Resumen

La presente memoria detalla el proceso de creación de un módulo de inteligencia artificial, diseñado para detectar personas y animales en vídeo de vigilancia capturado con cámaras montadas en drones. En particular, lo que se busca con el desarrollo de este trabajo es detectar intrusos en streaming de vídeo como parte de un sistema más grande de vigilancia, que será implementado por la empresa Manta Beach. Esta empresa, que hace parte del programa de vinculación de la universidad, propuso el trabajo como parte de su plan de investigación y desarrollo.

Para la creación de este módulo,se hace un uso extensivo de los conocimientos de la rama de visión por computadora a través de redes neuronales convolucionales. De igual manera, se ha requerido una base sólida en Python, teniendo en cuenta que se partió de un algoritmo establecido con anterioridad.

I

Resumen

La presente memoria detalla el proceso de creación de un módulo de inteligencia artificial, diseñado para detectar personas y animales en vídeo de vigilancia capturado con cámaras montadas en drones. En particular, lo que se busca con el desarrollo de este trabajo es detectar intrusos en *streaming* de vídeo como parte de un sistema más grande de vigilancia, que será implementado por la empresa Manta Beach. Esta empresa, que hace parte del programa de vinculación de la universidad, propuso el trabajo como parte de su plan de investigación y desarrollo.

Para la creación de este módulo, se hace un uso extensivo de los conocimientos de la rama de visión por computadora a través de redes neuronales convolucionales. De igual manera, se ha requerido una base sólida en Python, teniendo en cuenta que se partió de un algoritmo establecido con anterioridad.

- Razones económicas: teniendo en cuenta que, al reducir al personal encargado de hacer seguimiento a las cámaras, o de hacer rondas de seguridad en horarios determinados, es posible reducir costos. De igual manera, el uso de drones permite una reducción sustancial en el número de cámaras que deberán ser instaladas, cosa que resulta particularmente útil en grandes áreas y permitirá una reducción muy importante en los costos de mantenimiento asociados.
- Calidad de la vigilancia: dado que el trabajo de gran parte del personal de vigilancia tiene la particularidad de que puede volverse muy monótono. Esto teniendo en cuenta que es un trabajo en el que se espera que no suceda nada. Sin embargo, es de vital importancia cuando esto deja de cumplirse, dado que hay algún evento que pueda afectar la seguridad.

  Al ser tan monótono, es muy probable que el personal encargado se vea expuesto a distracciones, que pueden tener consecuencias severas en el momento en el que se presente algún evento que comprometa la seguridad. La situación se puede agravar en caso de tener una gran cantidad de cámaras, dado que el estar constantemente obligados a supervisarlas puede generar niveles no razonables de carga en el personal.
- Velocidad de reacción: una vez identificada una potencial amenaza, a través del sistema encargado de controlar los drones se podrá enviar ordenes a los equipos a fin de tomar mejores decisiones para manejar la situación. Es por esto que es de vital importancia que el módulo de inteligencia artificial encargado de la detección de los intrusos también sea capaz de hacerlo en tiempos razonables, que en este caso, han sido definidos por el cliente y se encuentran por debajo de los 4 segundos. Este valor se ha calculado a partir de la velocidad esperada de vuelo de los drones de hasta 30 km/h, lo que otorga un radio máximo de 33 metros entre detecciones. Cabe anotar, sin embargo, que el tiempo ideal es menor o igual a un segundo, lo que genera un radio de 8 metros aproximadamente.
- Aporte de pruebas: los drones (a través de la necesidad de velocidad de reacción) son un método ideal para aportar pruebas confiables de los hechos. Es así como un modelo de inteligencia artificial permite generar una mayor claridad en la interpretación de las pruebas aportadas, lo que puede generar una mejor comprensión y análisis de los hechos por parte de todos los implicados.
- Identificación de intrusos en horas de la noche: teniendo en cuenta que los modelos de inteligencia artficial pueden ser capaces de identificar las clases requeridas con diferentes condiciones de iluminación, puede ayudar a la identificación de intrusos cuando las condiciones de iluminación no son ideales. Por otro lado, es muy probable que esto sea mejorado sustancialmente con el uso de cámaras infrarrojas.

