UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELECTRICA

Análisis y Detección de Características Faciales Usando Aprendizaje Estadístico

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRICISTA

OSCAR ALFONSO SANHUEZA RIVEROS

PROFESOR GUÍA:
JAVIER RUIZ DEL SOLAR

MIEMBROS DE LA COMISIÓN:
HÉCTOR MILER AGUSTO ALEGRIA
RODRIGO ANDRÉS VERSCHAE TANNENBAUM

SANTIAGO DE CHILE
MARZO 2008

RESUMEN DE LA MEMORIA
PARA OPTAR AL TITULO DE
INGENIERO CIVIL ELECTRICISTA
POR: OSCAR ALFONSO SANHUEZA RIVEROS
FECHA: 06 de Marzo, 2008
PROF. GUIA: SR. JAVIER RUIZ DEL SOLAR

"ANALISIS Y DETECCION DE DE CARACTERISTICAS FACIALES USANDO APRENDIZAJE ESTADÍSTICOS"

En el mundo de hoy, muchas aplicaciones requieren interactuar automáticamente con sus usuarios, y la necesidad de que estas aplicaciones puedan conseguir de forma automática información acerca del usuario hacen que la clasificación de características faciales sea muy importante. Así, estos módulos de clasificación se pueden ocupar en múltiples y diversas aplicaciones.

El objetivo general del presente trabajo es la clasificación de características faciales usando algoritmos de aprendizaje estadístico, esto significa poder detectar y clasificar el mayor número posible número de características que se pueden encontrar en una cara usando solo ejemplos de imágenes de estas mismas, sin utilizar a priori ninguna información de las características dadas.

En el presente trabajo se desarrollaron detectores y clasificadores de las características que se consideran más significativas, y en general, las primera en que una persona se fija al ver un rostro, así es como se decidió construir clasificadores de barba, bigotes y lentes que distinguieran si una persona en una foto posee o no barba, bigotes y/o lentes. Además de estas características más visuales, se desarrolló un clasificador de edades en cuatro tramos, niños, jóvenes, adultos y ancianos, que lograra dar como respuesta, usando su confidencia, el grado de certeza de la clasificación. Como objetivo secundario pero no menos importante, se desarrollo un detector de bocas, que entrega la posición central de la boca en las caras detectadas.

Excelentes resultados se reportaron en la detección de boca, con tasas de detección superiores al 99% y errores comparables al error proveniente del marcado manual de las bocas. El clasificador de lentes obtuvo también excelentes resultados, con tasas de detección del 95% para bases de datos con ambientes controlados y del orden del 90% para bases con ambientes no controlados. Clasificadores de barbas y bigotes luego de usar el detector de boca obtuvieron muy buenos resultados, con tasa de detección por sobre el 95% en bases de datos con ambientes no controlados. Por su parte, la nueva arquitectura diseñada para el clasificador de edad, que ocupa la información de las confidencias para da una respuesta más general, funciono de buena forma, aunque podrían hacerse mejoras en este último punto.

Se concluyó finalmente que los clasificadores Adaboost elegidos en este trabajo para hacer las clasificaciones reportan excelentes resultados, y no se duda que puedan también hacerlo en otros tipos de aplicaciones de similares características. Comparando con otros trabajos, se ha visto que el trade-off entre las tasas de detección y el tiempo de clasificación son excelentes. Ambas características ocupadas Rectangulares y LPBm reportan resultados positivos en distintos tipos de clasificadores, así, ninguno de las dos se descarta al momento de hacer un nuevo clasificador.

Agradecimientos

“No hay deuda que no se pague ni plazo que no se cumpla”…

.. a todos los que me ayudaron recordándome este viejo refrán

Índice General

Índice de Figuras

Índice de Tablas

1. Introducción

¿Es posible imaginar, por ejemplo, un centro comercial lleno de personas, en el cual, al subir las escaleras mecánicas, se observen no los simples avisos publicitarios, si no que, avisos que vayan dedicados específicamente hacia la persona en la escalera de acuerdo a las características que esta presenta? La respuesta es sí, es posible imaginarlo si existe en medio un sistema que detecte el rostro de la persona y lo clasifique de acuerdo a sus características. Así, si la persona es un niño, podrá mostrarse un video relacionado con la sección juguetes, si la persona es un joven, quizás con la sección música o si la persona es adulta con la sección libros. Más aún, que tal si también en dicha clasificación se estudia si la persona lleva o no lentes, entonces, podrá mostrarse un aviso publicitario de una tienda de lentes y si nos los lleva y es época de verano, la última moda en lentes de sol, o bien si el sujeto presenta barba y bigotes, mostrar el aviso de la nueva afeitadora.

Qué tal si un día, se decide sacar una foto como las del pasaporte, en una de las típicas máquinas automáticas de fotos, la cual, por alguna razón debe ser sin lentes, se llega a la máquina y luego que se tiene la foto en las manos, se da cuenta que la foto no sirve, pues se tenían los lentes puesto. ¿Podría ser posible que la máquina hubiera advertido de la presencia de los lentes en el rostro de la persona?, nuevamente la respuesta es sí, es posible siempre y cuando la máquina integrara un clasificador que estudiaría si la persona lleva lentes o no e hiciera una advertencia al usuario antes de sacar las fotos.

Resultado de aplicaciones como estas y otras es que el estudio de las características faciales alcanza cada día una importante posición dentro de muchas interfaces humano-computador.

1.1. Objetivos Generales

El objetivo general de este trabajo es el análisis y detección de características faciales basadas en algoritmos estadísticos. Ambos, el análisis y la detección se efectúan usando clasificadores los cuales son basados en algoritmos de aprendizaje estadístico. El que sean basados en algoritmos de aprendizaje estadísticos, debe entenderse como, la propiedad de

entrenarse automáticamente (aprendizaje) dejando de lado completamente la utilización de características que se pueden pre visualizar en las imágenes.

1.2. Objetivos Específicos

Los objetivos específicos que esta memoria debe cumplir son:

i) La construcción de bases de datos lo suficientemente grandes y diversas que permitan entrenar clasificadores estadísticos para detectar diversas características faciales. Esto implica, la búsqueda de bases de datos que se encuentren disponibles y que puedan servir para estos objetivos y además, la creación de bases de datos propias con imágenes extraídas de la red.

ii) El diseño e implementación de un detector de Boca, el cual, ayude a la extracción de diversas áreas de la cara para su posterior clasificación. Este debe entregar como salida el área dentro de la cara donde con mayor seguridad se encuentra la boca del individuo del cual se tiene una imagen.

iii) El diseño y implementación de un clasificador de Barba, Bigotes y Lentes, el cual, entregue como resultado si el rostro de la persona contenida en una imagen posee o no, Barba, Bigotes y/o Lentes.

iv) Para finalizar, el diseño y implementación de un clasificador de Edad, el cual este dividido en 4 clases, niños, jóvenes, adultos y ancianos. Que además de entregar un grado de pertenencia a una de las clases (como los clasificadores de barba, bigotes y lentes), tenga como salida, cierto grado de seguridad en su resultado. De esta forma se evita el hacer clasificaciones completamente erróneas cuando se tiene un problema con alto grado de traslape, como es el de la edad.

1.3. Motivación

En el mundo de hoy, se están haciendo cada vez más comunes y necesarias las interfaces humano-computador, como las presentadas previamente, e imaginarse un sistema que hace pocos años era sólo visto en una película de Hollywood hoy día ya no es tan difícil. Debido a diversos factores, el uso de interfaces automáticas, en que la persona tenga que interactuar de forma rápida e inobstaculizada se hacen cada vez más común. La motivación principal de este trabajo de memoria es el estudio de las características faciales, con el propósito de usarlas por ejemplo en interfaces humano computador. Claro está, existen otros muchos usos posibles para el estudio de las características faciales, como seguridad, indexación automática o también como ayuda al siempre difícil objetivo del reconocimiento facial.

Las características que se busca detectar y clasificar en este trabajo de memoria, corresponden a las características más comunes que podemos encontrar en una cara, y muchas veces, las características que como humanos, tendemos a memorizar al ver un rostro por primera vez. Estas características pueden ayudar a discernir entre distintas características de la persona, que luego puede usarse con distintos objetivos.

Existen, en los clasificadores y detectores antes mencionados múltiples dificultades a la hora de discernir, la más compleja, es el alto grado de variedad que presentan. Por ejemplo, los lentes, existen infinitos tipos y estilos de lentes, lo que hace pensar, será una dificultad para hacer una clasificación automática de estos. Para que más mencionar las barbas, que en la actualidad, han pasado de formar características religiosas o políticas a una especie de accesorio dentro de los rostros de hombres. Si bien la Real Academia Española las define como “pelo ubicado debajo de la zona de la boca”, existen diversos tipos y posiciones de estas.

Luego, si se aborda el problema de la clasificación de Edad, se encuentra con un problema mucho más difícil. Si se compara con la clasificación de barba, en la cual existe cierto

tipo de traslape, es decir, que dos personas distintas puedan dar respuestas diferentes para la clasificación de barba en un sujeto respectivo. El grado de traslape encontrado en la clasificación de edad es mayor aun. Se hace muy difícil diferenciar si una persona tiene 32 o 27 años, o si ya ha pasado de la adultez a la vejez, o bien, si aun es un niño o ya es un joven. Es por esto que es imposible suponer que la clasificación automática de esta característica de resultados cercanos al 100%. Sin embargo, el intentar dar respuestas con grado de seguridad, hacen del clasificador un producto mucho más tangible y utilizables dentro de una futura aplicación.

1.4. Estructura del documento

El presente documento se encuentra estructurado en 6 capítulos distintos. A continuación se entrega una breve descripción de cada uno de estos.

El capítulo uno entrega una introducción y visión general del problema, así como las motivaciones que indujeron en la realización del proyecto y los objetivos que se desean alcanzar en él.

En el capítulo dos se presenta la arquitectura general de un sistema de detección y clasificación de caras usando algoritmos estadísticos, así como las opciones previamente tomadas para su uso en el presente trabajo. Se da también una pequeña explicación del por qué de estas decisiones.

En el capítulo tres se entrega una visión más detallada de cómo se construyen los clasificadores propuestos en el capitulo anterior y utilizados durante todo el presente trabajo, así como la base teórica del funcionamiento de estos.

En el capítulo cuatro se describe como se lleva a cabo la construcción y entrenamiento del detector de boca y los clasificadores mencionados paso por paso. Incluyendo cada uno de los problemas que se tuvieron que superar con el objetivo de obtener buenos resultados

En el capítulo cinco se hace un análisis de los resultados. Además se validan estos comparándolos con resultados obtenidos por otras personas o grupos de trabajo. Si trabajos

anteriores no han sido encontrados se intenta dar un evaluación por el sentido común del resultado que deberían tener los clasificadores antes mencionados.

En el capítulo seis se concluye si se han logrado o no los objetivos propuestos. Se propone también si lo expuesto a lo largo de este trabajo es o no una buena manera de clasificar y detectar características faciales.

2. Arquitectura del sistema

En el siguiente capítulo se dan a conocer las características generales de la detección de caras así como los procedimientos utilizados particularmente en este trabajo para la detección de características faciales. Se comienza por definir la detección de caras como tal y luego su posterior clasificación. No se da en este capítulo, ninguna explicación formal de los métodos pero si una visión general de su funcionamiento y el porqué de las elecciones hechas para su posterior uso.

2.1. Detección de Caras

2.1.1. Definición

La detección de caras es una tarea computacional que intenta, de manera rápida y robusta, detectar caras en imágenes o videos. Como tal, es el primer y uno de los más importantes pasos en la tarea del Análisis de Caras, tal como el reconocimiento facial, el reconocimiento de expresiones faciales, el reconocimiento de género y otros.

Con la rápida evolución de los sistemas automáticos a nivel mundial, la detección y análisis de cara se está haciendo una herramienta necesaria. Es por esto que, pese a empezar en teoría en los años 70, ha tenido un enorme auge en la última década en conjunto con el exponencial incremento de memoria y capacidad de proceso en los computadores.

Las caras, al no ser un objeto definible facilmente, y además por ser tan variables, enfrentan entonces una doble dificultad al intentar detectarlas. Es por esto que, el detector de caras debe ser lo suficientemente robusto como para detectar caras de distintos tamaños, edades, géneros, colores, con distintos cabellos, lentes, barbas o bigotes. Sin dejar de lado el que estas puedan tener distintos fondos y iluminaciones.

Es así que dada una imagen cualquiera, la detección de caras se define como la determinación de la posición y tamaño de todas las caras que en ella pueda haber. Cuando se habla de detección en tiempo real o video, entonces solo se intenta detectar las caras en cada uno de los frames del video.

Además de las dificultades ya señaladas, también existe la dificultad espacial. O sea, cualquier objeto que se desee detectar en una foto o un video, es una proyección bidimensional de un objeto tridimensional. Así, las rotaciones que este pueda presentar es un gran reto para la detección. Las distintas rotaciones se pueden clasificar en:

- Rotaciones en el plano

  Rotaciones del objeto entre 0 y 360 grados entorno a su centro.

- Rotaciones Verticales fuera del plano

  Al tratarse de caras los objetos a detectar, es también importante si la persona en la foto está mirando hacia arriba o hacia abajo, o sea, tiene una rotación vertical fuera del plano. Este tipo de rotación se hace importante cuando se está detectando caras de frente.

- Rotaciones Horizontales fuera del plano

  Es la rotación que produce cuando se ven los objetos de perfil o de frente, si el objeto a detectar no es uniforme, representan una gran dificultad para la detección.

Ejemplos de cada una de las rotaciones se pueden observar en la Figura 1.

Rotación Vertical Fuera del Plano | Rotación Horizontal Fuera del Plano

Rotación en el Plano

Figura 1 "Ejemplos de Rotaciones de Caras"

2.1.2.Arquitectura del Detector de Caras

Los diferentes tipos de detectores, particularmente los de caras, se separan en dos grandes niveles de arquitectura: los basados en las características del objeto a detectar, en este caso características predefinidas en las caras y los basados en imágenes de ejemplos, en este caso, de caras y no caras.

Como se puede apreciar en el esquema de la Figura 2, los primeros, se pueden también separar en: Análisis de Bajo Nivel, de Componentes y Modelos Geométricos Activos. El Análisis de Bajo Nivel es el que ocupa la información de los pixeles, por ejemplo intensidad y color. El segundo busca componentes geométricos que se puedan comparar con el objeto a detectar. Y el tercero, busca ajustar modelos geométricos que se asocien a lo que se desea detectar en la imagen, por ejemplo una elipse a la cara, círculos a los ojos y una línea a la boca. Algunos de los detectores combinan estos métodos obteniendo buenos resultados. Con la desventaja que se presenta al momento de tener fondos variables, donde algunas estructuras podrían parecer

mucho a una cara. Como el trabajo presentado en [1] donde primero usando un detector de piel se separa la piel de las imágenes y luego usando un detector de blobs con forma de elipse se separan las caras, para luego detectar ojos y boca.

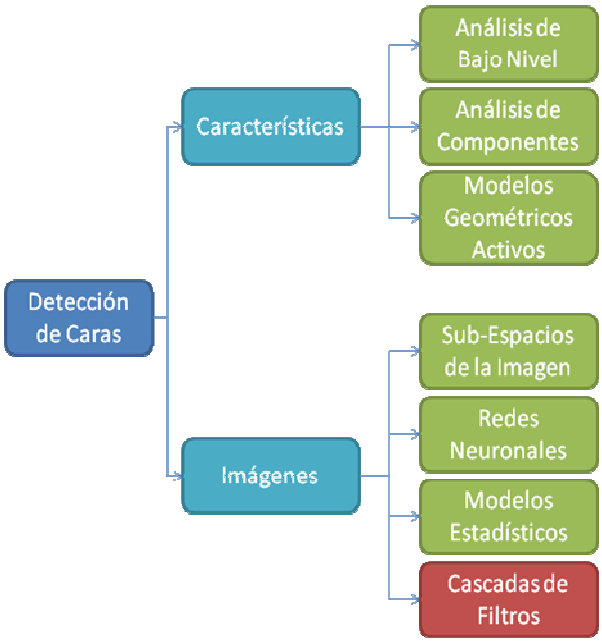


Figura 2 "Clases Arquitectura de Detectores"

Por otro lado, el segundo tipo de detectores de objetos, se basa en las imágenes tales de la clase que se requiere detectar, positiva (en este caso caras) y de la clase negativa (en este caso no-caras). Estos métodos no usan características de las caras, sino que, se forman sus propias características que diferencian de mejor forma las caras y no-caras para la detección. Resulta así mucho más robusto pata ambientes no controlados, o sea, con fondos confusos y malas o distintas iluminaciones. Además, pueden ser totalmente invariantes al color.

El método elegido en este trabajo de memoria fue el de Cascadas de Filtros, desarrollado por Rodrigo Verschae en [2]. El método se basa en AdaBoost introducido en [3]. Ha sido utilizado por su efectividad y velocidad, no solo en el contexto de detección de caras. Su simplicidad lo hace sumamente efectivo y preciso.

2.1.3. Clasificadores en Cascada para la Detección de Caras

Boosting, algoritmo en que se basa AdaBoost, se refiere a una manera general y probablemente efectiva de producir reglas de alta precisión (clasificadores "Fuertes") combinando reglas moderadamente imprecisas (clasificadores "Débiles"). Generalmente cada clasificador débil, no es más que una regla simple que puede usarse para generar cualquier tipo de clasificadores, pero que por sí sola carece de total certeza.

Así bien, una cascada combinada de clasificadores AdaBoost (Arquitectura en Cascada de Filtros), construido por etapas, constituye un clasificador AdaBoost adaptado, que permitirá hacer detecciones de mayor y mejor nivel. Cada una de estas etapas se interrelaciona a través de un método de entrenamiento llamado Boostraping. Esto se explicara más detenidamente en el capítulo 3.

El clasificador Adaboost, luego de ser construido y entrenado, no es por sí mismo un detector de caras. Es solo una forma de diferenciar caras de no caras. Para conformar el detector se necesita de otras etapas, como se puede observar en la Figura 3.

Existen 5 módulos que se deben llevar a cabo antes de poder detectar las caras en una foto cualquiera. El primero que se debe hacer es un Análisis de Multiresolución con el cual se pueden extraer todas las caras a diferentes escalas. La resolución de la imagen es disminuida en un factor de 1.2 hasta que la última imagen es más pequeña (de alto o de ancho) que la ventana a procesar. Luego, se encuentra el modulo de Extracción de Ventanas, que, como su nombre lo indica, extrae ventanas de 24 × 24 pixeles (en el caso del detector de caras) de todas las imágenes de salida del modulo anterior. Dependiendo de la aplicación no todas las ventanas de este tamaño que se encuentran en las imágenes son extraídas, algunos pixeles pueden ser "saltados" sin alterar los resultados.

En seguida, cuando ya se han obtenidos todas las ventanas a clasificar, se puede aplicar algún método de pre-procesamiento para obtener por ejemplo, mejor invariancia a la luminosidad. Finalmente, cada una de las ventanas se entrega al clasificador en cascada Adaboost, donde son analizadas entregando un resultado binario. Luego de que todas las posibles caras han sido clasificadas, se procede al módulo de detección de traslapes, donde las ventanas clasificadas como caras son analizadas y fusionadas para determinar la posición y tamaño de la detección final. En este último modulo, la confidencia[1] de cada una de las caras detectadas se usa para la fusión de estas. O sea, si el número de ventanas traslapadas es mayor que un cierto valor, y si el volumen de detección[2] de las ventanas detectadas en una posición dada es mayor que cierto valor, solo entonces, la ventana es considerada como una detección verdadera.

