Casa abierta al tiempo
UNIVERSIDAD AUTÓNOMA METROPOLITANA
Unidad Iztapalapa

DIVISIÓN DE CIENCIAS BÁSICAS E INGENIERÍA

DOCTORADO EN CIENCIAS Y TECNOLOGÍAS DE LA INFORMACIÓN

Determinación del perfil antropométrico en secuencias de video

Alumno:
M. en C. Contreras Murillo Miguel

Asesores:

Dr. De los Cobos Silva Sergio Gerardo
Dr. Lara Velázquez Pedro

Mayo de 2021

Casa abierta al tiempo
UNIVERSIDAD AUTÓNOMA METROPOLITANA
Unidad Iztapalapa

DIVISIÓN DE CIENCIAS BÁSICAS E INGENIERÍA

DOCTORADO EN CIENCIAS Y TECNOLOGÍAS DE LA INFORMACIÓN

Determinación del perfil antropométrico en secuencias de video

Alumno:
M. en C. Contreras Murillo Miguel

Asesores:

Dr. De los Cobos Silva Sergio Gerardo

Dr. Lara Velázquez Pedro

Mayo de 2021

Resumen

La obtención del perfil antropométrico en secuencias de video es un problema abierto que representa un gran desafío en la visión por computadora. Esta información es útil para fines estadísticos o hasta en el reconocimiento de personas en videos de vigilancia; ya que estas características suelen utilizarse como descriptores semánticos. Hasta ahora, el mejor rendimiento se puede lograr mediante el uso de cámaras 3D, pero este enfoque requiere de hardware especial. Otros enfoques 2D logran buenos resultados en situaciones normales, pero fallan cuando la persona usa ropa suelta, lleva bolsas o el ángulo de la imagen cambia ya que se basan en el cálculo de bordes, siluetas, o la energía de la persona en la imagen.

Este trabajo tiene como objetivo proporcionar una nueva metodología de obtención de características humanas basada en la creación de un esqueleto virtual para cada individuo a partir de imágenes y vídeo en 2D, luego se miden las distancias entre algunos puntos del esqueleto, y funcionan como entrada de un clasificador que determina su género. Esto mejora los resultados en comparación con otros algoritmos para el mismo fin, ya que la ropa, las bolsas y el ángulo de la imagen afectan poco al proceso de esqueletización.

Dedicatoria

Con dedicatoria a mi familia y amigos.

Agradecimientos

Se agradece al Consejo Nacional de Ciencias y Tecnologías por las becas otorgadas para mis estudios de posgrado.

A mis asesores de tesis, Dr. De los Cobos Silva Sergio Gerardo y Dr. Lara Velázquez Pedro, y al Dr. Eric Alfredo Rincón García por su apoyo y consejos en la realización de esta tesis, y publicación de un artículo junto al Dr. Román Anselmo Mora Gutiérrez y al Dr. Miguel Ángel Gutiérrez Andrade.

Al coordinador del programa Dr. Enrique Rodrí-guez de la Colina, y al personal de la división de Ciencias Básicas e ingeniería y de la Universidad Autónoma Metropolitana-Iztapalapa, por la amable atención que me ha sido brindada.

No olvidando el muy importante apoyo de mi familia y amigos, a quienes también agradezco.

Índice

Índice de Figuras

Índice de Tablas

Índice de Fórmulas

Introducción

Una de las primeras implementaciones de la biometría fue el sistema Bertillonage para la identificación de personas, propuesto por un oficial de policía francés de nombre Alphonse Bertillon en 1879. Este método se basó en la medición de diferentes partes del cuerpo y la descripción de marcas únicas de cada persona[1], [2]. Esta estrategia aún se utiliza como apoyo para localizar a las personas, ya que es la forma en que las personas pueden describirse semánticamente.

Desde entonces, el uso de biometría dura y suave para identificar a las personas se ha incrementado a medida que se mejora la tecnología necesaria para su automatización. La biometría dura incluye características que son únicas en cada individuo, y pueden ser físicas (huellas dactilares, iris, cara, etc.) o conductuales (voz, movimiento, etc.); mientras que la biometría suave es aquella que donde varios individuos pueden compartir las mismas características, y puede utilizarse para describir a una persona semánticamente (género, altura, peso, etc.).

Hoy en día, la mayoría de los sistemas automáticos de reconocimiento de personas se basan en la biometría física dura, ya que la eficacia de los algoritmos mejora a medida que aumenta la diferencia entre las muestras. Sin embargo, requieren una interacción muy cercana o incluso directa con hardware especial (escáneres de huellas digitales, cámaras 3D, etc.). Además, estos algoritmos suelen comparar las muestras obtenidas con cada una de las almacenadas en una base de datos. Por lo tanto, esta estrategia puede tener un impacto negativo en la eficiencia y la precisión de la clasificación cuando se aplica en poblaciones muy grandes, ya que el número de comparaciones requeridas aumenta con cada individuo adicional.

Por otro lado, la biometría suave es muy útil para describir a las personas verbalmente; no es invasiva, ya que no se requiere ninguna interacción cercana con el hardware; y pueden agrupar fácilmente a las personas que comparten las mismas características. Sin embargo, la biometría blanda puede ser similar en diferentes individuos y su uso para identificar personas se limita a poblaciones pequeñas[3], [4].

Aunque la biometría dura es más precisa que la suave, no siempre es posible tener una colaboración directa de la persona que está siendo buscada o el hardware puede no estar lo suficientemente cerca como para obtener buenos resultados. Para estos casos, la mejor opción parece ser una combinación de descriptores duros y suaves para la identificación de personas a través de técnicas no invasivas, como métodos basados en sistemas de cámaras de vigilancia.

Algunos de los métodos no invasivos más eficaces para la detección de género se basan en cuatro enfoques principales de identificación por la forma de caminar [5]. El primero compara el contorno de la persona con muestras etiquetadas y lo asigna a la clase más similar. El segundo enfoque utiliza una estrategia similar, sin embargo, en esta metodología se utiliza toda la silueta, y no sólo el contorno; por lo tanto, los errores de segmentación se producen con menos frecuencia. El tercero se basa en la energía de la caminata, que es la silueta media de todos los cuadros en un ciclo de caminata, esta silueta de energía se compara con las de todas las muestras almacenadas de personas para el entrenamiento del algoritmo; por lo tanto, estos métodos son ineficaces si la silueta en la muestras no coincide con la figura esperada; además, la cámara siempre debe capturar el mismo ángulo de la persona [6]–[8]. El último enfoque para la identificación de personas por su forma de caminar utiliza imágenes RGB-D. Esta tecnología proporciona información detallada de la imagen y simplifica su segmentación, logrando una visión más clara del objeto o persona que se está grabando. Este último método no sólo es útil en la identificación de género, o de personas por su forma de caminar, sino también en otras aplicaciones como el reconocimiento de actividades [8]–[10] y de representación[11].

La detección de sexo es especialmente útil para encontrar personas perdidas o buscadas cuando sólo se cuenta con una descripción semántica o no existe suficiente información para aplicar otros métodos de búsqueda automática, y el proceso se realiza manualmente, ya que ayuda a reducir el espacio de búsqueda al proporcionar únicamente las muestras que coinciden con las características buscadas.

En el aprendizaje supervisado, la computadora aprende a identificar diferentes clases a partir de algunas entradas. Ya que la intervención humana durante el proceso de entrenamiento es casi nula, el algoritmo selecciona autónomamente algunas características para ser identificadas. Otros clasificadores de sexo toman como entrada la imagen completa o la silueta de la persona, y como resultado se obtiene el sexo de la persona en una única etapa. Esto provoca que la máquina aprenda a identificar las figuras más comunes en la imagen de entrada, pero a pesar de que sea muy común la aparición de algunas figuras en personas de un sexo en específico, podrían ni siquiera estar relacionadas con las expresiones de género del individuo (longitud de cabello, vestimenta, etc.). Es por esto que la mayoría de los algoritmos fallan al clasificar el sexo cuando la vestimenta de la persona es muy holgada, ya que aprenden a identificar la vestimenta. Usar la información del esqueleto en dos etapas ayuda a controlar cuál es la información que el algoritmo aprende a identificar, de esta forma se evita que el modelo generado se base en la forma de vestir de cada individuo.

En esta investigación se propone un nuevo enfoque de dos etapas para la detección de sexo basado en la información de un esqueleto virtual [12]–[16]. Gracias al uso de la información del esqueleto virtual, este algoritmo puede obtener buenos resultados incluso cuando una

persona lleva equipaje o viste ropa holgada, ya que este método no se basa en el análisis de siluetas; además, no es necesario recuperar varios fotogramas de un ciclo de caminata como lo harían algunos otros métodos basados en energía. La metodología propuesta se evalúa en el conjunto de datos CASIA B [17], [18], ya que se ha convertido en el principal punto de referencia para otras investigaciones similares. Como esta base de datos no fue diseñada originalmente para la esqueletización en 2D, fue necesario etiquetar manualmente 17,732 puntos de interés para entrenar los clasificadores. Las etiquetas generadas para las partes del cuerpo de cada individuo se proporcionan como parte de esta investigación.

Es importante destacar que el dimorfismo sexual antes de la edad adulta no es suficiente para que los resultados de éste y otros algoritmos similares logren obtener buenos resultados. Además de que se requiere que los individuos estén de pie y sin obstrucciones importantes en las imágenes de entrada. Por lo que esta investigación, las investigaciones contra las que se compara, se enfocan únicamente en la detección de sexo de peatones adultos con imágenes de cuerpo completo.

1.1 Pregunta de investigación

¿Determinar el perfil antropométrico en secuencias de video por medio de la medición del esqueleto virtual es más preciso que los métodos actuales usados para el mismo fin?

1.2 Justificación

El perfil antropométrico ha sido utilizado para fines estadísticos y forenses desde finales del siglo XIX, cuando el método Bertillonage fue inventado [1], y sigue siendo utilizado hasta el día de hoy como apoyo en la localización de personas, ya que es la forma en la que la gente es descrita semánticamente.

Los métodos tradicionales se basan en la comparación de siluetas obtenidas del video analizado con perfiles conocidos; y suelen fallar si el ángulo de la imagen es diferente, o si la persona lleva ropa muy voluminosa o carga equipaje.

El método propuesto podría mejorar los resultados en estas situaciones al no depender de una silueta global.

Por otro lado, utilizar datos biométricos suaves puede reducir el número de candidatos a ser comparados para obtener mejores resultados en menos tiempo [2], [3].

1.3 Objetivos

1.3.1 Objetivo general

- Diseñar un algoritmo que genere el perfil antropométrico de personas en secuencias de video utilizando información del esqueleto.

1.3.2 Objetivos específicos

- Identificar las articulaciones de las personas presentes en el video.
- Unir las articulaciones en un único esqueleto para cada persona.
- Medir el esqueleto de cada persona.
- Diseñar y entrenar un clasificador con muestras de personas con distintas características.
- Realizar pruebas y evaluar el algoritmo.

1.4 Hipótesis

La obtención del perfil antropométrico en secuencias de video utilizando información del esqueleto tiene mejores resultados que los métodos que se basan en la silueta total de la persona.

1.5 Antecedentes

El método Bertillonage fue el primero en utilizar el perfil antropométrico (medidas del hombre) para la identificación de personas, y fue inventado por un policía francés del mismo nombre. Este método se basaba en la medición de varias partes del cuerpo y la descripción de marcas únicas en cada persona [2]. Su uso sigue siendo importante en la localización de personas, ya que facilita una descripción semántica de cada individuo. Por ejemplo, en la aplicación de la Alerta Amber.

Dado que varias personas pueden compartir el mismo perfil, se necesitó de desarrollar métodos de biometría más precisos basados en la unicidad de las características de cada individuo. La biometría dura resuelve en parte la falta de precisión de la biometría suave, pero no es capaz de ser descrita verbalmente por una persona. A pesar de esto, la biometría suave sigue siendo objeto de estudio, ya que sí permite el uso de descriptores semánticos como parámetro de búsqueda en distintos ámbitos [1].

Aunque no existe un estándar que indique cuáles son las características que mejor definen a una persona, se usa la varianza para determinar cuáles son las variables que más cambian de un individuo a otro, siendo que se obtienen mejores resultados si existe una mayor diferencia entre individuos [19].

En [19], se comparan las características más significativas del rostro que se busca con los de los 100 individuos almacenados en la base de datos, se usan las etiquetas (+2,-2) “mucho más grande que”, “más grande que”, “igual que”, “más chico que”, “mucho más chico que”; se tiene una precisión del 100% después de 30 comparaciones descartando a aquellos que no coinciden con la descripción hasta encontrar el rostro buscado. Y utilizando sólo las etiquetas +1,-1, se logra la identificación correcta de una persona después de sólo 10 comparaciones.

Poder describir a una persona con palabras permite identificarla en una escena sin necesidad de tener una imagen previa del individuo, aunque es una tarea aparentemente sencilla es un trabajo que puede tomar días en realizarse manualmente y que aún no logra ser automatizado totalmente. En este sentido la investigación se ha enfocado

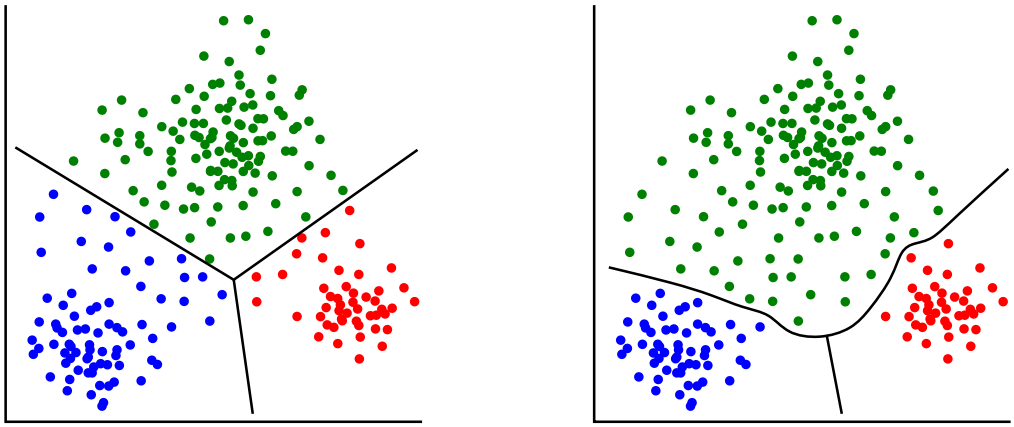
principalmente en la identificación por los colores de la vestimenta que el individuo porta al momento de su búsqueda.

En [20], se tiene como fin encontrar a personas que vistan de acuerdo con una descripción semántica. Ya que no se cuenta con una imagen previa del sujeto en cuestión, es necesario crear un avatar a partir de la descripción dada, este se crea a partir de secciones de siluetas prefabricadas del color deseado (torso, cabeza, piernas, etc.). Una vez creada la silueta, esta se compara con la escena por medio de una ventana deslizante.

Para la búsqueda, identificación, y re-identificación de personas en imágenes, la mayoría de las investigaciones se basan en dividir la imagen en ventanas del tamaño suficiente para albergar al individuo, cada una de estas ventanas es analizada independientemente verificando si la silueta encontrada coincide con la de una persona [20], [21], el principal problema que esto conlleva es que no se logrará una buena identificación si una persona tiene una pose que no estaba contemplada en el clasificador, o su ropa o equipaje modifican su silueta. De forma similar, se puede escanear el cuerpo completo con ondas de alta frecuencia para el reconocimiento de personas, es altamente efectivo, pero requieren de interactuar con el hardware, que además, es muy especializado; y se debe tomar la misma pose cada vez que se requiere de la autentificación [22], [23].

Además, para minimizar la aparición de falsos positivos, se recorre la imagen completa en busca de patrones similares a los de una persona, en algunas zonas ocurren errores mientras que en otras la ventana coincide en más ocasiones con las características buscadas, a esto se le llama mapa de confianza, y cada zona podría pertenecer a una persona específica. Posteriormente, se utiliza *Mean-Shift clustering* para determinar si todos los puntos resultantes pertenecen a la misma persona ya que este algoritmo de agrupamiento no necesita de definir el número de grupos presentes en la muestra (personas que aparecen en la imagen) [24].

Mean-shift Clustering es un algoritmo de agrupamiento por densidad, que a diferencia de otros algoritmos de agrupamiento por centroide más comunes como *k-means*, no necesita que se le indique la cantidad de grupos que hay en el conjunto de entrada, en su lugar se indica el radio de búsqueda de cada centroide. Esto permite resulta en la creación de tantos grupos como puntos de mayor densidad existan. En la Figura 1 se observa cómo los conjuntos generados por *K-means* tienen tamaños similares entre ellos, aunque los originalmente son de diferentes dimensiones, mientras *Mean-shift Clustering* genera el mismo número de conjuntos en tamaños más y formas más acordes a cada grupo.



a) b)

Figura 1. En a) se muestra cómo al agrupar un conjunto de datos con *k*-means se generan conjuntos distribuidos uniformemente a pesar de ser conjuntos con diferentes densidades. En b) se muestra el resultado de agrupar con *Mean-shift Clustering*, los grupos resultantes pueden variar en tamaño ya que se agrupan de acuerdo con la densidad de los datos.

En *Mean-shift Clustering*, cada centroide inicia en una ubicación al azar, y se recalcula el centro de cada conjunto tomando en consideración únicamente a los valores contenidos dentro un radio específico. Después de varias iteraciones, el centroide deja de cambiar de ubicación y el algoritmo se detiene en el punto con mayor densidad, Figura 2. Este algoritmo es muy útil para calcular el punto donde se encuentra cada extremidad, ya que la mayoría de los algoritmos de detección de objetos en imágenes encuentran no un punto sino un conjunto de puntos donde posiblemente se encuentra la extremidad y sólo se requiere conocer el centro de esta zona, ya que es donde generalmente se encuentra, pero no es posible sólo tomar el centro del conjunto debido que no suelen resultar ser zonas con formas geométricas regulares.

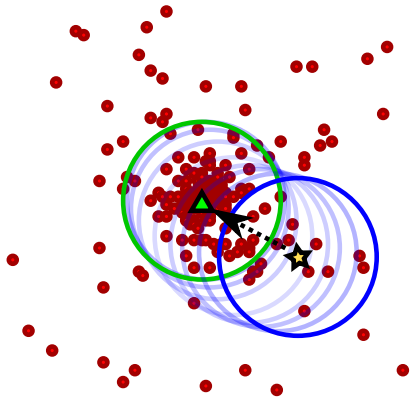


Figura 2. El centroide se calcula iterativamente con los valores contenidos dentro de un radio predefinido. el algoritmo converge cuando el centroide deja de moverse significantemente, sucede cuando llega al punto con mayor densidad.

Además, este algoritmo es capaz de inferir la cantidad de conjuntos existentes dentro del mapa de calor sin necesitar de indicárselo, lo cual es muy útil cuando el número de personas que se encuentran en la escena puede variar, y sus posiciones no son fijas a lo largo del tiempo. Esto lo logra generando una gran cantidad de centroides iniciales, ya sea tomando cada valor como uno, o generando un conjunto al azar, u ordenado de centroides; siendo esto último lo recomendado, ya que se asegura que no habrá zonas sin asignar, y el proceso será menos exhaustivo que tomar todos los puntos.

En cada iteración, se recalculan todos los centroides, los puntos que han pertenecido a un centroide se asignan a ese, aunque ya se haya movido. Además, si varios centroides se fusionan se reasignan las etiquetas de todos los puntos que tenían asignados, de esta forma se disminuye la cantidad de conjuntos posibles hasta contar únicamente con los necesarios, como se observa en la Figura 3. Finalmente, el algoritmo se detiene cuando el centroide ya no se mueve.

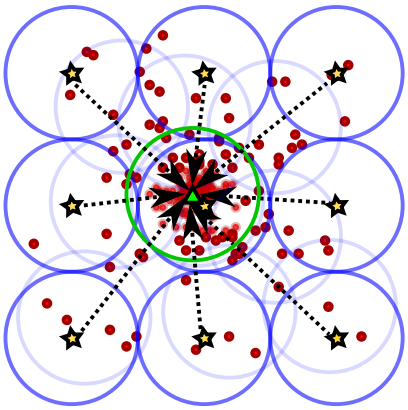


Figura 3. Cuando se desea agrupar un conjunto de datos se puede iniciar con múltiples centroides, en caso de que haya más de los necesarios, en algún punto se fusionarán para crear un único conjunto. Los puntos encontrados por múltiples centroides fusionados se reasignan para estar en un único conjunto, de esta forma, no es necesario definir a priori el número de conjuntos existentes.

Por otro lado, detectar a las personas por su forma de caminar es un método no invasivo que además da buenos resultados, pero al igual que los demás métodos de identificación, requiere de comparar al individuo con gran parte de las muestras almacenadas haciendo poco factible su implementación en poblaciones grandes. Un ejemplo de esto se aplica en [25], donde se analizan los movimientos del individuo siguiendo las articulaciones, y en [2] se aplica un método similar resaltando la importancia de las características antropomorfas en la precisión del algoritmo.

Existen dos métodos para identificar la pose de personas en imágenes: identificar a cada persona presente en la imagen y después identificar los puntos de interés (*top-down*), o identificar cada articulación y a partir de ellas identificar la pose de la persona (*bottom-up*). El primero da buenos resultados si cuando la figura humana es detectada correctamente, pero suele fallar cuando varias personas se encuentran muy cercanas unas a otra y se bloquean entre ellas. En la presente investigación, primero se identifica la silueta de la persona para disminuir el área de búsqueda, y posteriormente se identifican los puntos de interés para generar el esqueleto virtual.

Esta tesis se organiza de la siguiente manera: en el siguiente capítulo se presentan algunas obras importantes relacionadas con esta investigación. En un capítulo posterior se describen los principales pasos de la metodología propuesta. Los resultados experimentales se incluyen en el capítulo de Experimentos y resultados. Después, las conclusiones que se obtuvieron de esta tesis. Finalmente, se incluye un apéndice donde se detallan los algoritmos de clasificación utilizados en el algoritmo propuesto.

Capítulo 2. Estado del arte

En este capítulo se presenta un análisis del estado del arte con la finalidad de mostrar algunos trabajos presentes en la literatura que están relacionados con la presente tesis. Se exponen algunos algoritmos que permiten obtener diferentes características a partir de ciclos de caminado, las bases de imágenes más conocidas en investigaciones similares, y algunos algoritmos similares a la propuesta de esta investigación.

Existen diferentes métodos para reconocer a las personas captadas en video en tomas de cuerpo entero, pero se pueden dividir en dos enfoques principales: los basados en el movimiento y los basados en modelos estáticos. El primer enfoque analiza los movimientos que se producen durante un ciclo de caminata, como se muestra en la Figura 4 a; mientras que el segundo utiliza toda la información que se puede obtener de un cuerpo estático en un solo fotograma de vídeo, Figura 4 b.

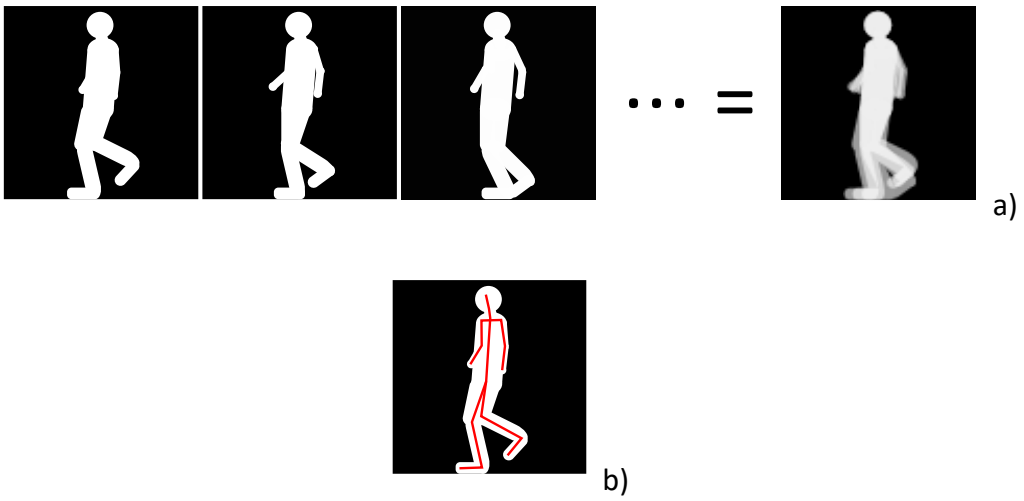


Figura 4. a) Cálculo de energía, disminuye los cambios entre siluetas, pero requiere varios fotogramas. b) Esqueletización, sólo requiere un fotograma.

A principios de este siglo, algunos artículos se centraron en la comparación de siluetas y funcionaron como base para investigaciones posteriores. Es importante analizarlos ya que se han utilizado técnicas similares para identificar el sexo de personas.

En los métodos más simples, se compara un fotograma en el que aparece la persona a ser identificada, esta imagen se binariza, esto es, los pixeles pertenecientes al primer plano se etiquetan con 1, y los del fondo con 0, la matriz resultante se compara con otra de la que ya se conocen las características, se aplica el operador *and* entre cada pixel de la imagen de entrada contra el correspondiente en la mismas coordenadas en la imagen de referencia, los pixeles resultantes se suman y dividen entre el número de pixeles de la imagen

resultante. Los pixeles que en ambas imágenes coinciden resultan en un valor 1, y 0 cuando son diferentes, así que al sumar todos los valores se obtendrá un valor alto que se normaliza entre 0 y 1 al dividirlo entre la cantidad de pixeles presentes en la imagen. Dependiendo de la calificación obtenida se decide si el objeto en la imagen de entrada es o no similar a la de referencia.

Este método es efectivo sólo cuando el objeto buscado y el de referencia están en la misma posición, tamaño y rotación. Es por esto que se suele utilizar en líneas de producción de fábricas para controlar la calidad de productos, ya que es posible ordenar los objetos en una orientación específica y se espera que todos los objetos sean siempre iguales, como en la Figura 5; también es posible usar este algoritmo como parte de reconocedores ópticos de caracteres (OCR) ya que las letras impresas suelen variar muy poco. Pero como las personas al caminar difícilmente se encontrarán en exactamente la posición esperada, no se suele utilizar directamente un fotograma sino el promedio de un ciclo de caminata que varía menos.

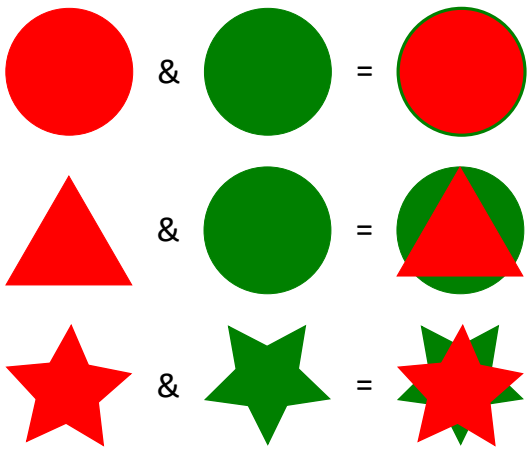


Figura 5. La imagen de entrada se compara con la de referencia, el área que comparte cada par de imágenes será mayor si son similares. En caso contrario, tendrán poca área compartida. Este método es muy rápido y eficaz cuando las figuras de entrada varían poco, pero fallan con ligeras modificaciones.

En [26], se analiza la dinámica de la caminata de personas. Dado que la altura del cuerpo varía a medida que se da cada paso, estos movimientos se pueden analizar como ondas únicas para cada persona durante un ciclo de caminata. En primer lugar, se obtiene una silueta limpia de la persona. Para ello, se toma una imagen del fondo, y es restada cuando una persona aparece en la escena. Posteriormente, el centroide de la silueta se calcula y se normaliza según el ángulo de la persona con respecto a la cámara. Según el teorema de Fourier, todos los movimientos ondulatorios pueden representarse como la suma de otras ondas armónicas (que mantienen la misma longitud y amplitud a lo largo del tiempo), como

se observa en la Figura 6 . Descomponiendo la onda generada por el andar de una persona en particular durante un ciclo de caminata, es posible calcular los patrones que generan esa onda en específico, siendo siempre muy similares en el caminar de cada individuo, pero diferentes para cada persona. Finalmente, estas ondas se comparan con las ya conocidas en la base de datos para identificar a la persona.