Este tipo de propuestas, sin embargo, no se han hecho posibles hasta hace relativamente poco, cuando los modelos de inteligencia artificial alcanzaron una madurez que permite que no requieran grandes capacidades computacionales para poder ser ejecutados, de forma que implementarlos no requiera una inversión importante en equipos de cómputo.

2 Capítulo 1. Introducción general

- Razones económicas: teniendo en cuenta que, al reducir al personal encargado de hacer seguimiento a las cámaras, o de hacer rondas de seguridad en horarios determinados, es posible reducir costos. De igual manera, el uso de drones permite una reducción sustancial en el número de cámaras que deberán ser instaladas, cosa que resulta particularmente útil en grandes áreas y permitirá una reducción muy importante en los costos de mantenimiento asociados.
- Calidad de la vigilancia: dado que el trabajo de gran parte del personal de vigilancia tiene la particularidad de que puede volverse muy monótono. Esto teniendo en cuenta que es un trabajo en el que se espera que no suceda nada. Sin embargo, es de vital importancia cuando esto deja de cumplirse, dado que hay algún evento que pueda afectar la seguridad.

  Al ser tan monótono, es muy probable que el personal encargado se vea expuesto a distracciones, que pueden tener consecuencias severas en el momento en el que se presente algún evento que comprometa la seguridad. La situación se puede agravar en caso de tener una gran cantidad de cámaras, dado que el estar constantemente obligados a supervisarlas puede generar niveles no razonables de carga en el personal.
- Velocidad de reacción: una vez identificada una potencial amenaza, a través del sistema encargado de controlar los drones se podrá enviar órdenes a los equipos a fin de tomar mejores decisiones para manejar la situación. Es por esto que es de vital importancia que el módulo de inteligencia artificial encargado de la detección de los intrusos también sea capaz de hacerlo en tiempos razonables, que en este caso, han sido definidos por el cliente y se encuentran por debajo de los 4 segundos. Este valor se ha calculado a partir de la velocidad esperada de vuelo de los drones de hasta 30 km/h, lo que otorga un radio máximo de 33 metros entre detecciones. Cabe anotar, sin embargo, que el tiempo ideal es menor o igual a un segundo, lo que genera un radio de 8 metros aproximadamente.
- Aporte de pruebas: los drones (a través de la necesidad de velocidad de reacción) son un método ideal para aportar pruebas confiables de los hechos. Es así como un modelo de inteligencia artificial permite generar una mayor claridad en la interpretación de las pruebas aportadas, lo que puede generar una mejor comprensión y análisis de los hechos por parte de todos los implicados.
- Identificación de intrusos en horas de la noche: teniendo en cuenta que los modelos de inteligencia artficial pueden ser capaces de identificar las clases requeridas con diferentes condiciones de iluminación, puede ayudar a la identificación de intrusos cuando las condiciones de iluminación no son ideales. Por otro lado, es muy probable que esto sea mejorado sustancialmente con el uso de cámaras infrarrojas.

Este tipo de propuestas, sin embargo, no se han hecho posibles hasta hace relativamente poco, cuando los modelos de inteligencia artificial alcanzaron una madurez que permite que no requieran grandes capacidades computacionales para poder ser ejecutados, de forma que implementarlos no requiera una inversión importante en equipos de cómputo.

- Estado: descartado

5.2 **Opción B:** el módulo debe ser capaz de recibir un *streaming* de vídeo y ejecutar la detección de intrusos en hardware dedicado a bordo.

- Tipo: requisito modificado
- Estado: descartado

5.3 Se requiere que el sistema pueda conectarse con otro módulo capaz de tomar decisiones con base en la información de detección de intrusos.

- Tipo: obligatorio
- Estado: finalizado

A pesar de que algunos de los requerimientos funcionales fueron descartados dados algunos cambios que se presentaron en las plataformas del cliente. Sin embargo, también hubo requerimientos que fueron agregados:

6. Requerimientos nuevos:

6.1 El módulo deberá retransmitir los *frames* procesados a un servidor indicado por el cliente.

- Tipo: obligatorio
- Estado: finalizado

Es importante tener en cuenta que el trabajo no contempló:

- Reconocimiento del intruso: es decir, en caso de que se detecte a una persona, no se deberá dar con su identidad.
- Reacción del dron: es decir, una vez se haya detectado satisfactoriamente a un intruso, el sistema no deberá dar instrucciones específicas al dron, por ejemplo, de seguirlo.
- No existen requisitos específicos de cómo realizar la retransmisión de los frames ya procesados.

