Presentación:

Instituto Tecnológico de las Américas

Grupo*:*

#8

Materia*:*

Programación 3

Sección:

G4

Día/hora:

Martes y jueves / 2:00 PM – 4:00 PM

Maestro:

Willis Polanco

Fecha de entrega:

Viernes 17 de abril del 2021

Tema:

Análisis y diseño del proyecto

Análisis y Diseño: App Visión Computacional

El reconocimiento facial:

El reconocimiento facial (RF) automatizado por computadora es un área de investigación en pleno desarrollo. Identificar a una persona por medio de una imagen de su rostro implica emular el proceso cognitivo que realiza un ser humano al reconocer a sus pares. Aunque existen trabajos en el campo de la psicofísica y la neurociencia, aún no se sabe con certeza cómo funcionan estos procesos internamente en el cerebro humano.

Independientemente de la técnica utilizada para la solución, el RF requiere de tres etapas: 1) la detección del rostro en una imagen, 2) la extracción de características y 3) la identificación y/o verificación de la cara mediante la clasificación de las características. En la identificación, el sistema informa la identidad de la persona, mientras que en la verificación confirma o rechaza esta. Cada una de estas áreas ha sido objeto de investigaciones, dando origen a diferentes técnicas para resolver cada problema.

Detección del rostro

La detección implica encontrar las áreas dentro de una imagen que contienen un rostro. Básicamente se trata de descartar todo lo que sea fondo, y así obtener la ubicación y tamaño exacto de la cara.

Uno de los primeros m´etodos para resolver este problema fue el de plantillas deformables. Cada caracter´ıstica del rostro (ojos y boca en este caso) es representada por medio de una plantilla. La idea general es tener una funci´on de energ´ıa que relacione las intensidades de la imagen (valles y picos) con las propiedades de la plantilla. Una vez ubicada la plantilla se ajustan sus par´ametros para minimizar la funci´on de energ´ıa.

Otros enfoques se vuelcan a la clasificaci´on de patrones por medio de ejemplos [SP98]. Como s´olo hay que detectar caras (sin identificar sujetos espec´ıficamente) se crean patrones para zonas que representan una cara y zonas que no. Generalmente la soluci´on para encontrar patrones que no son rostros, es por medio de la modalidad denominada *"bootstrap"*, en la cual se comienza con patrones que son rostros, se pone en marcha el clasificador, y todos los falsos positivos se agregan como patrones de *no-rostro*. En una siguiente pasada, si aparecen nuevos falsos positivos, ´estos se siguen agregando como patrones.

Cabe destacar que el m´etodo de clasificaci´on es independiente de las im´agenes utilizadas como patrones. En [SP98] se utiliza una red neuronal artificial como clasificador, espec´ıficamente un perceptr´on multicapa.

Los u´ltimos trabajos se basan en el m´etodo propuesto en [VJ04], que alcanza las tasas de reconocimiento de los mejores sistemas, pero con la ventaja de obtener una alta velocidad de detecci´on. Una de sus contribuciones son los clasificadores en cascada, que permiten deshacerse de las regiones de fondo lo antes posible. Varios clasificadores son utilizados en serie, los primeros descartan la mayor´ıa de regiones que no son propensas a contener un rostro. Las regiones m´as prometedoras pasan a los clasificadores m´as complejos y que requieren de mayor tiempo de c´alculo.

1.1.2 Extracci´on de caracter´ısticas

La extracci´on de caracter´ısticas se refiere a la obtenci´on de propiedades o par´ametros particulares de cada rostro para luego poder ser clasificados. Se pueden tomar tres enfoques diferentes: 1) un enfoque hol´ıstico, bas´andose en la imagen del rostro como un todo, 2) un enfoque mediante caracter´ısticas

locales, dando mayor importancia a las diferentes partes del rostro, o 3) un enfoque h´ıbrido basado en la idea de que el sistema de percepci´on humana combina caracter´ısticas locales y globales para el reconocimiento [ZCPR03].

En el enfoque hol´ıstico, la t´ecnica m´as influyente es la de las *eigenfaces* [TP91]. Se utiliza un an´alisis de componentes principales para representar la imagen completa del rostro en un espacio de dimensiones reducido. Esta t´ecnica no necesita de grandes cantidades de fotos por sujeto, tiene una baja sensibilidad al ruido, y aunque presenta problemas para manejar las variaciones de iluminaci´on, pueden realizarse pequen˜os cambios para mejorar la efectividad en dichas condiciones [BHK97].

Por otro lado, se encuentran los modelos basados en las caracter´ısticas locales. Uno de los m´etodos de mayor ´exito es el de *Elastic Bunch Graph Matching* [WFKM97]. Aqu´ı, los rostros son representados en forma de grafos. Los puntos importantes en la cara que conforman los nodos del grafo (como los ojos o la nariz), son descritos por un conjunto de onditas de Gabor de diferentes escalas y rotaciones, logrando robustez para cambios en iluminaci´on, traslaci´on, distorsi´on, rotaci´on y escalado [ZJN07].

Los modelos h´ıbridos son adaptaciones del m´etodo de las *eigenfaces* aplicado a caracter´ısticas locales [PMS94]. As´ı, se combinan ventajas de los m´etodos hol´ısticos y las caracter´ısticas locales.

1.1.3 Reconocimiento

El reconocimiento consiste en clasificar las caracter´ısticas extra´ıdas de cada rostro. Esta clasificaci´on puede ser realizada de manera *supervisada*, en la cual un patr´on de entrada es identificado como miembro de una clase predefinida, o de manera *no supervisada*, donde el patr´on es asignado a una clase desconocida. Aqu´ı, cada clase es un sujeto, por lo tanto, al clasificar las caracter´ısticas se est´a indicando a qu´e sujeto pertenecen.

Para el disen˜o de clasificadores se pueden distinguir tres aproximaciones basadas en: 1) concepto de similaridad, 2)

aproximaci´on probabil´ıstica y 3) optimizaci´on de un criterio de error [JDM00].

Las aproximaciones basadas en el concepto de similaridad son las m´as simples e intuitivas, donde los patrones similares son asignados a la misma clase. Para ello, se establece una m´etrica que define la similaridad y luego se clasifica por medio de plantillas o m´ınima distancia usando uno o varios prototipos por clase. La t´ecnica de *eigenfaces* original [TP91] aplica la regla del vecino m´as cercano, utilizando como m´etrica la distancia Eucl´ıdea, donde

1.2. Propuesta del proyecto

cada prototipo es la media de los patrones de entrenamiento.

En el enfoque probabil´ıstico, se utilizan los conceptos de la teor´ıa de la decisi´on estad´ıstica para establecer los bordes de decisi´on de las diferentes clases. Se asume que las caracter´ısticas, que representan a un patr´on, tienen una funci´on de densidad de probabilidad condicionada a la clase. As´ı, un vector patr´on $\mathbf{x}$ perteneciente a la clase ω_i es visto como una observaci´on aleatoria desde la funci´on de probabilidad $p(\mathbf{x}|\omega_i)$. Las reglas de decisi´on m´as conocidas son la *regla de decision de Bayes* y la *regla de m´axima probabilidad*.

Por u´ltimo, la tercer aproximaci´on se basa en construir los bordes de decisi´on optimizando algu´n criterio de error. El objetivo es minimizar el error de clasificaci´on entre la respuesta deseada y la salida del clasificador. Un ejemplo de este tipo de clasificadores son las *redes neuronales* [JMM96], las cuales pueden considerarse como sistemas distribuidos y paralelos que consisten de pequen˜as unidades de procesamiento masivamente conectadas. Est´an conformadas por redes de grafos ponderados donde los nodos son las neuronas artificiales y los bordes (con pesos) son las conexiones entre las neuronas de entrada y salida de la red. Para lograr una clasificaci´on

adecuada son entrenadas con un algoritmo de entrenamiento a partir de un conjunto de datos. Los tipos de redes m´as comu´nmente usados son las redes hacia adelante, como por ejemplo, el perceptr´on multicapa.

1.2 Propuesta del proyecto

La idea central de este proyecto consiste en realizar el disen˜o y la implementaci´on de un sistema integral para identificaci´on de personas mediante el reconocimiento del rostro. El ambiente de aplicaci´on del sistema ser´a el Centro de Investigaci´on y Desarrollo en Sen˜ales, Sistemas e Inteligencia Computacional (*sinc(i)*) de la FICH, UNL y el Centro de Investigaci´on y Desarrollo de Ingenier´ıa en Sistemas de Informaci´on (CIDISI) de la UTN-FRSF.

Para ello, se desarroll´o una interfaz de usuario y se instal´o una c´amara web en una PC de escritorio en ambos centros de investigaci´on para la captura de im´agenes. Con el conjunto de im´agenes obtenidas se realiz´o una clasificaci´on manual de las mismas conformando una base de rostros para las pruebas del sistema.

Ya con la base de rostros, se desarroll´o el sistema en base a las tres etapas b´asicas de todo sistema de reconocimiento facial:

1.3. Objetivos

1. Localizaci´on del rostro: el cual ubica la cara en la imagen aplicando el algoritmo propuesto en [ML06], basado en una segmentaci´on de piel para detectar posibles regiones caras, y luego mediante combinaci´on de plantillas decide por una regi´on, recort´andola como posible rostro.

2. Extracci´on de caracter´ısticas: en esta etapa se obtienen las propiedades m´as relevantes del rostro mediante el m´etodo

eigenfaces, obteniendo como resultado un vector caracter´ıstico que representa la imagen de entrada en un espacio dimensional reducido para su posterior clasificaci´on.

3. Clasificaci´on: aqu´ı se disen˜´o una red neuronal para el reconocimiento de patrones, m´as especificamente un perceptr´on multicapa. La red se entren´o mediante el algoritmo de retropropagaci´on del error a partir de un conjunto de patrones de ejemplo. Con ´esto, se logr´o que la red “aprenda” sobre el conjunto de potenciales usuarios.

Cabe destacar dos fases del proceso de construcci´on: una de entrenamiento y otra de clasificaci´on. En la primera se ajustan varios par´ametros del sistema, siendo el paso m´as importante el entrenamiento de la red para el conjunto de datos especificado. Una vez logrado ´esto, se pasa a la fase de clasificaci´on verificando el desempen˜o mediante la presentaci´on de im´agenes rostros de usuarios tanto conocidos como desconocidos.

1.3 Objetivos

Objetivos generales

- Desarrollar un prototipo de herramienta computacional de c´odigo abierto que permita realizar reconocimiento facial, basado en la utilizaci´on de t´ecnicas del ´area de Procesamiento de Im´agenes e Inteligencia Computacional.
- Aplicar los conocimientos adquiridos en el transcurso de la carrera a un proyecto que realice un aporte ingenieril a la comunidad en general.

- Colaborar en proyectos de investigaci´on que requieran la participaci´on de profesionales inform´aticos en el ´area de procesamiento de im´agenes digitales e inteligencia artificial.

1.4. Alcances

Objetivos espec´ıficos

- Implementar los m´odulos del localizador, normalizador de caras y extractor de caracter´ısticas a partir de t´ecnicas de procesamiento digital de im´agenes.
- Disen˜ar e implementar el m´odulo del reconocedor mediante t´ecnicas de la Inteligencia Computacional.
- Desarrollar un prototipo que permita la prueba del sistema de RF. El software ofrecer´a las siguientes funcionalidades.
  - Tomar fotograf´ıas en vivo desde una c´amara.
  - Registrar y almacenar individuos en el sistema, asoci´andole una o m´as im´agenes capturadas.
  - Soportar el RF mediante la comparaci´on de una fotograf´ıa actual con las registradas en el sistema. Se proveer´an dos modos de funcionamiento: a) hacer corresponder la imagen tomada a la persona con la que m´as se ajuste de las conocidas en el sistema, y b) verificar la identidad del usuario por otro m´etodo (por ej. el ingreso del nombre y/o DNI por teclado).
  - Exportar e importar la informaci´on del individuo almacenada en el sistema. Esto permitir´ıa transportar su informaci´on para ser´ utilizada en otra ubicaci´on.