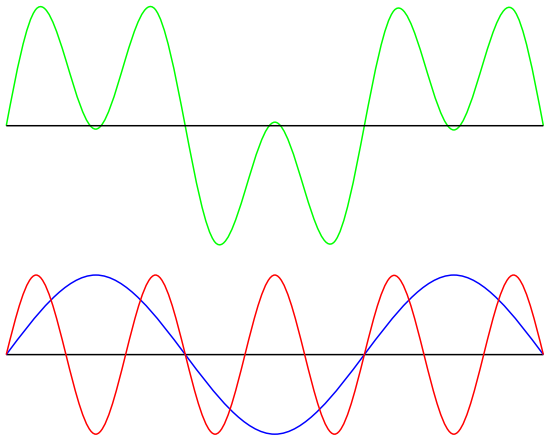


Figura 6. Cualquier onda se puede generar aproximadamente la suma de múltiples ondas armónicas. Los valores resultantes de esta descomposición pueden ser utilizados para la identificación de la onda compuesta por medio de algún clasificador.

En [27], se comparan diferentes bases de imágenes, resultando CASIA B la que cuenta con las mejores características para el propósito de esta investigación, ya que en [17], Yu et al. propusieron un marco para evaluar el rendimiento de los algoritmos de reconocimiento por la forma de caminar. Su propuesta incluye más de 15,000 videos de muestra tomados de 124 voluntarios, donde 93 son hombres y 31 son mujeres, con tomas simultáneas desde 11 ángulos con respecto a la cámara, que van desde 0° a 180°. Además, se tomaron múltiples tomas de la misma persona en diferentes condiciones: caminata normal, con equipaje y con abrigos. Esta base de imágenes, incluida en el conjunto de datos CASIA Gait, se creó originalmente para identificar cuánto afecta el ángulo de la imagen y la ropa de la persona en cada fotograma a los algoritmos de clasificación e identificación de personas. Sin embargo, al ser una base de datos muy completa, ha sido referenciada por varias investigaciones en diferentes áreas de investigación.

A continuación, en la Tabla 1 se muestra una comparativa con las principales bases de imágenes usadas para fines similares a los de la investigación presentada en este trabajo:

Base de imágenes	Sub-conjunto	Tipo de problema	Muestras	Secuencias	Origen	Caminadora	Vistas	Caminata	Año
UCSD	No	Escenas con sombras	6	7	Exterior	No	Side	Circular	1998
HID-UMD	No	Sin definir	25	1	Exterior	No	Frontal, Lateral	Recta	2001
MoBo	No	Reconocimiento en múltiples ángulos	25	4	Interior	Yes	6	Recta	2001
SOTON	Large	Multi-propósito	100	6	Interior / Exterior	Algunas secuencias	0°, 45°, 90°	Recta	2002
	Small	Cambios en la caminata	12	15	Interior	No	0°, 45°, 90°	Recta	2002
CASIA	A	Sin definir	20	12	Exterior	No	0°, 45°, 90°	Recta	2001
	B	Reconocimiento en múltiples ángulos, con equipaje y ropa holgada	124	10	Interior	No	11 vistas 0° - 180°	Recta	2005
	C	Cambios en la caminata	153	10	Exterior	No	Lateral	Recta	2005
USF Human ID	No	Condiciones variantes	122	Hasta 5	Exterior	No	Lateral	Elíptica	2005
TUM-IITKGP	No	Oclusiones	35	1	Interior	No	Lateral	Recta	2011
OU-ISIR	A	Cambios en velocidad	34	68	Interior	Yes	Lateral	Recta	2012
	B	Cambios en vestimenta	68	Hasta 32	Interior	Yes	Lateral	Recta	2012
	D	Cambios en caminata	370	185	Interior	Yes	Lateral	Recta	2012
AVA	No	Reconocimiento en múltiples ángulos	20	10	Interior	No	6	Curva / recta	2013

Tabla 1: Comparativa de bases de imágenes.

En [28], varias siluetas se calculan durante un ciclo de caminata y el promedio de cada una de ellas genera una silueta de energía única. Más tarde, se aplica lógica difusa en una variante de un algoritmo basado en LBP (*Local Binary Patterns*) para mejorar los resultados de otras investigaciones. Hasta el momento, los mejores resultados logrados se basan en modificaciones de algoritmos que usan LBP, por lo que es contra estos métodos contra los que se compara la presente investigación.

En LBP, se crea un vector de características a partir de un histograma de texturas que sirve como entrada de algún algoritmo de clasificación, a partir de este es posible determinar el sexo de la persona. A continuación, se presentan los pasos del algoritmo (ver Figura 7):

- Se recorre la imagen completa con una ventana deslizante, donde cada ventana es analizada y su resultado es un pixel en una nueva imagen filtrada. La cantidad de vecinos puede variar si se modifica el tamaño de la ventana, pero generalmente se identifican los 8 vecinos más cercanos con una ventana de 3x3.
- Se compara cada pixel de la imagen de entrada con sus 8 vecinos más cercanos. A cada pixel vecino se le asigna un valor binario; 1 si su valor es mayor que el pixel evaluado, o 0 en caso contrario. En algunas modificaciones de LBP no se comparan los valores vecinos directamente con el pixel evaluado, sino con un umbral que puede estar ligeramente por sobre o debajo de él, la forma en la que se define este umbral depende de la variante del algoritmo utilizado.
- Los valores binarios de los 8 vecinos contiguos se almacenan como un byte que después se convierte en un valor decimal, donde cada vecino tiene un peso diferente definido a priori como 2^p, donde la posición p es la ubicación de cada uno de los 8 vecinos con respecto al pixel central, por lo que p va de 0 a 7.
- El pixel analizado ahora tendrá un valor entre 0 y 255 que representa el patrón encontrado en esa ventana.
- Se calcula el histograma de los valores encontrados en toda la escena, se crea contando todas las ocurrencias de cada textura, resultando en un vector de tamaño 256.
- El histograma resultante funciona como entrada en un clasificador previamente entrenado con muestras etiquetadas de escenas similares.

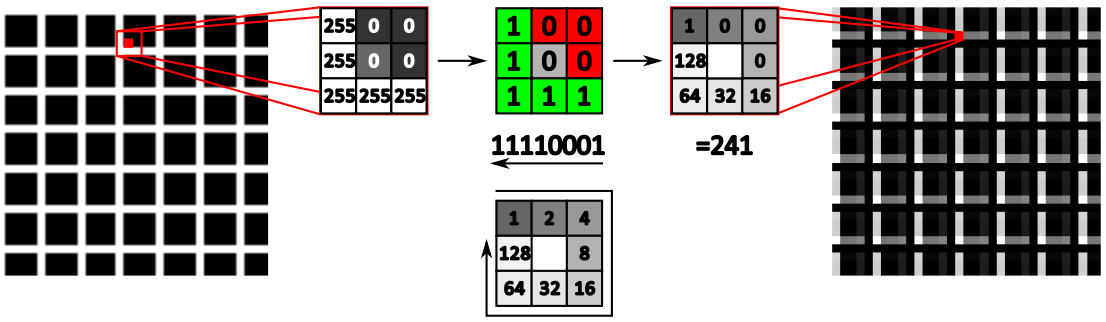


Figura 7. La imagen de entrada es filtrada con una ventana deslizante, donde cada pixel central se compara con sus vecinos, se asigna 1 o 0 si son mayores o menores. Los valores binarios resultantes son concatenados y convertidos en un valor decimal.

Es importante recalcar que en LBP, aun cuando los patrones similares tienden al mismo resultado, el algoritmo es muy sensible a ligeras variaciones, como se muestra en el ejemplo de la Figura 8. Por lo que para la detección del sexo o cualquier otra aplicación en imágenes de personas de cuerpo completo no suele aplicarse directamente a la escena, sino a la energía generada por varias siluetas a lo largo de un ciclo de caminata para disminuir estas variaciones.

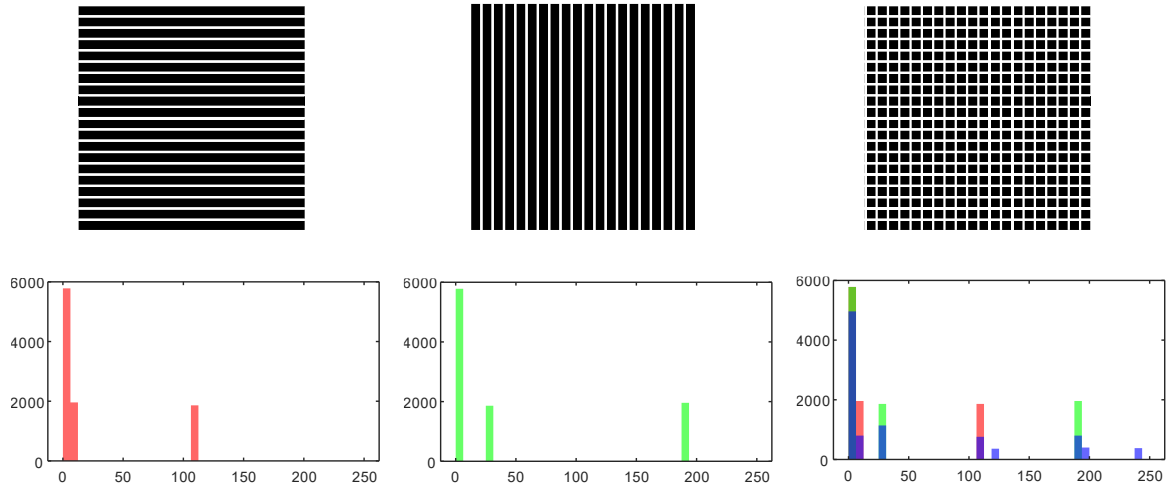


Figura 8. Se genera un histograma con los valores decimales resultantes de la imagen filtrada, los patrones que más se repiten resultarán en picos que un clasificador puede identificar.

El rendimiento de este algoritmo se evalúa en la base de datos de imágenes CASIA B y ha sido capaz de mejorar los resultados obtenidos por otros algoritmos hasta ahora. Sin embargo, mantiene la misma tendencia al error cuando se clasifica a personas con equipaje, ropa muy holgada, o el ángulo con respecto a la cámara cambia. Esto debido a que los histogramas generados difieren cuando los bordes o las texturas en las imágenes se modifican, como se muestra en la Figura 9.

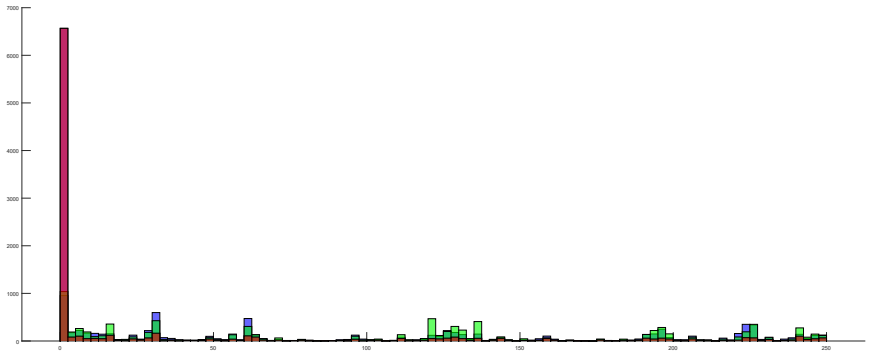


Figura 9. Los histogramas generados durante un ciclo de caminata en diferentes condiciones podrían variar demasiado, aunque a simple vista las imágenes de entrada no sean tan diferentes. Debido a esto, los algoritmos de clasificación basados en LBP podrían fallar ya que dependen de estos histogramas.

También es posible analizar los movimientos de una persona siguiendo los movimientos de sus articulaciones o del esqueleto completo [8], [25]. En[2], se calculan las medidas de diferentes partes del cuerpo, la distancia de los pasos, el tiempo de cada ciclo y la velocidad del individuo. Más adelante, esta información se compara con la almacenada en la base de datos con *k*-NN, pero puede tardar en calcularse si se maneja una gran cantidad de datos. Sin embargo, en [2], [8], los autores concluyen que las características antropomórficas pueden mejorar la precisión del algoritmo.

En la Tabla 2, se muestra una comparativa de los métodos usados para determinar el sexo de una persona:

Método:	Ventajas	Desventajas
Basados en siluetas	• Sólo requieren de un fotograma segmentado • Es computacionalmente barato	• Requiere de una segmentación perfecta • Falla cuando la pose no es igual con la que se compara • Tiene problemas cuando se usa ropa holgada o se carga equipaje • Tiene poca precisión
Basados en energía	• Es computacionalmente barato • Ya que la pose es el promedio de varias, es muy parecida con la que se compara • Buena precisión en buenas condiciones	• Requieren de varios fotogramas segmentados • Tiene problemas cuando se usa ropa holgada o se carga equipaje

Medición del esqueleto (Propuesta)	• Sólo requiere de un fotograma • Le afecta poco el uso de ropa holgada y equipaje • Buenos resultados	• Es computacionalmente costoso, pero se puede escalar y paralelizar • No requiere segmentación, pero los resultados mejoran con segmentación

Tabla 2. Comparativa de algoritmos para la determinación de sexo en secuencias de video.

La mayoría de las investigaciones sobre la esqueletización 2D tienen como objetivo identificar las actividades de personas en grabaciones de vídeo, pero a medida que se analiza cada fotograma, también es posible recuperar información del esqueleto de una sola imagen con alta precisión. Para lograr esto, a diferencia de otras investigaciones donde se busca a toda la persona, cada articulación se busca en la imagen utilizando redes neuronales convolucionales (CNN) en un primer paso [29]. Más adelante, los mapas de confianza mejoran los resultados definiendo áreas donde es muy probable que se encuentre una parte del cuerpo, las áreas con la menor probabilidad se descartan si están por debajo de un umbral que puede ser dinámico o uno predefinido [24]. Las distancias entre los puntos de interés se calculan para completar el esqueleto (campos de afinidad de partes [*Part Affinity Fields*]). De esta manera, se obtienen buenos resultados incluso cuando hay obstrucciones parciales entre la persona y la cámara [30].

Esta investigación genera esqueletos virtuales a partir de imágenes y video en 2D. A continuación, en la Tabla 3, se comparan los diferentes enfoques que existen actualmente:

Método:	Ventajas	Desventajas
Cámaras 3D	• Las mediciones de distancias precisas • El fondo se puede segmentar sin necesidad de conocer el fondo • Seguro para la salud • Poco invasivo	• Requiere hardware muy especializado
Rayos de alta frecuencia (X, etc.)	• Las mediciones de distancias precisas • El fondo se puede segmentar sin necesidad de conocer el fondo	• Requiere hardware muy especializado, invasivo y nocivo para la salud cuando el usuario se expone con frecuencia
Cámaras 2D y Redes neuronales convolucionales	• El hardware necesario es fácil de conseguir y ya está ampliamente implementado • Seguro para la salud • Poco invasivo	• Las mediciones no son exactas • No requiere de segmentación, pero da mejores resultados con imágenes segmentadas (se requiere conocer el fondo)

Tabla 3. Comparativa de algoritmos para la determinación de sexo en secuencias de video.

Finalmente, la metodología presentada en el siguiente capítulo hace uso de algunos algoritmos de inteligencia artificial que van desde la preparación de los datos, hasta la creación de modelos de clasificación. Pero debido a que algunos de ellos podrían ya ser conocidos por el lector, y son temas muy extensos, no se detallan en este capítulo sino en el Apéndice A. Algoritmos de clasificación. En el siguiente capítulo, se detalla la metodología utilizada en esta investigación.

Capítulo 3. Metodología

En este capítulo se describe la metodología propuesta para esta tesis, y tiene como objetivo determinar el sexo de las personas en función de los puntos clave del esqueleto recuperados de las imágenes estáticas. A diferencia de otros métodos que se basan en siluetas, en esta investigación se buscan diferentes partes del cuerpo en un solo cuadro de video a través de una CNN, se miden las distancias entre ellos y se utiliza esta información para hacer la clasificación final. Al hacerlo, la persona podría estar en diferentes posiciones, cambiar de vestimenta o llevar bolsas sin afectar los resultados. Esta metodología se entrenó y evaluó con un total de 5,456 cuadros y 17,732 etiquetas obtenidas de CASIA Gait Dataset [17], ya que es el conjunto de imágenes más adecuado y referenciado en trabajos similares [20].

Es importante recordar que el sexo de una persona es un concepto biológico, mientras el género es una construcción social, y ambos pueden manifestarse de diferentes formas. En biología, el término dimorfismo sexual es una condición donde algunas características físicas de individuos de la misma especie difieren entre los diferentes sexos. Mientras las expresiones de género son la conducta o apariencia que las personas utilizan para mostrar su género, y no siempre se relacionan con su sexo biológico, identidad de género, u orientación sexual.

Por otro lado, esta investigación (al igual que otras investigaciones similares) se limita a la clasificación de sexo de personas adultas en escenas tomadas de cuerpo completo, debido a que el dimorfismo sexual en niños y adolescentes no es el suficiente para ser clasificado correctamente por este algoritmo; y que además se requiere que la persona esté en una posición erguida, y sin obstrucciones mayores en las imágenes de entrada;

En las siguientes secciones se presentan los pasos de esta metodología. En la primera etapa, se entrena una CNN para identificar los diferentes puntos de interés. En la segunda etapa, se utilizó la CNN entrenada para generar un esqueleto virtual a partir de una imagen de entrada, y el sexo de la persona se clasifica con base en las distancias entre los puntos de interés que conforman el esqueleto.

3.1 Etapa 1: entrenamiento de la CNN

Una red neuronal convolucional (*CNN* por sus siglas en inglés, ver A.5 CNN (*Convolutional Neural Network*)) es un tipo de red neuronal de tipo *feed forward*, donde las primeras capas aprenden a identificar y filtrar patrones específicos que conforman una figura; destacando que, en esta primera etapa, el modelo no aprende a identificar la figura completa sino los patrones que la componen. Después de filtrar completamente la imagen de entrada, el

resultado es un vector que contiene las características de la imagen. A partir de este vector es posible inferir cuál es el objeto contenido en la imagen. De esta forma, la imagen de entrada no requiere ser exactamente igual a las usadas durante el entrenamiento, sólo contener las mismas características, además de que los filtros usados son creados durante el entrenamiento para ser especializarse en las características usadas.

Con el fin de identificar el conjunto de puntos de interés que forman el esqueleto virtual, es necesario entrenar a una CNN. Para lograrlo, se aplicaron los siguientes pasos en cada uno de los once ángulos de cámara considerados en la base de datos: En primer lugar, se seleccionaron algunos fotogramas de vídeo de CASIA B. Después, algunos puntos de interés fueron etiquetados manualmente, recortados y pre-procesados. Finalmente, las imágenes resultantes se utilizaron para entrenar once CNN, una por cada ángulo de cámara (posición y ángulo de la cámara con respecto al primer plano), para identificar partes del cuerpo. Cada uno de estos pasos se describen a continuación.

3.1.1 Selección de cuadros

A diferencia de otros métodos, esta propuesta sólo requiere un cuadro para clasificar a una persona como hombre o mujer. Para ello, se selecciona un cuadro donde la persona aparezca en el centro de la escena, como se representa en la Figura 10. De esta manera, todas las muestras se toman desde una distancia similar y se evitan problemas causados por escalas diferentes.

Figura 10. Se toma un fotograma de la secuencia de video original donde la persona está justo en el centro de la imagen para asegurar que todas las partes del cuerpo sean visibles.

La detección del fotograma donde el individuo está en el centro de la escena se logra seleccionando una zona de la imagen en la que se verifica la diferencia entre dos fotogramas consecutivos. Cuando no hay objetos en movimiento en esa zona, la diferencia es casi nula;

pero cuando la persona entra en esta zona en el centro de la escena, la diferencia se incrementa, y ese cuadro se selecciona.

3.1.2 Etiquetado

Existen 3 categorías de aprendizaje de máquina diferentes en los que la computadora es capaz de realizar inferencias: aprendizaje automático supervisado, no supervisado, y por refuerzo. El primero es especialmente útil para clasificar datos, y es necesario mostrarle por medio de ejemplos cuál es el resultado deseado, a partir de ellos se genera un modelo capaz de identificar los patrones aprendidos. En el aprendizaje no supervisado, el algoritmo analiza datos sin necesidad de tener conocimiento previo, es muy usado para agrupar datos. Y, por último, el aprendizaje por refuerzo aprende con prueba y error en simulaciones donde sólo conoce reglas básicas que irá probando hasta encontrar la mejor forma de solucionar el problema.

Dado que este algoritmo propuesto se basa en aprendizaje automático supervisado, ha sido necesario seleccionar 1364 cuadros y etiquetar manualmente 17,732 puntos de interés (13 por cuadro): cabeza, hombros, codos, manos, cadera, rodillas y pies. Para ello, se encierra cada parte del cuerpo en un cuadrado de tamaño suficiente para albergar completamente cada una de las partes del cuerpo sin espacio de sobra como se muestra en Figura 11. Debido a que todas las imágenes tienen la misma resolución, el tamaño de todas las etiquetas se definió de $28x28$ para esta base de imágenes. A continuación, para cada cuadrado, se guardan las coordenadas de su esquina superior izquierda con el nombre de la parte del cuerpo que contiene. Con el fin de facilitar la comprensión de esta propuesta, algunas figuras muestran sólo las partes superiores del cuerpo, pero el mismo proceso se aplica al resto del cuerpo.

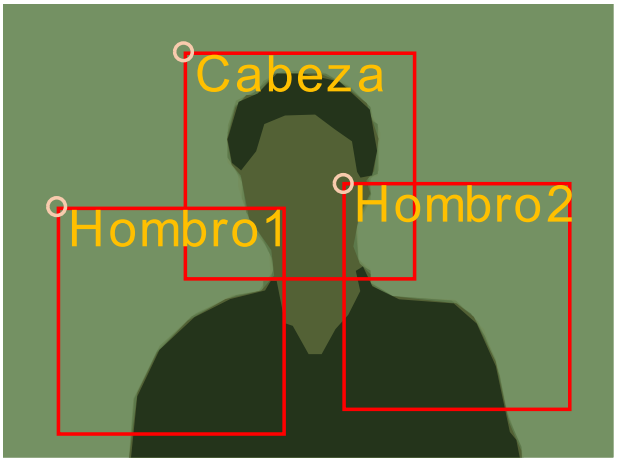


Figura 11. Etiquetas de la cabeza y hombros.

Es necesario señalar que el número del fotograma en el vídeo para cada ángulo de cámara, la versión de la muestra, y las regiones seleccionadas que pertenecen a cada etiqueta, están disponibles como parte de esta investigación [31].

3.1.3 Preparación de imágenes

Una vez finalizado el etiquetado, cada selección se recorta y se almacena en carpetas independientes, una para cada parte del cuerpo, como se muestra en Figura 12. Los conjuntos de etiquetas resultantes se utilizan para entrenar una CNN por ángulo de cámara.

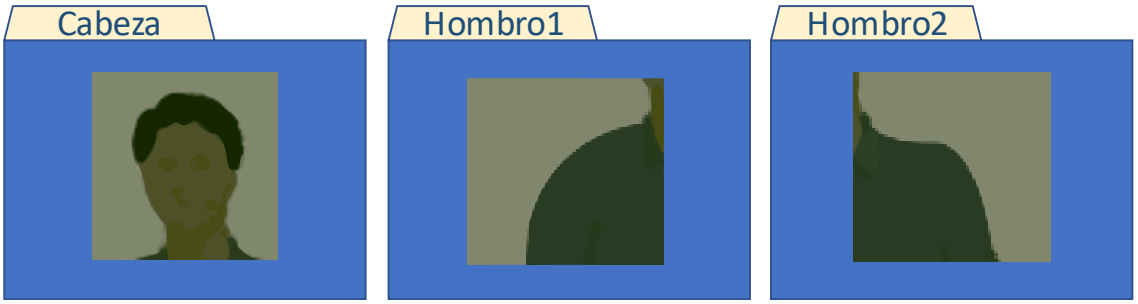


Figura 12. Cada sección recortada se almacena en un folder para cada parte del cuerpo.

El espacio de color más conocido en el que las imágenes se captan y reproducen es el RGB (rojo, verde, y azul por sus siglas en inglés), donde cada pixel contiene los datos sobre cuánta cantidad de luz de cada color se emite. Pero en ocasiones, estos datos no son suficientes para analizar una imagen y necesitan convertirse a otros espacios de color. Por ejemplo, otras características muy útiles que no existen en el espacio RGB, pero se pueden calcular a partir de él son la cromaticidad, saturación, y luminiscencia; estas características existen en el espacio HSV, la primera se refiere a la tonalidad o matiz del color que puede variar con respecto a la longitud de onda que se representa o la combinación de diferentes ondas, que van desde el rojo hasta el violeta, la saturación del color va desde el estado más puro del color hasta el blanco, y el brillo se refiere a la cantidad de luz representada en cada pixel, la luminosidad mínima representa el negro. En la Figura 13 se muestran ambos espacios.

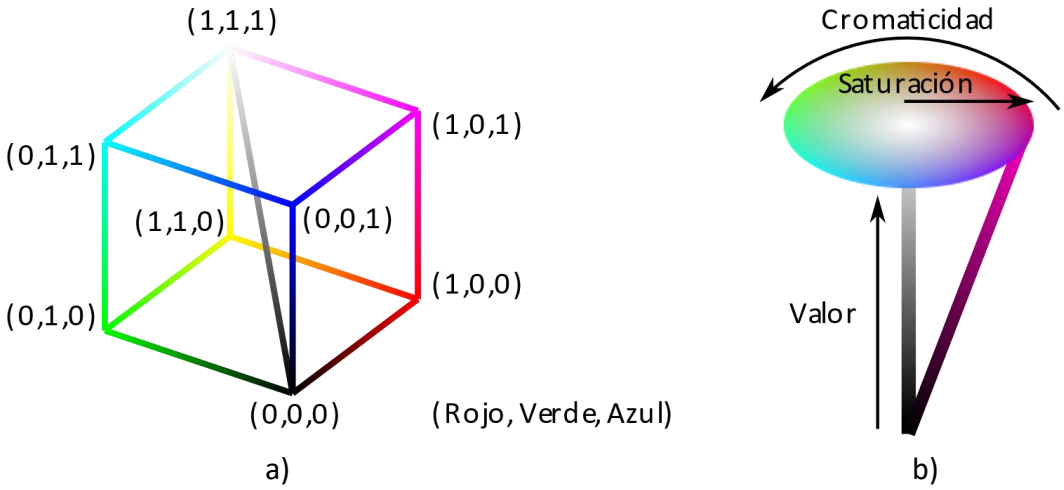


Figura 13. a) espacio de color RGB, b) espacio de color HSV.

Los colores representados en el espacio RGB también pueden convertirse a otros espacios, pero para fines de esta investigación sólo se muestran las equivalencias al espacio HSV. Mientras en el espacio RGB, el color está representado en un vector tridimensional (ver Figura 13a); en HSV, la cromaticidad se calcula como el ángulo de este vector con respecto al color rojo, debido a que es una rueda, el color rojo en HSV puede ser representado por 0 o 1 (valores equivalentes a 0° y 360°), como se muestra en la Figura 13b y la Figura 14; la saturación es la distancia entre el vector a ser convertido y el punto más cercano en un vector que va del origen al punto [1, 1, 1] que representa la escala de grises, ver Figura 13a.

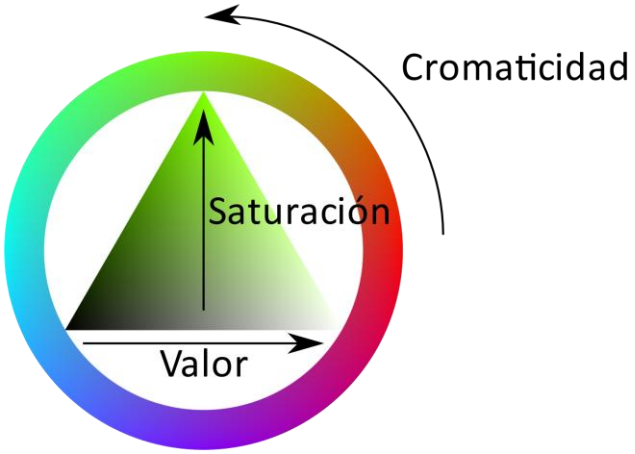


Figura 14. En el espacio de color HSV, la cromaticidad (H [*Hue*]) es un ángulo donde el color rojo se presenta por 0 y 1, mientras la saturación (S [*Saturation*]) y el valor o brillo (V [*Value*]) son sólo valores en el rango [0, 1].

