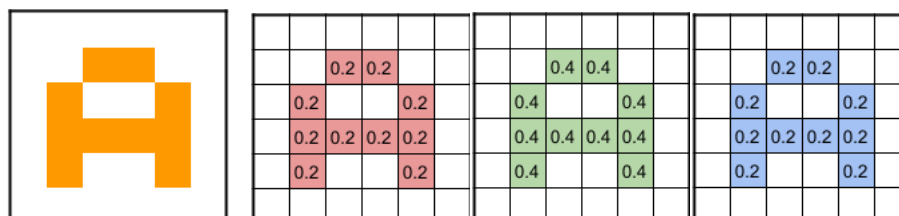


Figura 12 - Representación de los 3 canales de una imagen a color [20].

Las redes neuronales convolucionales contienen varias capas ocultas donde se realizan distintas operaciones como detección de líneas, curvas, siluetas, bordes...



La profundidad de una red depende del número de capas, cuantas más capas, se dice que más profunda es la red.

La red en su primera capa toma los píxeles de cada imagen. Estas imágenes son procesadas a través de un tensor que incluye el número de imágenes, las dimensiones y los canales (3 en caso de imagen a color). Por ejemplo, si tenemos imágenes 100x100 píxeles, equivalen a 10000 neuronas. Si, además, esas imágenes son a color, tendríamos 30000 neuronas de entrada.

Hay varias arquitecturas de redes convolucionales neuronales disponibles que han demostrado ser muy eficientes y precisas, algunos ejemplos de ellas son, LeNet, AlexNet, VGGNet, GoogLeNet y ResNet. En el sistema de detección de gasas se reentrenarán algunas de las citadas a fin de encontrar cual es la que más se ajusta al modelo y mejores resultados en cuanto a clasificación de las gasas quirúrgicas obtiene.

2.4.1 Arquitectura de una red neuronal convolucional

Las redes convolucionales son similares a las artificiales, están constituidas por capas de neuronas que trabajan con pesos diferentes. La característica que las distingue de las no convolucionales es que las capas de la red tienen 3 dimensiones, largo, ancho y profundidad. Cada neurona recibe los valores de los píxeles de las imágenes.

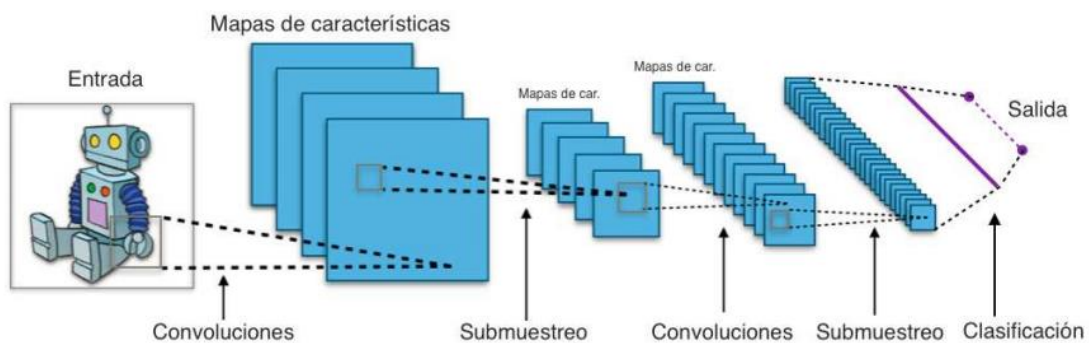


Figura 13 – Arquitectura de una red neuronal convolucional [21].

Una red neuronal convolucional está formada por una secuencia de capas. Las tres capas principales son la capa de convolución, 'pooling' o submuestreo y 'fully-connected'.

- **Capa de entrada:** se trata de la primera la capa de la arquitectura de una red convolucional, es la entrada de la red. En nuestro caso serán imágenes de dimensiones 100x100 píxeles. Dependiendo de la red neuronal que



entrenemos, esta imagen deberá tener unas dimensiones predefinidas en la primera capa, para ello se realizarán reescalamientos y transformaciones para adecuarla a esas dimensiones. Más adelante en los apartados de AlexNet, ResNet y GoogLeNet se desarrollarán estas dimensiones.

- **Capa convolucional:** el nombre de estas redes es dado por la operación que realiza sobre ellas. Nos permite extraer las características de nuestra imagen de entrada. Se divide la imagen en pequeñas regiones, se aplica una transformación a cada región y guardamos la salida como una nueva representación de nuestra imagen. Como resultado cada salida acaba con las partes más importantes de la imagen original. Estas regiones se obtienen a través de una matriz filtro que va recorriendo la matriz principal de los valores de los píxeles de la imagen. Por ejemplo, con una imagen 5x5, donde los valores de los píxeles valen 0 o 1 y usamos un filtro 3x3. La convolución se realiza de la siguiente manera:

Tenemos una matriz 5x5:

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Y el filtro 3x3 siguiente:

1	0	1
0	1	0
1	0	1

La convolución entre la matriz 5x5 y la 3x3 se realiza de la siguiente manera:

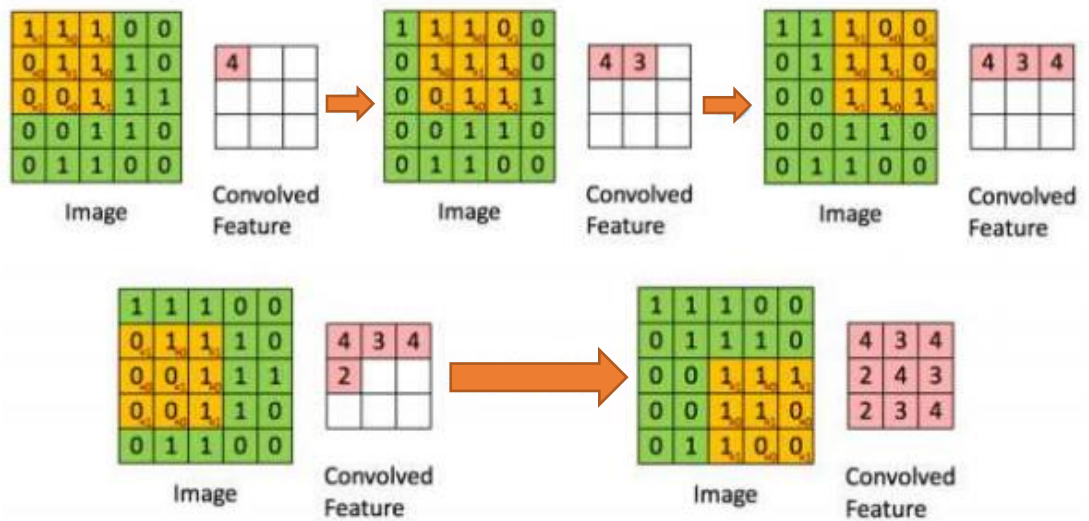


Figura 14 – Ejemplo de realización de una convolución [22].

El filtro naranja se va desplazando a través de la imagen verde (matriz 5x5) de píxel en píxel realizando la multiplicación de las 2 matrices superpuestas. El producto se guarda en una matriz nueva, la matriz rosa.

Cambiar los valores del filtro permite detectar diferentes características por lo que se obtiene una matriz rosa diferente. Por ejemplo, se pueden identificar bordes o curvas, y por tanto la matriz es diferente.

En la figura 15 se muestra un ejemplo de una imagen de entrada y de salida usando el siguiente filtro de transformación:

$$\begin{pmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{pmatrix}$$

Insertamos el filtro por ejemplo encima de la ceja derecha y realizamos la operación de convolución usando la matriz del filtro arriba mencionada. Como resultado da el valor 198 que de una escala de 0 a 255 da el cuadrado situado en la imagen de la derecha.

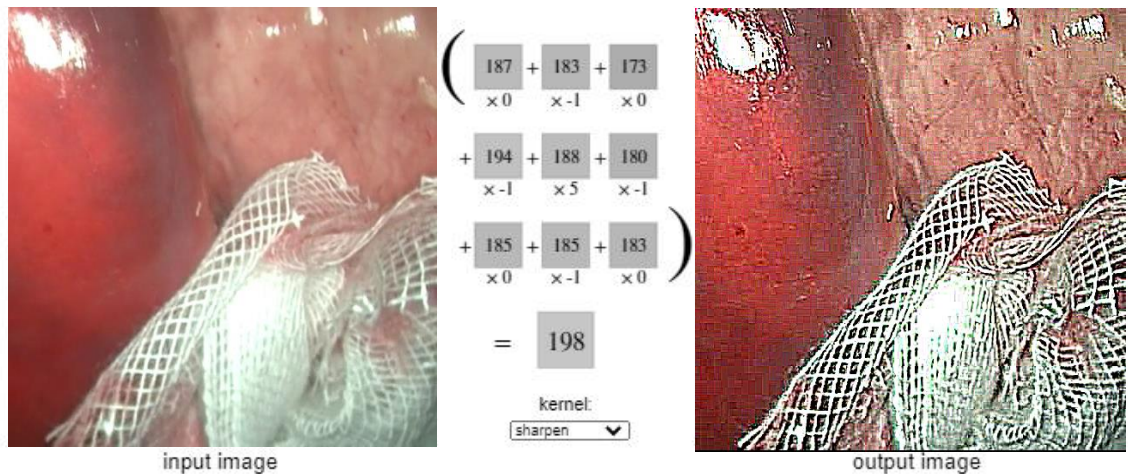


Figura 15 – Ejemplo de imagen resultante al aplicar un filtro de convolución [23].

Los valores de estos filtros son determinados durante el entrenamiento de la red. El número de filtros viene marcado por las especificaciones de la red, cuantos más filtros quiere decir que la red puede extraer muchas más características y poder tener un modelo de detección mejor.

Dependiendo del filtro de convolución que se utilice, se pueden encontrar diferentes imágenes de salida. Por ejemplo, para la siguiente imagen de nuestra red, de una gasa en el interior del cuerpo:



Figura 16 – Imagen de una gasa sobre la que se van a realizar operaciones convolucionales con filtros.

En la siguiente tabla se recogen las diferentes imágenes que resultan de aplicar diferentes filtros. Los filtros son modificados únicamente cambiando los valores de la matriz de convolución, y con ellos se es capaz de detectar bordes, líneas, curvas, etc.








Operación	Filtro	Imagen resultada
Identidad	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Detección de bordes	$\begin{bmatrix} 1 & 2 & 1 \\ 2 & -11 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Nitidez	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Desenfoque gaussiano	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

Tabla 1 - Tabla comparativa de los efectos de diferentes filtros de convolución

En las redes neuronales, estos filtros aprenden durante el entrenamiento de la red, y son configurados durante la creación de las capas de la red. Cada filtro está especializado en detectar un tipo de característica de la imagen, por esta razón, cuantos más filtros tenga nuestra red, más robusta será y mejor reconocerá los patrones de las imágenes.



- **Capa de Pooling:** de manera similar a la capa de convolución, es la responsable de reducir la matriz de convolución obtenida anteriormente mediante submuestreos de los filtros. Esto permite reducir considerablemente el coste computacional.

El método más habitual es el de 'MaxPool'. Consiste en extraer las características dominantes de la imagen definiendo varios pixeles vecinos (por ejemplo, un cuadrado 2x2) y se toma el elemento más grande entre esos vecinos. Aplicando esta capa devuelve el máximo valor de cada porción de la imagen.

Otro método bastante utilizado es el de 'AvePool' donde en vez devolver el máximo valor, se devuelve el valor medio los valores del cuadrante.

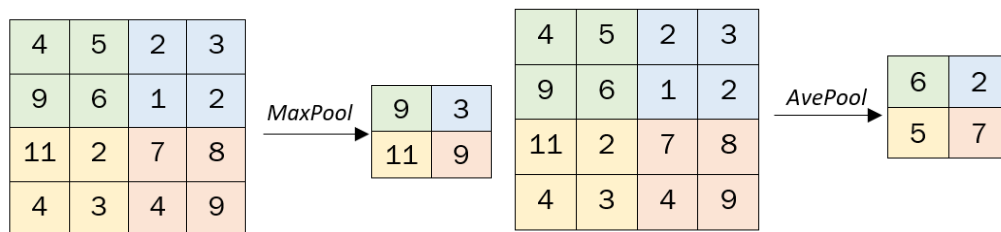


Figura 17 - Ejemplos de aplicar las capas de MaxPool y AvePool [Fuente: elaboración propia].

En la figura 17 se ejemplifica la operación de *Max Pooling* y *Average Pooling*; en ambos, se ha escogido una ventana de 2x2 que constituyen los vecinos en los que se realizará el Pooling. Por ejemplo, en la ventana verde, utilizando el método de MaxPool, el máximo valor entre 4,5,6 y 9 es 9, y para el método average Pooling, es 6, la media entre los números 4,5,6 y 9.

- **Fully connected layer**

Para terminar, la salida de la Pooling Layer actúa como entrada para la denominada Fully Connected Layer. Es la última capa de las redes convolucionales y se trata de una capa de clasificación para indicar a que clase pertenece la imagen de entrada. Se encarga de clasificar las imágenes y etiquetarlas conforme a su clase. Está formada por tantas neuronas como clases, en nuestro caso tenemos 2 clases (gasas y no-gasas), por tanto, tiene 2 neuronas, cada neurona está conectada con todos los elementos de la capa de Pooling.

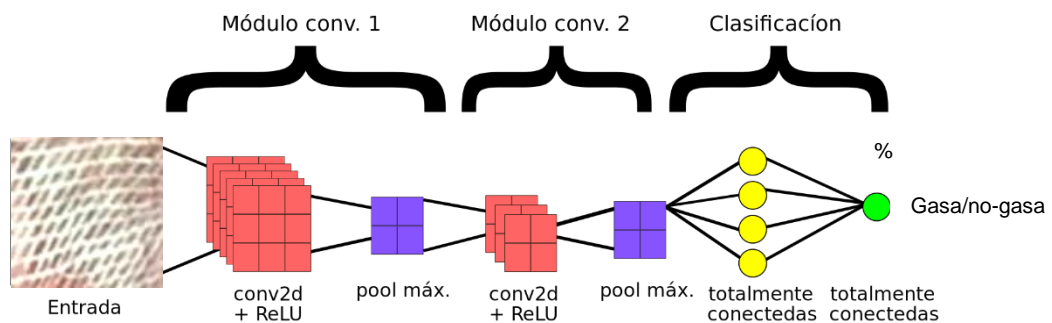


Figura 18 – Estructura de una red neuronal convolucional simple formada por dos módulos convolucionales y dos capas totalmente conectadas. [24]

El objetivo principal es el de coger los resultados de las capas anteriores para clasificar la imagen. La salida de la capa está traspuesta en un vector lineal donde se identifica la imagen con la probabilidad que tiene de que pertenezca a esa clase.

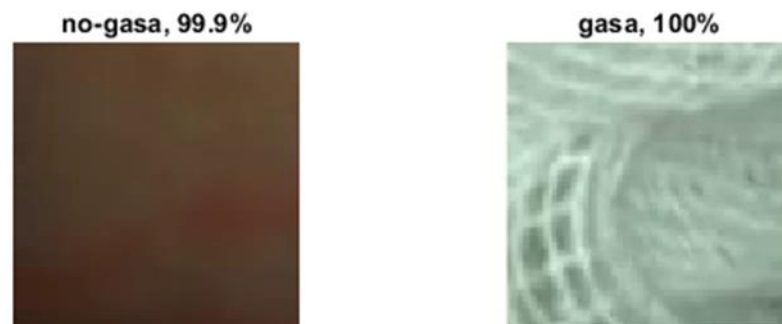


Figura 19 – Ejemplo de clasificación de ‘no-gasa’ y ‘gasa’ utilizando GoogLeNet

Por ejemplo, utilizando la red neuronal GoogLeNet se entrenó la red para clasificar las imágenes en “gasas” y “no-gasas”, como se observa en la figura 19, la primera imagen la última capa ha decidido (con ayuda de las anteriores) que no era una gasa con una probabilidad del 99,9%, mientras que la segunda ha sido de estar totalmente seguro de haber clasificado correctamente. En general, en este proyecto se ha conseguido entrenar varias redes neuronales con muy buenos resultados como el que se acaba de exponer, y que se desarrollarán en el capítulo 3.



2.4.2 Alexnet

Se trata de la primera arquitectura de red neuronal convolucional que se popularizó en el año 2012. Fue desarrollada con estructura CUDA (Compute Unified Device Architecture) un conjunto de herramientas creadas por nVidia para funcionar usando una GPU por Alex Krizhevsky, Ilya Sutskeer y Geoff Hinton para una competición. Consiguió un error de un 15.3%, 10.8 puntos por encima del siguiente finalista en el LSVRC (Large Scale Visual Recognition Challenge)) [17]. Se trata de una red más profunda que su antecesora histórica, LeNet y con más filtros y capas convolucionales.

Su arquitectura es más sencilla que la de la red ResNet y consta de ocho capas. El modelo inicial dispone de más de 1 millón de fotografías que puede clasificar más de 1000 categorías de objetos (como teclados, cafés y muchos animales).

Alexnet cuenta con su propia base de imágenes para entrenar esas 1000 categorías, pero no incluye gasas. Es por ello por lo que tenemos que coger esa arquitectura de red y re-entrenarla para que sea capaz de reconocer las gasas.

Utilizando Matlab, vamos a implementar la estructura de Alexnet para que podamos reentrenarla correctamente.

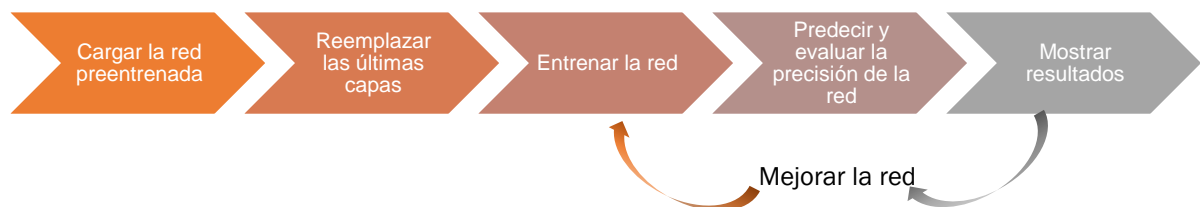


Figura 20- Esquema de reutilización de una red preentrenada [fuente: elaboración propia].

El código se puede encontrar al final de este proyecto, no obstante, se explicará en este apartado lo más relevante de cara a poder reutilizar una red entrenada previamente.

Primeramente, se cargan las imágenes en el entorno de Matlab, se introducen las imágenes de prueba y de entrenamiento. Posteriormente se llama a la función predefinida Alexnet, mediante Matlab comprobamos su estructura:

```
>> net=alexnet;  
>> analyzeNetwork(net);
```




	Name	Type	Activations
1	data 227x227x3 images with 'zerocenter' normalization	Image Input	227x227x3

Figura 21 – Dimensiones de la primera capa de la red Alexnet.

La primera capa, requiere imágenes de 227 x 227 x 3 donde el número 3 indica los canales de color (rojo, verde y azul). Por tanto, nuestras imágenes tienen que ser transformadas de 100x100 a 227x227. Esto en Matlab se hace fácilmente con la función [augmentedImageDatastore\(\)](#).

El resto de las capas para reentrenar la red quedan invariables a las originales. Las que tenemos que cambiar son las 3 últimas capas. Esto es así porque están configuradas para 1000 clases. El cambio se producirá de 1000 clases a solamente 2 clases, gasa y no-gasa.

```
net = alexnet;  
layers = net.Layers;  
numClasses=2;  
layers(end-2) = fullyConnectedLayer(numClasses);  
layers(end) = classificationLayer;
```

Una vez que se han modificado las 3 últimas capas y se han adecuado las imágenes de entrada a las dimensiones de la primera capa, se puede proceder al entrenamiento de la red.

El entrenamiento de la red de Alexnet se realiza con la función [trainNetwork\(\)](#), cuyos parámetros de entrada son el dataset de imágenes, las 28 capas modificadas anteriormente y las opciones de entrenamiento. Especificar las opciones de entrenamiento es fundamental a la hora de ahorrar costes computacionales y conseguir la máxima eficacia y eficiencia en la detección posterior de las gasas.

2.4.3 ResNet-50

Tras la victoria de Alexnet en el 'LSVRCen' 2012, las redes se han ido convirtiendo en redes convolucionales cada vez más profundas. ResNet consiguió el primer puesto en el mismo torneo celebrado en 2015 y hoy es una de las redes más utilizadas en reconocimiento y clasificación. [25]

La red residual ResNet es capaz de entrenar cientos o incluso miles de capas. Es la red más profunda hasta la fecha [26]. Posee una gran capacidad de entrenamiento con un gran rendimiento, es utilizada en muchas aplicaciones de visión por computador, clasificación de imágenes, detección de objetos y reconocimiento facial.



Sin embargo, hacer más profunda la red (ampliar el número de capas) no implica que sea más eficiente. De hecho, el apilamiento de capas produce el denominado error de gradiente. Es decir, el error de la capa anterior se va propagando entre las sucesivas capas empeorando la precisión en la detección y clasificación. Se puede generalizar, que a medida que las capas van siendo más profundas, el rendimiento puede llegar a saturarse por el error de gradiente acumulado.

Antes de la llegada de ResNet había algunas formas de paliar el problema del error de degradación, pero ninguna conseguía corregirlo completamente [26].

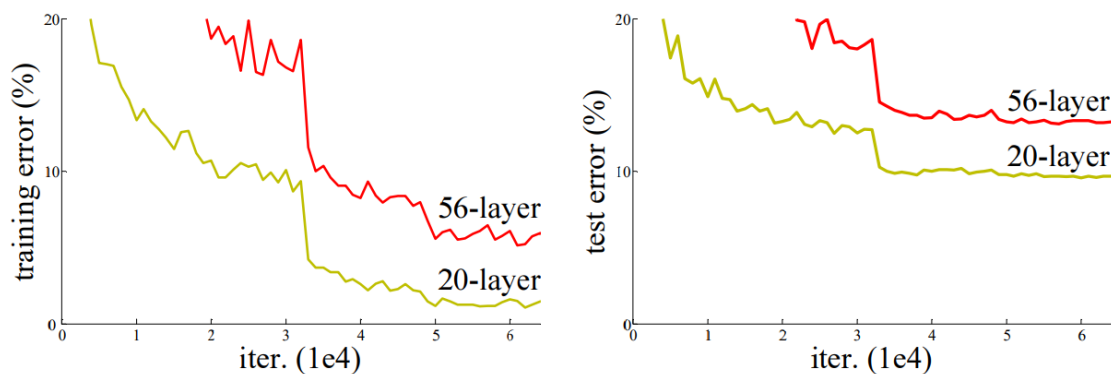


Figura 22 – Error de degradación en un entrenamiento en la imagen izquierda con 20 y 56 capas. En la imagen derecha, el error acumulado al clasificar las imágenes. [27]

La manera de paliar esta situación y poder disponer de una red más profunda consiste en introducir una conexión directa entre la capa de salida y la capa de entrada, de manera que, durante el entrenamiento, los pesos se adapten sustituyendo la capa saltada y por consiguiente se utilicen menos capas para el entrenamiento.