1.4 Alcances

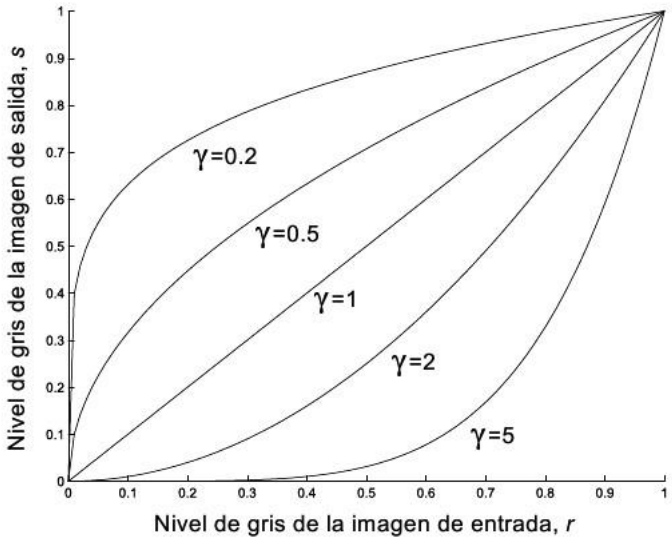
- La cantidad de usuarios se limitar´a a 11 en esta etapa de desarrollo del prototipo.
- No se har´a rechazo de impostores, sino que se supondr´a que todos los usuarios del sistema se encuentran registrados.
- No incluir´a el desarrollo del m´odulo de administraci´on de usuarios.

Marco te´orico

2.1.1 Imagen digital

Una imagen en escala de grises puede ser definida como una funci´on bidimensional *f*(*x*,*y*), donde *x* e *y* son coordenadas *espaciales*, y el valor de *f* en un par de coordenadas dado es denominado *intensidad* o *nivel de gris*. Cuando *x*, *y* y *f* son cantidades discretas, la imagen es llamada *imagen digital* [GW02]. De este modo, ´esta puede ser reprensentada como una matriz de $M \times N$ elementos llamados *p´ıxeles*.

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M-1,0) & f(M-1,1) & \dots & f(M-1,N-1) \end{bmatrix} \quad (2.1)$$



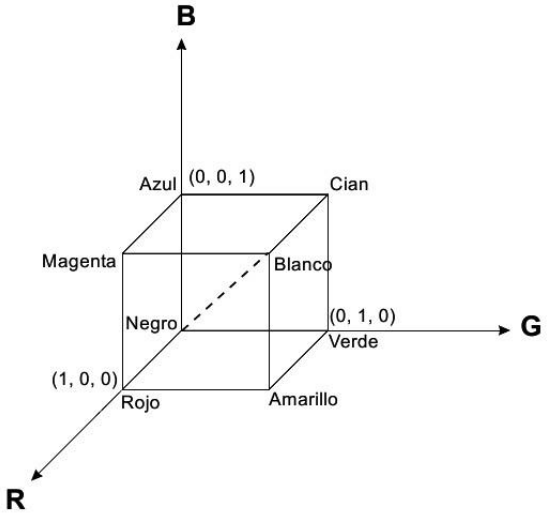
2.1.2 Correcci´on gamma

En el ´area de procesamiento de im´agenes, uno se encuentra con la necesidad de mejorar una imagen con el objetivo de obtener otra m´as aceptable para una aplicaci´on espec´ıfica. Dependiendo del problema en s´ı, existen diversas t´ecnicas para lograrlo, entre las cuales nos encontramos con las *transformaciones de potencia*. Estas transformaciones son u´tiles para la manipulaci´on de contraste y tienen la forma b´asica

$$s = cr^{\gamma},$$

donde r es el valor de entrada, s es el de salida y c y γ son constantes positivas. En la Figura 2.1 se muestran las transformaciones para diferentes valores de γ. Para $\gamma < 1$, ´esta amplifica los valores oscuros de entrada y contrae los claros, mientras que lo opuesto ocurre para $\gamma > 1$.

Una variedad de dispositivos usados para la captura de im´agenes responden de acuerdo a la ley de potencia. Por convenci´on, el exponente en la ecuaci´on es referido como *gamma*, y el proceso usado para la correcci´on de este fen´omeno es llamado *correcci´on gamma*.



.

2.1.3 Modelos de color

El uso del color en el procesamiento de im´agenes es de importancia ya que suele simplificar la identificaci´on y extracci´on de objetos en una escena. Un modelo de color especifica un sistema de coordenadas y un subespacio en el cual un color es definido por un punto. Muchos modelos est´an orientados al hardware como lo son el RGB (rojo, verde y azul), CMY (cian, magenta y amarillo) y el CMYK (cian, magenta, amarillo y negro); otros est´an orientados hacia aplicaciones donde se pretende manipular el color, tales como HSI (tono, saturaci´on e intensidad), HSV (tono, saturaci´on y valor) y el YCbCr (luminancia, crominancia azul y crominancia roja).

A continuaci´on se dan las bases de los modelos empleados en este trabajo.

Modelo de color RGB

Este modelo se basa sobre un sistema de coordenadas cartesianas y el subespacio en el cual se define el color es el cubo normalizado entre [0,1] (Ver Figura 2.2). Los valores rojo (R), verde (G) y azul (B) est´an sobre los v´ertices (1,0,0), (0,1,0) y (0,0,1) respectivamente; el origen corresponde al negro, en el punto m´as lejano a ´este se ubica el blanco, y el resto de los v´ertices corresponden a los colores secundarios: cian, magenta y amarillo. El resto de colores est´a compuesto por la combinaci´on de los colores primarios R, G y B, y son puntos dentro del cubo definido por un vector extendido desde el origen.

Las im´agenes representadas en el modelo RGB consisten de tres planos, uno por cada color primario. El nu´mero de bits usados para representar cada p´ıxel es llamado *profundidad*.

Modelo de color YCbCr

Este es un modelo de color usado en sistemas de video y fotograf´ıa digital,´ donde Y es la componente de luminancia y Cb y Cr son las componentes de crominancia azul y roja.

YCbCr es el equivalente digital al modelo YUV (Sistema de TV). Concentra la mayor parte de la informaci´on de la imagen en la luminancia y menos en la crominancia. El resultado es que los elementos de YCbCr est´an menos correlacionados y pueden ser codificados por separado.

El paso de un sistema RGB a un YCbCr es obtenido mediante las ecuaciones:

$$\begin{aligned} Y &= 0{,}3R + 0{,}6G + 0{,}1B \\ Cb &= \frac{B-Y}{2} + 0{,}5 \\ Cr &= \frac{R-Y}{2} + 0{,}5. \end{aligned}$$

2.1.4 Procesamiento morfol´ogico

El procesamiento morfol´ogico es una herramienta que nos permite extraer componentes de una imagen que son u´tiles para la representaci´on y descripci´on de regiones. Dos operaciones de inter´es en esta ´area son la dilataci´on y erosi´on.

Dilataci´on

Consideremos a las im´agenes binarias como conjuntos de elementos en Z^2, donde cada elemento de un conjunto corresponde a un par de coordenadas (x,y) en la imagen. Sean A y B dos conjuntos en Z^2, la dilataci´on de A por B es denotada por $A \oplus B$ y se define como

$$A \oplus B = \left\{ z | (\hat{B})_z \cap A \neq \oslash \right\},$$

donde $B\hat{}$ es la reflecci´on de B alrededor de su origen. A la imagen B se la suele llamar *elemento estructurante*. Esta ecuaci´on nos dice que la dilataci´on de A por B es el conjunto de todos los desplazamientos z, tal que $B\hat{}$ y A est´en solapadas al menos por un elemento. Los desplazamientos son realizados en base al origen de $B\hat{}$.

Una de las aplicaciones de la dilataci´on es para “relleno de huecos” de taman˜o igual o menor que el elemento estructurante.

Erosi´on

Al igual que en la dilataci´on, para conjuntos A y B en Z^2, la erosi´on de A por B se denota por $A \ominus B$y es definida como

$$A \ominus B = \{ z | (B)_z \subseteq A \}.$$

Esto nos indica que la erosi´on corresponde a todos los puntos´ z tales que B, trasladado por z, es un subconjunto de A. Esta operaci´on es la opuesta a la dilataci´on y su mayor uso es para eliminar detalles irrelevantes de taman˜o menor o igual que B.

Extensi´on a im´agenes en escala de grises

Aqu´ı, las im´agenes son representadas como $f(x,y)$ y $b(x,y)$, imagen de entrada y elemento estructurante respectivamente, donde se asume que ambas son funciones discretas.

La dilataci´on de f por b se define como

$$f \oplus b(s,t) = max\{f(s-x,t-y) + b(x,y) | (s-x),(t-y) \in D_f; (x,y) \in D_b\}, \quad (2.6)$$

donde D_f y D_b son el dominio de f y b, respectivamente. Las condiciones de que $(s - x)$ y $(t - y)$ deben estar en el dominio de f, y que x e y tienen que estar en el dominio de b, indican solapamiento de las dos funciones en al menos un elemento, al igual que en el caso binario.

El efecto logrado es una imagen de salida que tiende a ser m´as brillante que la de entrada, si todos los valores de los elementos estructurantes son positivos, y la eliminaci´on o reducci´on de detalles oscuros dependiendo de los valores y forma de b.

La erosi´on es definida como

$$f \ominus b(s,t) = min\{f(s+x,t+y) - b(x,y) | (s-x),(t+y) \in D_f; (x,y) \in D_b\} \quad (2.7),$$

donde D_f y D_b son el dominio de f y b, respectivamente. Las condiciones de que (s+x) y (t+y) tienen que estar en el dominio de f, y x e y en el dominio de b, expresan que el elemento estructurante tiene que estar completamente contenido en la imagen erosionada, es decir, f.

Como en el caso binario, la imagen de salida tiende a ser m´as oscura que la imagen de entrada, si todos los valores del elemento estructurante son positivos. Los detalles brillosos en la imagen de entrada son reducidos si ´estos son m´as pequen˜os en ´area que el elemento estructurante, con un grado de reducci´on dependiendo de los valores de grises circundantes a los detalles brillosos y por la forma y amplitud de los valores de b.

2.2 An´alisis de las componentes principales aplicado a im´agenes faciales

El An´alisis de las Componentes Principales (PCA, del ingl´es *Principal Component Analysis*) es una t´ecnica estad´ıstica que realiza una transformaci´on lineal y ortogonal de los datos a un nuevo sistema de coordenadas tal que la m´axima varianza queda proyectada sobre la primer coordenada, la segunda mayor varianza sobre la segunda coordenada, y as´ı sucesivamente. Transforma un conjunto de variables correlacionadas en un conjunto de variables no correlacionadas, y simplifica la transformaci´on encontrando las componentes m´as cercanas a las variables originales pero ordenadas en forma decreciente al orden de su varianza. Puede ser utilizada para reducir la dimensionalidad del conjunto de datos reteniendo aquellas caracter´ısticas del conjunto que mayor contribuci´on hacen a su varianza, manteniendo las primeras componentes principales de m´as bajo orden. Tales componentes frecuentemente contienen la mayor informaci´on del conjunto de datos.

Para el caso de im´agenes faciales, se pretende extraer las caracter´ısticas m´as relevantes en una imagen facial, las cuales

pueden o no estar relacionadas a las caracter´ısticas faciales como boca, nariz, ojos, etc.. En t´erminos matem´aticos, se quiere encontrar las componentes principales de la distribuci´on de caras, o lo que es lo mismo, los eigenvectores de la matriz de covarianza del conjunto de im´agenes caras [TP91]. Estos eigenvectores pueden ser vistos como el conjunto de vectores que caracterizan la variaci´on entre las im´agenes faciales. A partir de ´esto, cada cara puede ser representada exactamente como una combinaci´on lineal de los eigenvectores. Adem´as, cada rostro se puede aproximar usando aquellos que tienen asociados los eigenvalores m´as altos, es decir, los que responden a la mayor varianza del conjunto de im´agenes faciales.