6 Capítulo 1. Introducción general

- Estado: descartado

5.2 **Opción B:** el módulo debe ser capaz de recibir un *streaming* de vídeo y ejecutar la detección de intrusos en hardware dedicado a bordo.

- Tipo: requisito modificado
- Estado: descartado

5.3 Se requiere que el sistema pueda conectarse con otro módulo capaz de tomar decisiones con base en la información de detección de intrusos.

- Tipo: obligatorio
- Estado: finalizado

A pesar de que algunos de los requerimientos funcionales fueron descartados dados algunos cambios que se presentaron en las plataformas del cliente, también hubo requerimientos que fueron agregados:

6. Requerimientos nuevos:

6.1 El módulo deberá retransmitir los *frames* procesados a un servidor indicado por el cliente.

- Tipo: obligatorio
- Estado: finalizado

Es importante tener en cuenta que el trabajo no contempló:

- Reconocimiento del intruso: es decir, en caso de que se detecte a una persona, no se deberá dar con su identidad.
- Reacción del dron: es decir, una vez se haya detectado satisfactoriamente a un intruso, el sistema no deberá dar instrucciones específicas al dron, por ejemplo, de seguirlo.
- No existen requisitos específicos de cómo realizar la retransmisión de los *frames* ya procesados.

Capítulo 2

Introducción específica

En este capítulo, se presentan los conceptos necesarios para poner en funcionamiento el trabajo de detección. Se inicia con la descripción de los objetos a detectar, seguido de los modelos de visión por computadora disponibles. De igual manera, se aborda los protocolos de transmisión y recepción del vídeo y se finaliza con los requerimientos computacionales necesarios para llevar a cabo la detección.

2.1. Definición de Intruso

Para cumplir con el objetivo específico del trabajo, es importante tener en cuenta la definición de intruso dada por el cliente, que incluye la detección de personas y animales de gran porte, lo que implica que la respuesta del módulo deberá activarse con cualquier animal más grande que un perro o un gato. Es importante tener en mente que el *dataset* COCO permite el reconocimiento de una serie de animales de gran porte que no hacen parte de la fauna argentina, por lo que se decidió en conjunto con el cliente que algunas etiquetas se filtrarían a fin de evitar que el modelo intente reconocer estas clases. Algunos ejemplos de etiquetas filtradas incluyen:

- Elefantes
- Jirafas
- Cebras

Por otro lado, y a fin de mejorar el reconocimiento de intrusos, se decidió incorporar algunas clases que corresponden a objetos que suelen cargar las personas y vehículos. De esta manera se hace posible una suerte de reconocimiento indirecto, dándole al modelo una "segunda oportunidad" en casos en los que por algún motivo generó un falso negativo en el reconocimiento de una persona. Es importante tener en cuenta que estos objetos propuestos ya hacían parte del *dataset* COCO, por lo que no se requirió reentrenar el modelo para lograr su detección. Dentro de estos objetos relacionados encontramos:

- Vehículos
- Objetos de porte personal, como maletines o sombrillas

Es importante tener en cuenta que el módulo marcará toda detección como una intrusión. Esto quiere decir que las detecciones en lotes vecinos no se filtrarán dado que se considera que la totalidad del *frame* es área vigilada. Dicho esto, las clases que generarán un reporte positivo para intruso al ser detectadas son:

7

Capítulo 2

Introducción específica

En este capítulo, se presentan los conceptos necesarios para poner en funcionamiento el trabajo de detección. Se inicia con la descripción de los objetos a detectar, seguido de los modelos de visión por computadora disponibles. De igual manera, se abordan los protocolos de transmisión y recepción del vídeo y se finaliza con los requerimientos computacionales necesarios para llevar a cabo la detección.