Finalmente, el valor necesario para balancear los colores en la escena es el brillo, y es calculado como la magnitud del vector del color; existen otros métodos para convertir una imagen a blanco y negro en la que los colores son corregidos para distinguidos más

fácilmente por humanos, ya que las personas distinguimos algunas cromaticidades mejor que otras; pero esta corrección además de innecesaria podría alterar los resultados.

Si bien el color blanco técnicamente no existe ya que podría ser cualquier cromaticidad (Figura 15), se considera un color semántico, y para realizar un balance de blancos se considera como el valor más alto en la imagen, y se calcula con la fórmula de distancia euclidiana con respecto al origen (1).

$$Brillo = \sqrt[2]{R^2 + G^2 + B^2} \qquad (1)$$

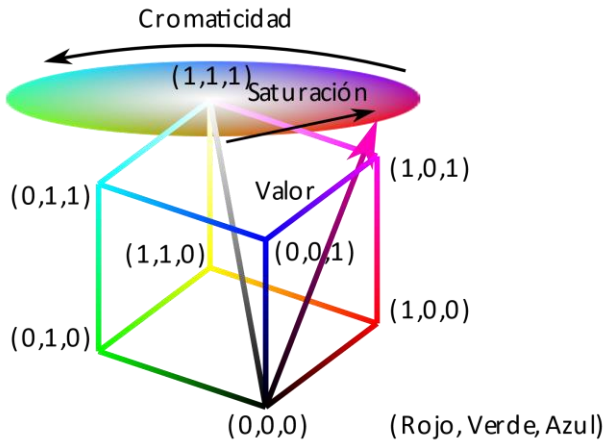


Figura 15. Conversión entre espacios de color RGB y HSV.

Cuando en una escena la iluminación tiene alguna coloración diferente a la luz blanca, todos los pixeles tienen una tendencia a esa cromaticidad, esto significa que en el espacio RGB todos los pixeles tienen un sesgo hacia esa cromaticidad, incluso los colores que se encuentran en el sentido opuesto del vector de la iluminación de la escena. Esto se puede corregir si se encuentra la diferencia entre una muestra de color blanco con el sesgo de la escena, y el color [1, 1, 1] (o [255, 255, 255] si los colores están representados en un formato de 24 bits). La diferencia encontrada se suma a cada pixel en la imagen con lo que toda la imagen tendrá una nueva coloración más neutral, como se representa en la Figura 16.

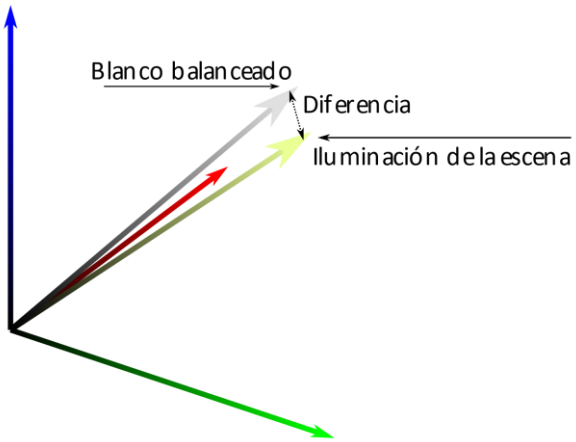


Figura 16. El vector que representa la escala de grises se encuentra desfasado con respecto al verdadero eje de las escalas de grises, después de calcularse cuán desfasados están, se aplica esa corrección a cada pixel de la imagen.

Con el fin de mejorar la clasificación, todos los colores presentes en las muestras se corrigen mediante balanceo de blancos. Este proceso mejora el contraste en la imagen y disminuye las diferencias de iluminación entre las escenas tomadas en diferentes momentos del día. Para ello, al pixel más claro de la imagen se le resta el valor de un color blanco predefinido (con la iluminación más alta y la saturación más baja), y esta diferencia resultante es añadida a cada pixel.

3.1.4 Entrenamiento de la CNN

La selección del cuadro, el etiquetado y la preparación de la imagen descritos anteriormente se repitieron para cada uno de los once ángulos de cámara considerados en el conjunto de datos CASIA B. De esta manera, se produjeron once conjuntos de muestras utilizados para entrenar las diferentes CNN que aprenden a identificar partes del cuerpo desde cada ángulo de cámara.

La entrada de una CNN es una imagen RGB del conjunto de entrenamiento, y ambos deben tener las mismas dimensiones ($3x28x28$). Después de una iteración, la salida es un vector de probabilidad de $13x1$ donde cada parte del cuerpo tiene mayor o menor probabilidad de que esté presente en la muestra de entrada.

- Capa de entrada: es una porción de la imagen recortada, tiene 3 canales por ser RGB
- Capas intermedias: se aplican operaciones en cada capa y se obtienen mapas de características.
    - Kérnel de convolución: busca características, recorrido por ventana deslizante diferente a la anterior, a medida, filtra la imagen, tiene tamaño

igual a la entrada, se hace una multiplicación y suma, la salida es de tamaño inferior y de igual número de kérnels.

- Normalización por lotes: las entradas de una red se suelen normalizar para que cada característica tenga el mismo peso en la misma escala, entre capas por la convolución los valores crecen muy rápidamente, es necesario normalizar los valores de nuevo, el rango en el que se deben normalizar se debe calcular por lotes y no con base en el valor de entrada de cada iteración.

- Capa de rectificación (ReLu): en caso de segmentación pueden existir valores negativos que no aportan información, y pueden incrementar el tiempo de entrenamiento de la red. Por lo que todos los valores negativos se convierten en 0, el resto de los valores se mantienen intactos.

- Capa de agrupamiento: disminuye la resolución sin perder información importante, las características que sí se encontraron en la convolución tienen valores altos, los valores bajos ocurren cuando no se encuentran, los valores bajos no son relevantes, se suele usar más *Max Pooling*, también existe *Average Pooling* que calcula el promedio.

- Aplanamiento: una CNN tiene un comportamiento similar a una ANN (Red Neuronal Artificial [*Artificial Neural Network*]), una ANN requiere de un vector de entrada y no funcionaría bien con una matriz, esta capa convierte de matriz a vector, es posible disminuir las dimensiones sólo con convolución y agrupamiento, pero se podría perder información sobre la distribución de información o información importante, cada valor en esta última capa representa una característica importante.

- Capa totalmente conectada: comportamiento igual que una ANN, realiza la última clasificación con base a las características encontradas.

- Capa de salida: vector de probabilidades, cada valor corresponde a una clase

El diseño de la CNN utilizada es el siguiente:

- Entrada 3@28x28
- Kérnel de Convolución 5x5

Mapa de características 1@28x28

- Normalización por lotes
- Capa de rectificación (ReLU)
- Capa de agrupamiento (MaxPooling) 2x2

Mapa de características 1@14x14

- Kérnel de Convolución 5x5

Mapa de características 32@14x14

- Normalización por lotes
- Capa de rectificación (ReLU)
- Capa de agrupamiento (MaxPooling) 2x2

Mapa de características 32@7x7

- Kérnel de Convolución3x3

Mapa de características 64@4x4

- Normalización por lotes
- Capa de rectificación (ReLU)
- Capa de agrupamiento (MaxPooling) 2x2

Mapa de características 64@2x2

- Aplanamiento

Capa oculta 256

- Capa totalmente conectada

Salida: Vector de probabilidades 13

La arquitectura utilizada se detalla en la Figura 17.

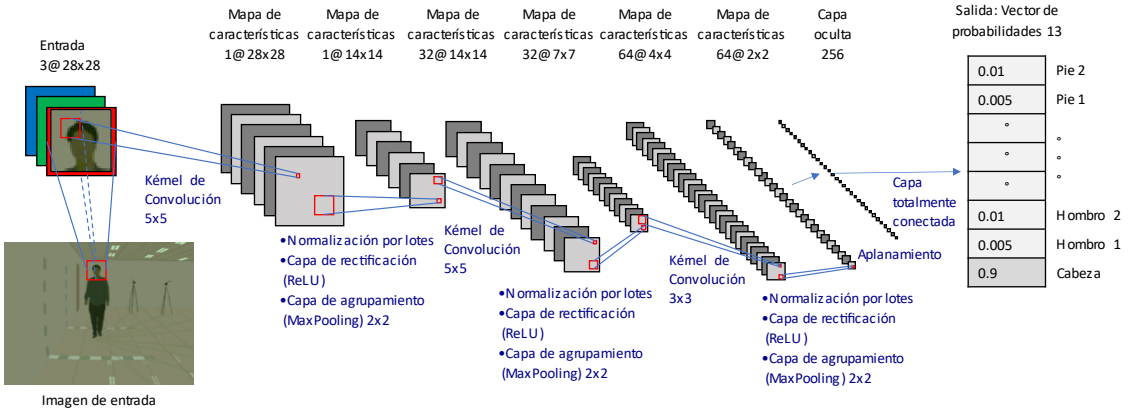


Figura 17. Arquitectura de la CNN

Después del entrenamiento, los modelos de aprendizaje automático generalmente dan buenos resultados con los datos usados durante el entrenamiento, pero estos datos difícilmente se replicarán en muestras nuevas. Para determinar cuán bueno será el desempeño con datos no vistos durante el entrenamiento, las muestras disponibles se suelen dividir en al menos dos conjuntos, uno para entrenamiento y otro para pruebas.

Dependiendo del desempeño del algoritmo durante el entrenamiento y las pruebas, existen tres escenarios posibles: bajo ajuste, sobreajuste (o sobre entrenamiento), y entrenamiento óptimo. En el primero, los resultados en el entrenamiento y las pruebas son bajos; ocurre cuando el algoritmo usado para generar el modelo no es adecuado, o los datos no son suficientes. En el sobreajuste, los resultados del entrenamiento son muy buenos, pero el modelo falla durante las pruebas; sucede cuando el modelo memoriza los datos sin realmente aprender. Finalmente, el entrenamiento óptimo dará buenos resultados durante el entrenamiento, y similares en las pruebas; ocurre cuando el modelo obtenido es capaz de generalizar conocimiento sin memorizarlo, y se aceptan la existencia de algunos datos atípicos que son clasificados incorrectamente.

Para evitar un posible bajo ajuste, se diseña un modelo adecuado para los datos usados, y se usa la mayor cantidad de datos disponibles. Y para evitar entrenar el modelo de más, se crea un tercer conjunto de datos con datos que no se usan directamente para entrenar el modelo, sino para validar los resultados durante el entrenamiento; para esto, el entrenamiento se pausa cada determinado número de iteraciones, se valida, y se continúa el entrenamiento si los resultados mejoran con respecto a la validación anterior, en caso contrario, se detiene aún si los resultados del entrenamiento continúan mejorando. Posteriormente, se prueba el modelo con el conjunto de prueba.

Para esta etapa se crearon manualmente y usaron un total de 17,732 etiquetas disponibles como parte de esta investigación. Dada la naturaleza de este algoritmo, se destinó el 70% de estas etiquetas para un entrenamiento iterativo, y 15% para pruebas. Para evitar un sobreajuste, cada modelo fue validado durante el proceso de entrenamiento con el 15% de etiquetas no se incluidas en los conjuntos de entrenamiento o pruebas.

El entrenamiento de una CNN es un proceso iterativo, donde cada imagen del conjunto de entrenamiento sirve de entrada una a la vez. En las capas intermedias la imagen se filtra para exaltar las características más relevantes. Después los datos se rectifican para evitar que haya datos negativos que incrementan el proceso de aprendizaje y no aportan información relevante, y los datos se normalizan para evitar que alguna característica tome mayor importancia que el resto. Posteriormente, las dimensiones de la imagen filtrada se reducen manteniendo mayoritariamente las características relevantes encontradas en la capa anterior. Este proceso puede repetirse en capas que tienen el mismo comportamiento, pero cuentan con dimensiones diferentes. En la última capa, la imagen de entrada es clasificada como una parte del cuerpo, ya que se cuenta con imágenes etiquetadas, es posible saber si el modelo acertó o se equivocó, y en caso de haber errado, se puede saber la magnitud del error. Cuando los resultados no son los esperados, la magnitud del error sirve para corregir los pesos de los filtros y capa oculta, a mayor error mayor será la rectificación. Finalmente, y después de varias iteraciones, el error disminuye hasta sobrepasar un umbral de aceptación, o la calidad en el conjunto de validación decae, y el entrenamiento se detiene. Este proceso detalla en A.5.

Los pasos principales de la primera etapa se incluyen en Tabla 4 y Figura 18. Además, todos los scripts están disponibles como parte de esta investigación en [31].

Entrada: Base de imágenes CASIA Gait Database (videos originales)

Salida: Modelo entrenado para el reconocimiento de partes del cuerpo

1. Selección de cuadros
2. Etiquetado de puntos de interés
3. Preparación de imágenes
4. Entrenamiento del modelo para el reconocimiento de partes del cuerpo (CNN)

Tabla 4. Pseudocódigo de la primera etapa.

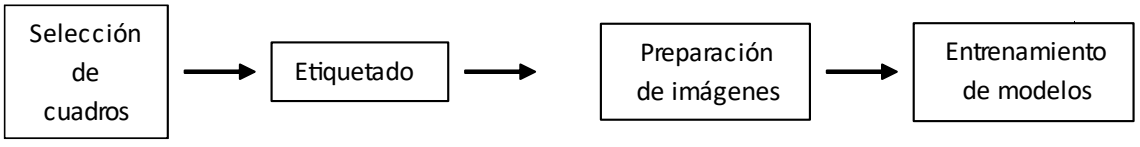


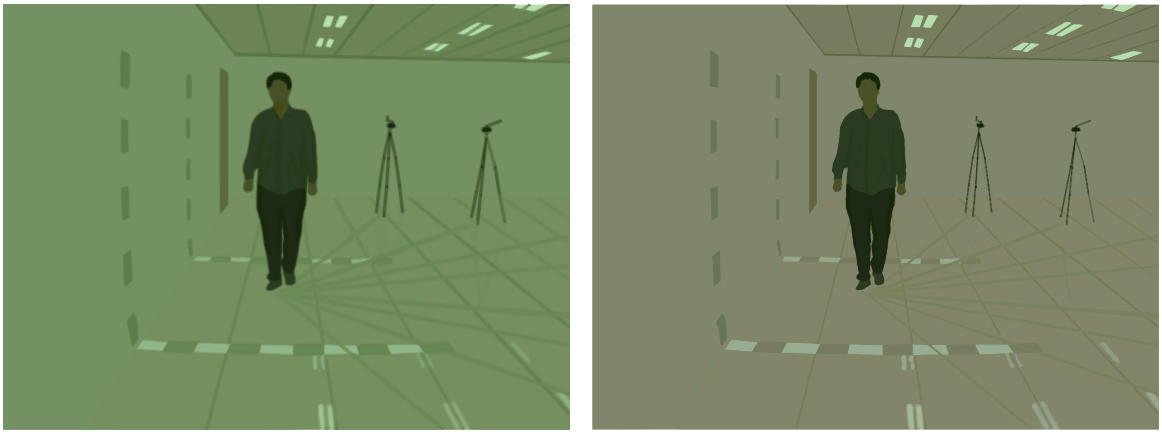
Figura 18. Diagrama de flujo de la primera etapa.

3.2 Etapa 2: clasificador de sexo

Esta es la última etapa, utiliza los modelos entrenados en la capa anterior para determinar si la persona en la imagen de entrada es hombre o mujer. Después de que las CNN han sido entrenadas, se implementan para encontrar los puntos de interés presentes en un fotograma, con los que se crea un esqueleto virtual, y las distancias entre estos puntos se utilizan para determinar el sexo del individuo. Antes de ser evaluadas, las imágenes de entrada de esta etapa deben prepararse para contar con características similares a las de las imágenes utilizadas durante el entrenamiento de la CNN.

3.2.1 Preparación de imágenes

Al igual que las imágenes etiquetadas utilizadas para el entrenamiento de las *CNN*, los cambios en la iluminación podrían modificar los colores de la escena. Por lo tanto, todos los cuadros también requieren de balanceo de blancos, de la misma forma que se describió anteriormente en la sección Preparación de imágenes. En Figura 19a y 19b, se muestra una escena antes y después del balance de blancos, respectivamente. De esta forma, se reducen las diferencias de iluminación entre los cuadros de entrada y el modelo entrenado. Esta estrategia se aplica para evitar cualquier clasificación errónea debido a cambios de color. Por ejemplo, de no aplicarse el balanceo de blancos, el modelo podría aprender a identificar colores en lugar de figuras, por lo que no podría aplicarse a diferentes condiciones de iluminación.



a) b)

Figura 19. Balanceo de blancos.

A continuación, se identifica si una zona pertenece al fondo o al primer plano por medio de diferentes técnicas de segmentación de imágenes. De esta forma, no es necesario explorar toda la imagen con la CNN para identificar los puntos de interés. Al hacerlo, el algoritmo requiere menos iteraciones y los falsos positivos ocurren con menor frecuencia.

La segmentación de imágenes permite dividir la imagen de entrada en regiones más pequeñas para después extraer las características locales de esas zonas. Dependiendo de la aplicación que se necesite existen diferentes técnicas de segmentación de imágenes:

- Basadas en umbrales: Generalmente se aplican en imágenes en blanco y negro, pero también es posible aplicarlas en imágenes a color. Se define un umbral de brillo o cromaticidad, a partir del cual cada segmento se asignará a una u otra clase dependiendo de si está por encima o por debajo de tal umbral.
- Basadas en bordes: La imagen de entrada es filtrada con detectores de bordes, se identifica un borde cuando existe una diferencia importante entre pixeles contiguos. Puede fallar en zonas donde los bordes no rodean totalmente una zona, o cuando existen zonas degradadas.
- Basadas en agrupamiento: Agrupa los pixeles que comparten características similares en segmentos no necesariamente contiguos. El algoritmo más conocido es *K*-means, en el que se le deben definir la cantidad de grupos en los que se quiere dividir la imagen. Generalmente se utiliza para segmentar imágenes por colores.
- Basadas en texturas: Una textura es un patrón repetitivo en la imagen, estas técnicas identifican desde figuras geométricas hasta cualquier patrón asimétrico repetitivo.
- Basadas en regiones: Asigna cada pixel a un segmento basándose en si es similar o no a su vecino en cuanto a textura, color, etc. Las técnicas más conocidas son la de crecimiento de región por semilla (*Seeded Region Growing*), y dividir y unir (*Split and merge*). En el primer algoritmo se seleccionan uno o varios pixeles, y recursivamente se asignan o no los vecinos a esta sección dependiendo de si cumplen con algún criterio de similitud. En el segundo, la imagen es dividida inicialmente en 4 regiones simétricas, se verifica si los pixeles dentro cada región son similares entre ellos, o existe una varianza muy alta; las regiones que resulten ser poco homogéneas se dividen independientemente de nuevo hasta que lo sean; finalmente, cada región se compara con sus vecinos y se fusionan en un mismo segmento si son similares; generalmente se obtienen regiones de diferentes tamaños, esto para evitar una búsqueda tan exhaustiva en caso de ser posible.
- Mixtas: Es posible combinar diferentes técnicas de segmentación para mejorar los resultados, o realizar un proceso más eficiente.

Si bien generalmente las técnicas de segmentación suelen usar sólo una imagen para agrupar los elementos de la imagen, para esta investigación se usó una técnica basada en umbrales ya que únicamente se requiere separar el fondo y el primer plano, y la base de imágenes CASIA B incluye muestras del fondo para este fin [17]. Debido a que las imágenes se capturaron con cámaras es estáticas, la mayoría de los pixeles en la imagen con el fondo y los de la escena analizada coinciden, y se eliminan de la imagen, quedando solo los que pertenecen al primer plano. Para lograrlo, el fondo se resta de la escena analizada, consulte

la Figura 20a y b. Los valores absolutos más altos en la imagen resultante suelen corresponder a los objetos con mayor movimiento, como se muestra en la Figura 20 c.

a) b) c)

Figura 20. Fondo, escena de entrada, y primer plano.

Es una resta pixel por pixel con información de cada capa, se suman los resultados de las 3 dimensiones, los valores absolutos más altos indican movimiento. En escenarios con demasiados cambios de luz se puede usar el espacio hsv con sólo la cromaticidad, como se muestra en la Figura 21; y segmentar antes la imagen, ya sea por agrupamiento, o por reducción de bits.

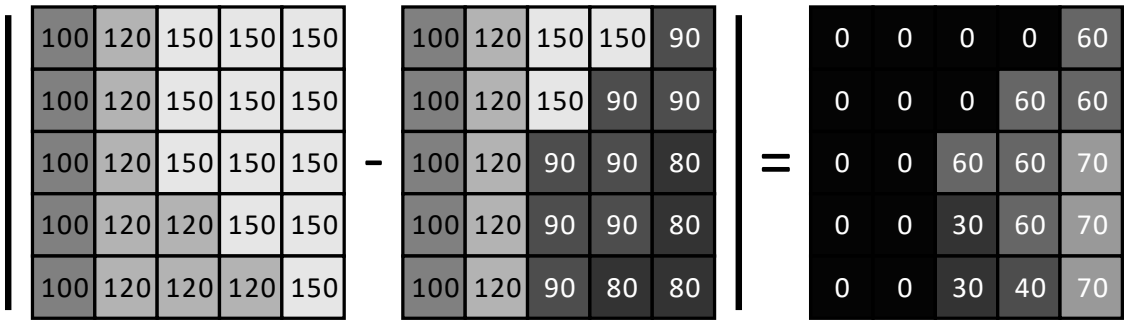


Figura 21. Al restar los valores absolutos de cada pixel entre dos imágenes, se obtiene que los pixeles que difieren entre ambos resultan es valores altos, mientras los pixeles similares tienden a 0.

En caso de no contar con un fondo en blanco, se puede calcular la media de varios fotogramas.

Finalmente, la imagen se binariza para segmentar la imagen. Esto es, a todos los pixeles en primer plano se les asigna un 1 como valor, y a los del fondo un 0 para diferenciarlos, en la Figura 22 se muestra un ejemplo.

0	0	0	0	60
0	0	0	60	60
0	0	60	60	70
0	0	30	60	70
0	0	30	40	70

Primer plano detectado

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

Primer plano binarizado

Figura 22. Los valores en la imagen en escala de grises más bajos se convierten en 0 ya que pertenecen al fondo de la imagen, mientras los valores más altos se asignan en 1 que representan el primer plano.

Cuando otros objetos se mueven entre escenas, o el fondo es similar a la vestimenta de la persona, aparece ruido en la imagen resultante como se muestra en la Figura 23.

Figura 23. Primer plano binarizado.

Para corregir este problema, cada segmento se dilata y erosiona. Esto significa que el borde del primer plano se modifica dos veces; primero se aumenta para rellenar agujeros, y a continuación se reduce para desaparecer zonas más pequeñas. Los resultados de este proceso se muestran en la Figura 24.

Se recorre la imagen, si un pixel del primer plano tiene vecinos marcados como fondo, todos estos vecinos se marcan como pertenecientes al primer plano. La erosión tiene un proceso similar, pero con el proceso invertido, donde todos los pixeles del borde del primer plano se convierten en parte del fondo.

Figura 24. Primer plano binarizado sin ruido.

3.2.2 Creación del esqueleto virtual

En esta etapa, se crea un modelo anatómico de la estructura interna del cuerpo humano de cada individuo. Se decidió generar los esqueletos virtuales utilizando una metodología similar a la descrita en [30], ya que esta estrategia ha reportado los mejores resultados en entornos 2D. Dependiendo de la aplicación, el esqueleto virtual podría contar con más o menos elementos, para esta aplicación se consideran únicamente: la cabeza, los hombros, los codos, las manos, la cadera (ambos lados), las rodillas y los pies.

En la etapa anterior, se definió una CNN capaz de identificar estos puntos del esqueleto con muestras de cada una de estas partes del cuerpo, con vistas desde cada uno de los 11 ángulos de cámara. En esta etapa cada imagen de entrada se analiza con una CNN para identificar los puntos de interés presentes, y con ellos se genera un esqueleto virtual como se detalla a continuación.

Después de segmentar la imagen, sólo se analiza el primer plano a través de una ventana deslizante de 28x28 pixeles de dimensión. Es importante recordar que esta ventana requiere tener las mismas dimensiones que la capa de entrada de la CNN, y las etiquetas usadas durante el entrenamiento, ya que cada ventana sirve como una entrada de una CNN previamente entrenada, que a su vez devuelve un vector de probabilidad, el cual se utiliza para crear un mapa de confianza para cada punto de interés, como se ve en la Figura 25.

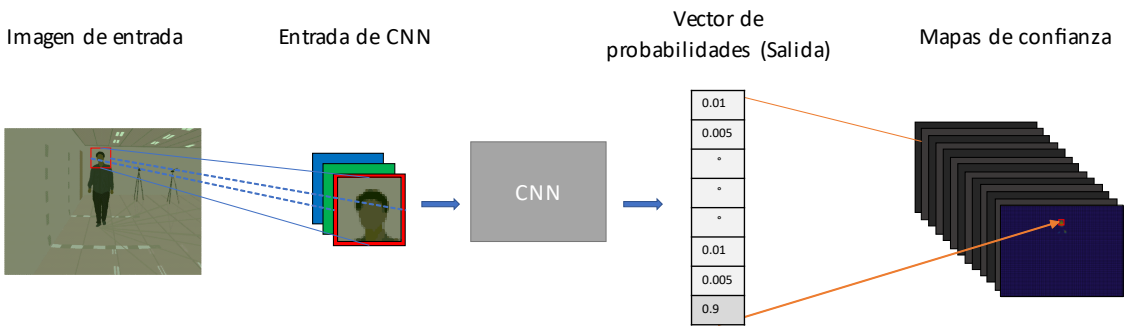


Figura 25. Cada CNN genera un pixel en todos los mapas de confianza.

Cuando se crean los mapas de confianza con los valores resultantes de la CNN, algunos falsos positivos pueden ocurrir durante la clasificación, como se ve en la Figura 26, donde algunas áreas se están clasificadas erróneamente como cabeza. Por lo tanto, es necesario identificar los pixeles que indican correctamente la posición de cada parte del cuerpo. Para ello, la siguiente estrategia se aplica en cada mapa de confianza. En primer lugar, se identifica la probabilidad más alta, p_{max}. Posteriormente, se eliminan todos los pixeles con un valor por debajo de un umbral t. Dado que p_{max} no es un valor fijo, y t requiere ser recalculado para cada mapa de confianza, después de varios experimentos, se llegó a la conclusión de que la mejor opción era $t = p_{max} - 0.1$. De esta forma, se mantiene la suficiente información para calcular cada punto del esqueleto virtual, ya que un valor menor a $p_{max} - 0.1$ introduciría ruido innecesario en el mapa de confianza reduciendo la precisión, y un valor superior podría eliminar información necesaria en el siguiente paso.

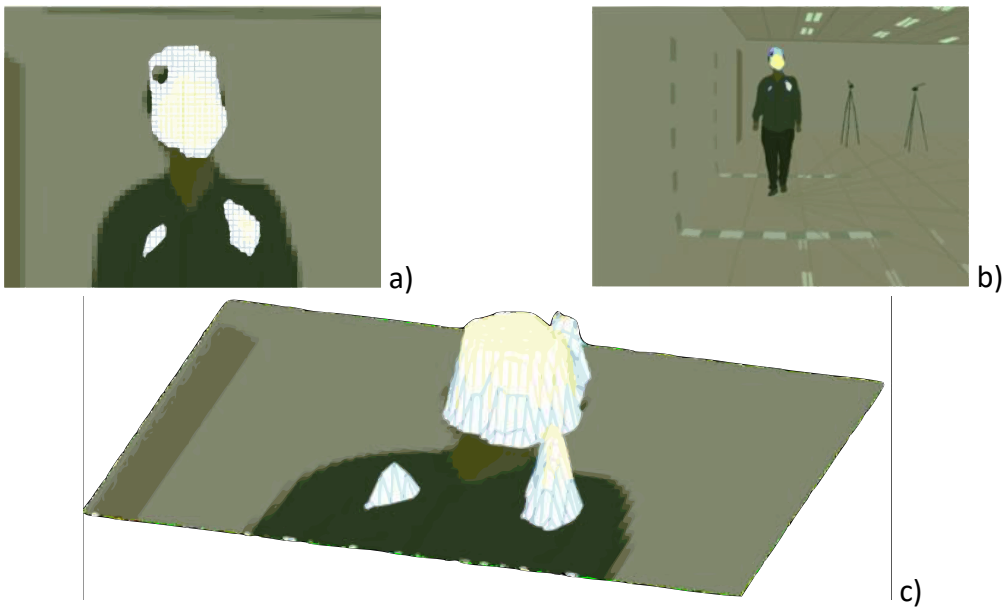


Figura 26. Acercamiento en el mapa de confianza generado para una cabeza.