2.2.1 C´alculo de eigenfaces

Sea I (x,y) una imagen facial en escala de grises de $N \times N$. Una imagen puede ser representada como un vector de dimensi´on N^2, o equivalentemente, un punto en el espacio N^2-dimensional. Por lo tanto, un conjunto de im´agenes es mapeado a una colecci´on de puntos en un espacio inmenso, considerando el taman˜o de las im´agenes con las que se suele trabajar. De esta manera, un conjunto de im´agenes faciales, dado que todas tienen un aspecto similar, se mapean en un subespacio del espacio de im´agenes y as´ı pueden ser descritas por un subespacio de menor dimensi´on. La idea es encontrar los vectores que mejor describan la distribuci´on de im´agenes cara dentro del espacio completo de im´agenes. Estos vectores de longitud N^2, llamados *eigenfaces*, definen el subespacio de im´agenes rostro, el cual se denomina *espacio de caras*.

Sea $\mathbf{\Gamma_1}$,$\mathbf{\Gamma_2}$,$\mathbf{\Gamma_3}$,...,$\mathbf{\Gamma_M}$ un conjunto de im´agenes faciales. La cara media del conjunto es definida por

$$\mathbf{\Psi} = \frac{1}{M} \sum_{n=1}^{M} \mathbf{\Gamma_n}$$

Cada cara difiere de ´esta por el vector $\mathbf{\Phi}_i = \mathbf{\Gamma}_i - \mathbf{\Psi}$. A partir de este conjunto de vectores, se busca un conjunto de M vectores ortonormales, $\mathbf{u}_n$, los cuales mejor describan los datos. El k-´esimo vector, $\mathbf{u}_k$, es seleccionado tal que

$$\lambda_k = \frac{1}{M}\sum_{n=1}^{M}\left(\mathbf{u}_k{}^T\mathbf{\Phi}_n\right)^2$$

es un m´aximo, sujeto a que los vectores $\mathbf{u}_k$ sean ortonormales. Los vectores $\mathbf{u}_k$ y los escalares λ_k son los eigenvectores y eigenvalores, respectivamente, de la matriz de covarianza

$$C = \frac{1}{M}\sum_{n=1}^{M}\mathbf{\Phi}_n\mathbf{\Phi}_n{}^T$$

$$= AA^T,$$

donde $A = [\mathbf{\Phi}_1\, \mathbf{\Phi}_2\, \mathbf{\Phi}_3 \ldots\, \mathbf{\Phi}_M]$. La matriz C resultante es de $N^2 \times N^2$ y obtener N^2 eigenvectores y eigenvalores es un tarea costosa para el taman˜o de im´agenes comu´nmente procesadas.

Una consideraci´on a tener en cuenta es que si el nu´mero de puntos en el espacio de im´agenes es menor que la dimensi´on del mismo ($M < N^2$), habr´a solamente $M - 1$ eigenvectores significantes, el resto tendr´a asociados eigenvalores iguales a cero. En este caso, se tienen M im´agenes caras de dimensi´on N^2 y se pueden encontrar los eigenvectores N^2-dimensionales resolviendo los eigenvectores para una matriz de $M \times M$ y luego realizar una combinaci´on lineal de las im´agenes faciales $\mathbf{\Phi}_l$.

Consideremos los eigenvectores $\mathbf{v}_l$ de A^TA tal que

$$A^TA\mathbf{v}_l = \mu_l\mathbf{v}_l.$$

Premultiplicando ambos lados por A, se tiene

$$AA^TA\mathbf{v}_l = \mu_lA\mathbf{v}_l.$$

De aqu´ı se ve que $A\mathbf{v}_l$ son los eigenvectores de la matriz de covarianza C. Entonces, se construye la matriz de $M \times M$, $L = A^TA$, donde $L_{mn} = \mathbf{\Phi}_m^T\mathbf{\Phi}_n$, y se encuentran los M eigenvectores $\mathbf{v}_l$, de L. Estos

vectores determinan la combinaci´on lineal de las M im´agenes faciales para obtener los eigenvectores

$$\mathbf{u}_l = A\mathbf{v}_l$$

$$= \sum_{k=1}^{M} \mathbf{v}_{lk}\mathbf{\Phi}_k.$$

De esta forma se reducen enormemente los c´alculos del orden del nu´mero de p´ıxeles en las im´agenes (N^2) al orden del nu´mero de im´agenes en el conjunto (M). Los eigenvalores asociados permiten ordenar los eigenvectores de acuerdo al aporte que hacen a la variaci´on entre im´agenes.

Proyecci´on y reconstrucci´on de una imagen facial

Una imagen facial $\mathbf{\Gamma}$ es proyectada sobre el espacio de caras por medio de la operaci´on

$$x_k = \mathbf{u}^T{}_k(\mathbf{\Gamma} - \mathbf{\Psi}), \qquad \text{para } k = 1,...,M. \tag{2.14}$$

Los pesos x_k forman un vector $\mathbf{x}^T = [x_1\, x_2\, x_3 \ldots x_M]$ que describen el aporte de cada eigenface en representar la imagen cara, tratando a las eigenfaces como un conjunto base para las im´agenes faciales.

La reconstrucci´on exacta de una cara se realiza a trav´es de una combinaci´on lineal de las eigenfaces ponderadas con los respectivos pesos x_k

M

$$\mathbf{\Gamma} = {}^{\mathrm{X}}x_k\mathbf{u}_k.$$

k=1

2.3 Reconocimiento de patrones

Un *patr´on* es un *arreglo de caracter´ısticas* que describen una sen˜al, y es representado por un vector multidimensional. Una *clase* es un conjunto de patrones que comparten una propiedad en comu´n. Las clases son denotadas por $\omega_1,\omega_2,...,\omega_W$, donde W es la cantidad de clases. Las t´ecnicas de reconocimiento de patrones son

aquellas que asignan los patrones a sus respectivas clases teniendo en cuenta las caracter´ısticas de ´estos.

El reconocimiento de patrones se divide en dos ´areas: *reconocimiento basado en la teor´ıa de la decisi´on* y *reconocimiento estructural*. El primero trata con patrones representados por caracter´ısticas cuantitativas, y el segundo con las relaciones estructurales de las caracter´ısticas de los patrones.

Reconocimiento basado en la teor´ıa de la decisi´on

Estos m´etodos se basan en *funciones de decisi´on*. Siendo $\mathbf{x} = (x_1,x_2,...,x_n)^T$ un vector patr´on y existiendo W clases $\omega_1,\omega_2,...,\omega_W$, la idea es encontrar W funciones $d_1(\mathbf{x}),d_2(\mathbf{x}),...,d_W(\mathbf{x})$ tal que si un patr´on **x** pertenece a la clase ω_i, entonces

$$d_i(\mathbf{x}) > d_j(\mathbf{x}), \qquad j = 1,2,...,W; \qquad j \neq i$$

Es decir, un patr´on **x** desconocido es evaluado en las W funciones de decisi´on, si el valor m´as alto es obtenido en la funci´on d_i, entonces se dice que **x** pertenece a la clase ω_i.

2.3.1 Redes neuronales

Uno de los m´etodos basados en la teor´ıa de la decisi´on es la red neuronal. *Una red neuronal es un procesador masivamente distribuido que tiene una propensi´on natural para almacenar el conocimiento experimental y ponerlo a disposici´on para su uso* [Hay99].

La idea de una red neuronal surge con el objetivo de modelar el comportamiento del cerebro humano al realizar una tarea, como es el reconocimiento de las personas, empleando una interconexi´on masiva de pequen˜as unidades de procesamiento las cuales se denominan *neuronas* o *nodos*. Las redes neuronales se asemejan al cerebro en dos aspectos: 1) el conocimiento es adquirido

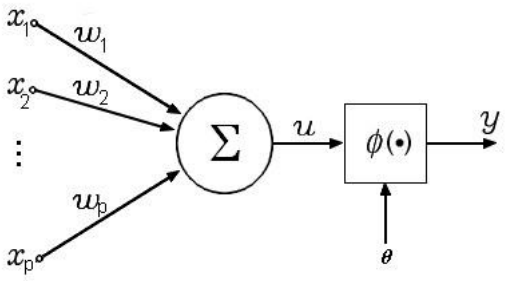


Figura 2.3: Arquitectura de un perceptr´on simple.

por un proceso de aprendizaje y 2) los pesos de conexi´on entre neuronas, conocidos como *pesos sin´apticos*, se usan para almacenar el conocimiento. El algoritmo de aprendizaje modifica los pesos sin´apticos hasta lograr el conocimiento requerido.

La manera en la cual las neuronas de una red est´an estructuradas est´a ligada al algoritmo de aprendizaje usado para el entrenamiento de la red. Entre las arquitecturas posibles para modelos neuronales est´an las denominadas *redes hacia adelante*, como lo son el perceptr´on simple, perceptr´on multicapa y redes de funci´on de base radial, las *redes recurrentes* y otros *modelos h´ıbridos*.

A continuaci´on se explicar´a el funcionamiento de un perceptr´on simple, la arquitectura utilizada en este trabajo, el perceptr´on multicapa, y se har´a una introducci´on a las redes de funci´on de base radial.

Perceptr´on simple

El perceptr´on es la forma m´as simple de red neuronal, consistiendo de una u´nica neurona. Su arquitectura se ilustra en la Figura 2.3. Est´a constituida por un combinador lineal que suma las sen˜ales de entrada x_j ponderadas por los pesos sin´apticos w_j correspondientes. La sen˜al de entrada es denotada por

$$\mathbf{x} = (x_1, x_2, \ldots, x_p)^T$$

y el conjunto de pesos sin´apticos est´a dado por el vector

$$\mathbf{w} = (w_1, w_2, ..., w_p)^T.$$

La salida del combinador lineal, representada por u, es la entrada a una *funci´on de activaci´on* que limita la amplitud de la salida de la neurona. T´ıpicamente, el rango de amplitud normalizado de la salida es entre [0, 1] o
[-1, 1].

La arquitectura tambi´en incluye un umbral θ que tiene como objetivo fijar un valor m´ınimo que debe superar la salida del combinador lineal para activar la neurona. Este umbral puede ser agregado al vector **w** como $w_0 = \theta$, considerando una entrada adicional $x_0 = 1$.

En t´erminos matem´aticos, una neurona est´a representada por medio de las ecuaciones

$$u = \sum_{j=0}^{p} w_j x_j$$

y

$$y = \varphi(u),$$

donde φ representa la funci´on de activaci´on e y es la salida de la neurona.

Existen tres tipos b´asicos de funciones de activaci´on para determinar la salida:

a) *Funci´on umbral.* Aplica la siguiente ecuaci´on

$$y = \begin{cases} 1 & \text{si } u \geq 0 \\ 0 & \text{si } u < 0 \end{cases}$$

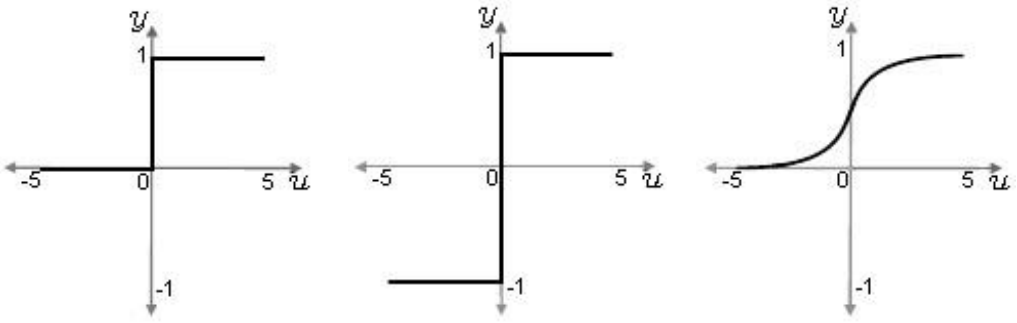
b) *Funci´on bipolar.* Se la define como

$$y = \begin{cases} 1 & \text{si } u \geq 0 \\ -1 & \text{si } u < 0 \end{cases} \quad (2.20)$$

c) *Funci´on sigmoidal*. Es la m´as comu´nmente usada en el modelado de redes neuronales, tambi´en llamada *funci´on log´ıstica*. Se define como una funci´on estrictamente incremental de la siguiente manera

$$\phi(u) = \frac{1}{1 + e^{-u}}$$

En la Figura 2.4 se muestran las tres funciones descritas. Cuando la salida de la funci´on de activaci´on es cercana o igual a su valor m´aximo, el perceptr´on es activado; mientras que cuando est´a cercano a su valor m´ınimo, se dice que est´a desactivado. Por lo tanto, se puede ver al perceptr´on simple como un clasificador de dos clases.