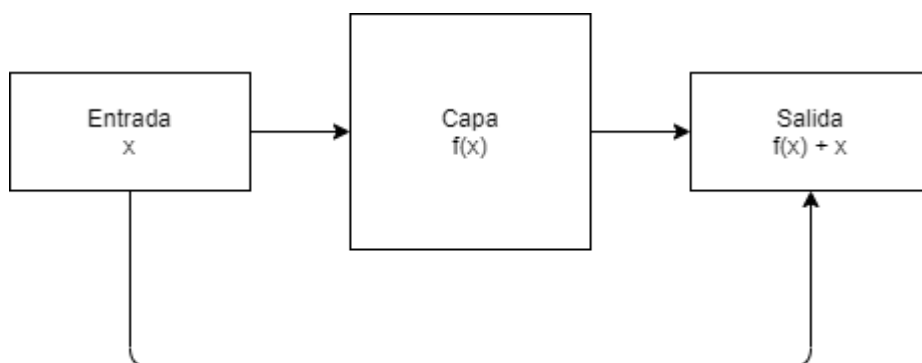


Figura 23 – Diagrama mostrando un salto de capa. [Fuente: elaboración propia]



Al agregar la entrada a la salida como se ve en la figura 23, se omiten las capas intermedias, en estos saltos se incluyen matrices de transformación con pesos adicionales que aprenden de los pesos saltados mediante esta conexión.

Una manera de entender esto se basa en suponer que los pesos son 0 en el inicio del entrenamiento, todas las capas empiezan con el peso 0. Sin embargo, en las redes neuronales como ResNet, la red empieza con la función identidad puesto que la entrada es pasada directamente a la salida, ella sola aprende los residuos que quedan en cada capa.

2.4.4 GoogLeNet

La red convolucional neuronal GoogLeNet fue la ganadora en 2014 del ILSVRC (Large Scale Visual Recognition Challenge), un reto que se propuso a los algoritmos de detección de objetos y clasificación de imágenes. [28] El nombre de la red deja entrever, que es desarrollada por Google, y también contiene “LeNet” para rendir tributo al profesor Yan LeCun, creador de una de las primeras redes neuronales en 1998 [29].

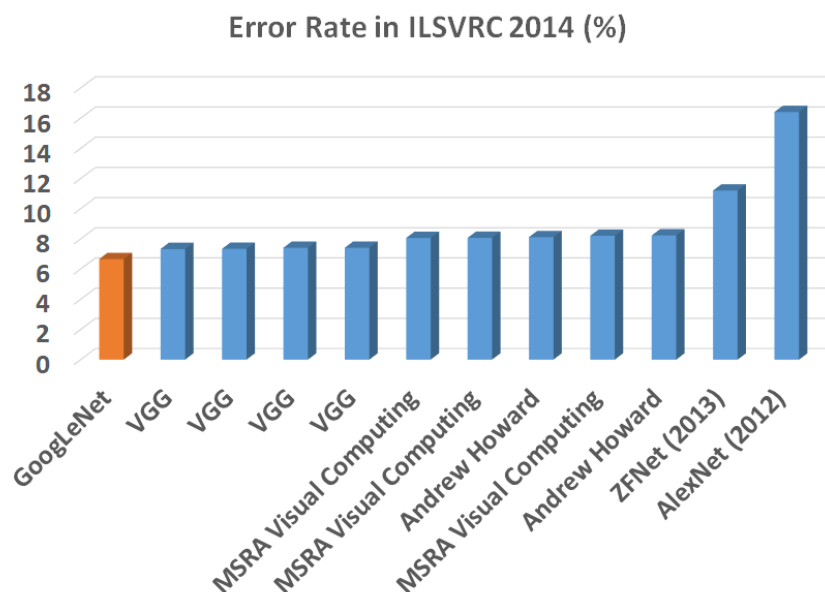


Figura 24 – Tasa de error (%) en la competición ILSVRC de 2014.

GoogLeNet está formada por 22 capas y de al menos 12 parámetros menos que Alexnet. Consta de una capa de convolución en el medio de la red y una capa de submuestreo (“Pooling”) al final de la red en vez de utilizar la capa *Fully connected* como el resto.



Inicialmente la red fue diseñada para que pudiera ser utilizada en un teléfono móvil [30]. Por este motivo se incluye una capa de convolución de dimensiones 1x1 en el medio de la red que consigue reducir de computación e incluir más capas. Por ejemplo, si se utilizara una convolución sin disponer de la capa intermedia 1x1, tendríamos lo que se muestra en la figura 25.

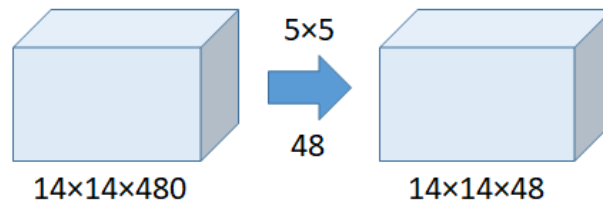


Figura 25 - Ejemplo de una convolución sin etapa intermedia.

El número de operaciones sería = $(14 \times 14 \times 48) \times (5 \times 5 \times 480) = 112,9$ millones. En cambio, utilizando una convolución 1x1 intermedia:

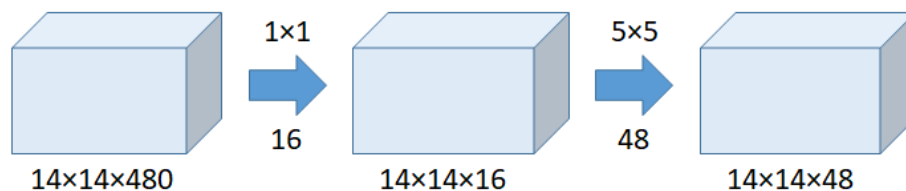


Figura 26 - Ejemplo de una convolución intermedia.

Las operaciones para realizar serían:

- Convolución 1x1 = $(14 \times 14 \times 16) \times (1 \times 1 \times 480) = 1,5$ millones
- Convolución 5x5 = $(14 \times 14 \times 48) \times (5 \times 5 \times 16) = 3,8$ millones

Por lo tanto, el número total de operaciones sería igual a 5,3 millones, valor muy por debajo de los casi 113 millones de operaciones. [31]

En consonancia con la capa intermedia de dimensiones 1x1, en esta red también se incorpora un módulo denominado *Inception Module*, cuya traducción en español sería “módulo de inicio”. Es el encargado de realizar en paralelo diferentes transformaciones de tamaño de las imágenes, desde las matrices 1x1 a más grandes 5x5.

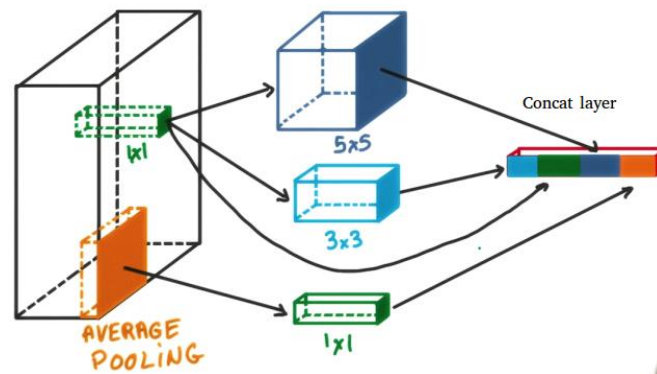


Figura 27 - Módulos de la capa “Inception Layer”.

La idea consiste en entrenar a la red con más variedad de tamaños y de filtros. La manera más sencilla para mejorar la precisión del entrenamiento de la red consiste en usar más capas y más imágenes de muestra.

Otra opción que hace especial a GoogLeNet es la no utilización de la capa *fully connected layer* que, por ejemplo, utiliza Alexnet. Utiliza una nueva capa llamada, “Global Average Pooling”, es una capa encargada de coger el valor medio de cada matriz resultante del *inception module*. Mediante esta técnica se puede conseguir una mejora de la precisión del 0,6%.

GoogLeNet tiene 22 capas en total, es una red bastante profunda comparada con Alexnet que tan sólo tiene 8 capas, pero no tanto como ResNet-50 desarrollada posteriormente con 50 capas.

Inicialmente, viene preparada para funcionar con el dataset de CIFAR-10, un banco de 60000 imágenes de 10 clases. Para hacerla funcionar con nuestro dataset de imágenes modificamos las últimas capas al igual que con Alexnet y ResNet.





3 DESARROLLO DEL SISTEMA DEL SEGUIMIENTO DE GASAS QUIRÚRGICAS

El sistema desarrollado consiste en varios programas, todos ellos creado con el entorno de programación Matlab. El sistema final de seguimiento implementado es capaz de rastrear y detectar gasas quirúrgicas en vídeos de cirugías laparoscópicas. El algoritmo mediante el cual se realiza esta clasificación se expone en la figura 28.

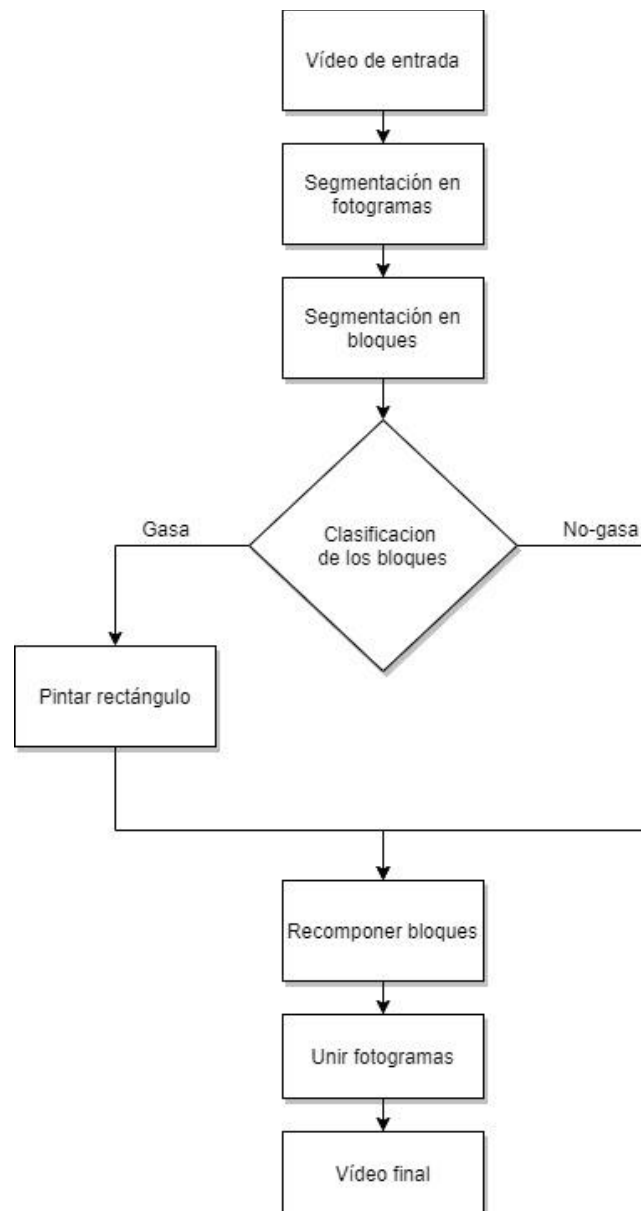


Figura 28 – Diagrama de flujo del algoritmo empleado en el sistema de detección [Fuente: elaboración propia].



El vídeo que se toma como entrada se divide en fotogramas, es un parámetro programable de manera que se pueden separar los fotogramas a la misma velocidad que está grabado el video original o se pueden segmentar en menos fotogramas. La frecuencia de imágenes habitual oscila entre 20 y 30 fotogramas por segundo.

Una vez que está segmentado el vídeo en imágenes, cada una de ellas se divide en cuadrados de dimensiones en píxeles. Estudios previos [32] han determinado que los cuadrados en los que hay que fragmentar la imagen más precisos y que dan mejores resultados a la hora de clasificar las gasas son de tamaño 100x100 píxeles.

Tras la segmentación de la imagen en bloques y se almacenan en el disco duro, se clasifican uno por uno y se guardan los resultados de si son gasa o no gasa. Acto seguido, se recomponen los bloques en una imagen y en aquellos donde se ha detectado gasa, se recuadran en verde y se pasa al siguiente fotograma para segmentarlo en bloques y clasificarlo de igual manera.

Una vez que se han clasificado todos los fotogramas, se crea el vídeo uniendo todos los fotogramas con el seguimiento y el rastreo de la gasa.

3.1 SOFTWARE

Para el procesado de las imágenes y entrenamiento de la red neuronal, se requiere un gran trabajo de programación con un elevado coste computacional. Es por esta razón, por la que se ha elegido un software flexible, muy utilizado y completo. Con el software Matlab se satisface lo expuesto anteriormente y se reduce además el tiempo de desarrollo de la aplicación. Matlab ofrece un entorno de desarrollo integrado con su propio lenguaje de programación. Cuenta con muchas funciones para trabajar con imágenes (transformaciones, morfología, adulteración de colores...) La documentación de Matlab es pública y en su página web se pueden encontrar infinitud de textos, códigos y manuales que facilitan esta tarea.

Además, se han instalado los nuevos controladores de mi tarjeta gráfica Nvidia, con el programa GameReady GeForce. Para los entrenamientos de las diferentes redes se utiliza la GPU puesto que el aprendizaje profundo requiere gran cantidad de cálculos, la mayor parte son multiplicaciones de matrices. Mediante la utilización de la GPU liberamos trabajo de la CPU principal. Matlab es compatible con la tecnología CUDA de las gráficas Nvidia, lo que permiten acelerar los procesos de entrenamiento y clasificación sin necesidad de hacer cambios en el código.



3.1.1 Matlab

Matlab es la abreviatura de *MATrix LABoratory* (laboratorio de matrices), se trata de un IDE (entorno de desarrollo integrado) con su propio lenguaje de programación. Es muy potente trabajando con matrices, algoritmos y la comunicación con hardware externo (como Arduino). En Matlab existen *toolboxes* que ahorran grandes partes de código. Para la realización de este sistema de detección de gasas se han instalado las siguientes *toolbox* de Matlab:

- **Deep Learning Toolbox Model for Alexnet Network:** esta toolbox implementa la red Alexnet, con la base de datos de ImageNet con más de 14 millones de imágenes. No se empleará ese banco de datos, ya que se ha reentrenado para detectar gasas quirúrgicas. [33]
- **Deep Learning Toolbox Model for ResNet-50 Network:** también se ha hecho uso de esta aplicación para reentrenar mediante la red ResNet nuestro sistema. Esta, solamente funciona a partir de la R2017b. [34]
- **Deep Learning Toolbox Model for GoogLeNet Network:** para la red GoogLeNet también se precisó de su barra de herramientas, para disponer de las 22 capas. [35]

El código se ha desarrollado en ficheros con extensión 'mlx', un formato de archivo utilizado para poder compilar de manera sencilla varias partes de código independientes. Los ficheros con extensión 'm' son los predeterminados para funcionar con Matlab, en este trabajo se utilizarán para realizar funciones. En el anexo se incluyen los códigos de los ficheros con extensiones 'mlx' y 'm'.

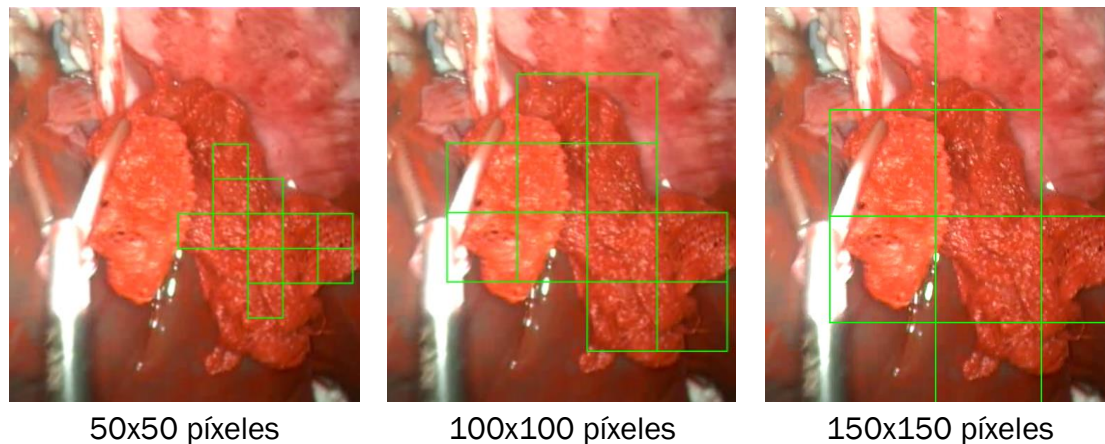
3.1.2 Dataset

Para entrenar las redes, se necesita un *dataset* o banco de imágenes. En este caso se quiere entrenar para poder hacer un rastreo de gasas en cirugías, para ello se disponen de dos tipos de imágenes, 'gasa' y 'no-gasa'. Las imágenes de gasas pertenecen a 2 tipos de gasas, limpias y manchadas con sangre y fluidos. En internet no existe un *dataset* público de gasas quirúrgicas, por lo que las imágenes son privadas, todas las gasas que se han utilizado para entrenar y validar la red son imágenes captadas de vídeos en la Universidad de Valladolid mediante un endoscopio empleando un entorno simulado con vísceras de animal y gasas quirúrgicas.

Las imágenes del *dataset* tienen como medidas 100x100 basadas en estudios previos [32], que indicaban que esas medidas eran las más apropiadas a la hora de



clasificar las gasas. En la figura 29 se ha ratificado y corroborado que es el mejor tamaño de las imágenes para su clasificación.



50x50 píxeles 100x100 píxeles 150x150 píxeles
Figura 29 – Ejemplos de clasificación mediante redes neuronales utilizando diferentes tamaños de bloques. Los bloques verdes indican la presencia de gasa. [Fuente: elaboración propia]

Los bloques 50x50 clasifican de manera muy pobre, dejando la mitad de la gasa sin clasificar. Los de tamaño 100x100 lo realizan correctamente, dejándose alguna parte sin recuadrar con bloques que con el ejemplo 150x150 sí que se completa, pero al ser tan grandes los bloques se tarda más en procesar, y se procesan partes que tienen un mínimo de gasa. Lo ideal sería clasificar píxel a píxel y que los bloques fueran lo más pequeño posible, pero en bloques de menos de 100x100 la clasificación no es correcta. Por tanto, tal y como decía la bibliografía [32] y se ha ratificado en la clasificación con redes neuronales, el tamaño más adecuado es el de 100x100.

El total de imágenes de ‘gasas’ que disponemos son de 1785 y de ‘no-gasas’ de 4892. Para entrenar la red se han utilizado 1000 imágenes de cada categoría elegidas aleatoriamente.

3.2 HARDWARE

Para la realización de este proyecto se ha decidido indicar las características en las que se han realizado la red computacional, de cara a cuando se exponen los tiempos de procesamiento de las distintas redes, se tomen como un dato de referencia que puede variar y mucho entre diferentes ordenadores. Se ha empleado un portátil Acer Aspire F5-573-702H cuyas especificaciones son:

- Tipo de procesador: Intel Core i7-6500U 2.5GHz con TurboBoost a 3.1GHz.
- Memoria RAM: 16GB DDR4



- Disco Duro: 500GB SSD WD.
- Tarjeta Gráfica: NVIDIA GeForce 940MX con 4GB dedicados. DDR5.
- Sistema Operativo: Windows 10 Home 64bits.

Es un ordenador de gama media-alta, con buenas prestaciones para realizar los entrenamientos y clasificación de las imágenes en las redes. Los datos de tiempos de ejecución, procesamiento, entrenamiento y clasificación pertenecen a un ordenador con estas características, por tanto, es muy probable que esos tiempos se mejoren, mejorando las características del procesador y de la tarjeta gráfica.

3.3 PROCEDIMIENTO

Para implementar el algoritmo en el sistema de detección de gases quirúrgicas en el interior del cuerpo mediante *Deep Learning*, se ha empleado el esquema que se muestra a continuación.



Figura 30 – Diagrama del algoritmo de entrenamiento y detección de gases

Para el entrenamiento de la red, se utilizará el método de “Aprendizaje Supervisado” debido a que incluimos en las imágenes la etiqueta de si es gasa o no lo es. Cuantos más ejemplos se propongan al sistema de cara a su entrenamiento, mejor definición de gasa tendrá. Para ello se disponen de 2 carpetas, una con imágenes de gasas en el interior del estómago y otra con las paredes de éste sin gasa. Las gasas de las imágenes se encuentran en 3 modos, limpias de sangre, parcialmente manchadas de sangre y manchadas por completo de sangre. Para elegir el banco de imágenes de entrenamiento utilizaremos 1000 imágenes elegidas de manera aleatoria con la función de Matlab `splitEachLabel()`. Las 4670 imágenes restantes servirán de prueba del algoritmo y las que nos darán la precisión.

Una vez elegidas las imágenes, se procesan dependiendo de la red que vayamos a emplear (Alexnet, ResNet50 o GoogLeNET). El procesamiento consiste en adecuar las dimensiones de las imágenes con las dimensiones de la capa de entrada de la red. Originalmente las imágenes que se disponen son de dimensiones 100x100 píxeles.

A continuación, tras elegir las imágenes de entrenamiento y procesarlas adecuadamente en nuestra red, se procede al entrenamiento. Para el entrenamiento de la red se utilizan las imágenes de entrenamiento que previamente se habían



seleccionado de manera aleatoria. El entrenamiento se realiza en lotes, cada lote tiene un número de iteraciones que se pueden configurar, así como las veces que se entrenan las imágenes. En Matlab se utilizará la función `trainNetwork()`. Esta función trae como parámetros: el *dataset* de las imágenes de entrenamiento, las capas de la red y opciones configurables tales como el número máximo de intervalos de entrenamiento, las imágenes de validación o el tamaño mínimo del lote, entre otros.