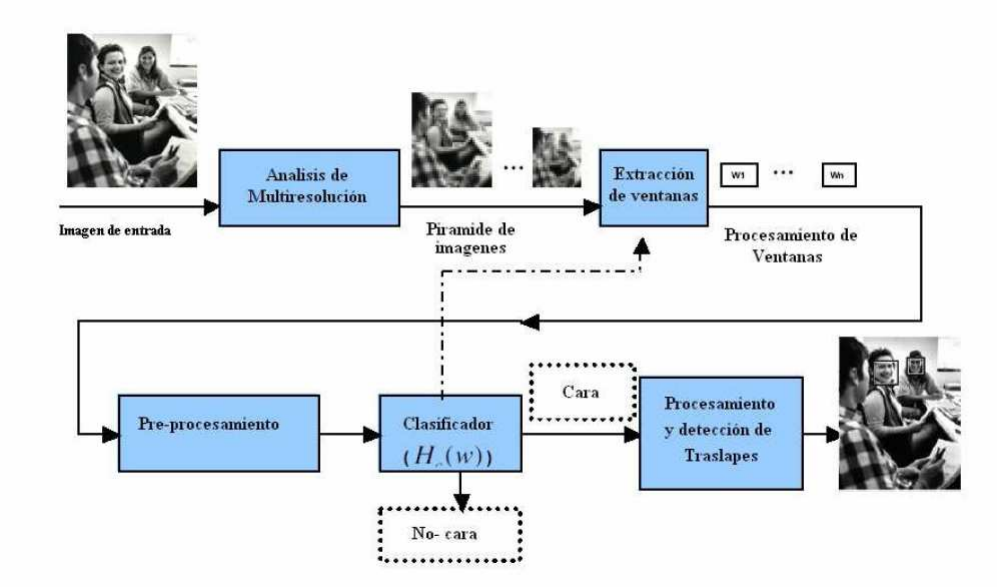


Figura 3 "Arquitectura del Detector de Caras [2]"

Los clasificadores AdaBoost, por su construcción, pueden ser utilizados tanto en la detección de caras dentro de una imagen dada, como también para la posterior clasificación de

[1] Confidencia de un Clasificador: Es el grado de seguridad o certeza que tiene este al hacer una clasificación, esto puede resultar importante al momento de analizar mejor su efectividad.

[2] El Volumen de Deteccion corresponde a la suma de todas las confidencias correspondiente a un set de ventanas que detectaron la misma cara.

estas o detección de características dentro de la cara. En general solo algunos cambios en la forma de proceder son necesarios para ocuparlo en las otras aplicaciones. Estos se explicaran a continuación.

2.2. Detección y Clasificación de Características Faciales

2.2.1. Definición

Antes de proceder, se hará una separación técnica entre la Detección de Características Faciales y la Clasificación de las Caras, con el objetivo de entender de mejor manera los siguientes párrafos.

Cuando se habla de Detección de Características Faciales, se enfrenta un problema sumamente parecido al de la Detección de Caras, o sea, dada una Imagen C, en este caso una cara ya detectada, encontrar la posición, tamaño y orientación de la característica facial C_f dentro de la Cara. Estas características pueden ser, predominantemente, los ojos, la nariz, la boca u otras cosas que se deseen detectar y que puedan encontrarse en la superficie de la cara.

Sin embargo, cuando el problema se convierte en Clasificación de Características Faciales es sutilmente distinto. Dado una cara ya previamente detectada, y dado cierto número de clases, donde cada una conforma un grupo de caras con características en común, el problema se convierte en determinar a cuál de estas clases pertenece la cara ya detectada. Lo que en definitiva se podría entender como, clasificación de caras según las características faciales detectadas. Donde se pueden encontrar trabajos en los cuales se busca clasificar el género o la edad, o bien, si la cara tiene o no lentes, o barba, o bigotes. Si bien estos últimos pueden sonar a detección más que a clasificación, no lo son, pues solo nos entregan una clasificación de la cara, de pertenencia a la clase respectiva que se busca clasificar. Pero no así, su posición, tamaño u orientación dentro de la cara, que es la definición de detección.

Para algunos efectos, es posible que se necesite hacer lo primero, o sea, detectar alguna característica facial para luego clasificar la cara. Por ejemplo, suena convincente que para clasificar si la cara posee o no lentes, primero se detectan los ojos, para así poder obtener un área de la cara más conveniente a clasificar. O bien, que para clasificar el género, primero se detecten los ojos, para así poder rotar la cara a una posición vertical que ayude a hacer las clasificaciones como lo presentado en [2] y utilizado en el presente trabajo.

Estas características faciales detectadas, sin embargo, no tienen nada en común con el conjunto de clasificación, pero ayudaran a obtener posiciones relevantes dentro de la cara para luego hacer una clasificación de mejor manera.

Cualquier método de clasificación usa un conjunto de patrones para caracterizar cada uno de los objetos a estudiar. Estos patrones deben ser relevantes para el objetivo de la clasificación. En esta trabajo de memoria se utilizaran métodos llamados de clasificación supervisada donde un experto humano ha determinado en que clases un objeto puede ser caracterizado. A-priori se necesita cierto conocimiento del objeto a clasificar, sin embargo, no se necesita de conocimiento exacto sobre las diferencias que el clasificador ocupa para hacer las detecciones. O sea se usa un aprendizaje estadístico.

Tanto la clasificación de caras como la detección de sus características se hace muy útil en herramientas de indexación automática como en [4], en interfaces humano computador, en reconocimiento facial (como método para reducir el tiempo de búsqueda en bases de datos, si estas están ordenadas también por dichas características), y en otras muchas aplicaciones.

2.2.2. Alternativas de detección de características faciales y clasificación de estas.

Si bien existen ciertos grupos de clasificación o grupos de detección, en los cuales se pueden encontrar patrones ciertamente definidos, como por ejemplo barba o lentes, en el caso de la clasificación, u ojos o boca, en el caso de la detección (patrones geométricos por ejemplo).

Existen otros grupos de clasificación en los cuales no se sabe con claridad cuáles son los mejores y distintos patrones a utilizar, como por ejemplo el género, la edad o la raza. Incluso, aunque algunos como barbas, lentes, ojos y bocas tengan patrones definidos, sus diferencias hacen que tratar usar de antemano un patrón predefinido sea altamente riesgoso. Por ejemplo, la expresión de la cara, influye enormemente en cómo se encuentra la boca. Según su expresión, pueden o no verse los dientes, o estar o no cerrada, lo cual hace de las bocas objetos más variados y difíciles de detectar como se puede ver en la Figura 4.

Figura 4 "Distintas formas de Boca debido la expresión facial"

Es por esto que los métodos de detección y de clasificación, resultan más robustos si son también basados en las imágenes. O sea, dado un conjunto de imágenes de la clase positiva y negativa, el entrenamiento de estos clasificadores busca la mejor regla que diferencia la clase positiva de la negativa.

Sin lugar a dudas los métodos basados en las imágenes son los más usados en los procesos de detección de este tipo, a diferencia de los métodos basados en las características, donde se necesita un real conocimiento sobre las características que hacen la diferencia entre dos clases. En este trabajo y en general con los métodos basados en las imágenes se necesita seguir un aprendizaje estadístico u otro algoritmo de aprendizaje automático para poder extraer los patrones necesarios para determinar las diferencias entre dos clases.

Como fue mencionado en la sección 2.1.2 dentro de los métodos basados en las imágenes que ocupan aprendizajes estadísticos para construir sus patrones, existen distintas alternativas. Dentro de las más comúnmente usadas están PCA (Análisis de Componentes Principales), SVM (Soporte de Máquina Vectorial), Arboles de decisión, redes neuronales y el mencionado anteriormente, AdaBoost.

La mayor dificultad en usar estos métodos es la alta dimensionalidad del objeto a clasificar, por ejemplo: caras. Debido a esto, un paso sumamente importante debe ser el reducir la dimensionalidad del problema buscando soluciones lo más simples posibles.

En este trabajo se decidió utilizar la base del método de cascada de clasificadores AdaBoost utilizado para la detección de cara, o sea, un clasificador AdaBoost (clasificador "fuerte", Véase 2.1.3). En particular una etapa de una cascada AdaBoost, que por sus buenos resultados para algunos problemas de dos clases, da las herramientas necesarias para la detección de patrones y posterior clasificación de las caras.

Se puede, por ejemplo, observar excelentes resultados para la detección de ojos dentro de caras ya detectadas y además la clasificación de las caras por genero en [2], donde se utilizo solamente un clasificador AdaBoost, en vez de una cascada de clasificadores. Es fácil imaginar que el problema se hace mucho más simple cuando solo se tiene que detectar ojos en una cara ya detectada, o bien, cuando solo se deben clasificar las caras ya detectadas en dos distintas clases.

2.2.3. Detección de características faciales y clasificación de caras utilizando clasificadores AdaBoost

Puede sonar extraño, que un mismo método sirva tanto para la detección de caras como para la posterior detección y clasificación de características faciales. Sin embargo, siempre que se trate de un problema de dos clases AdaBoost cumple con todos los requisitos. En un principio se utilizan CARAS y NO CARAS, para la detección de las caras en las imágenes, y luego, según lo

que se requiera detectar o característica que se requiera clasificar, se utilizan imágenes de la clase POSITIVA y clase NEGATIVA.

La mayor diferencia en el proceder ante el problema de detección y el de clasificación de características faciales es que cuando se está detectando con un clasificador AdaBoost se ocupa el mismo método que al detectar caras. Pero esta vez sobre una cara ya detectada como se puede observar en la Figura 5. Es decir, se obtienen todas las ventanas posibles dentro de la Imagen (que esta vez podría ser solo una cara, o bien una parte de esta). Luego cada una de estas imágenes es entregada a un clasificador basado en las imágenes, si el resultado es positivo se tiene un acierto a la detección. De lo contrario en esa ventana extraída de la imagen no se encuentra lo que se desea detectar. Cabe destacar que existen otros pasos importantes en la detección de caras que ya fueron explicados en la sección 2.1.3.

Figura 5 "Detección con Clasificadores Basados en las Imágenes"

Por otra parte cuando el objetivo es clasificar la cara según cierta característica como se puede ver en la Figura 6. Se debe obtener la imagen, que puede ser, una cara o una parte de la cara ya detectada. Por ejemplo, la parte superior de la cara para la clasificación de lentes, o la parte inferior para la clasificación de barba. Luego, esta imagen se entrega al clasificador Basado en las imágenes de donde se obtendrá una respuesta positiva o negativa. En este caso la magnitud de esta respuesta se podrá traducir como el grado de pertenencia a una u otra clase.

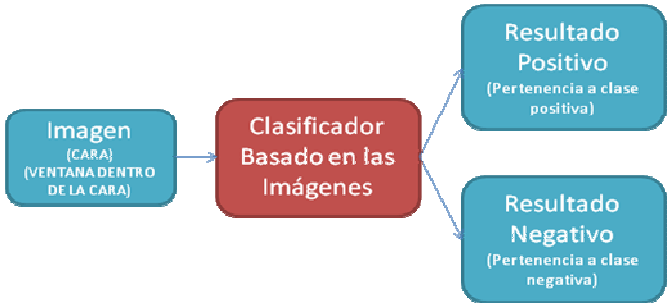


Figura 6 "Clasificación Basada en las Imágenes"

Como se ha mencionado antes, la detección de características faciales usando clasificadores AdaBoost no necesita, a-priori, de patrones que determinen la detección. Sin embargo se necesita de un vasto entrenamiento basado en algoritmos estadísticos para llegar al mejor clasificador posible. Desde ya cabe destacar que el número de ejemplos disponible es proporcional al resultado obtenido, por lo que el uso de grandes bases de datos es requerido en estos casos.

También cabe recalcar que, como se mencionó antes, en esta situación no se utiliza una cascada AdaBoost, sino que un solo clasificador fuerte AdaBoost. Esto se debe a que al tener la cara ya detectada el área de búsqueda para hacer la detección de cierto patrón es menor y por lo tanto mucho más rápida. En el caso de la clasificación, una razón importante de ocupar un solo clasificador fuerte es que el obtener objetos de la clase negativa o positiva no se hace tan automático. Es por esto que es mucho más difícil poder conducir un entrenamiento tan largo como el que implica ocupar mas cascadas. Sin embargo, el hecho de utilizar una sola cascada, no merma el funcionamiento de los detectores o clasificadores, ya que aun así, los resultados son muy alentadores.

3. Algoritmo de Entrenamiento de Clasificadores

Esta sección introduce de forma teórica y explicativa el funcionamiento de AdaBoost que se ha mencionado en variadas oportunidades como modulo importante dentro de un detector de caras, como luego en la posterior clasificación de estas.

3.1. Generalidades para el Entendimiento de Detectores

Antes de empezar con la explicación de Adaboost se definirá ciertos criterios que ayudarán a entender de mejor manera los detectores y determinarán el rendimiento de estos. Lo primero que se estima conveniente introducir es la llamada Tasa de Detección (Detection Rate en Inglés). Para el caso del detector de caras, la tasa de detección se define como el porcentaje de caras que han sido detectadas en una imagen, y como tal es el porcentaje que se busca maximizar. Por otro lado también se deben definir las Tasas de Falsos, que son dos. La primera es la tasa de Falsos Negativos, que es el opuesto a la Tasa de Detección, o sea las caras que han sido olvidadas por el detector. Y la segunda la tasa de Falsos Positivos, que serian las no caras que han sido erróneamente detectadas por el detector como caras.

Si se piensa detenidamente es muy fácil obtener un detector con alta Tasa de Detección. Solo se necesita detectar caras en todas partes y seguramente se obtendrán entre ellas las verdaderas caras. Sin embargo, el verdadero objetivo de un clasificador es maximizar la Tasa de Detección minimizando la Tasa de Falsos Positivos. En la vida real, es sumamente complejo o imposible obtener detectores que cumplan a la perfección con estas dos condiciones, por lo que se busca hacer lo mejor posible. Se sub entiende que existe algún tipo de Trade-Off entre estas dos condiciones.

Otro criterio importante para el análisis de clasificadores en base a las tasas de rendimiento (Tasa de Detección y Falsos positivos) son las curvas ROC. Las curvas ROC (por la sigla en inglés de

Receiver Operating Characteristic) son comúnmente utilizadas por doctores en el diagnostico de enfermedades. Estas resultan de graficar la tasa de detección vs los falsos positivos, moviendo en cada punto el umbral de detección. Punto desde donde los resultados mayores que este son decisiones positivas y menores decisiones negativas. De esta forma se pueden comparar abiertamente dos clasificadores de una misma característica. El objetivo principal es que exista un punto umbral desde el cual la tasa de detección sea máxima y la tasa de falsos positivos sea mínima. En la Figura 7 se presenta un grafico de dos curvas ROC donde se observa que el clasificador 2 alcanza mejores resultados que el clasificador 1 cuando la Tasa de Detección es mayor que un 73%.

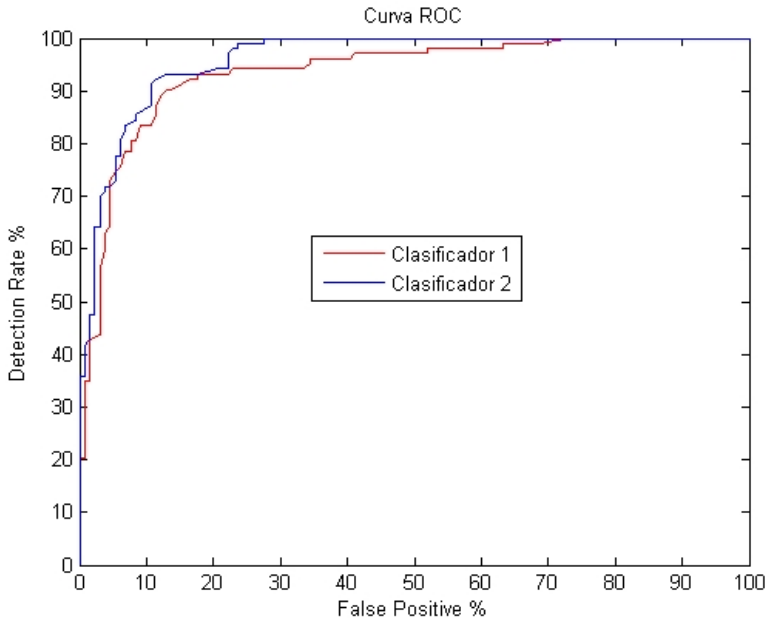


Figura 7 "Ejemplo Curvas ROC"

3.1.1. Definición

Como ya se ha mencionado antes AdaBoost se emplea para encontrar reglas de clasificación altamente precisas (fuertes) combinando muchas reglas de baja precisión (débiles). Estos clasificadores débiles son linealmente combinados obteniendo así un clasificador fuerte.

Para hacer más simple la explicación del AdaBoost propuesto en [5] se explica el algoritmo básico de AdaBoost con un ejemplo simple. Se tiene un set de entrenamiento $(x_1, y_1), \dots, (x_m, y_m)$ donde cada x_i pertenece a una clase o espacio X y cada etiqueta y_i pertenece a un set de etiquetas Y, para darle simpleza en esta explicación de asumirá que $Y = \{-1, +1\}$. AdaBoost se genera a partir de un algoritmo de aprendizaje estadístico en $t = 1, \dots, T$ ciclos. Una de las principales ideas del algoritmo es mantener un conjunto de pesos sobre el set de entrenamiento. El peso perteneciente a este conjunto de pesos, sobre el ejemplo i en el ciclo t se le llama $D_t(i)$. Inicialmente, todos los pesos se colocan iguales (o sea cada uno de los ejemplos con los que se hace el entrenamiento poseen pesos iguales), pero en cada ciclo los pesos de los ejemplos incorrectamente clasificados son aumentados. Con esto el próximo clasificador débil estará forzado a enfocarse en los ejemplos difíciles del set de entrenamiento que intuitivamente se ubican más cercanos a la frontera de clasificación. Es por esto el nombre del algoritmo, AdaBoost, viene de las palabras Adaptive Boosting (Boosting Adaptivo), pues el clasificador se va adaptando al grupo de entrenamiento.

El trabajo del clasificador débil es encontrar una hipótesis débil $h_t : X \to \{-1, +1\}$ apropiada para el conjunto D_t. Cuan bueno es esta hipótesis débil se mide a través de su error de la forma indicada en (1).

$$\epsilon_t = Pr_{i \sim D_t}[h_t(x_i) \neq y_i] = \sum_{i : h_t(x_i) \neq y_i} D_t(i) \qquad (1)$$

Se debe hacer hincapié en que éste error es medido con respecto al conjunto D_t en el cual el clasificador débil fue entrenado y además que el error es medido con respecto a los a los ejemplos mal clasificados y es la suma de estos. En la práctica el clasificador débil puede ser un algoritmo que use lo pesos D_t sobre el set de entrenamiento. Alternativamente, cuando esto no es posible, un subconjunto del set de los ejemplos de entrenamiento puede ser muestreado de acuerdo a D_t y estos ejemplos muestreados, que si pueden ser utilizados, se usan como ejemplos de entrenamiento para el clasificador débil.

Entonces el entrenamiento del clasificador fuerte AdaBoost procede de la siguiente manera:

Dado $(x_1, y_1), \ldots, (x_m, y_m)$ donde $x_i \in X, y_i \in Y = \{-1, +1\}$

Inicializar $D_1(i) = \frac{1}{m}$

FOR $t = 1, \ldots, T$:

- Entrenar un clasificador débil ocupando la distribución D_t.
- Obtener una hipótesis débil $h_t: X \to \{-1, +1\}$ con error:

$$\epsilon_t = Pr_{i \sim D_t}[h_t(x_i) \neq y_i] \quad (2)$$

- Elegir:

$$\alpha_t = \frac{1}{2} ln\left(\frac{1 - \epsilon_t}{\epsilon_t}\right) \quad (3)$$

- Actualizar:

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t}, & h_t(x_i) = y_i \\ e^{\alpha_t}, & h_t(x_i) \neq y_i \end{cases} \quad (4)$$

$$= \frac{D_t(i) e^{\left(-\alpha_t y_i h_t(x_i)\right)}}{Z_t}$$

Donde Z_t es un factor de normalización (elegido para que así D_{t+1} tenga la forma de una distribución.