(a) (b) (c)

Figura 2.4: Funciones de activaci´on. a) Umbral, b) Bipolar, c) Sigmoidal.

Como se mencion´o anteriormente, los pesos sin´apticos se utilizan para almacenar el conocimiento de la neurona, siendo fijos o adaptados mediante el aprendizaje. Este aprendizaje implica que los par´ametros de la red sean adaptados a trav´es de un proceso continuo de estimulaci´on, por el ambiente en el cual se encuentra. La estimulaci´on es llevada a cabo por medio de los patrones de entrenamiento, y cuando algu´n criterio de finalizaci´on es

satisfecho, el algoritmo es detenido y se dice que la red est´a entrenada y lista para su funcionamiento.

Hay una variedad de algoritmos y reglas de aprendizaje para los diferentes disen˜os de redes. Para el caso del perceptr´on, una de ellas es la denominada *regla delta*. Esta se basa en ajustar los pesos con el objetivo de minimizar´ la diferencia entre la respuesta deseada y la obtenida por el perceptr´on. La respuesta deseada es denotada por $d(n)$ y la salida producida por un est´ımulo $\mathbf{x}(n)$, por $y(n)$. Entonces, el error medido en el paso n del entrenamiento est´a dado por

$$e(n) = d(n) - y(n).$$

El ajuste de los pesos se realiza minimizando el criterio de error cuadr´atico instant´aneo

$$E(\mathbf{w}) = \frac{1}{2}e^2(n),$$

cuyo gradiente es

$$\nabla E(\mathbf{w}) = -e(n)\varphi^0(n)\mathbf{x}(n).$$

De esta manera, el valor actualizado est´a dado por

$$w_j(n+1) = w_j(n) + \eta e(n)\varphi^0(n)x_j(n),$$

donde η es una constante positiva que determina la tasa de aprendizaje. Este par´ametro tiene un impacto importante en el desempen˜o del algoritmo. Si es pequen˜o, el proceso de aprendizaje avanza lentamente pero la convergencia a una soluci´on estable puede tardar demasiado tiempo; en cambio si η es grande el aprendizaje es m´as acelerado pero hay peligro de que el proceso de aprendizaje diverja y se obtenga una soluci´on inestable. El perceptr´on es capaz de lograr un ajuste de los pesos con error cero solamente cuando los patrones son *separables linealmente*, es decir, los patrones pertenecen a dos clases diferentes que pueden ser separados geom´etricamente por un hiperplano.

Perceptr´on multicapa

El perceptr´on multicapa (MLP, del ingl´es *Multi-Layer Perceptron*) representa una generalizaci´on del perceptr´on simple. Se constituye por una capa de entrada, una o varias capas ocultas y una capa de salida. Un esquema de una red con una capa oculta se muestra en la Figura 2.5. La capa de entrada simplemente copia el patr´on de entrada, las capas ocultas y de salida la conforman un conjunto de neuronas. La sen˜al de entrada se propaga por la red de capa en capa, de izquierda a derecha. Adem´as, se puede observar que cada neurona en alguna capa est´a conectada a todas las neuronas de la capa previa y la siguiente.

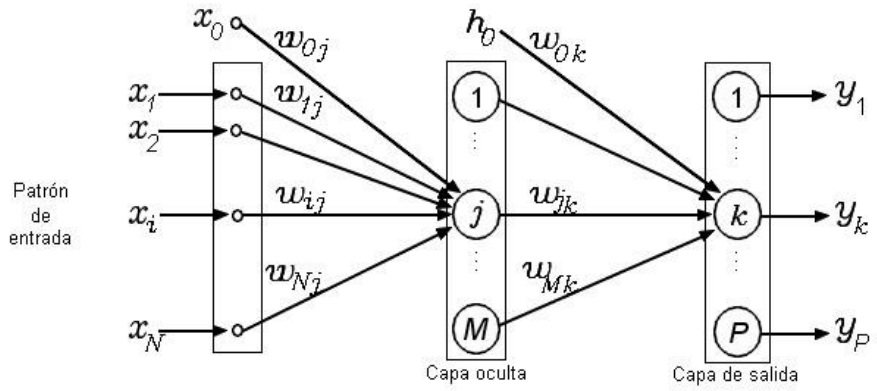
El MLP es entrenado en una manera supervisada con algoritmo de *retropropagaci´on del error*, el cual est´a basado en regla delta, mencionado anteriormente para el caso del perceptr´on simple [Hay99]. Este algoritmo comienza con una fase de propagaci´on hacia adelante, donde un patr´on de entrenamiento es aplicado en la capa de entrada obteniendo la salida de cada neurona que constituye la red. Espec´ıficamente, la salida de la neurona *j* es calculada de la misma manera que para el caso del perceptr´on simple,

$$y_j(n) = \varphi_j(u_j(n)), \tag{2.26}$$

donde φ_j es la funci´on de activaci´on y $u_j(n)$ la salida del combinador lineal definida por

$$u_j(n) = \sum_{i=0}^{p} w_{ij}(n) y_i(n), \tag{2.27}$$

donde *p* es el nu´mero de entradas aplicado a la neurona *j*, $w_{ij}(n)$ es el peso sin´aptico que conecta la neurona *i* a la neurona *j*, y $y_i(n)$ es la entrada de la neurona *j*, o equivalentemente, la salida de la neurona *i*.



Siguiendo con el esquema de la Figura 2.5, consideremos los nodos de la capa de salida. El error medido en la neurona *k* en el paso *n* es definido por

$$e_k(n) = d_k(n) - y_k(n).$$

Entonces, partiendo del error cuadr´atico instant´aneo en la neurona *k* definido como $\frac{1}{2}e_k^2(n)$, la suma instant´anea de errores cuadr´aticos es obtenida por sumar $\frac{1}{2}e_k^2(n)$ sobre todas las neuronas, expresado matem´aticamente como

$$E(n) = \frac{1}{2}\sum_{k=1}^{P} e_k^2(n),$$

donde *P* es la cantidad de neuronas en la capa de salida. La suma instant´anea de errores cuadr´aticos *E*(*n*) es una funci´on de todos los pesos sin´apticos de la red, por lo tanto, el objetivo del proceso de aprendizaje es ajustar los pesos minimizando *E*(*n*).

La regla delta establece que la correcci´on aplicada a los pesos sin´apticos en los nodos de la capa de salida es

$$\begin{aligned} \triangle\omega_{jk}(n) &= -\eta\frac{\partial E(n)}{\partial\omega_{jk}(n)} \\ &= \eta e_k(n)\phi_k'(u_k(n))y_j(n) \end{aligned}$$

,

siendo η la tasa de aprendizaje. Esto es posible dado que se conocen las´ respuestas deseadas de cada neurona, mientras que no ocurre los mismo para los nodos de las capas ocultas. De todos modos, es posible inferir una regla de aprendizaje basada en modificar estos pesos para tratar de disminuir el error en la capa de salida, mediante la propagaci´on hacia atr´as de las cantidades e_k. As´ı, la actualizaci´on de los pesos en las capas ocultas se realiza mediante el gradiente descendiente de la funci´on criterio error cuadr´atico, pero calculado con respecto a los pesos w_{ij} como

$$\triangle\omega_{ij}(n) = -\eta\frac{\partial E(n)}{\partial\omega_{ij}(n)}.$$

A trav´es de la regla de la cadena, se puede calcular la derivada parcial de (2.31) segu´n

$$\frac{\partial E(n)}{\partial\omega_{ij}(n)} = \frac{\partial E(n)}{\partial y_j(n)}\frac{\partial y_j(n)}{\partial u_j(n)}\frac{\partial u_j(n)}{\partial\omega_{ij}(n)},$$

donde las u´ltimas dos derivadas parciales son calculadas como

$$\frac{\partial y_j(n)}{\partial u_j(n)} = \phi'_j(u_j(n))$$

y

$$\frac{\partial u_j(n)}{\partial\omega_{ij}(n)} = y_i;$$

y por u´ltimo, se calcula la derivada parcial $\partial E(n)/\partial y_j(n)$ como

$$\frac{\partial E(n)}{\partial y_j(n)} = \sum_{k=1}^{P} e_k(n)\frac{\partial e_k(n)}{\partial y_j(n)}.$$

Usando la regla de la cadena para la derivada parcial $\partial e_k(n)/\partial y_j(n)$ se reescribe la ecuaci´on y se obtiene

$$\begin{aligned}\frac{\partial E(n)}{\partial y_j(n)} &= \sum_{k=1}^{P} e_k(n)\frac{\partial e_k(n)}{\partial u_k(n)}\frac{\partial u_k(n)}{\partial y_j(n)} \\ &= -\sum_{k=1}^{P} e_k(n)\phi'_k(u_k(n))\omega_{jk}(n)\end{aligned}.$$

Reemplazando en (2.31) se obtiene la regla de actualizaci´on de pesos en las capas ocultas

$$\triangle\omega_{ij}(n) = \eta \left[\sum_{k=1}^{P} e_k(n)\phi'_k(u_k(n))\omega_{jk}(n) \right] \phi'_j(u_j(n))y_i \quad .$$

La actualizaci´on de los pesos es realizada por cada patr´on de entrenamiento que se presente a la red. La convergencia del algoritmo de aprendizaje, a una soluci´on adecuada, se comprueba al finalizar la presentaci´on del conjunto completo de entrenamiento, verificando el valor de alguna funci´on error prefijada o mediante alguna t´ecnica de estimaci´on del error.

Con respecto a la tasa de aprendizaje η, aqu´ı se presenta el mismo caso que para el perceptr´on simple. Un m´etodo para incrementar la tasa de aprendizaje y evitar una soluci´on inestable es modificar la regla delta incluyendo un t´ermino denominado *momento*

$$\triangle\omega_{ij}(n) = \alpha\triangle\omega_{ij}(n-1) - \eta\frac{\partial E(n)}{\partial\omega_{ij}(n)},$$

donde α es frecuentemente un nu´mero positivo llamado *constante de momento*. Cabe aclarar que los sub´ındices en la ecuaci´on hacen referencia tanto a pesos de neuronas ocultas como de salidas.

Red de funci´on de base radial

Una red de funci´on de base radial (RBF, del ingl´es *Radial Basis Function*) est´a constituida por tres capas cuando es usada para clasificaci´on, una capa de entrada, una oculta y una de salida [HH93]. Un esquema de una red RBF se observa en la Figura 2.6. Al igual que en una red MLP, la capa de entrada copia el patr´on de entrada. La capa oculta y de salida la conforman un conjunto de nodos. Las salidas de los nodos en la capa de salida son una combinaci´on lineal ponderada de las salidas de los nodos de la capa oculta.

En t´erminos matem´aticos, la salida del nodo *k* en la capa de salida est´a dada por

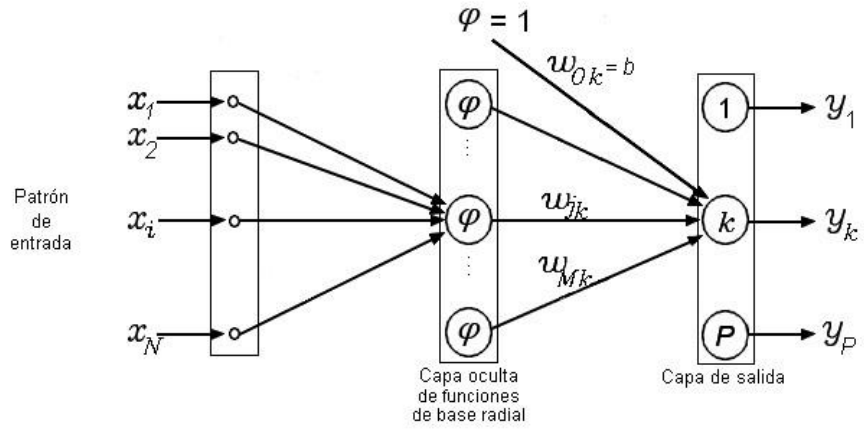
$$y_k = \sum_{j=0}^{M} w_{jk}\phi_j,$$

donde ω_{jk} es el peso sin´aptico que conecta el nodo *j* de la capa oculta al nodo *k* de la capa de salida, y ϕ_j para $j = 1,2,...,M$ es un conjunto de funciones conocidas como *funciones de base radial*. Estas funciones son las salidas de los nodos de la capa oculta, y comu´nmente tienen una forma Gaussiana dada por

$$\varphi_j = e^{\frac{-u_j}{2\sigma_j^2}},$$

donde

$$N\,uj = \sum_{i=1}(xi - wij)(xi - wij).$$



Aqu´ı, ϕ_j es la salida del nodo *j* en la capa oculta, x_i es la entrada a la red y w_{ij} corresponde a los pesos sin´apticos del nodo *j*. Adem´as, se deben notar que los pesos w_{ij} son el centro de la funci´on Gaussiana, y que σ_j^2 define el ancho de la misma. La salida del nodo *j* est´a en el rango de 0 a 1, de modo que cuanto m´as cerca est´a la entrada de

la red al centro de la funci´on Gaussiana, mayor ser´a la respuesta del nodo *j*. El nombre *funci´on de base radial* se debe a que ϕ_j es radialmente sim´etrica.