A través de las imágenes de validación que se incluyen en el campo de opciones de la función `trainNetwork()` se obtienen indicadores como la matriz de confusión, el porcentaje de precisión, porcentaje de error...

La matriz de confusión consiste en una matriz ordenada, donde se representan las categorías que la red tiene que clasificar. En el eje de coordenadas se indica el valor real de la categoría y en el de abscisas el valor predicho por el algoritmo como se muestra en la siguiente matriz:

		VALOR REAL	
		'gasa'	'no-gasa'
PREIDCCIÓN	'gasa'	Gasa verdadero	Falso gasa
	'no-gasa'	Falso no-gasa	Verdadero no-gasa

Tabla 2 - Explicación de los componentes de la matriz de confusión.

En el interior de la tabla 2 se pueden ver con claridad los valores de las categorías que el algoritmo ha clasificado correctamente, así como los que ha clasificado de manera errónea. El valor que se busca es que coincidan los valores reales con los predichos, con esta matriz se puede analizar de manera sencilla la precisión de la clasificación.

La precisión está relacionada con el porcentaje de acierto entre la predicción y el valor real. Este valor se puede calcular viendo la matriz de confusión que se puede sacar con Matlab a través de la siguiente fórmula:

$$\textit{Precision en gasa} = \frac{\textit{Gasas verdaderas}}{\textit{Gasas verdaderas} + \textit{Falso gasas}}$$



$$\textit{Precision en NoGasa} = \frac{\textit{NoGasas verdaderas}}{\textit{NoGasas verdaderas} + \textit{Falso Nogasa}}$$

Mediante la sensibilidad medimos la cantidad de muestras que el sistema ha sido capaz de identificar del total de muestras de 'gasa' o 'no-gasa'. A partir de la matriz de confusión se puede calcular:

$$\textit{Sensibilidad en gasa} = \frac{\textit{Gasas verdaderas}}{\textit{Gasas verdaderas} + \textit{Falso Nogasas}}$$

$$\textit{Sensibilidad en NoGasa} = \frac{\textit{NoGasas verdaderas}}{\textit{NoGasas verdaderas} + \textit{Falso gasa}}$$

El algoritmo adecuado tendrá un equilibrio entre la precisión y su sensibilidad, es decir, que identifica las muestras correctamente de una clase y no se equivoca en otra clase. Esta precisión se puede medir a través del 'F₁ score'.

$$F_1 = \frac{2}{\frac{1}{\textit{precision}} + \frac{1}{\textit{sensibilidad}}}$$

Finalmente, la métrica que evalúa la precisión respecto al total se denomina exactitud. Al igual que las anteriores, se puede calcular de manera sencilla, echando un vistazo a la matriz de confusión como:

$$\textit{Exactitud} = \frac{\textit{Gasas verdaderas} + \textit{NoGasas verdaderos}}{\textit{Gasas verdaderos} + \textit{NoGasas verdaderos} + \textit{falsas Gasas} + \textit{falsos NoGasa}}$$



4 RESULTADOS

Se han desarrollado diferentes redes neuronales convolucionales, a fin de realizar un estudio exhaustivo de varios parámetros de las redes. Se medirán conceptos explicados en el capítulo anterior, como precisión, sensibilidad y F_1 . De cara a una posible implementación en el mundo real, se deberán tener en cuenta estos parámetros además de los tiempos de procesamiento y de ejecución. Estos tiempos que se explicarán en detalle, dependen mucho del sistema de procesamiento que empleemos, en el capítulo 3.2 se indican las características del ordenador empleado en los experimentos.

Para la realización de este trabajo se han empleado diferentes ejemplos de imágenes de gasas en plenas cirugías, así como imágenes de lo que se puede encontrar en el interior del cuerpo. La coloración de las gasas dependiendo de la cantidad de sangre que tienen se ha tenido en cuenta. En la Figura 31 se puede observar las diferentes muestras que se han tenido en cuenta a la hora del entrenamiento de la red.

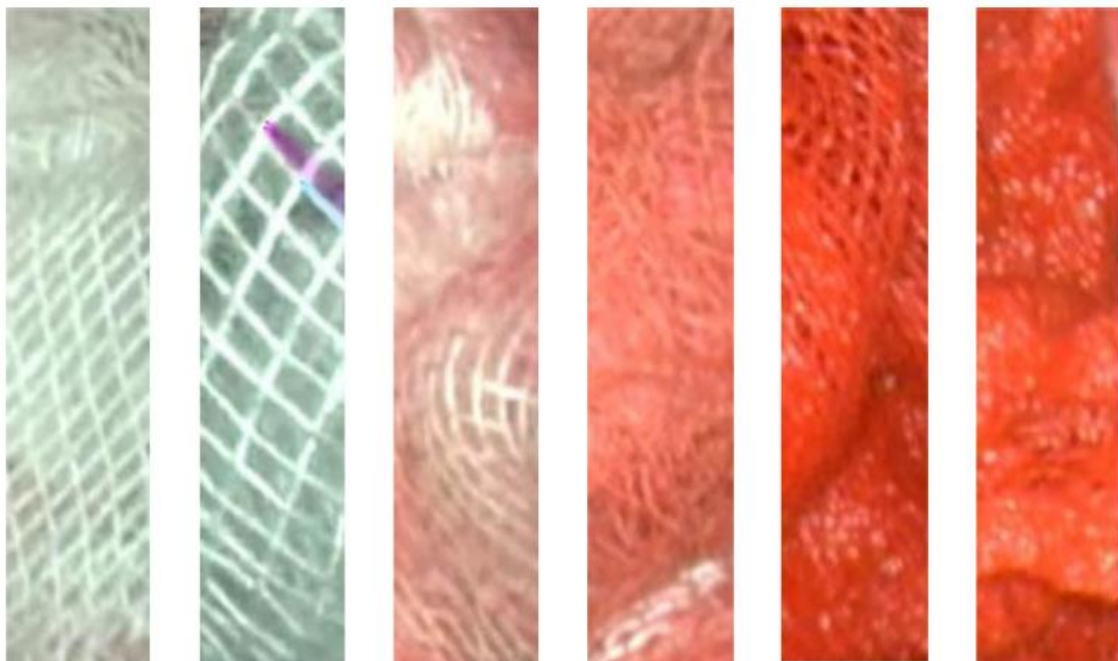


Figura 31 – Ejemplos de diferentes tipos de gasas que el sistema es capaz de rastrear. [32]

Además, se cuentan con imágenes donde hay una presencia del laparoscopio. Estas imágenes, se han clasificado de manera que, si hay más de un 80% de presencia de gasa, están catalogadas como gasa y de lo contrario están catalogadas como fondo, a pesar de una presencia de gasa. Es por ello por lo que a veces, el algoritmo detecta que la imagen está clasificada como no-gasa, a pesar de haberla detectado como



gasas por su escasa presencia en la imagen. Por esta razón, es posible que a veces, resultados que el ordenador ha contado como negativo, se tomen de manera correcta, ya que, ante una mínima presencia de gasas, el sistema es capaz de detectarla.

4.1 ALEXNET

Primeramente, se partirá de la red Alexnet. En el anterior capítulo ya se ha puesto en contexto sobre cómo es el funcionamiento de esta red. Se toma como punto de partida imágenes de tamaño 100 x 100, y se redimensionan al tamaño de entrada de la primera capa de la red de 227x227 píxeles.

El entrenamiento se realizó utilizando 1000 imágenes de gasas y otras 1000 del fondo escogidas de manera aleatoria por el algoritmo. El resto de las imágenes, 4670, forman parte de las imágenes de prueba. Para el entrenamiento de la red se escogieron los siguientes parámetros:

Tasa de aprendizaje inicial	0.001
Número máximo de “épocas”	2
Frecuencia de validación	30

Tabla 3 - Parámetros iniciales de entrenamiento de la red Alexnet

La tasa de aprendizaje inicial controla la rapidez con la que la red se adapta al problema. Es un número natural entre 0 y 1, está relacionado con las épocas de manera inversamente proporcional. Cuanto menor sea la tasa de aprendizaje inicial requerirá más épocas dados los cambios más pequeños realizados en los pesos en cada época. A su vez, cuanto mayor sea la tasa de aprendizaje, da como resultado cambios más rápidos y por lo tanto requieren menos épocas de entrenamiento. [36]

Tras realizar varios entrenamientos a modo de prueba y error para indicar esos parámetros, este es un buen resultado para la red, con un 99.70% de precisión.

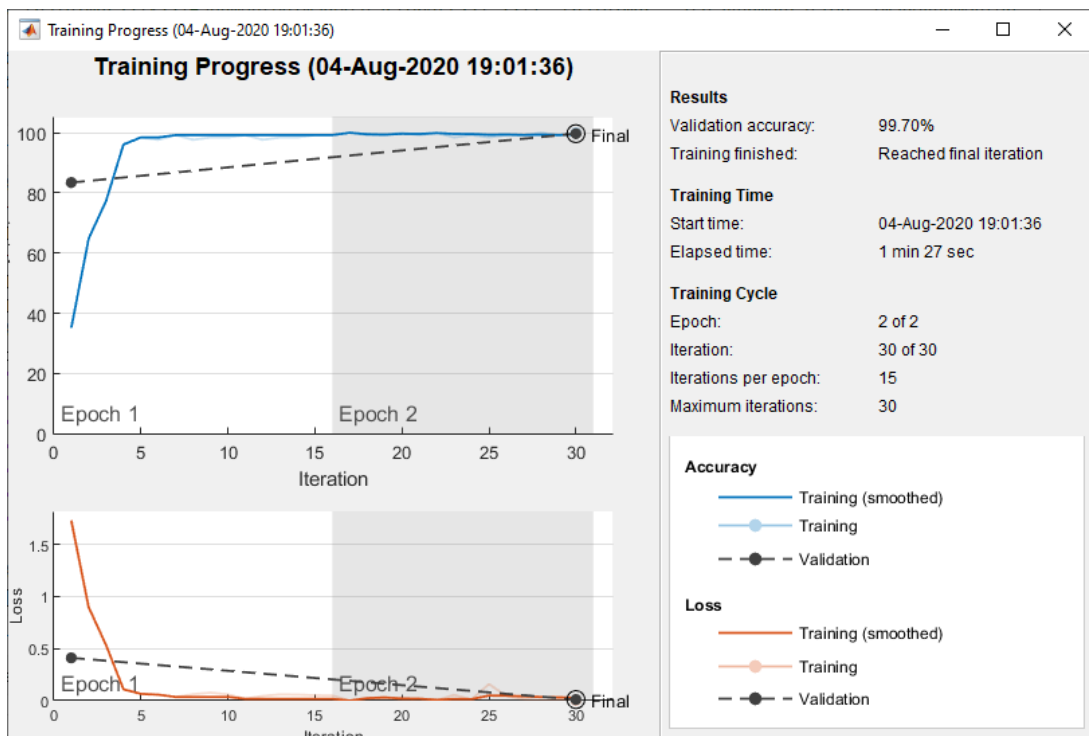


Figura 32 – Resultados del entrenamiento de Alexnet con 30 iteraciones

Viendo la gráfica superior de la figura 32 que muestra la precisión del entrenamiento frente a las iteraciones, se puede observar que, a partir de la quinta iteración, el algoritmo es suficientemente bueno y no mejora mucho la precisión. Por ello se ha decidido reducir las iteraciones a la mitad y por consiguiente las pasadas que dará el algoritmo o *epoch* quedarán reducidas a una ya que en la segunda pasada no se aporta ningún tipo de información nueva.

Tasa de aprendizaje inicial	0.001
Número máximo de “épocas”	1
Frecuencia de validación	30

Tabla 4- Parámetros finales del entrenamiento utilizando AlexNet

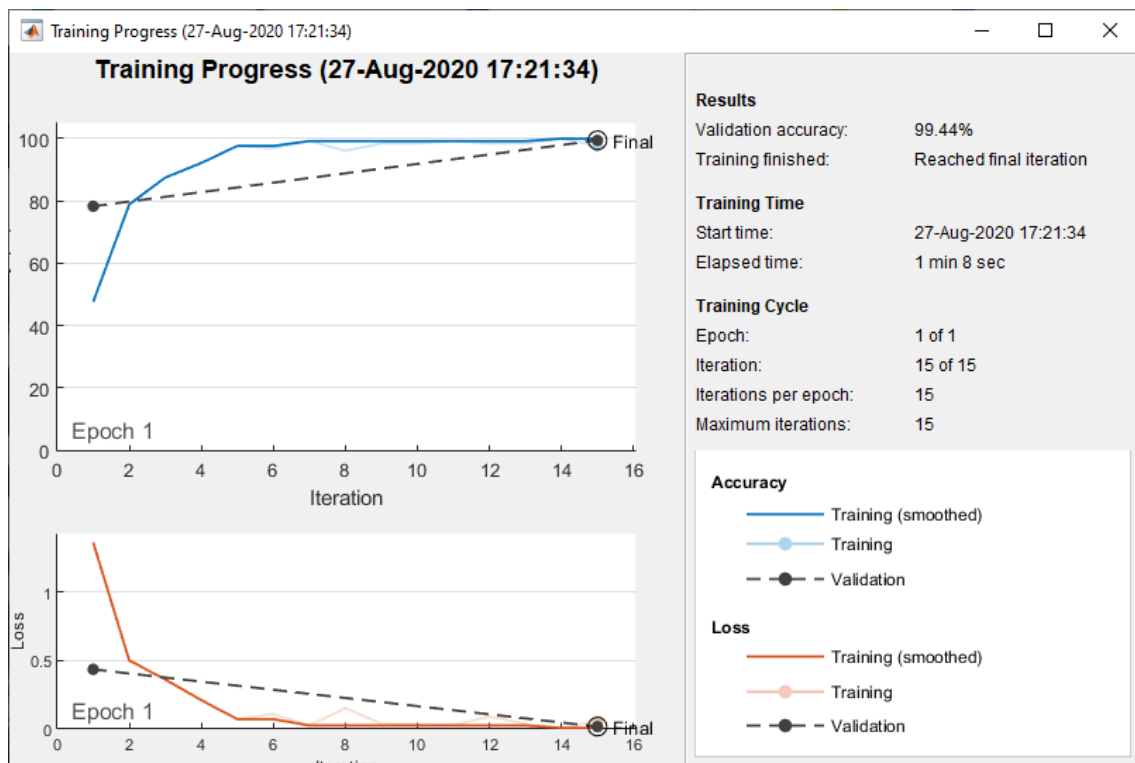


Figura 33 – Entrenamiento de la red AlexNet con 15 iteraciones.

En la figura 33 se muestran los gráficos de la precisión del entrenamiento y el porcentaje de error, en azul y en naranja respectivamente. Desde el inicio del entrenamiento, a partir de quinta iteración el entrenamiento observa muy estable a lo largo de todas las iteraciones.

A la vista de las dos imágenes anteriores se confirma que ampliar el número de iteraciones no mejoraba de manera considerable el entrenamiento. El tiempo total que ha tardado en el entrenamiento ha sido muy rápido, tan sólo de 1 minuto y 8 segundos, influido también por el ordenador en el que se realizó (ver apartado 3.2).

Una vez entrenada la red, procedemos a analizar la matriz de confusión.



True Class	gasa	1769	16
	no-gasa	21	4871
		gasa	no-gasa
		Predicted Class	

Figura 34 – Matriz de confusión de la clasificación utilizando Alexnet.

Fijándonos en la matriz de confusión, del total de las 6677 imágenes, se han analizado correctamente el 99,44%. Se han identificado de manera errónea 16 gasas, indicando que no eran gasas. Y 21 que no eran gasas y se han clasificado como que sí que eran gasas.

El repositorio de imágenes que disponemos, las imágenes de gasas están separadas en limpias y manchadas de sangre. Las primeras 717 imágenes del repositorio son gasas limpias, mediante un script explicado en los anexos procedemos a calcular el porcentaje de acierto.

Tipo	Clasificadas correctamente	Total	Precisión
Gasas Limpias	713	717	99,44%
Gasas con sangre	1055	1067	98,87%
No-gasas	4871	4892	99,57%

Tabla 5– Cuadro resumen de la precisión en la clasificación mediante Alexnet

En el caso de gasas limpias ha sido de un 99,44% y de gasas manchadas 98,87%. Por lo tanto, sí que se aprecia una mayor complicación a la hora de detectar gasas



manchadas de sangre, pero aun así de las 1067 imágenes, ha habido 12 que ha tenido problemas, por lo que sigue siendo una medida muy buena. La precisión total en la clasificación de las gasas es de un 99,10% y de la clasificación de las imágenes de fondo en la categoría 'no-gasa' es de un 99,57%.

Para calcular la sensibilidad hay que recurrir a la matriz de confusión utilizando la fórmula que viene a continuación:

$$\text{Sensibilidad en gasa} = \frac{\text{Gasas verdaderas}}{\text{Gasas verdaderas} + \text{Falso Nogasas}}$$

De manera que las gasas totales verdaderas (correctamente clasificadas) son 1769 y las no-gasas clasificadas erróneamente que son 21 imágenes. Por lo tanto, resulta una sensibilidad detectando gasas de un 98,82%. Este valor nos indica la probabilidad que hay de clasificar correctamente una gasa, es decir que la probabilidad de que en una detección sea un resultado de gasa correcto.

De igual manera se procede a calcular la sensibilidad en 'no-gasas' con la misma idea de la fórmula anterior:

$$\text{Sensibilidad en 'no - gasa'} = \frac{\text{'no - gasa' verdadero}}{\text{'no - gasa' verdadero} + \text{Falso gasa}}$$

La clasificación correcta en la categoría No-gasa fue de 4871 imágenes mientras que hubo 16 gasas detectadas como que no eran gasas, falsos positivos de gasas. La sensibilidad en no-gasa resulta ser un 99,67%, por lo tanto, la probabilidad de detectar el fondo, o cualquier cosa que no sea una gasa es más alta que la proporcionada al detectar las gasas, por consiguiente, es un dato a tener en cuenta de cara a elegir una red para su implementación en el quirófano.

El equilibrio entre la precisión y la sensibilidad se calcula con la siguiente fórmula:

$$F_1 = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{sensibilidad}}}$$

Para las gasas el valor de F_1 es de 98,95% y para el resto de las imágenes (no-gasa) es de un 99,61%.

Tipo	Sensibilidad	F_1
'gasa'	98,82%	98,95%
'no-gasa'	99,67%	99,61%.

Tabla 6 – Sensibilidad de la red Alexnet



Procedemos a hacer una revisión manual de las imágenes de gasa que no ha detectado, para poder sacar conclusiones. En la siguiente figura se muestran 6 imágenes de ejemplo de las 16 que ha detectado incorrectamente.

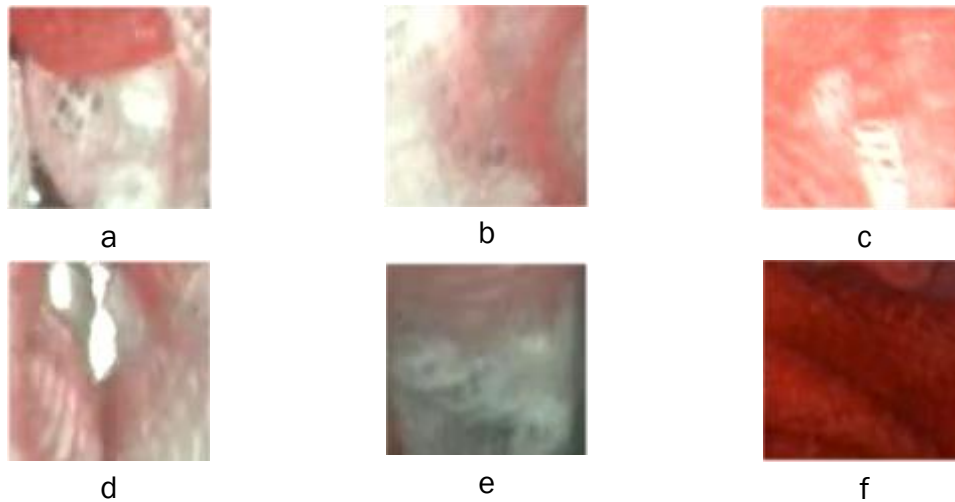


Figura 35 – Ejemplos de imágenes mal detectadas por Alexnet

Observando las imágenes, hay varias que no están claras ni siquiera a la vista del ojo humano. El denominador común es la presencia parcial de gasa o de elementos externos como en la imagen d que se aprecia la punta de un instrumento. No obstante, el algoritmo es muy preciso y tiene un porcentaje de error inferior al 0.7%.

Del mismo modo que hemos actuado con las gasas, investigaremos una breve tirada de imágenes que ha catalogado como gasa erróneamente.

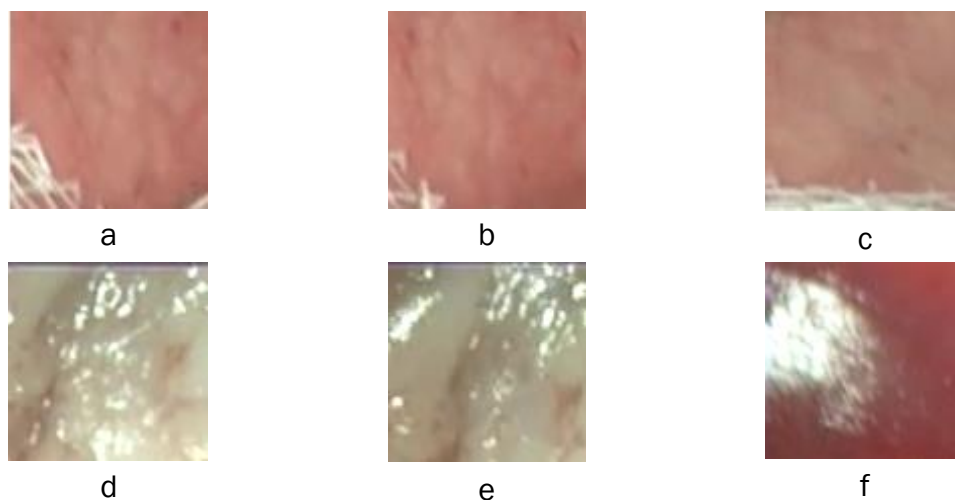


Figura 36 – Ejemplos de imágenes mal clasificadas por Alexnet

En las imágenes de la primera fila, se puede apreciar que efectivamente hay una gasa, por lo tanto, realmente, no está equivocado, pero para su clasificación, se ha



decidido que esa imagen sea catalogada como no-gasa, puesto que la mayor parte de la imagen es la pared del estómago. Los fotogramas d y e, son dos ejemplos donde se ha equivocado puesto que el color es semejante al de una gasa limpia. Finalmente, la imagen f, da un reflejo de la luz del laparoscopio, en lo que el ordenador ha detectado como una gasa.