Por último, la ecuación (2) es implementada para calcular el centroide ponderado de los pixeles restantes, donde x_i y y_i son las coordenadas del i-ésimo pixel, p_i es su probabilidad y n es el número de pixeles seleccionados. De esta forma, el centroide resultante se coloca dentro de la zona con las probabilidades más altas. En la Figura 27, los puntos resultantes para la cabeza y los hombros se muestran sobre el primer plano.

$$\bar{x} = \frac{\sum_{i=1}^{n}(x_i * p_i)}{\sum_{i=1}^{n} p_i}, \bar{y} = \frac{\sum_{i=1}^{n}(y_i * p_i)}{\sum_{i=1}^{n} p_i} \quad (2)$$

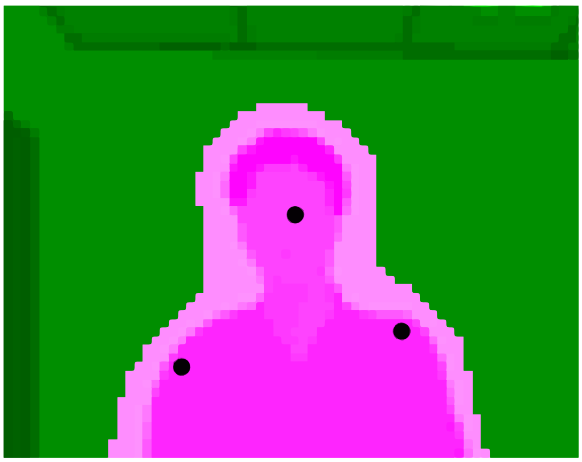


Figura 27. Puntos de interés resultantes para cabeza y hombros (el resto de los puntos de interés existen de forma similar, pero no se muestran en esta imagen).

3.2.3 Clasificador de sexo

Una vez obtenido el esqueleto virtual de la persona en la escena, se calculan las distancias euclidianas entre cada punto de interés. A continuación, se utiliza la altura en pixeles de la persona analizada para normalizar estas distancias, ya que la escala de la escena varía según la distancia y el ángulo con respecto a la cámara, de otra forma no sería posible saber si la persona difiere en altura con respecto al resto de muestras, o si únicamente está más alejado o cercano a la cámara en la escena.

Un clasificador tiene como objetivo identificar las características principales de una muestra y asignarle la etiqueta que mejor la describa, en el aprendizaje supervisado las clases elegible ya están definidas desde antes del entrenamiento, mientras en el aprendizaje no supervisado los grupos posibles se definen durante el análisis de las muestras. Dada la naturaleza del problema presente en esta investigación, se utiliza aprendizaje supervisado, siendo el sexo de la persona en la muestra, la clase que se asigna a la imagen de entrada.

En esta etapa final, todas las distancias encontradas en el esqueleto virtual se utilizan como entrada para los clasificadores que identifican el sexo del individuo. Se consideraron 3 algoritmos diferentes: *k*-NN, SVM, y ANN.

K-nn es un algoritmo de clasificación (y regresión) simple que usa directamente los datos históricos disponibles. A diferencia de otros algoritmos de clasificación, este no se requiere de un entrenamiento especial más allá de la recolección de datos, ya que la clasificación se hace directamente sobre los datos en tiempo de ejecución, por lo que permite la inserción de nuevos datos sin requerir un reentrenamiento. Para realizar la clasificación de una muestra se calcula la distancia entre sus valores y los de sus *k* vecinos más cercanos en el conjunto de entrenamiento, dependiendo de la zona en la que se localice tendrá más vecinos de una clase específica, y a esta se asignará. Existen diferentes formas de calcular distancias, entre las que se destaca la distancia euclidiana.

SVM busca un vector capaz de dividir el conjunto de entrenamiento bajo el supuesto de que también lo hará correctamente con el conjunto de pruebas, esta separación es lineal y sólo puede identificar dos clases a la vez, en caso de necesitar más clases se entrenan SVM adicionales. Debido a que la mayoría de las veces los conjuntos no son linealmente separables, es necesario incrementar las dimensiones del conjunto original por medio de funciones matemáticas o kérnels específicos para este fin. En muchas ocasiones es necesario probar diferentes funciones hasta encontrar la que dé mejores resultados, y el vector de soporte que divide los conjuntos puede ser calculado por diferentes técnicas de optimización.

ANN modela una representación general los datos de entrenamiento en forma de funciones, los valores de entrada se multiplican contra pesos establecidos en funciones lineales en múltiples capas. Entre cada capa se aplican funciones no lineales los valores resultantes de la capa anterior para modificar las dimensiones de los datos. A diferencia de las SVM, al agregar capas intermedias es posible incrementar aún más las dimensiones en caso de ser necesario. Su entrenamiento es iterativo por lo que el entrenamiento puede resultar tardado. Durante el entrenamiento, se calcula el error para cada muestra para actualizar los pesos de cada capa; esto se repite hasta que error sea mínimo, que ocurre cuando los resultados son buenos. Si bien tiene como desventaja realizar requerir una gran cantidad de operaciones matemáticas, en su mayoría son multiplicaciones de matrices que pueden calcularse de forma paralela en una Unidad de Procesamiento de Gráficos (GPU), con lo que se mejora significativamente el tiempo de ejecución.

En el Apéndice A. Algoritmos de clasificación se detalla el funcionamiento general de estos algoritmos. Los experimentos y resultados se muestran en el siguiente capítulo. Los pasos principales de la segunda etapa se incluyen en la Tabla 5 y la Figura 28. Además, todos los scripts están disponibles como parte de esta investigación.

Entrada: Imagen de fondo y cuadro a analizar

Salida: Clase (masculino o femenino) correspondiente a la persona en el cuadro de entrada

1. Preparación de imágenes
2. Generación de esqueleto virtual
3. Clasificación de sexo

Tabla 5. Pseudocódigo del clasificador de sexo.

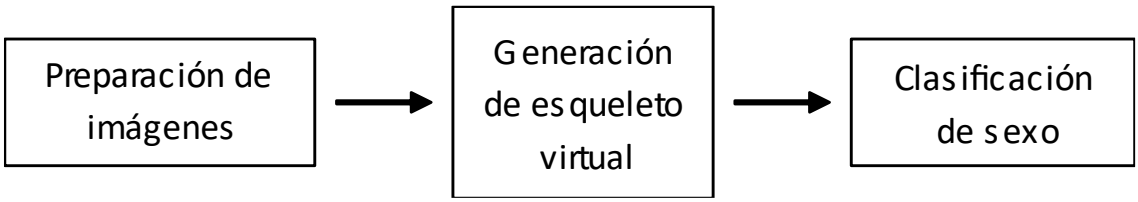


Figura 28. Diagrama de flujo del algoritmo propuesto.

Capítulo 4. Experimentos y resultados

En este capítulo se muestran los experimentos realizados y los resultados obtenidos después de entrenar y evaluar el algoritmo propuesto en el capítulo anterior. Se probaron diferentes algoritmos de clasificación con diferentes configuraciones con el fin de encontrar los que mejor se ajustan a esta investigación. Los resultados obtenidos se muestran en comparación con los obtenidos en investigaciones similares presentes en el estado del arte.

4.1 Experimentos

Para identificar cuál es el mejor clasificador para la determinación de sexo, se evaluaron las distancias resultantes entre los puntos de interés con tres clasificadores diferentes en MATLAB ®. En primer lugar, se utilizaron los clasificadores SVM y *k*-NN incluidos en *Statistics and Machine Learning Toolbox 11.4*. Por último, se implementó un ANN con *Neural Network Start Toolbox.* Los algoritmos SVM y k-NN fueron entrenados y evaluados con validación cruzada *k-fold* (*k*=5); y para la ANN, se destinó el 70% de los datos para el entrenamiento, el 15% para la validación y el 15% para la prueba.

Se realizaron varios experimentos para encontrar la mejor configuración de SVM, *k*-NN y ANN. Y se concluyó utilizar las siguientes configuraciones:

- SVM cuadrática: Separación cuadrática entre clases.
- Coseno *k*-NN: Similitud del coseno, 10 vecinos más cercanos
- ANN: *Feed-forward backpropagation neural network* de tres capas, con función sigmoide y 20 neuronas en la capa oculta, y *SoftMax* en la capa de salida.

Para el entrenamiento de cualquier algoritmo de aprendizaje supervisado, es necesario proporcionar datos sobre los diferentes escenarios que se puedan presentar. Para esta investigación, se consideraron tres escenarios: caminata normal, caminata con equipaje y con vestimenta holgada en 11 ángulos de cámara diferentes. Una vez entrenados estos modelos, el sexo de cada individuo se puede identificar utilizando las distancias entre los puntos de interés encontrados en la imagen de entrada.

Con el fin de evaluar y comparar el rendimiento de los resultados obtenidos con los de otros algoritmos conocidos, se implementó *F1-score* como se propuso en [28]. Dónde:

$$F1 = \frac{2 \times Precisión \times exhaustividad}{Precisión + exhaustividad} \quad (3)$$

$$Precisión = \frac{Positivos\ verdaderos}{Positivos\ verdaderos\ +\ Falsos\ positivos} \quad (4)$$

$$Exhaustividad = \frac{Positivos\ verdaderos}{Positivos\ verdaderos +\ Falsos\ negativos} \quad (5)$$

4.2 Resultados

De la Tabla 6 a la Tabla 8, se muestran las puntuaciones F1 obtenidas por esta propuesta cuando se aplican al conjunto de datos de imágenes CASIA B en las diferentes condiciones de caminata, y lo reportado en investigaciones anteriores: *local binary pattern* (LBP), *Local XOR Pattern* (LXP), *Ternary Pattern* (LTP), *Local Ternary Pattern* (LTP), *Soft LBP* (SLBP), *Fuzzy Local Binary Pattern* (FLBP), y *Local Binary Patterns With Tuned Parameters* (FLBP*).

Se puede observar que, en la mayoría de los casos, esta propuesta con ANN reportó las mejores puntuaciones, y en los pocos casos donde fue superada por otras técnicas, la diferencia con la mejor puntuación fue pequeña. Por ejemplo, FLBP* reportó mejores puntuaciones que esta propuesta en algunos ángulos de cámara con caminata normal; sin embargo, la mayor diferencia entre ambos algoritmos es 0.07 para 72°, como se muestra en la Tabla 6.

Aunque los resultados reportados por FLBP* en condiciones normales son buenos, disminuyen cuando la persona en la escena lleva equipaje o usa ropa suelta, ya que la silueta que genera se modifica por estas nuevas condiciones. Dado que nuestra propuesta no se basa en esas características, los resultados del algoritmo no se ven afectados, como se muestra en el Tabla 7 y Tabla 8.

	0°	18°	36°	54°	72°	90°	108°	126°	144°	162°	180°
LBP	0.86	0.92	0.91	0.90	0.85	0.86	0.86	0.87	0.92	0.91	0.88
LXP	0.88	0.92	0.93	0.92	0.91	0.84	0.89	0.87	0.90	0.90	0.93
LTP	0.87	0.85	0.88	0.85	0.89	0.80	0.84	0.84	0.88	0.86	0.88
SLBP	0.91	0.89	0.93	0.94	0.93	0.84	0.93	0.92	0.95	0.85	0.95
FLBP	0.93	0.92	0.94	0.97	0.92	0.90	0.91	**0.97**	0.97	0.92	0.96
FLBP*	**0.95**	**0.96**	**0.97**	**0.98**	**0.98**	0.94	**0.95**	**0.97**	**0.98**	0.96	**0.98**
SVM (prop.)	0.93	0.89	0.84	0.89	0.90	0.94	0.92	0.92	0.94	0.95	0.92
k-NN (prop.)	0.90	0.86	0.83	0.87	0.84	0.90	0.91	0.88	0.91	0.90	0.89
ANN (prop.)	0.94	0.94	0.94	0.94	0.91	**1.00**	0.94	**0.97**	0.97	**0.97**	0.94

Tabla 6. *F1-score* de una caminata normal bajo diferentes ángulos de cámara.

	0°	18°	36°	54°	72°	90°	108°	126°	144°	162°	180°
LBP	0.84	0.74	0.73	0.71	0.72	0.79	0.77	0.75	0.82	0.86	0.80
LXP	0.84	0.80	0.77	0.69	0.68	0.73	0.73	0.75	0.85	0.85	0.82
LTP	0.78	0.78	0.81	0.71	0.73	0.70	0.66	0.75	0.73	0.78	0.86
SLBP	0.85	0.86	0.85	0.66	0.80	0.80	0.79	0.80	0.83	0.87	0.85
FLBP	0.83	0.82	0.81	0.74	0.60	0.79	0.83	0.81	0.89	0.92	0.87
FLBP*	0.91	0.91	0.89	0.82	0.86	0.86	0.87	0.81	0.90	0.94	0.93
SVM (prop.)	0.93	0.88	0.90	0.82	0.86	**0.92**	**0.93**	0.90	0.92	0.95	0.92
k-NN (prop.)	**0.94**	0.86	0.88	0.84	0.85	0.91	0.90	0.87	0.91	0.94	0.90
ANN (prop.)	**0.94**	**0.94**	**0.94**	**0.88**	**0.91**	0.91	0.91	**0.94**	**0.94**	**0.97**	**0.97**

Tabla 7. *F1-score* de una caminata con equipaje bajo diferentes ángulos de cámara.

	0°	18°	36°	54°	72°	90°	108°	126°	144°	162°	180°
LBP	0.79	0.75	0.70	0.85	0.75	0.76	0.77	0.79	0.79	0.78	0.80
LXP	0.79	0.84	0.75	0.81	0.79	0.80	0.77	0.83	0.81	0.75	0.80
LTP	0.75	0.83	0.72	0.72	0.79	0.73	0.75	0.77	0.72	0.67	0.78
SLBP	0.74	0.83	0.79	0.78	0.81	0.80	0.79	0.83	0.82	0.83	0.84
FLBP	0.79	0.83	0.82	0.82	0.80	0.80	0.80	0.84	0.82	0.87	0.82
FLBP*	0.85	0.86	0.86	0.85	0.87	0.87	0.87	0.84	0.88	0.87	0.88
SVM (prop.)	0.91	0.87	0.85	0.88	0.90	0.88	0.93	0.92	0.91	0.93	0.92
k-NN (prop.)	0.91	0.86	0.86	0.88	0.86	0.90	0.91	0.93	0.90	0.90	0.91
ANN (prop.)	**0.94**	**0.91**	**0.91**	**0.91**	**0.91**	**0.97**	**0.97**	**0.94**	**0.97**	**0.97**	**0.94**

Tabla 8. *F1-score* de una caminata con ropa holgada bajo diferentes ángulos de cámara.

Del mismo modo, en las Figuras 29 a 31 se muestran los diagramas de caja de las puntuaciones reportadas en las tablas anteriores (de la Tabla 6 a la Tabla 8). En la Figura 29, se puede observar que FLBP* y ANN obtienen el mejor rendimiento para caminar normalmente. Sin embargo, de acuerdo con las Figuras 30 y 31, la ANN supera a sus contrapartes cuando la persona lleva equipaje o ropa holgada. Se considera que la principal ventaja de esta propuesta es la poca variación en las distancias entre los puntos de interés en el esqueleto virtual independientemente del ángulo, ropa o accesorios que lleve la persona, lo cual ayuda a mejorar el rendimiento de la clasificación. De hecho, hay que destacar que las puntuaciones obtenidas por SVM, *k*-NN y ANN difieren poco entre los tres escenarios.

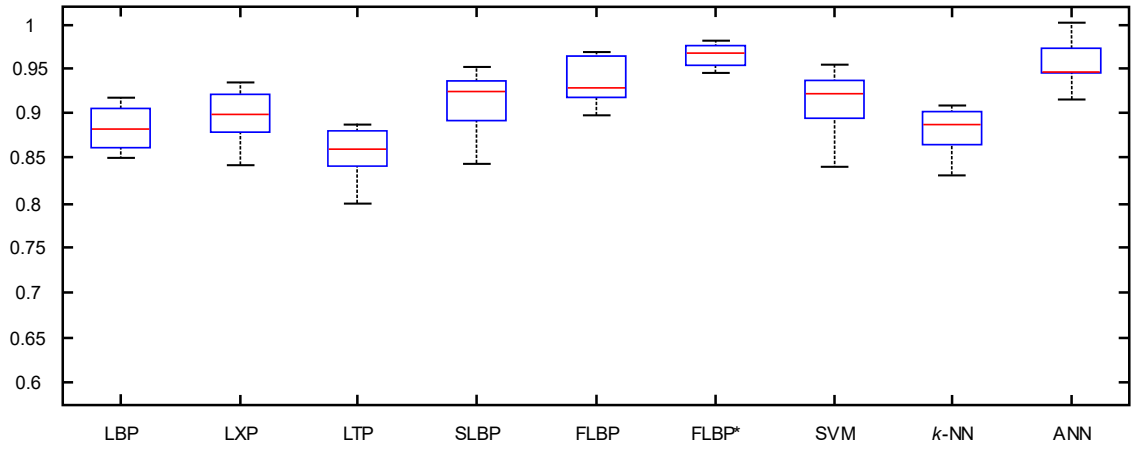


Figura 29. Diagrama de caja de una caminata normal.

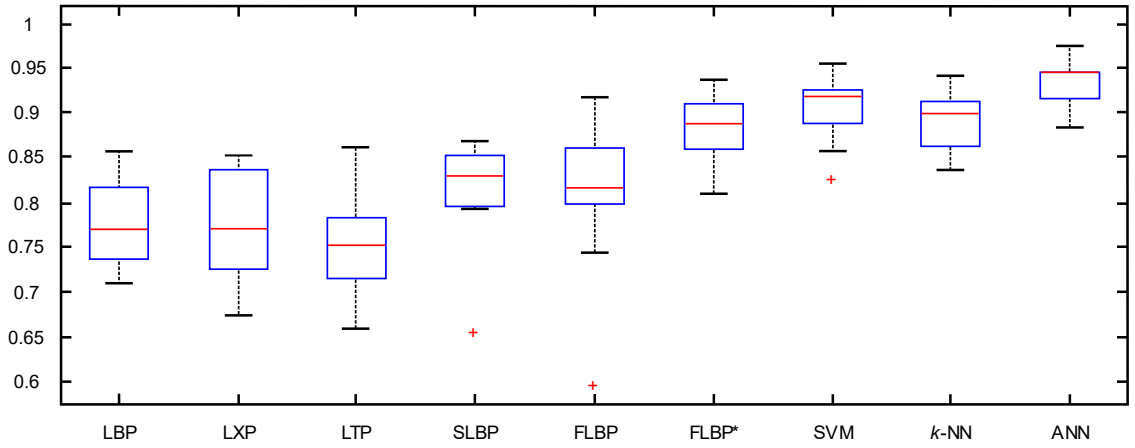


Figura 30. Diagrama de caja de una caminata con equipaje.

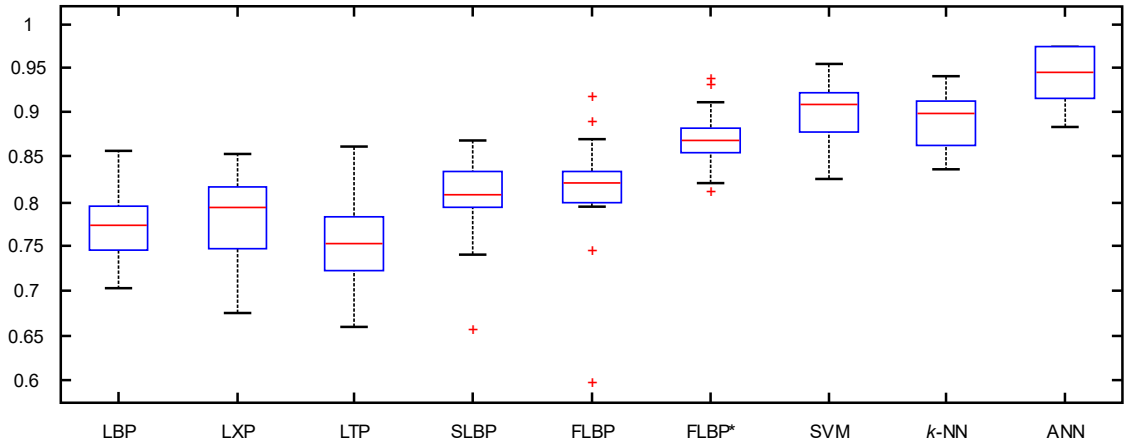


Figura 31. Diagrama de caja de una caminata con ropa holgada.

Por último, con el fin de comparar el rendimiento general de todos los algoritmos, se analizaron todas las puntuaciones reportadas para cada técnica (mostradas de la Tabla 6 a la Tabla 8). En la Figura 32 se incluye un diagrama de caja para todos los algoritmos teniendo en cuenta su rendimiento bajo todos los ángulos de cámara y condiciones de caminata. De acuerdo con esta cifra, se puede inferir que la ANN tiene el mejor rendimiento; sin embargo, para confirmar esta observación se realizó una prueba de Wilcoxon, con un nivel de significancia del 5%.

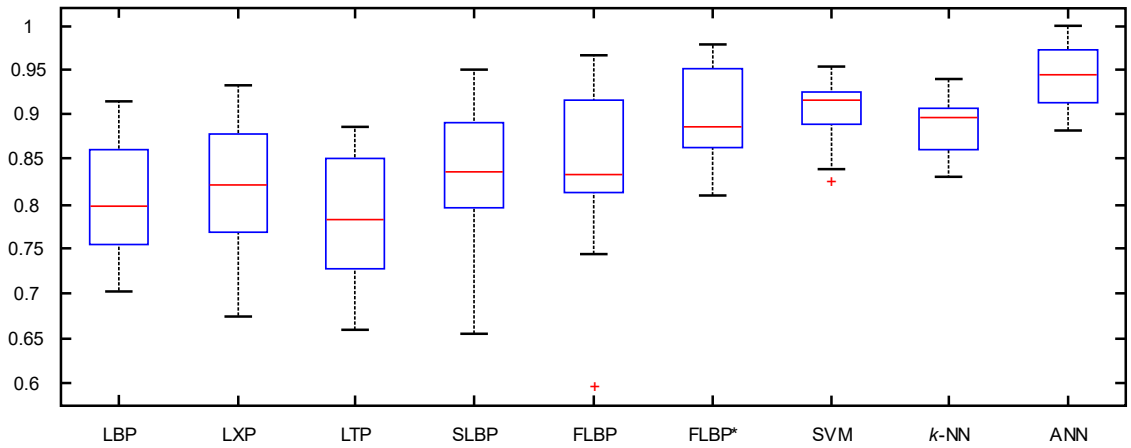


Figura 32. Diagrama de caja considerando todos los escenarios.

La prueba de Wilcoxon analiza la hipótesis nula de que las medianas de los algoritmos comparados en pares son idénticas. Una vez más, se utilizan todas las puntuaciones reportadas de la Tabla 6 a la Tabla 8. Si se aceptó la hipótesis nula, se asignó un valor 0 a ambos algoritmos, lo que representa que ambas estrategias presentan el mismo comportamiento (no existe una diferencia significativa entre ambos conjuntos de soluciones). Si se rechazó la hipótesis nula, se asignó un valor de 1 a ambos algoritmos. En la Tabla 9 se observa que FLBP* es estadísticamente similar a SVM y *k-NN,* mientras que ANN es diferente a todas las demás técnicas. Combinando los resultados de este análisis con el diagrama de caja de la Figura 32, se concluye que la metodología propuesta con una ANN obtiene los mejores resultados, superando las estrategias restantes consideradas en este trabajo. Por otro lado, esta metodología con SVM o k-NN, en lugar de ANN, puede considerarse como la segunda mejor opción junto con FLBP*.

	LBP	LXP	LTP	SLBP	FLBP	FLBP*	SVM (prop.)	*k*-NN (prop.)	Ann (prop.)
LBP	1	1	1	0	0	0	0	0	0
LXP	1	1	1	1	1	0	0	0	0
LTP	1	1	1	0	0	0	0	0	0
SLBP	0	1	0	1	1	0	0	0	0
FLBP	0	1	0	1	1	0	0	0	0
FLBP*	0	0	0	0	0	1	1	1	0
SVM (prop.)	0	0	0	0	0	1	1	0	0
k-NN (prop.)	0	0	0	0	0	1	0	1	0
Ann (prop.)	0	0	0	0	0	0	0	0	1

Tabla 9. Resultados binarios de la prueba de Wilcoxon.

5 Conclusiones

En este trabajo se propuso una nueva metodología para la identificación del sexo de personas en secuencias video. Primero, se entrenó una red neuronal convolucional para identificar la cabeza, hombros, codos, manos, cadera, rodillas y pies de una persona en un fotograma. A continuación, las distancias entre estos puntos se utilizaron para clasificar a la persona como hombre o mujer. Para esta etapa, se utilizaron tres algoritmos basados en SVM, k-NN y ANN. Finalmente, el rendimiento de esta metodología se evaluó en la base de imágenes CASIA B y se comparó con 6 algoritmos reportados anteriormente.

Los resultados numéricos muestran que la metodología propuesta funciona mejor cuando se utiliza con ANN en lugar de SVM o k-NN. Sin embargo, si se consideran todas las instancias disponibles en CASIA B, las dos estrategias tienen el mejor o el segundo mejor comportamiento entre todos los algoritmos comparados en este trabajo. Además, todas las etiquetas y clasificadores capacitados utilizados para la clasificación de sexo están disponibles para cualquier otra investigación similar que pueda requerirlos.

Las principales ventajas de esta investigación pueden resumirse en dos puntos. En primer lugar, es posible utilizar solo un fotograma, en lugar de una sección de un vídeo, para obtener una clasificación de alta precisión. En segundo lugar, las pequeñas variaciones en las distancias entre los puntos de interés en el esqueleto virtual mejoran el rendimiento general de los algoritmos de clasificación. De hecho, en la Figura 32 se observa que el rango de puntuaciones para SVM, *k*-NN y ANN es pequeño, en comparación con los algoritmos. Por lo tanto, se infiere que los algoritmos tenían aproximadamente el mismo comportamiento independientemente del ángulo de cámara o las características específicas de la persona en el cuadro. Por último, se llegó a la conclusión de que esta metodología puede conducir a mejorar el rendimiento de los algoritmos de clasificación con fines similares.

Es importante mencionar que debido a que el algoritmo necesita encontrar todos los puntos de interés de la persona para funcionar como se espera, se limita a identificar el sexo de personas en fotogramas donde la persona esté de pie, en una vista de cuerpo completo y sin obstrucciones importantes sobre el primer plano. Además, debido a que el algoritmo aprende a distinguir diferentes medidas del cuerpo a partir de las muestras con las que es entrenado, antes de implementarlo en alguna población, se recomienda reentrenarlo con muestras de personas de esa misma población.

Finalmente, como trabajo futuro se propone investigar si es posible identificar otros descriptores semánticos además del sexo con base en el algoritmo presentado en esta tesis. Esto requeriría la adaptación y reentrenamiento del algoritmo, y la creación de una nueva base de imágenes, que además sería de gran utilidad para investigaciones de otras áreas.

