Escuela Técnica Superior de Ingeniería
Departamento de Ingeniería de Computadores
Facultad de Informática de A Coruña
UNIVERSIDADE DA CORUÑA

TRABAJO FIN DE MÁSTER
MÁSTER INTERUNIVERSITARIO EN
COMPUTACIÓN DE ALTAS PRESTACIONES

Reconocimiento facial con aprendizaje semisupervisado en arquitecturas embebidas y de altas prestaciones

Estudiante: Omar Montenegro Macía
Director/a/es/as: Carlos Vázquez Regueiro

A Coruña, 30 de noviembre de 2025.

A la mama, pase lo que pase

Agradecimientos

a

Resumen

El auge del Deep Learning (DL) y las redes neuronales profundas han impulsado numerosos avances en aplicaciones como la visión artificial. Sin embargo, su implementación en sistemas embebidos en tiempo real (conocido como TinyML) sigue representando un reto debido a los elevados recursos computacionales que requieren. La optimización de dichas redes para su ejecución en sistemas embebidos es clave para proporcionar el internet de las cosas (IoT) y otras aplicaciones relacionadas.

Este proyecto propone (**ha logrado**) la migración y optimización de un método de adaptación a los cambios mediante aprendizaje incremental semisupervisado llamado Dynamic Ensemble of SVM (De-SVM) en el sistema embebido NVIDIA Jetson Orin Nano. Dicho método se integrará en un sistema de detección y reconocimiento de personas con el fin de implementar la clasificación de personas desconocidas y su inclusión (Open-World) bajo un modelo de reconocimiento facial. El sistema está implementado en la arquitectura ROS, que permite ejecutar procesos de forma distribuida en diferentes máquinas.

Para evaluar el rendimiento, se analizarán métricas de velocidad de inferencia y precisión, tanto de forma aislada (en interacción con el modelo de reconocimiento facial) como en conjunto con el sistema completo. Adicionalmente de la NVIDIA Jetson, se evaluará el procesador del robot móvil Summit_XL y otros dispositivos con aceleradoras, simulando distintas cargas de trabajo según el número de cámaras activas y personas detectadas en escena.

Abstract

El auge del Deep Learning (DL) y las redes neuronales profundas han impulsado numerosos avances en aplicaciones como la visión artificial. Sin embargo, su implementación en sistemas embebidos en tiempo real (conocido como TinyML) sigue representando un reto debido a los elevados recursos computacionales que requieren. La optimización de dichas redes para su ejecución en sistemas embebidos es clave para proporcionar el internet de las cosas (IoT) y otras aplicaciones relacionadas.

Este proyecto propone (**ha logrado**) la migración y optimización de un método de adaptación a los cambios mediante aprendizaje incremental semisupervisado llamado Dynamic Ensemble of SVM (De-SVM) en el sistema embebido NVIDIA Jetson Orin Nano. Dicho método se integrará en un sistema de detección y reconocimiento de personas con el fin de implementar la clasificación de personas desconocidas y su inclusión (Open-World) bajo un modelo de reconocimiento facial. El sistema está implementado en la arquitectura ROS, que permite ejecutar procesos de forma distribuida en diferentes máquinas.

Para evaluar el rendimiento, se analizarán métricas de velocidad de inferencia y precisión, tanto de forma aislada (en interacción con el modelo de reconocimiento facial) como en conjunto con el sistema completo. Adicionalmente de la NVIDIA Jetson, se evaluará el procesador del robot móvil Summit_XL y otros dispositivos con aceleradoras, simulando distintas cargas de trabajo según el número de cámaras activas y personas detectadas en escena.

Palabras chave:

- Open-Set
- Open-World
- Cámaras RGBD
- Reconocimiento facial
- SVM
- Jetson
- TensorRT
- ROS

Keywords:

- Open-Set
- Open-World
- RGBD cameras
- Facial recognition
- SVM
- Jetson
- TensorRT
- ROS

Índice general

1 Introducción **3**

1.1 Motivación . . . 3
1.2 Objetivos . . . 4
1.3 Estructura de la memoria . . . 4

2 Gestión del proyecto **5**

2.1 Requisitos . . . 5
2.1.1 Funcionales . . . 5
2.1.2 No funcionales . . . 5
2.1.3 Actores . . . 6
2.1.4 Casos de uso . . . 6
2.2 Gestión de riesgos . . . 6
2.3 Metodología Kanban . . . 6
2.4 Planificación . . . 6
2.4.1 T1. Instalación del sistema en los diferentes dispositivo . . . 6
2.4.2 T2. Realización de pruebas en los diferentes equipos . . . 7
2.4.3 TODO: Ti. Creación de nuevos datasets . . . 7
2.4.4 TODO: Ti. Nuevos sistemas embebidos: AGX Jetson y Thor . . . 7
2.4.5 (TODO: no estaba previsto inicialmente) T3. Migración ROS 2 y actualización de librerías . . . 7
2.4.6 T4. Estudio del sistema Open-World . . . 7
2.4.7 T5. Implementación del sistema Open-World . . . 7
2.4.8 T6. La ejecución eficiente en sistemas embebidos . . . 8
2.4.9 T7. Documentación del proyecto . . . 8
2.5 Seguimiento . . . 8
2.5.1 Análisis de desviaciones temporales . . . 8

3 Fundamentos teóricos y tecnológicos **9**
3.1 Métricas de evaluación . . . 9
3.2 Evaluación de modelos . . . 10
3.2.1 Modelos de detección . . . 10
3.2.2 Modelos de reconocimiento . . . 10
3.3 Evaluación de la adaptación . . . 11
3.4 Hardware . . . 11
3.4.1 Summit_XL . . . 11
3.4.2 LiDAR . . . 11
3.4.3 Cámaras RGBD . . . 11
3.4.4 NVIDIA Jetson . . . 11
3.4.5 Equipos x86 . . . 12
3.5 Software . . . 12
3.5.1 ROS . . . 12
3.5.2 TensorRT . . . 12
3.5.3 ONNX . . . 12
3.5.4 OpenVino . . . 12
3.5.5 NumPy . . . 12
3.5.6 Scipy . . . 13
3.5.7 OpenCV . . . 13
3.5.8 Docker . . . 13
3.6 Sistema final . . . 13
3.7 Conceptos . . . 14
3.7.1 Computación en el borde (Edge computing) . . . 14
3.7.2 Reconocimiento open world . . . 14
3.7.3 Redes neuronales convolucionales (CNN) . . . 14
3.7.4 Reconocimiento facial . . . 14
3.7.5 Aprendizaje incremental semi-supervisado . . . 15
3.7.6 Support Vector Machine (SVM) . . . 15
3.8 Otras herramientas . . . 15
3.8.1 VS Code . . . 15
3.8.2 Git . . . 15
3.8.3 Trello . . . 15
3.8.4 Draw.io . . . 15

4 Sistema de reconocimiento y seguimiento de personas **17**
4.1 Arquitectura del sistema . . . 17
4.2 Modificaciones propuestas . . . 18

4.2.1 Ventanas de frames . . . 18
4.2.2 Escalabilidad y tolerancia a fallos de las cámaras . . . 19
4.2.3 Migración a ROS 2 . . . 19
4.3 Optimización de inferencias con TensorRT . . . 19
4.3.1 Realización de pruebas . . . 20
4.3.2 Resultados en dispositivos Jetson . . . 21
4.3.3 Resultados en equipos x86 . . . 22

5 Reconocimiento facial adaptativo **23**
5.1 Introducción . . . 23
5.1.1 Fundamentos . . . 23
5.1.2 Arquitectura del sistema adaptativo . . . 24
5.2 Implementación . . . 24
5.2.1 Inicialización . . . 24
5.2.2 *Ensemble Decision Function* (EDF) . . . 25
5.2.3 *Recognition Decision Function* (RDF) . . . 26
5.2.4 Update Module . . . 28
5.2.5 Limitation Module . . . 29
5.2.6 Creación de nuevos comités (*Open-World*) . . . 29
5.2.7 Coherencia entre cámaras . . . 30
5.2.8 otra cosa . . . 30

6 Pruebas y resultados **31**
6.1 Diseño de las pruebas . . . 31
6.2 Resultados Open-World . . . 31
6.3 Fuentes de los datos (nuestro dataset combinado con YTF) . . . 32
6.4 Pruebas del sistema final . . . 32
6.5 Solapamiento de secuencias: buffering . . . 32
6.6 Número de IoI . . . 32
6.7 Número de negativos . . . 32
6.8 Umbral de Weibull (TW) . . . 33
6.9 Percentiles . . . 33
6.10 Tamaño del comité . . . 33
6.11 Tamaño de la plantilla . . . 34
6.12 Tamaño de secuencia . . . 34
6.13 Solapamiento . . . 34
6.14 Inicialización del sistema . . . 35
6.15 Criterio de selección de frames para nuevas SVM . . . 35

6.16 Criterio de favorabilidad . . . 35
6.17 Creación de nuevos comités . . . 35
6.18 Open-set runtime . . . 36
6.19 Open-world selected . . . 36
6.19.1 Caso concreto: . . . 36
6.20 Open-world runtime . . . 36

7 Protección de templates **37**
7.1 Cifrado . . . 37
7.2 Hash . . . 37
7.3 Bloom filters . . . 37
7.4 Homomorphic encryption . . . 39
7.5 Random projection . . . 39
7.6 Random Distance Method . . . 40
7.6.1 Procedimiento . . . 40
7.7 Correspondencia en el espacio vectorial . . . 41

8 Conclusiones y trabajo futuro **43**
8.1 Conclusiones . . . 43
8.2 Trabajo Futuro . . . 43

A Apéndices **45**

Bibliografía **47**

Índice de figuras

4.1 Arquitectura general del sistema (figura extraída de [1]) 18
4.2 Funcionamiento de la secuenciación, como salida se obtiene una lista de frames agrupados por entidad 19
4.3 Componentes del nodo cámara 20

5.1 a 24
5.2 Pipeline del método De-SVM, figura extraída de [2] 26
5.3 Algoritmo de reconocimiento basado en EVT, extraido de [2] 27

Índice de cuadros

4.1 Resultados de rendimiento comparados entre dispositivos jetson.
** CPU=**OpenCV en CPU**; GPU=**OpenCV CUDA** 22

Capítulo 1
Introducción

En este capítulo se expone la motivación y los objetivos de este proyecto. Adicionalmente, se comenta la estructura por capítulos de esta memoria y brevemente su contenido.

1.1 Motivación

Con la aparición del Deep Learning y las redes neuronales profundas, se han logrado notables avances en términos de precisión en áreas como la visión artificial. Este éxito ha motivado a los investigadores a explorar su aplicación en ámbitos como el internet de las cosas (IoT), la robótica, entre otros. Sin embargo, debido a sus elevados requerimientos computacionales, los modelos de Deep Learning suelen ejecutarse en entornos en la nube [3]. En este contexto, surge el concepto de TinyML [4], que consiste en la implementación de modelos compactos de forma eficiente en sistemas embebidos como microcontroladores (Low-end TinyML, ej: Arduino) u ordenadores de placa única (High-end TinyML, ej: NVIDIA Jetson), más potentes que los microcontroladores, pero menos económicos.