La detección de las gasas ha sido muy efectiva, excluyendo estos ejemplos que se han comentado. La manera de poder mejorar el algoritmo es incluyendo más imágenes de muestra, sobre todo de reflejos en la pared del estómago, imágenes donde aparezcan instrumentos quirúrgicos e incluso más imágenes de órganos y vesículas blancas.

Con las condiciones de entrenamiento y el hardware empleado, el tiempo promedio de redimensionado y clasificación de 50 imágenes de gasas y de otras 50 imágenes de fondo fue de 0.612 segundos. Lo que equivaldría a 163 fotogramas por segundo procesando imágenes originales de resolución 100x100.

Imaginemos, un supuesto más cercano a la realidad, con imágenes procedentes de un laparoscopio, de dimensiones 720x576 y que el tiempo de procesamiento es lineal respecto a las dimensiones de las imágenes.

$$TotalPíxeles_{100x100} = 100 \times 100 = 10^4 \text{ píxeles}$$

$$TotalPíxeles_{720x576} = 720 \times 576 = 414,72 \times 10^3 \text{ píxeles}$$

La imagen en HD, dado el supuesto, tardará 42 veces más en ser procesada. Por lo tanto 100 imágenes procedentes del laparoscopio tardarían, en el ordenador donde se han realizado las pruebas 26 segundos. Consiguiendo una ratio de 3 fotogramas por segundo, en el mejor de los casos.

Visto esta comparativa, se abren 2 posibles vías de mejora, la primera y más obvia es la de mejorar el ordenador donde se implante el sistema para mejorar tiempos, y por otra parte se podría plantear bajar la resolución a 640x480 píxeles para poder clasificarlos en menos tiempo.

Tras la clasificación de las imágenes, se procede a comprobar la funcionalidad del algoritmo y de cómo responde a diferentes vídeos con gasas en el interior del cuerpo. Como se indicó anteriormente, se divide cada fotograma del vídeo en bloques 100x100, y se procede a clasificar fotograma a fotograma para posteriormente agruparlos en vídeo.



En la figura 37 se muestran dos fotogramas extraídos de un vídeo donde se observa la clasificación y el seguimiento de la gasa (cuadrados verdes), de manera muy precisa.

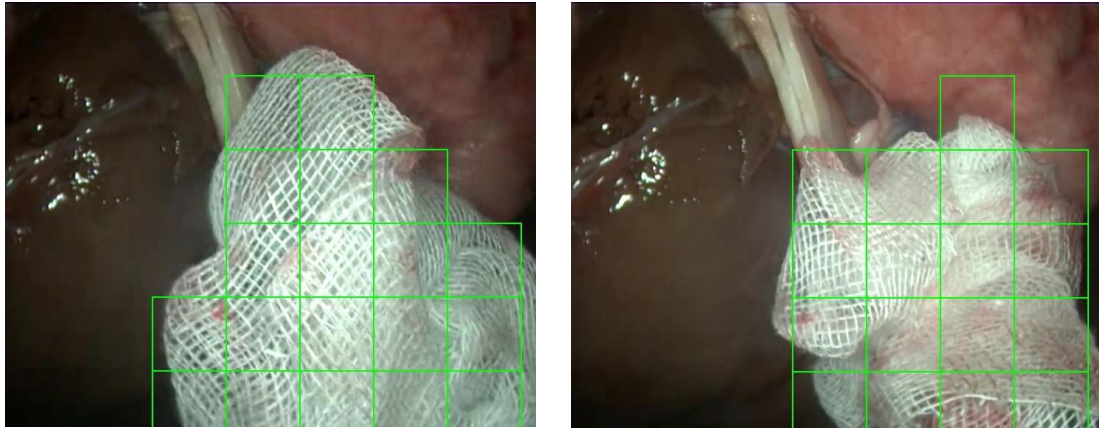


Figura 37 – Gasa clasificada en bloques 100x100 utilizando AlexNet.

La clasificación en general es muy buena, se han testeado diferentes vídeos con gasas limpias y manchadas y los resultados van en consonancia con lo anteriormente explicado. Hay reflejos de la luz del laparoscopio que el sistema lo clasifica como una gasa, hay momentos en los que en la pared también se equivoca, o fotogramas donde la gasa no está completa y no se tiene en cuenta.

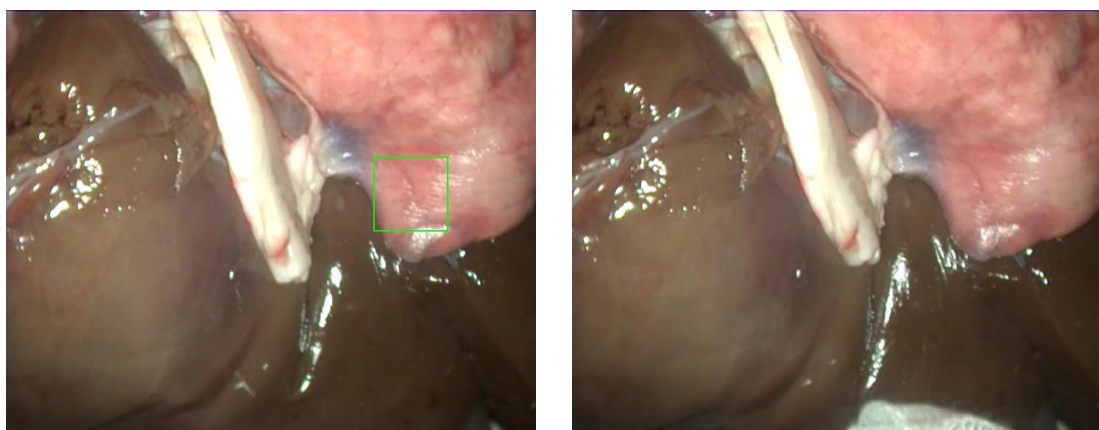


Figura 38 – Ejemplos de clasificaciones parcialmente erróneas. En la imagen de la izquierda ha detectado una gasa cuando no la había y en la imagen de la derecha aparece por la parte inferior una gasa que no ha detectado probablemente por su escasa presencia en la imagen.

En las imágenes de la figura 38 se presentan 2 fotogramas consecutivos donde ha habido un fallo de clasificación en el primero. Lo ideal sería que la red tuviera un



100% de precisión, y es a lo que se propone llegar en un futuro con un banco de imágenes de gasas más grande, no obstante, la red clasifica de media en torno a 2 segundos por imagen.

A continuación, se prueban diferentes vídeos con gasas manchadas en este caso:

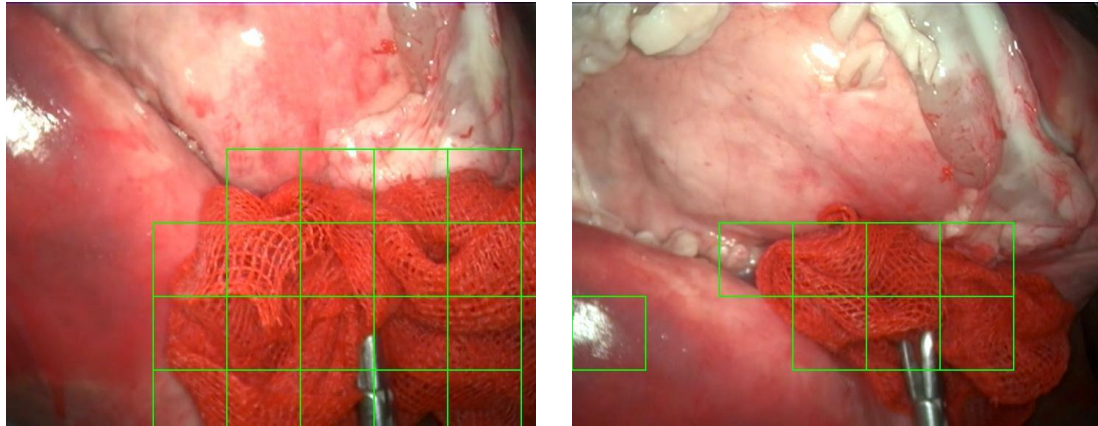


Figura 39 – Fotogramas de detecciones en gasas manchadas de sangre.

La clasificación en todo el vídeo ha sido muy buena, ha tenido pocos falsos positivos, el más común ha sido el mostrado en la imagen de la derecha de la figura 39, donde el reflejo de la parte inferior izquierda ha sido clasificado como gasa. En el apartado de líneas futuras (Capítulo 5) se propone un método para evitar que esto sea clasificado como gasa, principalmente basado en seleccionar más imágenes de reflejos para el entrenamiento.

Para la implementación del algoritmo en tiempo real, con el Hardware disponible, sería posible a una tasa de 0.5 fotogramas por segundo, lo que resultaría muy engorroso y poco práctico, aunque útil si el laparoscopio se maneja con suavidad y lento. No obstante, para mejorar estos tiempos con un hardware con una GPU dedicada más potente podría implementarse perfectamente en un quirófano.

4.2 RESNET50

Continuando con el experimento, se procede a testear la red RedNet-50. Ésta consta de 177 de capas, y para reentrenar la red se han modificado, tal y como se explicó en el capítulo anterior, las tres últimas capas. Las imágenes se redimensionan a 224 x224 píxeles, que son las dimensiones con las que trabaja la capa de entrada de la red. Al aumentar el número de capas, también aumenta el tiempo de procesamiento con respecto a Alexnet. El entrenamiento ha tardado 16 minutos y 7 segundos y ha dado como resultado una precisión del 99,55%.

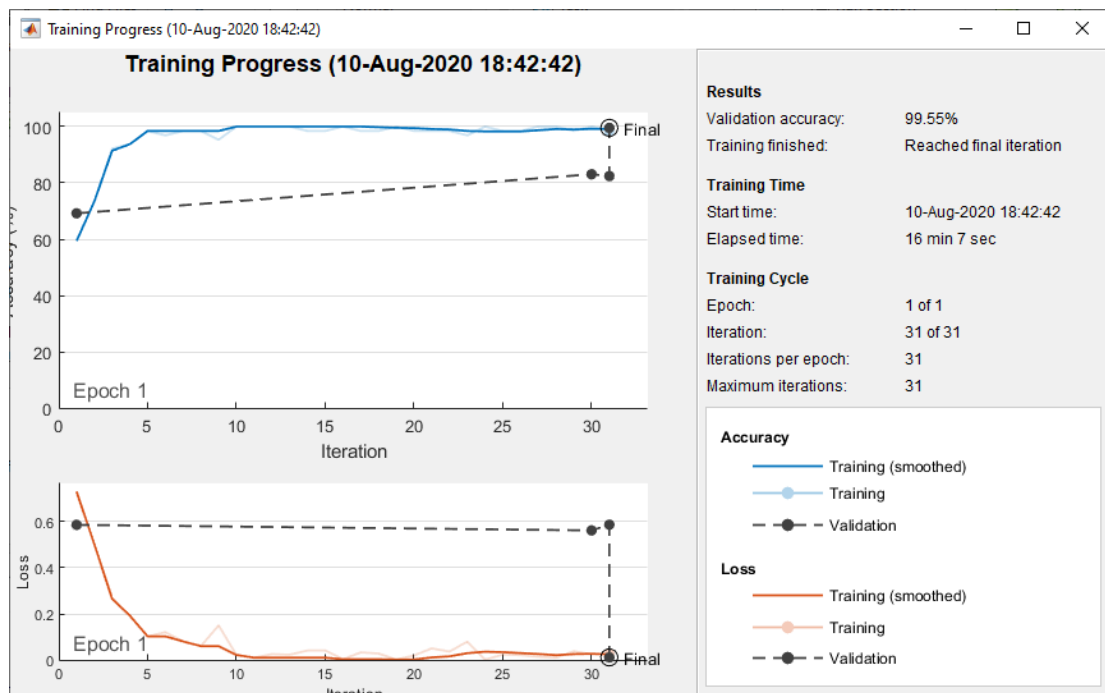


Figura 40 – Entrenamiento de la red mediante ResNet50

Tras entrenar la red, se procede a utilizar las imágenes de prueba para comprobar la precisión de la red dentro de los diferentes tipos de gasas, ya sean manchadas o limpias. La matriz de confusión del total de imágenes resulta:

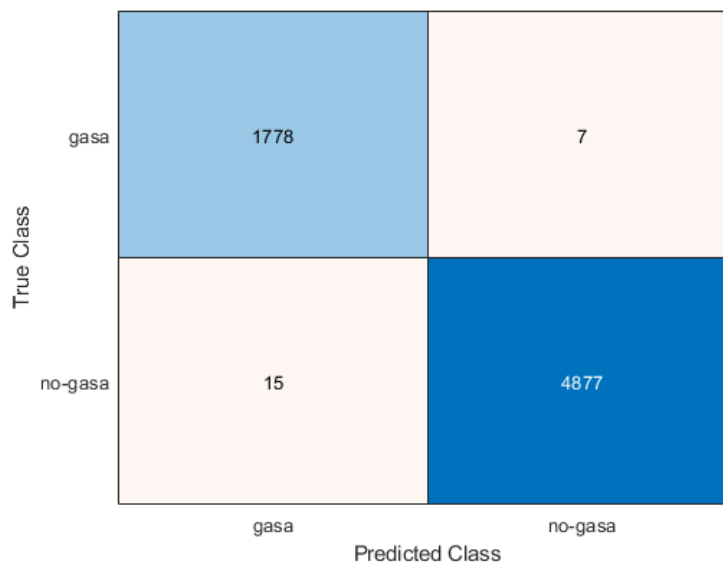


Figura 41 – Matriz de confusión de la red ResNet50

Se han analizado un total de 6677 imágenes, de las cuales se han detectado 6655 de manera correcta. Los 22 restantes, se dividen en 2 tipos de errores. Hubo 7 gasas



que fueron clasificadas como no gasas y 15 no-gasas que se predijo que sí que lo eran. Se procede a inspeccionar de manera visual, qué imágenes no ha detectado correctamente la red neuronal para poder extraer conclusiones. El porcentaje de acierto dentro de las gasas manchadas de sangre fue de 99.53% mientras que de las limpias fue de un 99.72%.

Tipo	Clasificadas correctamente	Total	Porcentaje de acierto
Gasas Limpias	715	717	99,72%
Gasas con sangre	1062	1067	99,53%
Fondo	4877	4892	99,69%

Tabla 7- Cuadro resumen de la clasificación de la red mediante ResNet50

Este porcentaje de gasas limpias arroja un mejor resultado en las gasas manchadas con sangre respecto a Alexnet, reduciendo a la mitad los 11 resultados erróneos.

Una vez calculada la precisión de las gasas, se procede a calcular un parámetro muy importante para ver la probabilidad que hay de detectar una gasa cuando haya una gasa. Utilizando la matriz de confusión calculamos la sensibilidad en gasa como:

$$\text{Sensibilidad en gasa} = \frac{\text{Gasas verdaderas}}{\text{Gasas verdaderas} + \text{Falso Nogasas}}$$

Las gasas verdaderas son 1778 y el falso positivo en no-gasas son 15, por tanto, la sensibilidad en gasa es de un 99.16% mientras que en no gasa se tiene:

$$\text{Sensibilidad en 'no - gasa'} = \frac{\text{'no - gasa' verdadero}}{\text{'no - gasa' verdadero} + \text{Falso gasa}}$$

La clasificación de no-gasa correcta ha sido de 4877 imágenes, y los falsos de gasas han sido 7. La sensibilidad en 'no-gasa' es de un 99,85%.

Para ver la relación entre la sensibilidad y precisión empleamos el estadístico F_1

$$F_1 = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{sensibilidad}}}$$

El F_1 para las 'gasas' es de 99.38% y para las 'no-gasa' de 99,77%.



Tipo	Sensibilidad	F ₁
'gasa'	99,16%	98,38%
'no-gasa'	99,85%	99,77%.

Tabla 8 – Sensibilidad de la red ResNet

A continuación, se realiza una revisión de algunas imágenes que ha dado erróneo el algoritmo dentro de gasas.

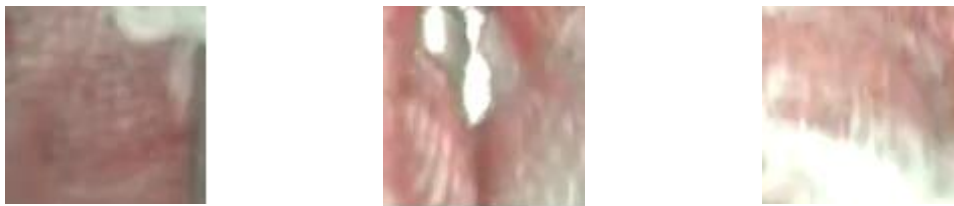


Figura 42 – Ejemplos de imágenes mal clasificadas en la red

Estas gasas han sido catalogadas como manchadas de sangre, puesto que no se ven blancas totalmente. La imagen central vuelve a aparecer como mala clasificación, es una imagen que podría ser retirada del banco de imágenes, pero es una condición que se puede dar en una situación real, por lo que conviene dejarla y manejarla.

Se pueden observar varios casos generales, el primer caso encontrado son falsos negativos, es decir, imágenes que se han clasificado como no-gasa por la escasa aparición de gasa en la imagen. Un ejemplo se muestra a continuación:

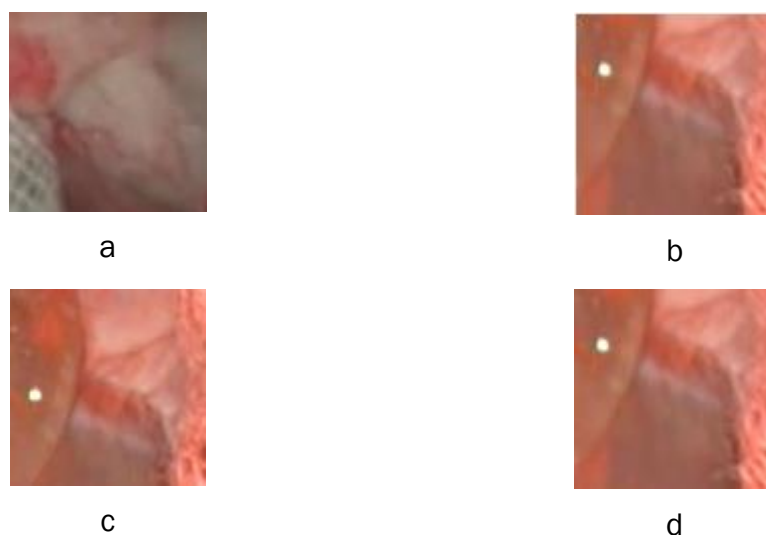


Figura 43 – Imágenes clasificadas cómo gasas a pesar de la escasa presencia de esta.



En los cuatro ejemplos anteriores el factor común es que hay una mínima presencia de gasa. Por lo tanto, el algoritmo es aún mejor que lo que muestran las estadísticas automáticas si nos fijamos en la clasificación. En este caso, es capaz de detectar incluso una presencia de gasa menor de 10% de la imagen.

El otro caso detectado de errores viene por los reflejos. En las siguientes figuras se aprecian estos reflejos en la pared del estómago.

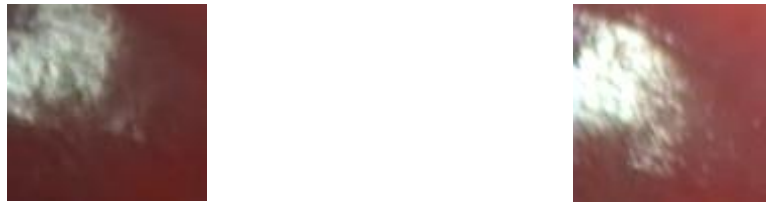


Figura 44 – Error típico de clasificación como si fuera una gasa en presencia de reflejos.

Esos reflejos darían como resultado un falso positivo indicando que habría una gasa, pero al mover el laparoscopio rápidamente el sistema indicará que no lo había.

En el algoritmo de clasificación se ha incluido una variable que almacena la precisión con la que se ha clasificado cada imagen inicial. Se ha realizado un script sencillo que se puede consultar en los anexos, al igual que el resto de código, para visualizar las imágenes que se ha clasificado, pero con un porcentaje <60%. Porcentaje el cual se ha decidido arbitrariamente.

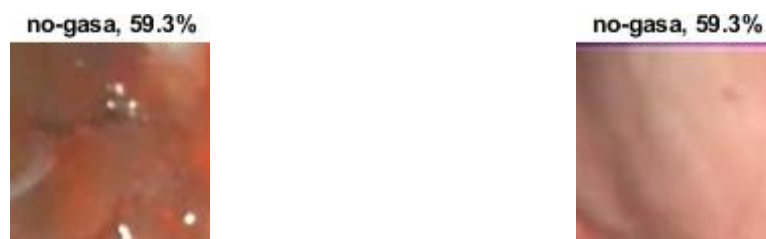


Figura 45 – Imágenes con porcentaje inferior al 60% de confianza en la clasificación.

El tiempo promedio que ha empleado el ordenador en realizar una clasificación de 100 imágenes escogidas aleatoriamente de un tamaño 100 x 100 ha sido de 2,49 segundos. Tiempo casi 4 veces superior al empleado por Alexnet.

Tras el entrenamiento y la comprobación de la precisión de los bloques, se procede a implementar el algoritmo para rastrear gasas en vídeos. Se utiliza el procedimiento de segmentar en bloques explicado en el capítulo 3, y se testea en 4 vídeos diferentes con distintos tipos de gasas y reflejos en el fondo para ver los resultados finales.