Apéndice A. Algoritmos de clasificación

A.1 Aprendizaje automático

En inteligencia artificial, el aprendizaje automático o aprendizaje máquina tiene como fin que las computadoras aprendan de forma autónoma a predecir o agrupar patrones por medio de algoritmos matemáticos y estadísticos. La computadora aprende a identificar patrones con base en su experiencia o datos históricos. Para lograrlo, se obtiene un modelo capaz de generalizar el conocimiento aprendido, sin la necesidad de haber sido explícitamente dotado con reglas para solucionar cada situación [32].

El modelo generado contiene reglas, parámetros e hiperparámetros con los que la computadora es capaz de tomar decisiones autónomamente. Los hiperparámetros son diferentes para cada algoritmo y problema que se requiere solucionar, son definidos por el programado antes de la etapa de entrenamiento, y contienen la definición de algunas características de modelo; por ejemplo, el tamaño del modelo y la velocidad con la que aprende, entre otras. Los parámetros son definidos durante el entrenamiento por la computadora, y aunque la forma en la que se obtienen depende del tipo de aprendizaje que se aplique, suele iniciar como parámetros aleatorios y se modifican iterativamente hasta terminar el entrenamiento.

La mayoría de los algoritmos de aprendizaje automático corresponden a un problema de optimización (minimización de una función objetivo), donde se deben encontrar los parámetros de un modelo capaz de realizar una predicción con un error mínimo. Idealmente, la función de error tiene forma de U, esto significa que existe un punto (conjunto de parámetros) que corresponde al error mínimo rodeado por una infinidad de soluciones posibles de peor calidad, Figura 33. Por lo tanto, es posible aplicar técnicas de optimización para espacios convexos que minimicen el error del modelo [33].

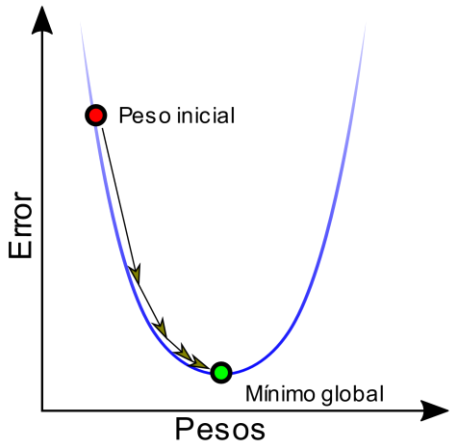


Figura 33. La solución óptima corresponde a la que tenga el error mínimo global.

Para poder entrenar el algoritmo es necesario contar con información histórica suficiente, y con datos que aporten información útil. Además, la calidad del modelo debe ser medible con base en sus resultados; y en caso de tener un entrenamiento iterativo, debe ser medible durante el proceso.

Para obtener mejores resultados, es recomendable preparar los datos a usar antes de entrenar el algoritmo. Para esto se cuenta con técnicas como: limpieza de datos, que se refiere a evitar usar valores vacíos, repetidos, inválidos, inconsistentes, o atípicos; selección de características, donde se escogen los atributos más relevantes para resolver el problema, y se eliminan aquellas que son irrelevantes o redundantes; la transformación de características, donde los datos cambian de formato por medio de la normalización y la reducción de dimensiones. De esta forma se simplifica el proceso y se mejoran los resultados [34].

Es importante evaluar la calidad del clasificador con algunos datos históricos disponibles, para tener una aproximación de su desempeño cuando se aplique a datos nuevos de los que se desconocen sus clases. Por lo anterior, se recomienda evaluarlo con datos diferentes a los reservados para el entrenamiento. Así que los datos disponibles suelen dividirse en conjuntos de entrenamiento y prueba, para algoritmos no iterativos, y además uno de validación para algoritmos iterativos; donde generalmente el conjunto de entrenamiento contiene más muestras que los de validación y prueba. Los datos de entrenamiento contienen el conocimiento necesario para encontrar los parámetros óptimos del modelo de aprendizaje automático. El conjunto de validación contiene ejemplos que permiten afinar los parámetros del modelo, evaluando los resultados obtenidos durante el entrenamiento. Finalmente, el conjunto de prueba contiene datos no utilizados durante el entrenamiento y permite evaluar el desempeño del modelo generado.

Los datos necesarios para entrenar y evaluar el modelo deben contener atributos predictores y objetivo. Donde los predictores son la entrada del modelo, y a partir de ellos el modelo predice los objetivos. Durante la validación y prueba, se utilizan sólo los atributos de entrada para obtener una salida que se compara con los objetivos reales, de esta forma se puede obtener la calidad del modelo. Es necesario definir un umbral de aceptación, y en caso de que la calidad obtenida sea inferior a la esperada, el modelo se debe reentrenar con hiperparámetros (definidos antes de iniciar el entrenamiento) diferentes, o incluso utilizar un algoritmo diferente [33].

Debido a que el entrenamiento del clasificador se realiza con un único conjunto de datos, e iterativamente en algunos casos, el modelo puede llegar a dos estados de mala calidad: bajoajuste (*underfitting*) y sobreajuste (*overfitting*). El primero se refiere a un bajo desempeño del modelo con los datos de entrenamiento y de prueba, esto suele deberse a que el algoritmo no encuentra la relación entre los valores de entrada y los objetivos, se mejora entrenando el modelo hasta sobrepasar el umbral de aprobación con los datos de entrenamiento y prueba, o modificando los hiperparámetros. El sobreajuste ocurre cuando

el modelo sólo es capaz de identificar los datos del conjunto de entrenamiento, pero no los de prueba, esto se debe a que memoriza los datos en lugar de generalizar el conocimiento necesario para obtener buenos resultados [32], [34]. En la Figura 34 se observa cómo los modelos óptimos para clasificación o regresión buscan cuál es la tendencia de los datos en lugar de imitarlos completamente, bajo la consideración de que existen datos atípicos, logrando generalizar el conocimiento dado; mientras el sobreajuste se moldea perfectamente a todos los datos, sin considerar que los datos de entrenamiento no son iguales a los de prueba; finalmente, se observa el bajo desempeño cuando el algoritmo tienen un bajoajuste, como se observa en la Figura 34.

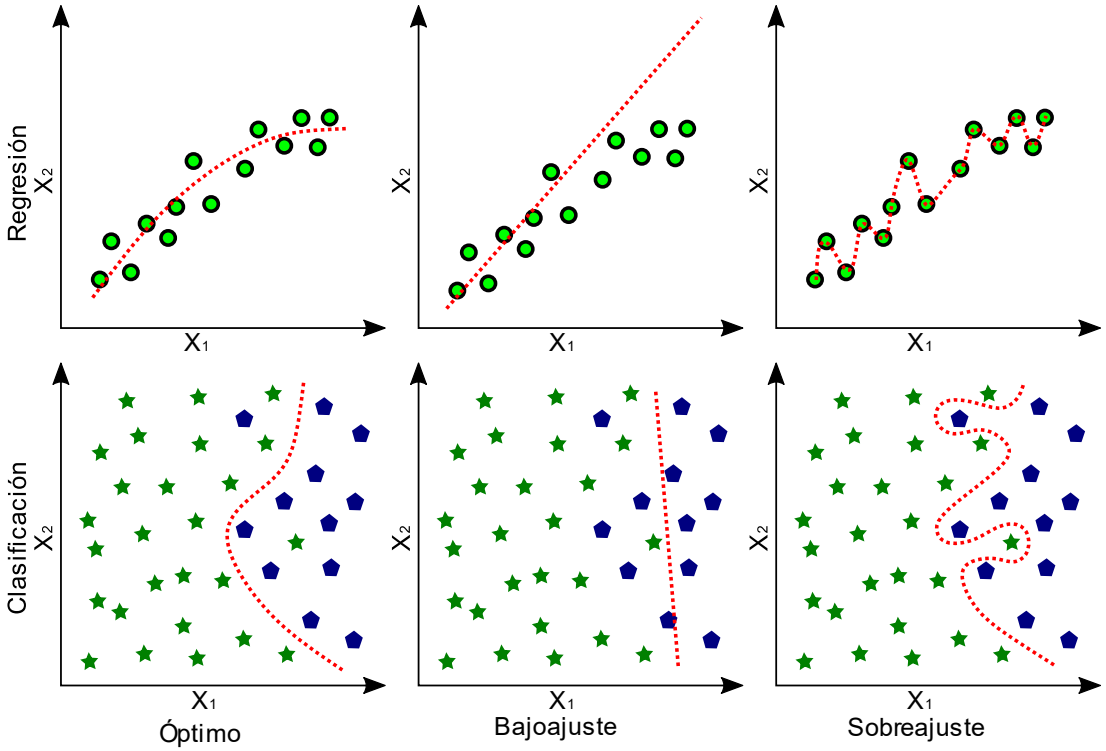


Figura 34. Los modelos óptimos son capaces de clasificar o realizar regresión siguiendo la tendencia de los datos considerando la existencia de datos atípicos. Cuando un bajoajuste ocurre, el modelo no ha aprendido lo suficiente, mientras un modelo sobreajustado memoriza los datos de entrenamiento, por lo que será incapaz de obtener buenos resultados en cualquier otra situación.

Existen diferentes técnicas para evitar el sobreajuste. Por ejemplo, se puede detener el entrenamiento cuando el desempeño en el conjunto de validación disminuye, aunque en el conjunto de entrenamiento siga mejorando, Figura 35.

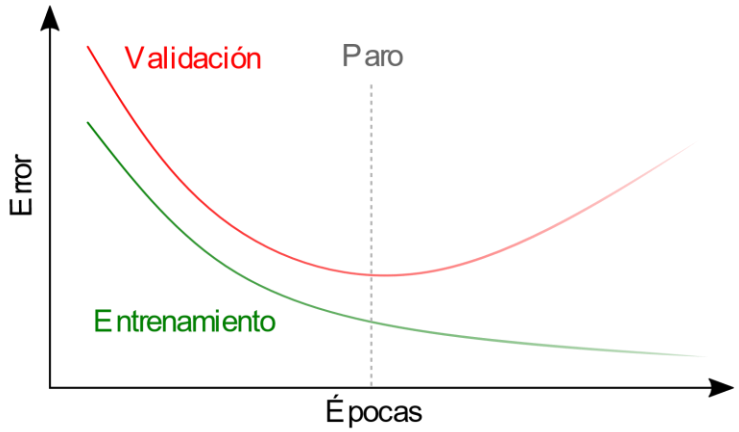


Figura 35. Condición de paro.

Para un entrenamiento más preciso en métodos no iterativos, se recomienda utilizar una validación cruzada, esto implica realizar el experimento varias veces, alternando parte de los datos para entrenamiento y prueba. Se trata de validación cruzada aleatoria cuando los valores elegidos para cada experimento se seleccionan al azar, o validación cruzada de *K* iteraciones cuando se seleccionan de forma ordenada separándolos en *K* conjuntos de prueba. De esta forma, todos los datos históricos disponibles son usados al final de la evaluación para entrenamiento y prueba, pero no al mismo tiempo, con lo que se asegura que no existe un sobreajuste [34], [35].

Dependiendo de cada tipo de problema se emplean diferentes técnicas de aprendizaje, de las que destacan: por refuerzo, supervisado, y no supervisado.

A.1.1 Supervisado

Es útil para entrenar modelos de regresión o clasificación. Utiliza datos históricos ya etiquetados, esto significa que para cada muestra se conocen todas sus características, y la clase o valor al que pertenece (datos etiquetados). El modelo resultante debe ser capaz de generalizar las características de cada clase o valor final, pero sin memorizar cada dato, para poder ser aplicado a cualquier dato nuevo [33], [34].

Durante la fase de entrenamiento, el algoritmo genera un modelo a partir de un conjunto de datos de entrenamiento con la intención de que la salida sea la esperada. En esta etapa, la mayoría de los algoritmos de aprendizaje supervisado minimizan alguna función de costo, que representa el error obtenido en la salida con respecto al valor esperado (la etiqueta de cada valor). Después, el modelo obtenido es utilizado durante la fase de prueba para asignar nuevas etiquetas a los datos de prueba, Figura 36.

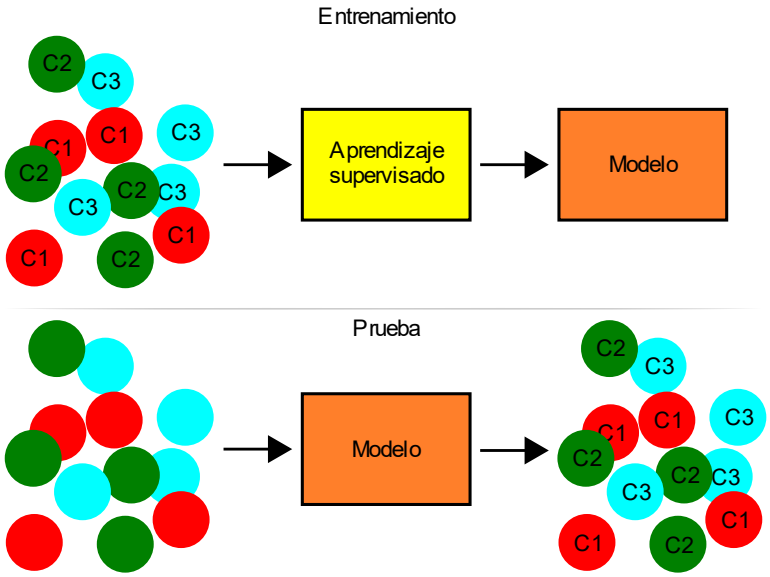


Figura 36. Entrenamiento y prueba de un modelo con aprendizaje supervisado.

El algoritmo debe ser capaz de identificar los patrones implícitos dentro del conjunto de datos para crear una función que genere datos de salida similares a los esperados. Dependiendo del tipo de dato de salida del algoritmo se trata de un problema de regresión o clasificación. Se trata de regresión cuando el resultado esperado puede ser cualquier valor dentro un intervalo (valores continuos); y de clasificación cuando el resultado puede ser sólo un valor ya establecido (discretos, o categóricos), este puede ser de tipo número, binario, o categórico, como se muestra en la Figura 37.

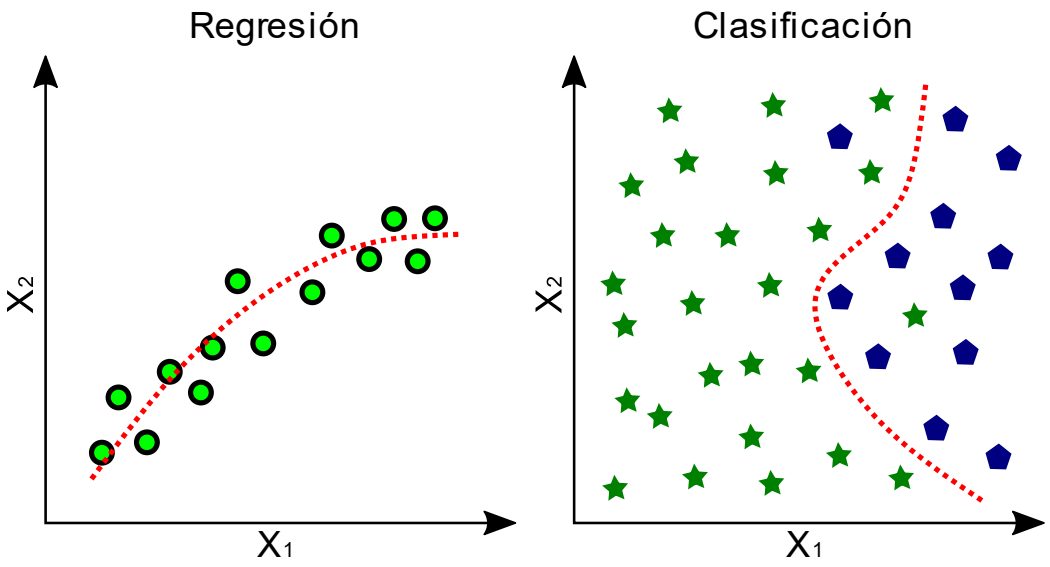


Figura 37. Regresión y clasificación.

A.1.2 No supervisado

En este modelo de aprendizaje, se requiere de datos históricos relativos a un problema a investigar, la computadora se encarga de inferir la relación entre ellos. El modelo aprende por sí mismo, y no requiere de supervisión alguna, ni de datos etiquetados. Aunque es capaz de catalogar nueva información de acuerdo con lo que logra inferir [33], [34].

Para lograr clasificar nueva información, generalmente se asume que se tiene información ya catalogada que sirve como base para realizar clasificaciones futuras, pero no siempre se cuenta con estas etiquetas, o incluso se desconocen los grupos en los que se distribuyen los datos.

El modelado de datos de este tipo de algoritmos puede ser descriptivo, o predictivo. En el primero se resume la información relevante encontrada en los datos, y se muestra lo que ya ha ocurrido; mientras que en el modelado predictivo se sintetizan los datos históricos para predecir lo que podría ocurrir con datos futuros. Este tipo de aprendizaje revela patrones que no se pueden observar a simple vista, siendo de gran utilidad en conjuntos de datos muy grandes, o cuando ni siquiera se conoce que patrón se está buscando.

Durante el entrenamiento, el algoritmo recibe información sin etiquetar, aprende nuevos patrones a partir de esta información, y genera un nuevo modelo. Después, en la fase de prueba, el algoritmo utiliza el modelo para agrupar o generar nuevos datos a partir de datos sin etiquetar, Figura 38.

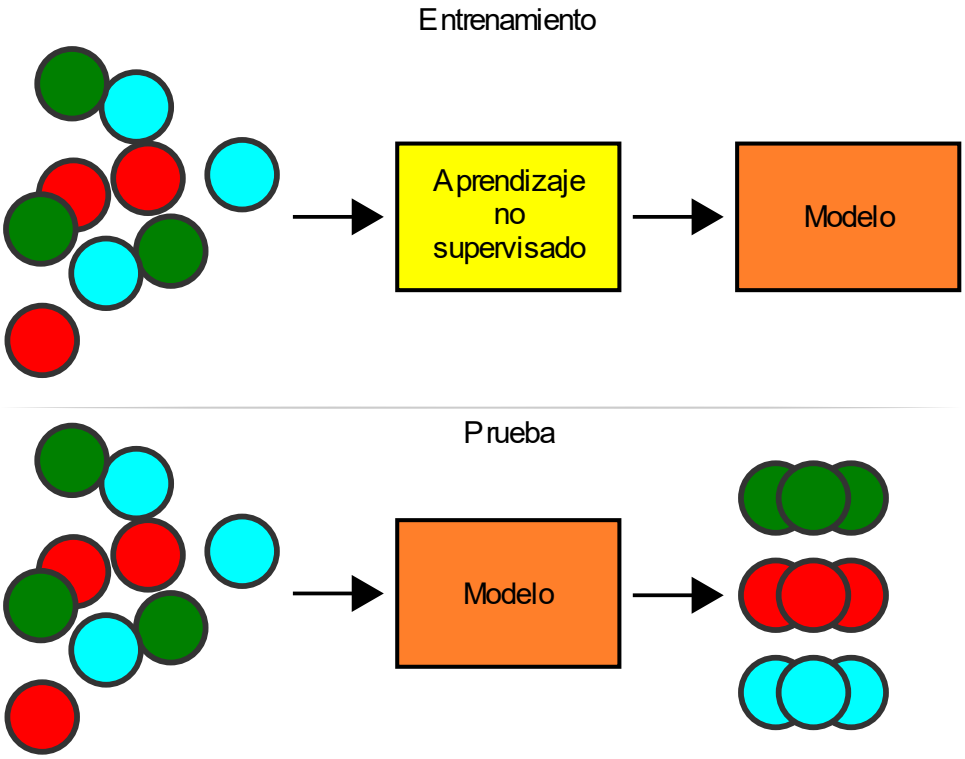


Figura 38. Entrenamiento y prueba de un modelo con aprendizaje no supervisado.

Existen dos categorías de algoritmos de este tipo: Agrupamiento y asociación. Para el primer caso, los datos se agrupan en conjuntos donde los individuos comparten una o varias características, haciéndolos más similares a otros individuos en este grupo que a los individuos de cualquier otro; existen deferentes algoritmos para agrupar información, pero generalmente se basan en medir la similitud entro los elementos del conjunto. Por otro lado, existen algoritmos capaces de descubrir las relaciones entre las variables de un conjunto de datos, al asociar datos de diferentes naturalezas son capaces de sugerir o incluso generar nueva información, Figura 39.

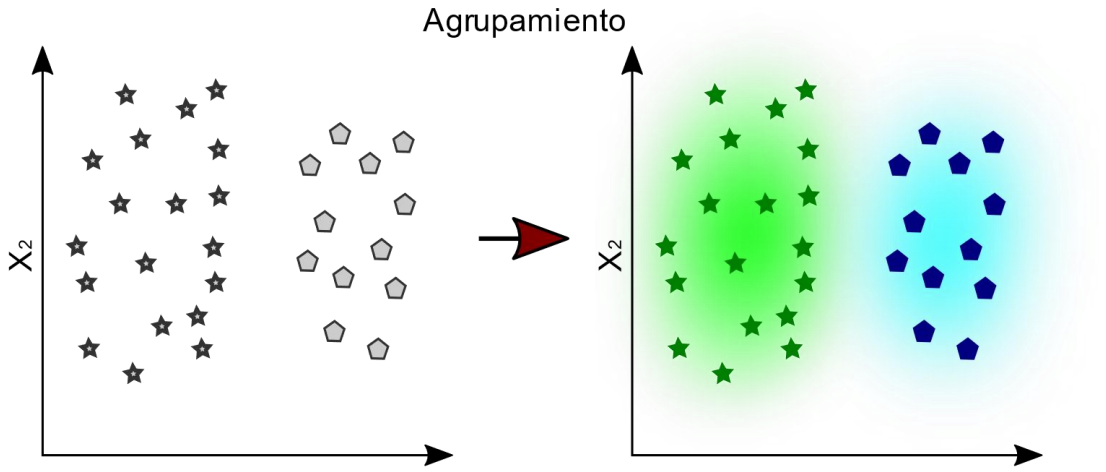


Figura 39. Se agrupan los datos con características similares.

A.1.3 Por refuerzo

Este tipo de algoritmos no requiere de conocimiento previo para aprender, ni de un humano para indicarle las acciones que el modelo debe seguir, sino que aprende de las consecuencias de sus actos. Es un aprendizaje basado en objetivos, donde la experiencia se obtiene al interactuar con el ambiente [33], [34], [36], [37].

Para lograrlo, el algoritmo por medio de un simulador realiza acciones al azar y recibe incentivos o castigos por cada acción. Después de varias iteraciones, el algoritmo aprende reglas para realizar una tarea.

Durante el aprendizaje, el algoritmo por medio de un simulador interactúa con el ambiente de forma iterativa, adaptando sus parámetros con base en los resultados obtenidos, antes de realizar una nueva acción. La recompensa obtenida por cada acción puede ser positiva o negativa, y también es conocida como señal de refuerzo. Después de varias iteraciones en

un proceso de prueba y error, el algoritmo aprende a realizar aquellas acciones que dan una recompensa positiva.

En algunos escenarios la recompensa solo se puede obtener al finalizar una tarea, lo cual podría retrasar el aprendizaje. Por eso se recomienda otorgar una recompensa por cada acción (en tanto sea posible).

El agente puede explorar todas las acciones hasta encontrar una que le dé una buena recompensa, o explotar únicamente la última acción que le dio un buen resultado. Al explorar todas las opciones, el agente podría obtener una recompensa pobre o incluso negativa; mientras que, al únicamente explotar la mejor acción conocida, podría estar evitando aprender una acción incluso mejor. Al no poderse explorar o explotar al mismo tiempo, lo ideal es determinar en qué contextos se debe usar cada comportamiento.

Si bien existen diferentes algoritmos en este tipo de aprendizaje, el agente suele realizar los siguientes pasos:

- Realiza una acción para interactuar con el ambiente.
- Cambia de estado de acuerdo con la acción realizada.
- Recibe una recompensa por su acción.
- Si la acción fue buena, tendrá mayor prioridad para repetirse en situaciones similares. Durante el proceso de entrenamiento se repiten todos los pasos hasta que el modelo no obtenga mejores resultados. Cuando el modelo sea implementado, realizará únicamente las acciones aprendidas más adecuadas para cada situación, pero durante la etapa de aprendizaje se recomienda explorar diferentes acciones en algunas iteraciones, Figura 40.

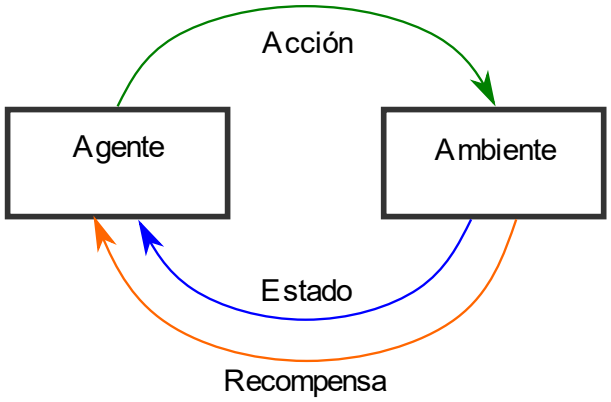


Figura 40. Aprendizaje por refuerzo.

A.2 *k*-NN (*k-nearest neighbours* [*k*-vecinos más cercanos])

k-NN es un algoritmo de clasificación basado en aprendizaje supervisado, que genera un modelo a partir de muestras previamente clasificadas sin necesidad de alguna etapa de entrenamiento más allá de la recolección de datos, ya que todos los cálculos se realizan en tiempo de ejecución. Este método permite identificar dos o más clases [38].

Para cada muestra en el conjunto de entrenamiento $(X_1, Y_1), (X_2, Y_2), \dots, (X_n, Y_n)$ se requiere un vector con los valores de las características de cada dato X y la etiqueta de la clase a la que pertenece (Y). Es recomendable normalizar todos los datos para mejorar los resultados; de no ser así, algunas características podrían tener mayor peso en la clasificación, lo cual produciría un sesgo en la búsqueda del modelo óptimo, llevando a resultados pobres o incorrectos.

en el conjunto de entrenamiento. Se seleccionan los *k* vecinos más cercanos resultantes, se puede realizar por medio de un reordenamiento de los vectores del conjunto de entrenamiento $(X_{(1)}, Y_{(1)}), (X_{(2)}, Y_{(2)}), \dots, (X_{(n)}, Y_{(n)})$, donde la distancia d entre el nuevo vector a clasificar y cada elemento es menor al inicio de la lista, $||X_{(1)} - X|| \leq ||X_{(2)} - X|| \leq \dots \leq ||X_{(n)} - X||$. Por medio de una votación se encuentra la clase que corresponde a más elementos en este subconjunto, y se asigna la etiqueta de la clase predominante al nuevo elemento. Opcionalmente, se puede agregar el nuevo elemento clasificado al conjunto de entrenamiento para futuras clasificaciones. En la Figura 41 se puede observar cómo la cantidad de vecinos k influye en el resultado de la clasificación; cuando $k = 3$ la muestra se asigna a la clase roja, mientras con $k = 5$ se asigna a la clase azul.

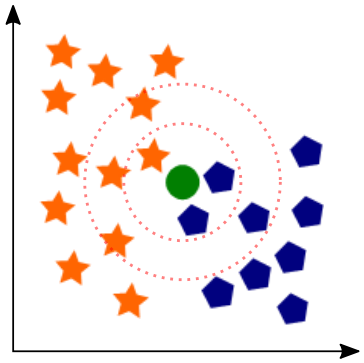


Figura 41. Los resultados de la clasificación pueden variar si se modifica el número de vecinos *k*.

El número de vecinos más cercanos a considerarse (*k*) requiere ser definido a priori, al ser un algoritmo de búsqueda local, este parámetro define cuán amplia será la búsqueda. Otro aspecto que se debe considerar para mejorar la clasificación es la función con la que se define la distancia d entre los elementos como [34]:

$$|| \underset{AB}{\rightarrow} ||_p = \sqrt[p]{(b_1 - a_1)^p + (b_2 - a_2)^p + \cdots + (b_m - a_m)^p} \quad (6)$$

Siendo la más común la distancia euclidiana, Figura 42:

$$\left|\left| \underset{AB}{\rightarrow} \right|\right|_2 = \sqrt[2]{(b_1 - a_1)^2 + (b_2 - a_2)^2 + \cdots + (b_m - a_m)^2} \quad (7)$$

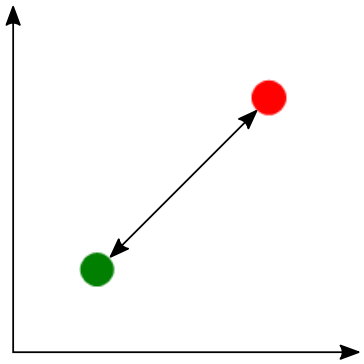


Figura 42. Distancia euclidiana entre dos puntos.