En el campo de TinyML ya se han analizado varios modelos [3, 5] y propuesto múltiples optimizaciones [3] para sistemas NVIDIA Jetson. Este proyecto extenderá la investigación realizada aplicando TensorRT, para la ejecución eficiente de redes neuronales en varios dispositivos Jetson, en un entorno de operación real con múltiples cámaras. Se analizará el rendimiento de los modelos tanto de forma aislada como integrado en un sistema de detección y reconocimiento de personas en tiempo real [1], formado por otros modelos que convergen para lograr un mismo objetivo. Se usará el framework ROS [6], estándar de facto en robótica, que permite la ejecución distribuida de procesos en diferentes máquinas y posee una amplia biblioteca de sensores y algoritmos.

TODO: El método De-SVM implementa aprendizaje semi-supervisado, que permite el reconocimiento de caras a partir de un conjunto reducido de datos etiquetados y que se van actualizando en base a las propias predicciones del modelo. Dado que la recopilación de imá-

genes faciales está fuertemente regulada por la ley de protección de datos, el uso de este enfoque resulta especialmente relevante para el presente proyecto [2]. El objetivo es aplicar este método para la identificación de personas desconocidas (Open-Set) y su posterior inclusión (Open-World), ampliando así las capacidades del sistema propuesto [1].

1.2 Objetivos

Bibliografía de interés: TODO: han conseguido ejecutar un sistema de detección y seguimiento de personas en una Jetson TX2: [7]

TODO: hablan de la Jetson también[8]

1.3 Estructura de la memoria

Capítulo 2

Gestión del proyecto

En este capítulo se ahondará en la planificación del proyecto, su seguimiento y del plan de gestión de riesgos utilizado.

2.1 Requisitos

Los requisitos definen las pautas que el sistema desarrollado debe cumplir para alcanzar los objetivos establecidos. Los requisitos pueden dividirse en 2 tipos:

2.1.1 Funcionales

Definen las funcionalidades que el sistema debe de ofrecer. Para este proyecto, se busca ampliar el sistema base para que cumpla con los siguientes requisitos:

-
- Incorporación de nuevas cámaras: debe ser sencillo añadir nuevas cámaras en adición a las 2 iniciales.
- Pasar de reconocimientos frame a frame a ventanas de frames, aprovechando así la coherencia espacio-temporal.
- Funcionalidad Open-Set y Open-World: el sistema debe ser capaz de reconocer y agregar nuevas personas a su base de datos.

2.1.2 No funcionales

- Que el sistema evolucione adecuadamente (ejemplo: evitar caídas drásticas en la precisión) a lo largo del tiempo.
- Que la instalación del sistema sea rápida y prácticamente automática.

- Operación en tiempo real: sujeto a la frecuencia del componente más ”lento”. En el caso de este proyecto, las cámaras RGBD otorgan una imagen RGB cada 100 ms (**10 Hz**).
- Portabilidad: la ejecución del sistema no debe de depender exclusivamente de la arquitectura o de las diferentes versiones de librerías.
- Escalabilidad: debe permitir incluir nuevos componentes (ejemplo: cámaras) sin comprometer significativamente al rendimiento.

2.1.3 Actores

A continuación se definen los actores (o roles) implicados y sus papeles:

- Persona que se somete a las pruebas del sistema (ser detectada y reconocida).
- Operador del sistema: encargado de levantar y mantener el sistema.

2.1.4 Casos de uso

2.2 Gestión de riesgos

2.3 Metodología Kanban

Este enfoque basado en iteraciones ha permitido establecer pequeñas tareas concretas para así cumplir con los objetivos principales del proyecto.

2.4 Planificación

El trabajo correspondiente al presente proyecto se compone de las siguientes tareas, junto con sus hitos (“Hx”) y los entregables (“Ex”) correspondientes:

2.4.1 T1. Instalación del sistema en los diferentes dispositivo

- **Descripción y metodología**: Primero de todo, se creará un Dockerfile de despliegue del sistema para cada dispositivo del apartado de infraestructuras.
- **H1**. Sistema instalado y funcionando en los dispositivos.
- **E1**. Archivos Dockerfile de despliegue para cada dispositivo.

2.4.2 T2. Realización de pruebas en los diferentes equipos

- **Descripción y metodología**: Se realizarán diferentes pruebas para medir el rendimiento del sistema en los equipos del apartado de infraestructuras. Los análisis pondrán a prueba los componentes desarrollados frente a múltiples situaciones del sistema (ejemplo: variación del número de cámaras y/o número de personas en escena).
- **H2**. Resultados de rendimiento, de forma que se puedan utilizar para realizar comparaciones con otros dispositivos.

TODO: incluir las otras 2 Jetson

2.4.3 TODO: Ti. Creación de nuevos datasets

- **Descripción y metodología**: Se crearán nuevos datasets a partir de los datos de 2 cámaras RGBD con información de nuevas entidades, para medir la capacidad de aprendizaje Open-World del método a implementar.
- **H3**. Nuevos datasets creados útiles para las pruebas.
- **E3**. Datasets con información de entidades desconocidas por el sistema.

2.4.4 TODO: Ti. Nuevos sistemas embebidos: AGX Jetson y Thor

2.4.5 (TODO: no estaba previsto inicialmente) T3. Migración ROS 2 y actualización de librerías

2.4.6 T4. Estudio del sistema Open-World

- **Descripción y metodología**: Se estudiará qué partes del método Open-World escogido [2] se pueden adaptar al sistema propuesto. También se estudiarán los conceptos teóricos que los sustentan.
- **H4**. Obtener los conocimientos para aplicar la funcionalidad al sistema.

2.4.7 T5. Implementación del sistema Open-World

- **Descripción y metodología**: En base al conocimiento adquirido, se implementarán los componentes que mejor se adaptan al sistema y que permiten la funcionalidad Open-World.
- **H5**. Implementar el reconocimiento de desconocidos en el sistema.
- **E5**. Software con dicha implementación.

TODO: separar T4, fusionar T5 y T6.

- **Descripción y metodología**: Se realizarán múltiples análisis de la nueva funcionalidad en los equipos descritos en el apartado de infraestructuras.
- **H5**. Registrar varias pruebas empíricas del sistema.

2.4.8 T6. La ejecución eficiente en sistemas embebidos

- **Descripción y metodología**: Se aplicarán mejoras a modo de reducir la latencia sin comprometer a la precisión de los reconocimientos.
- **H6**. Obtener una mejora en el rendimiento.
- **E6**. Nuevos resultados acerca del rendimiento, junto con las optimizaciones realizadas.

2.4.9 T7. Documentación del proyecto

- **Descripción y metodología**: Los avances del proyecto se documentarán de forma continua en una memoria, con el objetivo de asegurar la reproducibilidad del trabajo.
- **H7**. Registro de todo el trabajo realizado.
- **E7**. Memoria con todo el trabajo registrado, de forma que pueda ser reproducido.

2.5 Seguimiento

2.5.1 Análisis de desviaciones temporales

Capítulo 3

Fundamentos teóricos y tecnológicos

En este capítulo se detallan las métricas de evaluación y herramientas hardware y software utilizadas, así como los conceptos a entender para el resto de la memoria.

3.1 Métricas de evaluación

Para evaluar la clasificación de los modelos de visión artificial utilizados, se emplearon los siguientes componentes cuyo significado se expone a continuación:

- ***True Positive (TP)***: predicción positiva correcta. Para los modelos de detección, sería detectar correctamente a una persona. Para los modelos de reconocimiento, asignar la etiqueta correcta a la persona detectada.

- ***True Negative (TN)***: predicción negativa correcta. Para los modelos de detección, sería no detectar correctamente a una persona, lo que carece de sentido a la hora de evaluar un buen detector. TODO (es mejor quitar esta métrica, no la uso en el sistema): Para los modelos de reconocimiento, sería predecir que la persona no pertenece a ninguna de las registradas, de aquí surge el concepto de **desconocido**.

- ***False Positive (FP)***: predicción positiva erronea. Para los modelos de detección, sería detectar una persona donde no la hay. Para los modelos de reconocimiento, sería otorgar una etiqueta incorrecta a la persona detectada.

- ***False Negative (FN)***: predicción negativa errónea. Para los modelos de detección, sería no detectar a la persona presente. Para los modelos de reconocimiento, sería asignar la entidad de desconocido a una persona ya registrada en el sistema.

Estos componentes se aplicarán a los datasets de prueba formados a partir de los videos grabados. En un entorno real de operación no sería posible aplicar dichas métricas por falta de un *ground truth* (TODO: ponerlo en el glosario de términos).

3.2 Evaluación de modelos

Las pruebas empleadas para evaluar los modelos de detección y reconocimiento son las mismas que en [1]. Teniendo en cuenta que ahora se considera el desconocido como entidad posible, en las pruebas de reconocimiento se añaden las métricas de *recall* y *F1_score* al tener falsos negativos (FN) y positivos (FP).

3.2.1 Modelos de detección

Se aplican las siguientes métricas:

- ***Precision***: **cuando el modelo detecta un objeto**, cuantas veces la detección corresponde con una persona presente.

$$Precision = \frac{TP}{TP + FP} \tag{3.1}$$

- ***Recall***: **cuando realmente hay una persona**, cuantas veces el modelo detecta a esa persona.

$$Recall = \frac{TP}{TP + FN} \tag{3.2}$$

- ***F1_score***: dicha métrica se calcula como una media armónica entre el *recall* (proporción de objetos detectados) y el *precision* (proporción de detecciones correctas) [1].

$$F1\_score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{3.3}$$

3.2.2 Modelos de reconocimiento

Las métricas TODO: se calculan de la misma forma, lo único que cambia es su significado que se expone a continuación:

- ***Precision***: refleja la proporción de aciertos cuando el modelo TODO: no asigna a una persona como desconocida.

$$Precision = \frac{TP + TN}{TP + TN + FP} \tag{3.4}$$

- ***Recall***: refleja la proporción de veces que el sistema TODO: decide no asignar la entidad desconocida.

$$Recall = \frac{TP + TN}{TP + TN + FN} \tag{3.5}$$

- ***F1_score***: lo mismo que en los modelos de detección.

Cuando el sistema tiene dudas acerca de la entidad de una persona, es preferible que se asigne dicha entidad como **desconocida** antes que otorgar una predicción **incorrecta**. Por este motivo, no se tiene en cuenta los falsos negativos a la hora de calcular la precisión, ni los falsos positivos cuando se calcula el recall.

3.3 Evaluación de la adaptación

Como se ahondará en el capítulo 5, uno de los factores más importantes a analizar es la capacidad del sistema de retener la información con la que se inicializó. Existe un riesgo de que el sistema elimine toda la información con la que fué entrenado (lo que se conoce como *catastrophic forgetting*)

TODO: César emplea una métrica para analizar la homogeneidad de los comités, analizar el catastrophic forgeting (comprobar si el sistema no ha eliminado su memoria inicial).