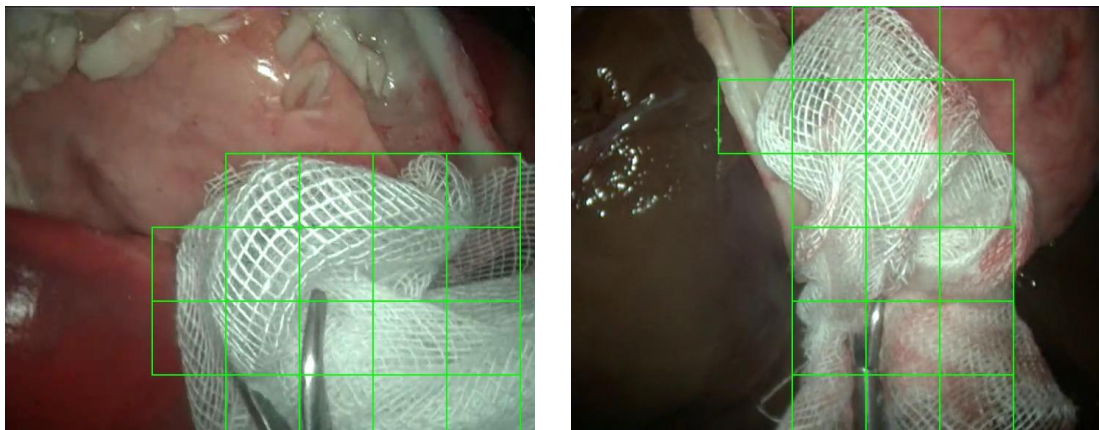


Figura 46 – Ejemplo de clasificación de gasas limpias en vídeo.

En la figura 46, se muestran dos fotogramas con la clasificación de las gasas de manera muy precisa y correcta, ha pasado la prueba de no detectar reflejos, como en la imagen a, donde se ve un reflejo en la zona izquierda, y también ha detectado la gasa incluso con la presencia de una pinza quirúrgica.

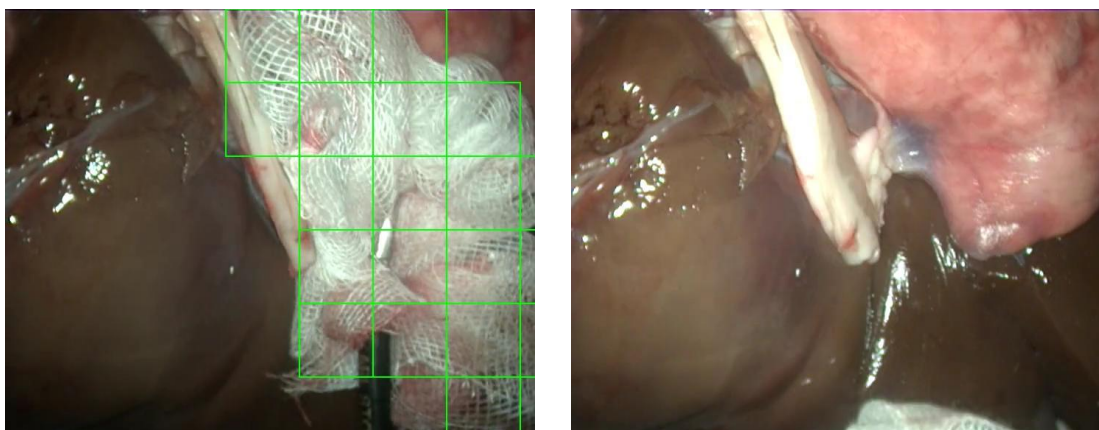


Figura 47 – Ejemplo de clasificación de gasas limpias en vídeo.

Esta detección ha sido adecuada y se complementa con los fotogramas de la figura 47. En la imagen de la izquierda, no ha detectado ningún falso positivo referido a los brillos y en la imagen de la derecha, no ha detectado ninguna falsa gasa, por lo que la clasificación ha sido correcta y precisa.

Continuando con la clasificación, se pasa a analizar el vídeo número 2, donde se aprecian diferentes zonas y vísceras blancas para comprobar como de robusta es el rastreo de la gasa.

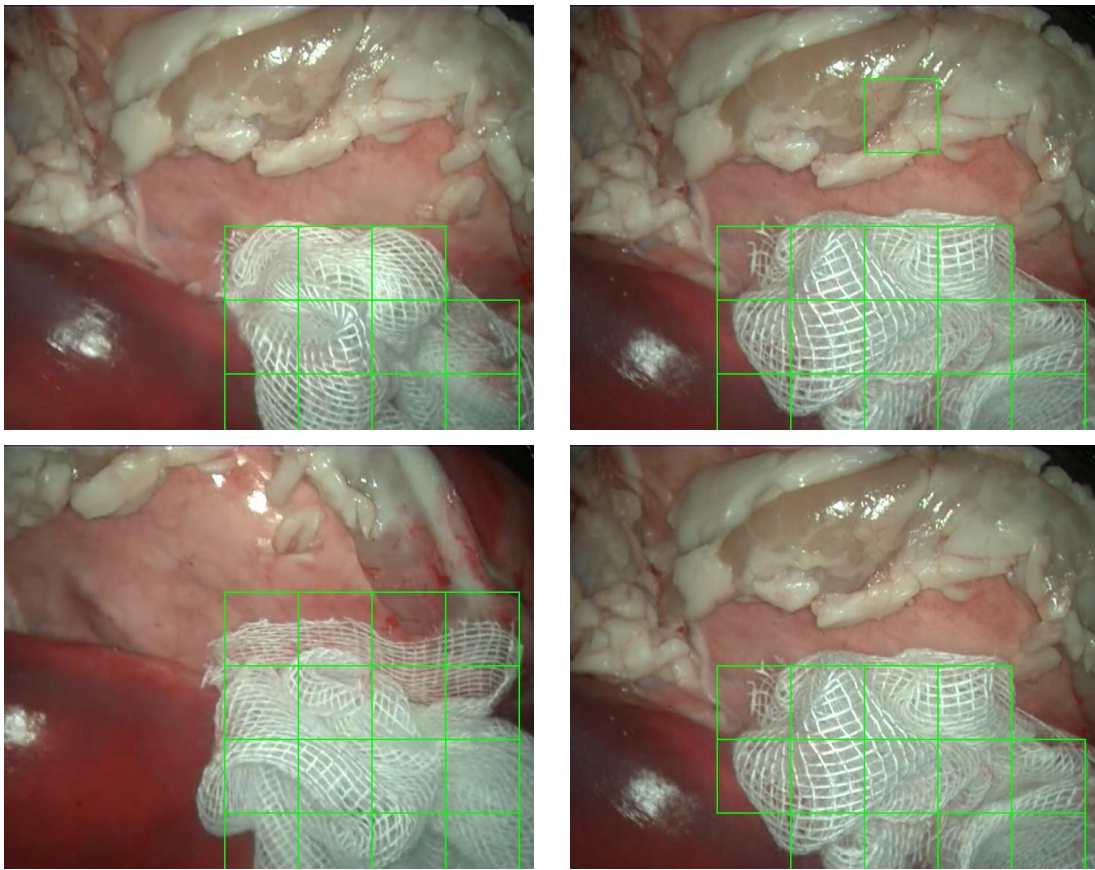


Figura 48 – Ejemplo de clasificación de gasas limpias en vídeo.

En los cuatro fotogramas seleccionados de la Figura 48, el rastreo ha sido muy bueno. Los reflejos de la parte izquierda no los ha detectado erróneamente por lo que es un dato positivo para destacar, pero por el contrario ha detectado en un fotograma, concretamente en la imagen de arriba a la derecha, es un fallo muy puntual que no se ha repetido en todo el vídeo, por lo que no es muy relevante.

Finalmente se pasan a analizar los fotogramas de un vídeo con las gasas empapadas en sangre y fluidos.

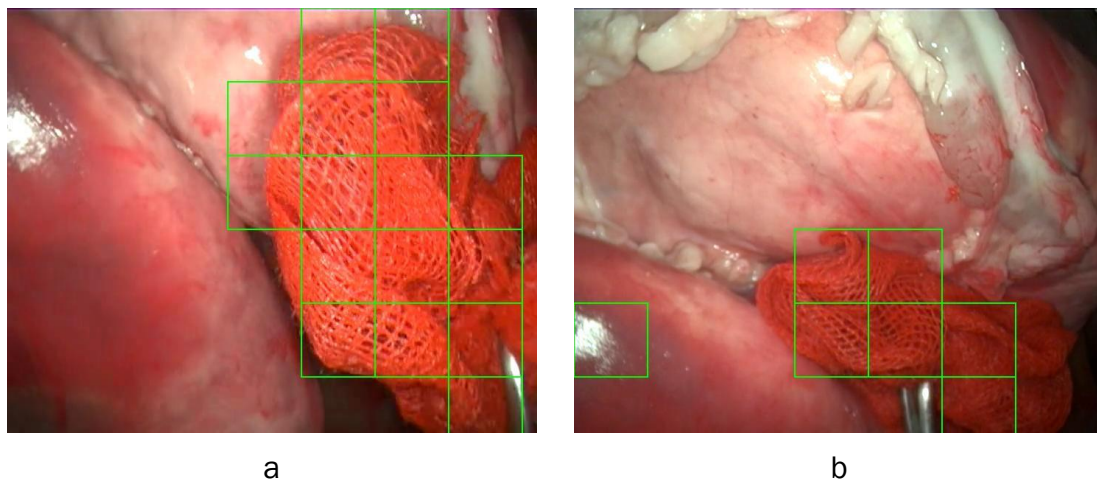


Figura 49 - Ejemplo de clasificación de gasas manchadas de sangre en vídeo.

Los fotogramas que se han seleccionado en la figura 49, muestran 2 buenas clasificaciones, con una alta sensibilidad a las gasas y buena precisión a pesar de faltar 2 bloques en la zona derecha de la imagen b. El fotograma b se ha incluido porque muestra un caso puntual, que sólo se ha dado en este segundo, pero está en consonancia con los datos analizados anteriormente y ha dado un falso positivo de gasa en la parte izquierda de la imagen b. Pertenecer a un reflejo, un fallo muy común observado en los entrenamientos de las redes. La mejor manera de solucionarlo sería incluir en las imágenes de entrenamiento, imágenes de reflejos semejantes a esta indicando que no se trata de una gasa. Esto quedará reflejado en el capítulo 5 en trabajos futuros.

4.3 GOOGLNET

De igual manera que en los apartados anteriores, se procede a implementar el sistema de gasas quirúrgicas mediante la red neuronal convolucional GoogLeNet. GoogLeNet está preparada para clasificar 1000 categorías, como teclados, ratones y multitud de animales, pero no está entrenada para detectar gasas. Por esta razón hay que realizar cambios en sus capas finales, para poder adaptar la red y que se pueda reentrenar.



Para reentrenar la red, modificamos la capa *Fully Connected* y la llamamos 'new_fc', le damos a esa capa un peso de 10. Además de cambiar la última capa de clasificación donde se incorporan las categorías gasa y no-gasa.

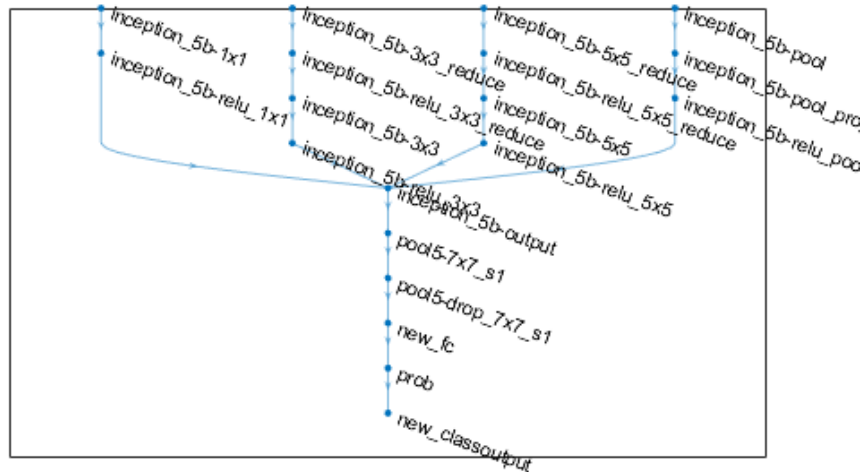


Figura 50 – Esquema de las últimas capas de GoogLeNet

Una vez hechos los cambios en la red, se procede al reentrenamiento de la red. Las especificaciones son las mismas que se han realizado en los anteriores experimentos, 1000 imágenes de gasas y 1000 de fondo del estómago.

Tasa de aprendizaje inicial	0.001
Número máximo de “épocas”	2
Frecuencia de validación	30

Tabla 9 – Opciones de entrenamiento de la red GoogLeNet

El tiempo total que ha llevado han sido 3 minutos y 55 segundos. Y la precisión que se ha conseguido ha sido de 98.85%, un poco inferior que Alexnet y Resnet.

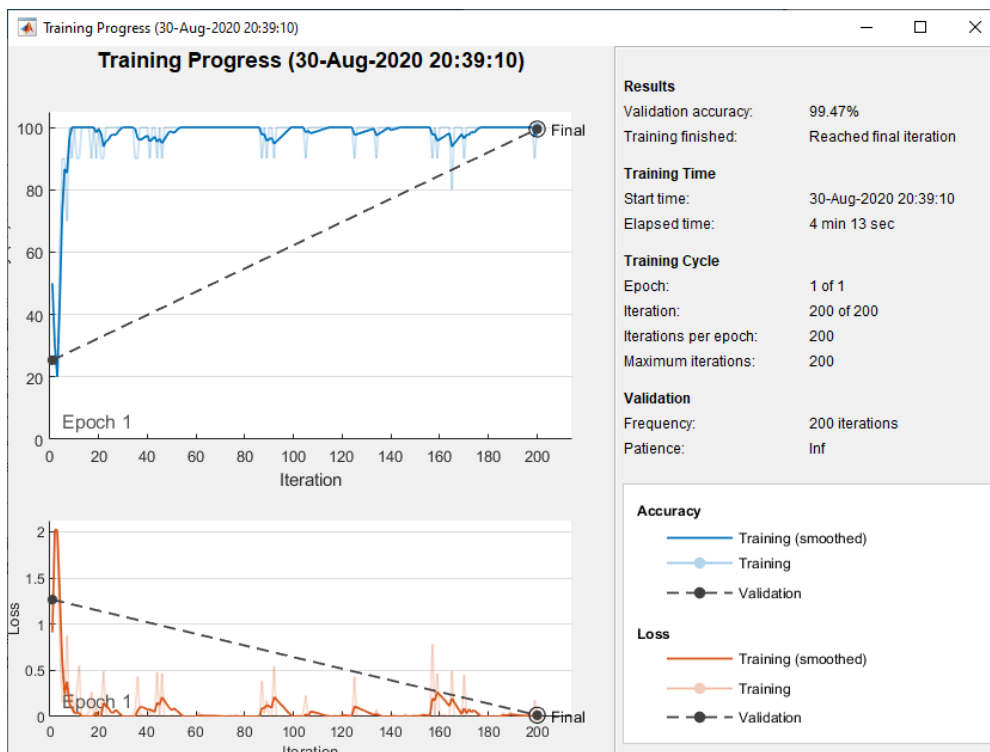


Figura 51 – Resultados del reentrenamiento de GoogLeNet.

A continuación, se presenta su matriz de confusión:

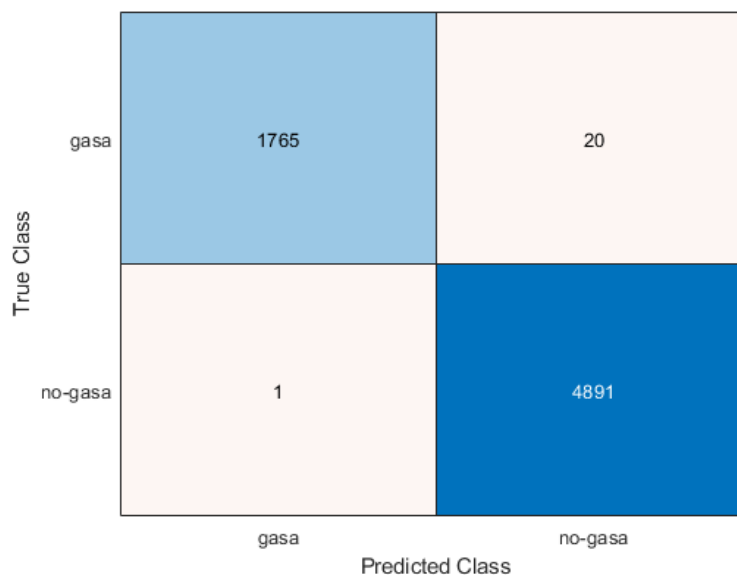


Figura 52 – Matriz de confusión del entrenamiento de GoogLeNet.

Analizando todas las imágenes, se observa que, del total de gasas, veinte fueron clasificadas erróneamente indicando que no lo eran, y que únicamente una imagen



del fondo que no era gasa se clasificó como que sí lo era. Respecto a esas veinte gasas no detectadas, 3 fueron gasas limpias y el resto, 17, eran gasas manchadas con sangre. Se procede a calcular la sensibilidad, parámetro que nos da la probabilidad que hay de clasificar una 'gasa' o un 'no-gasa' de manera correcta.

$$\text{Sensibilidad en gasa} = \frac{\text{Gasas verdaderas}}{\text{Gasas verdaderas} + \text{Falso Nogasas}}$$

Con la ayuda de la matriz de confusión, las gasas verdaderas son 1765 y el falso 'no-gasa' es uno, por tanto, la sensibilidad en gasa es un 99,99%. Esto quiere decir que utilizando GoogLeNet, hay un 99,99% de identificar una gasa.

Procediendo con la misma fórmula, la sensibilidad en 'no gasa'

$$\text{Sensibilidad en 'no - gasa'} = \frac{\text{'no - gasa' verdadero}}{\text{'no - gasa' verdadero} + \text{Falso gasa}}$$

Siendo los 'no-gasa' verdaderos 4891 y los falsos de gasa 20, la sensibilidad es un 99,59%. La sensibilidad de esta red es más alta para gasa que para no-gasa, aspecto muy positivo de cara a poder rastrear correctamente las gasas. Para comprobar la relación entre la sensibilidad y la precisión se recurre al F_1 .

$$F_1 = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{sensibilidad}}}$$

El F_1 de las 'gasas' es de un 99,43%, mientras que de las 'no-gasas' fue de un 99,79%. Este indicador tendrá una importante relevancia a la hora de elegir una red para el rastreo de las gasas y se analizará en el apartado 4 en comparación con las otras dos redes.

Tipo	Sensibilidad	F_1
'gasa'	99,99%	99,43%
'no-gasa'	99,59%	99,79%.

Tabla 10 - Sensibilidad en la red GoogLeNet.



A continuación, se muestran los porcentajes de precisión en función de lo que esté o no manchada la gasa.

Tipo	Clasificadas correctamente	Total	Precisión
Gasas Limpias	714	717	99,58%
Gasas con sangre	1050	1067	98,40%
No-gasas	4891	4892	99,97%

Tabla 11 - Cuadro resumen de la clasificación de la red mediante ResNet50.

Del total de gasas limpias solamente 3 fueron identificadas erróneamente, en la figura 53 se muestran cuáles son:



Figura 53 – Imágenes clasificadas como 'no gasa'.

En la imagen a de la figura 53 la presencia de gasa es muy poco visible, tan sólo en la esquina superior derecha, es razonable que la red no la haya detectado. En la imagen b, más de la mitad de la imagen es gasa por lo que sí que se debería de haber detectado, al igual que en la c. La imagen C es la que más puede preocupar ya que en su totalidad se aprecia la gasa, por lo que se ha procedido a revisar manualmente el porcentaje que ha tenido de certeza en la clasificación, resultando tan sólo un 52%, por lo que se puede considerar un error puntual.



Figura 54 – Gasas clasificadas de manera errónea.



Siguiendo con el análisis de los errores, lo siguiente que se va a comprobar son las gasas manchadas con sangre y fluidos. Se han identificado correctamente 1050 de 1067, por tanto, ha fallado en 17 imágenes, un 1.6%. Las tres imágenes de la figura 54 son tres ejemplos de gasas que la red neuronal GoogLeNet ha clasificado incorrectamente y que pertenecen a las gasas manchadas con sangre.

La imagen 'a' y la imagen 'c' de la figura 54 son ejemplos que se han escogido del total de las 17 imágenes que representan en su mayoría el tipo de imagen de gasa con sangre que más se ha repetido en la clasificación errónea. La segunda imagen es ya recurrente en los análisis exhaustivos que se han realizado en las anteriores redes convolucionales, y la presencia del laparoscopio capta toda la atención de la clasificación y obvia que haya una gasa. La precisión respecto al total de gasas sin sangre ha sido de un 98.88%, mientras que para las gasas manchadas de sangre ha sido de un 98.40%.

Tras estos resultados, se procede a comprobar de igual manera visualmente la única imagen que la red ha clasificado de manera errónea. Se trata de una imagen clasificada como fondo que ha sido identificada como gasa.



Figura 55 – Imagen clasificada como gasa erróneamente

La imagen de la figura 55, es un tipo de imagen que en las redes Alexnet y ResNet también ha detectado de manera errónea y se trata de un reflejo de la luz del laparoscopio en la pared del estómago que la red interpreta como una gasa. Una mejora de cara a la detección de este tipo de imágenes para hacer más robusta la red, consistiría en realizar varios experimentos presenciales donde se haga una captura de imágenes de reflejos en la pared de los órganos y se cataloguen como 'no-gasa', y con ello se entrene la red. Esta propuesta, entre otras se analiza en profundidad en el capítulo 5.

En la siguiente figura se han propuesto cuatro ejemplos de imágenes aleatorias, donde se indica el porcentaje de confianza que ha tenido la red en la clasificación de estas, es decir en la primera imagen, ha tenido un 100% de seguridad en que en esa imagen no había una gasa.

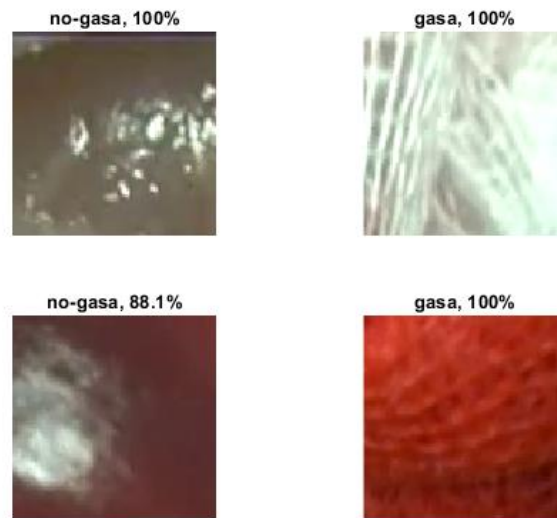


Figura 56 – Ejemplos de porcentajes de seguridad en la clasificación con GoogLeNet.