2.1. Definición de Intruso

Para cumplir con el objetivo específico del trabajo, es importante tener en cuenta la definición de intruso dada por el cliente, que incluye la detección de personas y animales de gran porte, lo que implica que la respuesta del módulo deberá activarse con cualquier animal de mayor tamaño al de un perro o un gato. Es importante tener en mente que el *dataset* COCO permite el reconocimiento de una serie de animales de gran porte que no hacen parte de la fauna argentina, por lo que se decidió en conjunto con el cliente que algunas etiquetas se filtrarían a fin de evitar que el modelo intente reconocer estas clases. Algunos ejemplos de etiquetas filtradas incluyen:

- Elefantes
- Jirafas
- Cebras

Por otro lado, y a fin de mejorar el reconocimiento de intrusos, se decidió incorporar algunas clases que corresponden a objetos que suelen cargar las personas y vehículos. De esta manera se hace posible una suerte de reconocimiento indirecto, dándole al modelo una "segunda oportunidad" en casos en los que por algún motivo generó un falso negativo en el reconocimiento de una persona. Es importante tener en cuenta que estos objetos propuestos ya hacían parte del *dataset* COCO, por lo que no se requirió reentrenar el modelo para lograr su detección. Dentro de estos objetos relacionados encontramos:

- Vehículos
- Objetos de porte personal, como maletines o sombrillas

Es importante tener en cuenta que el módulo marcará toda detección como una intrusión. Esto quiere decir que las detecciones en lotes vecinos no se filtrarán dado que se considera que la totalidad del *frame* es área vigilada. Dicho esto, las clases que generarán un reporte positivo para intruso al ser detectadas son:

TABLA 2.1. Clases detectadas como intruso por el módulo

Clases detectadas como intruso	
Persona	Vaca
Camión	Bus
Automóvil	Sombrilla
Motocicleta	Mochila
Bicicleta	Maletín
Perro	Maleta
Caballo	Gato

Hay que recordar que la sola detección de una de estas clases generará un reporte positivo de intruso. El modelo no está en capacidad de identificar personas específicas en el vídeo, por lo que le resultará imposible diferenciar entre personal autorizado en un área (el mismo personal de seguridad, o la persona al mando del dron), y a intrusos en la zona.

2.2. Modelo utilizado

El modelo utilizado para el desarrollo de la totalidad del trabajo es YOLOv3. La principal ventaja que tiene este modelo es que es capaz de ejecutarse con una velocidad considerablemente mayor a la de modelos anteriores como RetinaNet-50 y RetinaNet-101.

Este modelo se basa en el uso de redes neuronales convolucionales (CNN) que, al dividir la imagen en una cuadrícula, permiten calcular la probabilidad de ocurrencia de objetos en cada celda. Con esta información, el modelo determina tres parámetros clave para su funcionamiento:

- *Intersection over Union* (IoU)[8], se trata del criterio utilizado para eliminar cajas redundantes, y asegurar que solo se mantenga una por cada objeto detectado, incluso si hay varios objetos cercanos en la imagen. De esta manera, se evita la detección de objetos duplicados y se obtiene una detección más precisa y clara de los objetos presentes en la imagen.

  Se calcula a través de la siguiente fórmula:

  $IoU = \frac{A_s}{A_u}$

  Donde A_s representa el área de superposición entre ambas cajas y A_u el área de unión.

  Para lograr esto, cada celda es responsable de calcular la probabilidad de que haya un objeto dentro de sus fronteras. se genera una serie de cajas propuestas alrededor del objeto, y a través del uso de redes neuronales convolucionales se calcula la probabilidad de ocurrencia.

  Finalmente, el modelo elimina las cajas que no cumplen con dos criterios importantes: En primer lugar, se descartan aquellas cajas que tienen una probabilidad de ocurrencia menor que la máxima del objeto de la zona. Este algoritmo es conocido como *Non-Maximum Suppression* [9] y utiliza la probabilidad de ocurrencia, también conocida como *objectness score* [10], para seleccionar solo las cajas más relevantes. En segundo lugar, se eliminan las

8 Capítulo 2. Introducción específica

TABLA 2.1. Clases detectadas como intruso por el módulo

Clases detectadas como intruso	
Persona	Vaca
Camión	Bus
Automóvil	Sombrilla
Motocicleta	Mochila
Bicicleta	Maletín
Perro	Maleta
Caballo	Gato

Hay que recordar que la sola detección de una de estas clases generará un reporte positivo de intruso. El modelo no está en capacidad de identificar personas específicas en el vídeo, por lo que le resultará imposible diferenciar entre personal autorizado en un área (el mismo personal de seguridad, o la persona al mando del dron), y a intrusos en la zona.