Así la respuesta final del clasificador débil queda dada por:

$$H(x) = signo\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right) \quad (5)$$

Una vez que la hipótesis débil h_t está completa, AdaBoost elige un parámetro α_t como se puede apreciar en el algoritmo previamente descrito. Intuitivamente α_t mide la importancia

que es asignada a h_t. Notar siempre que $\alpha_t \geq 0$ si $\epsilon_t \leq \frac{1}{2}$, y que α_t se hace más grande cuando ϵ_t se hace más chico, o sea, son inversamente proporcionales.

Luego la distribución D_t es actualizada usando la regla mostrada en la ecuación (4). El efecto de esta regla es incrementar el peso de los ejemplos mal clasificados por h_t, y disminuir el peso de los ejemplos correctamente clasificados. Así, el peso tiende a concentrarse en los ejemplos más difíciles. Y la hipótesis final o clasificador fuerte queda dado por (5), donde H es una suma de mayoría de pesos de las T hipótesis débiles donde α_t es la importancia asignada al clasificador débil h_t.

Como se puede entender, este clasificador fuerte entrando tiene una salida binaria. Sin embargo en [6] fue estudiado como el resultado de este clasificador fuerte puede ser extendido a una salida de valor real o predicción con confidencia real. En otras palabras, para cada ejemplo x, la hipótesis débil h_t tiene como salida $h_t(x) \in \mathbf{R}$, cuyo signo es la etiqueta clasificada (-1 o 1) y cuya magnitud $|h_t(x)|$ da una medida de la confidencia en la clasificación.

Entonces al utilizar la extensión de AdaBoost este queda definido como se muestra en la ecuación (6), un clasificador AdaBoost donde cada una de las funciones $h_t(x)$ es un clasificador débil, T es el numero de clasificadores débiles y b es el umbral que define el punto de operación del clasificador. La clase asignada a la salida corresponde al signo de $H(x)$ o sea, positivo o negativo.

$$H(x) = \sum_{t=1}^{T} \alpha_t h_t(x) - b \qquad (6)$$

En este caso, cuando los ejemplos son imágenes, los clasificadores débiles son aplicados sobre características (Features en Inglés) calculadas en cada uno de los patrones a ser procesados. Cada clasificador débil tiene asociado una sola característica.

El diseño de estos clasificadores débiles se hace bajo el paradigma de la "partición de dominio" [6]. Bajo este paradigma los clasificadores débiles hacen sus predicciones basadas en la partición del dominio. Si se llama a todo el dominio X , cada clasificador tiene un valor asociado a una partición de este dominio. Este dominio a su vez se separa en bloques disjuntos $X_1, \ldots, X_n$, para el cual cada $h(x) = h(x')$ para cada $x, x' \in X_j$. Así, dado un ejemplo a clasificar, la predicción del clasificador débil dependerá únicamente de en cual bloque X_j se encuentre dicho ejemplo. En este caso, cuando los clasificadores débiles son aplicados sobre características, cada clasificador débil tiene asociado una única característica, por lo siguiente cada dominio de una característica F es particionado de la misma manera en $F_1, \ldots, F_n$, y un clasificador débil h tendrá un resultado para cada una de las particiones del dominio de la característica asociada, o sea $f: h(f(x)) = c_j \ni f(x) \in F_j$.

Para cada clasificador, el valor asociado a cada una de las particiones del dominio c_j, o sea, el resultado o salida, es calculado minimizando la función de pérdida de margen, de la que es responsable el error de entrenamiento [6]. El valor c_j depende, primero, de las veces que su característica correspondiente cae dentro de la partición F_j (Histograma), segundo, de la clase de este ejemplo (y_i), y tercero, de su importancia o peso $D(i)$. Así el valor de c_j esta dado por la siguiente ecuación (7)[6].

$$c_j = \frac{1}{2} ln\left(\frac{W_{+1}^j + \varepsilon}{W_{-1}^j + \varepsilon}\right)$$

Con

$$W_l^j = \sum_{i: f(x_i) \in F_j \wedge y_{i=l}} D(i) = Pr[f(x_i) \in F_j \wedge y_{i=l}] \quad (7)$$

donde $l = \pm 1,$

Y ε es un parámetro de regulación [6].

Cada uno de los resultados de los clasificadores débiles c_j es almacenado en una LUT[3] para ahorrar tiempo de computación al hacer las clasificaciones. Así un clasificador débil quedara definido por $h(f(x)) = h_{LUT}[x] = LUT[index\ f(x)]$ con $index$ la función que entrega el índice (de la partición) asociado al valor de la característica en la LUT.

Los clasificadores débiles usados para detectar las caras, los ojos y clasificar el género en [2] son características rectangulares como las de Haar [7] y mLBP [8]. Estas serán correctamente definidas y explicadas en la sección 3.1.2 de este capítulo. Ambas características son también los clasificadores débiles usados en el presente trabajo para la clasificación de características faciales.

Es más claro ver ahora la conformación de una cascada de clasificadores fuertes AdaBoost. Una cascada tipo nested (anidada), compuesta de M etapas, se define como la unión de M clasificadores AdaBoost H_C^k (o fuertes), como se muestra en la ecuación (8).

$$C = \bigcup_{k=1}^{M} \{H_C^k\} \tag{8}$$

Cada uno de estos clasificadores fuertes H_C^k queda definido por:

$$H_C^k(x) = H_C^{k-1}(x) + \left(\sum_{t=1}^{T_k} h_t^k(x) - b_k \right) \tag{9}$$

Con

$$H_C^0(x) = 0$$

[3] LUT, de la sigla en Ingles de Look Up Table, es usualmente un arreglo usado para reemplazar tiempo de calculo con una simple operacion de busqueda. El aumento en la velocidad de los algoritmos puede ser significativa, ya que, el obetener datos guardados desde memoria suele ser mucho mas rapido que el calculo de estos (http://en.wikipedia.org/wiki/Lookup_table).

Donde ahora el clasificador débil queda dado por h_t^k, T_k es el número total de clasificadores débiles para la etapa k y b_k es el umbral usado para esa etapa. Donde la salida o resultado O_C de C está dado por la ecuación (10):

$$O_C(x) = \begin{cases} sign\left(H_C^q(x)\right), & H_C^k(x) \geq 0, k = 1, \dots, q-1 \wedge \left(H_C^q(x) < 0 \vee q = M\right) \\ sign\left(H_1^q(x)\right), & H_C^1(x) < 0 \end{cases} \quad (10)$$

Con el valor de la confidencia de una detección positiva dado por la ecuación (11).

$$f(x) = \begin{cases} H_C^q(x), & H_C^k(x) \geq 0, \quad k = 1, \dots, q-1 \wedge H_C^q(x) < 0 \\ H_C^M(x), & H_C^k(x) \geq 0, \quad k = 1, \dots, M \end{cases} \quad (11)$$

En el diagrama de la Figura 8 se puede observar de mejor manera, la interrelación entre cada cascada de AdaBoost y los clasificadores débiles.

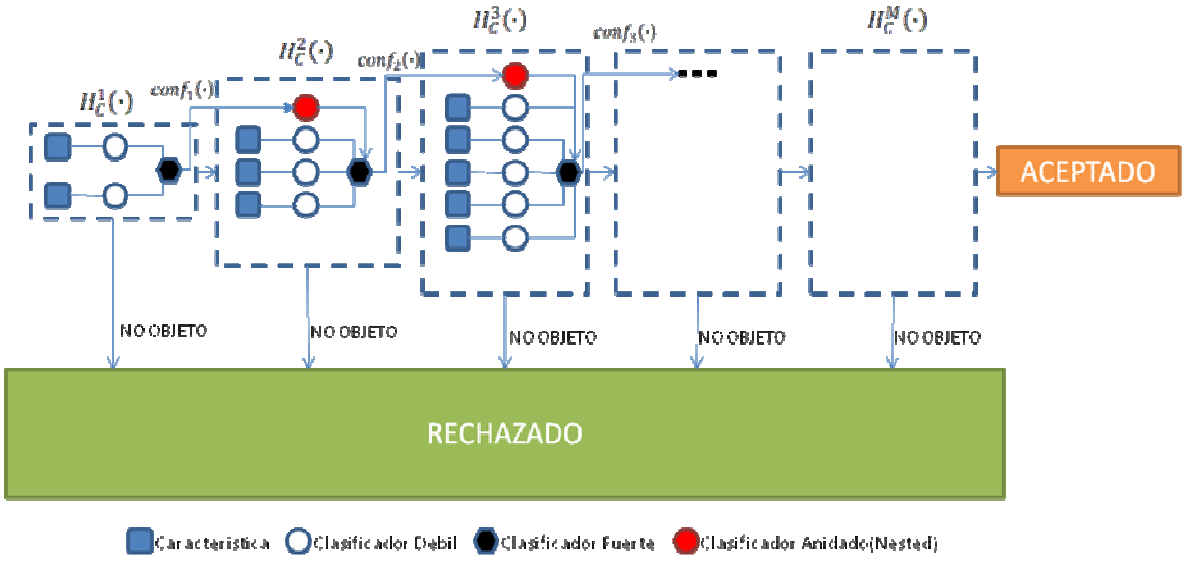


Figura 8 "Diagrama de un Clasificador AdaBoost en Cascada"

El diagrama de la Figura 8 representa una cascada de clasificadores Adaboost. Esta cascada por si sola no alcanza mejores resultados que un único clasificador Adaboost debido solamente a su arquitectura. Para obtener mejores resultados es de suma importancia la forma en que se entrena la cascada. Esta forma de entrenar o método llamado Boostraping, consiste en una buena forma de obtener los mejores ejemplos negativos de lo que se desea clasificar.

Por ejemplo si se están detectando caras, cada parte de una imagen donde no se encuentra una cara es un posible candidato a ejemplo negativo. Sin embargo, lo que se desea siempre es encontrar los ejemplos negativos más difíciles, o sea, que más se parezcan a la clase positiva, en este caso caras. Así, el Boostraping corresponde a entrenar cada etapa iterativamente adhiriendo todos los ejemplos clasificados de forma errónea a la base de entrenamiento, así el clasificador se entrena con ejemplos cada vez más difíciles. Con esto se incrementa la calidad del conjunto de entrenamiento en cada iteración. Para más información remítase el lector a [2].

3.1.2. Tipos de Features

Como ya se ha mencionado en este trabajo se utilizaron 2 tipos de características, los LBP y las Rectangulares. A continuación se describe cada una de estos.

LBP (Local Binary Pattern)

El LBP (Patrón Local Binario, LBP por su sigla en Inglés), fue introducido y por primera vez ocupado en [9]. Es un operador de análisis de textura que se define como una medida invariante en las escalas de grises, derivado de la definición de textura en una vecindad local. Este ha sido ampliamente utilizado en un sinnúmero de trabajos de análisis de imágenes. Existen distintas versiones de los LBP [10] aunque básicamente la idea es la misma: un código binario que describe la textura local de un pixel y es construido haciendo una operación binaria entre este pixel y sus vecinos. El primer LBP

que se introdujo en [9]. Este opera con los 8 vecinos de un pixel y usaba el valor de la intensidad del pixel del centro (en escala de grises) como referencia para hacer una comparación binaria con sus vecinos, como se observa en la Figura 9. Con esto se obtiene un arreglo ordenado de dichas comparaciones.

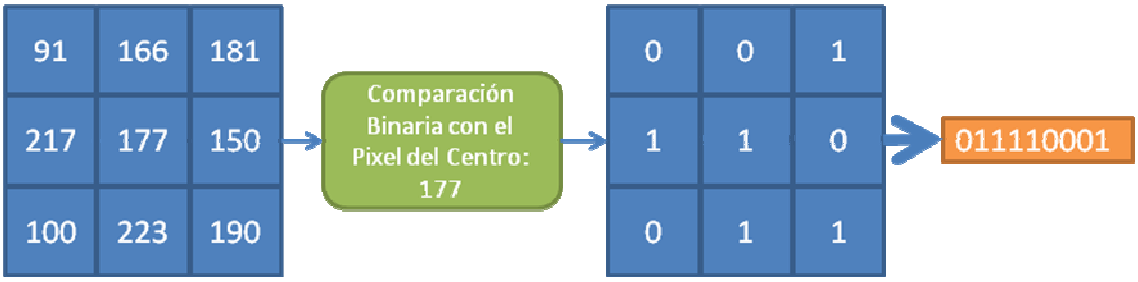
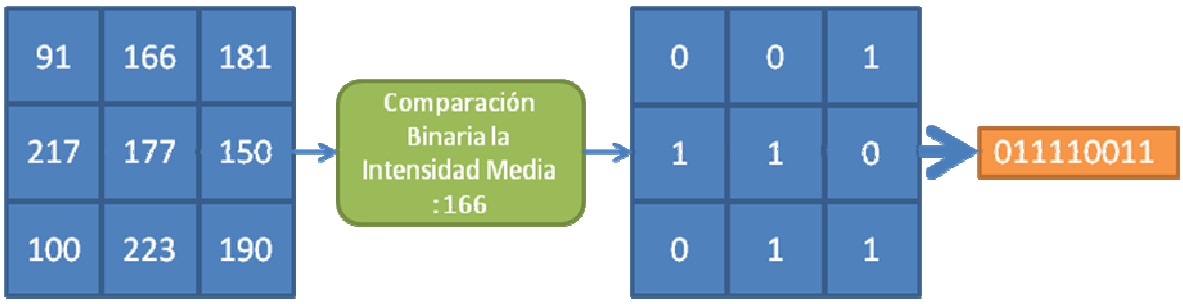


Figura 9 "Ejemplo Calculo LBP"

En este trabajo se hizo uso del mLBP, (LBP modificado), introducido en [8]. La diferencia con el LBP ya explicado es que el mLBP utiliza la intensidad media de todos los pixeles ubicados en la vecindad como referencia y hace la comparación binaria con esta intensidad media, como se observa en la Figura 10. Una de las ventajas de utilizar la intensidad media como referencia es que lo hace invariante a los cambios de iluminación que afecten a todos los pixeles por igual. Al usar la intensidad media de la vecindad de pixeles cualquier cambio en la iluminación se hace invariante a él.



$$\bar{I} = \left(\frac{91 + 166 + 181 + 217 + 177 + 150 + 100 + 223 + 190}{9}\right) = 166$$

Figura 10 "Ejemplo Calculo mLBP"

Otra propiedad interesante de este patrón es su facilidad de cálculo. Se pueden calcular pasando solamente una vez por cada pixel de la imagen. Esta rapidez le permite ser utilizado en aplicaciones de tiempo real como en [2] y [11].

Rectangulares

Las características rectangulares son similares a las características wavelets de Haar. Fueron introducidas en [12] y usadas eficazmente para la detección en [7] y [2]. Estas se usan generalmente cuando se desea hacer una detección rápida y eficaz. Las principales razones de su uso es que, al ser basada en operadores sobre la imagen y no en los pixeles de la imagen como las características LBP, dan una gama mucho más amplia de posibilidades para las situaciones en que el entrenamiento se encuentra frente a dominios de alta dificultad. Además el cómputo de este tipo de características es mucho más rápido.

Específicamente se utilizaron 3 tipos de características rectangulares, la característica de dos rectángulos cuyo valor es la diferencia entre la suma de la intensidades dentro de los dos áreas, estos rectángulos tienen la misma áreas, y son adyacentes vertical u horizontalmente. La característica de tres rectángulos, cuyo valor es la suma de las intensidades de los dos rectángulos de los bordes menos la suma de la intensidades del rectángulo del medio. Y finalmente la característica de cuatro rectángulos que ocupa la diferencia entre la suma de los rectángulos en diagonal. Cada una de están se pueden entender de mejor forma con la Figura 11, donde las áreas de color azul deben ser restadas a las áreas de color marrón.

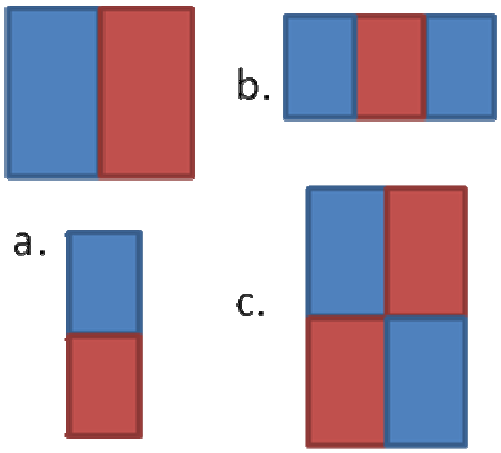


Figura 11 "Ejemplo Características Rectangulares"

El cómputo de estas características se hace a través de la imagen integral [7]. Esta calcula la suma de las intensidades de todos los pixeles de la izquierda y arriba de un pixel dado. De esta forma se puede computar cualquier rectángulo dentro de la imagen haciendo solo cuatro sumas de los resultados de la integral imagen en sus vértices. Entonces si la integral imagen es calculada en un principio para todos los pixeles de la imagen el cómputo de las características rectangulares es lo suficientemente veloz para ocuparlos por ejemplo en tareas de seguimiento visual como lo presentado en [11].

3.2. Reducción del problema multi-clase a 2-Clases

Todo lo anteriormente explicado se ha basado en problemas de dos clases, clase positiva y clase negativa. Sin embargo en las tareas de reconocimiento y clasificación existen un enorme número de problemas que tienen un número mayor de clases. Un ejemplo claro está dado en [11] donde el numero de gestos que la mano puede representar es mayor que dos. El método utilizado fue, dado N distintas clases, entrenar N clasificadores binarios, uno para cada clase, así luego, entregar los resultados de las detecciones a otro clasificador entrenado que dará la mejor decisión. Otro ejemplo es el método ocupado en [4], donde se aborda los problemas multi-clase de la clasificación de raza y edad. En este caso el método usado fue distinto. Se utilizo para ambos casos el método presentado en [13], o sea reduciendo el problema multi-clase a N-1 problemas binarios. Este método obtiene

buenos resultados siempre y cuando los clasificadores débiles sean lo suficientemente efectivos incluso para distribuciones difíciles como las que se crearán al juntar dos clases distintas en una [5]. La única diferencia entre las dos clasificación efectuadas (Edad y Raza) fue el tipo de estructura en que se hizo la separación binaria del problema.

Cabe destacar que existen otros métodos para la reducción del problema multi-clase al problema binario. Todos estos más sofisticados que el presentado anteriormente. Por ejemplo el algoritmo AdaBoost.HM presentado en [6] o el Adaboost.M2 presentado en [13]. El problema de estos métodos "más sofisticados", es que también necesitan del diseño de clasificadores débiles más sofisticados y otros método como corrección de errores. Sin embargo, para este trabajo de memoria se eligió hacer una aproximación al problema multi-clase siguiendo la misma línea del problema de dos clases.

En este trabajo se aborda nuevamente el problema de las edades. Ahora con el objetivo de determinar si se puede usar el resultado de la confidencia para dar resultados más generales. Es decir, si el conjunto es separado en trazos de 4 edades se intenta, usando la confidencia, crear intervalos en los cuales la decisión no es 100% cierta sino que con cierto grado de suposición. Esto será explicado de mejor manera en la sección 5.3.5.