A parte de esto hay imágenes que han sido clasificadas correctamente, pero con un porcentaje de seguridad en la clasificación de menos de un 60%. Son imágenes que contienen reflejos en la pared del estómago o gases muy manchadas de sangre. A continuación, se muestran varios ejemplos:

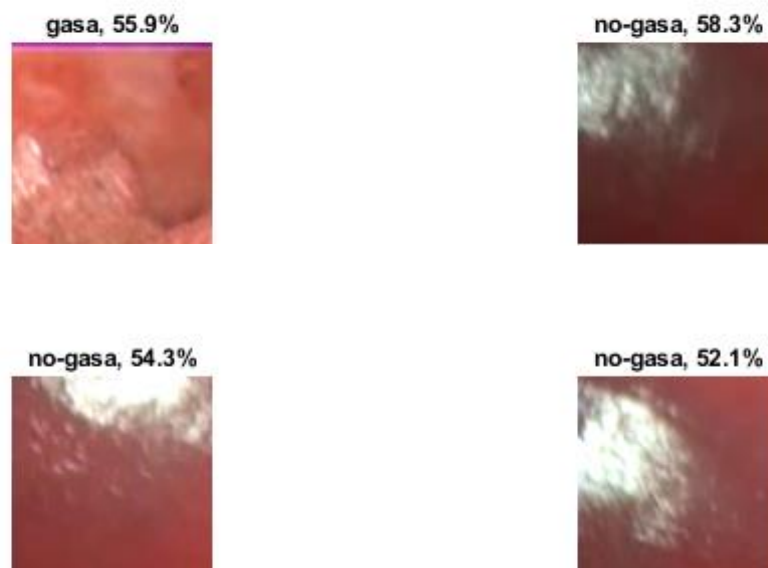


Figura 57 – Imágenes clasificadas correctamente con un porcentaje de seguridad menor del 60%.



Se comprueba la robustez en el seguimiento mediante un rastreo de las gasas en varios vídeos de ejemplo y se muestran los fotogramas más significativos en la figura 58 con los resultados del rastreo.

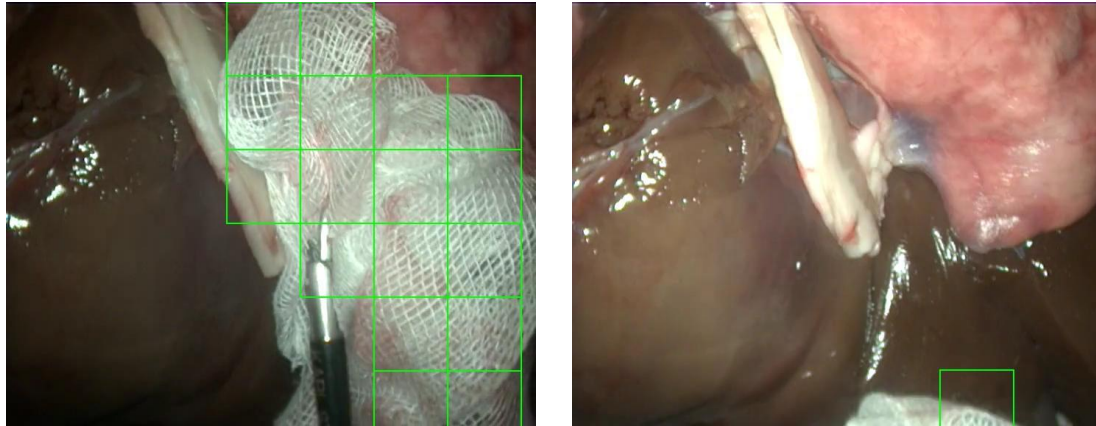
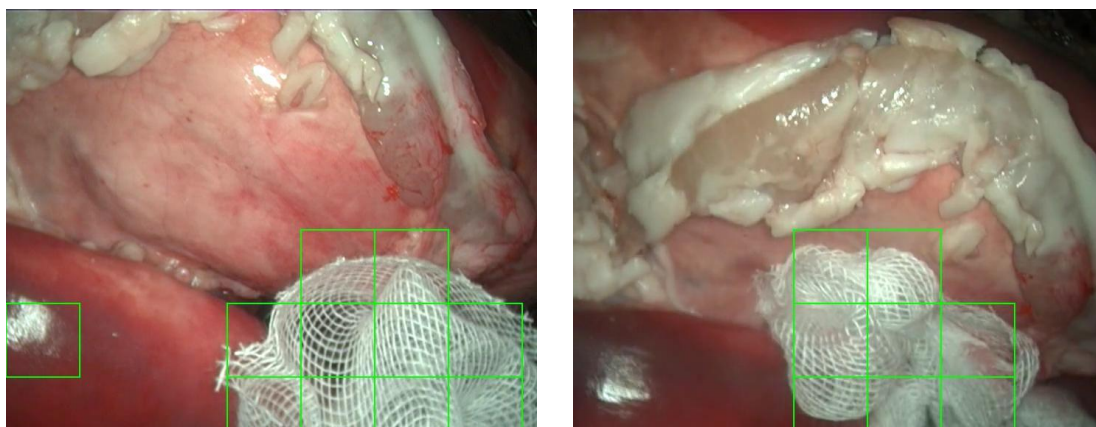


Figura 58 – Fotogramas de ejemplo clasificando un vídeo con GoogLeNet

En el primer vídeo se han seleccionado esos fotogramas clave que muestran la buena precisión y sensibilidad en la clasificación de la gasa, en la primera imagen hay un bloque incluso que tiene presencia de una pinza quirúrgica delante de la gasa y ha sido identificada. En el fotograma de la derecha de la figura 58, GoogLeNet ha detectado parte de una gasa de manera efectiva incluso apareciendo en menos de la mitad del bloque, la razón de esto es la alta sensibilidad a las gasas que ha demostrado estar entrenada GoogLeNet.

A continuación, en la figura 59, se escogen diferentes fragmentos del segundo vídeo de prueba, en el que aparecen zonas del fondo donde otras redes han demostrado tener falsos positivos confundiendo el fondo o los reflejos de la luz del laparoscopio. Del total de fotogramas, tan sólo en el primero que se muestra ha dado un falso positivo en el reflejo de la pared del estómago.



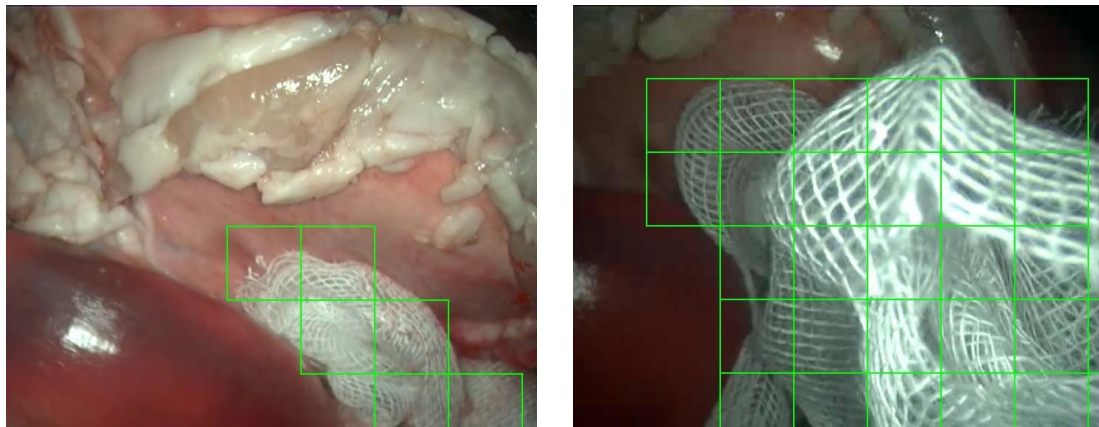


Figura 59 - Fotogramas de ejemplo clasificando un vídeo con GoogLeNet

En el siguiente vídeo, representado en cuatro fotogramas de la figura 60, hay presencia de gasa manchada de sangre y fluidos. Además, hay reflejos en el fondo y zonas de vesículas blancas. La detección ha sido muy precisa, no ha dado el falso negativo del reflejo y el rastreo está correcto.

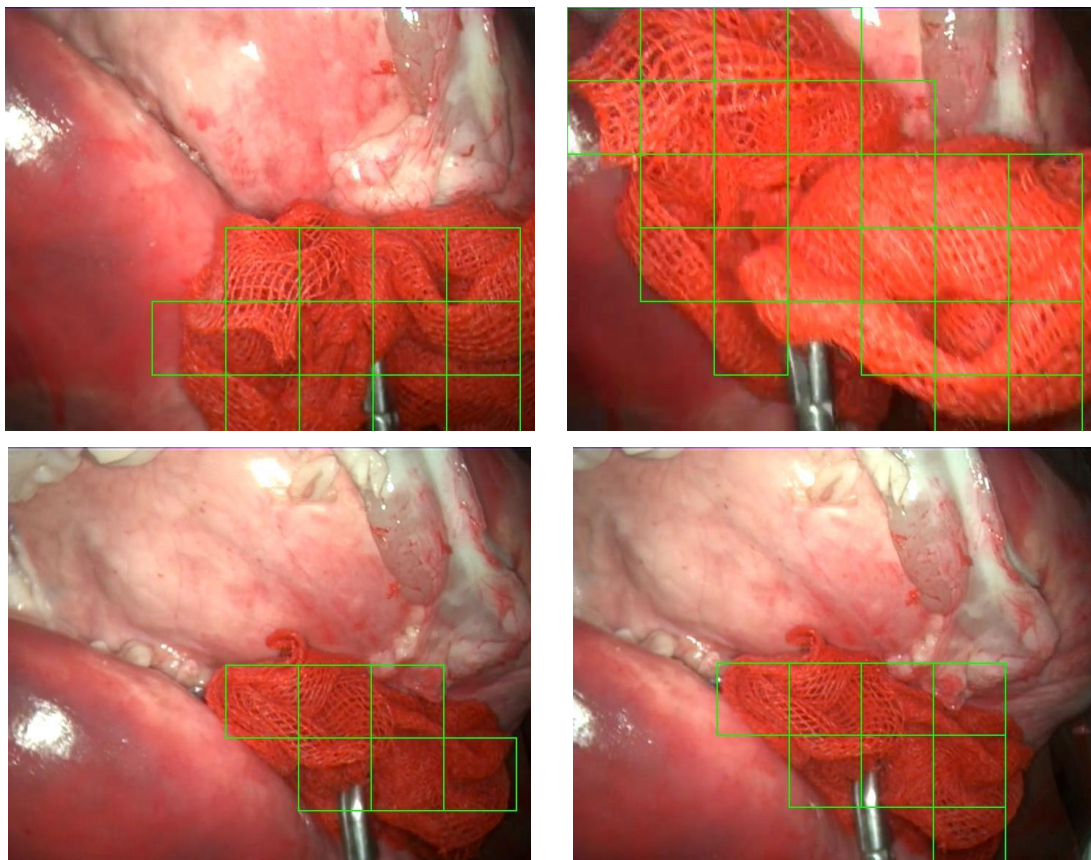


Figura 60 - Fotogramas de ejemplo clasificando un vídeo con GoogLeNet



A la vista de estos resultados, el rastreo de las imágenes de la red ha sido muy satisfactorio, se obtuvo una precisión del 99,47% y se ha comprobado mediante la aplicación a vídeos que la clasificación de los bloques ha sido un éxito; incluida la clasificación de las gasas manchadas de sangre que comparadas con las limpias la red clasificaba peor. Es una red muy robusta, adecuada para la detección de gasas por su alta sensibilidad (<99%) y precisión para la clasificación, por lo que estaría bien tenerla en cuenta a la hora de elegirla en una posible implementación.

4.4 COMPARATIVA

El entrenamiento de las redes Alexnet, ResNet-50 y GoogLeNet y la clasificación de las imágenes en 'gasa' y 'no-gasa' ha sido muy positivo, con unos resultados de tiempos de procesamiento, de entrenamiento, sensibilidad y precisiones muy buenos.

Los valores de la comparativa son realizados con el hardware previamente mostrado, es muy posible que, mejorando las características técnicas del ordenador en este caso, se consigan datos mejores conforme sobre todo a tiempos de procesamiento, no obstante, el equipo utilizado es de gama media-alta, por lo que los resultados están cercanos a los más altos que se podrían conseguir actualmente.

Con respecto a la precisión de clasificación de las imágenes, se ha dividido en 3 tipos de imágenes:

	Alexnet	ResNet-50	GoogLeNet
Gasas sin sangre	99,44%	99,72%	99,58%
Gasas manchadas de sangre	98,87%	99,53%	98,40%
Fondo	99,57%	99,69%	99,97%

Tabla 12 - Precisión de clasificación en las redes analizadas.

A continuación, se presenta el gráfico con los datos de la tabla anterior de la precisión en la clasificación de las imágenes correspondientes a gasas limpias, gasas manchadas y a imágenes del interior del estómago:

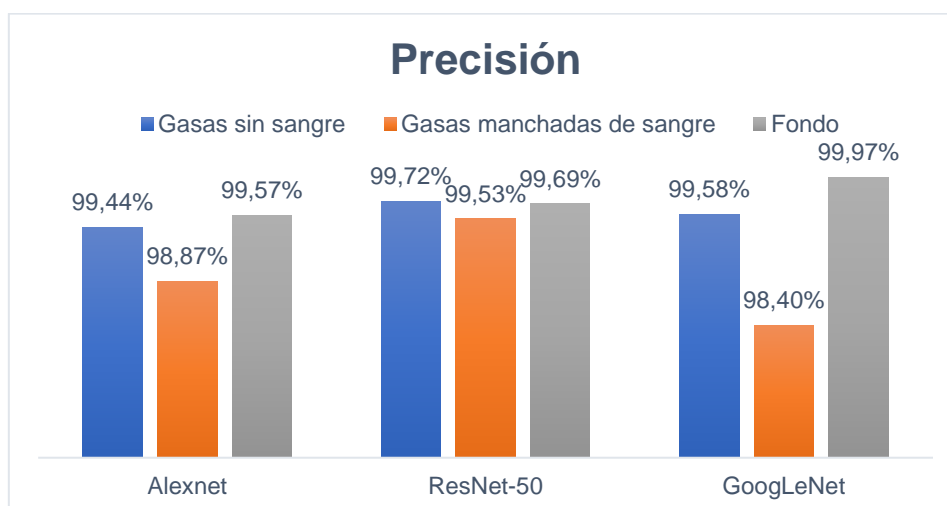


Figura 61 – Comparativa de la precisión al clasificar diferentes tipos de imágenes mediante redes neuronales.

De manera gráfica, en la figura 61, se pueden extraer conclusiones en cuanto a cuál sería la mejor a la hora de clasificar las gasas, si nos basamos únicamente en la capacidad de detección de las gasas, ResNet es la que ha emitido los resultados más favorables, con un 99,72% de precisión en gasas limpias y un 99,53% en gasas manchadas de sangre. El siguiente puesto se atribuiría a GoogLeNet con un 99,58% y un 99,53% de gasas limpias y manchadas, respectivamente. Finalmente, Alexnet es la que ha arrojado los peores resultados respecto a la detección de gasas en esta comparativa, sin embargo, es la que más precisión ha conseguido respecto a la identificación de lo que no es gasa, y forma parte del estómago.

Además de la precisión, hay que tener en consideración la sensibilidad en la clasificación, tanto detectando gasas como detectando el fondo. La sensibilidad indica la probabilidad de que la red detecte una gasa en caso de que la haya. La sensibilidad tiene en cuenta las detecciones de, es lo que lo diferencia de la precisión. En el caso de la detección de gasas, los falsos positivos (que indique que hay una gasa, y no la había) no son tan importantes como los falsos negativos, es decir, que el sistema detecte una gasa cuando en verdad no la hay es menos grave a que la haya y no la detecte. Por eso, el porcentaje de sensibilidad en gasa es mejor que cuanto más alto sea en gasa y más bajo en no-gasa.

	Alexnet	ResNet-50	GoogLeNet
'Gasa'	98,82%	99,16%	99,99%
'No-gasa'	99,67%	99,85%	99,59%

Tabla 13 – Sensibilidad en la clasificación de las redes AlexNet, ResNet y GoogLeNet.

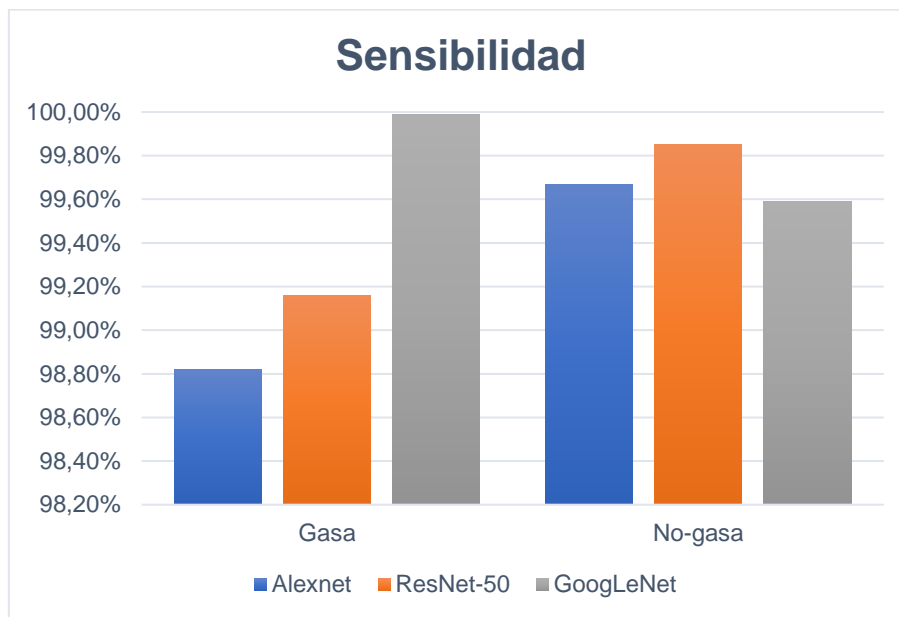


Figura 62 – Gráfico de la sensibilidad al clasificar 'gasa' y 'no-gasa' en diferentes redes

La red que tiene la mayor probabilidad de que cuando se detecte una gasa, en verdad la haya, es GoogleNet, seguido de ResNet y Alexnet.

Una vez que se tienen las redes favoritas a la hora de poder implementarse en un quirófano, si sólo se tiene en cuenta la precisión, ResNet encabeza la lista y si sólo se buscan resultados fiables, con el menor número de falsos positivos, GoogLeNet sería la más adecuada.

En un escenario real, donde lo que se busca es encontrar una gasa en el interior del cuerpo en el mayor número de casos posibles, sopesando que haya casos en los que se detecten gasas sin haberlas, debe priorizar la precisión a la sensibilidad, por lo tanto, la red preferida es ResNet-50. ResNet es la más equilibrada, tiene una precisión en la detección de gasas de un 99.61% y una sensibilidad de un 99.16%.

Otro factor que se debe tener en consideración, y que puede tener relevancia, es el tiempo de procesamiento que tiene la red clasificando las imágenes. Los fotogramas por segundo que se consiguen para poder realizar en tiempo real una detección de esas gasas. En la tabla 14 se muestran los tiempos de entrenamiento de 1000 imágenes en cada red y el tiempo de clasificación de 48 imágenes que es número de bloques en los que se divide cada fotograma de un vídeo de resoluciones de 720x576 píxeles (resolución de un laparoscopio), 42 bloques son de medidas 100x100 y 6 corresponden a los bordes de medidas 20 x 100.



	Tiempo de entrenamiento	Tiempo de clasificación de 48 imágenes
Alexnet	1 min 8 seg	0,30 segundos
ResNet-50	16 min 7 seg	1,17 segundos
GoogLeNet	4 min 13 seg	0,61 segundos

Tabla 14 – Resumen del tiempo de entrenamiento y de clasificación de las redes.

El tiempo que se tarda en entrenar la red es un parámetro que influye una única vez y que en algunos casos puede ser notable, ya que, si se quieren realizar pruebas para entrenar con distintas imágenes, es más interesante que tardara poco, pero no es un dato decisivo. Sin embargo, el tiempo de clasificación sí que es determinante de cara al rastreo de gasas por vídeo en tiempo real, tarea en la que esta clasificación tendría más sentido. Alexnet es la más rápida, con un tiempo de clasificación promedio por imagen inferior a 7 milisegundos, en segundo lugar, GoogLeNet, doblando ese tiempo y por último ResNet cuadruplicando el resultado de Alexnet y duplicando el de ResNet.

Si se actualizara el Hardware que se ha empleado durante todo el trabajo, estos tiempos se podrían mejorar, pero aun así Alexnet y GoogLeNet irían muy por delante respecto a tiempos de clasificación frente a ResNet.

	Tiempo de procesamiento	Precisión Gasa	Precisión 'no-gasa'	Sensibilidad Gasa	Sensibilidad 'no-gasa'
Alexnet	0,30 segundos	99.10%	99.57%	98,82%	99,67%
ResNet	1,17 segundos	99.61%	99.69%	99,16%	99,85%
GoogLeNet	0,61 segundos	98.88%	99.98%	99,99%	99,59%

Tabla 15 – Resumen general de las redes Alexnet, ResNet y GoogLeNet.

En la tabla 15 se muestran los resultados finales de la comparativa de las 3 redes con un código de colores básico, siendo el verde para el mejor resultado de la columna, amarillo para el segundo mejor y rojo para el peor resultado.

AlexNet ha resultado ser muy rápida, pero con mala sensibilidad para gasas y poca precisión. El duelo está entre ResNet y GoogLeNet donde GoogLeNet supera en tiempo de procesamiento a ResNet, en sensibilidad para detectar gasas y en la



precisión detectando el fondo, sin embargo, ResNet es la que tiene la mejor precisión en la detección de gasas y una sensibilidad superior al 99%.

La red más conveniente para ser utilizada en un quirófano es ResNet. Siempre y cuando, se ejecute en un Hardware que pueda disminuir en gran medida el tiempo de procesamiento. Con ResNet se va a conseguir una alta precisión en la detección de gasas.