3.4 Hardware

3.4.1 Summit_XL

3.4.2 LiDAR

3.4.3 Cámaras RGBD

TODO: habremos usado las 4 kinects?

3.4.4 NVIDIA Jetson

Otra característica destacable es su bajo consumo, que oscila entre los 7 W en el modo menos potente y 25 W en el modo de mayor rendimiento, lo que es muy ligero en comparación con una CPU Intel i7 de 125 W de consumo `https://es.whatpsu.com/psu/cpu/Intel-Core-i7-13700K/gpu/NVIDIA-GeForce-RTX-3050-GB`, lo que la hace ideal sobretodo en robótica móvil para alargar la vida de la batería.

No está pensada para entrenar modelos de IA.

3.4.5 Equipos x86

Para las pruebas se emplearon dos equipos con las siguientes características:

- **Equipo 1 (portátil personal)**: cuenta con una CPU **Intel i7 i7-12650H**, una GPU **NVIDIA GeForce RTX 3050** con **4 GB de RAM**. Tiene una memoria principal de **16 GB** y el sistema operativo Linux Mint 21.3 (equivalente a un Ubuntu 22.04).
- **Equipo 2 (PC del laboratorio del CITIC)**: cuenta con una CPU **Intel i7 i7-12700**, una GPU **NVIDIA GeForce GTX 1650** con **4 GB de RAM**. Tiene una memoria principal de **32 GB** y el sistema operativo Ubuntu 24.04.

3.5 Software

3.5.1 ROS

Robot Operating System (ROS) es un middleware de **código abierto** que incorpora las herramientas necesarias para la interacción y desarrollo de software en robots. Cuenta con una amplia biblioteca de distribuciones y librerías. TODO

3.5.2 TensorRT

Es un software de **código abierto** desarrollado por NVIDIA para la optimización de inferencias de modelos de IA en aceleradoras de NVIDIA. Permite ejecutar modelos entrenados en frameworks como Pytorch y Tensorflow, aunque se aconseja exportarlos al formato ONNX previamente, para aprovechar al completo las funcionalidades de este software. Se ha empleado TensorRT para optimizar la mayoría de modelos CNN tanto en equipos ARM (Jetson) como x86 (Intel).

3.5.3 ONNX

3.5.4 OpenVino

Es un software de **código abierto** desarrollado por Intel para la optimización de inferencias en CPUs (ARM,x86) y en aceleradoras de Intel (GPUs,NPUs). Permite la conversión directa de modelos entrenados en frameworks como Tensorflow y Pytorch. Se ha empleado para reducir la latencia de modelos a la hora de ejecutarlos en CPUs.

3.5.5 NumPy

Librería de Python de **código abierto** usada en este proyecto para representar imágenes y la información generada por los modelos en forma de vectores y matrices multidimensio-

nales. Estos elementos pueden manipularse por medio de una amplia cantidad de funciones matemáticas que dicha librería ofrece.

3.5.6 Scipy

Libreria de Python de **código abierto** que implementa numerosos algoritmos relacionados con la computación científica. En este proyecto se ha usado para aplicar el método húngaro, hallar los parámetros que modelan una distribución de Weibull, calcular distancias coseno, entre muchas otras funcionalidades que dicho software ofrece.

3.5.7 OpenCV

La librería de **código abierto** por excelencia para visión por computadora de código abierto. Otorga herramientas para la obtención y procesado de las imágenes, además de un módulo de visión artificial (dnn), que incluye modelos ya entrenados de visión artificial. Dicha librería tiene implementación en CUDA (si se compila el código fuente), lo que permite ejecutar operaciones y kernels de modelos en aceleradoras de NVIDIA (ejemplo: NVIDIA Jetson).

3.5.8 Docker

Es un software de virtualización de **código abierto**. Permite realizar despliegues automatizados mediante ficheros de configuración, el software se ejecuta en un entorno aislado del sistema operativo llamado **contenedor**, dichos contenedores proporcionan seguridad y portabilidad. Esta herramienta ha sido de gran utilidad para desplegar el software del proyecto en los diferentes dispositivos.

3.6 Sistema final

TODO: El sistema final es una fusión de los modelos de la sección 3.2 (en la sección 4.1 se detalla su arquitectura), además de un modelo de seguimiento de personas a partir de las nubes de puntos otorgadas por un sensor **LiDAR** instalado en el robot. En el inicio de este proyecto, el sistema estaba sustentado en ROS 1 Noetic (ver sección 3.5), compatible con el ROS 1 Melodic instalado en el Summit_XL. Como se comentará en la sección 4.2.3, se ha realizado una migración del sistema a ROS 2. El objetivo era migrar la versión de ROS del robot en consonancia, sin embargo, debido a que la migración no es trivial y el LiDAR instalado no era compatible con ROS 2, se han migrado todos los componentes excepto el seguimiento con el LiDAR.

Las métricas para evaluar el sistema final son las mismas que en [1]:

- ***Accuracy_id***: es la métrica *precision* de la sección 3.2.2.

- **Desconocidos (*unks*)**: esta métrica indica a qué porcentaje de detecciones no se les ha podido asignar una identidad. El objetivo es poder comparar si las variaciones en la métrica *Accuracy_id* se producen a cambio de dejar de identificar a más gente.
- ***Precision_det*** : se refiere a la métrica *precision* definida en la sección 3.2.1.
- ***Recall_det*** : es la métrica *recall* definida en la sección 3.2.1.

3.7 Conceptos

3.7.1 Computación en el borde (Edge computing)

En proyectos en los que se demanda el procesamiento de datos en tiempo real, es imprescindible obtener los resultados de forma inmediata para la correcta operabilidad del sistema. El edge computing mantiene el procesamiento de los datos lo más cerca posible de la fuente que los suministra, un ejemplo sería tener una cámara conectada localmente a un equipo, de manera que se reduce la latencia de transmisión por la red y todo el ancho de banda que implicaría transmitir una imagen, entre otras muchas ventajas.

3.7.2 Reconocimiento open world

Atendiendo a la definición de [9], un sistema de reconocimiento open world debe de ser capaz de realizar las siguientes 4 tareas:

- Detectar desconocidos: identificar cuando una muestra de entrada no pertenece al conjunto de datos del entrenamiento. TODO: Un ejemplo sería definir una **función de Weibull** adaptada a los datos del entrenamiento.
- Escoger los puntos que puedan aportar información del desconocido al modelo
- Etiquetar dichos puntos (con un número o un pseudónimo).
- Actualizar el modelo: TODO: **re-entrenar los clasificadores**.

3.7.3 Redes neuronales convolucionales (CNN)

3.7.4 Reconocimiento facial

Representación de características (Feature embedding)

Se refiere a la representación de los vectores de características (información generada por los modelos) a un espacio vectorial **más pequeño**, de forma que la relación entre los vectores dentro del espacio **no se rompa** [10].

La tesis de De-SVM utiliza una ResNet sin la capa de clasificación (sección 3.4.2)[2].

3.7.5 Aprendizaje incremental semi-supervisado

En el campo de la visión artificial, existe una gran cantidad de parámetros que afectan al contexto de adquisición de los datos para el entrenamiento de modelos deep learning (CNN). Por desgracia, dichos modelos son susceptibles a estos cambios en el contexto. Por ejemplo, el modelo ArcFace [11] utiliza datasets con miles de recortes de caras centradas para su entrenamiento. Sin embargo el modelo, en el contexto de este proyecto, se tiene que enfrentar a condiciones de imagen diferentes, así como tener que reconocer personas a diferentes distancias y poses. Por este motivo, es necesaria la perspectiva de la **adaptación** para generalizar el uso de CNNs a cualquier contexto.

El aprendizaje incremental **no supervisado** usa sus propias predicciones a partir de datos entrantes (datos no etiquetados) para refinar los datos ya registrados. De esta forma, el modelo se **auto-entrena**, mejorando así su capacidad de reconocimiento y eliminando la necesidad de re-entrenar el modelo. Debido a que puede ser necesaria la intervención humana para etiquetar ciertas muestras, el enfoque de dicho aprendizaje se vuelve **semi-supervisado**.

Es muy importante que las imágenes recolectadas de las identidades **sean de calidad**, sino la estrategia de auto-entrenamiento **no será eficaz**.

3.7.6 Support Vector Machine (SVM)

Es un algoritmo de **clasificación** que dibuja un hiperplano entre categorías de datos, buscando siempre el mayor margen (distancia entre el punto que define la frontera de la categoría, denominado support vector, y el hiperplano), de modo que agrega cierta tolerancia al posible ruido producido. Es un **algoritmo de aprendizaje supervisado** (requiere de un conjunto inicial de datos para su entrenamiento) y conforma la unidad más básica de clasificación dentro del sistema de este proyecto [12].

3.8 Otras herramientas

3.8.1 VS Code

3.8.2 Git

3.8.3 Trello

3.8.4 Draw.io

Capítulo 4

Sistema de reconocimiento y seguimiento de personas

En este capítulo se exponen los resultados de rendimiento del sistema propuesto para varias arquitecturas. Se ha partido de una implementación final del sistema pensada para la arquitectura x86, este capítulo explica las optimizaciones aplicadas para su ejecución en arquitecturas ARM64.

4.1 Arquitectura del sistema

El sistema de este proyecto posee la capacidad de fusionar diferentes fuentes de datos para otorgar un único resultado de reconocimiento para cada persona detectada. Los nodos que componen la arquitectura se detallan a continuación y se muestran en la figura 4.1:

- Nodo cámara: recibe como fuente los datos de una cámara **RGBD** y se encarga de detectar, reconocer y obtener la posición 3D de las personas que aparecen en la imagen.

- Nodo LiDAR: recibe los datos de un sensor LiDAR (nube de puntos 3D) y se encarga de detectar y obtener la posición 3D de las personas del mapa (no se ha implementado el reconocimiento).

- Nodo integrador de sensores: fusiona los datos de los 2 nodos anteriores. La función de este nodo es el seguimiento de las personas reconocidas, más allá de lo que la cobertura de las cámaras ofrece (para un Kinect de la Xbox 360 son 57º).