4. Desarrollo de Detectores y Clasificadores de Características Faciales

En este capítulo se explicara de forma detallada el desarrollo de los detectores y clasificadores de características faciales, desde la creación de las bases de datos, hasta su entrenamiento y técnica para mejorar los resultados. Cabe destacar que, detectores y clasificadores poseen características similares en la construcción de ellos, así pues, los puntos siguientes rigen para cada uno de los clasificadores desarrollados.

Como ya antes se ha señalado existe cierta diferencia intrínseca entre la detección de características faciales y la clasificación de características faciales. Esta diferencia también se hace notar en esta sección del trabajo. En este trabajo de memoria primero se intentó desarrollar clasificadores de características faciales: Lentes, Barba y Bigotes, usando solamente los detectores de caras y ojos previamente mencionados y desarrollados en [4]. Sin embargo, cuando se desarrolla de esta forma el clasificador de barba y bigotes se presenta un problema singular; es imposible obtener el área específica de la cara que se requiere para la clasificación de barba y bigotes con una precisión pertinente teniendo solo la información de la posición de los ojos. Por tal motivo se decide la creación de un detector de bocas. Con el detector de bocas se puede obtener la posición de está, para así luego extraer el área deseada de la cara (bigotes o barba). Esto se explica de forma más detallada en el transcurso de esta sección.

4.1. Diseño y construcción de detectores y clasificadores de características faciales

El Diseño y la construcción de los clasificadores y detectores de características faciales se basa en el diseño y construcción de clasificadores AdaBoost. El primer paso es juntar una base de datos lo suficientemente amplia y general de modo que no abarque solo imágenes con ciertas características. Como ya se ha mencionado previamente, el tamaño y la variedad de las bases de datos son directamente proporcionales a la calidad de los resultados que se obtienen, por lo que adquirir y construir las bases de datos es sumamente importante.

4.1.1. Base de Datos

La construcción de las bases de datos se puede separar en distintos pasos debido a que se requiere trabajar sobre las imágenes antes de entrenar los clasificadores. Los pasos a seguir son: a) adquisición de las imágenes, b) marcado de los puntos de interés de las imágenes, c) pre-clasificación (asistida) de las imágenes y d) recorte de las caras. Se explica en esta sección los tres primeros pasos de manera más general pero el último paso solo de forma más particular, ya que el recorte de la cara tiene más relación con cada uno de los clasificadores y detector entrenados.

a) Adquisición de las imágenes

Para los distintos clasificadores se usaron las bases de datos, UDBChileGender [4], ARFdb [14], Equinox [15], FGNET [16], Cas-Peal [17], JAFFE [18], CALTECH [19] y VALID DATABASE [20] así como también un conjunto de imágenes propias y bajadas de internet. Pese a sonar un paso corto y simple es una de las fases más largas y demorosas del proyecto. Buscar las bases de datos que han sido reunidas o desarrolladas por alguna Universidad o Instituto de Investigación, conseguir permisos, bajarlas de Internet y luego definir si realmente serán útiles o no para lo que se requiere en este trabajo puede tomar un largo periodo. Bases de datos de conjuntos más raros como barba, lentes y bigotes, no siempre se encuentran disponibles en dichas bases de datos, por lo que se debe hacer una base de datos propia, usualmente bajando imágenes de internet. Esto también requiere de mucho tiempo, sobre todo porque las imágenes que se encuentran de manera rápida en internet no siempre cumplen de la mejor forma las características que se requieren para una base de datos. Luego de tener las bases de datos se procedió al marcado de las imágenes.

b) Marcado de los puntos de Interés

Dado el tipo de clasificadores que se desarrolla en este trabajo, existen distintos puntos de interés que ayudan a obtener con más facilidad bases de datos coherentes para los entrenamientos. En particular se usa los puntos del centro de los ojos y la boca para obtener la información que previamente fue utilizada en los entrenamientos.

El marcado de las imágenes se desarrolla con una aplicación existente en la cual de forma simple se carga un conjunto de imágenes, para luego manualmente ir marcando una a

una, ojos, nariz y boca. Pese a que existen métodos de detección de ojos con resultados más que aceptables, se prefiere hacer un marcado manual debido a que se obtiene una mayor precisión. La interfaz de usuario de la aplicación para marcar los ojos es de fácil uso y se presenta en la Figura 12. El resultado de esta aplicación es un archivo XML con todos los nombres y las coordenadas de los puntos de las imágenes marcadas. El formato de este archivo se presenta en el Anexo A.

Luego de obtener este archivo XML se procedió a separarlo a un único archivo por foto. Así se obtiene la información de todas las imágenes por separado que pueden ser accesadas por distintos clasificadores al momento de hacer los entrenamientos. Esta separación del archivo XML se hace a través de una pequeña rutina en Linux, que se puede encontrar en el Anexo B, con la cual se obtuvo archivos con el mismo nombre de la foto, pero con extensión .GND. Cada uno de los archivos presenta la información de cada una de las caras situadas en la imagen y las respectivas coordenadas de los ojos, la nariz y la boca. El formato del archivo .GND es presentado en el Anexo A.

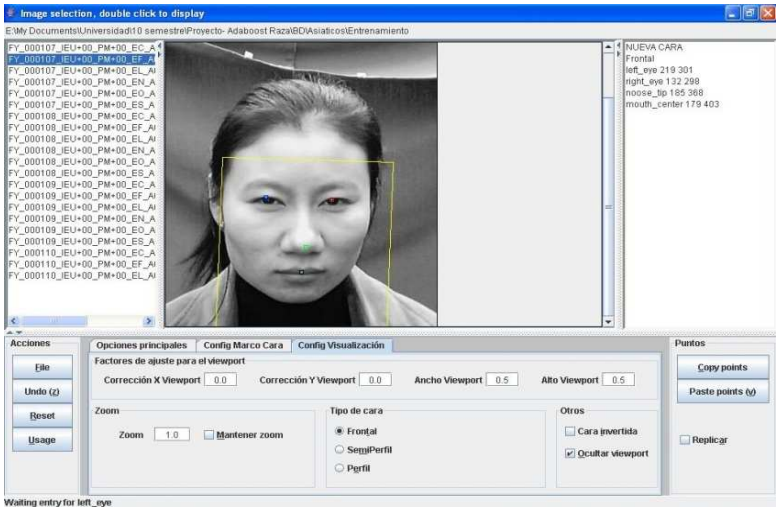


Figura 12 "Aplicación para el marcado de los ojos"

c) Pre-clasificación de las BASES

Luego de marcar todas las imágenes de las que se dispone se las separa dependiendo de sus características para cada uno de los clasificadores a usar. En general existen muchas imágenes repetidas entre cada base de dato de los clasificadores. Esto se debe a que existen ciertas características que pueden estar repetidas en una cara, como barba, bigotes o lentes. Si se pudiera al mismo tiempo que se marca las imágenes marcar a que clases estas pertenecen se evitaría la repetición innecesaria de imágenes entre cada una de las bases de datos. La actualización o construcción de un nuevo marcador de caras con este tipo de característica sería muy bueno para aumentar la velocidad de todo el procedimiento de generación de las bases de datos.

d) Recorte de las caras

Una vez separadas todas las imágenes por clase, se procede al recorte de las caras para cada imagen. Al tener la información de la posición de cada ojo y boca en la cara, se rota la imagen al plano de los ojos para luego con la relación entre la distancia de los ojos y el punto central entre ellos, recortar el área de la cara.

Como se puede ver en la secuencia de la Figura 13, la cara primero se rota y luego se recorta la zona de donde se encuentra la cara.

Figura 13 "Secuencia de Recorte de Cara"

Ya teniendo la información de la posición de los ojos el rotar la cara es bastante simple. Lo que se hace es calcular el ángulo entre los ojos y establecer una nueva posición para estos en la cara rotada. Se forma así una nueva imagen donde el eje horizontal queda paralelo al de la línea que une los ojos. Luego, para recortar el área de la cara se utiliza como referencia el punto central entre los ojos y la distancia entre estos. Se puede apreciar en la Figura 14 el mismo ejemplo de la Figura 13 donde se muestra paso a paso el cálculo de las coordenadas y luego el recorte de la imagen. En este caso las distancias de recorte son las mostradas en la Figura 14.

Entonces teniendo como:

(Xod, Xod) : Par Ordenado Posición Ojo Derecho sin Rotación

(Xoi, Xoi) : Par Ordenado Posición Ojo Izquierdo sin Rotación

(Xb, Xb) : Par Ordenado Posición Boca sin Rotación

An : Ancho de la Imagen Original

Al : Alto de la Imagen Original

Se obtiene que:

$$d = \sqrt{((Xoi - Xod)^2 + (Yoi - Yod)^2)}$$

$$\alpha = \tan^{-1}\left(\frac{(Yoi - Yod)}{(Xoi - Xod)}\right) \quad (12)$$

Con d la distancia entre los ojos y α el ángulo de rotación de la imagen. Con esto ya se pudo formar una imagen rotada, en la cual los ojos estarán en el eje horizontal como lo muestra la Figura 14. Cabe destacar que la rotación de la imagen se realiza de forma simple solo con una instrucción en Matlab.

Luego se tiene que:

$$diagd = \sqrt{Xod^2 + Yod^2}$$

$$diagi = \sqrt{Xoi^2 + Yoi^2}$$

$$diagb = \sqrt{Xb^2 + Yb^2}$$

$$angd = \tan^{-1}\left(\frac{Xod}{Yod}\right) \quad (13)$$

$$angi = \tan^{-1}\left(\frac{Xid}{Yid}\right)$$

$$angb = \tan^{-1}\left(\frac{Xb}{Yb}\right)$$

Como se puede apreciar en la Figura 14 se puede calcular la nueva posición de los ojos para la imagen rotada con la cual se obtendrá el nuevo punto central entre ellos.

Entonces:

$$Xnod = diagd \cdot \sin(angd + \alpha)$$

$$Ynod = diagd \cdot \cos(angd + \alpha) + An \cdot \sin\alpha \quad (14)$$

Y consecuentemente:

$$Xnoi = diagi \cdot \sin(angi + \alpha)$$

$$Ynoi = diagi \cdot \cos(angi + \alpha) + An \cdot \sin\alpha \quad (15)$$

$$Xnb = diagb \cdot \sin(angb + \alpha)$$

$$Ynb = diagb \cdot \cos(angb + \alpha) + An \cdot \sin\alpha$$

Finalmente:

$$Pm = \left(\frac{Xnod + Xnoi}{2}, \frac{Ynod + Ynoi}{2}\right) \quad (16)$$

Con el par ordenado $(Xnod, Ynod)$, $(Xnoi, Ynoi)$ y (Xnb, Ynb) posiciones respectivamente del ojo derecho, ojo izquierdo y boca en la imagen rotada, donde Pm es el par ordenado que representa el punto central entre los ojos, como se puede apreciar en la Figura 14.

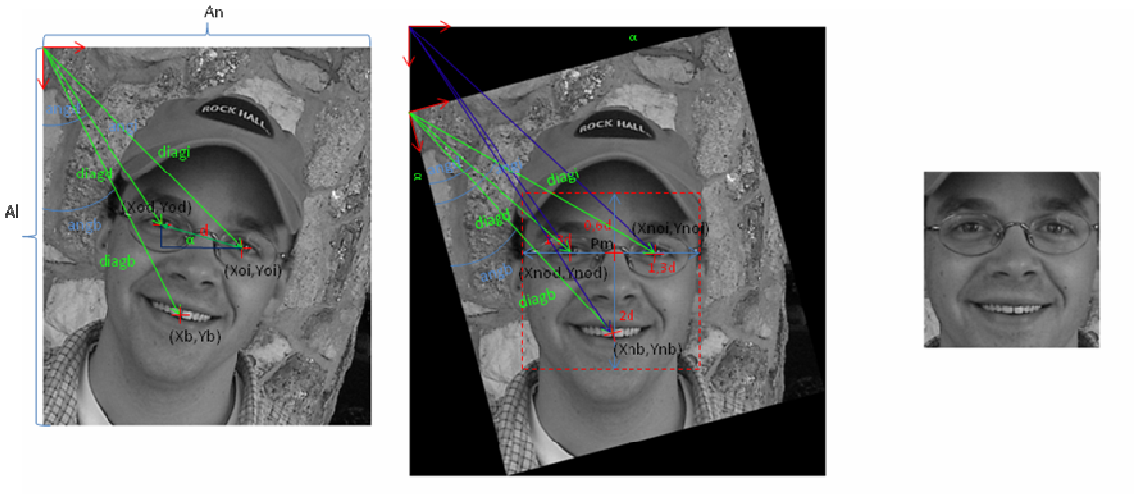


Figura 14 "Secuencia de Recorte con Referencias"'

Si la cara estuviera rotada en el otro sentido el cálculo tiene un cambio, ya que el punto de referencia es distinto. Esto se puede generalizar para el segundo caso muy fácilmente y no será explicado, de todos modos, se encuentra a disposición en los códigos de recorte de cara.

Luego de obtener todas las posiciones en la imagen rotada se procede al corte del área de interés para hacer las verdaderas bases de datos de los clasificadores y detectores. ¿Porque un área de interés? En un principio en este trabajo de memoria se intenta clasificar lentes, barbas y bigotes usando la cara entera de 24× 24 pixeles y luego de 48× 48 pixeles. Sin embargo los resultados son nefastos. La razón de estos malos resultados es que se usa información extra de la cara. La primera solución es usar una máscara sobre la cara, así los clasificadores no ocupan la información fuera de la máscara. Debido al uso de la máscara el problema es ahora que el tamaño del área que deja la máscara es demasiado pequeño, como se puede apreciaren la Figura 15. Adicionalmente los resultados no mejoran lo suficiente.

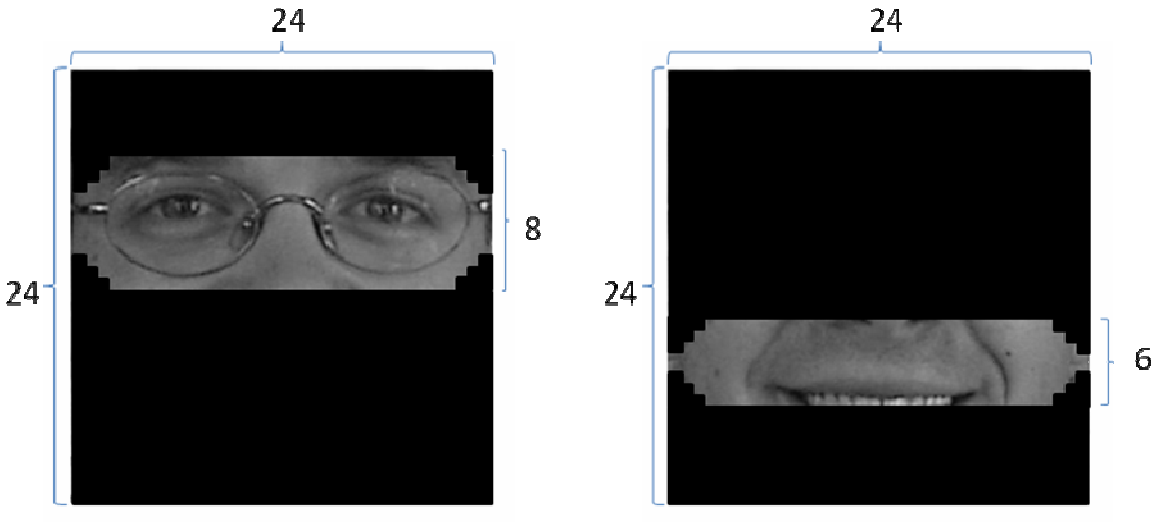


Figura 15 "Ejemplo Cara Entera con Mascara, Lentes y Bigotes"

Finalmente, y siguiendo el método usado en [4] y [2] para la detección de los ojos, se utiliza un área de la cara. Área que es más interesante para cada clasificación y que además es de un porte fijo, mayor que el resultado de ocupar mascaras. Un ejemplo de estas áreas de interés se puede apreciar en la Figura 16 donde para un conjunto variado de imágenes se recorta el área de interés de los lentes, barba y bigotes.

Figura 16 "Ejemplo Área de Interés, Lentes, Barba y Bigotes"

Lo siguiente fue entonces recortar con la información que se dispone las áreas antes mostradas. En un principio se utiliza la misma forma que en [4] y que se puede apreciar en la Figura 14, para la extracción de la cara. O sea, extraer un cuadrado desde el punto central entre los ojos con distancias proporcionales a la distancia entre los ojos. Por ejemplo, dado las siguientes distancias:

d: Distancia entre los ojos.

d_1: Distancia desde el Punto medio de los Ojos hacia arriba.

d_2: Distancia desde el Punto medio de los Ojos hacia abajo.

d_3: Distancia desde el Punto medio de los Ojos hacia la derecha.

d_4: Distancia desde el Punto medio de los Ojos hacia la izquierda

Para la extracción de la cara se debe tomar $d_3 = d_4 = 1.3d$, $d_1 = 0.6d$ y $d_2 = 2d$, con lo que se obtiene una ventana cuadrada de $2.6d \times 2.6d$, como se puede apreciar de mejor forma en la Figura 17. Entonces para extraer el área de lentes, barbas y bigotes solo será necesario cambiar el factor por el cual se debía multiplicar d .

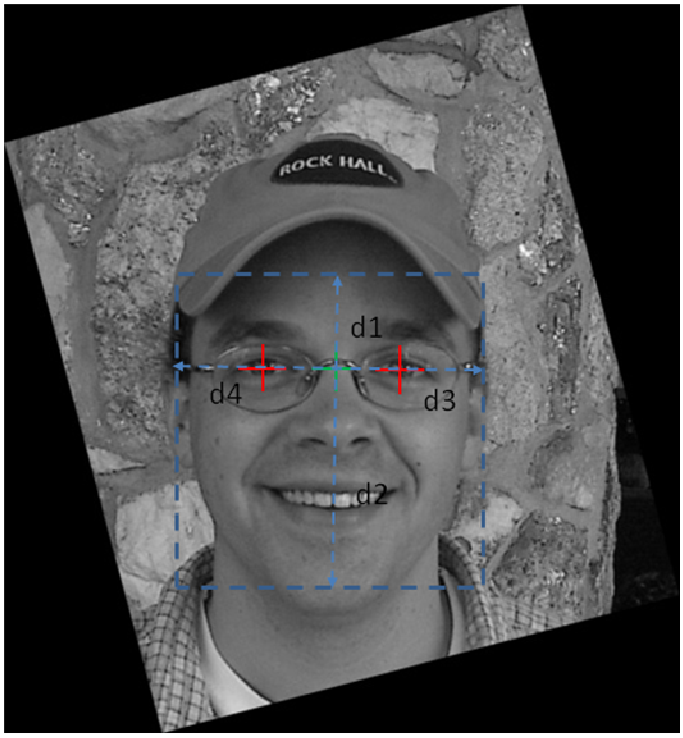


Figura 17 "Ejemplo de Distancias para Recorte de Área de la Cara"

Se obtienen buenos resultados al recortar cualquier área de interés de la forma mostrada. Dado que las caras son bastante proporcionales entre sí, se puede extraer la información de barba y bigotes solo con la información de la posición de los ojos y la distancia entre ellos. Se puede encontrar incluso en artículos para dibujantes donde se hace uso de las proporciones de la cara, como en [21]. El problema surge ya que clasificadores tanto de barba como de bigote ocupan áreas de interés de la cara que distan mucho de la posición de los ojos. El resultado entonces de recortar barba y bigotes de la forma antes señalada no es tan bueno. La proporcionalidad de la cara de la cual se está haciendo uso no es tan exacta como lo que se había pensado, o bien no entrega los resultados con la exactitud necesaria. En la Figura 18 se puede apreciar las diferencias que aparecen al recortar el área de la barba proporcionalmente a la distancia entre los ojos, se aprecia que comúnmente aparece información sobre la boca e incluso los bigotes. Esta información extra puede afectar de manera contundente los resultados en el entrenamiento, o sea, confundir los clasificadores de una u otra menara.