5 CONCLUSIONES Y LÍNEAS FUTURAS

El uso de la inteligencia artificial en el ámbito de la medicina supone un gran avance, desde el diagnóstico y tratamiento de enfermedades. En el ámbito de la cirugía laparoscópica en el que se enmarca este proyecto cada vez es más común la incorporación de sistemas de visión artificial y asistentes robóticos.

En este trabajo de fin de grado, se ha conseguido desarrollar un sistema de seguimiento y detección de gases quirúrgicos retenidos en el interior del cuerpo en vídeo mediante redes neuronales convolucionales para evitar un error médico que puede tener consecuencias muy graves para el paciente. Para ello se ha hecho un análisis exhaustivo de 3 redes muy populares en el ámbito de la inteligencia artificial las cuales se han entrenado y modificado para conseguir detectar las gasas.

Las redes analizadas, Alexnet, ResNet y GoogLeNet, son muy distintas en su arquitectura y obtienen diferentes resultados especialmente en lo que a tiempo de ejecución se refiere.

La red más equilibrada en cuanto a precisión, sensibilidad y tiempo de procesamiento es ResNet. Esta red alcanza un porcentaje de precisión en la detección de gasas superior a Alexnet y GoogLeNet y con rangos de sensibilidad intermedios entre ambas redes. ResNet no presenta la mejor velocidad de procesamiento, pero esta característica no es preocupante debido a que la adopción de un hardware más potente permitiría una ejecución en tiempo real.

De cualquier forma, no hay que descartar la utilización de Alexnet y GoogLeNet puesto que la primera ha conseguido un tiempo de procesamiento muy inferior, hasta tres veces menos que ResNet lo que la haría muy interesante en entornos en los que se dispusiera de un hardware más limitado. GoogLeNet, aunque presenta una precisión inferior a ResNet, logra el mejor resultado en cuanto a sensibilidad a la hora de detectar gasas (99,99%).

Si la característica necesaria es la rapidez sin renunciar a una buena precisión, Alexnet es el candidato ideal. Si la velocidad de ejecución no es determinante, pero sí una muy buena precisión con buena sensibilidad, ResNet cumple estas características. Si lo que prima es la sensibilidad con un tiempo de procesamiento aceptable, GoogLeNet se muestra como la opción más adecuada.

El algoritmo desarrollado no pretende eliminar de los quirófanos los protocolos que se llevan actualmente a cabo para evitar la retención involuntaria de gasas si no está



destinado a ser una ayuda adicional que actúe de apoyo a la detección de conteos erróneos que suelen aparecer en largas operaciones o intervenciones de urgencia.

Como propuestas de continuación del trabajo se postulan varias vertientes:

- Mejorar el banco de imágenes disponibles: Convendría añadir más imágenes de reflejos en la categoría 'no-gasa', causantes de la mayor parte de los falsos positivos.
- Mejora del procesamiento de vídeo en tiempo real, implementando el algoritmo en un lenguaje más eficiente como C++ depurando el código para una más rápida ejecución.
- Los pocos falsos positivos que aparecen en las imágenes podrían ser eliminado eliminando del resultado proporcionado por la red bloques aislados ya que la aparición real de una gasa siempre vendrá marcada por varios bloques.
- Implementación del algoritmo YOLO, especialmente diseñado para la detección de objetos en imágenes en tiempo real. Las últimas versiones de este algoritmo, YOLOv4 y YOLOv5, que han aparecido en la literatura científica en los últimos meses se presentan como especialmente rápidas y eficientes.

Finalmente, se anima a extender esta línea de investigación a la detección y seguimiento de todo tipo de instrumental quirúrgico. Aunque la retención involuntaria de herramientas y utensilios ocurre con menor frecuencia que las gasas en operaciones de cirugía laparoscópica, el desarrollo de un sistema con capacidad para detectar todos estos elementos permitiría liberar al personal sanitario de esta tarea a la vez que erradicaría la aparición de estos errores tan peligrosos para los pacientes.



6 BIBLIOGRAFÍA

- [1] «Indicaciones de la Colectomía en pacientes de Cirugía Bariátrica,» [En línea]. Available: <http://iqaquiron.com/portal/indicaciones-de-la-colectomia-en-pacientes-de-cirurgia-bariatrica/>. [Último acceso: 13 08 2020].
- [2] «MedlinePlus, Biblioteca Nacional de Medicina de EE. UU,» [En línea]. Available: <https://medlineplus.gov/spanish/pruebas-de-laboratorio/laparoscopia/>. [Último acceso: 09 07 2020].
- [3] «Revista de salud y Bienestar,» [En línea]. Available: <https://www.webconsultas.com/pruebas-medicas/laparoscopia-11425>. [Último acceso: 09 07 2020].
- [4] C. M. Ramos, «Robótica y cirugía laparoscópica,» *Cirugía Española*, vol. 80, nº 4, pp. 189-194, 2006.
- [5] «Cirugía robótica: las nuevas manos de los médicos,» [En línea]. Available: <https://www.portafolio.co/tendencias/cirurgia-robotica-las-nuevas-manos-de-los-medicos-498465>. [Último acceso: 17 08 2020].
- [6] «Cirugía robótica,» [En línea]. Available: <https://www.mayoclinic.org/es-es/tests-procedures/robotic-surgery/about/pac-20394974>. [Último acceso: 13 8 2020].
- [7] Sackier JM, Wang Y., Robotically assisted laparoscopic surgery: from concept to development, 1994.
- [8] Alfredo Córdova Dupeyrat, Garth H. Ballantyne, «SISTEMA QUIRÚRGICOS ROBÓTICOS Y TELEBÓTICOS,» *Gastroenterol*, vol. 23, pp. 58-66, 2003.
- [9] Alfredo Córdova Dupeyrat, Garth H. Ballantyne, «SISTEMA QUIRÚRGICOS ROBÓTICOS Y TELEBÓTICOS,» *Gastroenterol. Perú*, vol. 1, nº 23, pp. 58-66, 2003.
- [10] Gabriela Savuc, Dorian Forna, Norina Consuela Forna, Gheorghe Asachi,, «THE CONTRIBUTION OF MEDICAL ROBOTS TO CLINICAL,» *Romanian Journal of Oral Rehabilitation*, vol. 4, nº 4, pp. 67-75, Diciembre 2012.
- [11] Lanfranco A, Castellanos A, Desai J, Meyers W. Robotic Surgery, A Current Perspective: Background and History of Surgical Robots, 2004, pp. 14-21.



- [12] «Cirugía Robótica da Vinci en IMED Valencia,» IMED, [En línea]. Available: <https://davinci.imedhospitales.com/cirugia-robotica>. [Último acceso: 25 07 2020].
- [13] E. Servicios, «Especialistas apuestan a cirugía robótica para optimizar la salud en México». *La estrella de Panamá*.
- [14] H. V. Mavrich, «Cirugía laparoscópica avanzada robótica Da Vinci: origen, aplicación clínica actual en Urología y su comparación con la cirugía abierta y laparoscópica,» de *Servicio de Urología. Fundació Puigvert. Barcelona.*, 2006.
- [15] Motta-Ramírez GA, González-Burgos O, Castillo-Lima JA, Villalobos-García E, «Material quirúrgico olvidado: Gossypiboma, textiloma, gazoma,» *Anales de Radiología Mex* 2007, nº 4, pp. 285-296.
- [16] S. Haykin, *Neural Networks and Learning Machines*, Hamilton, Ontario, Canada: Pearson, 2008.
- [17] «Convolutional Neural Networks (CNNs / ConvNets),» Stanford University, [En línea]. Available: <https://cs231n.github.io/convolutional-networks/>. [Último acceso: 20 07 2020].
- [18] F. Chollet, *Deep Learning with Python*, Manning, 2018, pp. 9-11.
- [19] ujjwalkarn, «A Quick Introduction to Neural Networks,» [En línea]. Available: <https://ujjwalkarn.me/2016/08/09/quick-intro-neural-networks/>. [Último acceso: 20 08 2020].
- [20] «¿Cómo funcionan las Convolutional Neural Networks? Visión por Ordenador,» [En línea]. Available: <https://www.aprendemachinelearning.com/como-funcionan-las-convolutional-neural-networks-vision-por-ordenador/>. [Último acceso: 09 09 2020].
- [21] P. Costa, «Redes neuronales convolucionales,» [En línea]. Available: <https://pochocosta.com/podcast/redes-neuronales-convolucionales-explicadas/>. [Último acceso: 09 09 2020].
- [22] «Red convolucional en PyTorch,» [En línea]. Available: <https://medium.com/@cleverpysolutions/red-convolucional-en-pytorch-c499ae8af6ca>. [Último acceso: 09 08 2020].
- [23] V. Powell, «Image Kernels,» [En línea]. Available: <http://setosa.io/ev/image-kernels/>. [Último acceso: 18 8 2020].



- [24] «Curso intensivo de aprendizaje automático,» Google, [En línea]. Available: <https://developers.google.com/machine-learning/practica/image-classification/convolutional-neural-networks>. [Último acceso: 20 08 2020].
- [25] «Intuition For: ResNet — Deep Residual Learning for Image Recognition,» [En línea]. Available: <https://medium.com/@realmichaelye/intuition-for-resnet-deep-residual-learning-for-image-recognition-39d24d173e78>. [Último acceso: 16 08 2020].
- [26] «Understanding Advanced Convolutional Neural Networks,» [En línea]. Available: <https://pythonmachinelearning.pro/understanding-advanced-convolutional-neural-networks/#ResNet>. [Último acceso: 25 07 2020].
- [27] He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian, «Deep Residual Learning for Image Recognition,» *IEEE*, 2016.
- [28] «Large Scale Visual Recognition Challenge (ILSVRC),» [En línea]. Available: <http://www.image-net.org/challenges/LSVRC/>. [Último acceso: 15 8 2020].
- [29] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, «Gradient-Based Learning Applied to Document Recognition,» *IEEE*, Noviembre 1998.
- [30] «GoogleNet - Artificial Intelligence,» [En línea]. Available: https://leonardoaraujosantos.gitbook.io/artificial-intelligence/machine_learning/deep_learning/googlenet. [Último acceso: 12 08 2020].
- [31] «Review: GoogLeNet (Inception v1)— Winner of ILSVRC 2014 (Image Classification),» [En línea]. Available: <https://medium.com/coinmonks/paper-review-of-googlenet-inception-v1-winner-of-ilsvrc-2014-image-classification-c2b3565a64e7>. [Último acceso: 15 8 2020].
- [32] Eusebio de la Fuente López, Álvaro Muñoz García, Lidia Santos del Blanco, Juan Carlos Fraile Marinero, Javier Pérez Turiel, «Automatic gauze tracking in laparoscopic surgery using image texture analysis,» *Computer Methods and Programs in Biomedicine*, p. 3, 2020.
- [33] M. D. L. T. Team, «Deep Learning Toolbox Model for AlexNet Network,» [En línea]. Available: <https://es.mathworks.com/matlabcentral/fileexchange/59133-deep-learning-toolbox-model-for-alexnet-network>. [Último acceso: 15 08 2020].
- [34] «Deep Learning Toolbox Model for ResNet-50 Network,» [En línea]. Available: <https://es.mathworks.com/matlabcentral/fileexchange/64626-deep-learning-toolbox-model-for-resnet-50-network>. [Último acceso: 15 8 2020].



- [35] «Deep Learning Toolbox Model for GoogLeNet Network,» [En línea]. Available: <https://es.mathworks.com/matlabcentral/fileexchange/64456-deep-learning-toolbox-model-for-googlenet-network>. [Último acceso: 15 8 2020].
- [36] J. Brownlee, «Understand the Impact of Learning Rate on Neural Network Performance,» [En línea]. Available: <https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/#:~:text=The%20amount%20that%20the%20weights,range%20between%200.0%20and%201.0..> [Último acceso: 2020 08 14].
- [37] «GoogleNet convolutional neural network,» [En línea]. Available: <https://es.mathworks.com/help/deeplearning/ref/googlenet.html>. [Último acceso: 12 08 2020].



7 ANEXO

En este anexo se muestran los códigos para entrenar y comprobar las tres redes desarrolladas en el trabajo, y los referentes a la clasificación de las gasas en vídeo.

Los códigos se dividen en 2 tipos.

- El primer tipo de código son 3 ficheros encargados cada uno del entrenamiento de las diferentes redes Alexnet, ResNet y GoogLeNet y corresponden a los apartados 7.1, 7.2 y 7.3.
- El otro tipo de código corresponde a la detección de las gasas en vídeo. El fichero principal de este tipo de código es el indicado en el apartado 7.4 (creaSecuenciaVideoTotal)

Las carpetas se estructuran de la siguiente forma:

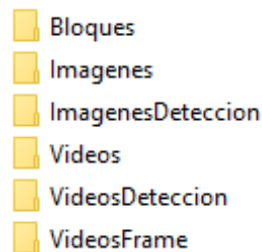


Figura 63 – Carpetas en las que se desarrolla el proyecto.

- **Bloques:** En esta carpeta se incluyen de manera automática la segmentación en bloques de cada fotograma para posteriormente ser clasificados.

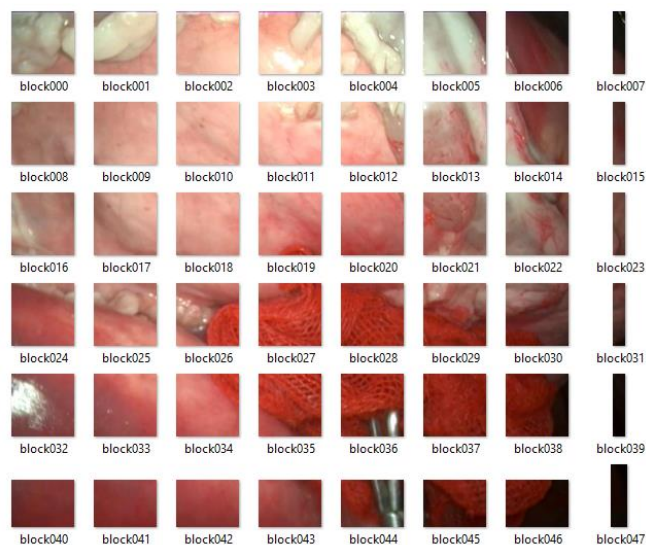


Figura 64 – Interior de la carpeta “Bloques”.



- **Imágenes:** En esta carpeta se localizan las imágenes de entrenamiento y test que se han utilizado a lo largo de todo el trabajo.
 - Dentro de esta carpeta se encuentran las subcarpetas “gasa” y “no-gasa” donde están las imágenes clasificadas de cada categoría.



Figura 65 – Interior de la carpeta “Imágenes”

- **ImagenesDetección:** Aquí se guardan los frames de los vídeos con el seguimiento de la gasa recuadrada en verde.

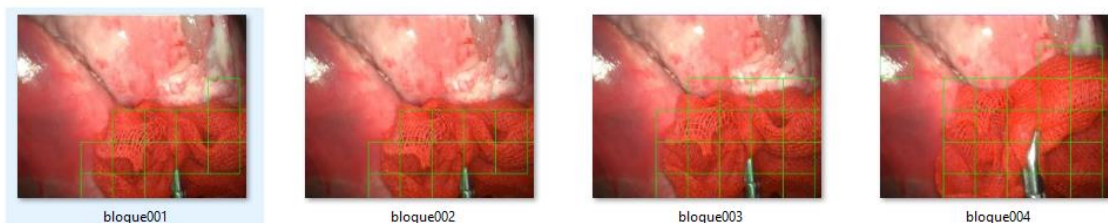


Figura 66 – Ejemplos de imágenes en la carpeta “ImagenesDeteccion”.

- **Videos:** Carpeta donde se ubican 13 vídeos utilizados para probar el seguimiento.



Figura 67 – Ejemplos de vídeos en la carpeta “Videos”.

- **VideosDetección:** Video resultante de juntar todos los fotogramas de imágenes detección. En él se aprecia el rastreo de la gasa. Los vídeos son generados con la estructura:

NOMBREREDn°Video_FotogramasPorSegundo.avi

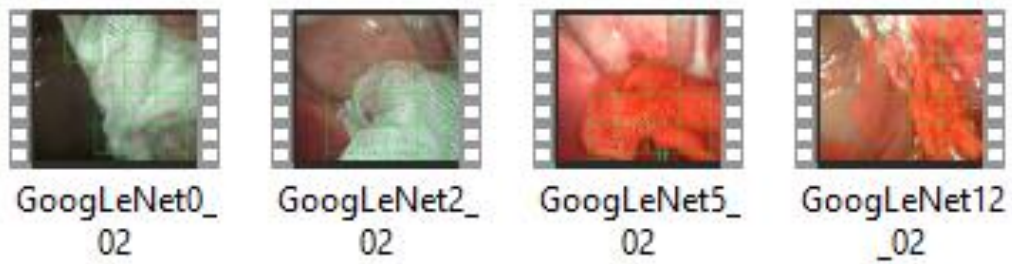


Figura 68 – Ejemplos de vídeos finales en la carpeta VideosDeteccion

- **VideosFrame:** Se incluyen todos los fotogramas en los que se divide el vídeo inicial, la estructura del nombre de las imágenes es la siguiente:

“vid”+n°Video_frame_n°defotograma.jpg

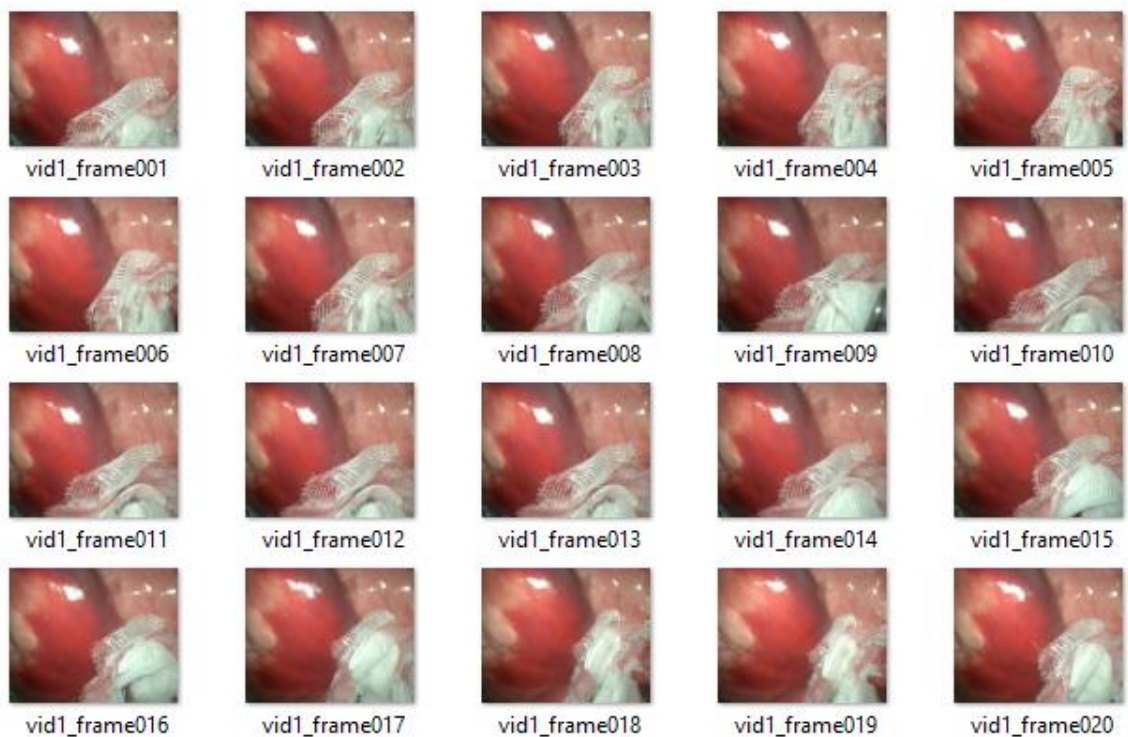


Figura 69 – Interior de la carpeta VideosFrame.