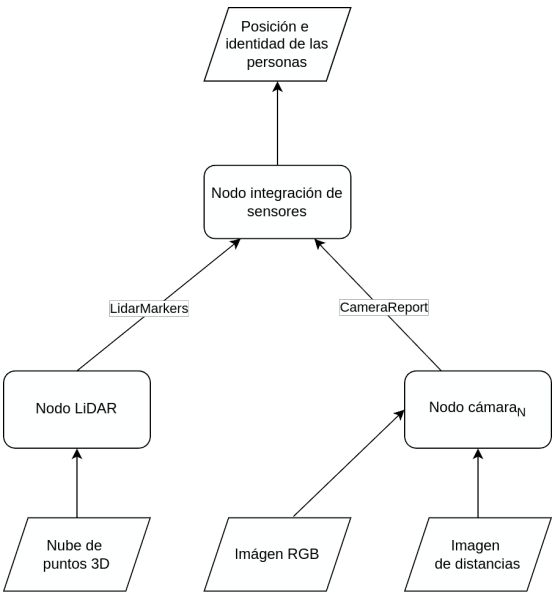

Figura 4.1: Arquitectura general del sistema (figura extraída de [1])

4.2 Modificaciones propuestas

4.2.1 Ventanas de frames

El sistema expuesto trabaja a nivel de frame, lo que otorga sencillez y una elevada frecuencia, pero al mismo tiempo un alto coste computacional y una no muy alta confianza (el reconocimiento depende enteramente del frame capturado). En este proyecto se ha optado por trabajar con **secuencias de frames**, de manera que se exprimen las capacidades de la GPU para trabajar por lotes (conjuntos de frames) y se otorgan mejores reconocimientos (basados en múltiples frames). El funcionamiento de esta técnica se muestra en la figura (TODO), se escoge un tamaño de secuencia (15 o 25 en las pruebas realizadas) y se extraen los frames. En cada frame, puede haber entre 0 y n entidades, como no se conoce la identidad de cada individuo en este punto, es necesario realizar un **rastreo** (tracking) de las entidades en cada uno de los frames. El tracking implementado calcula una matriz de costes entre las detecciones de un frame y el frame posterior, cada coste representa un valor de solape entre bounding boxes, cuanto menor mayor es el solape, finalmente, se realiza una asignación óptima entre detecciones mediante el método húngaro (como se plantea en [13]).

Re identificación de entidades

Es posible que durante la detección de individuos, las bounding boxes de una misma persona en frames consecutivos no se solapen, debido a la velocidad de movimiento del individuo

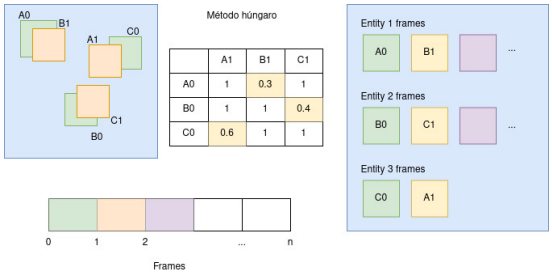

Figura 4.2: Funcionamiento de la secuenciación, como salida se obtiene una lista de frames agrupados por entidad

por ejemplo. En este caso, existe el riesgo de que el método húngaro asocie incorrectamente la identificación a otra persona a la que le ocurre esta misma situación. Otro problema es el no seguimiento de la persona cuando esta se encuentra totalmente ocluida (ejemplo: se cruza un individuo justo delante) y vuelve a aparecer.

A partir del método de tracking propuesto no es posible recuperar el rastro de las entidades que desaparecieron de forma intermitente. Para resolver este problema se ha optado por calcular la distancia euclidiana entre la última bounding box del individuo en cuestión y la bounding box de una nueva detección, si la distancia entre ellas es menor a un umbral, se reasigna la detección al individuo.

4.2.2 Escalabilidad y tolerancia a fallos de las cámaras

TODO: con el sistema inicial, sólo se podían ejecutar 2 cámaras y tenían que funcionar las 2 a la vez. Por estas razones, se ha reemplazado la solución del *ApproximateTimeSynchronizer* implementada en [1] por la clase *MessageFiltersCache*.

4.2.3 Migración a ROS 2

Con el motivo del fin de soporte de ROS 1 [14], se ha optado por migrar el sistema para ser ejecutado en ROS 2 con el fin de mantener su continuidad. Se han migrado los nodos cámara e integrador, la migración del nodo LiDAR requeriría actualizar el código a una versión soportada para Ubuntu 22.04 de la librería pcl, entre otros detalles que llevarían a rediseñar casi todo el código. TODO: Se probaron diferentes alternativas como RoboStack

4.3 Optimización de inferencias con TensorRT

TODO: Se han recortado bucles for y se ha exprimido la librería de numpy, se han reducido unos milisegundos.

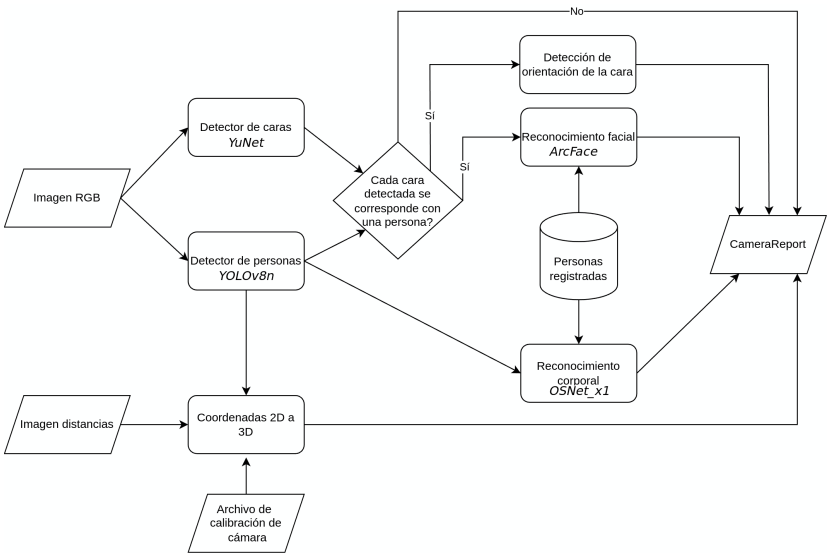

Figura 4.3: Componentes del nodo cámara

El objetivo de este capítulo es descubrir la capacidad de los dispositivos Jetson de poder asumir la carga de trabajo del sistema de forma que pueda escalarse. El rendimiento esperado en tiempo real para el sistema son los **10 Hz** (o 10 FPS), por lo tanto, todos los nodos deben de funcionar a 10 Hz. Las NVIDIA Jetson sólo tienen que ocuparse de ejecutar los **nodos cámara a 10 Hz** y que otro equipo ejecute el nodo LiDAR e integrador del sistema.

Que los nodos cámara trabajen a 10 Hz implica que a cada **100 ms se recibe un nuevo frame** (10 * 100 = 1000 ms = 1 segundo), la figura 4.3 muestra el procesado realizado por dicho nodo en cada frame, que debe de finalizar **antes de que transcurran 100 ms**. En cada iteración del procesado, se ejecutan **4 redes neuronales (CNN)**. 2 de ellas, YOLOv8n y YuNet, para la detección de cuerpos y de caras respectivamente. Las otras 2, OSNet_x1 y ArcFace, para el reconocimiento de cuerpos y caras respectivamente. A pesar de lo que la figura 4.3 muestra, las redes de detección YOLOv8n y YuNet **no se ejecutan en paralelo**, lo mismo sucede con las redes ArcFace y OSNet_x1. Se ha considerado ejecutar en paralelo estas redes. Sin embargo, las limitaciones de memoria principal de la Jetson restringen tomar esta aproximación.

TODO: En equipos x86 esta meta ya se ha cumplido. Sin embargo, para la arquitectura de la Jetson (ARM64) ha sido necesario aplicar varios cambios.

4.3.1 Realización de pruebas

Las pruebas del sistema se han focalizado en evaluar las redes neuronales utilizadas en los nodos cámara y el rendimiento del sistema completo. Para este fin, se cuentan con **datasets** generados a partir de los datos grabados de una prueba de 2 minutos dentro del laboratorio

de robótica del CITIC con 6 personas.

Para las pruebas de los modelos de detección, se reproduce el video para que el modelo realize las detecciones en cada frame. Las predicciones obtenidas por el modelo se comparan contra el dataset para evaluar si son correctas. En la tabla (TODO) se muestra para cada modelo de detección (YOLOv8n y YuNet) la métrica F1 (definida en 3.1).

Para las pruebas de los modelos de reconocimiento, se guardan como imágenes los recortes de los cuerpos y de las caras de cada individuo en cada frame del video grabado por la cámara. Para cada recorte, se ejecuta el modelo y se compara su predicción con la recogida en el dataset. En la tabla (TODO), se utiliza para los modelos de reconocimiento (ArcFace y OSNet_x1) la métrica de **precisión** (proporción de reconocimientos correctos) [1].

4.3.2 Resultados en dispositivos Jetson

El sistema cuenta con un motor de inferencia de redes neuronales orientado a procesadores y gráficas **Intel**, llamado **OpenVINO** [15]. OpenVINO otorga herramientas para convertir archivos ONNX (formato reconocido para exportar redes neuronales) a ficheros XML, que son versiones **optimizadas** de las redes neuronales y que se utilizan para realizar las inferencias. Todas las redes salvo YuNet (que utiliza el motor de OpenCV en CPU) se han exportado a un equivalente optimizado en OpenVINO.

Los primeros resultados arrojados en la tabla (TODO) muestran que las placas Jetson **no son capaces de cumplir el objetivo de 10 Hz** usando OpenVINO. Sólo la ejecución de YOLOv8n consume los 100 ms en la Jetson Xavier NX. En cuanto a la Jetson Orin Nano, la suma de la ejecución de las 4 redes supera también los 100 ms.

TensorRT es un framework para inferencia de alto rendimiento creado por NVIDIA, pensado para la ejecución de redes neuronales en sistemas embebidos de NVIDIA como es la Jetson [3, 5].

TensorRT se ha empleado para optimizar modelos ONNX al formato propio de la herramienta (denominado engine) y así poder realizar inferencias a partir de su API. Un ejemplo de código de inferencia en TensorRT es el que se muestra en TODO.

Todas las redes neuronales fueron exportadas a un engine de TensorRT salvo YuNet, que utiliza **OpenCV con CUDA**. Para disponer de soporte CUDA, es necesario compilar OpenCV **desde el fuente** [16].

NVIDIA consta que con TensorRT, el Speed up de las inferencias aumenta **36 veces**. En este caso se ha logrado reducir hasta **5 veces** el tiempo de inferencia en el caso de YOLOv8n en la Jetson Xavier NX. Si sumamos todos los tiempos de inferencia usando el motor de TensorRT, da un total de **63.6 milisegundos** para la Jetson Xavier NX y **49.1 milisegundos** para la Jetson Orin Nano.

	Sobremesa							*Siste*
	PC lab		**Portátil**		**Jetson Xavier NX**		**Jetson Orin Nano**	
	Openvino	TensorRT	Openvino	TensorRT	Openvino	TensorRT	Openvino	Tensor
YOLO11n (640x640)					L: 146.3 F1: 88	L: 28.7 F1: 87.7	L: 77.9 F1: 88	L: 22.4 F1: 89.1
*YuNet **(480x480)*					L: 33.5 F1: 87.9	L: 14.5 F1: 87.9	L: 14.9 F1: 87.9	L: 11.8 F1: 87.9
ArcFace (112x112)					L: 23.5 P: 44.6	L: 7 P: 45	L: 15.9 P: 44.6	L: 4.8 P: 45.2
OSNet_x1 (256x128)					L: 45.1 P: 54.3	L: 13.4 P: 59.2	L: 21.7 P: 59.4	L: 10.1 P: 59.2

Cuadro 4.1: Resultados de rendimiento comparados entre dispositivos jetson.
** CPU=**OpenCV en CPU**; GPU=**OpenCV CUDA**

4.3.3 Resultados en equipos x86

Afortunadamente, todas las librerías del proyecto cuenta con soporte para ambas arquitecturas x86 y ARM64, por lo que fué prácticamente inmediato el despliegue entre dispositivos mediante Docker (exceptuando pequeños cambios en el código por el cambio de versión de TensorRT y omitir ciertas optimizaciones (ejemplo: flag msse de las CPUs x86)).