Existen varias aproximaciones para el entrenamiento de una red RBF. Algunas de ´estas, suelen dividir el aprendizaje en dos etapas. En la primer etapa, el aprendizaje se realiza en la capa oculta mediante un m´etodo no supervisado, como por ejemplo, un algoritmo de clustering; en la segunda etapa, el aprendizaje se realiza en la capa de salida usando un m´etodo supervisado. Una vez obtenida una soluci´on inicial, se puede aplicar un algoritmo de entrenamiento supervisado en ambas capas, para optimizar los par´ametros de la red.

Hay una variedad de algoritmos de clustering que se pueden aplicar en la capa oculta. Uno de los m´as conocidos por su simplicidad y producir buenos resultados, es el algoritmo *K-means*. Por otro lado, para el caso de la capa de salida, se suele usar la regla delta; mientras que para la optimizaci´on de par´ametros se puede aplicar un m´etodo de gradiente.

Desarrollo del software

En este proyecto se desarroll´o un sistema que permite identificar a una persona por medio del reconocimiento de una imagen de su rostro. Para ello, se instal´o una c´amara web en una PC de escritorio para la captura de im´agenes. Por medio de ´esta, el sistema toma una foto de la persona, la procesa e informa la identidad de la misma.

A la hora de modelar sistemas, una de las herramientas m´as utilizadas es el Lenguaje Unificado de Modelado (UML, del ingl´es *Unified Modeling Language*) [BRJ98]. Este est´andar es un lenguage gr´afico para visualizar, especificar, construir y documentar un sistema. Provee un vocabulario y reglas que permiten realizar modelos conceptuales de varios aspectos del sistema, con el objetivo de facilitar la comunicaci´on entre el equipo de desarrollo y los usuarios finales, quienes son los que especifican las funciones y el comportamiento deseado del mismo.

A fin de realizar un an´alisis y documentaci´on del sistema, en las siguientes secciones se explican las funcionalidades necesarias por medio de UML. Se especifican los requerimientos identificados, y se realizan los diagramas de casos de uso, clases y secuencia respectivamente.

3.1 An´alisis de requerimientos

Un requerimiento puede ser definido como:

- Una condici´on o capacidad que un usuario necesita para resolver un problema o lograr un objetivo.
- Una condici´on o capacidad que debe tener un sistema para satisfacer un contrato, norma, especificaci´on, u otros documentos formales.
- Una representaci´on documentada de una condici´on o capacidad como las mencionadas anteriormente.

A su vez, estos requerimientos pueden ser diferenciados en *requerimientos funcionales* y *requerimientos no funcionales*. Los

primeros, son declaraciones de las funciones que el sistema debe ser capaz de realizar, de como debe responder ante entradas particulares; mientras que los segundos, son restricciones del sistema, tales como disponibilidad, mantenimiento, seguridad, capacidad de los dispositivos de entrada/salida, rendimiento (como velocidad y tiempo de respuesta), etc..

En base a estas definiciones, se realiza un an´alisis de requerimientos con el objetivo de identificar tanto las funcionalidades que se esperan del software como sus limitaciones.

3.1.1 Requerimientos funcionales

El sistema debe ser capaz de:

1. Capturar una imagen a trav´es de una c´amara web, con la posibilidad de guardarla.
2. Identificar a una persona por medio de una imagen de su rostro.

3.1.2 Requerimientos no funcionales

Como requerimientos no funcionales se especifican los siguientes:

1. El sistema debe funcionar con una c´amara web est´andar y luz artificial.
2. El sistema debe ejecutarse en tiempo real, lo que implica que las operaciones computacionales deben procesarse en un tiempo aceptable.

3.2 Diagrama de casos de uso

Un caso de uso captura el comportamiento deseado de un sistema en desarrollo, sin tener que detallar como se implementa ´este. Es una

descripci´on de un conjunto de secuencias de acciones que el sistema lleva a cabo para producir un resultado de inter´es para un usuario, reflejando como deber´ıan interactuar ambas partes.

Un diagrama de casos de uso muestra la relaci´on entre los usuarios del sistema y los casos de uso, modelando los aspectos din´amicos del mismo.

Para el caso analizado aqu´ı, se pretende visualizar como interactu´a el usuario con el sistema haciendo uso de las funcionalidades ofrecidas por el mismo. En la Figura 3.1 se puede observar un diagrama de alto nivel del sistema, y en la Figura 3.2 un diagrama m´as detallado.

A continuaci´on se describen los casos de uso correspondientes al diagrama de la Figura 3.1.

Caso de uso: Capturar imagen
Actor: Usuario
Descripci´on: Captura una imagen de la persona con la c´amara web.
Curso normal:
1) Comienza cuando el usuario presiona el bot´on ’Capturar’ o ’Identificar’. 2) Se toma una foto de la persona.
Cursos alternativos:
2.A) La c´amara no est´a inicializada. 2.A.1) El sistema inicializa la c´amara web previsualizando lo que ´esta captura.

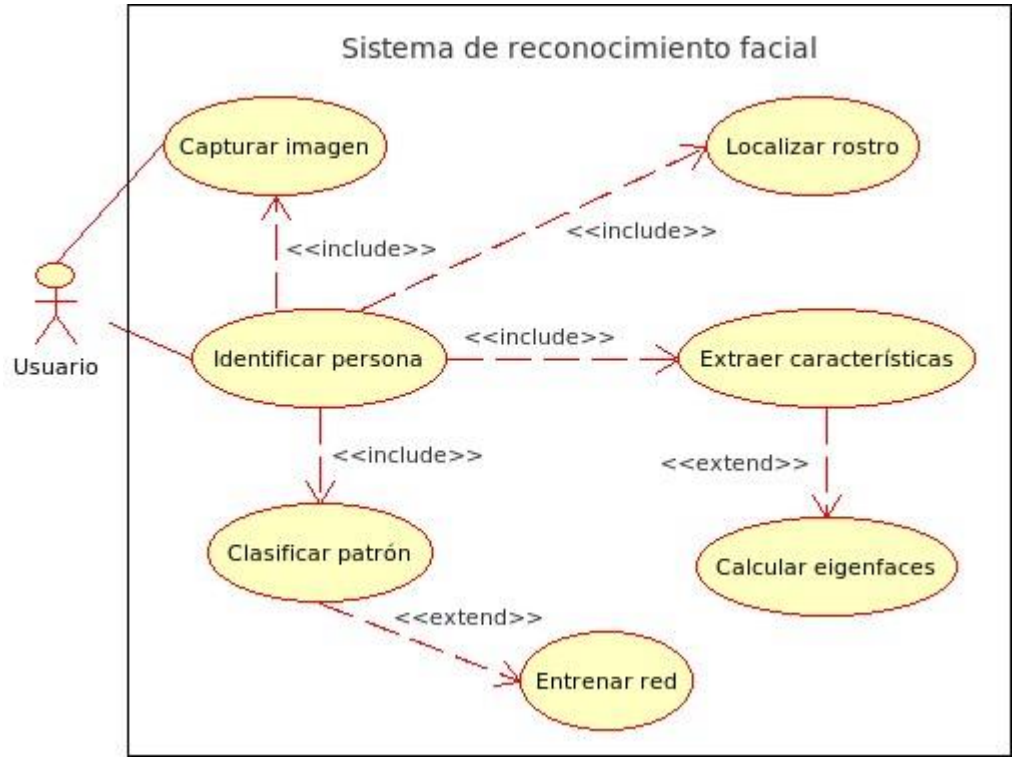


Figura 3.1: Diagrama de casos de uso de alto nivel.

Caso de uso: Identificar persona
Actor: Usuario
Descripci´on: Identifica a una persona a partir de una imagen de su rostro.
Curso normal:
1) Comienza cuando el usuario presiona el bot´on ’Identificar’. 2) El sistema captura una imagen del usuario. 3) Se localiza el rostro en la imagen. 4) Se proyecta la imagen facial mediante las eigenfaces y se obtiene un patr´on representativo de ´esta. 5) Se clasifica el patr´on mediante una red neuronal. 6) El sistema informa la identidad de la persona.

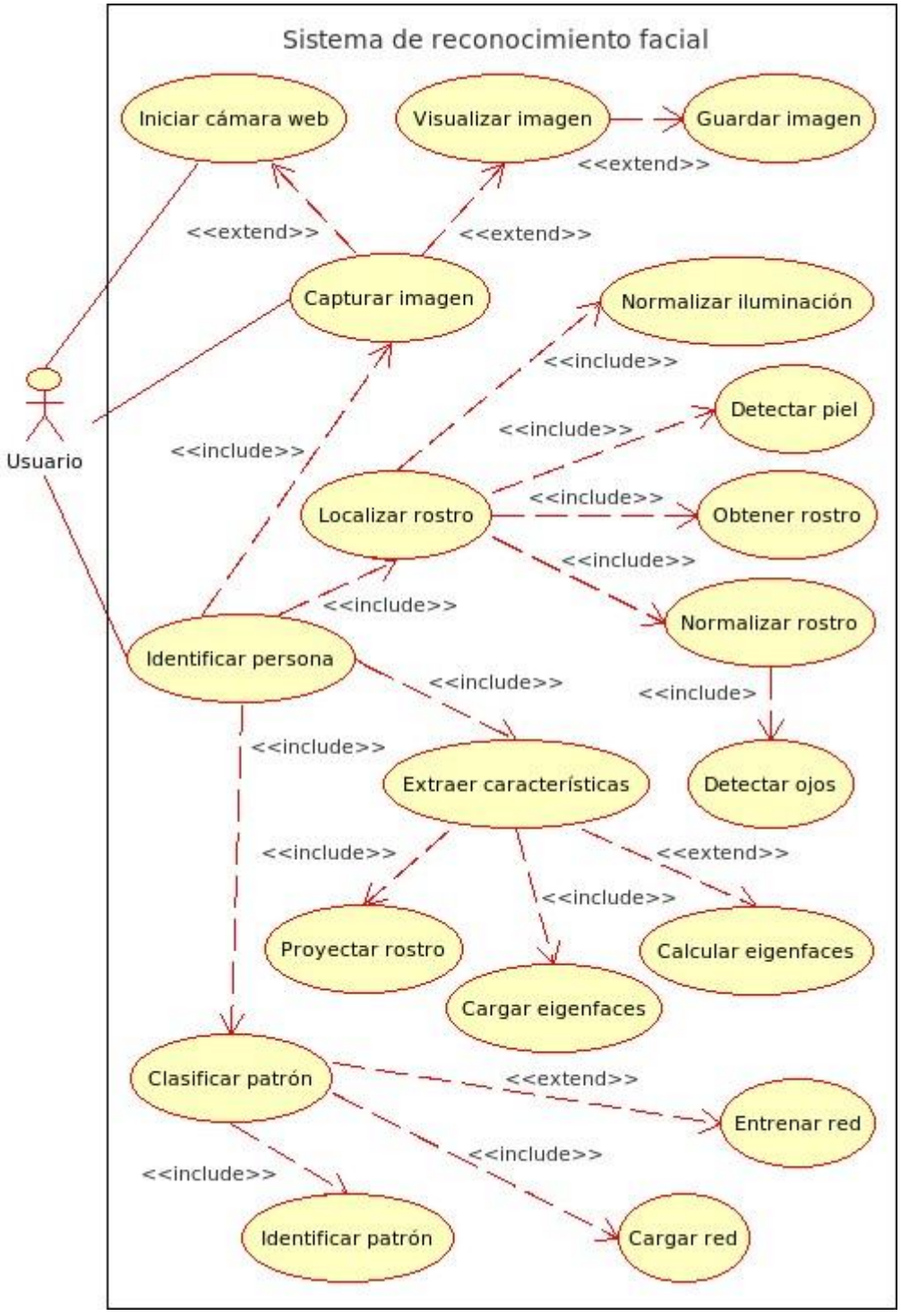


Figura 3.2: Diagrama de casos de uso detallado.