7.1 CÓDIGO FUENTE RED ALEXNET

```
% ----- Fichero Alexnet.mlx -----%
% ----- Entrenamiento y clasificación -----%

cd 'C:\Users\JAVIER\Desktop\TFG\V2.0' %Espacio de trabajo donde están las
imágenes en dos carpetas, gasa y no-gasa
tiles_ds =
imageDatastore('Imágenes','IncludeSubfolders',true,'LabelSource','foldernames'); %Carga de las imágenes

%Coge 1000 imag de fondo y 1000 de gasa para entren.
%Se podrían subir hasta 1400 porque hay 1781 de fondo
%trainImgs contiene 2000 imágenes. El resto (6673-2000=4670) para testImgs
[trainImgs,testImgs] = splitEachLabel(tiles_ds,1000,"randomized");

%Redimensionamiento de las imágenes 100x100 a las dimensiones de la red
227x227
trainImgs227x227=augmentedImageDatastore([227 227], trainImgs);
testImgs227x227=augmentedImageDatastore([227 227], testImgs);

%-----Modificación de las últimas capas de Alexnet para que clasifique
gasas -----%
net = alexnet;
layers = net.Layers;
numClasses=2;
layers(end-2) = fullyConnectedLayer(numClasses);
layers(end) = classificationLayer;

%-----Establecer las opciones del algoritmo de entrenamiento-----%
options = trainingOptions('sgdm', ...
'InitialLearnRate',0.001, ...
'MaxEpochs',1, ...
'Shuffle','every-epoch', ...
'ValidationData',testImgs227x227, ...
'ValidationFrequency',15, ...
'Verbose',false, ...
'Plots','training-progress');

[gauzenet,info] = trainNetwork(trainImgs227x227, layers, options);
save gauzenet.mat
%-----Utilizar la red entrenada para clasificar imágenes de prueba--%

[testPreds,probs] = classify(gauzenet,testImgs227x227);
gauzeActual=testImgs.Labels;
```



```
numCorrect=nz(testPreds==gauzeActual)
fracCorrect=numCorrect/numel(testPreds) %Precisión
confusionchart(gauzeActual,testPreds) %Matriz de confusión

%-----Probar la red con todas las imágenes-----%
cd 'C:\Users\JAVIER\Desktop\TFG\V2.0'
tiles_ds =
imageDatastore('Imágenes','IncludeSubfolders',true,'LabelSource','folder
names');
testImgs227x227=augmentedImageDatastore([227 227], tiles_ds);
[testPreds,probs] = classify(gauzenet,testImgs227x227);
gauzeActual=tiles_ds.Labels;
numCorrect=nz(testPreds==gauzeActual)
fracCorrect=numCorrect/numel(testPreds)
confusionchart(gauzeActual,testPreds)

%-- Clasificar las gasas dependiendo de lo manchadas que estén --%
%Las gasas limpias son de la 1 hasta la 717
B = zeros(1); % Matriz donde se van a almacenar el numero de la gasa
j=1;
for i=1:717
    if (categorical(testPreds(i)) ~= "gasa")
        B(j) = i;
        figure;
        imshow(cell2mat(testImgs227x227.Files(i)))
        j=j+1;
    end
end
B
precisionLimpias=100-(max(size(B))/717)*100

%Gasas Manchadas Sangre (el resto desde la iamgen 718 a 1785)
A = zeros(1);
j=1;
for i=718:1785
    if (categorical(testPreds(i)) ~= "gasa")
        A(j) = i;
        figure
        imshow(cell2mat(testImgs227x227.Files(i)))
        j=j+1;
    end
end
precisionManchadas=100-((max(size(A))/(1785-718))*100)
A
```



```
%-- Total de gasas que ha fallado
A = zeros(1);
i=0;
j=1;
for i=1:1785
    if (categorical(testPreds(i)) == "no-gasa")
        A(j) = i;
        figure;
        imshow(cell2mat(testImgs227x227.Files(i)))
        j=j+1;
    end
end
precisionGasasTotales=100-((max(size(A))/(1785))*100)
A

%-- Ver cuales ha fallado dentro de 'no-gasa'
B = zeros(1);
i=0;
j=1;
for i=1786:6677
    if (categorical(testPreds(i)) == "gasa")
        B(j) = i;
        figure;
        imshow(cell2mat(testImgs227x227.Files(i)))
        j=j+1;
    end
end
precisionNoGasasTotales=100-((max(size(B))/(6677-1786))*100)
B

%Tiempos clasificacion de 48 imágenes simulando los 48 bloques que tiene
un fotograma (24 imagenes aleatorias de cada uno)

cd 'C:\Users\JAVIER\Desktop\TFG\V2.0'
tiles_ds =
imageDatastore('Imagenes','IncludeSubfolders',true,'LabelSource','foldernames');
testImgs = splitEachLabel(tiles_ds,24,"randomized");
testImgs227x227=augmentedImageDatastore([227 227], testImgs);
tic
testPreds = classify(gauzenet,testImgs227x227)
toc
idx = randperm(numel(testImgs.Files),4676);
```




```
figure
a=0;
for i = 1:4676
    if 100*max(probs(idx(i),:))<60
        a=a+1;
        %subplot(50,1,a)
        figure;
        I = readimage(testImgs,idx(i));
        imshow(I)
        label = testPreds(idx(i));
        title(string(label) + ", " + num2str(100*max(probs(idx(i),:)),3) +
"%");
    end
end
```



7.2 CÓDIGO FUENTE RED RESNET

```
% ----- Fichero ResNet.mlx -----%
% ----- Entrenamiento y clasificación -----%

cd 'C:\Users\JAVIER\Desktop\TFG\V2.0'%Espacio de trabajo donde están las
imágenes en dos carpetas, gasa y no-gasa
tiles_ds =
imageDatastore('Imágenes','IncludeSubfolders',true,'LabelSource','foldernames'); %Carga de las imágenes

%Coge 1000 imágenes de fondo y 1000 de gasa para entren.
%Se podrían subir hasta 1400 porque hay 1781 de fondo
%trainImgs contiene 2000 imágenes. El resto (6673-2000=4670) para testImgs

[trainImgs,testImgs] = splitEachLabel(tiles_ds,1000,"randomized");

%Redimensionamiento de las imágenes 100x100 a las dimensiones de la red
224x224
trainImgs224x224=augmentedImageDatastore([224 224 3], trainImgs);
testImgs224x224=augmentedImageDatastore([224 224 3], testImgs);

%-----%
%-Modificación de las últimas capas de ResNet para que clasifique gasas
-----%
net=resnet50;
numClasses = numel(categories(trainImgs.Labels))
lgraph = layerGraph(net);

newFCLayer =
fullyConnectedLayer(numClasses,'Name','new_fc','WeightLearnRateFactor',10
,'BiasLearnRateFactor',10);
lgraph = replaceLayer(lgraph,'fc1000',newFCLayer);

newClassLayer = classificationLayer('Name','new_classoutput');
lgraph =
replaceLayer(lgraph,'ClassificationLayer_fc1000',newClassLayer);
figure('Units','normalized','Position',[0.3 0.3 0.4 0.4]);
plot(lgraph)
ylim([0,10])

%-----Establecer las opciones del algoritmo de entrenamiento-----
-----
inputSize = net.Layers(1).InputSize;
options = trainingOptions('sgdm', ...
```



```
'InitialLearnRate',0.001, ...
'MaxEpochs',1, ...
'Shuffle','every-epoch', ...
'ValidationData',testImgs224x224, ...
'ValidationFrequency',30, ...
'Verbose',false, ...
'MiniBatchSize',64, ...
'Plots','training-progress');
trainedNet = trainNetwork(trainImgs224x224,lgraph,options);

%-----Utilizar la red entrenada para clasificar imágenes de prueba--%

[testPreds,probs] = classify(trainedNet,testImgs224x224);
gauzeActual=testImgs.Labels;
numCorrect=nnz(testPreds==gauzeActual)
fracCorrect=numCorrect/numel(testPreds)
confusionchart(gauzeActual,testPreds)

%-----Probar la red con todas las imágenes-----%
cd 'C:\Users\JAVIER\Desktop\TFG\V2.0'
tiles_ds =
imageDatastore('Imágenes','IncludeSubfolders',true,'LabelSource','folderNames');
testImgs224x224=augmentedImageDatastore([224 224 3], tiles_ds);
[testPreds,probs] = classify(trainedNet,testImgs224x224);
gauzeActual=tiles_ds.Labels;
numCorrect=nnz(testPreds==gauzeActual)
fracCorrect=numCorrect/numel(testPreds)
confusionchart(gauzeActual,testPreds)

%-- Clasificar las gasas dependiendo de lo manchadas que estén --%
%Las gasas limpias son de la 1 hasta la 717
B = zeros(1);
j=1;
for i=1:717
    if (categorical(testPreds(i)) ~= "gasa")
        B(j) = i;
        figure;
        imshow(cell2mat(testImgs224x224.Files(i)))
        j=j+1;
    end
end
B
precisionLimpias=100-(max(size(B))/717)*100
```



```
%Gasas Manchadas Sangre (el resto desde la imagen 718 a 1785)
A = zeros(1);
j=1;
for i=718:1785
    if (categorical(testPreds(i)) ~= "gasa")
        A(j) = i;
        figure
        imshow(cell2mat(testImgs224x224.Files(i)))
        j=j+1;
    end
end
precisionManchadas=100-((max(size(A))/(1785-718))*100)
A

%-- Ver cuales ha fallado dentro de gasa
A = zeros(1);
i=0;
j=1;
for i=1:1785
    if (categorical(testPreds(i)) == "no-gasa")
        A(j) = i;
        figure;
        imshow(cell2mat(testImgs224x224.Files(i)))
        j=j+1;
    end
end
precisionGasasTotales=100-((max(size(A))/(1785))*100)
A

%-- Ver cuales ha fallado dentro de 'no-gasa'
B = zeros(1);
i=0;
j=1;
for i=1786:6677
    if (categorical(testPreds(i)) == "gasa")
        B(j) = i;
        figure;
        imshow(cell2mat(testImgs224x224.Files(i)))
        j=j+1;
    end
end
precisionNoGasasTotales=100-((max(size(A))/(6677-1786))*100)
B

%Tiempos clasificacion de 48 imágenes simulando los 48 bloques que tiene
un fotograma (24 imagenes aleatorias de cada uno)
```



```
cd 'C:\Users\JAVIER\Desktop\TFG\V2.0'
tiles_ds =
imageDatastore('Imagenes','IncludeSubfolders',true,'LabelSource','foldernames');

testImgs = splitEachLabel(tiles_ds,24,"randomized");
testImgs224x224=augmentedImageDatastore([224 224 3], testImgs);

tic
testPreds = classify(trainedNet,testImgs224x224);
toc

%Elapsed time is 2.313631 seconds.
idx = randperm(numel(testImgs.Files),4676);

figure
a=0;
for i = 1:4676
    if 100*max(probs(idx(i),:))<60
        a=a+1;
        figure;
        I = readimage(testImgs,idx(i));
        imshow(I)
        label = testPreds(idx(i));
        title(string(label) + ", " + num2str(100*max(probs(idx(i),:)),3) +
"%");
    end
end
end
```



7.3 CÓDIGO FUENTE RED GOOGLNET

```
% ----- Fichero GoogLeNet.mlx -----%
% ----- Entrenamiento y clasificación -----%

cd 'C:\Users\JAVIER\Desktop\TFG\V2.0'%Espacio de trabajo donde están las
imágenes en dos carpetas, gasa y no-gasa

tiles_ds =
imageDatastore('Imágenes','IncludeSubfolders',true,'LabelSource','foldern
ames');
%Coge 1000 imágenes de fondo y 1000 de gasa para entren.
%Se podrían subir hasta 1400 porque hay 1781 de fondo
%trainImgs contiene 2000 imágenes. El resto (6673-2000=4670) para testImgs
testImgs

[trainImgs,testImgs] = splitEachLabel(tiles_ds,1000,"randomized");

%Redimensionamiento de las imágenes 100x100 a las dimensiones de la red
224x224
augimdsTrain=augmentedImageDatastore([224 224 3], trainImgs);
augimdsTest=augmentedImageDatastore([224 224 3], testImgs);

%-Modificación de las últimas capas de GoogLeNet para que clasifique----
gasas -----%
net = googlenet;
net.Layers(1)
inputSize = net.Layers(1).InputSize;
if isa(net,'SeriesNetwork')
    lgraph = layerGraph(net.Layers);
else
    lgraph = layerGraph(net);
end

[learnableLayer,classLayer] = findLayersToReplace(lgraph);
[learnableLayer,classLayer]

numClasses = numel(categories(trainImgs.Labels));

if isa(learnableLayer,'nnet.cnn.layer.FullyConnectedLayer')
    newLearnableLayer = fullyConnectedLayer(numClasses, ...
        'Name','new_fc', ...
        'WeightLearnRateFactor',10, ...
        'BiasLearnRateFactor',10);
```



```
elseif isa(learnableLayer, 'nnet.cnn.layer.Convolution2DLayer')
    newLearnableLayer = convolution2dLayer(1,numClasses, ...
        'Name','new_conv', ...
        'WeightLearnRateFactor',10, ...
        'BiasLearnRateFactor',10);
end

lgraph = replaceLayer(lgraph,learnableLayer.Name,newLearnableLayer);
newClassLayer = classificationLayer('Name','new_classoutput');
lgraph = replaceLayer(lgraph,classLayer.Name,newClassLayer);
figure('Units','normalized','Position',[0.3 0.3 0.4 0.4]);
plot(lgraph)
ylim([0,10])
layers = lgraph.Layers;
connections = lgraph.Connections;
layers(1:10) = freezeWeights(layers(1:10));
lgraph = createLgraphUsingConnections(layers,connections);

%-----Establecer las opciones del algoritmo de entrenamiento-----%

augimdsValidation = augmentedImageDatastore(inputSize(1:2),testImgs);
miniBatchSize = 10;
valFrequency = floor(numel(augimdsTrain.Files)/miniBatchSize);
options = trainingOptions('sgdm', ...
    'MiniBatchSize',miniBatchSize, ...
    'MaxEpochs',1, ...
    'InitialLearnRate',3e-4, ...
    'Shuffle','every-epoch', ...
    'ValidationData',augimdsValidation, ...
    'ValidationFrequency',valFrequency, ...
    'Verbose',false, ...
    'Plots','training-progress');

net = trainNetwork(augimdsTrain,lgraph,options);

%-----Utilizar la red entrenada para clasificar imágenes de prueba--%
[YPred,probs] = classify(net,augimdsValidation);
accuracy = mean(YPred == testImgs.Labels)
gauzeActual=testImgs.Labels;
confusionchart(gauzeActual,YPred)
idx = randperm(numel(testImgs.Files),4676);
figure
a=0;
for i = 1:4676
    if 100*max(probs(idx(i),:))<60
```



```
a=a+1;
%subplot(50,1,a)
figure;
I = readimage(testImgs,idx(i));
imshow(I)
label = YPred(idx(i));
title(string(label) + ", " + num2str(100*max(probs(idx(i),:)),3) +
"%");
end
end

%-----Probar la red con todas las imágenes-----%
cd 'C:\Users\JAVIER\Desktop\TFG\V2.0'
tiles_ds =
imageDatastore('Imágenes','IncludeSubfolders',true,'LabelSource','foldernames');
testImgs224x224=augmentedImageDatastore([224 224], tiles_ds);
[testPreds,probs] = classify(net,testImgs224x224);
gauzeActual=tiles_ds.Labels;
numCorrect=nz(testPreds==gauzeActual)
fracCorrect=numCorrect/numel(testPreds)
confusionchart(gauzeActual,testPreds)

%-- Clasificar las gasas dependiendo de lo manchadas que estén --%
%Las gasas limpias son de la 1 hasta la 717
B = zeros(1); % Matriz donde se van a almacenar el numero de la gasa
j=1;
for i=1:717
    if (categorical(testPreds(i)) ~= "gasa")
        B(j) = i;
        figure;
        imshow(cell2mat(testImgs224x224.Files(i)))
        j=j+1;
    end
end
B
%Gasas Manchadas Sangre (el resto desde la imagen 718 a 1785)

A = zeros(1);
j=1;
for i=718:1785
    if (categorical(testPreds(i)) ~= "gasa")
        A(j) = i;
        figure
        imshow(cell2mat(testImgs224x224.Files(i)))
```




```
        j=j+1;
    end
end
precisionManchadas=100-((max(size(A))/(1785-718))*100)
A

%-- Ver cuales ha fallado dentro de gasa
A = zeros(1);
i=0;
j=1;
for i=1:1785
    if (categorical(testPreds(i)) == "no-gasa")
        A(j) = i;
        figure;
        imshow(cell2mat(testImgs224x224.Files(i)))
        j=j+1;
    end
end
precisionGasasTotales=100-((max(size(A))/(1785))*100)
A

%-- Ver cuales ha fallado dentro de 'no-gasa'
B = zeros(1);
i=0;
j=1;
for i=1786:6677
    if (categorical(testPreds(i)) == "gasa")
        B(j) = i;
        figure;
        imshow(cell2mat(testImgs224x224.Files(i)))
        j=j+1;
    end
end
precisionNoGasasTotales=100-((max(size(B))/(6677-1786))*100)
B

%Tiempos clasificacion de 48 imágenes simulando los 48 bloques que tiene
un fotograma (24 imagenes aleatorias de cada uno)
cd 'C:\Users\JAVIER\Desktop\TFG\V2.0'
tiles_ds =
imageDatastore('Imagenes','IncludeSubfolders',true,'LabelSource','foldernames');
testImgs = splitEachLabel(tiles_ds,24,"randomized");
testImgs227x227=augmentedImageDatastore([224 224], testImgs);
testPreds = classify(net,testImgs227x227);
```



7.4 CÓDIGO FUENTE DETECCIÓN EN VÍDEO

En este apartado se comentan varias funciones y scripts desarrollados en Matlab para la detección de gases en vídeo. El programa principal es el de [creaSecuenciaVideoTotal](#).

7.4.1 creaSecuenciaVideoTotal

```
metodo = "GoogLeNet"; %Puede ser Alexnet, ResNet o GoogLeNet
if metodo == "Alexnet"
    load gauzenet.mat
end
if metodo == "ResNet"
    load TrainedNet.mat
end
if metodo == "GoogLeNet"
    load net.mat
end

Tam = 100; %Tamaño bloques
N = 5; %Numero video // PROBAR VIDEO 0, VIDEO 2, VIDEO 5 Y VIDEO 12
fotogramas = 2; %fotogramas x segundo
cuenta = sacaFrames(N,fotogramas)
tic
for contador=1:cuenta

nombreImagenVideo=strcat('C:\Users\JAVIER\Desktop\TFG\V2.0\VideosFrame\vi
d',num2str(N),'_frame',num2str(contador,'%03g'),' .jpg');
I=imread(nombreImagenVideo);
guardaBloques(I,Tam)
tiles_ds = imageDatastore('Bloques');
    if metodo == "Alexnet"
        testImgsAugmented=augmentedImageDatastore([227 227], tiles_ds);
        [testPreds,probs] = classify(gauzenet,testImgsAugmented);
    end
    if metodo == "ResNet"
        testImgsAugmented=augmentedImageDatastore([224 224], tiles_ds);
        [testPreds,probs] = classify(trainedNet,testImgsAugmented);
    end
    if metodo == "GoogLeNet"
        testImgsAugmented=augmentedImageDatastore([224 224], tiles_ds);
        [testPreds,probs] = classify(net,testImgsAugmented);
    end
    cuadrosGasas = zeros(1);
    j=1;
```



```
for i=1:size(testPreds)
    if categorical(testPreds(i)) == "gasa"
        cuadrosGasas(j) = i;
        j=j+1;
        i = num2str(i-1);
        a =
strcat('C:\Users\JAVIER\Desktop\TFG\V2.0\Bloques\block',i, '.png');

        end
    end
    cuadrosGasas;
    pintaCuadros(I,Tam,cuadrosGasas,contador)
end
toc
creaVideo(N,cuenta,metodo,fotogramas)
```

7.4.2 sacaFrames.m

Esta función lo que hace es segmentar el vídeo en los fotogramas que se le indiquen.

Devuelve el valor de los fotogramas en los que se ha segmentado el vídeo en la variable cuenta y tiene como parámetros de entrada el número del vídeo (0-12) y la tasa de fotogramas por segundo que se desea muestrear.

```
function cuenta = sacaFrames(N,fotogramas) %N es el numero del video
nombreVideo=strcat('C:\Users\JAVIER\Desktop\TFG\V2.0\Videos\video',num2str(N)
, '.mp4')
v = VideoReader(nombreVideo);
fprintf("Este video generarÃ¡ %d imagenes\n", v.NumFrames);
cuenta = 0;
rate=round((1/fotogramas)*25)

for i=1:rate:v.NumFrames
    cuenta = cuenta+1;
    frame=read(v,i);
nombreFichero=strcat('C:\Users\JAVIER\Desktop\TFG\V2.0\VideosFrame\vid',num2s
tr(N), '_frame',num2str(cuenta,'%03g'), '.jpg');
    imwrite(frame,nombreFichero);
end

end
```

7.4.3 GuardaBloques.m

Esta función segmenta los fotogramas en bloques para su clasificación.



No devuelve ningún valor y como entradas tiene el fotograma y el tamaño de bloque que se desea.

```
function guardaBloques(I,tamBloque)
[filas,columnas,planos]=size(I);
numBloque=0000;
for j=1:tamBloque:filas
    for i=1:tamBloque:columnas
        block= imcrop(I, [i j tamBloque-1 tamBloque-1]);
        num_str=num2str(numBloque,'%03g');

nombre_block=strcat('C:\Users\JAVIER\Desktop\TFG\V2.0\Bloques\block',num_str,
'.png');
        %si se quieren guardar sólo los bloques completos descomentar la
        linea siguiente
        % if( (i-1 + tamBloque-1) <= columnas && (j-1 + tamBloque-1) <= filas)
            imwrite(block,nombre_block);
            numBloque=numBloque+1;
        %end
    end
end
%dibujo reticula para más informacion
%pintaReticula(I,tamBloque);
end
```

7.4.4 pintaCuadrados.m

Aquí se encarga de unir todos los bloques, y en aquellos donde se haya clasificado una gasa recuadrarlo de verde. Como entradas tiene el fotograma, tamaño de bloque, los bloques donde tiene que pintar los cuadrados y un contador para pasar al fotograma siguiente

```
function pintaCuadrados(I,tamBloque,cuadrosGasas,contador)
[filas,columnas,planos]=size(I);
numBloque=0;
for j=1:tamBloque:filas
    for i=1:tamBloque:columnas
        % si se quieren guardar sólo los bloques completos descomentar la
        linea siguiente
        % if( (i-1 + tamBloque-1) <= columnas && (j-1 + tamBloque-1) <= filas)
            num_str=num2str(numBloque);
            % I = insertText(I,[i,j],num_str);
            numBloque=numBloque+1;
        % end
    end
end

v=1;
c=1;
f=1;
% Dibuja reticula
```



```
for j=1:tamBloque:filas
    for i=1:tamBloque:columnas
        if v<=size(cuadrosGasas,2)
            if c == cuadrosGasas(v)
                % si se quieren guardar sólo los bloques completos descomentar
                la linea siguiente
                %if( (i-1 + tamBloque) <= columnas && (j-1 + tamBloque) <=
                filas)
                    I=insertShape(I,'rectangle',[i,j,tamBloque,tamBloque],
                    'Color','green','LineWidth',2);
                    v=v+1;
                end
                c=c+1;
            end
        end
    end
end
nombreFichero=strcat('C:\Users\JAVIER\Desktop\TFG\V2.0\ImágenesDeteccion\bloque',num2str(contador,'%03g'),' .jpg');
imwrite(I,nombreFichero);
imshow(I)
end
```

7.4.5 creaVideo.m

Finalmente se juntan todos los fotogramas que ha clasificado el algoritmo y ha guardado con los recuadros en las gasas y se crea un vídeo con la siguiente estructura: **NOMBREREDn°Video_FotogramasPorSegundo.avi**

Como entradas tiene el nº de vídeo, el total de frames, el método utilizado y la tasa de fotogramas por segundo.

```
function creaVideo(numero, frames, metodo,fotogramas)
nombreVideo=strcat('C:\Users\JAVIER\Desktop\TFG\V2.0\VideosDeteccion\',metodo
,num2str(numero),'_',num2str(fotogramas,'%02g'),' .avi')
video = VideoWriter(nombreVideo); %create the video object
video.FrameRate = fotogramas;
open(video);
for ii=1:frames
    imagenVideo=strcat('C:\Users\JAVIER\Desktop\TFG\V2.0\ImágenesDeteccion\bloque',num2str(ii,'%03g'),' .jpg');
    I = imread(imagenVideo);
    writeVideo(video,I);
end
close(video);
end
```