Figura 18 "Ejemplo de las Diferencias al Cortar la Barba solo con la Información de los Ojos"

La solución es entonces ocupar la posición del centro de la boca que se obtiene del marcado manual de las caras. Ocupando esta posición se obtiene de mejor manera el área de la barba y los bigotes. Y se soluciona el problema de la información extra en las imágenes de entrenamiento. Pese a esto el recorte de las imágenes es un proceso que va muy de la mano con el entrenamiento. Dependiendo de lo que se quiso detectar o clasificar se recorto la foto de distintas formas y tamaños. Muchos portes y proporciones se tuvieron que probar para obtener los mejores resultados. Cabe recalcar que los recortes de la Figura 16 fueron hechos con el punto medio de la boca para evitar los errores.

Junto con la solución previamente descrita se suma como objetivo al presente trabajo el poder detectar la boca de forma automática una vez detectada la cara y los ojos dentro de una imagen. Así fue como luego de muchas pruebas se decidió dar inicio a la detección de bocas dentro de la cara. Siempre con el objetivo de mejorar los resultados del clasificador de barba, el clasificador de bigotes y además por la ventaja que tiene agregar una característica al detector de ojos ya desarrollado en [2]. Como ya se ha mencionado y explicado en este trabajo la construcción y uso de un detector dista en ciertos pasos de la de un clasificador, incluso, en la construcción de las bases de datos respectivas. Primero se dará a conocer los pasos para obtener la base de datos del detector de boca, para luego, continuar con el resto de los clasificadores.

- Base de datos del Detector de Boca

Para detectar las bocas dentro de las caras se procedió de forma similar a la detección de ojos. Se recortaron aproximadamente 8200 bocas centradas para la clase de entrenamiento y se

uso el espejo de estas para la clase de validación. Cada una con tamaño horizontal igual a la distancia entre los ojos y tamaño vertical igual a la mitad de la distancia entre los ojos. Las distancias de recorte de las bocas surgieron luego de probar con tres diferentes proporciones. Como se muestra en la Figura 19 la primera foto de cada ejemplo representa las medidas ya mencionadas y las siguientes dos otras medidas ocupadas extrayendo las bocas de la imagen con distancias mayores.

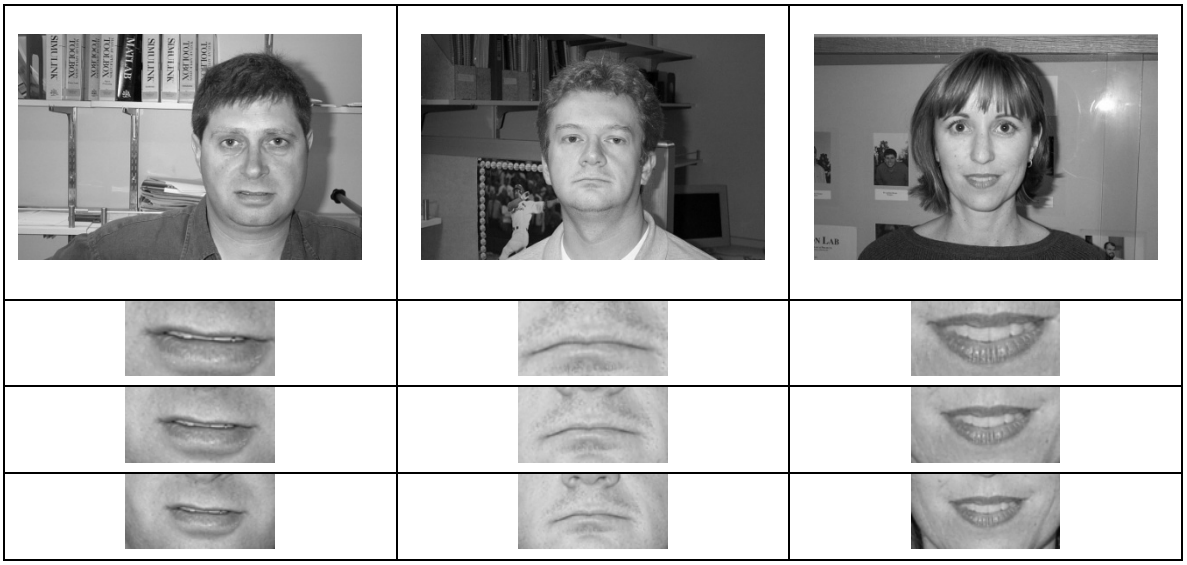


Figura 19 "3 Ejemplos de distancias para el corte de Boca: Cerca, Medio y Lejos"

Como ya se sabía que el detector de ojos ocupaba ventanas cuadradas de 24× 24 pixeles se tomó la decisión de recortar la boca en un principio de 12× 24 pixeles y luego en 18× 36 pixeles. Se sabe que el detector de ojos funciona con imágenes de ojos de al menos 24× 24 pixeles. Entonces no se necesitaran imágenes de mayor resolución al elegir cualquiera de las dos medidas para el detector de boca (12× 24 o 18× 36 pixeles). Una foto en que se puedan detectar los ojos, sería suficiente para poder detectar la boca. Ambos resultados fueron probados en la sección 5.3.1.

Para obtener los ejemplos de la clase negativa (no-boca) se recorto por cada una de las bocas cuatro imágenes de bocas no centradas para la clase de entrenamiento y 4 imágenes de

bocas no centradas para la clase de validación. El número de la clase negativa fue aproximadamente de 32800 imágenes.

El proceso de recortar las no-bocas no es tan directo y luego de variadas pruebas se decidió hacer lo siguiente. Al tener una boca centrada existen infinitas posibilidades de obtener una no-boca. Se decide entonces recortar las ocho imágenes de bocas no centradas que ejemplifican (según el autor) de mejor manera el conjunto de no-bocas, como se puede apreciar en la Figura 20. Luego, sobre estas ocho bocas no centradas o no-bocas se eligen cuatro al azar para el conjunto de entrenamiento y las cuatro restantes para el conjunto de validación. Con esto se aprovecha de obtener un conjunto de validación más variado que solamente ocupando el espejo de las imágenes. También se variaron estas ocho distancias de manera aleatoria. Evitando así un recorte de exactamente la misma área para cada uno de los ejemplos del conjunto de las no-bocas.

Figura 20 "Ejemplo de Recorte de Bocas, 8 Opciones de Bocas no Centradas"

Las distancias desde el punto central de la boca a las cuales se centraron estas no bocas también fueron motivo de muchas pruebas hasta alcanzar el mejor resultado.

Finalmente se obtuvo la base de datos que dio paso a los entrenamientos del detector de boca. Como ya se ha señalado previamente, múltiples entrenamientos y pruebas deben ser desarrollados antes de obtener las bases de datos que den el mejor resultado. No fue directo encontrar la mejor combinación antes de hacer los entrenamientos. En la Figura 21 se pueden apreciar ejemplos de la clase Boca y No-boca.

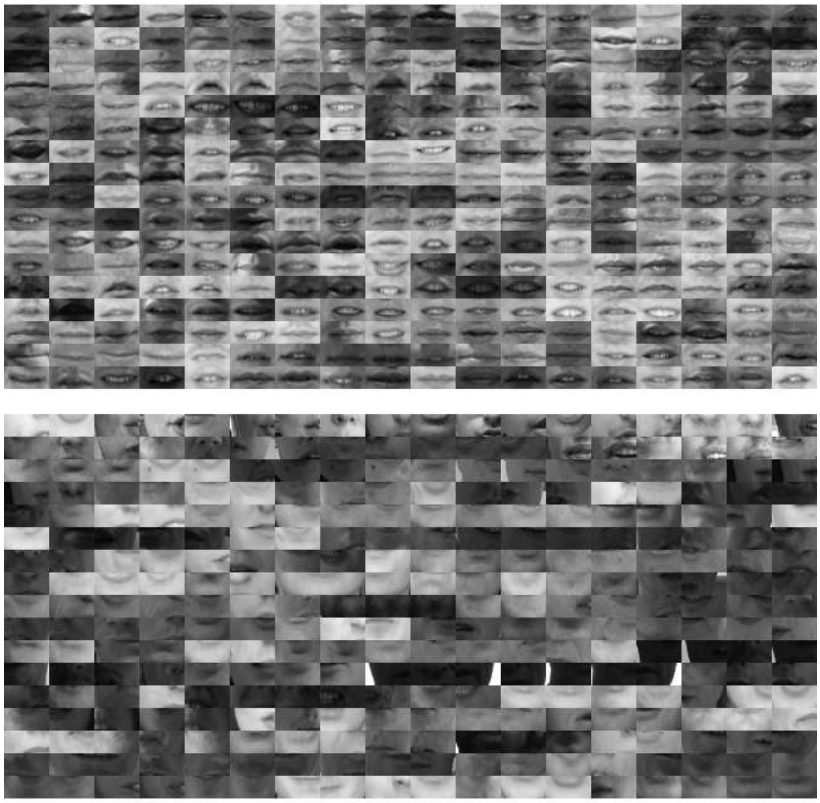


Figura 21 "Ejemplos Bocas y No-Bocas Recortadas"

Cada uno de los programas de recorte, tanto de bocas, no bocas, barba, lentes y bigotes se trata actualizaciones de los programas ocupados en [4] para estos propósitos.

- Base de datos del Clasificador de Lentes

Dentro de los 3 clasificadores de características faciales desarrollados, la base de dato del clasificador de lentes fue la más simple de obtener. Como ya se tenían los ojos previamente marcados, se dispuso solamente a cambiar las distancias del recorte desde el centro de los ojos. Como lo explicado previamente para cortar la cara desde la imagen.

En este caso y usando las mismas distancias que se explicaron con anterioridad para el recorte de la ventana de la cara en la Figura 17, se dispuso que $d_3 = d_4 = 1.2d$, $d_1 = 0.6d$ y $d_2 = 0.6d$, o sea, un rectángulo de proporciones 1:2 y tamaño $1.2d \times 2.4d$. La ventana recortada fue llevada a una resolución o tamaño de 24× 48 pixeles. El funcionamiento del clasificador de lentes se restringe al del detector de ojos. Debido a que los ojos habían sido entrenados con áreas cuadradas de 24× 24 pixeles cualquier imagen donde se detectaran de forma positiva los dos ojos tendría una resolución suficiente para el clasificador de lentes entrenado con 24× 48 pixeles. De todas maneras se evaluó también el uso del clasificador con ventanas de 12× 24 pixeles. A continuación en la Figura 22 se muestra un ejemplo del resultado del recorte de las imágenes para generar la base de datos tanto de la clase Lentes como de No-Lentes.

Figura 22 "Ejemplos Extracción Área Lentes y No-Lentes"

Luego de cortadas las imágenes la base de datos de lentes cuenta con 2665 imágenes de ejemplos y la base de datos de la clase no-lentes con 4928 imágenes de ejemplos. A continuación en la Figura 23 se muestra un set de ejemplos de imágenes de Lentes y No-Lentes extraídas de la base de datos y usadas en los entrenamientos.

Figura 23 "Ejemplos Lentes y No-Lentes Recortados"

- Base de datos del Clasificador de Barba

Los diferentes tipos y matices de las barbas hacen suponer que su clasificación no será del todo fácil. Así también, como generalmente solo un bajo porcentaje de las imágenes en las bases de datos contienen imágenes de personas con barba el reunir una base de datos que fuese lo suficientemente descriptiva no es un trabajo fácil.

Dado que se decide la construcción del detector de bocas, el cual entregaría como resultado el punto medio de la boca, el recorte de área de la barba varió sutilmente con respecto al recorte de la cara entero o de los lentes. Esta vez se toma como punto de referencia el punto central de la boca. Siendo nuevamente d la distancia entre los ojos las distancias para hacer el recorte fueron $d_3 = d_4 = d$,$d_1 = -0.2d$ y $d_2 = 1.2d$, como se puede apreciar en la Figura 24. Con estas dimensiones se abarca toda el área donde se puede encontrar barba y se evita tener información extra de dientes y labios. Se obtiene así un rectángulo de $d \times 2d$ que fue llevado a una dimensión de 24$\times$ 48 pixeles. Esta dimensión no representa un problema si se piensa que las bocas son detectadas con rectángulos de 12$\times$ 24 pixeles. Al momento de hacer cualquiera de estos recortes primero se alinea la cara con respecto a los ojos marcados como se ha explicado con anterioridad y se aprecia en la Figura 14. El mismo procedimiento determinará la clasificación de las características faciales. Los ojos serán detectados primero, luego la boca y enseguida será extraída el área de interés respectiva. Esto se verá con más detalle en la sección 4.1.3.

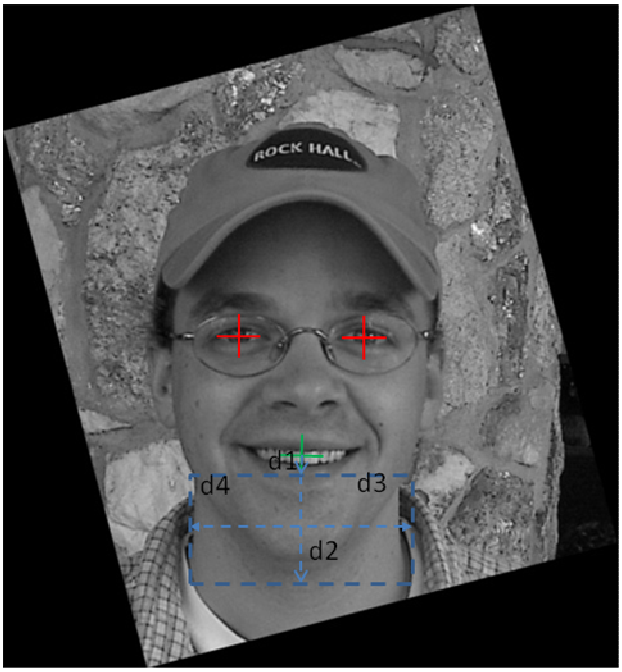


Figura 24 "Ejemplo Distancias Recorte para Barba"

Luego de recortar todas las imágenes de la clase Barba y de la clase No-Barba, el conjunto de las imágenes de Barba fue demasiado pequeño. Se decide entonces utilizar pequeñas variaciones de cada una de las imágenes del conjunto. Por cada foto de ejemplo se generó cuatro variaciones.

Estas variaciones, fueron introducidas en [4] y aplicadas en este trabajo con ciertos cambios que se pueden ver en el Anexo B. Se forman al extraer el área de interés haciendo traslaciones en un porcentaje de la distancia entre los ojos. A diferencia que en [4] donde se utilizo traslaciones de a lo más dos pixeles. Este cambio se debe a que las traslaciones se realizan antes de cambiar el tamaño de la foto al tamaño deseado, en este caso 24 × 48 pixeles. Si la imagen es de una resolución medianamente alta la variación en a lo más dos pixeles no significaba casi nada o incluso nada al cambiar la resolución del recorte a los 24 × 48 pixeles requeridos. En cambio si se usa una medida proporcional a la distancia entre los ojos se garantiza que a pesar de la dimensión de la imagen original las variaciones contendrán distinta información. Habría servido también hacer la traslación después de recortar la imagen pero, debido a la naturaleza del algoritmo creado era imposible. A continuación se muestra en la Figura 25 ejemplos de áreas de barbas con sus respectivas variaciones.

Figura 25 "Ejemplos Extracción Áreas de Barbas y Variaciones"

Si bien estas variaciones ayudan en el entrenamiento, lo mejor sería tener el mayor número posible de imágenes de distintas personas ocupando distintos tipos de barbas. Es

evidente que se requiere de un gran trabajo de recaudación de imágenes el cual no fue posible para este trabajo. En la Figura 26 se muestra un set de ejemplos de la clase Barba y No-Barba. Luego de recortadas todas las imágenes con sus respectivas variaciones se obtuvo una base de datos para los entrenamientos de 2675 ejemplos de la clase Barba y 3370 ejemplos de la clase No-Barba.

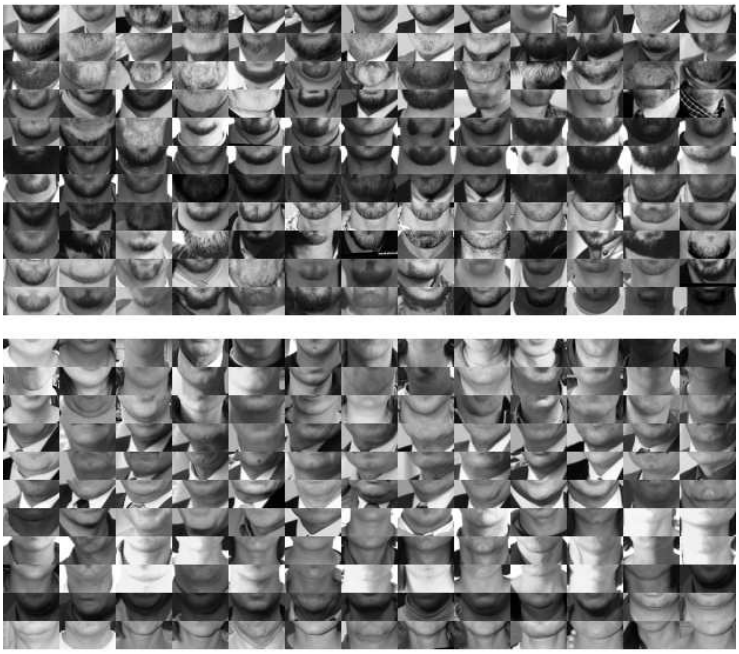


Figura 26 "Ejemplos Barba y No-Barba Recortadas"

- Base de datos del Clasificador de Bigotes

Al igual que en la clasificación de barba, existen numerosas diferencias en los distintos tipos de bigotes que comúnmente se usa. Debido a lo último, la clasificación de bigotes enfrentara las mismas dificultades que el de barba.

Siguiendo la regla anteriormente descrita para el recorte del área de la barba el detector de bigotes solamente vario las distancias desde el centro de la boca. Así, siendo nuevamente d la distancia entre los ojos, las distancias para hacer el recorte son $d_3 = d_4 = 0.5d$,$d_1 = 0.5d$ y $d_2 = 0$. Tomando estas distancias se obtiene un rectángulo de $0.5d \times d$. Este

rectángulo es llevado a un principio a una escala de 24× 48 pixeles y luego a 12× 24 pixeles. Si se considera las dimensiones para la detección de ojos y la detección de boca tomar 24× 48 pixeles para la clasificación de bigotes es notablemente excesivo. Podría ser que en una foto donde se detecten de muy buena forma ojos y bocas no sea posible clasificar los bigotes. Sin embargo el usar una dimensión mayor en las imágenes de entrenamiento puede dar mejores resultados para la clasificación. De todos modos, el clasificador se evalúa también con las bases de 12×24 pixeles para determinar cual funciona de mejor manera. Puede ser también, debido a que los el área de los bigotes es tan pequeña y aunque existan muchos estilos todos exhiben características similares usar imágenes solo de 12× 24 pixeles sea suficiente. A continuación se muestra la Figura 27 con un ejemplo de extracción del área de los bigotes.

Figura 27 "Ejemplo Extracción Área Bigotes"

También se hizo uso de las variaciones para el clasificador de bigotes, al igual que para la barba, de cuatro variaciones por cada imagen en la clase positiva (bigotes). Luego de recortadas todas las imágenes con sus respectivas variaciones se obtuvo una base de datos para los entrenamientos de 3252 ejemplos de la clase bigotes y 3443 ejemplos de la clase No-bigotes. En la Figura 28 se muestra un set de ejemplos de la clase bigotes y No-bigotes.