2.2. Modelo utilizado

El modelo utilizado para el desarrollo de la totalidad del trabajo es YOLOv3. La principal ventaja que tiene este modelo es que es capaz de ejecutarse con una velocidad considerablemente mayor a la de modelos anteriores como RetinaNet-50 y RetinaNet-101.

Este modelo se basa en el uso de redes neuronales convolucionales (CNN) que, al dividir la imagen en una cuadrícula, permiten calcular la probabilidad de ocurrencia de objetos en cada celda. Con esta información, el modelo determina tres parámetros clave para su funcionamiento:

- *Intersection over Union* (IoU)[8], se trata del criterio utilizado para eliminar cajas redundantes, y asegurar que solo se mantenga una por cada objeto detectado, incluso si hay varios objetos cercanos en la imagen. De esta manera, se evita la detección de objetos duplicados y se obtiene una detección más precisa y clara de los objetos presentes en la imagen.

  Se calcula a través de la siguiente fórmula:

  $IoU = \frac{A_s}{A_u}$

  Donde A_s representa el área de superposición entre ambas cajas y A_u el área de unión.

  Para lograr esto, cada celda es responsable de calcular la probabilidad de que haya un objeto dentro de sus fronteras, se genera una serie de cajas propuestas alrededor del objeto, y a través del uso de redes neuronales convolucionales se calcula la probabilidad de ocurrencia.

  Finalmente, el modelo elimina las cajas que no cumplen con dos criterios importantes: en primer lugar, se descartan aquellas cajas que tienen una probabilidad de ocurrencia menor que la máxima del objeto de la zona. Este algoritmo es conocido como *Non-Maximum Suppression* [9] y utiliza la probabilidad de ocurrencia, también conocida como *objectness score* [10], para seleccionar solo las cajas más relevantes. En segundo lugar, se eliminan las

cajas que tienen un *Intersection Over Union (IoU)* mayor a un valor determinado, que suele ser 0.45, aunque también se puede ajustar como parámetro en el modelo.

- *Offset*: el *offset* se refiere a la distancia medida desde la esquina superior izquierda de la celda con mayor probabilidad de contener un objeto [11]. Esta medida es una forma eficiente de representar la posición de las cajas y reduce significativamente la cantidad de cálculos necesarios, lo que aumenta la eficiencia computacional del modelo [12]. Una vez obtenido el offset, se utilizan estos valores para calcular las coordenadas de las cajas que no fueron descartadas.
- Tensor: hace referencia a un vector utilizado para representar la probabilidad [13] de que cada uno de los objetos incluidos en el *dataset* COCO [6] se encuentre en la caja . Esta representación no solo permite determinar la posición de diferentes objetos en la imagen, sino también discriminarlos por clases. La clase con mayor probabilidad es entonces reportada por el modelo.

Es importante tener en cuenta que estos no son los parámetros finales arrojados por el modelo, sino que más adelante, la red calcula las coordenadas de la caja con la que se rodeará al objeto. Es decir, para cada objeto detectado, el modelo va a devolver (en ese orden):

- pc: *objectness score*, la probabilidad de que haya un objeto.
- bx: coordenada x del centro de la caja propuesta.
- by: coordenada y del centro de la caja propuesta.
- bh: altura de la caja propuesta.
- bw: ancho de la caja propuesta.
- c: tensor con la probabilidad de ocurrencia de las 80 clases que forman parte del dataset COCO.

Esto se logra a través del uso de Anchor Boxes: cajas de predicción predefinidas que el modelo va refinando a medida que analiza la imagen y calcula las métricas de cada uno de los objetos encontrados. A través de este método es posible eliminar la necesidad de una "ventana deslizante", lo que a su vez permite analizar la totalidad de la imagen en un tiempo mucho menor que los métodos basados en aquel método, y, en consecuencia, se hace posible llevar a cabo un análisis en tiempo real.