Existen otras funciones como la distancia *city-block*, Figura 43:

$$\left|\left| \underset{AB}{\rightarrow} \right|\right|_1 = |b_1 - a_1| + |b_2 - a_2| + \cdots + |b_m - a_m| \quad (8)$$

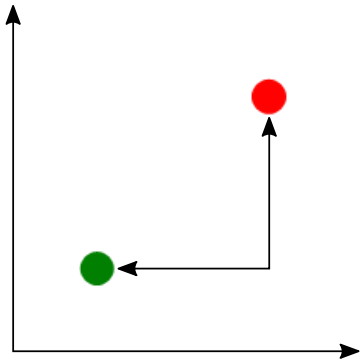


Figura 43. Distancia *city-block* entre dos puntos

En algunos casos es posible aplicar funciones de similitud en lugar de distancia, donde se busca la similitud en la dirección vectorial de los datos sin importar su magnitud. Tal es el caso de la similitud del coseno, Figura 44:

$$Similitud\ (A,B) = \cos(\theta) = \frac{A \cdot B}{||A||\,||B|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\ \sqrt{\sum_{i=1}^{n} B_i^2}} \quad (9)$$

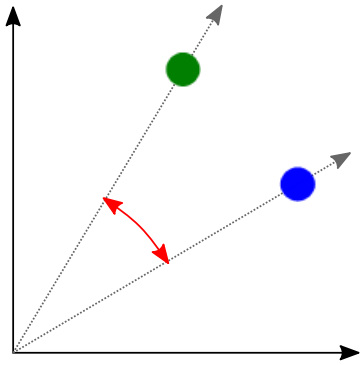
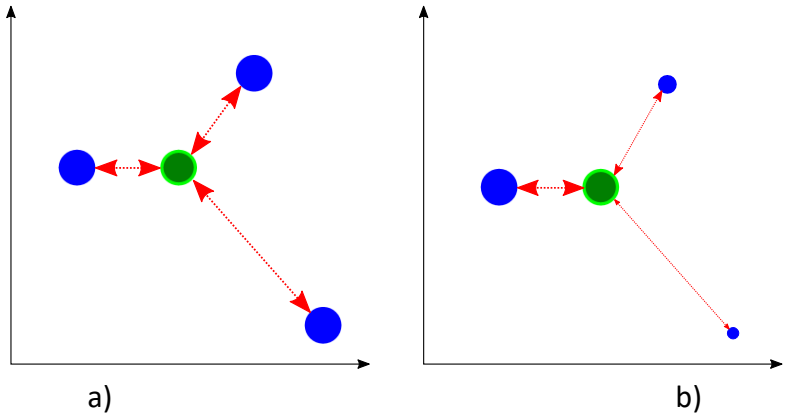


Figura 44. Similitud del coseno.

También hay funciones con algunas modificaciones como la aplicación de pesos diferentes $w = 1/i$ a cada elemento, donde el vecino más cercano $i = 1$ tiene mayor importancia en la votación que el *k*-ésimo vecino $i = k$, Figura 45:

$$d_i = w_i * || \underset{X_i X}{\longrightarrow} || \quad (10)$$



a) b)

Figura 45. Distancia ponderada entre puntos. En a) todos los puntos tienen el mismo peso, en b) los puntos más cercanos tienen mayor peso que los más lejanos.

A.3 SVM (*Support Vector Machine*)

Las máquinas de vectores de soporte son técnicas de aprendizaje supervisado que aprenden a clasificar un conjunto de datos pertenecientes únicamente a dos clases. En caso de contarse con más de dos clases, es necesario entrenar varias SVM (una para cada clase). Este algoritmo de clasificación requiere que los datos sean linealmente separables ya que se busca un vector de soporte óptimo que divida los datos en dos clases; en caso de ser un conjunto no linealmente separable, es posible cambiar la dimensionalidad de los datos por medio de una función (*kernel*) a una donde sí se logre esta característica [32], [34], [39].

El entrenamiento del algoritmo requiere de dos fases: especificar el *kernel* que se utilizará para incrementar la dimensionalidad de los datos, y maximizar el margen entre las clases para encontrar el hiperplano óptimo.

Es necesario elegir a priori la función *kernel* que se utilizará para el entrenamiento del algoritmo. Aunque la función lineal es la que se usa por defecto, existen otras funciones populares como la de gauss, polinomial, y la sigmoidal, Figura 46.

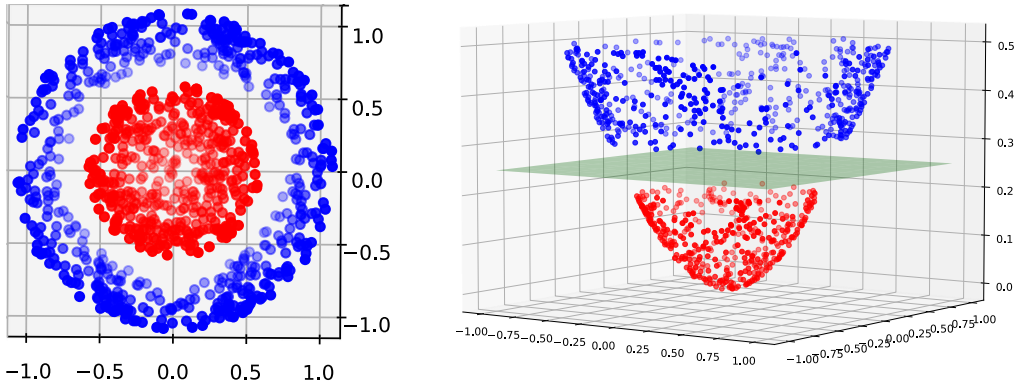


Figura 46. Se aplica una función a un conjunto no linealmente separable para poder ser separado en un espacio diferente.

Es posible generar hiperplanos infinitos dentro del conjunto de datos, pero el óptimo se encuentra entre los márgenes más alejados que dividen ambas clases. Al ser sólo estos márgenes los necesarios para encontrar el hiperplano óptimo, el resto de los datos se vuelven poco relevantes al momento del entrenamiento, como se observa en la Figura 47.

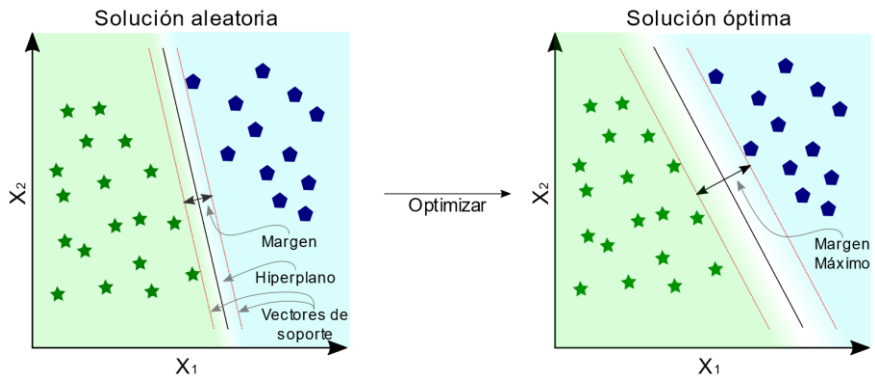


Figura 47. Existen diferentes soluciones que logran separa el conjunto, el algoritmo busca la solución óptima.

En algunos conjuntos de datos, pueden existir datos pertenecientes a una clase dentro de la otra, esto podría significar que el problema no es linealmente separable, o que existen datos atípicos irrelevantes que podrían ignorarse. Dependiendo de la forma en la que se resuelva el problema de optimización, se puede tener un margen duro o suave, el margen duro no permite que existan datos de una clase dentro de la otra; mientras un margen suave permite que algunas muestras en el conjunto de entrenamiento existan en el lado incorrecto, Figura 48.

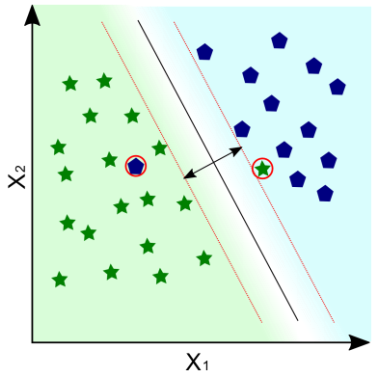


Figura 48. Se toma en consideración la existencia de datos atípicos.

Finalmente, el entrenamiento requiere de resolver un problema de optimización donde se maximiza el margen entre las clases para encontrar el hiperplano correcto. A diferencia de *k*-nn, este algoritmo realiza una búsqueda global y no local, con lo que se previene el sobre entrenamiento.

A.4 ANN (*Artificial Neural Networks*)

Están basadas en el comportamiento de las neuronas humanas, donde las neuronas crean puentes de comunicación (sinapsis) por medio de dendritas (entrada) y axones (salida), la fuerza de la sinapsis se ve afectada por factores externos, ya que cada neurona se activa o no en función del estímulo que recibe, y a su vez, la señal de salida que se genera se transmite a las neuronas posteriores, este proceso resulta en aprendizaje [32], [33], [40], Figura 49.

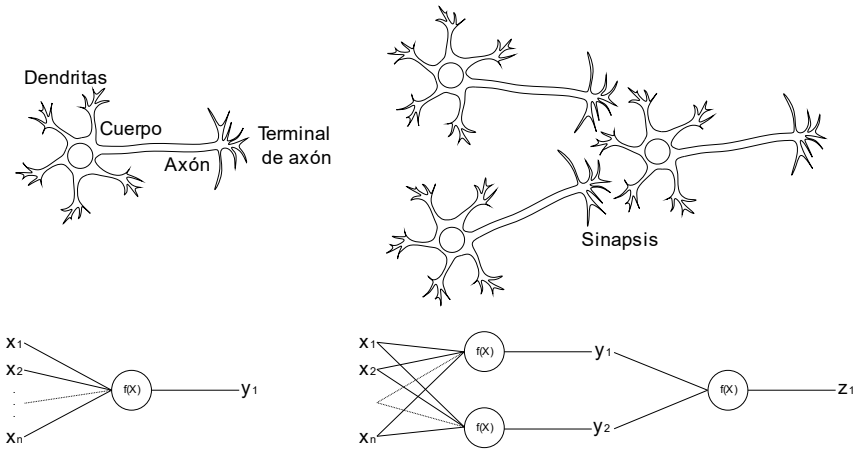


Figura 49. Las neuronas artificiales y redes neuronales están basadas en neuronas y redes neuronales reales.

Esto se simula asignando un vector de pesos diferente a cada neurona capaz de amplificar o invertir los valores de entrada y enviar un nuevo valor a las capas siguientes, a cada peso le corresponde una de las entradas; y como salida se tiene la suma resultante de multiplicar estas entras y pesos, y en algunos casos, de un *bias* para ajustar la salida de la neurona y asegurar que, aunque todos los valores de entrada sean 0 se tenga algún tipo de activación en la salida, Figura 50.

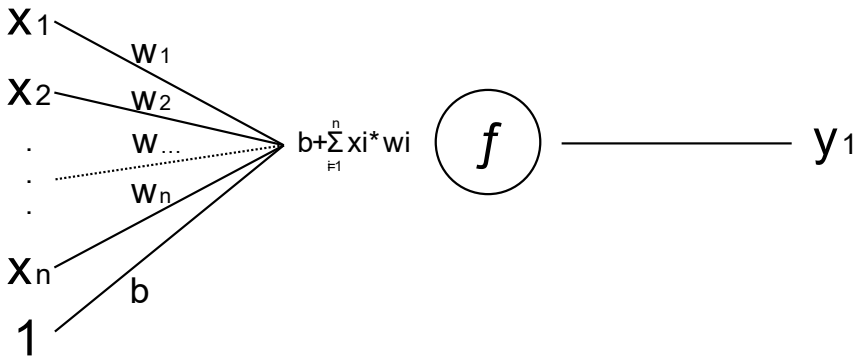


Figura 50. Neurona artificial.

El entrenamiento de este algoritmo busca determinar los pesos de las neuronas para lograr una correcta clasificación o regresión. Cada neurona se asigna a una capa. Existen diferentes tipos de redes neuronales, pero generalmente contienen las siguientes capas, Figura 51:

- Capa de entrada.
- Capas ocultas.
- Capa de salida.

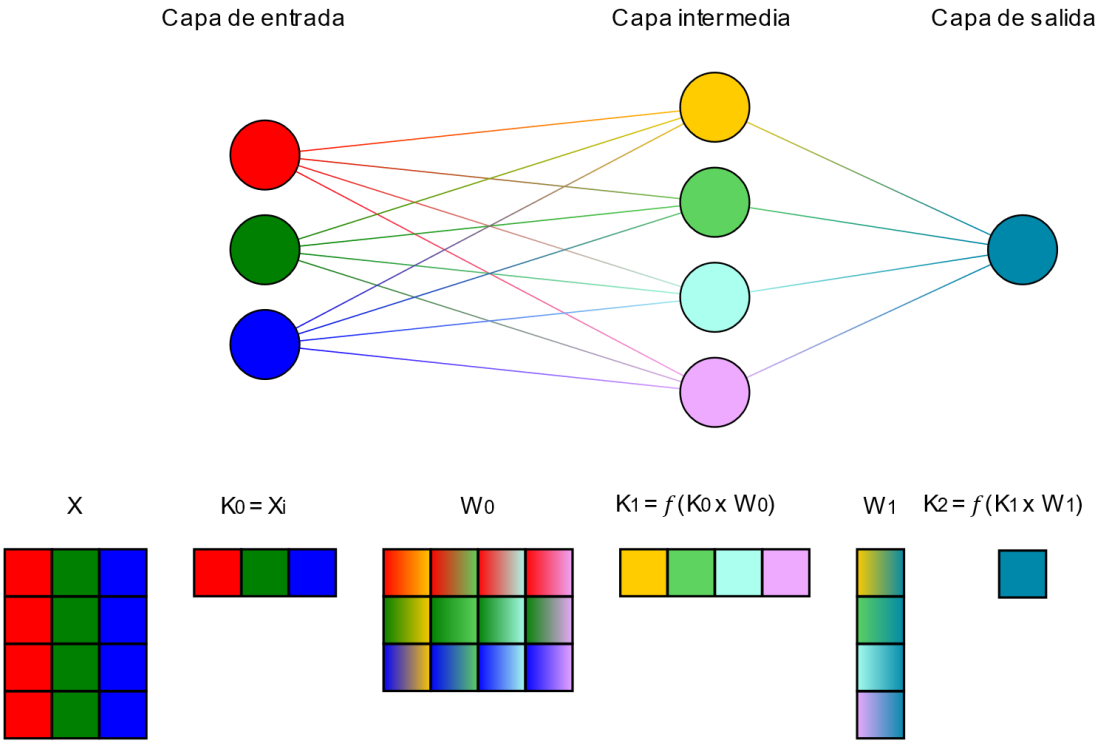


Figura 51. Red neuronal como multiplicación de matrices.

Es posible incluir varias capas ocultas dependiendo de la dificultad del problema, y considerando que la salida de cada capa es la entrada de la siguiente (en el modelo *feed forward*), cuando todas las neuronas de una capa están conectadas a todas las neuronas de la siguiente capa, se conocen como capas totalmente conectadas (*Fully Connected Layers*), Figura 52.

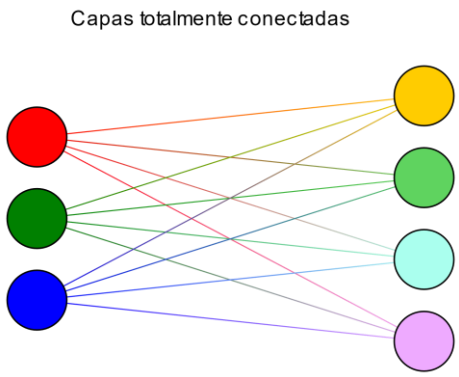


Figura 52. Capas totalmente conectadas.

A.4.1 Capa de entrada

Los datos en esta capa son los datos de entrada del algoritmo, en esta capa no se requiere de alguna modificación en los datos, pero es recomendable que los datos estén normalizados (al igual que en la mayoría de los algoritmos de clasificación).

Los datos de entrada se encuentran en un único vector de tamaño $1 \times n$, donde cada valor corresponde a cada uno de los n atributos de la muestra, así que los datos siempre deben estar ordenados bajo el mismo el formato, donde cada uno de los n valores puede verse como una neurona que no realiza otra función más que traspasar el valor de entrada a la siguiente capa sin modificación alguna, Figura 53.

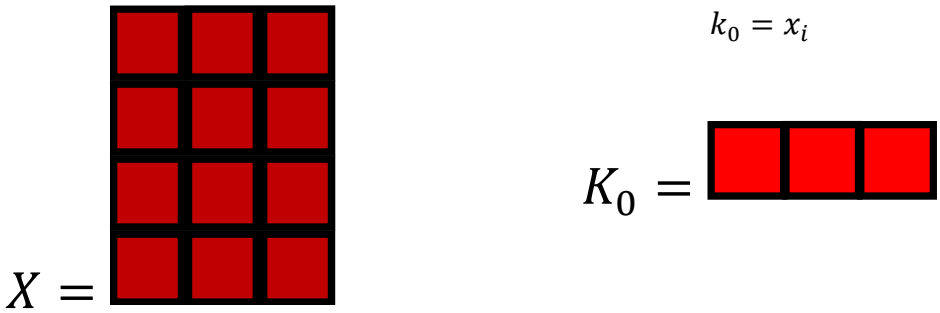


Figura 53. Se toma una muestra a la del conjunto de datos.

A.4.2 Capas ocultas

En estas capas el algoritmo aprende a identificar algunas características presentes en los datos de entrada, generalmente se representa como un conjunto de neuronas apiladas, donde cada neurona tiene asignado un peso correspondiente para cada una de las salidas de la capa anterior, y existe una única salida para cada neurona. Para lograrlo, se aplica una transformación a los datos, donde cada peso exalta los patrones encontrados por cada

neurona. Su funcionamiento es similar a la evaluación de una variable en un sistema de ecuaciones, aplicando la función lineal $Y = XW + b$ sobre el vector de entrada X, donde W y b corresponden a los pesos aprendidos por las neuronas, recordando que W es la pendiente de la recta y b es un factor de corrección. Y que el tamaño de la matriz W es de $n \times m$, donde n es el número de características en la capa de entrada (o en la capa anterior), y m es la cantidad de neuronas en esta capa, Figura 54.

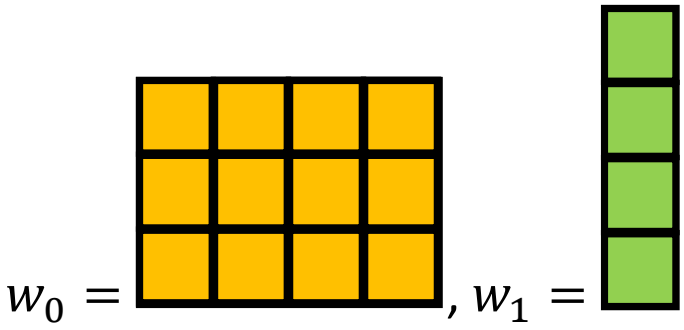


Figura 54. Matrices de pesos.

Es posible utilizar el resultado de aplicar la función lineal como entrada de la siguiente capa, pero el resultado de aplicar varias transformaciones lineales de forma anidada es equivalente a aplicar una única función lineal. Por esto se recomienda aplicar una función no lineal a cada neurona entre cada capa (funciones de activación), las más comunes son: sigmoidal, tangente hiperbólico, con umbral, y de rectificación, como se observan en la Figura 55.

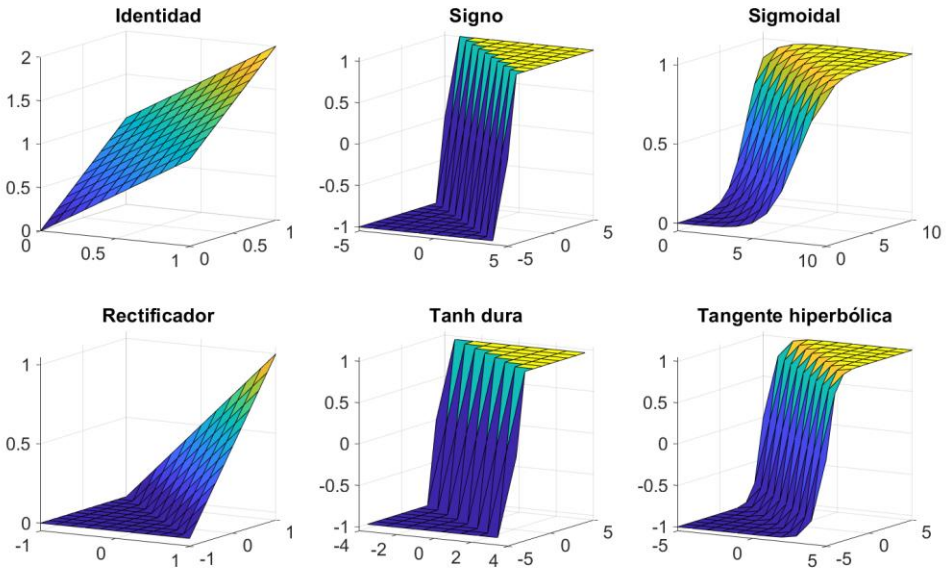


Figura 55. Funciones de activación.

A.4.3 Capa de salida

Esta capa se encarga de realizar la clasificación final, tiene un funcionamiento similar al de la capa oculta. Dependiendo del formato que se desee, los datos pueden representarse como valores continuos (para regresión), o discretos en el rango preferente de 0 a 1, en caso de contarse con más de 2 clases, se puede contar un vector de salida donde cada valor representa la probabilidad de que los datos de entrada correspondan a cada clase, Figura 56.

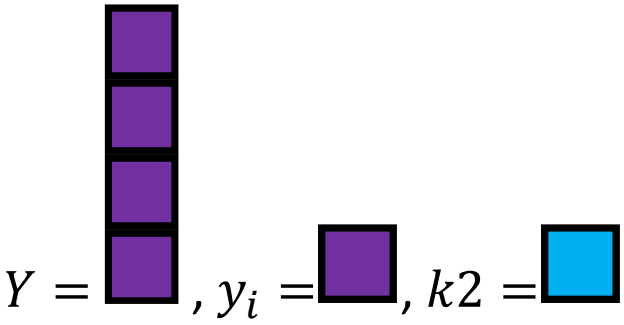


Figura 56. Valores de la capa de salida.

A.4.4 Aprendizaje superficial (de una capa) y aprendizaje profundo (multicapa)

Dependiendo de la cantidad de capas en una red neuronal, se pueden catalogar como aprendizaje superficial (de una capa) y aprendizaje profundo (multicapa). Al compararse la capacidad de aprendizaje de una red neuronal monocapa y una svm, se encuentran ciertas similitudes. Si bien, algoritmos como las svm logran clasificar conjuntos linealmente separables al igual que una neurona, generalmente los problemas de clasificación no cuentan con esta característica y es necesario utilizar un kernel que modifique las dimensiones de los datos, y de forma similar se aplica una función de activación a una neurona. Pero incluso bajo estas circunstancias, podrían darse el caso de no encontrarse las dimensiones donde la separación lineal sea posible.

Debido a que no siempre se puede generalizar el aprendizaje con sólo una función, en el aprendizaje profundo no se intenta modelar todo el conocimiento en una sola capa, si no en varias, donde cada una identifica algún patrón que incluso puede compartirse entre varias clases. Posteriormente, se aprende a identificar cuáles son las combinaciones específicas de estos patrones encontrados pertenecientes a cada clase.

Los modelos que agrupan el conocimiento en más de 3 capas se conocen como de aprendizaje profundo, ya que cada capa se especializa en identificar patrones particulares, y las salidas de cada capa sirven de entrada a una nueva capa que tiene un funcionamiento similar. De esta forma, se pueden apilar tantas capas como sea necesario, pasando del conocimiento particular a uno general.

Se tiene como ventaja que se pueden separar varias clases con un mismo entrenamiento, y los datos no necesitan ser linealmente separables ya que las funciones resultantes son más flexibles que los kernels usados en svm. Pero al generalizar el conocimiento partiendo de una búsqueda local, se corre el riesgo de un sobre-entrenamiento; esto es, memorizar las muestras del conjunto de entrenamiento en lugar de aprender a reconocer los patrones existentes.

A.4.5 Entrenamiento

El proceso de propagación hacia adelante es necesario durante el entrenamiento de la red, pero también corresponde a la evaluación de la red cuando es implementada. Consiste en ingresar valores de entrada y obtener una predicción. En la capa de entrada los valores únicamente se trasladan a la siguiente sin cambio alguno; en las capas intermedias se toman los valores intermedios y se realizan las multiplicaciones, sumas y aplicaciones de funciones de activación correspondientes. Finalmente, se obtiene un valor resultante de la capa de salida, Figura 57 y Figura 58.

$$k_1 = f(k_0 \cdot w_0) \qquad (11)$$

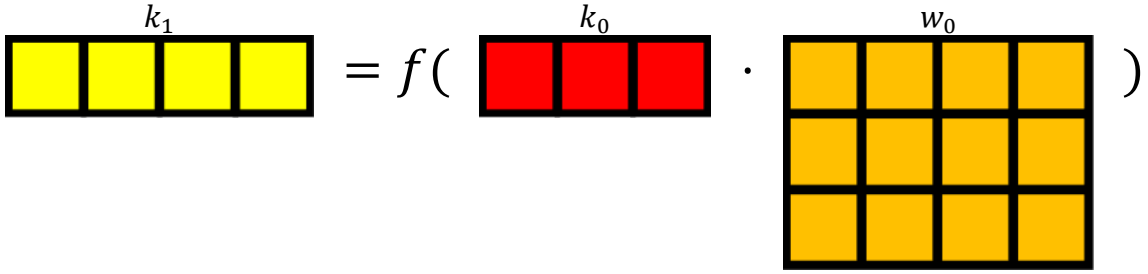


Figura 57. Propagación hacia adelante.

$$k_2 = f(k_1 \cdot w_1) \qquad (12)$$

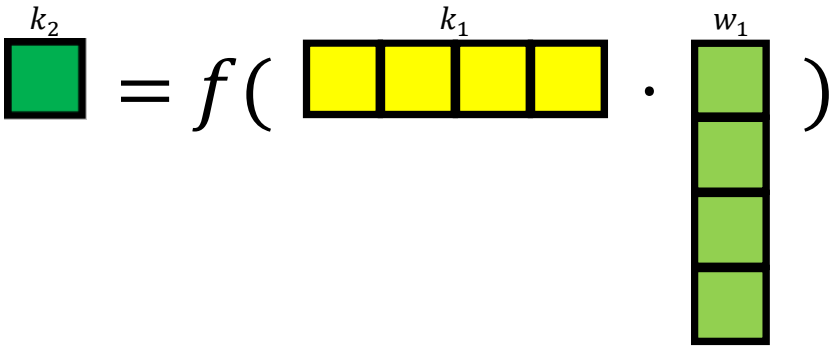


Figura 58. Propagación hacia adelante.

Durante el entrenamiento, este valor resultante es necesario para reajustar los pesos de las capas intermedias en un proceso iterativo con un amplio conjunto de entrenamiento. Los pesos iniciales son valores al azar, así que los primeros resultados suelen ser incorrectos. Se calcula el error de la red $E(X) = (y - \hat{y})$ al comparar el valor resultante $\hat{y}$ con el esperado y, Figura 59. Es importante recordar que se trata de un algoritmo de aprendizaje supervisado, por lo tanto, se tienen las etiquetas reales de los datos de entrenamiento, y.

$$k_2^{error} = yi \; - \; k_2 \tag{13}$$

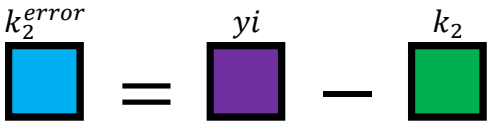


Figura 59. Cálculo del error.