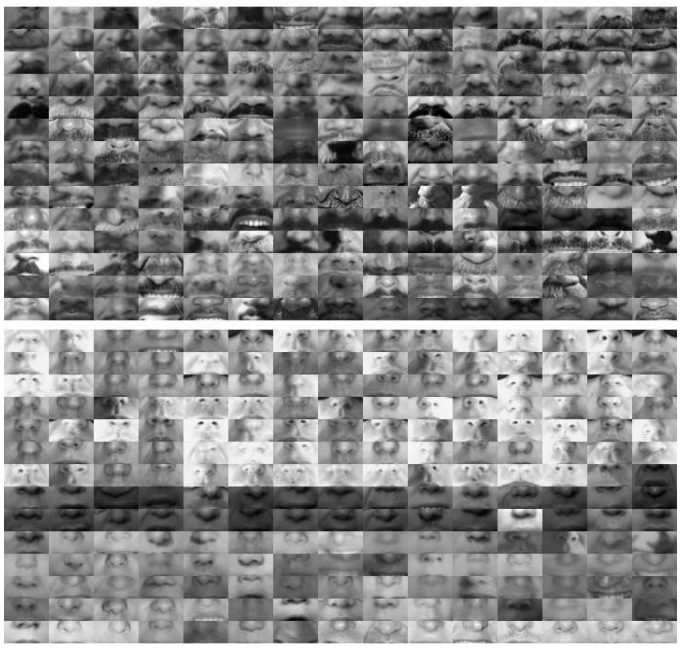


Figura 28 "Ejemplo Base de Datos Bigotes y No-Bigotes"

- Base de datos del Clasificador de Edades

A diferencia de todos los clasificadores antes explicados, el clasificador de edades difiere en varios puntos. Uno de ellos es que no se conoce predeterminadamente ninguna área especial o de mayor importancia dentro de la cara, la cual se podría extraer para hacer los entrenamientos. Se tuvo que generar las bases de datos con el área de toda la cara. Por consiguiente el recorte se hizo exactamente de las mismas dimensiones que el de la Figura 14. Por otro lado y más importante aún, es que este clasificador conto con cuatro clases distintas, niños, jóvenes, adultos y ancianos, por lo tanto la construcción del clasificador cambio significativamente con respecto a los anteriores. Más detalles de cómo se procedió a el entrenamiento de este clasificador serán explicados en la sección siguiente. Una vez reunidas las 4 bases de datos, con sus respectivas variaciones, se contó con un número de 2076 ejemplos de niños, 3284 ejemplos de jóvenes, 2651 ejemplos de adultos y 2040 ejemplos de Ancianos.

Ninguna base de datos disponible en la red para motivos de investigación cuenta con las características necesarias salvo la base de datos encontrada en [16]. Esta ayuda a obtener ejemplos en bases que carecen de estos, como los niños. Sin embargo, es una base de datos de

fotos escaneadas, por lo que las imágenes presentan ruido que puede afectar de sobremanera los entrenamientos. Por estas razones es que en su mayoría la base de datos se conformo de imágenes recogidas de internet. Es por esto que se usaron nuevamente las pequeñas variaciones para ampliar las bases de datos. Tomando siempre en cuenta que si bien ayuda a los entrenamientos no es el óptimo proceder.

Al construir las bases de datos para el clasificador de edades completamente de forma manual existe un grave problema, en ciertos casos es imposible discernir con precisión cual es la edad de la persona en las imágenes. Por esta razón es que pueden existir casos traslapados en las bases de datos, pero que fueron preclasificados bajo los criterios del autor. A continuación en la Figura 29 se muestra un ejemplo de cada clase del clasificador de edades.

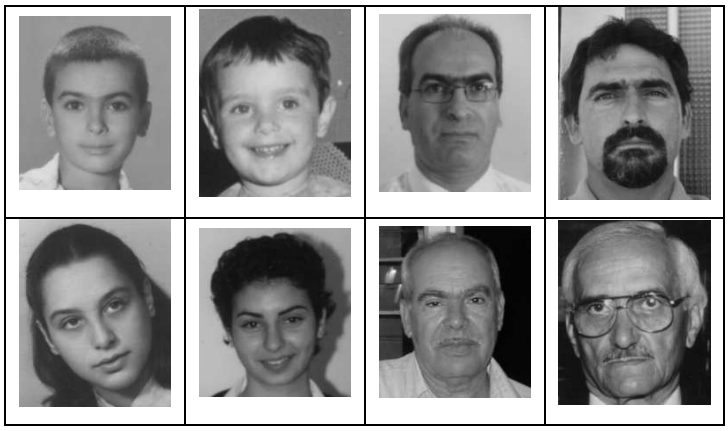


Figura 29"Ejemplos Base de Datos Edades"

4.1.2. Entrenamientos.

Los entrenamientos del detector y los clasificadores se llevan a cabo en el servidor Agami ubicado en el laboratorio de Visión Computacional del Dpto. de Ingeniería Eléctrica de la Universidad de Chile. Estos fueron concebidos durante todo el largo del trabajo de esta memoria, puesto que van muy de la mano con la extracción del área de interés para obtener la base de datos e incluso con el número de ejemplos necesarios en estas.

Ninguno de los clasificadores entrenados cuenta con un solo entrenamiento en su historial. Múltiples entrenamientos fueron realizados y sus efectividades probadas para ver si eran o no la mejor opción. En algunos clasificadores fue necesario el aumentar el número de imágenes de ejemplos en sus bases de datos. Incluso en otros deberían ser ingresados muchos más ejemplos, con el propósito de mejorar los resultados. También, cada distinta opción para la extracción del área de interés o número de variaciones significa que debe ser probado con uno o más entrenamientos. Sin dejar de lado que, como será explicado en esta sección, dentro de los mismos entrenamientos se pueden encontrar diferentes parámetros que ayudaran también a maximizar la efectividad de los clasificadores. También el cambiar estos parámetros se traduce en que se deben probar con múltiples entrenamientos.

En esta parte del trabajo se dará una visión general de los entrenamientos realizados para el detector de bocas y cada uno de los clasificadores, así como sus resultados en las clases de validación usada como parámetro de parada para el entrenamiento. Más detalles con respecto a este último punto serán presentados durante esta sección.

Los entrenamientos se realizaron ocupando los dos tipos de características de los cuales se disponían, LBP modificado y Rectangulares, explicadas previamente en la sección 3.1.2. Además de tener la opción de elegir con que característica hacer los entrenamientos se puede también definir el número de características que se pretende usar para el entrenamiento. Esto se hace mediante dos parámetros de muestro. El primero es el factor de pre-muestro. Este se define como el porcentaje de características que se elije usar del total que es posible generar sobre el área de interés. Este muestreo se genera antes de iniciar el entrenamiento y es elegido al azar por sobre todas las características disponibles. El segundo es el factor de muestreo, este se define como el porcentaje de características que se elije usar en cada iteración del entrenamiento. Estas son seleccionadas al azar del total de características generadas con el factor de pre-muestreo.

Por ejemplo, si se tienen imágenes en las bases de entrenamiento de tamaño 24× 48 pixeles y se decide utilizar características LBP modificadas, el total de características disponibles será de 1152 (En realidad 1151, como se demuestra en [8]). Si se elige un factor de pre muestreo igual al 80% entonces para todo el entrenamiento se dispondrá de 921 características seleccionadas al azar. Luego, si se elige un factor de muestreo del 10%, 92 características serán seleccionadas al azar por sobre las 921 previamente muestreadas en cada una de las iteraciones del entrenamiento.

Como se vio en la sección 3.1.2 las características rectangulares que se pueden generar en una imagen de 24× 48 pixeles es altamente mayor que las posibles generar con LBP modificados. Por esta razón los factores de muestreo deben ser ejidos con más cuidado para no realizar un entrenamiento demasiado largo y no tan efectivo. Este último detalle expuesto afecta de gran manera también la velocidad de los entrenamientos. Son mucho más lentos los entrenamientos con características rectangulares que los con mLBP.

El entrenamiento usando Adaboost no comienza cuando se juntan todos los ejemplos de las clases positivas y negativas, sino que con la generación de las bases binarias de entrenamiento. Estas bases binarias requeridas por el entrenamiento diseñado en [2] no son más que archivos binarios. Estos archivos contienen la información de todas las imágenes de un conjunto de entrenamiento, pixel por pixel en una sola columna y en formato binario. Existen cuatro conjuntos que es necesario reunir antes de empezar un entrenamiento. El conjunto de la clase positiva y negativa tanto de entrenamiento como de validación. Es decir, antes de empezar el entrenamiento se deberá reunir a lo menos los cuatro archivos binarios necesarios.

Este proceso se llevo a cabo en Matlab. Cuando se usa el espejo de las imágenes de las clases de entrenamiento para la validación solo se debe utilizar un comando en Matlab que hace el espejo de la matriz correspondiente a cada imagen. No se tuvo así que operar previamente con las imágenes para generar los espejos.

Una vez generadas las bases binarias se procedió a hacer los entrenamientos. Cada uno de los entrenamientos, como ya se ha mencionado, posee distintos parámetros que pueden ser modificados. Estos parámetros fueron definidos en un archivo de configuración para cada entrenamiento. Este archivo procede de la siguiente forma:

```
## USE "-1" to negate boolean variables, and "1" to confirm them

########################
# Identifier of the training (read by Train_1LayerCascade)
########################
id = "POSITI_vs_NEGATIVE_V1"
########################

########################
# Training Sets (read by Train_1LayerCascade)
########################
positive_train_set_0 = "../BINARYBASES/BinaryBase_POS_ENT.bin"
positive_val_set_0 = "../BINARYBASES/BinaryBase_POS_VAL.bin"
negative_train_set_0 = "../BINARYBASES/BinaryBase_NEG_ENT.bin"
negative_val_set_0 = "../BINARYBASES/BinaryBase_NEG_VAL.bin"

N_CLASSES_OB = 1 # number of positive classes

fromCascade = ""
DIM_H = 24 # Height in pixels of the example images
DIM_W = 48 # Width in pixels of the example images

########################
# Training Parameters (read by Train_1LayerCascade)
########################
classifier_type = 2 # 0 => all (rectangular+lbp), 1 => rectangular, 2 =>LBP
mirror_for_negative = -1 # do you want to mirror the validation sets?
max_Classifiers = 1000 # number of classifiers to be selected during boosting.

########################
# Training Parameters (read by train_Cascade_MO() at AdaBoost-MO.c)
########################
sampling_factor = 10.0 # percentage of classifiers considered during
each iteracion of AdaBoost (randomly sampled)
percentage_pre_sampling = 100.0 # percentage of classifiers pres-sampled
(sampled befor starting boosting)
image_mask_features = "../MASKFOLDER/Mask_v1.png" # "face mask": mask for pre-
filtering features
pre_weight = 1
min_TNR_layer = 0.995
min_TPR_layer = 0.995

########################
# Features
########################
nluts = 1 # Number of LUTs for the rectangular features
```

```
nbins = 16 # Number of bins for the LUTs for the rectangular features
```

Cada uno de los parámetros que se encuentran en esta este archivo de configuración cambia la forma de proceder de los entrenamientos. En seguida se explican los parámetros que se ocuparon en este trabajo al hacer cada uno de los entrenamientos.

```
id = "POSITI_vs_NEGATIVE_V1"
```

Nombre o ID del clasificador o detector a entrenar.

```
positive_train_set_0 = "../BINARYBASES/BinaryBase_POS_ENT.bin"
positive_val_set_0 = "../BINARYBASES/BinaryBase_POS_VAL.bin"
negative_train_set_0 = "../BINARYBASES/BinaryBase_NEG_ENT.bin"
negative_val_set_0 = "../BINARYBASES/BinaryBase_NEG_VAL.bin"
```

Ubicación y nombres de las bases binarias de cada clase.

```
DIM_H = 24 # Height in pixels of the example images
DIM_W = 48 # Width in pixels of the example images
```

Corresponde a la dimensión de las imágenes de ejemplo a utilizar en el entrenamiento. Este parámetro es de suma importancia ya que en los archivos binarios toda la información de todas las imágenes se encuentra en una sola columna.

```
classifier_type = 2 # 0 => all (rectangular+lbp), 1 => rectangular, 2 =>LBP
```

Tipo de clasificadores débiles que se desea ocupar. Estos pueden ser rectangulares, LBP modificados o bien los dos juntos.

```
max_Classifiers = 1000 # number of classifiers to be selected during boosting.
```

Número máximo de clasificadores que se pueden ocupar para cada uno de los entrenamientos. Esto si no se alcanza el límite de parada del entrenamiento que será explicado a continuación

```
sampling_factor = 10.0 # percentage of classifiers considered during
each iteracion of AdaBoost (randomly sampled)
percentage_pre_sampling = 100.0 # percentage of classifiers pres-sampled
(sampled befor starting boosting)
```

Corresponde al factor de muestreo y pre-muestreo del entrenamiento. Como se explico previamente, estos parámetros se ocupan para elegir el número de clasificadores totales y por iteración usados en el entrenamiento.

```
image_mask_features = "../MASKFOLDER/Mask_v1.png" # "face mask": mask for pre-filtering features
```

Ubicación y nombre del archivo de mascara a utilizar en los entrenamientos.

```
min_TNR_layer = 0.995
min_TPR_layer = 0.995
```

Estos dos parámetros corresponden a las mínimas tasas de detección que se requiere alcanzar en las clases de validación para hacer un stop del clasificador. Siempre y cuando no se alcance primero el número máximo de clasificadores definido previamente.

Con los parámetros antes expuestos se da inicio a cada uno de los entrenamientos. Para un set de bases binarias los parámetros que pueden influir de mayor forma en un entrenamiento son los parámetros de muestreo y la máscara. Luego del recorte del área de interés en la generación de las bases de datos fueron estos los parámetros con los que más se experimenta al momento de hacer los entrenamientos.

4.1.3. Arquitectura Seleccionada para Detectores y Clasificadores de Características Faciales.

En seguida se hará un pequeño resumen de las arquitecturas de funcionamiento del detector de Bocas y cada uno de los clasificadores entrenados.

i) Detector de Bocas

Como se ha visto y explicado previamente, el detector de bocas conto con una arquitectura muy parecida al detector de ojos creado en [4]. O sea, se extrae un área de la cara detectada para hacer la búsqueda más rápida y eficiente.

La única diferencia notable entre el detector de ojos y el de boca es que el segundo depende del primero. Esta dependencia surge primero en el recorte de las imágenes para la generación de las bases de dato y previamente en la detección de las bocas. Las rutinas encargadas de la extracción del área de interés de las caras rotan previamente la imagen y dejan los ojos en el plano horizontal. Ya que esta rotación hace del conjunto de entrenamiento un conjunto menos variable se decide ocuparlas también para el recorte de las bocas. Es por esto que las bases de

dato del detector de boca no presentan bocas rotadas. Producto también de esta decisión antes de realizar la detección de la boca se deben primero detectar los ojos. Con la posición de los ojos detectados se debe alinear la cara para así extraer el área de interés de la cara. Solo después de esto se puede comenzar con la detección en dicha área de interés. El detector de boca desarrollado en este trabajo no detecta de la mejor manera bocas con rotación en el plano, pero, tampoco lo necesita ya que al detectar los ojos rota la cara al plano horizontal.

Si bien el proceder anterior da una ventaja y es el que se requieran solo de bocas en el plano dentro del conjunto de entrenamiento. Crea también desventajas y es el que de no detectarse los ojos será imposible hacer una detección de la boca. Sin embargo, todos los clasificadores que se propondrán a continuación de una forma u otra utilizan la información de los ojos para la clasificación. De este modo el resultado negativo en la detección de los ojos trunca totalmente cualquier clasificación que posteriormente se desee hacer.

La arquitectura del detector de boca desarrollado es la mostrada en la Figura 30. Primero se detectan las caras sobre la imagen. Luego, a cada una de las caras es necesario detectar los ojos, con esto se obtienen las caras rotadas. En seguida se extrae el área de la cara necesaria para la detección de la boca. Finalmente, por sobre esta área que ya ha sido alineada con respecto a los ojos se detecta la boca. Siempre teniendo en cuenta que la detección de la boca sobre esta área incluye todas las etapas que un detector basado en las imágenes requiere, las cuales fueron vistas y explicadas en la sección 2.2.

Al detectar la boca en una imagen que es un área rotada de la imagen original se deben hacer múltiples cálculos matemáticos para obtener las coordenadas de la boca en la imagen original. La posición de la boca en la imagen del área recortada se puede simplemente sumar a la posición donde se inicio el recorte sobre la imagen rotada, con esto se obtiene la posición de la boca en la imagen rotada. Luego, siguiendo el procedimiento inverso al expuesto en la Figura 14, se encuentran las coordenadas de la boca en la imagen original.

Cabe destacar que todo este procedimiento se lleva a cabo en un programa en C++ diseñado por Rodrigo Verschae para la detección de caras, ojos y previa clasificación de género y modificado por el autor para la detección y clasificaciones aquí expuestas. En seguida en la Figura 30 se puede observar la arquitectura del detector de boca.

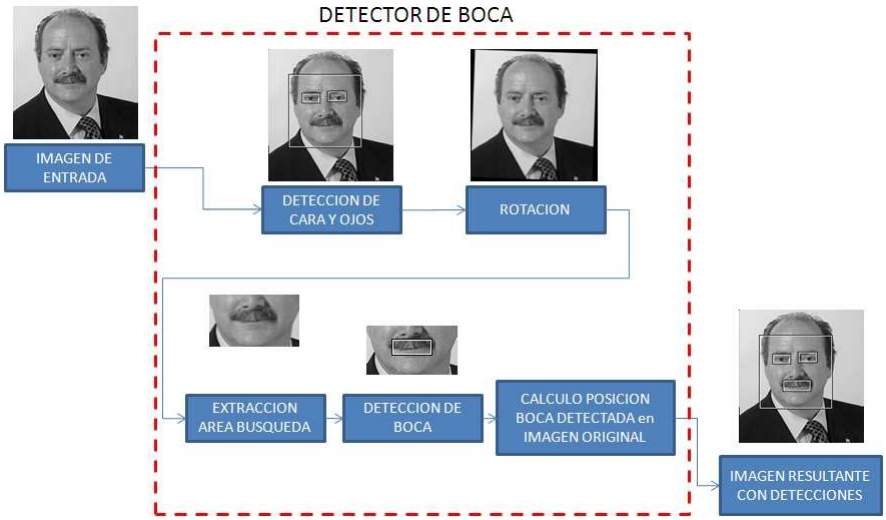


Figura 30 "Arquitectura Detector de Boca"

ii) Clasificador de Lentes

El clasificador de lentes también ocupa como pasos previos la detección de cara, la detección de ojos y la alineación de la cara con respecto a los ojos. Luego se procede a extraer el área de interés para hacer la clasificación de lentes. Esta clasificación entrega una confidencia, la cual puede ser mayor o menor que cero. En un principio se podría tomar como 0 el punto de corte de la decisión, o sea, mayores que cero poseen lentes y menores que 0 no. Pero, es posible que algún punto de corte distinto de resultados mejores, como se verá más detalladamente en la sección 5.2. La arquitectura del clasificador de lentes es la mostrada en la Figura 31 a continuación.