Dadas las bondades explicadas anteriormente, en conjunto con su facilidad de implementación en Python, se escogió partir del modelo YoloV3 y modificar su código fuente para lograr generar y extraer la información pertinente para completar el propósito del trabajo. Cabe aclarar que, la decisión de utilizar este modelo se vio reforzada una vez se ejecutó la primera versión en tanto que el modelo está optimizado para funcionar con la tarjeta gráfica (NVIDIA Titan X), el código se pudo ejecutar satisfactoriamente, sin dejar a un lado los requisitos del trabajo en CPU (AMD Ryzen 5).

2.2. Modelo utilizado 9

cajas que tienen un *Intersection Over Union (IoU)* mayor a un valor determinado, que suele ser 0.45, aunque también se puede ajustar como parámetro en el modelo.

- *Offset*: el *offset* se refiere a la distancia medida desde la esquina superior izquierda de la celda con mayor probabilidad de contener un objeto [11]. Esta medida es una forma eficiente de representar la posición de las cajas y reduce significativamente la cantidad de cálculos necesarios, lo que aumenta la eficiencia computacional del modelo [12]. Una vez obtenido el *offset*, se utilizan estos valores para calcular las coordenadas de las cajas que no fueron descartadas.
- Tensor: hace referencia a un vector utilizado para representar la probabilidad [13] de que cada uno de los objetos incluidos en el *dataset* COCO [6] se encuentre en la caja . Esta representación no solo permite determinar la posición de diferentes objetos en la imagen, sino también discriminarlos por clases. La clase con mayor probabilidad es entonces reportada por el modelo.

Es importante tener en cuenta que estos no son los parámetros finales arrojados por el modelo, sino que más adelante, la red calcula las coordenadas de la caja con la que se rodeará al objeto. Es decir, para cada objeto detectado, el modelo va a devolver (en ese orden):

- pc: *objectness score*, la probabilidad de que haya un objeto.
- bx: coordenada x del centro de la caja propuesta.
- by: coordenada y del centro de la caja propuesta.
- bh: altura de la caja propuesta.
- bw: ancho de la caja propuesta.
- c: tensor con la probabilidad de ocurrencia de las 80 clases que forman parte del *dataset* COCO.

Esto se logra a través del uso de *Anchor Boxes*: cajas de predicción predefinidas que el modelo va refinando a medida que analiza la imagen y calcula las métricas de cada uno de los objetos encontrados. A través de este método es posible eliminar la necesidad de una "ventana deslizante", lo que a su vez permite analizar la totalidad de la imagen en un tiempo mucho menor que los métodos basados en aquel método, y, en consecuencia, se hace posible llevar a cabo un análisis en tiempo real.

Dadas las bondades explicadas anteriormente, en conjunto con su facilidad de implementación en Python, se escogió partir del modelo YoloV3 y modificar su código fuente para lograr generar y extraer la información pertinente para completar el propósito del trabajo. Cabe aclarar que la decisión de utilizar este modelo se vio reforzada una vez se ejecutó la primera versión en tanto que el modelo está optimizado para funcionar con la tarjeta gráfica (NVIDIA Titan X), el código se pudo ejecutar satisfactoriamente, sin dejar a un lado los requisitos del trabajo en CPU (AMD Ryzen 5).

2.3. Protocolos de vídeo y transmisión

El módulo del trabajo encargado de extraer y permitir el cálculo sobre cada uno de los frames es OpenCV, por lo que los protocolos de vídeo aceptados corresponderán a los aceptados por OpenCV. En este sentido, el código está en capacidad de detectar intrusos en:

- Vídeo guardado localmente
- Secuencia de imágenes
- URL de streaming de vídeo
- Pipeline Gstreamer
- Datos obtenidos desde *webcam*

Sin embargo, es importante mencionar que, durante el desarrollo, se hizo pruebas únicamente con los métodos de vídeo guardado localmente y URL de streaming de vídeo. Los otros métodos no han sido probados, y, por lo tanto, no se podrá garantizar su correcto funcionamiento al solicitar al módulo ejecutarse utilizándolos como origen de datos.