Una ventaja de TensorRT es la **fácil portabilidad del código**, se puede ejecutar un mismo script de inferencia en cualquier equipo que posea una GPU de NVIDIA, siempre que sea posible instalar la misma versión de TensorRT (en caso contrario, es necesario realizar un cambio en el código, aunque este es leve).

En la tabla (TODO) se muestran los resultados obtenidos en 2 equipos x86, comparándolos con la Jetson Orin Nano, el sistema embebido con el que se ha obtenido el mejor rendimiento. En equipos x86, la latencia experimentada es hasta **5.3 veces menor** (YOLOv8n) para los modelos ejecutados en CPU y hasta **4.9 veces menor** (YuNet) para los modelos ejecutados en GPU que la obtenida por la Jetson Orin Nano. Debido a que OpenVINO está pensado para hardware de Intel, las optimizaciones en la Jetson no suponen ninguna mejora, al contrario que en los equipos x86 con una CPU **Intel i7**. Por otro lado, es sorprendente que también existan diferencias significativas respecto a la Jetson **cuando se realizan las inferencias en GPU**. La diferencia más notoria es con YuNet en el PC del laboratorio, que poseé una **GeForce GTX 1650**, que está **una generación por detrás** de la GPU de la Jetson Orin (las arquitecturas son Turing y Ampere respectivamente) [17]. Esto implica que, para la carga de trabajo de este proyecto, **la GPU no es el factor predominante** (TODO: con estos datos puedo sacar esta conclusión?).

Capítulo 5

Reconocimiento facial adaptativo

En este capítulo se ahonda en los fundamentos y detalles de implementación de la funcionalidad *Open-World* y demás componentes para otorgar la adaptación.

5.1 Introducción

El sistema visto en el anterior capítulo (capítulo 4) está limitado al reconocimiento de personas ya registradas de antemano, lo que se conoce como ***Closed-Set***.

5.1.1 Fundamentos

El principal objetivo de OSDe-SVM es lograr la capacidad de adaptación con independencia del contexto en el que se utilice. Dicho método ha sido probado en un contexto de video vigilancia bajo el dataset **FACE COX** [18]. También se ha utilizado el dataset **Youtube Faces** (YTF) [19] que, a diferencia del anterior, se encuentra accesible de forma pública. El método utiliza **secuencias de video** como entrada para devolver las predicciones acerca de un IoI o para determinar una identidad desconocida.

Comités de SVM

Las características extraídas por ArcFace se utilizan como datos para entrenar las SVM para la clasificación. OSDe-SVM se basa en el concepto de que múltiples SVM simples (ejemplo: lineales) como conjunto (**comité**) **generalizan mejor** que una única SVM compleja (ejemplo: sigmoide) [20]. Otras ventajas de los comités son la incorporación de nueva información sin tener que re-entrenar las SVM, lo que recorta una cantidad de tiempo significativa durante la operación del sistema, y la flexibilidad, que permite eliminar parte de la información que no es relevante borrando la SVM redundante. Para cada IoI se asigna un comité de SVM, cada SVM del comité se entrena con n muestras de la propia entidad (1 en la idea original [20], 5

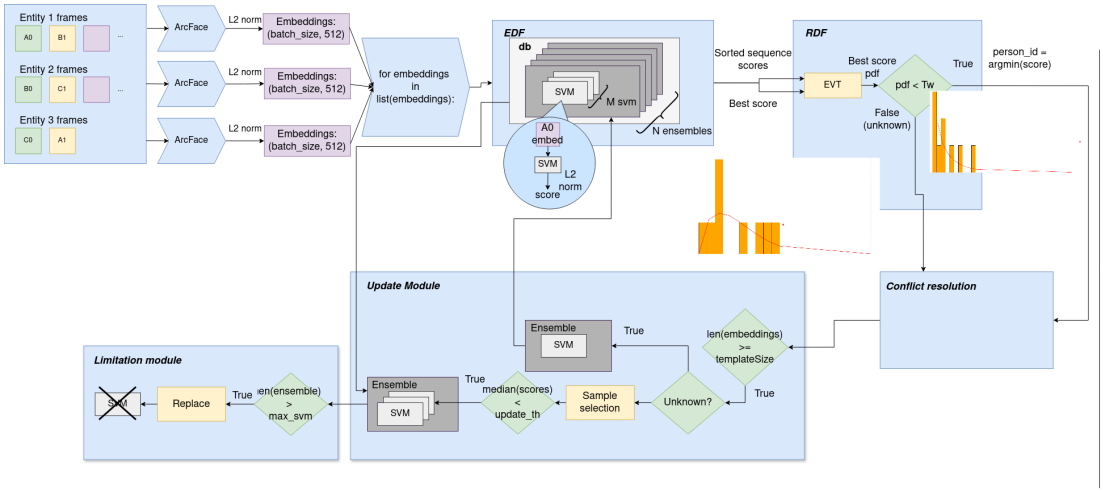

Figura 5.1: a

adaptado a este proyecto) y **m-n** muestras negativas (fotogramas de otras entidades), siendo **m** el número de IoI registrados en el sistema.

5.1.2 Arquitectura del sistema adaptativo

El sistema se estructura como muestra la figura 5.1.

5.2 Implementación

A partir de los mecanismos expuestos en la sección 4.2.1 se extraen las secuencias de frames que servirán como entrada del pipeline visto en la sección 5.1.2 y que se expone en detalle a continuación. La mayoría de los componentes se han implementado basándose en [2] TODO: como menciono también a César?.

5.2.1 Inicialización

La primera fase del sistema y sin lugar a dudas la más crítica. En este proyecto se han probado 2 aproximaciones:

- **Supervisado**: el operador realiza una selección de los 5-10 frames más representativos para cada individuo.
- **No supervisado**: el sistema recoge los frames cuando aparecen un mínimo de 5 entidades simultaneas en escena (el mínimo necesario de puntos para poder aplicar Weibull). A partir de los frames de las cámaras, que estarán sincronizadas, se obtienen las bboxes (TODO: glosario) que encierran los rostros y se procede a la creación de los comités con

los recortes de caras generados. **No se requiere de ninguna intervención del operador**, el sistema realiza la selección en base a unos criterios preestablecidos (ejemplo: la cara debe estar completamente dentro del rango de la cámara). En esta aproximación, es crucial el método de seguimiento (tracking) para recolectar las secuencias de caras.

Independientemente de la aproximación escogida, se obtienen las características de cada frame de la secuencia y se crea la SVM inicial con la que se **inicializa el comité**. Dicha SVM se entrena con las características del usuario, que compondrá el set positivo, y un subconjunto de las características del resto de usuarios, que compondrá el set de negativos (escogidas aleatoriamente). Dichas características son generadas por el modelo ArcFace [11], es la misma red que se utiliza en [1, 2] y que ha demostrado ser de las mejores en el SOTA (TODO: acrónimo) del reconocimiento facial.

La SVM inicial es la que define la identidad del comité. Si dicha SVM está compuesta por recortes de caras de baja calidad (ejemplo: borrosas o parcialmente ocluidas), entonces el comité **no estará bien definido** y, por lo tanto, generará **mayor confusión** a la hora de aplicar Weibull para los reconocimientos.

El resto de módulos del sistema que se exponen a continuación se ejecutan por cada secuencia de caras recolectada en cada secuencia de frames (figura 5.1).

5.2.2 *Ensemble Decision Function* (EDF)

Una vez inicializado los comités, el método determina identidades en base a secuencias de entrada. Por cada secuencia de entrada, siendo esta una secuencia de características extraídas de los recortes faciales, se calculan las **puntuaciones (scores) para cada comité**. La puntuación de un comité es a su vez un valor consensuado entre los resultados de las predicciones de las SVM que lo conforman, el criterio de consenso (o de fusión) se basa en un percentil (generalmente la media). Aplicar percentiles es igual a escoger un conjunto mas grande o pequeño de SVM, ya que pueden existir SVM dañinas para el comité (ejemplo: corresponden a otra persona). De esta forma, el percentil logra tolerar dichas puntuaciones dañinas. Para cada comité, se ejecutan las siguientes funciones:

- **FDF** (Frame Decision Function): se encarga de **fusionar las salidas** (o puntuaciones) de todas las SVM del comité cuando se analiza **un frame** de la nueva secuencia.

- **SDF** (Sequence Decision Function): se fusionan todas las puntuaciones de la función FDF para obtener un único resultado que representa a la **secuencia**.

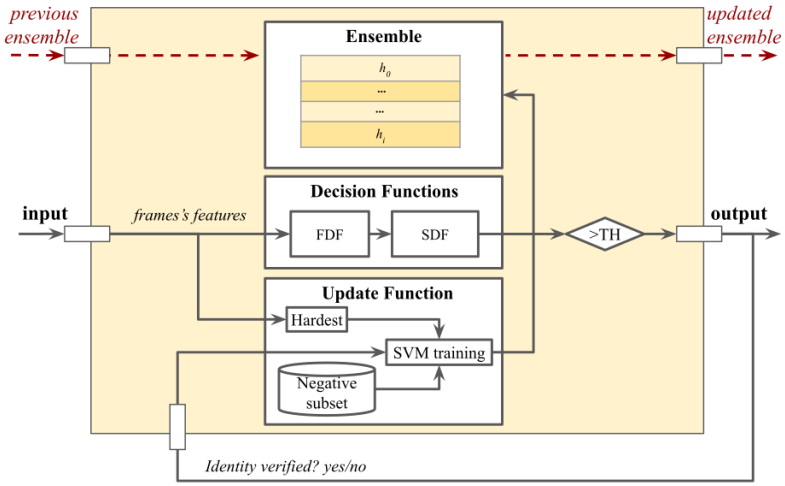

Figura 5.2: Pipeline del método De-SVM, figura extraída de [2]

5.2.3 *Recognition Decision Function* (RDF)

Esta función determina si la entidad detectada se corresponde a un individuo previamente reclutado o a una entidad **desconocida**.

Las SVM sólo pueden discernir dentro del conjunto de datos con las que fueron entrenadas (*Closed-Set*), por lo que un dato desconocido se clasificaría erróneamente como una de las clases del entrenamiento [9]. Varias investigaciones [9, 21] TODO: (y en [2]) utilizan el ***Extreme Value Theory*** (EVT o teorema de Fisher–Tippett–Gnedenko) para identificar clases no reclutadas. El *Extreme Value Theory* determina que el conjunto de máximos o mínimos de una muestra sigue una distribución de **Weibull**. El EVT otorga un conocimiento robusto, ya que convierte scores concretos de un algoritmo (en este caso las SVM) en probabilidades que siguen una teoría estadística. Esto permite la **fusión de datos de diferentes fuentes** como nuevas redes de reconocimiento o nuevas cámaras diferentes a las Kinect en el sistema.