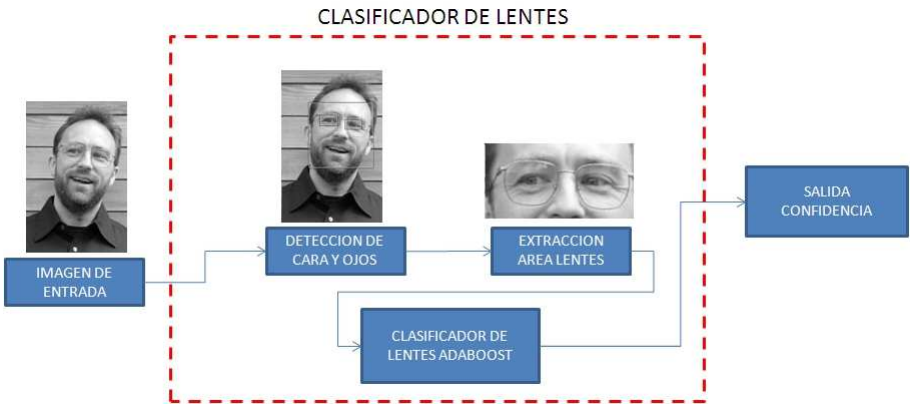


Figura 31 "Arquitectura Clasificador de Lentes"

iii) Clasificador de Barba

Por su parte el clasificador de barba usa la detección de la boca para la extracción del área de interés. Obtenido ya el punto central de la boca detectada en la imagen original se procede a la extracción del área de interés. El área de interés consiste en un área alineada con respecto a los ojos pero que usa como referencia la posición central de la boca detectada. Para hacer los recortes en el programa en C++ antes mencionados se necesitan las coordenadas de los ojos, con las cuales una función especial alinea la imagen y extrae el área que se desee. Para hacer más simple el procedimiento se decide hacer uso de la misma función en el caso del clasificador de barba. La diferencia es que ahora se usa la posición de la boca como posición de referencia. Para lograr esto se creó sendas coordenadas ficticias en la imagen rotada. Estas tienen como coordenada en X la coordenada en X respectiva de cada de cada ojo y coordenadas en Y la coordenada en Y del centro de la boca, como se aprecia en la Figura 32. En seguida se calculan estas coordenadas en la imagen original para luego, usando la función antes mencionada, extraer el área para el clasificador de barba.

De esta forma el área de interés resulta alineada con respecto a los ojos pero extraída usando como referencia el punto central de la boca en el eje Y y el punto central entre los ojos en el eje X . En la Figura 31 es posible apreciar de mejor manera cada uno de los pasos seguidos para la extracción del área de interés usando las coordenadas ficticias expuestas.

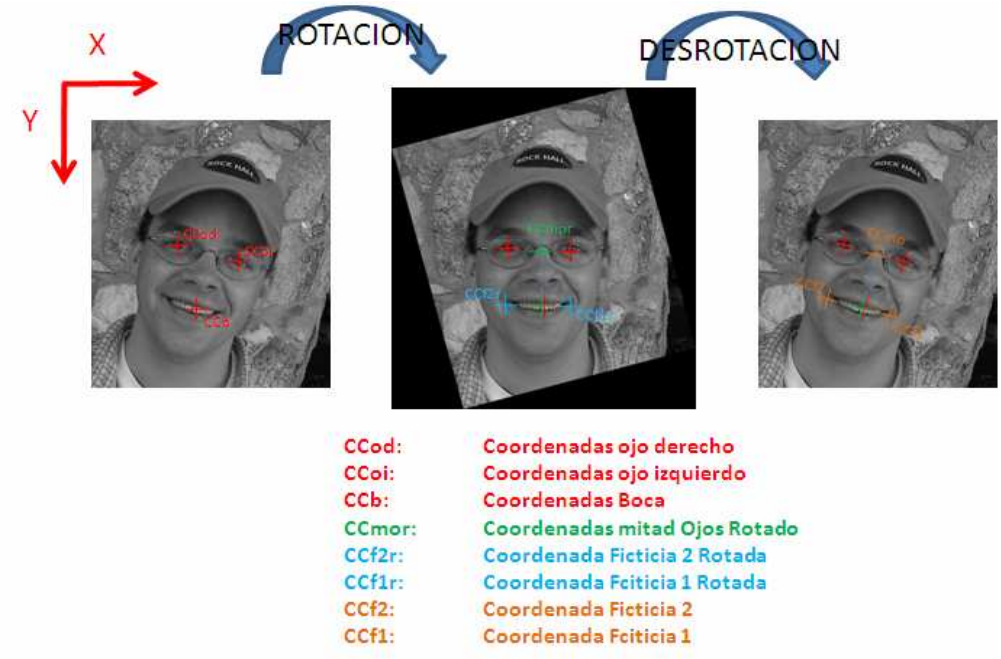


Figura 32 "Ejemplo Extracción con Coordenadas de la Boca"

Así, la arquitectura del clasificador de barba quedo dada por la Figura 31. Nuevamente se necesita de la detección de cara, ojos y ahora también de la detección de boca. Luego de la extracción del área de interés que queda definida por el procedimiento antes explicado, se pasa por el clasificador Adaboost previamente entrenado. Su salida, nuevamente y al igual que el detector de lentes queda dada por la confidencia.

Figura 33 "Arquitectura Clasificador Barba"

iv) Clasificador de Bigotes

El clasificador de bigotes siguiendo la regla del clasificador de barba ocupa el área extraída de la imagen usando la posición detectada de los ojos y la boca. No existe ninguna diferencia en su arquitectura con el anterior, que restando obviedad, queda dada por la Figura 34 a continuación. Es importante destacar nuevamente que entre la detección de cara, ojos y boca y la extracción del área de interés de los bigotes existe un vasto procedimiento ya explicado en el clasificador anterior.

Figura 34 "Arquitectura Clasificador de Bigotes"

v) Clasificador de Edad

Como se explico en la sección 3.2 la reducción del problema multi-clase a problema de dos clases tiene varios tipos de soluciones. La solución aplicada en este trabajo es la más conocido y presupone dividir el problema de n clases en n-1 problemas binarios. Existen también distintas formas de hacer esta división. Debido a la naturaleza del problema se decidió hacer lo siguiente. Se formo cada uno de los cuatro conjuntos por separado, con lo que se obtuvo cuatro bases de entrenamiento y validación, una por clase. Luego se junto la clase niños y jóvenes, y se entreno en contra de la clase adultos y ancianos, con lo que se obtuvo el primer problema binario. El que se haya procedido de esta manera se debe a que las características de niños y jóvenes son más cercanas entre ellas, al igual que las de adultos a las de ancianos. Luego de hacer esta primera separación entre niños-jóvenes y adultos-ancianos, se procedió a entrenar los niños versus los jóvenes por un lado, y los adultos versus los ancianos por el otro. Como ya se ha visto con anterioridad, al hacer la clasificación de edad, es imposible presuponer un área que funcione de mejor manera que otra, por lo que se ocupa toda el área de la cara detectada. Siempre, alineándola primero con respecto a los ojos detectados. Finalmente la arquitectura del clasificador de edades queda dada por la Figura 35 a continuación.

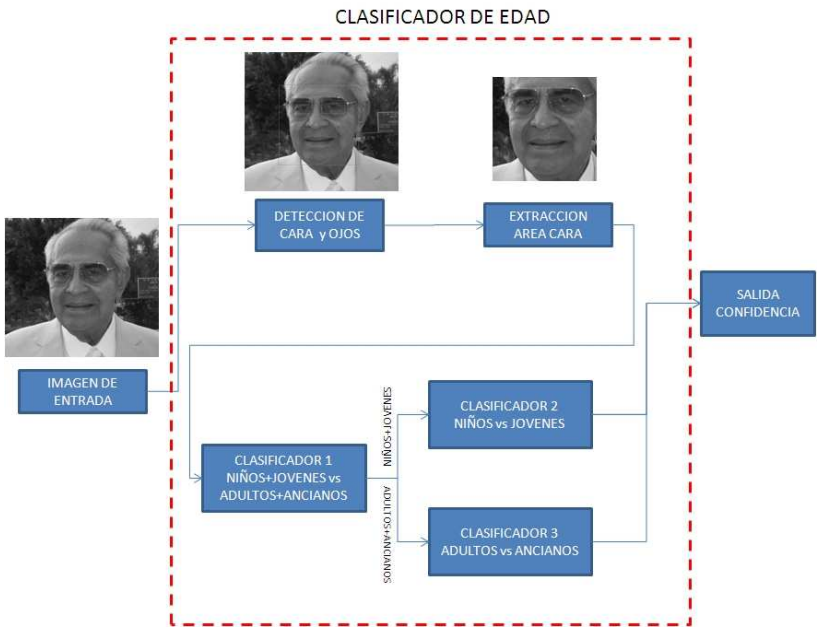


Figura 35 "Arquitectura Clasificador de Edad"

Nuevamente la salida del clasificador es una confidencia. Esta vez dependerá si ha salido del Clasificador 2, de niños versus jóvenes o del Clasificador 3, de adultos versus ancianos. En la siguiente sección se explica cómo esta arquitectura básica puede ser levemente alterada con el objetivo de entregar mejores resultados a una clasificación tan poco clara como la de la edad. Se intenta así manejar los resultados dependiendo del grado de confianza de cada clasificador, o sea, su confidencia.

4.2. Técnicas para manejar los resultados cruzados en base a las confidencias del Clasificador

¿Cómo se podrían mejorar los resultados de un clasificador cuando su respuesta no es más que un grado de pertenencia a una clase cuando se tienen que diferenciar más de una clase? Fue la pregunta que se formula al encontrarse con resultados no tan óptimos para la clasificación de edades. Existen diversos problemas que pueden causar estos resultados "no óptimos". Uno es que el resultado del clasificador siempre variará dependiendo de cuantos ejemplos se tengan en el entrenamiento tanto para la clase positiva como negativa. Sin embargo, este problema debiera hacerse mínimo, maximizando la cantidad y la calidad de los ejemplos en cada clase. Otro problema, como también se menciono en la sección 1.3 es que existen algunos objetivos de clasificación (como las edades) que poseen una gran similitud en la frontera de cada clase, o sea tiene un alto grado de traslape. Este segundo problema es incorregible y se presentara siempre cuando se intente hacer este tipo de clasificaciones. Sin embargo se puede intentar convertir una separación binaria (pertenencia o no a una clase) a una separación en la cual se analice el grado de pertenencia a cada clase, que es lo que se intento en este trabajo.

La clasificación de edad abordada en este trabajo cuenta con 4 clases distintas, las cuales son:

Clase1: Niños: Personas entre 0 y 12 años de edad.
Clase2: Jóvenes: Personas entre 13 y 30 años de edad.
Clase3: Adultos: Personas entre 31 y 60 años de edad.
Clase 4: Ancianos: Personas de más de 61 años de edad.

A diferencia de alguno de los clasificadores de características faciales expuestos, el clasificador de edades presenta un problema poco divisible entre clases. Hay numerosos ejemplos que no se puede decir con seguridad el resultado de la clasificación, o sea, traslapes. Este problema, por ejemplo, queda totalmente de lado al tratar de clasificar el grado de pertenencia de los lentes. Si bien existen muchos y diversos tipos de lentes, existe solamente dos grados de decisión: con o sin lentes. O bien el género de las personas, en el cual, a pesar de que puedan haber grandes similitudes que dificultan la clasificación, la persona pertenece a uno u otro género. Esto hizo incluso difícil construir las bases de datos y como ya se expuso antes quedo al criterio del autor la pre-clasificación de los ejemplos en la base de entrenamientos.

Cuando se habla de la edad es sumamente fácil encontrar ejemplos "traslapados". O sea, imágenes de una persona que parece tener una edad distinta a la que tiene o donde características que son muy comunes en ciertos conjuntos de edades pueden repetirse en los conjuntos adyacentes, como por ejemplo el color del pelo, la calvicie, arrugas, etc.

Obviamente como en todo clasificador, el problema siempre ocurre cuando se trata de los ejemplos difíciles que se encuentran en la frontera de la clasificación. Así, cuando se presenta una imagen de un niño se puede decir con certeza que es un niño. Sin embargo, cuando se trata de un adolecente entre 10 y 15 años la distinción se hace mucho más complicada. Lo mismo sucede cuando se presenta un ejemplo en las otras dos fronteras, entre 25 y 35 años y entre 55 y 65 años.

La solución que se intenta en este trabajo entrega grados de pertenencia según el resultado de las confidencias de cada clasificador. O sea, se intenta encontrar para que valores de las confidencias resultantes la clasificación presume resultados no seguros. Donde entonces se entregan valores de compromiso adicionales para las fronteras entre las clases.

Así, cuando la confidencia de una respuesta muy cerca de la frontera, se mide su distancia relativa al umbral elegido para esa frontera, con el cual se dará un grado de confianza a la clase resultante. Es decir, ante un ejemplo difícil de clasificar entre adultos y ancianos, por ejemplo, la respuesta es que el sujeto corresponde a la clase adulta con un 30% de certeza y a la clase ancianos con 70% de certeza. Se someterá el clasificador a un re-entrenamiento con la base de datos. De este re-entrenamiento se extraerán los intervalos donde se pueden someter a duda los resultados obtenidos y a prueba nuevos ejemplos. Dado que los entrenamientos de los clasificadores Adaboost

no alcanzan una tasa de detección del 100% ni siquiera con las bases de entrenamiento y validación, se puede someter a los clasificadores a un reentrenamiento usando las mismas bases. De este modo se podrán analizar de mejor manera los ejemplos que en el entrenamiento no pudieron ser clasificados positivamente.

La forma de proceder para una frontera es la siguiente. Primero se debe entrenar el clasificador de esta frontera y probarse con las mismas imágenes con las cuales se entrena el clasificador. Se toma entonces el ejemplo peor clasificado de una clase, o sea, que haya sido clasificado como la segunda clase con la confidencia más alta. Y el peor de los ejemplos de la segunda clase, o sea, que haya sido clasificado como la primera clase con la confidencia más alta. Con estos dos peores ejemplos se formulara el intervalo. Este se separara en 10 partes de igual dimensión de los cuales se extraerán los grados de pertenencia a cada clase. De esta forma se procede en cada una de las fronteras. Sin embargo cuando se trate de la frontera de decisión entre niños + jóvenes versus adultos + ancianos es diferente. En este caso el obtener un resultado dentro del intervalo de clasificados con menos seguridad supondrá un proceder distinto. Se tiene entonces que evaluar nuevamente con ambos clasificadores en las otras dos fronteras, o sea niños vs jóvenes y adultos vs ancianos. Los resultados de las confidencias se ponderan para encontrar la mejor clasificación. Por ejemplo, si el resultado de la clasificación entre niños + jóvenes versus adultos + ancianos es de un 30% y 70% respectivamente. Luego, si al clasificar las otras dos fronteras se clasifica el rostro en la imagen como 10% niño, 90% joven, 60% adulto y 40% anciano. El resultado final seria: 3% niño, 27% joven, 42% adulto y 28% anciano.

Así, la nueva arquitectura del clasificador de edad queda dada por la Figura 36.

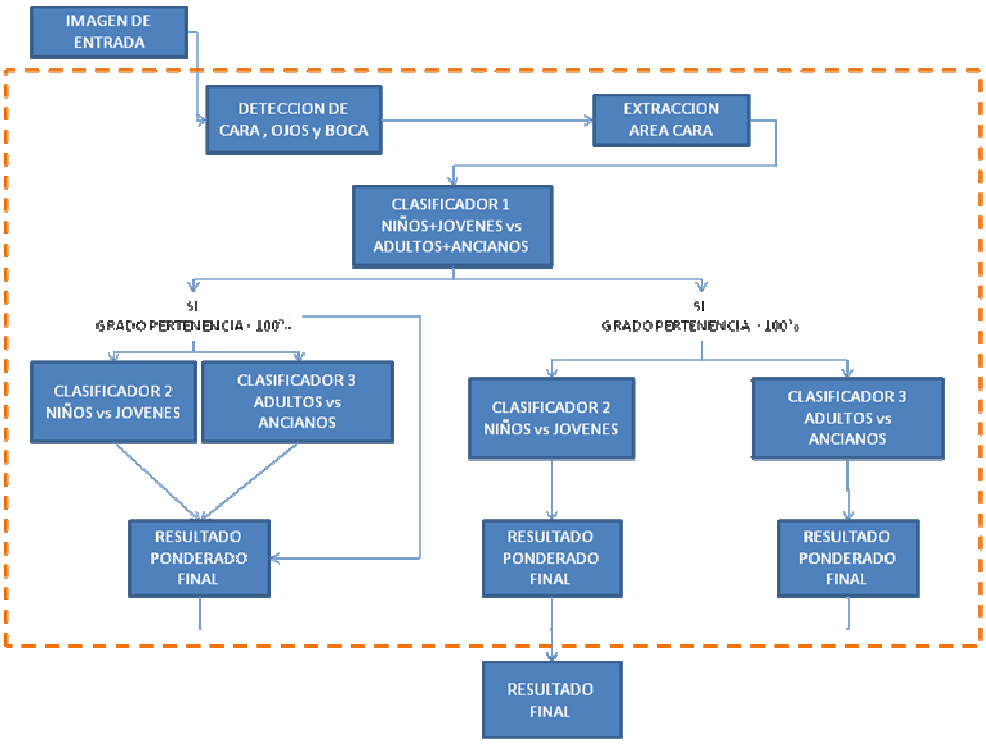


Figura 36 "Nueva Arquitectura Clasificador de Edad"

Si bien esta nueva arquitectura expuesta no representa un mejoramiento concreto de los resultados de cada clasificador, hace mucho más optimo y efectivo el clasificador de edad. O sea, sus resultados se podrán utilizar con mayor seguridad en una determinada aplicación. Es además importante el que, al usar nuevamente las bases de dato de entrenamiento, se podrá clasificar de mejor manera los ejemplos que no pudieron ser clasificados positivamente por el clasificador Adaboost en el entrenamiento.

5. Análisis y Validación de Resultados

El siguiente capítulo tiene por objetivo presentar los resultados del detector y clasificadores antes explicados. Se presentaran las bases en que fueron evaluados, los criterios usados para hacer estas evaluaciones y finalmente, los resultados obtenidos para el detector y cada clasificador. En seguida se hará un análisis de estos resultados y su pertinente comparación con métodos ya existentes.

5.1. Bases de Evaluación

Para probar el desempeño del detector de bocas y los distintos clasificadores se hizo uso de distintas bases de datos. Algunas de estas bases de datos han sido creadas con el propósito de evaluar resultados, por ejemplo detectores de caras o clasificadores de expresiones. Por esto cuentas con suficientes caras en diferentes condiciones y tamaños. Otras, al ser los clasificadores de características faciales tan particulares, tuvieron que ser construidas por el autor.

Para probar la eficiencia del detector de boca se utiliza una base de datos conocida BioID [22]. Esta base de dato está disponible en internet y es usada usualmente para probar la detección de caras. Además se utilizo la misma base de datos utilizada en [4] para probar el desempeño del detector de ojos, la base de datos UChileDB.

La base de datos BioID consiste en 1,521 imágenes con una resolución de 384 × 286 pixeles en escala de grises en ambientes controlados. Cada imagen posee una cara con vista frontal de las 23 personas que se utilizaron para construir la base de datos. Cada imagen presenta cambios de tamaño, fondo, expresión e iluminación. Además, cada imagen contiene un archivo donde se puede encontrar anotada la posición de la boca de donde fácilmente se puede obtener el centro de esta.

Por su parte, la base de datos UChileDB creada para evaluar la clasificación de género y algoritmos de detección de ojos en ambiente no controlados contiene 142 imágenes. En estas se pueden encontrar 343 caras frontales. Estas caras no tienen la posición de la boca anotada, por lo que se deberá anotar manualmente. Es importante también hacer pruebas en una base de datos con ambientes no controlados como esta. Bases de datos con ambientes no controlados se asemejan de mejor manera a como un clasificador podría funcionar en la vida real.