Por otro lado, y recordando que se trata de la biblioteca que hace posible extraer la información del vídeo para hacer posible su análisis, es de vital importancia tener en mente cuáles son los formatos de vídeo admitidos por OpenCV. Dentro de ellos se encuentran:

- H264
- MPEG4
- AV1

Ahora bien, para acceder a ellos, bastará con pasar el link apropiado al método encargado. Dentro de los protocolos probados durante el desarrollo del trabajo se encuentran:

- MQTT: es un protocolo que permite la comunicación entre dispositivos desarrollado por IBM. Este protocolo es considerado muy seguro debido a que utiliza un intermediario llamado MQTT Broker para el intercambio de información entre el servidor y el cliente. De esta forma, ninguna de las partes tiene acceso directo a los datos del otro, visto que solo cuentan con la información para acceder al MQTT Broker. En cambio, el cliente solicita una suscripción a un tema que le interesa (en este caso la transmisión de vídeo), y el agente se encarga de distribuir los mensajes enviados por el servidor. Está pensado especialmente para dispositivos IoT con recursos limitados y conexiones con poco ancho de banda.
- RTMP: es un protocolo diseñado específicamente para la transmisión de audio y vídeo con gran rendimiento. Esto lo logra dividiendo la información en pequeños fragmentos, cuyo tamaño puede ser negociado de manera dinámica entre el cliente y el servidor de forma que la transmisión de vídeo se adapta mejor a las condiciones de la red de la que depende la transmisión. Sin embargo, dado que se trata de un protocolo diseñado para la transmisión de vídeo y audio con el menor delay posible, este protocolo puede no ser el más adecuado cuando las condiciones de la red son limitadas.

10 Capítulo 2. Introducción específica

2.3. Protocolos de vídeo y transmisión

El módulo del trabajo encargado de extraer y permitir el cálculo sobre cada uno de los *frames* es OpenCV, por lo que los protocolos de vídeo aceptados corresponderán a los aceptados por OpenCV. En este sentido, el código está en capacidad de detectar intrusos en:

- Vídeo guardado localmente
- Secuencia de imágenes
- URL de *streaming* de vídeo
- *Pipeline* Gstreamer
- Datos obtenidos desde *webcam*

Sin embargo, es importante mencionar que, durante el desarrollo, se hicieron pruebas únicamente con los métodos de vídeo guardado localmente y URL de *streaming* de vídeo. Los otros métodos no han sido probados, y, por lo tanto, no se podrá garantizar su correcto funcionamiento al solicitar al módulo ejecutarse utilizándolos como origen de datos.

Por otro lado, y recordando que se trata de la biblioteca que hace posible extraer la información del vídeo para hacer posible su análisis, es de vital importancia tener en mente cuáles son los formatos de vídeo admitidos por OpenCV. Dentro de ellos se encuentran:

- H264
- MPEG4
- AV1

Ahora bien, para acceder a ellos, bastará con pasar el link apropiado al método encargado. Dentro de los protocolos probados durante el desarrollo del trabajo se encuentran:

- MQTT: es un protocolo que permite la comunicación entre dispositivos desarrollado por IBM. Este protocolo es considerado muy seguro debido a que utiliza un intermediario llamado MQTT Broker para el intercambio de información entre el servidor y el cliente. De esta forma, ninguna de las partes tiene acceso directo a los datos del otro, visto que solo cuentan con la información para acceder al MQTT Broker. En cambio, el cliente solicita una suscripción a un tema que le interesa (en este caso la transmisión de vídeo), y el agente se encarga de distribuir los mensajes enviados por el servidor. Está pensado especialmente para dispositivos IoT con recursos limitados y conexiones con poco ancho de banda.
- RTMP: es un protocolo diseñado específicamente para la transmisión de audio y vídeo con gran rendimiento. Esto lo logra dividiendo la información en pequeños fragmentos, cuyo tamaño puede ser negociado de manera dinámica entre el cliente y el servidor de forma que la transmisión de vídeo se adapta mejor a las condiciones de la red de la que depende la transmisión. Sin embargo, dado que se trata de un protocolo diseñado para la transmisión de vídeo y audio con el menor *delay* posible, este protocolo puede no ser el más adecuado cuando las condiciones de la red son limitadas.