Caso de uso: Localizar rostro

Actor: -
Descripci´on: Obtiene una subimagen facial normalizada.
Curso normal:
1) Se normaliza la iluminaci´on en la imagen. 2) Se detectan las regiones de piel. 3) Se aplica una plantilla el´ıptica para ubicar la cara y se recorta una subimagen de taman˜o definido. 4) Se detectan los ojos a partir de la imagen obtenida. 5) Se normaliza el rostro a partir de los ojos y se recorta la imagen original nuevamente.
Caso de uso: Extraer caracter´ısticas
Actor: -
Descripci´on: Obtiene un vector representativo de la imagen facial para clasificarlo.
Curso normal:
1) Se cargan las eigenfaces y la imagen media. 2) Se centra la imagen facial sustray´endole la imagen media. 3) Se proyecta la imagen centrada multiplic´andola por cada una de las eigenfaces, generando un punto en el espacio de caras.
Cursos alternativos:
1.A) Las eigenfaces y la imagen media no est´an calculadas. 1.A.1) Se calculan las eigenfaces y la imagen media. 1.A.2) Se guardan las eigenfaces y la imagen media.
Caso de uso: Calcular eigenfaces
Actor: -
Descripci´on: Calcula las eigenfaces de todas las im´agenes en la base de datos.
Curso normal:
1) Se calcula la imagen media de todas las im´agenes en la base de rostros. 2) Se calculan las eigenfaces. 3) Se guardan las eigenfaces y la imagen media.

Caso de uso: Clasificar patr´on
Actor: -
Descripci´on: Clasifica un patr´on para identificar a la persona.
Curso normal:
1) Se carga la red. 2) Se aplica el patr´on a la red devolviendo como resultado la identificaci´on de la persona.
Cursos alternativos
1.A) La red no est´a entrenada. 1.A.1) Se entrena la red. 1.A.2) Se guarda la red.
Caso de uso: Entrenar red
Actor: -
Descripci´on: Entrena la red neuronal para realizar la clasificaci´on de patrones.
Curso normal:
1) Se proyectan todas las im´agenes de la base de rostro al espacio de caras obteniendo los patrones correspondientes. 2) Se entrena la red mediante los patrones obtenidos. 3) Se guarda la red entrenada.

3.3 Diagrama de clases

Los diagramas de clases son los m´as usados en el modelado de sistemas orientados a objetos, los cuales muestran un conjunto de clases y sus relaciones. Se utilizan para modelar la vista de disen˜o est´atica de un sistema, lo que implica modelar el vocabulario del mismo, modelar colaboraciones o modelar esquemas.

As´ı como son importantes para visualizar, especificar y documentar modelos estructurales, tambi´en lo son para construir sistemas ejecutables, aplicando ingenier´ıa directa e inversa.

En la Figura 3.3 se puede observar el diagrama correspondiente al sistema de reconocimiento facial. Como se puede notar, se definieron distintas clases que permiten realizar tanto la captura de im´agenes como el reconocimiento de una persona.

Las clases *Application*, *MainFrame*, *WebCam* y *SnapshotDialog* son extensiones de clases contenidas en la librer´ıa *wxWidgets*[1], las cuales no son mostradas en el diagrama a modo de simplificar el mismo. Esta librer´ıa permite crear interfaces de usuario haciendo uso de las bibliotecas ya existentes en el sistema.

La clase *Application* (extensi´on de wxApp) representa la aplicaci´on misma. Es usada para definir y obtener propiedades de ´esta, como tambi´en para iniciarla.

La clase *MainFrame* (extensi´on de wxFrame) corresponde a la ventana de la aplicaci´on. Gestiona las funcionalidades del menu´ y los botones incluidos en ella. Inicializa la c´amara web, captura una imagen o identifica un usuario, segu´n la acci´on ejecutada por el usuario. Adem´as, visualiza lo que la c´amara web captura.

La clase *WebCam* (extensi´on de wxWindow) se encarga de manejar el dispositivo de captura de im´agenes. Detecta si se encuentra conectada una c´amara a la PC, inicializa y detiene su controlador, como tambi´en captura im´agenes a ser visualizadas en la vista previa por la clase MainFrame.

La clase *SnapshotDialog* (extensi´on de wxDialog) se utiliza para visualizar la imagen capturada por el usuario en una ventana de di´alogo. Permite guardar la misma en el disco r´ıgido o descartarla.

Para el procesamiento b´asico de las im´agenes, se utiliza la clase *CImg*[2], siendo eficiente y simple de usar. Es libre, de c´odigo abierto y es distribuida bajo las licencias CeCILL-C (similar a la LGPL) o CeCILL (similar a la GPL). Permite cargar y guardar im´agenes en

[1] http://www.wxwidgets.org/

[2] http://cimg.sourceforge.net

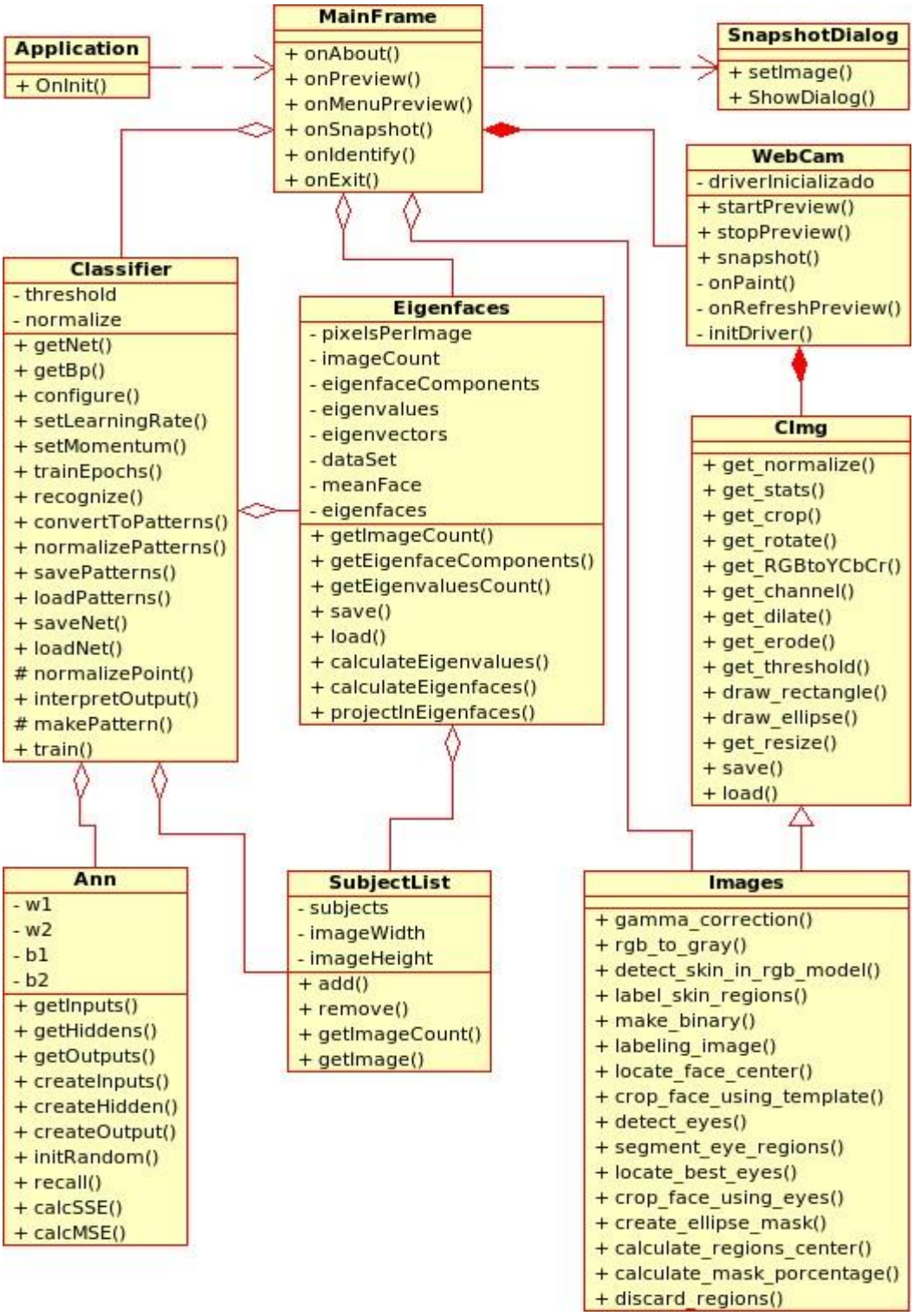
formatos ppm, jpg y otros, manipular im´agenes como matrices, aplicar operaciones de dilataci´on y erosi´on, rotar y cortar im´agenes, efectuar transformaciones a diferentes espacios de color, etc..

La clase *Images* hereda todas las operaciones de CImg, y adem´as incluye funciones para normalizar la iluminaci´on, segmentar la piel, localizar el rostro en una imagen y normalizar la geometr´ıa del mismo.

La clase *SubjectsList* se encarga de cargar las im´agenes de la base de rostros.

La clase *Eigenfaces* contiene las funciones necesarias para generar el espacio de caras. Calcula, guarda y carga la imagen media y las eigenfaces del

Figura 3.3: Diagrama de clases.



conjunto completo de rostros, y proyecta una imagen facial a este espacio.

La clase *Ann* implementa un perceptr´on multicapa de tres capas: entrada, oculta y salida. Construye la red segu´n el nu´mero de entradas y la cantidad de nodos en la capa oculta y salida e identifica un patr´on aplicado a la entrada de la misma.

Por u´ltimo, la clase *Classifier* se encarga de generar los patrones de entrenamiento haciendo uso de la clase Eigenfaces, proyectando cada imagen facial al espacio de caras. Con estos patrones, entrena la red neuronal mediante el algoritmo de retropropagaci´on del error, y permite guardar y cargar tanto la red entrenada como los patrones mismos.

3.4 Diagrama de secuencia

Los diagramas de secuencia son usados para modelar los aspectos din´amicos del sistema. Muestran las interacciones entre los distintos objetos participantes y los mensajes que intercambian. Estos diagramas se forman colocando los objetos y actores que interactu´an en el eje horizontal, y en el eje vertical se colocan los mensajes intercambiados en orden temporal, desde arriba hacia abajo. El objeto que inicia la interacci´on suele ubicarse a la izquierda y los objetos subordinados a la derecha.