La distribución de Weibull se modela a partir de las puntuaciones devueltas por cada comité (salida de la función EDF) **excepto la mejor puntuación** de todos los *ensembles* (que se corresponde con la más baja). Debido a que la entidad en cuestión sólo puede coincidir con un comité, se comprueba si el mejor resultado (o lo que se supone que es la entidad de la secuencia) **es un extremo** respecto de la distribución de puntuaciones no coincidentes (resto de *ensembles*), en caso afirmativo, se asigna la etiqueta del usuario del comité con la mejor puntuación, en caso contrario, el individuo se considera **desconocido** (no coincide con ningún comité). Para determinar si la puntuación es un extremo, se calcula la probabilidad (PDF) (TODO: glosario) del mejor *score* a partir de la función PDF con los parámetros obtenidos de la distribución de Weibull modelada. Si el PDF se corresponde con un valor muy cercano al

Algorithm 2 Recognition Decision Function (RDF) based on EVT.

S is the input sequence, T_W is the threshold in the Weibull function
$E = \{e^0, e^1, \ldots, e^{N-1}\}$ set of ensembles associated to *known* identities
$R = \{\varnothing\}$ set of scores given by each ensemble to a candidate
for e^i in E **do**
    $R \leftarrow SSF(e^i, S)$
end for
$c = \min(R)$; $m = median(R \setminus c)$
$V = \{\|x - m\| \mid x \in (R \setminus c) \wedge (x < m)\}$
Fit V to a Weibull function, W
if $W(\|c - m\|) < T_W$ **then**
    ID $= arg(c)$
else
    ID $=$*unknown*
end if

Figura 5.3: Algoritmo de reconocimiento basado en EVT, extraido de [2]

0, la puntuación se considera que está **al extremo de la distribución** [21]. La decisión de si una puntuación es un extremo o no se toma en base a un threshold (Tw), para el resultado de la función de Weibull (función PDF).

TODO: El algoritmo compone una función de Weibull ajustando los parámetros de la misma para que converga con la cola de la distribución. Dicha distribución está formada por las distancias de cada puntuación con la mediana, excluyendo la puntuación más baja (siendo teóricamente la puntuación que coincide para la entidad) y las puntuaciones por encima de la mediana. Finalmente, se determina la probabilidad de pertenencia de la puntuación más baja de la distribución, si la probabilidad es lo suficientemente baja como para concluir que no pertenece a la distribución, entonces el reconocimiento es correcto y se asigna la entidad. En caso contrario, se concluye que la entidad es desconocida. Se establece un threshold (Tw, como se muestra en el algoritmo 5.3).

(TODO: explicación?) El Extreme Value Theory determina que el valor **máximo** de una muestra sólo puede converger en una de las siguientes tres distribuciones: Fréchet, Gumbel o Weibull. Cada una de estas distribuciones poseen propiedades únicas (ejemplo: Fréchet es una distribución cuya cola decrece en menor medida que la de Gumbel) que las hacen adecuadas para determinados campos de estudio (TODO: ejemplo: Fréchet para predicción de inundaciones). TODO: En el campo del reconocimiento de personas, sería interesante hallar el mínimo por el que se puede determinar si cierto individuo pertenece o no a la distribución. Para este proyecto, el mínimo es la **mínima distancia entre el punto y el hiperplano creado por la**

SVM, como se aplican varias SVM, el valor final será una fusión de distancias del conjunto de SVM. Como la muestra sólo puede pertenecer a una entidad, entonces se deberían de obtener resultados no coincidentes (TODO: distancias negativas que fueron normalizadas usando no se que método) con el resto de individuos registrados. La distribución de los valores mínimos, en este caso los resultados no coincidentes, sigue una de las 3 distribuciones ya comentadas (TODO: es la de Weibull porque el comportamiento de la cola es acotado, es decir, hay un cierto valor x que devuelve 0).

En **??**, se muestra el algoritmo RDF.

(Pág 81, TODO: comentar en resultados) La calidad de la función RDF viene determinada por el número de puntos que componen la función de Weibull, cuantos más puntos, mayor definición y por tanto reconocimientos más precisos. Con un universo de 20 individuos, sólo 5 puntos componen la función de Weibull (de las 20 entidades sólo 10 se registran en el sistema, a modo de probar el open-set, y de esos 10, la mitad definen la función), lo que lleva a un comportamiento más impreciso que con un universo mayor. El sistema inicial cuenta con un universo de **6 individuos**, lo que claramente **es insuficiente** para realizar una diferenciación precisa de los IoI respecto a lo desconocido. Para este proyecto, ha sido necesario ampliar el universo a **20 personas**, algunas procedentes del CITIC, que se han prestado voluntariamente, otras extraídas del YoutubeFaces dataset.

5.2.4 Update Module

Es la función que implementa la actualización de los comités. Una vez se haya reconocido la entidad a través de la función anterior, pueden darse 2 posibles escenarios:

- **Entidad conocida (c<Tw)**: se crea una nueva SVM con los frames escogidos en el comité correspondiente. La SVM se entrena con los embeddings de la secuencia de entrada como conjunto de positivos, y un muestreo aleatoria de n embeddings del resto de comités.
- **Entidad desconocida (c>Tw)**: se crea un nuevo comité representando a la entidad con una nueva SVM. El conjunto de positivos y negativos es el mismo que en el caso anterior, salvo que ahora ya no es necesario extraer los embeddings del comité con el que coincide la secuencia del conjunto de negativos.

Una condición necesaria para que este módulo se ejecute es que la secuencia de entrada para el individuo **contenga un mínimo de n frames para su inicialización**. Otra precondición, que aplica a las entidades conocidas, es comprobar si las caras a añadir son lo suficientemente representativas. Las puntuaciones cerca del cero indican que las muestras se encuentran en el borde de lo que es nuevo y lo que la SVM ya conoce. A partir de un valor de umbral (*update_th*) se decide si dichas muestras añaden información nueva al comité.

5.2.5 Limitation Module

El limitation module es una función que se encuentra inherente al módulo de actualización. Si un comité excede un número prefijado de SVM almacenadas (10 en este caso), se toma una decisión para eliminar una de las SVM según los siguientes criterios:

Diversidad

Este criterio mide el valor de aportación de una SVM respecto al resto del comité. Se coge un conjunto aleatorio de n embeddings (siendo n=50) de entre todos los comités y se generan los scores utilizando las m SVM del comité, lo que resulta en n*m scores. TODO: **Basándose en el signo de los scores**, se acumula el producto de los signos entre scores, de forma que un valor discordante afecta en mayor medida a la propia SVM y en menor medida al resto de SVM. Un valor alto indica bajo nivel de diversidad, por ende un valor **pobre de aportación al comité**. El valor de diversidad se calcula como en [2]: TODO: poner la fórmula

Coherencia

Este criterio viene a determinar la precisión de una SVM en el reconocimiento cuando no se dispone de un dataset para su evaluación (ejemplo: entorno operacional). El valor de coherencia determina cuantas veces una SVM devuelve el mismo signo que el resultado consensuado del comité, si los signos coinciden, se suma 1 al valor de coherencia, en caso contrario, se resta -1 a dicho valor. Un valor alto indica buena precisión. En la creación de una SVM, este valor se inicializa a 0.

Favorabilidad

Finalmente, los dos criterios se fusionan en un valor llamado **índice de favorabilidad**, la SVM con el menor valor de dicho índice **se elimina del comité**. El signo del valor de diversidad **se invierte** a la hora de la fusión.

5.2.6 Creación de nuevos comités (*Open-World*)

Cuando se reconoce a un usuario como desconocido (supuestamente un individuo no antes reclutado), se registra su identidad en forma de un nuevo comité con una SVM inicial. De esta forma, el sistema adquiere la capacidad de expandir su conocimiento a partir de personas nunca antes vistas. Dicha SVM inicial tiene de muestras positivas las de la propia secuencia actual y de muestras negativas las del resto de individuos ya conocidos.

5.2.7 Coherencia entre cámaras

TODO: no es mas conveniente ejecutarlo antes del update module?

Si se dispone de varias cámaras sincronizadas en el mismo instante de tiempo, se puede asumir que no es posible que aparezca una misma persona en más de una cámara (aunque es necesario atender al grado de solape que pueda existir entre las cámaras). Si más de una cámara da una misma predicción, puede ocurrir que una o varias predicciones sean incorrectas, o que todas las predicciones sean erróneas. Se ha desarrollado el siguiente pipeline para intentar resolver las predicciones incorrectas: TODO.

5.2.8 otra cosa

En resumen, todo el pipeline comentado **realiza x*j*y*z iteraciones**, siendo x el nº de personas, j el nº de frames por persona, y el nº de comités (o entidades registradas) y z el nº de SVM por comité **para cada secuencia de entrada**.

TODO: YuNet en OpenCV **no admite batching** (o quizá YuNet en si), varias opciones:

- (Más simple) secuencial: por cada frame que se recibe, ya se procesa por YuNet y se envia la lista cuando se llame a on_frame (sabemos que la inferencia va a tardar menos de 100 ms).

Capítulo 6

Pruebas y resultados

En este capítulo se presentan los resultados de rendimiento del método con el modelo y junto con el sistema final, variando el número de cámaras. También se explica el proceso de creación de los datasets utilizados para las pruebas.

6.1 Diseño de las pruebas

6.2 Resultados Open-World

TODO: En la fase de inicialización de los modelos (o fase de entrenamiento), el sistema guarda la información estadística que representa a cierto modelo. Durante el funcionamiento del sistema, puede ocurrir que los datos que recibe de un individuo registrado varíen respecto a la información de entrenamiento, lo que se conoce como **concept drift**. Se debe de corregir el concept drift para evitar una pérdida en la precisión de reconocimiento.

TODO: actualización del modelo, añadir nuevas entidades puede corromper la información de las entidades iniciales (catastrophic forgetting).

TODO: él método tiene una precondición, **se necesitan de al menos 5 IoI para crear la distribución de Weibull**.

TODO: los criterios que conforman el módulo de limitación tienen en cuenta el signo de la puntuación de forma exclusiva, lo que puede no funcionar siempre, sobretodo si se dispone de pocos frames que definan al individuo.

TODO: las secuencias de pocos frames (ej: 2) son muy propensas a resultados erroneos, especialmente si introducen mucho ruido, como frames borrosos, con mucha oclusión, variación en la iluminación, entre otros muchos factores. Debido a que este sistema maneja secuencias solapadas (ej: frames 0 a 10, frames 5 a 15) se puede deducir que es muy probable que la identidad a reconocer en la secuencia actual sea la misma identidad que en la secuencia anterior. Dependiendo de diversos factores, como el número de frames de la secuencia, se

puede otorgar un mayor peso al reconocimiento anterior.