2.4. Software y hardware utilizados

Para su funcionamiento, este trabajo requiere la presencia de algunos paquetes de software instalados en la máquina, los cuales son llamados directamente por el usuario cuando requiera la funcionalidad (por ejemplo, OBS Studio), o bien, que ya vienen incorporados en el código y por lo tanto, son un requisito indispensable para el trabajo y deberán encontrarse instalados. Dicho esto, es posible clasificar el software utilizado en tres categorías:

1. Paquetes de Python: son paquetes importados en el código a través del comando *import*. Cabe mencionar que todos los paquetes son de código abierto. Ejemplos de estos paquetes son:

- Numpy
- Matplotlib
- OpenCV

2. Código en Python: en especial para la implementación de la metodología YOLOV3. En internet está disponible el código fuente en formato Python (py).

3. Software completo: en particular para la retransmisión del vídeo procesado, se utilizó OBS estudio. Esta decisión responde a que, dentro del alcance del trabajo, no se definió ninguna actividad que no se encuentre directamente relacionada con la implementación de la detección de intrusos. Sin embargo, dado que se trata de un trabajo en el que se debe devolver la imagen procesada, es importante asegurarse que la retransmisión de los datos funcione de la manera correcta.

Más específicamente, las diferentes bibliotecas y paquetes de software utilizados para el desarrollo de este trabajo son:

TABLA 2.2. Software requerido para el desarrollo y la ejecución del módulo

Software requerido	
Paquete	Función
PIP	Gestor de bibliotecas para Python
Numpy	Cálculos matriciales
OpenCV	Manipulación de imágenes y visión por computadora
Matplotlib	Creación de visualizaciones con Python
Tensorflow	Framework de machine learning
Pycharm	Ambiente de desarrollo
OBS Studio	Streaming de vídeo
Python	Lenguaje en el que fue programado el módulo
Ubuntu	Sistema operativo

Es importante tener en cuenta que la totalidad de programas y bibliotecas utilizadas, tanto para el desarrollo, como para la ejecución del trabajo son software libre, por lo que no se requerirá adquirir ningún tipo de licencia para ponerlo en funcionamiento. En el apéndice A se encuentra la tabla completa que incluye tipo de biblioteca, licencia y versión utilizada

2.4. Software y hardware utilizados 11

2.4. Software y hardware utilizados

Para su funcionamiento, este trabajo requiere la presencia de algunos paquetes de software instalados en la máquina, los cuales son llamados directamente por el usuario cuando requiera la funcionalidad (por ejemplo, OBS Studio), o bien, que ya vienen incorporados en el código y por lo tanto, son un requisito indispensable para el trabajo y deberán encontrarse instalados. Dicho esto, es posible clasificar el software utilizado en tres categorías:

1. Paquetes de Python: son paquetes importados en el código a través del comando *import*. Cabe mencionar que todos los paquetes son de código abierto. Ejemplos de estos paquetes son:

- Numpy
- Matplotlib
- OpenCV

2. Código en Python: en especial para la implementación de la metodología YOLOV3. En internet está disponible el código fuente en formato Python (py).

3. Software completo: en particular para la retransmisión del vídeo procesado, se utilizó OBS Studio. Esta decisión responde a que, dentro del alcance del trabajo, no se definió ninguna actividad que no se encuentre directamente relacionada con la implementación de la detección de intrusos. Sin embargo, dado que se trata de un trabajo en el que se debe devolver la imagen procesada, es importante asegurarse que la retransmisión de los datos funcione de la manera correcta.

Más específicamente, las diferentes bibliotecas y paquetes de software utilizados para el desarrollo de este trabajo son:

TABLA 2.2. Software requerido para el desarrollo y la ejecución del módulo

Software requerido	
Paquete	Función
PIP	Gestor de bibliotecas para Python
Numpy	Cálculos matriciales
OpenCV	Manipulación de imágenes y visión por computadora
Matplotlib	Creación de visualizaciones con Python
Tensorflow	*Framework* de machine learning
Pycharm	Ambiente de desarrollo
OBS Studio	*Streaming* de vídeo
Python	Lenguaje en el que fue programado el módulo
Ubuntu	Sistema operativo

Es importante tener en cuenta que la totalidad de programas y bibliotecas utilizadas, tanto para el desarrollo, como para la ejecución del trabajo son software libre, por lo que no se requerirá adquirir ningún tipo de licencia para ponerlo en funcionamiento. En el apéndice A se encuentra la tabla completa que incluye tipo de biblioteca, licencia y versión utilizada