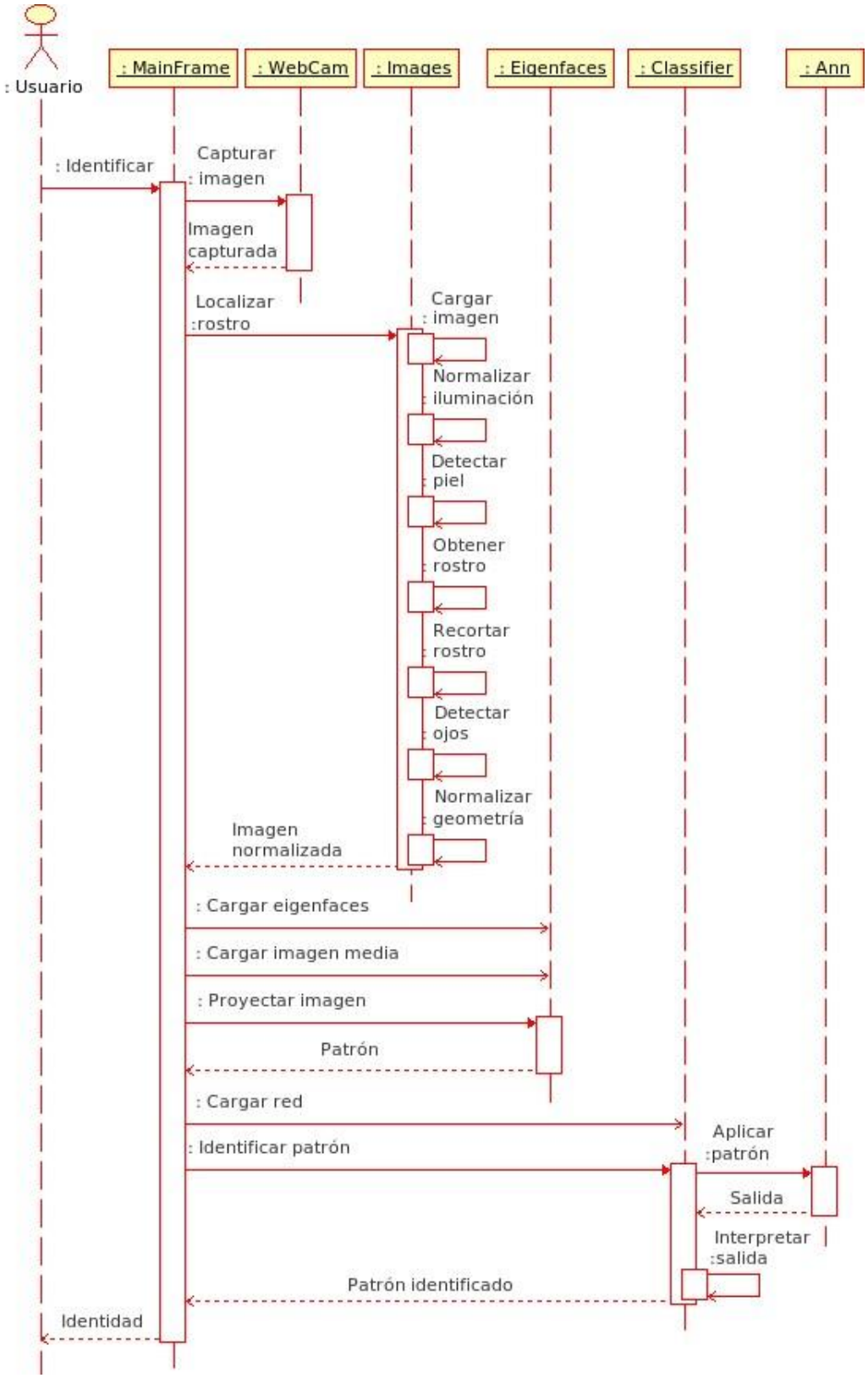
En la Figura 3.4 se puede observar el diagrama correspondiente al momento en que un usuario intenta identificarse ante el sistema. El usuario inicia el reconocimiento al solicitarle a la clase MainFrame que lo identifique. Esta´ instancia a la clase WebCam para que tome una foto del usuario. Con la imagen capturada, la clase Images la carga, la realza y, a partir de la detecci´on de la piel, ubica el rostro

obteniendo como resultado una subimagen que corresponde a la cara normalizada.

Una vez localizado el rostro, se instancia a la clase Eigenfaces que carga las eigenfaces y la imagen media. Toma la cara normalizada, la transforma en un vector y a este u´ltimo, lo proyecta al espacio de caras obteniendo un patr´on a clasificar. Luego, con la clase Classifier se carga la red neuronal entrenada y se aplica el patr´on obtenido a la entrada de la misma. Finalmente, se informa el resultado de la identificaci´on al usuario.

3.4. Diagrama de secuencia

Figura 3.4: Diagrama de secuencia de identificaci´on de persona.



3.5 Interfaz de usuario

A partir de los requerimientos especificados y los distintos diagramas expuestos, se desarroll´o la interfaz de usuario implement´andose tres funcionalidades: 1) iniciar la c´amara web, 2) capturar una imagen y 3) identificar un usuario. Esta se desarroll´o en lenguaje C++ utilizando la librer´ıa wxWid-´ gets. Como se indic´o anteriormente, esta librer´ıa permite crear interfaces de usuario basadas en las bibliotecas ya existentes en el sistema, con lo que se obtiene una apariencia muy similar a una aplicaci´on nativa. De esta manera, resulta muy portable entre distintos sistemas operativos, estando disponibles para Windows, OS X, Linux y Unix. Es libre, de c´odigo abierto y distribuida bajo una licencia LGPL (similar a la GPL). En la Figura 3.5 se puede observar la interfaz desarrollada. Aqu´ı se pueden notar tres botones en la parte inferior: Vista Previa, Capturar e Identificar. Cada bot´on realiza una acci´on como se describe a continuaci´on:

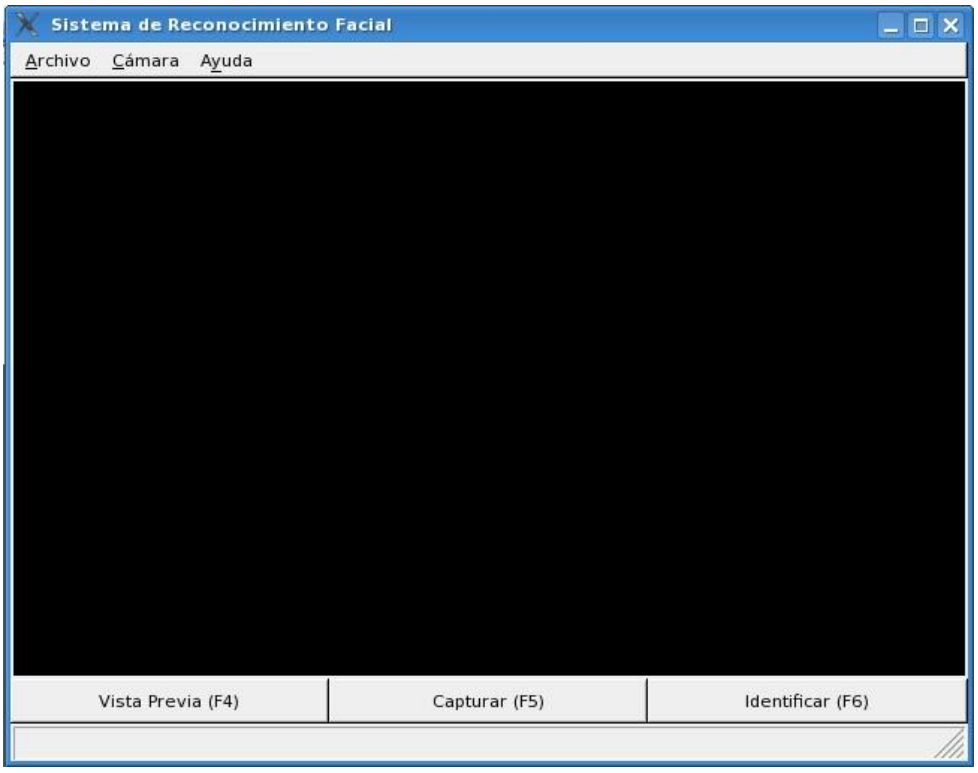


Figura 3.5: Interfaz de usuario.

- Vista Previa: inicializa la c´amara web, y reproduce en el recuadro negro lo que ´esta captura.
- Capturar: captura una imagen de la vista previa visualiz´andola en otra ventana, en la cual se puede optar por guardar la imagen tomada (bot´on Guardar) o descartarla (bot´on Cancelar). Un ejemplo se muestra en la Figura 3.6.
- Identificar: captura una imagen de la vista previa, aplica las tres etapas para el reconocimiento e indica qui´en es el usuario que se encuentra frente a la c´amara (Ver Figura 3.7).

Figura 3.6: Captura de imagen.

Estas mismas opciones tambi´en pueden ser accedidas desde el menu´, en la opci´on Archivo. Adem´as, se debe mencionar que la interfaz funciona sobre los sistemas Windows y Linux.

Figura 3.7: Identificaci´on de usuario.

Las im´agenes son capturadas utilizando la c´amara web en una resoluci´on de 320 x 240 p´ıxeles. Sobre Linux, se us´o el controlador no oficial *spca*[3], el cual soporta una variedad de dispositivos. Una lista de ´estos puede ser encontrada en su p´agina web. Bajo Windows, se requiri´o del controlador de la c´amara web utilizada.

CAP´ITULO 4

Un sistema de RF consta de tres etapas, como se mencion´o anteriormente: (1) detecci´on del rostro, (2) extracci´on de caracter´ısticas y (3) clasificaci´on. En la primera etapa se localiza el rostro en una imagen eliminando todo lo que sea fondo. Una vez obtenida la cara, el segundo paso se encarga de extraer caracter´ısticas representativas de ´esta, formando as´ı, un vector patr´on que se utiliza en la etapa de clasificaci´on para la identificaci´on o verificaci´on de la identidad de una persona con respecto a los usuarios registrados en el sistema. En la Figura 4.1 se puede observar una configuraci´on gen´erica de un sistema de RF.

A partir de la interfaz de usuario desarrollada, se capturaron un conjunto de im´agenes conformando una base de rostros denominada *SINCIDISI*, con la cual se llevaron a cabo las pruebas del sistema. Luego, se desarrollaron

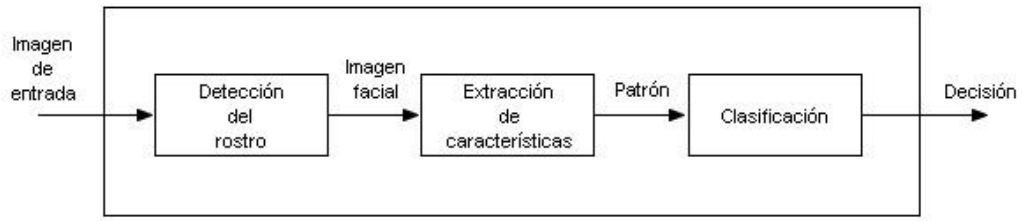


Figura 4.1: Configuraci´on gen´erica de un sistema de RF.

las tres etapas para el reconocimiento. Previo a la detecci´on del rostro, las im´agenes fueron sometidas a un proceso de normalizaci´on de la iluminaci´on con el objetivo de realzar las im´agenes capturadas. La detecci´on del rostro se bas´o en el algoritmo propuesto en [ML06], el cual combina detecci´on de piel, aplicaci´on de plantillas y detecci´on de caracter´ısticas faciales. Para la extracci´on de caracter´ısticas se proyecta la imagen facial a un espacio de caras, obteniendo un vector patr´on que refleja las propiedades m´as relevantes del rostro. Para la clasificaci´on se disen˜´o una estructura de red neuronal MLP, constituida por una capa de entrada, una oculta y otra de salida, entren´andola mediante el algoritmo de retropropagaci´on del error. Una vez entrenada, se le presentan los diferentes patrones y devuelve las identidades de los usuarios.

Por otro lado, se debe mencionar que el proceso de construcci´on se divide en dos fases, una de entrenamiento y una de clasificaci´on. En la primer fase, se captura un conjunto de N im´agenes a procesar, denominado *conjunto de im´agenes de entrenamiento* $I = \{I_1,...,I_N\}$, de un grupo de usuarios. En cada imagen se localiza el rostro obteniendo las im´agenes faciales F_i correspondientes, y se calculan los eigenvectores que mejor describen al conjunto facial resultante. Cada rostro F_i es proyectado al nuevo espacio obteniendo un vector caracter´ıstico $\mathbf{x}_i$. De esta manera, se genera un conjunto de patrones que es utilizado para entrenar la red neuronal. La fase de clasificaci´on toma una imagen de entrada I_q, localiza el rostro y proyecta ´este al nuevo espacio, en base a los eigenvectores calculados en la fase de entrenamiento, obteniendo el vector caracter´ıstico $\mathbf{x}_q$. Luego, este vector es aplicado a la entrada de la red neuronal obteniendo como resultado la identidad del usuario.