Para concluir, cuando el sistema no es capaz de reconocer a un individuo con alta confianza y, en cambio, lo denota como desconocido, el mecanismo se vuelve **contraproducente**. Si una entidad ya conocida se reconoce como desconocida, se crea un nuevo comité de la misma persona, lo que incrementa la duda en el reconocimiento, ya que la distribución de weibull contendrá puntos que representen al propio individuo, indicando que el score de coincidencia no es un caso extremo cuando si debería serlo. La solución más rápida y efectiva es subir el valor de threshold de la probabilidad devuelta por la función de Weibull, aún así, es inevitable que se generen nuevos comités para un mismo individuo, contaminando el sistema rápidamente.

6.3 Fuentes de los datos (nuestro dataset combinado con YTF)

6.4 Pruebas del sistema final

6.5 Solapamiento de secuencias: buffering

Supongase un tamaño de secuencia 25 y un offset de 15, la secuencia de frames sería la siguiente:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24

15 16 17 18 ... 39

Se pueden almacenar parte de los frames utilizados en un buffer para la siguiente secuencia.

6.6 Número de IoI

El número de IoI (o de entidades positivas) afecta directamente al reconocimiento mediante Weibull, ya que cuantas más entidades, más puntos definen la distribución, por lo tanto mejor definida está la función. TODO: me da igual o mejores resultados con 6 entidades que con 10.

6.7 Número de negativos

En la tabla (TODO) se muestra el *F1 score* para 10 y 50 muestras que conforman el conjunto de negativos de entrenamiento (ratios 1:1 y 1:5 si el tamaño del template es 10). Claramente los resultados mejoran al escoger 50 negativos, esto se debe a que en 10 frames cabe un conjunto reducido de entidades a diferencia de 50 frames, sobretodo si se realiza una selección aleatoria como ocurre en este caso.

6.8 Umbral de Weibull (TW)

Si el valor del umbral de Weibull es bajo, la precisión se mantiene en valores altos, sacrificando el recall, mientras que un valor alto en dicho umbral se traduce en un elevado recall a costa de la precisión. Como ya se ha explicado, el umbral de Weibull representa la probabilidad máxima a la que el mejor score pertenece a la distribución de scores no coincidentes, cuanto más bajo sea el valor, más extrema debe ser la coincidencia para ser reconocida (la precisión aumenta), por lo que la función es más selectiva en el reconocimiento (devuelve un mayor número de desconocidos, reduciendo así el recall). Un valor alto representa justo lo contrario. TODO: parece que un TW alto es mejor siempre, pero tengo que poner los resultados de una prueba con un desconocido de verdad, así muestro el verdadero problema de elevarlo.

6.9 Percentiles

Al fusionar los scores de todas las SVM en todos los frames, se obtiene un único score acordado por la mayoría mediante la mediana, siendo esta equivalente al percentil 50. Al variar el percentil, el número de SVM participantes también varía. Un percentil de 100 implica obtener el mayor score, que se corresponde con la peor coincidencia (por ende, el valor de recall es paupérrimo, puesto que se necesita del consenso de todas las SVM del comité para devolver un match). Por otro lado, un percentil de 0 implica recuperar el score más bajo de entre todas las SVM en todos los frames. En este caso también podría darse un bajo recall ya que si por ejemplo se añade una SVM de otra entidad, dicha SVM devolvería un score negativo, por lo que Weibull etiquetaría el reconocimiento como desconocido al haber más de un comité activado (que devuelve un score bajo).

Uno de los objetivos del sistema es obtener la mayor diversidad intra-comité y la mayor especificidad inter-comité. Un **percentil alto** encajaría si las SVM son **específicas** entre si (la mayoría de las SVM devuelven match). En cambio, un **bajo percentil** funcionaría con unas SVM **más diversas** (se espera que la minoría de las SVM den un resultado coincidente).

6.10 Tamaño del comité

En relación con el anterior punto, el número de SVM influye en la decisión de que percentil tomar. Si se utilizan 3 SVM por comité y la mediana, al menos 2 SVM deben de devolver un score bajo para dar un resultado coincidente. Por otro lado, si se utilizan 10 SVM y un percentil de 30, sólo 3 de las 10 SVM necesitan estar de acuerdo para devolver un match.

En los resultados se muestra como un comité de 3 SVM y TW=0.05 destaca respecto a un comité de 1 SVM y TW=0.5. Los resultados utilizando 10 SVM por comité no llegan a mejorar

notablemente el valor de 3 SVM, esto es de gran relevancia, puesto que se reduce el número de iteraciones de cada reconocimiento de 10*n a 3*n, siendo n el número de comités que existen y asumiendo que todos los comités han llegado al límite de SVM. Con estos resultados se ha demostrado que implementar varias SVM por comité mejora el recall manteniendo la precisión. 1 sóla SVM por comité es efectiva si dicha SVM es representativa del individuo, en caso contrario puede devolver respuestas coincidentes acerca de una entidad distinta del comité, generando ruido que perjudica a las predicciones de Weibull. Un mayor número de SVM combinado con la mediana devuelve un resultado conforme dicta una mayoría que omite los scores de SVM intrusas (de entidades distintas) y permite expulsarlas mediante los criterios ya comentados en la sección 5.2.5.

6.11 Tamaño de la plantilla

El tamaño de plantilla es igual al tamaño del conjunto de positivos para una entidad, cuantas más muestras conformen dicho conjunto, mejores predicciones realizará la SVM asociada. En la tabla (TODO) se puede ver que el mejor resultado ha sido con un tamaño de plantilla de 10 frames. Los resultados con 5 frames son ligeramente peores (79.7 frente a 76.5), lo que demuestra la robustez del sistema ante dicho número escaso de muestras para un individuo.

6.12 Tamaño de secuencia

En la tabla (TODO) se muestran 3 valores diferentes de este parámetro. Como es de esperar, los mejores resultados coinciden con el mayor tamaño (25), puesto que se dispone de una mayor cantidad de información que apoye al reconocimiento. Los resultados con una secuencia de 15 frames caen levemente, un menor tamaño de secuencia hace que el sistema funcione a una mayor frecuencia (devuelva más reconocimientos en el mismo tiempo), concretamente a cada 1,5 segundos con dicho tamaño de secuencia asumiendo que las cámaras funcionan a 10 Hz (capturan una imagen cada 100 ms).

6.13 Solapamiento

El solapamiento entre secuencias puede causar que las SVM sean más específicas, ya que se da lugar a un mayor número de SVM parecidas entre sí (con un solapamiento de 5 y un tamaño de secuencia de 25, se comparten 20 frames entre secuencias y por ende entre las SVM). Con un número reducido o nulo de frames compartidos, el comité tiende a generalizar mejor, lo que es un factor crítico para la precisión y el recall.

Según los resultados arrojados (TODO), ninguno de los solapamientos utilizados indica una variación significativa en los resultados.

6.14 Inicialización del sistema

La selección de los frames que conforman un nuevo comité es un factor crítico para el correcto funcionamiento del sistema. En la tabla (TODO) se comparan 2 técnicas de inicialización, una semi-supervisada (los frames se escogen manualmente y el comité se actualiza de forma autónoma) y la otra no supervisada (se escogen los frames en el momento que aparecen varias personas en escena para diferenciarlas).

6.15 Criterio de selección de frames para nuevas SVM

También es muy importante la selección de las muestras que conforman las SVM, en la tabla (TODO) se muestran 2 criterios para la selección, uno consiste en seleccionar los frames con los scores más cercanos al cero, mientras que el otro es una selección aleatoria. El primer criterio demuestra un aumento en el F1 score, dado que los frames cercanos al cero son los más complicados de reconocer, de esta forma la SVM amplía sus fronteras respecto a lo que ya reconoce. A la hora de crear una SVM de un nuevo comité en el modo open-world, no queda otra alternativa que realizar una selección aleatoria.

6.16 Criterio de favorabilidad

En la tabla (TODO) se muestra el efecto de dicho criterio a la hora de eliminar las SVM respecto a un criterio de eliminación aleatorio. Este ...

6.17 Creación de nuevos comités

En el modo Open-World, el sistema crea un nuevo comité cuando se reconoce a una nueva entidad en el sistema. Sin embargo, debido a las predicciones incorrectas del método, puede ocurrir que se creen nuevos comités para un individuo ya registrado. Estos nuevos comités pueden desplazar a los comités originales adueñándose de su identidad.

El número de comités redundantes creados varía conforme al umbral de Weibull. Cuanto menor sea el valor, más comités redundantes se crearán debido a que la incertidumbre del método aumenta.

6.18 Open-set runtime

6.19 Open-world selected

6.19.1 Caso concreto:

global 99 73 14 57.6 87.6 69.5

num ensembles: 15 svm of each ensemble: [('jose', 1), ('carlos', 1), ('martin', 1), ('omi', 1), ('rayas', 1), ('andres', 1), ('botella', 1), ('azul', 1), ('javi', 1), ('gris', 1), ('user10', 1), ('user11', 1), ('user12', 1), ('user13', 1), ('user14', 1)]

6.20 Open-world runtime

Capítulo 7

Protección de templates

En este capítulo se detallan las posibles opciones para la protección de templates.

7.1 Cifrado

Los algoritmos de cifrado actuales (AES, RSA, entre otros) cumplen la propiedad avalancha, que implica que un ligero cambio en el input (ej: 1 bit) genera una salida radicalmente distinta a la esperada. Para lograr esta propiedad, los algoritmos destruyen cualquier propiedad estadística y de similitud entre los vectores de características, por lo que no es posible realizar cálculos de distancias / similitudes en el dominio cifrado.

Un factor importante que pone en riesgo la seguridad de este método es el almacenamiento seguro de las claves generadas.

7.2 Hash

Son funciones que devuelven un tamaño fijo de los datos independientemente del tamaño de la entrada. Los hashes son computacionalmente eficientes y no son reversibles. Además, no es necesario almacenar ninguna clave.

De todos modos, los algoritmos de hash deben cumplir necesariamente la propiedad de la avalancha, que afecta irremediablemente a la comparación de los vectores de características, que son influenciados por el ruido de las imágenes [22].

7.3 Bloom filters

Del campo de protección de plantillas biométricas (BTP) surge el término **cancellable biometrics**, que son plantillas que cumplen con las siguientes propiedades [23]:

- No invertible o irreversible: dado un template protegido, el atacante no puede obtener computacionalmente el template original, incluso aunque la clave para su securizado sea conocida.
- Revocabilidad: debe de ser capaz de invalidar el template comprometido y generar uno nuevo con solo cambiar unos parámetros (ej: clave).
- Desvinculación: dados múltiples templates extraídos de distintas aplicaciones. Debe ser imposible averiguar si dichos templates corresponden a un mismo individuo.

Los principales métodos conocidos para generar cancellable biometrics son **BioHashing, bloom filter, and Index-of-Max (IoM)**. Artículos posteriores a la concepción de estos 3 métodos han logrado vulnerar el BioHashing y el IoM [23], se ha descubierto que los bloom filters no satisfacen la propiedad de la desvinculación (vulnerabilidad a ataques cross-matching), aunque se ha conseguido resolver realizando un par de permutaciones a la información de los vectores de características [24].