Cuando la red es multicapa, es necesario aplicar funciones de activación hacia adelante en cada capa, y durante la propagación hacia atrás es necesario calcular la afección del error resultante para cada capa. Con base en el error total de la evaluación hacia adelante, se calcula la derivada del error con respecto a los pesos de la capa anterior, Figura 60 a Figura 62. Como el error se propaga hacia atrás, es necesario aplicar la regla de la cadena (Figura 63), donde la derivada de la capa n depende de la derivada de la capa inmediata siguiente $n + 1$, Figura 60. El proceso se replica hasta llegar a la primera capa oculta, Figura 61 y Figura 62. Este proceso se repite para cada valor en el conjunto de entrenamiento iterativamente hasta que el error se encuentra un mínimo local.

$$k_2\Delta = k_2^{error} \odot f'(k_2) \tag{14}$$

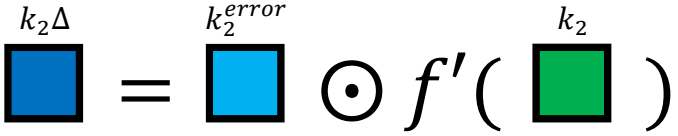


Figura 60. Replicación del error.

$$k_1^{error} = k_2\Delta \cdot w_1^t \quad (15)$$

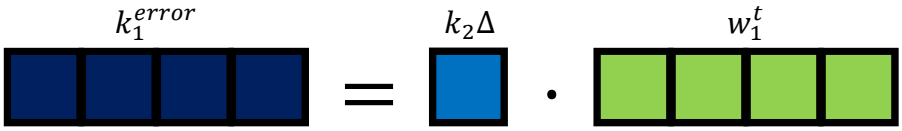


Figura 61. Replicación del error.

$$k_1\Delta = k_1^{error} \odot f'(k_1) \quad (16)$$

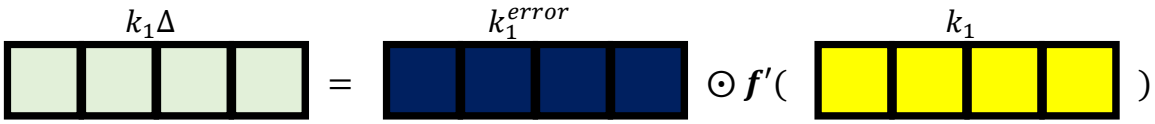


Figura 62. Replicación del error.

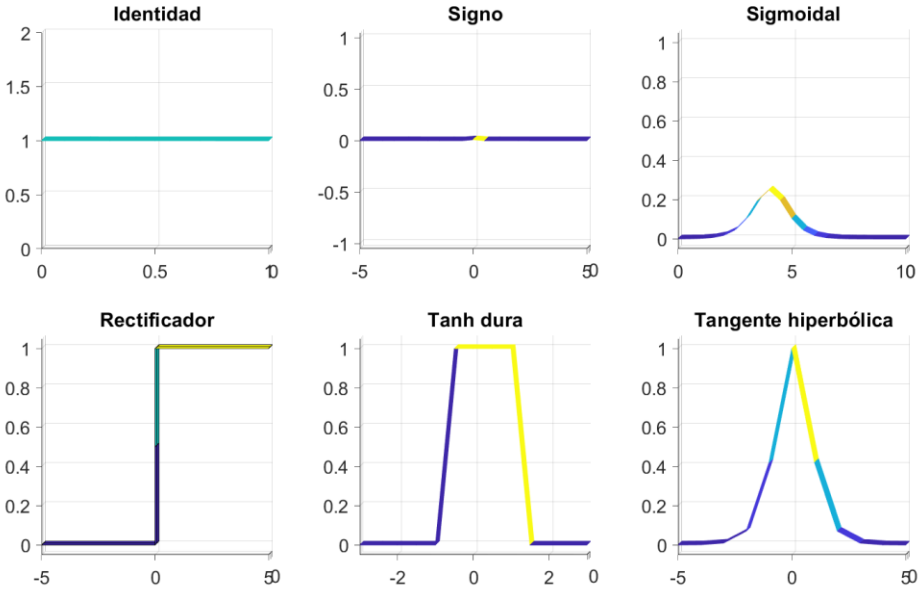


Figura 63. Derivadas de diferentes funciones de activación.

Para actualizar los pesos de la red se realiza un proceso de propagación hacia atrás, donde el error que se obtuvo mantiene el signo resultante ya que contiene la magnitud y dirección

en la que se corregirán los parámetros de la red en un proceso de minimización del error. En la forma más simple del algoritmo, se puede observar que el peso actualizado corresponde a un descenso del gradiente, $W \Leftarrow W + \alpha E(X)X$, donde la velocidad con la que se desciende o minimiza el error corresponde al hiperparámetro de aprendizaje α, y debe ser definido a priori, Figura 65 y Figura 66; si α es muy alto podría evitar que se logre un aprendizaje óptimo, pero si es demasiado bajo podría retardarse el entrenamiento, como se muestra en la Figura 64.

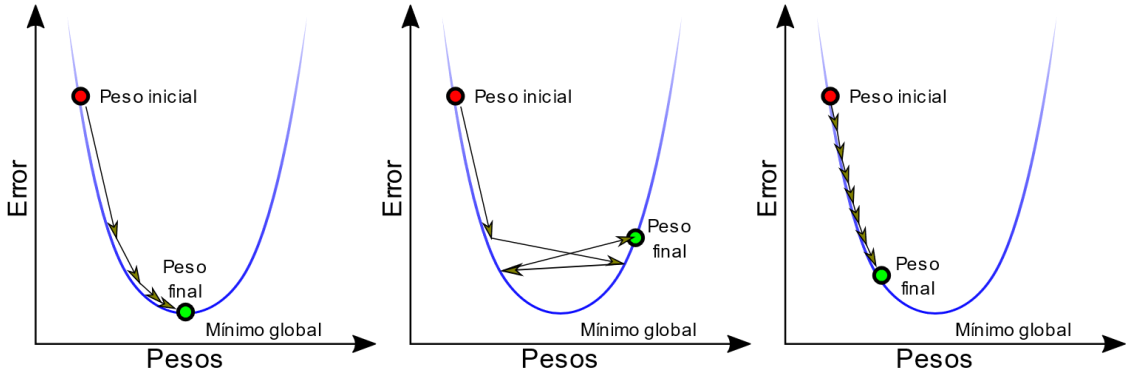


Figura 64. Cuando el parámetro de aprendizaje es el correcto, es posible encontrar el óptimo. Con un parámetro muy alto se podría alejar del óptimo. Un parámetro muy bajo ralentiza el proceso.

$$w_1 = w_1 + \alpha k_1^t \cdot k_2 \Delta \qquad (17)$$

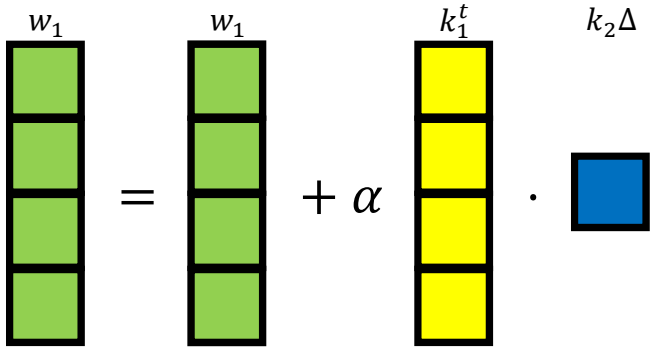


Figura 65. Actualización de pesos.

$$w_0 = w_0 + \alpha k_0^t \cdot k_1 \Delta \qquad (18)$$

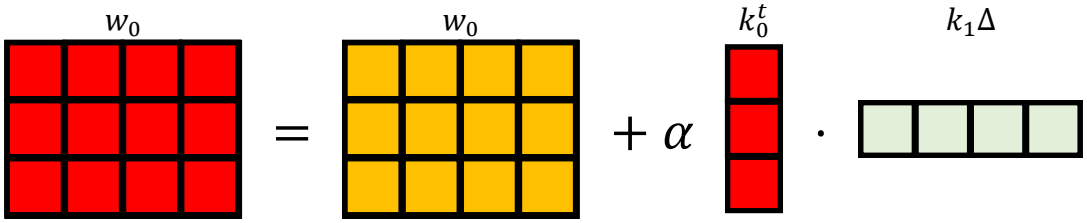


Figura 66. Actualización de pesos.

A.5 CNN (*Convolutional Neural Network*)

Una CNN es un tipo de red neuronal capaz de identificar objetos en imágenes con mejores resultados que los obtenidos por otros algoritmos similares, ya que distingue patrones similares a los usados durante el entrenamiento sin requerir que sean exactamente iguales en tamaño, posición, o rotación. Esto gracias a que aprende a detectar las características que componen un objeto en lugar de identificar el patrón completo [32], [33], [40], [41].

En la mayoría de los algoritmos de reconocimiento de objetos se tiene la capacidad de analizar imágenes de un tamaño predefinido (del tamaño esperado del objeto), pero cuando se presenta una nueva escena, generalmente se desconoce la ubicación y tamaño del objeto, que además podría coexistir con otros objetos que el modelo debe ser capaz de identificar, Figura 67. Así que no siempre es posible darle como entrada una escena que sólo contenga el objeto a identificar en las dimensiones exactas. Por esto se requiere analizar la escena por secciones, escaneando la escena completa por medio de ventanas deslizantes, donde el objeto podría o no estar presente en cada sección.

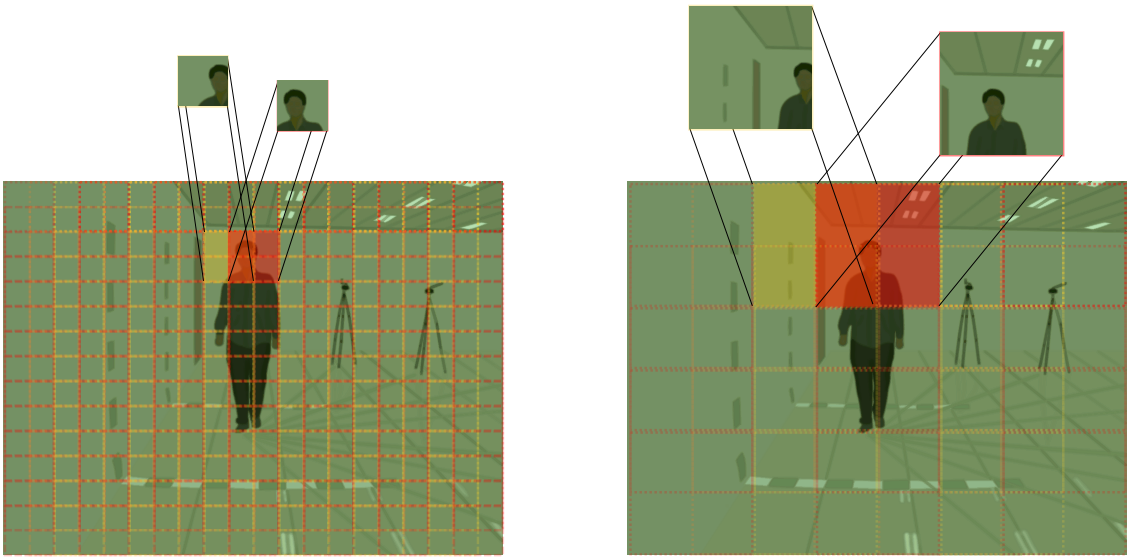


Figura 67. Ventana deslizante con diferentes dimensiones.

Uno de los métodos más populares para la detección de objetos en imágenes ha sido Viola-Jones, aunque no se limita a al reconocimiento de caras, este ha sido su principal implementación debido a su simplicidad y rapidez para detectar patrones básicos en imágenes por medio de filtros HAAR. Estos filtros predefinidos en tamaño y forma consisten en multiplicar los pixeles de una imagen por valores 0 o 1, y sumar los resultados para obtener un valor, este valor será similar en figuras que comparten las mismas características. El algoritmo de Viola-Jones analiza la imagen con una ventana deslizante aplicando inicialmente un filtro HAAR, en caso de que se detecte una coincidencia, se

aplican uno a uno los filtros posteriores sobre la misma ventana, si todos los filtros encuentran la característica en la que se especializan, se considera que en esa sección de la escena se encuentra el objeto buscado, en caso contrario la ventana deslizante continúa el escaneo, como se muestra a continuación en la Figura 68.

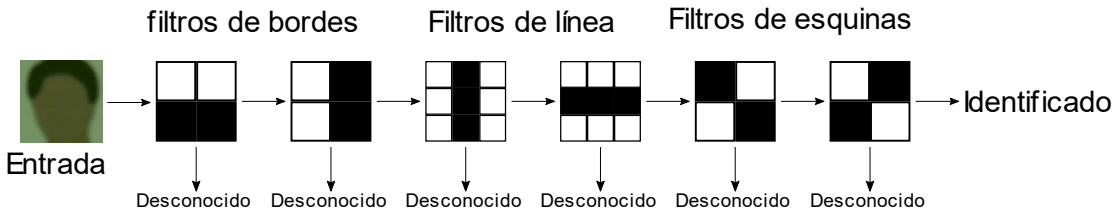


Figura 68. Algoritmo Viola-Jones con diferentes filtros HARR.

Desafortunadamente, el algoritmo Viola-Jones está limitado a que los filtros predefinidos logren coincidir con las formas del objeto buscado, y al ser filtros con muy pocas características, el algoritmo es muy poco flexible para identificar objetos con formas muy específicas, fallando por ejemplo cuando el objeto está rotado. Las CNN comparten la idea básica de analizar la imagen por medio de ventanas deslizantes, y filtrando la imagen en búsqueda de patrones específicos, pero los filtros utilizados por la CNN no son predefinidos y multipropósito, sino que se generan durante el entrenamiento del modelo y son útiles únicamente para identificar los objetos para los que fue entrenada. Además, cuenta con capas donde los filtros iniciales identifican las características más generales de varios objetos, terminando con filtros muy específicos para cada objeto, Figura 69.

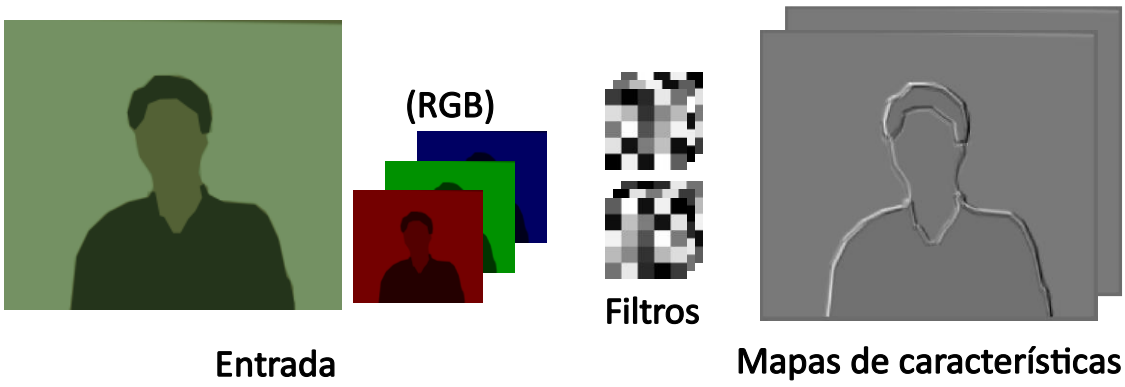


Figura 69. Imagen filtrada.

Este algoritmo es una variante de las Redes Neuronales Artificiales que emplean *Back Propagación*, por lo que la información de entrada fluye hacia la salida, y en caso de estar en entrenamiento, se calcula el error que se obtuvo y con este error se recalculan los pesos

de las capas intermedias iniciando con las capas más cercanas a la salida. Las capas que la componen son las siguientes:

- Capa de entrada.
- Capa convolucional.
- Capa de agrupamiento (*Pooling*).
- Capa de rectificación (*ReLU*).
- Capa totalmente conectada.
- Salida.

Es posible tener múltiples capas intercaladas del tipo convolucional, de agrupamiento y de rectificación, pero siempre en el mismo orden, y con la capa totalmente conectada al final.

A.5.1 *Capa de entrada*

Esta capa tiene un tamaño fijo y no suele ser muy grande, por lo que generalmente no es posible analizar una escena completa, es necesario analizar la imagen seccionadamente a través de ventanas deslizantes; además, cada una de estas secciones puede ser redimensionada al tamaño de esta capa durante el recorrimiento, y puede o no tener un espaciado mayor a 1 entre cada muestra tomada.

Requiere ser de dimensiones $\mathrm{m} \times \mathrm{m} \times \mathrm{r}$, donde m es el alto y ancho de la matriz, y r es el número de canales presentes en la imagen ($r = 3$ en el espacio *RGB*), Figura 70. El tamaño m no está restringido a un valor específico, pero es necesario considerar que debe tener tamaño suficiente para contener el objeto buscado, y que la resolución de la imagen influye en el tiempo de entrenamiento y ejecución del modelo, además de la calidad de los resultados.

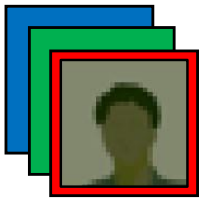


Figura 70. Imagen de entrada.

A.5.2 Capa convolucional

Durante este proceso se analiza si existen partes de la imagen de entrada, o de la capa anterior, que contienen características similares a los filtros. Cada capa aprende a identificar alguna característica específica. Las secciones donde el filtro encuentra similitudes suelen resultar en valores altos.

La convolución se realiza multiplicando el filtro por cada sección de la imagen, los resultados de esta multiplicación valor por valor son sumados, y almacenados ordenadamente en una nueva matriz que sirve de entrada para la siguiente capa, Figura 71.

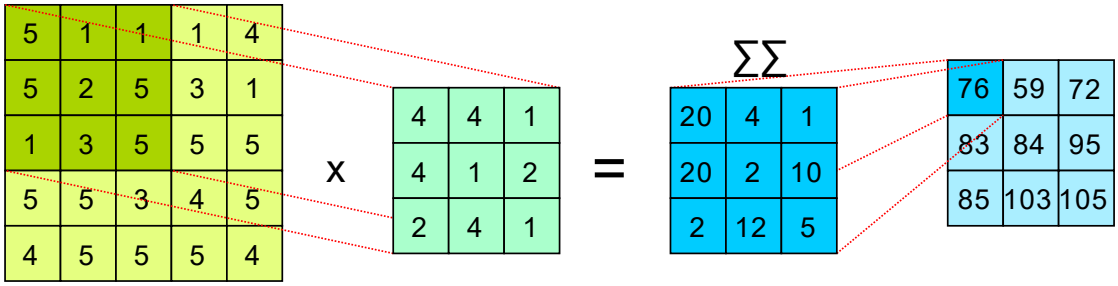


Figura 71. Convolución.

Nótese que de aplicar la convolución directamente a la imagen de entrada, la matriz resultante tendrá una dimensión inferior ($m - n + 1$), debido a que los valores de las orillas no son evaluados tan exhaustivamente como los valores centrales. Esto podría disminuir la calidad de la clasificación final si el objeto buscado no se encuentra centrado en la imagen de entrada. Para evitar que se pierda la información de las orillas, y conservar las dimensiones originales de la capa de entrada (o la capa intermedia anterior), se incrementan sus dimensiones agregando filas y columnas en las orillas, los nuevos espacios necesitan un relleno (*padding*) que puede ser con $0, 1$, o repitiendo los valores existentes más cercanos, Figura 72.

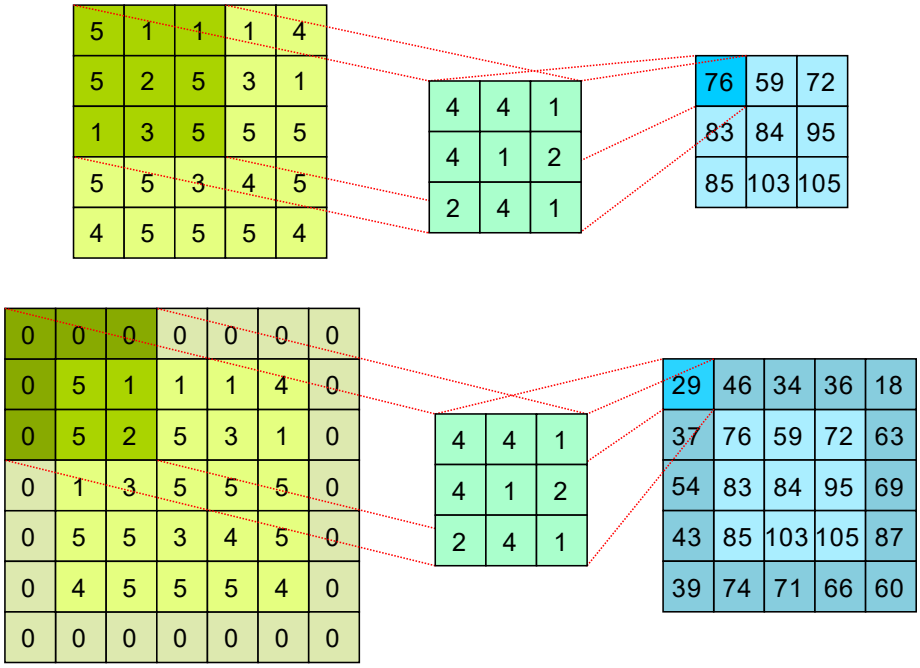


Figura 72. *Padding.*

Contiene k filtros, donde cada uno es de dimensión $n \times n \times r$, r es igual al número de canales en la capa de entrada en caso de ser la primera capa, o del tamaño de k de la capa anterior en caso de ser una dimensión intermedia; además, n es menor a m (o n en la capa anterior en caso de tratarse de una capa intermedia), ya que se realiza un nuevo recorrido en la capa anterior. Se recomienda rellenar la matriz para que después de aplicar los k filtros se obtenga como resultado una nueva matriz de igual tamaño a lo alto y ancho, pero con una mayor profundidad a la original, resultando con una dimensión $n \times n \times k$.

Por ejemplo, si se aplica la convolución a la imagen de entrada con dimensiones $28 \times 28 \times 3$ en el espacio de color RGB ($r = 3$); cada filtro analizará la imagen en todos sus canales, en este proceso se pierden los 3 canales originales, compactándolos en una matriz donde se resalta la existencia de alguna característica específica de tamaño $28 \times 28 \times 1$. Después de filtrarla con k filtros, los resultados se apilan en una matriz de $28 \times 28 \times k$, como se observa en la Figura 73.

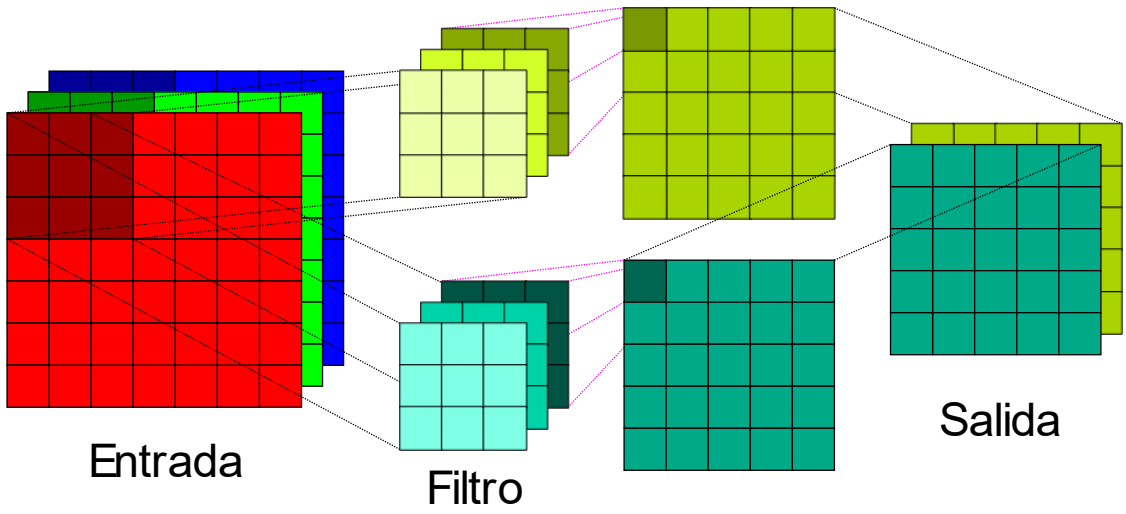


Figura 73. Las dimensiones de la imagen de entrada se modifican después de ser filtrada.

A.5.3 Capa de agrupamiento (*Pooling*)

Las dimensiones de la salida de la capa convolucional suelen incrementarse en cuanto a la profundidad en comparación con la imagen de entrada ($r < k$), pero aún después del filtrado las características que se buscan podrían estar en cualquier sección de la matriz, por lo que es necesario disminuir su tamaño en alto y ancho, manteniendo las porciones de la imagen más significativas.

En la capa de agrupamiento se recorre la matriz resultante con una ventana deslizante, se abstrae cada sección bidimensional en un único valor que se almacena en una nueva matriz

con menores dimensiones en altura y anchura, y con profundidad igual al número de filtros en la matriz anterior (k). Generalmente en este proceso sólo se mantienen los valores mayores en cada recorrido ya que esto mantiene las características principales disminuyendo la resolución de la imagen de entrada o capa anterior, pero es posible utilizar otras funciones como el promedio, o la suma de cada ventana, Figura 74.

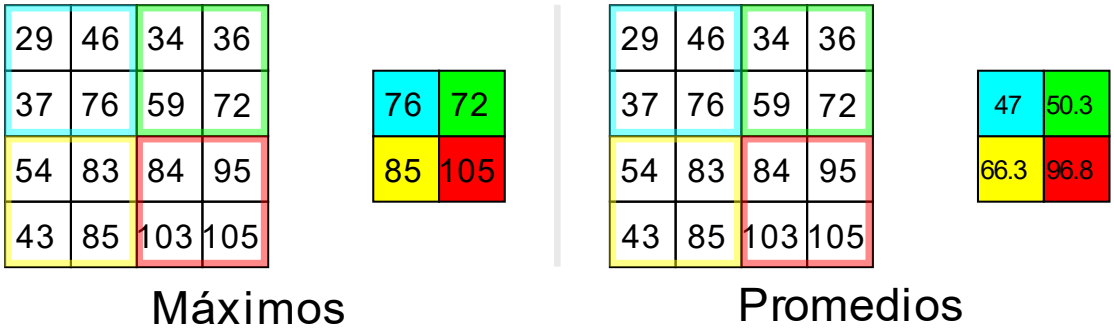


Figura 74. Agrupamiento.

Para este filtro se deben definir el tamaño (n) de la ventana deslizante (al ser cuadrada sus dimensiones son nxn), y el espaciado ($stride$) entre cada corrida, que puede ser de 1 a n pixeles. Entre mayor sea este espaciado, menor será el tamaño de la matriz resultante. Se recomienda un espaciado igual a n para disminuir las dimensiones de la matriz más rápidamente, y no repetir características ya analizadas, Figura 75.

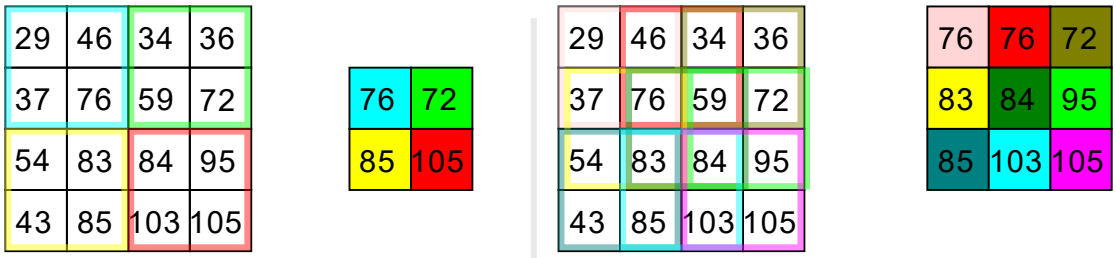


Figura 75. Diferentes espaciados entre ventanas deslizantes.