4.1 Base de datos de rostros SINCIDISI

Con la interfaz de usuario ya implementada se capturaron varias fotos en un ambiente real, creando as´ı la base de rostros *SINCIDISI*. Para ello, se instal´o una c´amara web en una PC del centro *sinc(i)*[4] de la FICH y en una PC del *CIDISI*[5] de la UTN FRSF (de all´ı su nombre SINCIDISI), coloc´andose una fuente de luz artificial para lograr una mejor iluminaci´on en las muestras. Las fotos fueron tomadas con fondo, pose, expresi´on facial y distancia a la c´amara totalmente natural. De esta manera, se obtuvo una cantidad de 448

4.1. Base de datos de rostros SINCIDISI

Figura 4.2: Ejemplos de todos los usuarios de la base de rostros SINCIDISI.

im´agenes de 21 personas.

A partir del conjunto de im´agenes capturadas, se realiz´o una selecci´on manual de las mismas a fin de descartar aquellas en las que no se tuvieran variaciones aceptables de pose e iluminaci´on, como

tambi´en aquellas con oclusiones parciales del rostro. Adem´as, se eliminaron im´agenes en las que se obtuvieron falsas localizaciones del rostro por la presencia de puertas, armarios u otros objetos con un color similar a la piel.

Por otro lado, se not´o que la cantidad de fotos para algunos usuarios era escasa y se decidi´o descartar ´estos. A partir de all´ı, se seleccionaron las im´agenes de mejor calidad, en una cantidad igual por cada usuario. De este modo, se obtuvo un conjunto final de 110 im´agenes, de un total de 11 personas (10 im´agenes por persona). Las fotos fueron almacenadas en formato PPM, con un taman˜o de 320 x 240 p´ıxeles. En la Figura 4.2 se exhibe un subconjunto de la base, con ejemplos de todos los usuarios.

(a) (b)

Figura 4.3: Normalizaci´on de la iluminaci´on. a) Imagen original, b) Imagen con iluminaci´on normalizada.

4.2 Metodolog´ıa

4.2.1 Detecci´on del rostro

Un cambio en la distribuci´on de la fuente de luz y en el nivel de iluminaci´on produce un cambio en el color de la piel en la imagen,

degradando el rendimiento de los sistemas de detecci´on de piel [KMB07]. Debido a que el algoritmo aplicado utiliza informaci´on sobre el color de piel para ubicar el rostro, es necesario aplicar un proceso de normalizaci´on que compense la luminosidad en las im´agenes capturadas por la c´amara web, ya sean im´agenes con poca o mucha iluminaci´on.

Para compensar la luminosidad se aplic´o una transformaci´on de potencia
(a cada canal del modelo RGB) dada por

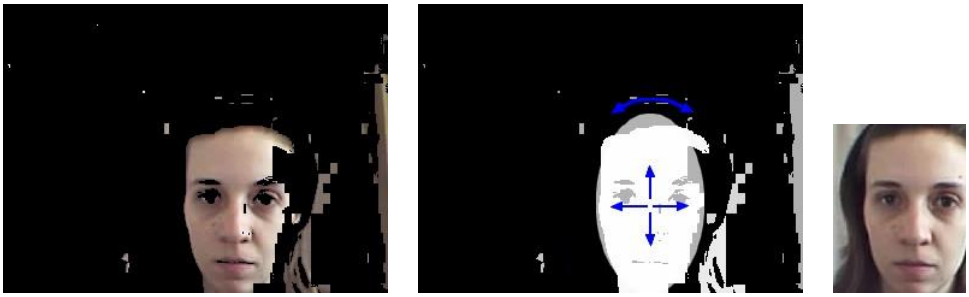
$$f(x,y) = f(x,y)^{\gamma},$$

donde γ es adaptada segu´n la cantidad de luz en cada imagen por

$$\gamma = c_1\mu_f + c_2.$$

Aqu´ı, c_1 y c_2 son constantes y μ_f es la media de la imagen a normalizar en escala de grises. Con ´esto se logra que γ sea directamente proporcional al brillo. En la Figura 4.3 se observa el resultado de aplicar esta transformaci´on.

A partir de la imagen normalizada, como primer paso, el algoritmo localiza el rostro en la imagen. Para ´esto, realiza una segmentaci´on de piel aplicando



(a) (b) (c)

Figura 4.4: Localizaci´on del rostro. a) Segmentaci´on de piel, b) Combinaci´on de plantilla sobre las regiones, c) Subimagen con el rostro localizado.

ciertas reglas definidas en [KPS97], basadas en que el color de piel se mantiene en un cierto rango del espacio de color RGB. De esta manera, logra localizar las posibles regiones caras en la imagen. La Figura 4.4(a) muestra el resultado de la segmentaci´on.

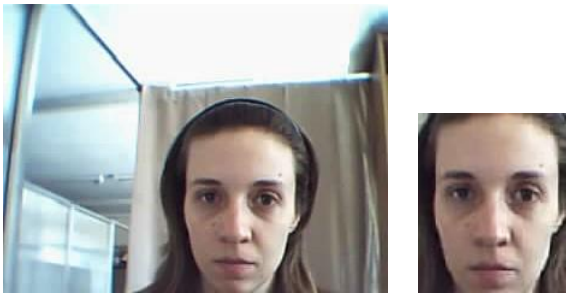
Una vez ubicadas las posibles regiones de piel, ´estas son etiquetadas mediante un algoritmo recursivo. Luego, se elige aquella regi´on que mejor se asemeje geom´etricamente a un rostro. Para ´esto, se aplican plantillas el´ıpticas a las diferentes regiones, haciendo ampliaciones, traslaciones y rotaciones de las mismas para lograr un mejor rendimiento ante variaci´on de pose y distancia a la c´amara (Figura 4.4(b)), y se elige aquella regi´on que contiene la mayor cantidad de p´ıxeles que caen dentro de la plantilla. Posteriormente se recorta una subimagen que contiene la regi´on seleccionada, o dicho de otro modo, el rostro del usuario (Figura 4.4(c)).

En segundo lugar, en base a la imagen recortada, el algoritmo normaliza el taman˜o y orientaci´on del rostro teniendo en cuenta la distancia entre ojos, posibilitando la comparaci´on de patrones estandarizados geom´etricamente. Primero detecta las posibles regiones de ojos mediante operaciones morfol´ogicas de dilataci´on y erosi´on aplicadas a la componente de luminancia Y del modelo YCbCr, filtrando aquellas que tengan demasiados o muy escasos p´ıxeles como para representar un ojo. En la Figura 4.5(a) y (b) se

observa la componente de luminancia Y y la segmentaci´on resultante respectivamente. Luego, busca el par de ojos candidatos tomando cada regi´on de la mitad izquierda, buscando su par a la derecha que caiga dentro de un rect´angulo predefinido (Figura 4.5(c)). Los pares de ojos candidatos resultantes son validados mediante una ponderaci´on de propiedades como posici´on en la imagen y distancia entre ellos, descartando aquellos que no superan un umbral preestablecido.

(a) (b) (c) (d)

Figura 4.5: Normalizaci´on de taman˜o. a) Componente Y, b) Posibles regiones ojos, c) Bu´squeda del par de ojos, d) Ojos detectados.



(a) (b)

Figura 4.6: Resultado de la detecci´on del rostro. a) Imagen con iluminaci´on normalizada, b) Imagen normalizada.

Con el par de ojos ganador (Figura 4.5(d)), se calcula la distancia ocular horizontal *Dx* y vertical *Dy* y el ´angulo de inclinaci´on del rostro $\alpha = arctan(Dy/Dx)$. Se corrige la inclinaci´on del rostro en base al ´angulo calculado y luego, en base a *Dx* y *Dy*, se recorta la subimagen facial normalizada a partir de la imagen realzada como se observa en la Figura 4.6.

4.2.2 Extracci´on de caracter´ısticas

En esta etapa se extraen las caracter´ısticas m´as relevantes de las im´agenes faciales aplicando el m´etodo *eigenfaces* explicado en la Secci´on 2.2.

Como se explic´o anteriormente, el primer paso es calcular las componentes eigenfaces que describen el conjunto de im´agenes faciales aplicando la ecuaci´on (2.13), escrita nuevamente para conveniencia del lector como

$$\mathbf{u}_l = A\mathbf{v}_l = \sum_{k=1}^{M} \mathbf{v}_{lk}\mathbf{\Phi}_k, \tag{4.3}$$

donde M es la cantidad de im´agenes en el conjunto de entrenamiento. De esta manera, se obtienen M eigenvectores $\mathbf{u}_l$ que conforman el conjunto base de las im´agenes faciales.

Dado que no se requiere la reconstrucci´on de las im´agenes, se utilizan u´nicamente $M^0 < M$ eigenfaces, las cuales corresponden a las m´as significativas segu´n el orden decreciente de los eigenvalores asociados. Estos M^0 eigenvectores contienen la mayor informaci´on del conjunto de datos.

Una vez calculadas las eigenfaces, tanto en el entrenamiento como en la clasificaci´on, las im´agenes faciales son proyectadas al eigenespacio mediante

$$x_k = \mathbf{u}_k^T (\mathbf{I} - \mathbf{\Psi}) \qquad \text{para } k = 1,...,M^0, \tag{4.4}$$

donde I representa una imagen, obteniendo el vector caracter´ıstico $\mathbf{x}$ correspondiente.

4.2.3 Clasificaci´on

Para el reconocimiento de patrones se disen˜´o una red neuronal MLP de tres capas (entrada, oculta y salida). Un esquema similar de esta estructura se puede observar en la Figura 2.5.

La entrada a la red se definió según las características extraídas de las imágenes faciales en la etapa anterior, dependiendo de la cantidad de eigenfaces utilizadas para la proyección en el espacio de caras. El número de nodos en la capa oculta se determinó mediante experimentación, donde se observó que cantidad era requerida para obtener un desempeño aceptable. La cantidad de salidas es el número de usuarios a reconocer por el sistema, cada salida representa a uno de ´estos. La salida cuyo valor es más alto corresponde al usuario reconocido.

Una vez establecido el diseño, la red se entrenó mediante el algoritmo de retro propagación del error, siguiendo las ecuaciones de la Sección 2.3.1. Esto´ se realizó en la fase de entrenamiento, en la cual se crearon patrones definiendo la relación entrada-salida para la red. Con dichos patrones, se estimuló la red a fin de minimizar el error entre salida y respuesta deseada. En la fase de

clasificación, la red entrenada es estimulada con el vector característico del usuario que intenta identificarse ante el sistema, y esta retorna la clase a la que pertenece o indica que no pertenece a ninguna de las clases conocidas

Conclusiones y desarrollos futuros

6.1 Conclusiones

En este trabajo se ha implementado un sistema grafico de reconocimiento facial que contempla todas las etapas requeridas, desde la interfaz gráfica de usuario para la captura de la imagen hasta

su clasificación, pasando por la localización del rostro y la extracción de características, identificando al usuario frente a la cámara.

El sistema realizado cumple con los objetivos propuestos logrando un rendimiento aceptable dado el ambiente en el cual se presentó el problema. Este´ presenta una interfaz gráfica sencilla que permite operar fácilmente tanto a un administrador del sistema a la hora de tomar fotos como a un usuario que intenta identificarse. A través de los experimentos se demostró que el clasificador MLP se desempeñó mejor frente a una red RBF y a la técnica del vecino más cercano, logrando un mayor reconocimiento.

6.2 Desarrollos futuros

Con la idea de mejorar el desempeño general del sistema, una mejora estaría dada por un montaje dedicado en el lugar de instalación del mismo. Esto implicaría colocar un fondo fijo de color blanco y un flash como fuente de´ luz para lograr luminosidad constante permitiendo, de esta manera, obtener una mejor calidad en las imágenes capturadas. Por otro lado, una ampliación del sistema estaría dada por el desarrollo de un módulo de administración de usuarios a fin de automatizar el proceso de carga de los mismos. Este incluiría funcionalidades para el registro de nuevos´ usuarios, dar de baja aquellos que ya no se requieran en el sistema, así como también asociar las imágenes capturadas a los usuarios correspondientes.