- https://openaccess.thecvf.com/content/WACV2021W/XAI4B/papers/Wang_Interpretable_Security_Analysis_of_Cancellable_Biometrics_Using_Constrained-Optimized_Similarity-Based_Attack_WACVW_2021_paper.pdf

Artículos

- **Helper Data Scheme for 2D Cancelable Face Recognition using Bloom Filters**
  - Requiere de guardar un PIN
  - No realiza pruebas de seguridad. **Fuera**
  - Utiliza LBLDA para feature extraction
  - Impacto en el reconocimiento de mínimo 0.1 (ej: de 0.9 decae a 0.8).
- **Protected Facial Biometric Templates Based on Local Gabor Patterns and Adaptive Bloom Filters (2014).**
  - **Fully-reproducible**
  - Utiliza LGBPHS para la extracción de features.
  - Codifica y binariza las features: coger la matriz de salida y representar con 1's los valores que no sean 0 (se pierde mucha información).
  - **Únicamente aplicable al reconocimiento facial**.
  - Afirma que los bloom filters pueden usarse para múltiples rasgos biométricos.

- **Unlinkable and irreversible biometric template protection based on bloom filters** (2016)
  - Continuación del anterior artículo
  - **Reproducible**
  - Bloom filters vulnerables a **cross-matching attacks** (si se utiliza el mismo dato biométrico para varias aplicaciones). El artículo propone un cambio en el sistema para resolverlo.
  - **Únicamente testado con caras**
- **Multi-biometric template protection based on bloom filters** (2018)
  - Continuación del anterior artículo
  - **Complejo de implementar**, necesario realizar una estimación de parámetros para crear los Bloom filters (proponen un framework para ello).

No sé a qué se refiere con honey templates: https://ietresearch.onlinelibrary.wiley.com/doi/pdfdirect/10.1049/iet-bmt.2015.0111

[25]

Inconveniente: requiere que el vector de características esté en formato entero o binario, lo que compromete la precisión.

7.4 Homomorphic encryption

7.5 Random projection

A cancelable biometric authentication system based on feature-adaptive random projection (2021-05)

- Utiliza Random Projection (también conocido como Biometric salting), que projecta los vectores transformados en un espacio random.
- La clave **es pública**
- El artículo propone una nueva medida de seguridad para solucionar las vulnerabilidades de Random Projection (ataque ARM, utiliza una matriz y una clave adicionales que son posteriormente desechadas).
- Aplicable a cualquier tipo de dato biométrico (deben estar en formato binario). ¿La imagen debe estar en escala de grises? **Únicamente probado con huellas dactilares**

- Compara las features cifradas con 2-norm.
- Cambiar el descriptor MP (afecta negativamente al poder discriminativo del modelo) por MCC.
- **Investigar si se ha vulnerado este sistema**.

7.6 Random Distance Method

Random Distance Method for Generating Unimodal and Multimodal Cancelable Biometric Features (2019-03)

- Log-Garbor para feature extraction + RDM (propuesto en este artículo) para la transformación del template
- RDM: mapea puntos del vector al espacio cartesiano y se calcula la distancia respecto a unos puntos random generados a partir de una clave del usuario. Finalmente aplica **median filtering**
- **Utiliza una clave fijada por el usuario**. La clave **puede ser pública**, puesto que no es suficiente para deshacer el vector original.
- **Fácil de implementar**.
- Necesita las features binarizadas?
- De momento no se conocen vulnerabilidades.
- Feature type: Unimodal and multimodal fusion (no sé qué es eso).
- **Requiere pre-aligment de features**

7.6.1 Procedimiento

fv = feature vector fs = fv salted

- fv se multiplica por una constante (ej: c = 100)
- Se obtiene fs = fv + RG, siendo RG un vector de las mismas dimensiones generado con valores de enteros aleatorios en el rango [1, 255]
- fs se divide en fX = fs(1 : N'/2) y en fY = fs(N'/2 + 1 : N')
- Se genera una clave K para el usuario de dimensiones 1xN', con valores aleatorios comprendidos en el rango [-100, 100]. La clave también se divide en 2 mitades K0 y K1

- Para todo 1 <= j <= N'/2, calcular la distancia dj entre los puntos FPj y RPj, siendo FPj = (xj = fX(j), yj = fY(j)) y RPj = (xj = K0(j), yj = K1(j))
- Se guardan todas las distancias (euclidianas) en un vector D de tamaño N'/2 y se aplica **median filtering** (p = 5 neighborhood?).
- Los valores de K deben estar en el mismo rango que los de fv?

7.7 Correspondencia en el espacio vectorial

Lp-norm es una métrica para medir distancias entre vectores, donde "p" es un entero positivo:

$$\|x\|_p := \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p} \tag{7.1}$$

- p=1: normalización de Manhattan
- p=2: normalización Euclidiana (es igual al teorema de pitágoras).

Para Random Distance Method pueden usarse ambos L1 y L2 para el cálculo de distancias entre vectores.

Capítulo 8

Conclusiones y trabajo futuro

8.1 Conclusiones

Cuando tenga los valores concretos de los resultados.

8.2 Trabajo Futuro

El sistema Open-World no ha tenido el comportamiento que se esperaba. Se ha observado que dicho sistema es **muy sensible** a la hora de ajustar el parámetro del umbral de Weibull. Ajustando el umbral a un valor bajo, el número de comités nuevos por entidad se dispara, ya que añade incertidumbre al sistema rápidamente (puesto que no se generan casos extremos si la entidad en cuestión se encuentra en la distribución de non-match scores). Un valor alto del umbral mitiga este problema, a costa de comprometer la precisión. Se podría estudiar la implementación de un nuevo criterio para agrupar los comitésy así solventar esta problemática

Implementar el modo secuencia y la adaptación en el sistema final, calcular distancia euclidiana usando posición 3d en vez de 2d.

Apéndice A

Apéndices

Bibliografía

[1] A. Pérez Pena, "Detección e identificación de persoas cun robot móbil equipado cun lidar 3d e cámaras rgbd," Master's thesis, Universidade de A Coruña, 2024, 30 de noviembre de 2025. [Online]. Available: http://hdl.handle.net/2183/39806

[2] E. López-López, "Incremental learning through unsupervised adaptation in video face recognition," Ph.D. dissertation, Universidade de A Coruña, 2021, 30 de noviembre de 2025. [Online]. Available: http://hdl.handle.net/2183/29498

[3] S. Mittal, "A survey on optimized implementation of deep learning models on the nvidia jetson platform," *Journal of Systems Architecture*, vol. 97, pp. 428–442, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1383762118306404

[4] F. Oliveira, D. G. Costa, F. Assis, and I. Silva, "Internet of intelligent things: A convergence of embedded systems, edge computing and machine learning," *Internet of Things*, vol. 26, p. 101153, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2542660524000945

[5] W. Rahmaniar and A. Hernawan, "Real-time human detection using deep learning on embedded platforms: A review," *Journal of Robotics and Control (JRC)*, vol. 2, no. 6, pp. 462–468, 2021.

[6] S. A. I. L. et al, "Robotic operating system," *N/A*, 2018, 30 de noviembre de 2025. [Online]. Available: https://www.ros.org

[7] I. Condés, J. Fernández-Conde, E. Perdices, and J. M. Cañas, "Robust person identification and following in a mobile robot based on deep learning and optical tracking," *Electronics*, vol. 12, no. 21, 2023. [Online]. Available: https://www.mdpi.com/2079-9292/12/21/4424

[8] C. B. Murthy, M. F. Hashmi, N. D. Bokde, and Z. W. Geem, "Investigations of object detection in images/videos using various deep learning techniques and embedded platforms—a comprehensive review," *Applied sciences*, vol. 10, no. 9, p. 3280, 2020.

[9] E. M. Rudd, L. P. Jain, W. J. Scheirer, and T. E. Boult, "The extreme value machine," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 762–768, 2017.

[10] GeeksForGeeks, "What are embedding in machine learning?" *N/A*, 2025, 30 de noviembre de 2025. [Online]. Available: https://www.geeksforgeeks.org/what-are-embeddings-in-machine-learning/

[11] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4690–4699.

[12] Wikipedia, "Support vector machine," *N/A*, 2025, 30 de noviembre de 2025. [Online]. Available: https://en.wikipedia.org/wiki/Support_vector_machine#Linear_SVM

[13] B. Sahbani and W. Adiprawita, "Kalman filter and iterative-hungarian algorithm implementation for low complexity point tracking as part of fast multiple object tracking system," in *2016 6th international conference on system engineering and technology (ICSET)*. IEEE, 2016, pp. 109–115.

[14] ROS, "Upcoming ros 1 end of life," *N/A*, 2025. [Online]. Available: https://www.ros.org/blog/noetic-eol/

[15] Intel, "Openvino™ toolkit: An open source ai toolkit that makes it easier to write once, deploy anywhere." *N/A*, 2025, 30 de noviembre de 2025. [Online]. Available: https://www.intel.com/content/www/us/en/developer/tools/openvino-toolkit/overview.html

[16] pyimagesearch, "How to use opencv's "dnn" module with nvidia gpus, cuda, and cudnn," *N/A*, 2020, 30 de noviembre de 2025. [Online]. Available: https://pyimagesearch.com/2020/02/03/how-to-use-opencvs-dnn-module-with-nvidia-gpus-cuda-and-cudnn/?utm_source=chatgpt.com

[17] NVIDIA, "Cuda gpu compute capability," *N/A*, 2025, 30 de noviembre de 2025. [Online]. Available: https://developer.nvidia.com/cuda-gpus

[18] Z. Huang, S. Shan, R. Wang, H. Zhang, S. Lao, A. Kuerban, and X. Chen, "A benchmark and comparative study of video-based face recognition on cox face database," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5967–5981, 2015.

[19] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *CVPR 2011*. IEEE, 2011, pp. 529–534.

[20] T. Malisiewicz, A. Gupta, and A. A. Efros, "Ensemble of exemplar-svms for object detection and beyond," in *2011 International conference on computer vision*. IEEE, 2011, pp. 89–96.

[21] R. M. Walter Scheirer, Anderson Rocha and T. Boult, "Robust fusion: Extreme value theory for recognition score normalization," *European Conference on Computer Vision (ECCV), pages 481–495*, 2010.

[22] C. Busch, *Privacy Protection*, 2020.

[23] H. Wang, X. Dong, Z. Jin, A. B. J. Teoh, and M. Tistarelli, "Interpretable security analysis of cancellable biometrics using constrained-optimized similarity-based attack," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 70–77.

[24] M. Gomez-Barrero, C. Rathgeb, J. Galbally, C. Busch, and J. Fierrez, "Unlinkable and irreversible biometric template protection based on bloom filters," *Information Sciences*, vol. 370-371, pp. 18–32, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025516304753

[25] C. Rathgeb, M. Gomez-Barrero, C. Busch, J. Galbally, and J. Fierrez, "Towards cancelable multi-biometrics based on bloom filters: a case study on feature level fusion of face and iris," in *3rd international workshop on biometrics and forensics (IWBF 2015)*. IEEE, 2015, pp. 1–6.