A.5.4 Capa de rectificación (ReLU)

Esta capa es opcional ya que en caso de que la imagen de entrada esté segmentada, es posible marcar los pixeles que pertenecen al fondo de la imagen con -1; e incluso después del filtrado, podría haber valores negativos que retarden el proceso de aprendizaje de la red. Para evitarlo se rectifican los valores negativos; esto es, cualquier valor negativo debe ser convertido en 0, Figura 76.

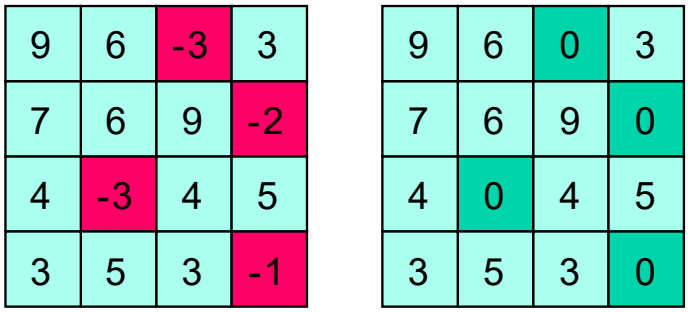


Figura 76. Rectificación (ReLU)

A.5.5 Capa totalmente conectada

Después de pasar por las capas anteriores, es necesario que la imagen de entrada reduzca sus dimensiones en altura y anchura, pero se incremente en profundidad. Esto es, que sus dimensiones pasen de $m \times m \times r$ a $1 \times 1 \times k$, ya que esta capa tiene el mismo funcionamiento que una *ANN*, y requiere de un vector y no de una matriz como entrada. Aunque es posible reducir todas las dimensiones en altura y profundidad a 1 por medio de una capa de agrupamiento, se prefiere mantener algunos de los valores finales en forma de matriz y convertirlos a vector por medio de un aplanado (reordenándolos para generar un único vector), Figura 77.

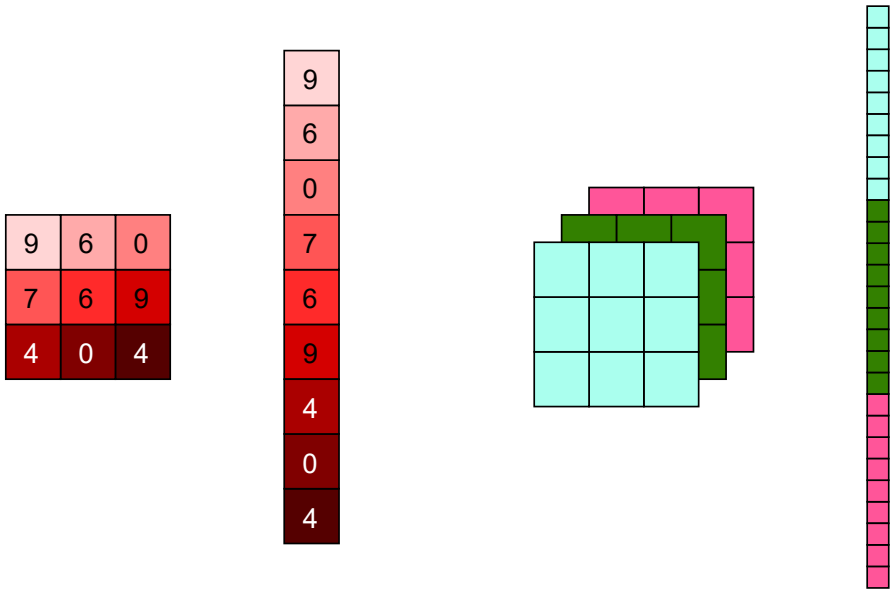


Figura 77. Aplanamiento en una capa totalmente conectada.

Esta capa tiene el mismo comportamiento que una *ANN* de tipo *feed forward*, y realiza la clasificación final de la imagen de entrada, Figura 78.

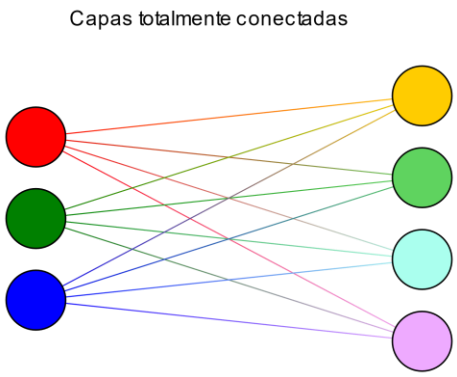


Figura 78. Capa totalmente conectada.

A.5.6 Salida

Después de realizarse la clasificación, se pueden tener una o más salidas que contienen la información del objeto encontrado en la imagen (en caso de que el algoritmo lo haya identificado). El vector de salida contiene valores entre 0 y 1, donde cada valor corresponde a la probabilidad de cada objeto aprendido de aparecer en la imagen de entrada; además se puede programar para que retorne la etiqueta del valor más alto, Figura 79.

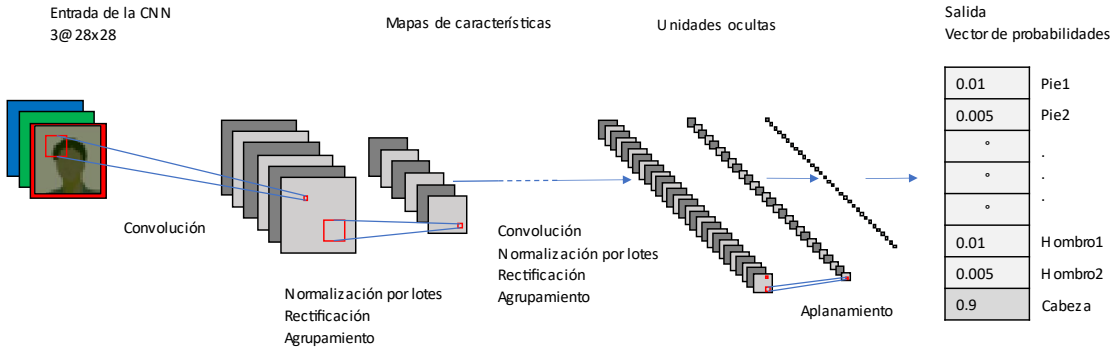


Figura 79. El vector de salida indica la clase que corresponde a la imagen de entrada.

A.5.7 Propagación hacia atrás

El procedimiento para entrenar una CNN es similar al descrito en A.4.5. Debe tenerse en consideración que al igual que cuando se multiplica una matriz x*w obteniendo y, es posible obtener una nueva matriz con las dimensiones w si se multiplica x*y. Cuando se aplica un filtro f a x se obtiene una matriz de tamaño y, por lo que cuando se aplica la convolución de y a x se obtiene una matriz de tamaño f.

Para poder realizar el proceso de propagación hacia atrás se requiere del error $\frac{\partial O}{\partial F}$ de la capa posterior que será replicado hacia las capas de atrás con el cálculo del gradiente local con respecto a la entrada de la capa $\frac{\partial L}{\partial X}$, y con respecto al filtro $\frac{\partial L}{\partial F}$ para actualizarlo, Figura 80.

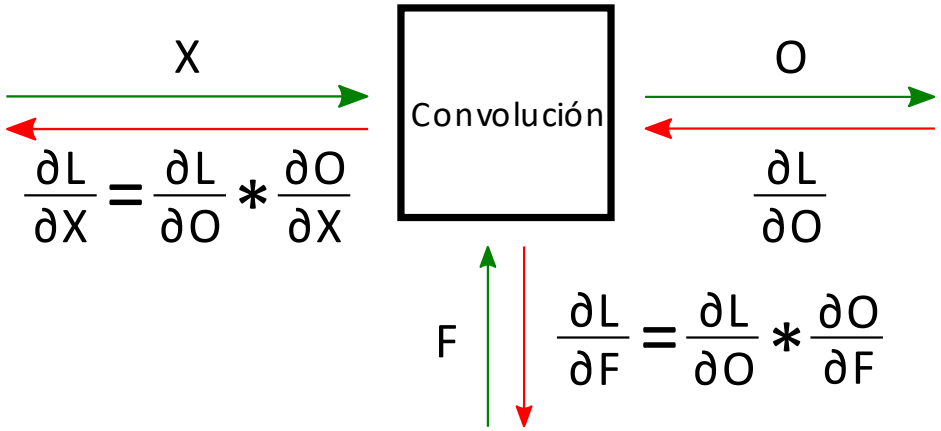


Figura 80. Propagación hacia atrás de una función de convolución.

Para actualizar los valores del Filtro F se calcula $\frac{\partial L}{\partial F} = \frac{\partial L}{\partial O} * \frac{\partial O}{\partial F}$, para cada elemento del filtro se necesita calcular $\frac{\partial L}{\partial F_i} = \sum_{k=1}^{M} \frac{\partial L}{\partial O_k} * \frac{\partial O_k}{\partial F_i}$, donde el gradiente de O está definido como $\frac{\partial O}{\partial F}$=X, Figura 81.

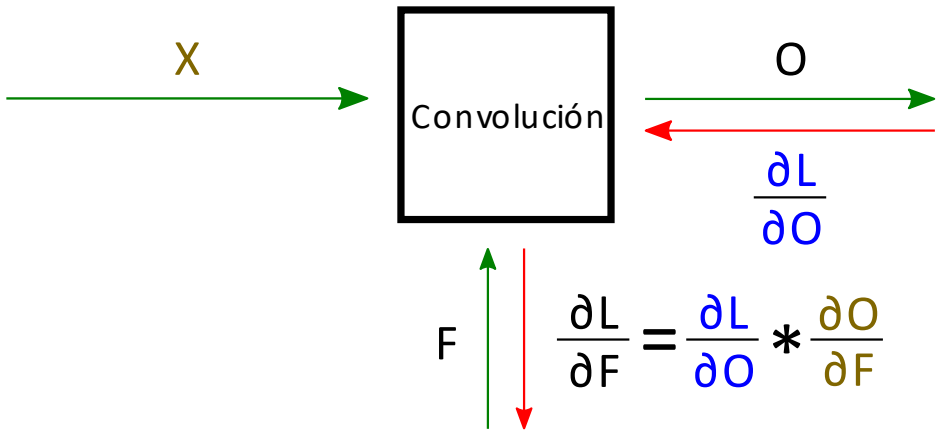


Figura 81. Actualización de los filtros.

Esto es equivalente a aplicar una convolución entre la entrada de la capa actual y el error de la siguiente.

$$\begin{bmatrix} \frac{\partial L}{\partial F_{11}} & \frac{\partial L}{\partial F_{12}} \\ \frac{\partial L}{\partial F_{21}} & \frac{\partial L}{\partial F_{22}} \end{bmatrix} = convolución \left(\begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix}, \begin{bmatrix} \frac{\partial L}{\partial O_{11}} & \frac{\partial L}{\partial O_{12}} \\ \frac{\partial L}{\partial O_{21}} & \frac{\partial L}{\partial O_{22}} \end{bmatrix} \right) \quad (19)$$

$\frac{\partial L}{\partial F}$ $\frac{\partial O}{\partial F}$ $\frac{\partial L}{\partial O}$

Finalmente, el filtro se actualiza de forma similar a la actualización de los pesos en una ANN con:

$$F = F - \alpha \frac{\partial L}{\partial F} \quad (20)$$

Para encontrar el error que se propagará a la capa anterior, se calcula $\frac{\partial L}{\partial X_i} = \sum_{k=1}^{M} \frac{\partial L}{\partial O_k} * \frac{\partial O_k}{\partial X_i}$, donde $\frac{\partial O}{\partial X} = F$, Figura 82.

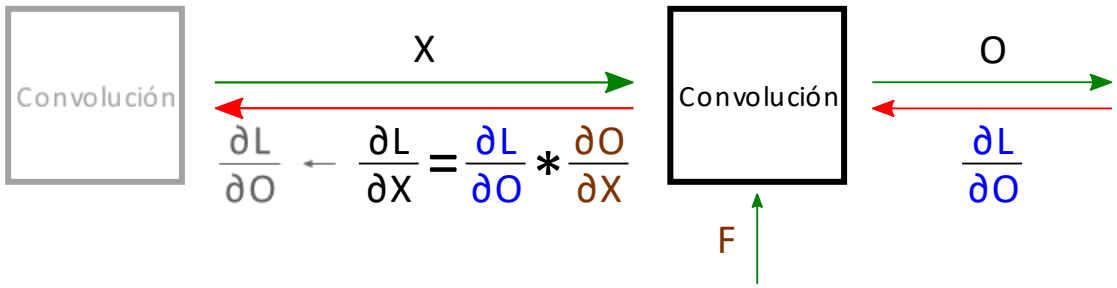


Figura 82. Error en la propagación hacia atrás.

El cálculo de esta operación también se puede realizar por medio de una nueva convolución entre el error que propagó en la capa anterior y el filtro aplicado.

$$\begin{bmatrix} \frac{\partial L}{\partial X_{11}} & \frac{\partial L}{\partial X_{12}} & \frac{\partial L}{\partial X_{13}} \\ \frac{\partial L}{\partial X_{21}} & \frac{\partial L}{\partial X_{22}} & \frac{\partial L}{\partial X_{23}} \\ \frac{\partial L}{\partial X_{31}} & \frac{\partial L}{\partial X_{32}} & \frac{\partial L}{\partial X_{33}} \end{bmatrix} = convolución \left(\begin{bmatrix} F_{22} & F_{21} \\ F_{12} & F_{11} \end{bmatrix}, \begin{bmatrix} \frac{\partial L}{\partial O_{11}} & \frac{\partial L}{\partial O_{12}} \\ \frac{\partial L}{\partial O_{21}} & \frac{\partial L}{\partial O_{22}} \end{bmatrix} \right) \quad (21)$$

$\frac{\partial L}{\partial X}$ $\frac{\partial O}{\partial X}$ $\frac{\partial L}{\partial O}$

Debido a que el filtro generalmente se aplica a una entrada con a la que se le aplicó un relleno en las orillas (*Padding*), es necesario aplicar ahora el rellenado al filtro para poder aplicar la convolución con la que se calcula el error hacia atrás, pero al realizar este procedimiento los valores del filtro no corresponden a la posición normal del filtro sino a una rotación de 180°, como se muestra en la Figura 83, y en la Figura 84.

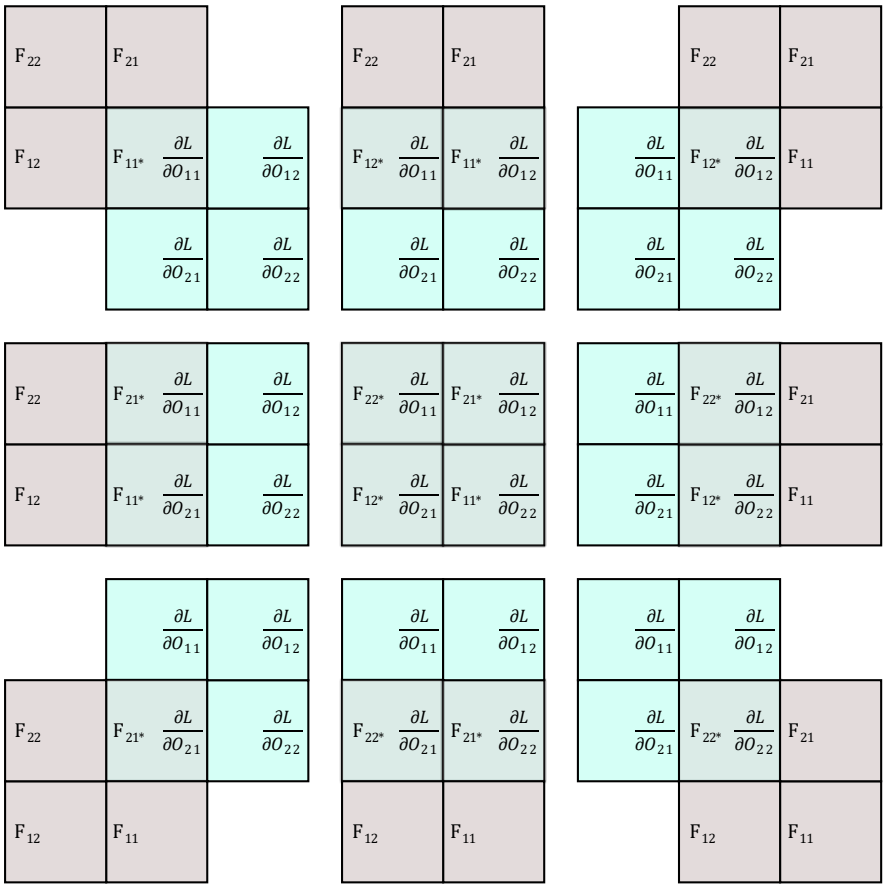


Figura 83. Cálculo de error en propagación hacia atrás. Se aplica una convolución con el filtro rotado 180° y un relleno en las orillas.

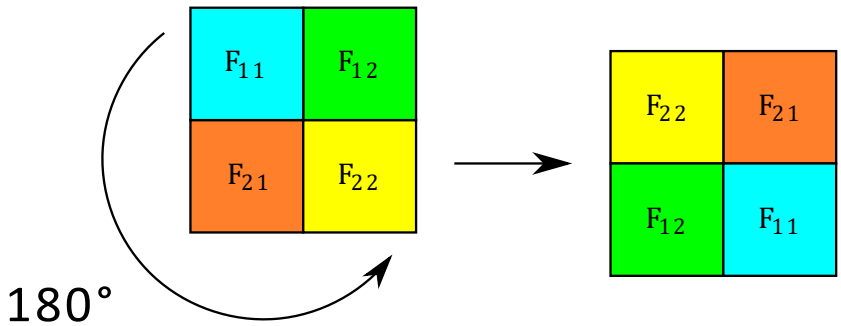


Figura 84. Rotación del filtro

Referencias

[1] A. Dantcheva, P. Elia, y A. Ross, «What Else Does Your Biometric Data Reveal? A Survey on Soft Biometrics», *IEEE Trans. Inf. Forensics Secur.*, vol. 11, n.º 3, pp. 441-467, mar. 2016, doi: 10.1109/TIFS.2015.2480381.

[2] V. O. Andersson y R. M. Araujo, «Full Body Person Identification Using the Kinect Sensor», en *2014 IEEE 26th International Conference on Tools with Artificial Intelligence*, nov. 2014, pp. 627-633, doi: 10.1109/ICTAI.2014.99.

[3] P. C y S. Sakkara, «Gait Recognition using skeleton data», en *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, ago. 2015, pp. 2302-2306, doi: 10.1109/ICACCI.2015.7275961.

[4] W. Min, M. Fan, J. Li, y Q. Han, «Real-time face recognition based on pre-identification and multi-scale classification», *IET Comput. Vis.*, vol. 13, n.º 2, pp. 165-171, mar. 2019, doi: 10.1049/iet-cvi.2018.5586.

[5] I. Rida, N. Almaadeed, y S. Almaadeed, «Robust gait recognition: a comprehensive survey», *IET Biom.*, vol. 8, n.º 1, pp. 14-28, ene. 2019, doi: 10.1049/iet-bmt.2018.5063.

[6] F. M. Castro, M. J. Marín-Jiménez, N. Guil, y N. Pérez de la Blanca, «Automatic Learning of Gait Signatures for People Identification», en *Advances in Computational Intelligence*, Cham, 2017, pp. 257-270, doi: 10.1007/978-3-319-59147-6_23.

[7] P. Arora, M. Hanmandlu, y S. Srivastava, «Gait based authentication using gait information image features», *Pattern Recognit. Lett.*, vol. 68, pp. 336-342, dic. 2015, doi: 10.1016/j.patrec.2015.05.016.

[8] J. Kovač y P. Peer, «Human Skeleton Model Based Dynamic Features for Walking Speed Invariant Gait Recognition», *Math. Probl. Eng.*, vol. 2014, pp. 1-15, 2014, doi: 10.1155/2014/484320.

[9] V. A. Chenarlogh y F. Razzazi, «Multi-stream 3D CNN structure for human action recognition trained by limited data», *IET Comput. Vis.*, vol. 13, n.º 3, pp. 338-344, 2019, doi: 10.1049/iet-cvi.2018.5088.

[10] M. Camplani *et al.*, «Multiple human tracking in RGB-depth data: a survey», *IET Comput. Vis.*, vol. 11, n.º 4, pp. 265-285, jun. 2017, doi: 10.1049/iet-cvi.2016.0178.

[11] H.-H. Pham, L. Khoudour, A. Crouzil, P. Zegers, y S. A. Velastin, «Learning to recognise 3D human action from a new skeleton-based representation using deep convolutional neural networks», *IET Comput. Vis.*, vol. 13, n.º 3, pp. 319-328, abr. 2019, doi: 10.1049/iet-cvi.2018.5014.

[12] B. Dikovski, G. Madjarov, y D. Gjorgjevikj, «Evaluation of different feature sets for gait recognition using skeletal data from Kinect», en *2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, may 2014, pp. 1304-1308, doi: 10.1109/MIPRO.2014.6859769.

[13] A. H. Bayat, M. M. Arzani, M. Fathy, A. Matinnejad, B. Minaei-Bidgoli, y R. Entezari, «A probabilistic graphical model approach for human activity recognition using skeleton data», en *2016 2nd International Conference of Signal Processing and Intelligent Systems (ICSPIS)*, dic. 2016, pp. 1-5, doi: 10.1109/ICSPIS.2016.7869894.

[14] E. Cippitelli, S. Gasparrini, E. Gambi, y S. Spinsante, «A Human Activity Recognition System Using Skeleton Data from RGBD Sensors», *Comput. Intell. Neurosci.*, vol. 2016, pp. 1-14, 2016, doi: 10.1155/2016/4351435.

[15] Q. Wu y G. Guo, «Gender Recognition from Unconstrained and Articulated Human Body», *Sci. World J.*, vol. 2014, pp. 1-12, 2014, doi: 10.1155/2014/513240.

[16] M. L. Anjum, S. Rosa, y B. Bona, «Tracking a Subset of Skeleton Joints: An Effective Approach towards Complex Human Activity Recognition», *J. Robot.*, vol. 2017, pp. 1-8, 2017, doi: 10.1155/2017/7610417.

[17] S. Yu, D. Tan, y T. Tan, «A Framework for Evaluating the Effect of View Angle, Clothing and Carrying Condition on Gait Recognition», en *18th International Conference on Pattern Recognition (ICPR'06)*, 2006, vol. 4, pp. 441-444, doi: 10.1109/ICPR.2006.67.

[18] S. Zheng, J. Zhang, K. Huang, R. He, y T. Tan, «Robust view transformation model for gait recognition», en *2011 18th IEEE International Conference on Image Processing*, sep. 2011, pp. 2073-2076, doi: 10.1109/ICIP.2011.6115889.

[19] N. Almudhahka, M. Nixon, y J. Hare, «Human face identification via comparative soft biometrics», en *2016 IEEE International Conference on Identity, Security and Behavior Analysis (ISBA)*, feb. 2016, pp. 1-6, doi: 10.1109/ISBA.2016.7477246.

[20] S. Denman, M. Halstead, C. Fookes, y S. Sridharan, «Searching for people using semantic soft biometric descriptions», *Pattern Recognit. Lett.*, vol. 68, pp. 306-315, dic. 2015, doi: 10.1016/j.patrec.2015.06.015.

[21] N. Dalal y B. Triggs, «Histograms of oriented gradients for human detection», en *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, jun. 2005, vol. 1, pp. 886-893 vol. 1, doi: 10.1109/CVPR.2005.177.

[22] E. Gonzalez-Sosa, R. Vera-Rodriguez, J. Fierrez, y V. M. Patel, «Exploring Body Shape From mmW Images for Person Recognition», *IEEE Trans. Inf. Forensics Secur.*, vol. 12, n.º 9, pp. 2078-2089, sep. 2017, doi: 10.1109/TIFS.2017.2695979.

[23] N. Yijie, W. Ziye, Q. Lingbo, y Z. Ziran, «Automatic target recognition method for millimeter-wave body scanner», en *2016 IEEE International Conference on Microwave and Millimeter Wave Technology (ICMMT)*, jun. 2016, vol. 2, pp. 928-930, doi: 10.1109/ICMMT.2016.7762489.

[24] F. Poiesi, R. Mazzon, y A. Cavallaro, «Multi-target tracking on confidence maps: An application to people tracking», *Comput. Vis. Image Underst.*, vol. 117, n.º 10, pp. 1257-1272, oct. 2013, doi: 10.1016/j.cviu.2012.08.008.

[25] A. A. Chaaraoui, J. R. Padilla-López, y F. Flórez-Revuelta, «Abnormal gait detection with RGB-D devices using joint motion history features», en *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, may 2015, vol. 07, pp. 1-6, doi: 10.1109/FG.2015.7284881.

[26] L. Wang y T. Tan, «Silhouette Analysis-Based Gait Recognition for Human Identification», *IEEE Trans. PATTERN Anal. Mach. Intell.*, vol. 25, n.º 12, p. 14, 2003, doi: 10.1109/TPAMI.2003.1251144.

[27] D. López-Fernández, F. Madrid-Cuevas, A. Carmona-Poyato, M. Marín-Jiménez, y R. Muñoz-Salinas, «The AVA Multi-View Dataset for Gait Recognition», ago. 2014, doi: 10.1007/978-3-319-13323-2_3.

[28] E.-S. M. El-Alfy y A. G. Binsaadoon, «Automated gait-based gender identification using fuzzy local binary patterns with tuned parameters», *J. Ambient Intell. Humaniz. Comput.*, vol. 10, n.º 7, pp. 2495-2504, jul. 2019, doi: 10.1007/s12652-018-0728-0.
[29] S. Yan, Y. Xia, J. S. Smith, W. Lu, y B. Zhang, «Multiscale Convolutional Neural Networks for Hand Detection», *Appl. Comput. Intell. Soft Comput.*, vol. 2017, pp. 1-13, 2017, doi: 10.1155/2017/9830641.
[30] Z. Cao, G. Hidalgo Martinez, T. Simon, S.-E. Wei, y Y. A. Sheikh, «OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields», *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1-1, 2019, doi: 10.1109/TPAMI.2019.2929257.
[31] M. Contreras-Murillo, S. G. de-los-Cobos-Silva, P. Lara-Velázquez, E. A. Rincón-García, R. A. Mora-Gutiérrez, y M. Á. Gutiérrez-Andrade, «Sex Classification via 2D-Skeletonization», *Math. Probl. Eng.*, vol. 2020, p. e6182654, nov. 2020, doi: 10.1155/2020/6182654.
[32] I. Vasilev, *Python deep learning: exploring deep learning techniques and neural network architectures with PyTorch, Keras, and TensorFlow*. 2019.
[33] G. Zaccone, R. Karim, y an O. M. C. Safari, *Deep Learning with TensorFlow - Second Edition*. 2018.
[34] R. Gopalakrishnan y A. Venkateswarlu, *Machine Learning for Mobile*. Place of publication not identified: Packt Publishing, 2018.
[35] G. Bonaccorso, *Mastering machine learning algorithms: expert techniques to implement popular machine learning algorithms and fine-tune your models*. Birmingham, UK: Packt Publishing, 2018.
[36] S. Ravichandiran, *Hands-On Reinforcement Learning with Python: Master Reinforcement and Deep Reinforcement Learning Using OpenAI Gym and TensorFlow.* Birmingham: Packt Publishing Ltd, 2018.
[37] M. Lapan, *Deep reinforcement learning hands-on: apply modern RL methods, with deep Q-networks, value iteration, policy gradients, TRPO,AlphaGo Zero and more*. Birmingham Mumbai: Packt Publishing, 2018.
[38] D. Rothman y Packt Publishing, *Artificial intelligence by example: develop machine intelligence from scratch using real artificial intelligence use cases*. Birmingham: Packt Publishing Ltd., 2018.
[39] M. P. Deisenroth, A. A. Faisal, y C. S. Ong, *Mathematics for machine learning*. Cambridge ; New York, NY: Cambridge University Press, 2020.
[40] C. C. Aggarwal, *Neural networks and deep learning: a textbook*. Cham: Springer, 2018.
[41] R. Shanmugamani y S. Moore, *Deep learning for computer vision: expert techniques to train advanced neural networks using TensorFlow and Keras*. Birmingham Mumbai: Packt, 2018.