Como ya se ha antes mencionado, el detector de bocas depende también de los detectores de caras y ojos. Por lo que se evaluaran solo las imágenes donde caras y ojos han sido detectadas con buenos resultados.

Para cada uno del resto de los clasificadores se creó una base de datos de prueba, es decir, UCH_BEARD para la evaluación del clasificador de barba, UCH_MOUSTACHE para la evaluación del clasificador de bigotes, UCH_GENDER para la evaluación del clasificador de género y UCH_EYEGLASSES para la del clasificador de lentes. Cada una de esta cuenta con alrededor de 50 a 100 ejemplos de cada una de las clases, o sea, entre 100 y 200 ejemplos en total. Aun cuando estas bases de prueba pueden ser pequeñas contienen imágenes con ambientes totalmente no controlados lo que las hace muy positivas para la evaluación.

Cabe destacar que muchas de las imágenes en estas bases están repetidas puesto que poseen características compartidas, como barba, bigotes y/u otras. Estas, en su totalidad, se obtuvieron de imágenes comunes y corrientes bajadas de Internet por lo que asegura su variedad tanto en tamaño, fondo e iluminación.

Adicionalmente, para la evaluación del clasificador de lentes se ocuparon las 130 imágenes de personas con y sin lentes de la base de datos FERET. Esta base de datos es un buen parámetro de medición ya que contiene imágenes de la mismas personas, pero con y sin lentes.

La base de datos FERET [23] fue creada para la evaluación de algoritmos de reconocimiento facial, lo cual, dista bastante del objetivo de este trabajo. Pero al contener 14,051 imágenes con caras en distintas posiciones (frontales, semi-perfiles y perfiles), entre estas contiene 1,016 imágenes de caras frontales, con la posición de los ojos marcadas. Dentro de estas 1,016 imágenes se puede encontrar alrededor de 130 imágenes de las mismas personas con y sin lentes.

5.2. Criterios de Evaluación

Los criterios usados dependieron si se usaron para la evaluación del detector o los clasificadores. Para la detección de boca al tener las coordenadas de la boca en las imágenes de evaluación se utilizó las curvas del error acumulado. También usadas en [4] para la evaluación del detector de ojos. Para hacer las curvas del error acumulado primero es necesario definir el error relativo. Este es la distancia Euclidiana entre el centro de la boca detectada y el centro de la boca anotada. Como cada imagen puede tener distinta resolución. O sea, una diferencia de 4 pixeles en una imagen de 320x240 no será lo mismo que en una imagen de 1240x760, esta distancia se normaliza por la distancia entre los ojos anotados. Así también se puede calcular el error promedio de todas las detecciones. El error acumulado esta dado entonces por el promedio del error medido en pixeles dividido por la distancia promedio entre los ojos.

El detector de boca será probado en las dos bases de datos antes mencionadas (BioID y UchileDB). Esto se debe a que una presenta ambientes controlados y la otra no. Se cree es fundamental siempre probar los detectores en bases de datos con ambientes no controlados. Estas pruebas entregan resultados más fiables si se requiere usar la aplicación en ambientes no controlados, como el común de las aplicaciones.

Para la evaluación de los clasificadores de Barba, Bigotes y Lentes se utilizan sus tasas de detecciones para el conjunto positivo y negativo. Para obtener el mejor punto de operación de los clasificadores se utilizaran las curvas ROC ya mencionadas y explicadas en la sección 3.1. Pero esta

vez graficando la tasa de detección del conjunto positivo versus la tasa de detección del conjunto negativo.

Como se ha mencionado con anterioridad el detector de edades posee una estructura que da resultados de forma no binaria. Entrega porcentajes de seguridad para cada clasificación. Por esto solo se evalúa el resultado para cada ejemplo y se presentan algunos ejemplos de su desempeño.

5.3. Resultados Obtenidos

5.3.1. Detector de Bocas

Como ya se mencionó los resultados del detector de boca son evaluados usando la curva del error acumulado. Estas se presentan en la Figura 37 y Figura 38 para cada una de las bases de datos evaluadas. Ambas bases fueron evaluadas con detectores entrenados a partir de imágenes de 12× 24 pixeles y 18× 36 pixeles. También se presenta en la Tabla 1 algunos de los puntos seleccionados desde la curva donde se pueden apreciar la tasa detección, el error normalizado y el promedio de error en pixeles.

Como se puede confirmar para la base de datos BioID con el detector de 18× 36 pixeles entrenado con características mLBP, para una tasa de detección del 99.7%, el error acumulado es solo de un 7%, y el promedio del error en pixeles es solo de 3.92%. O sea los resultados de la detección son realmente buenos para esta base de datos. Si se piensa que al marcar las bocas manualmente, el error de marcación puede estar al mismo nivel. O sea, el error detectado es más bien la diferencia entre la boca marcada y la detectada. Sin embargo no es una mala detección. Al contrario, si la diferencia entre estas es solo de 4 pixeles es una muy buena detección.

Del total de imágenes encontradas en la base de datos la detección de boca fue posible llevarse a cabo en 1425 de estas (de las 1520 imágenes originales donde detección de cara y ojos

fue correcta). De entre estas 1425, solo en una el error fue mayor que 100 pixeles, en 2 entre 50 y 100 pixeles, en solo 7 entre 20 y 50 pixeles y solo en 25 entre 20 y 10 pixeles. O sea, solo 35 imágenes tienen errores superiores a 10 pixeles.

También se puede observar, de la Figura 37, que usando el detector entrenado con imágenes de 12× 24 pixeles la curva del error acumulado va un poco más abajo en ciertos tramos que la del detector de 18× 36. Por esto se puede llegar a pensar que sus resultados no son tan buenos. Sin embargo se detectaron mas bocas, 1450 en vez de las 1425 bocas detectadas anteriormente.

Analizando ahora la Tabla 1 se puede observar que para un error normalizado del 7% ya tenemos una tasa de detección del 100%. Con un error en pixeles promedio más bajo que la del detector de 18× 36 pixeles. ¿Cual es mejor? Solo se puede contestar analizando los resultados en una base de datos con ambientes no controlados, como a continuación.

Para la base de datos con ambientes no controlados UchileDB primero se extrajo todas las imágenes en las cuales se detectaba con buenos resultados cara y ojos. Del total de 343 caras se detectaron correctamente ojos y cara en 308 de estas. Por sobre estas 308 caras se realizo la prueba del detector de bocas. Para el detector entrenado con imágenes de 12× 24 pixeles se encontraron 256 bocas, en cambio, para el detector entrenado con imágenes de 18× 36 pixeles solo 159 o sea casi un 40% menos bocas. Además, de la Tabla 1 se puede apreciar que ambos detectores tienen tasas de detección del 100% cuando el error normalizado es mayor que un 12%. A esto se suma que el error en pixeles promedio es nuevamente menor para el detector de 12× 24 pixeles.

Luego de obtener estos resultados para una base de datos con ambientes no controlados se puede asegurar que el detector de boca funciona mejor cuando es entrenado con imágenes de 12× 24 pixeles. Esto se debe a que se logran detectar más bocas, o sea el detector es más

robusto y el error del resultado de la detección es casi igual e incluso menor que para el detector entrenado con imágenes de 18 x 36 pixeles.

Es importante también observar que el detector posee resultados más pobres cuando se prueba en una base de datos con ambientes no controlados donde el error normalizado es casi del doble. Sin embargo los resultados del detector siguen siendo muy alentadores.

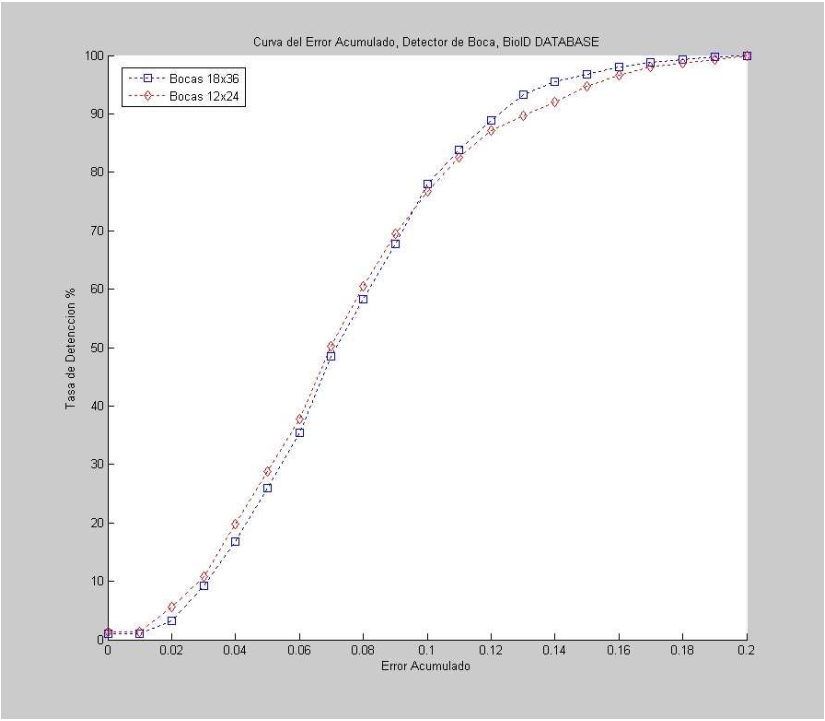


Figura 37 "Curva Error Acumulado Detector de Boca en BioID"

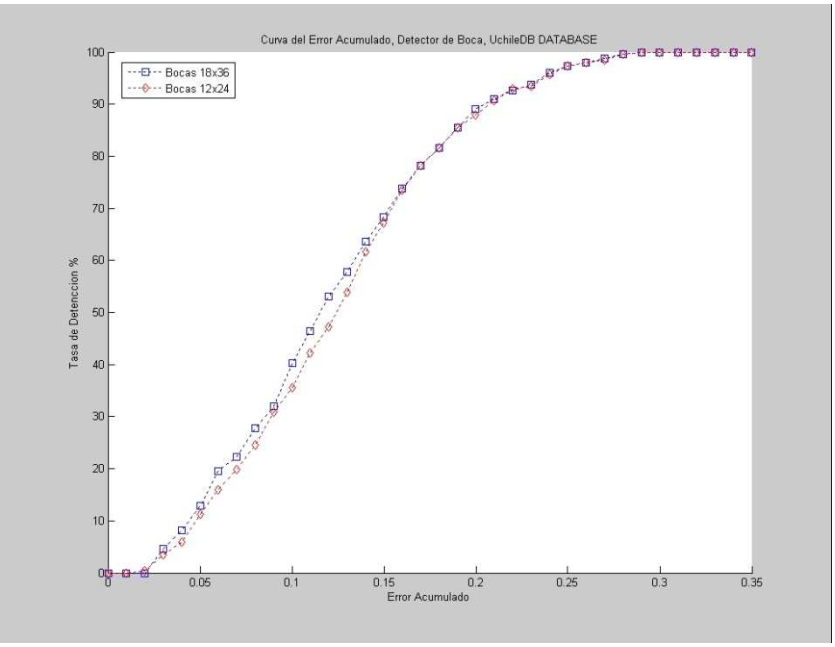


Figura 38 "Curva Error Acumulado Detecto de Boca en UCHILEDB"

BioID-DATABASE (18x36)											
ERROR NORMALIZADO	0.03	0.03	0.04	0.05	0.06	0.06	0.07	0.07	0.07	0.07	0.07
TASA DE DETECCION	1.1	9.4	25.9	48.7	67.7	83.9	93.3	96.7	98.9	99.7	100.0
ERROR PROMEDIO EN PIXELES	1.41	1.81	2.29	2.79	3.17	3.48	3.69	3.80	3.87	3.92	3.94
BioID-DATABASE (12x24)											
ERROR NORMALIZADO	0.00	0.01	0.03	0.04	0.05	0.06	0.06	0.07	0.07	0.07	0.07
TASA DE DETECCION	1.4	5.6	19.8	37.8	60.5	82.6	89.7	94.7	98.0	99.4	100.0
ERROR PROMEDIO EN PIXELES	0.00	0.80	1.57	2.13	2.67	3.21	3.43	3.62	3.78	3.86	3.90
UChileDB-DATABASE (18x36)											
ERROR NORMALIZADO	0.03	0.04	0.05	0.06	0.08	0.09	0.10	0.11	0.11	0.11	0.12
TASA DE DETECCION	4.7	12.9	22.3	32.0	53.1	68.4	78.1	89.1	93.8	98.0	100.0
ERROR PROMEDIO EN PIXELES	1.29	1.75	2.06	2.51	3.27	3.72	4.10	4.45	4.67	4.86	5.03
UChileDB-DATABASE (12x24)											
ERROR NORMALIZADO	0.03	0.03	0.05	0.07	0.09	0.10	0.11	0.11	0.11	0.12	0.12
TASA DE DETECCION	0.4	5.9	19.9	42.2	61.7	73.4	85.5	93.0	95.7	98.4	100.0
ERROR PROMEDIO EN PIXELES	1.00	1.27	2.02	2.84	3.40	3.74	4.12	4.30	4.42	4.54	4.67

Tabla 1 "Resultados Obtenidos para la Detección de Boca"

Los puntos anotados en la base BioID son la esquina derecha e izquierda de la boca y el borde superior e inferior del labio. Con estas 4 coordenadas se calculó el centro de cada una de las bocas usando la intersección de sendas rectas creadas por los 4 puntos. Luego esta información más la distancia entre los ojos también marcados se escribió en un archivo .GND por foto. Este archivo fue leído junto con los resultados del detector para hacer los cálculos.

En cuanto al tiempo de evaluación del detector, en ambas bases de datos donde el tamaño promedio de las fotos es de 380× 240 pixeles fue de 30 ms por foto. O sea, si se quisiera utilizar en video se podrían alcanzar la detección alrededor de 30 FPS. Cabe destacar que el detector al ocupar características mLBP es mas lento que si usara características Rectangulares. Sin embargo los resultados fueron mucho mejores con características mLBP. La resolución de Windows para observar el tiempo que demoran las aplicaciones es de 10ms, por lo tanto cierto error existe en las medidas tomadas.

A continuación, se pueden también observar ejemplos de detección en cada una de las bases de datos en la Figura 39 y Figura 41, adicionalmente se muestran ejemplos de detección de boca en la base de datos FERET en la Figura 40.

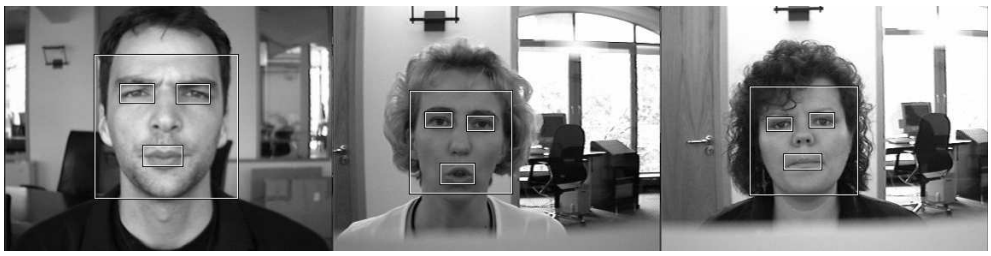


Figura 39 "Ejemplos Detección de Boca en BioID"

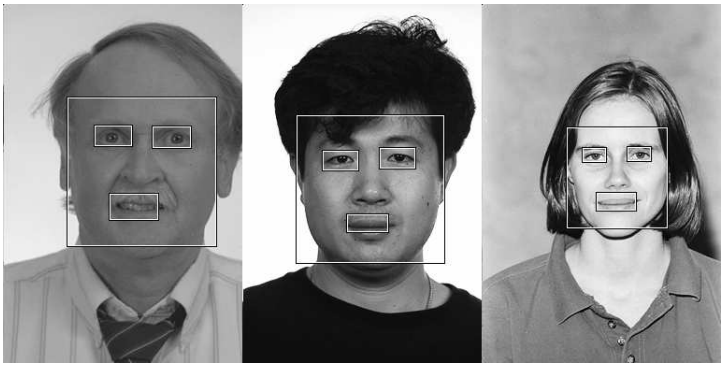


Figura 40 "Ejemplos Detección de Boca en FERET"

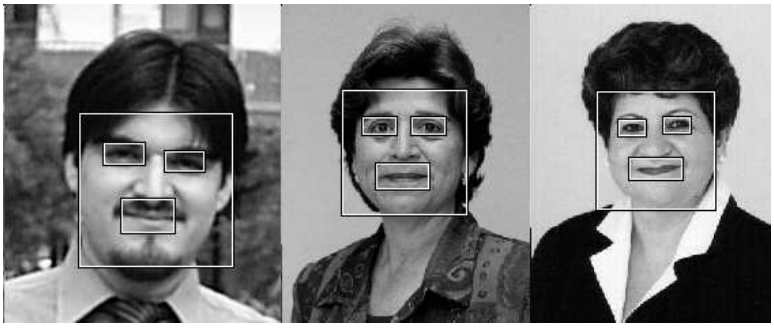


Figura 41 "Ejemplos Detección de Boca en UCHILEDB"

5.3.2. Clasificador de Lentes

La evaluación del clasificador de lentes se hace en la base de datos UCH_EYEGLASSES y en la base de datos FERET. Como se menciono la primera con ambientes no controlados y la segunda con ambientes controlados. De las pruebas se obtienen sendas curvas de tasa de detección que se puede observar en la Figura 42 y Figura 43. En las curvas se presentan los resultados para las características rectangulares y LBP modificados introducidos a lo largo de este trabajo. Como se puede apreciar, los resultados difieren cuando se utilizan distintas características. Se puede claramente observar en ambas figuras que la detección decae de sobremanera al utilizar ejemplos de 12x24 pixeles. Por lo que se utiliza como clasificador final el de 24x48. Se corrobora el que en una base de datos con ambientes controlados como FERET las detecciones son mucho mejores que en UCHEYEGLASSES.

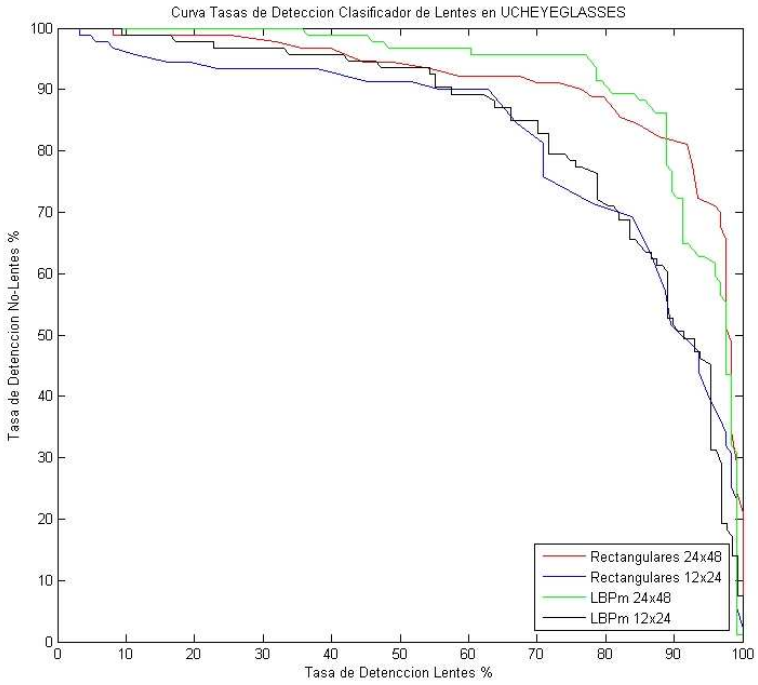


Figura 42 "Curva Tasa de Detección Clasificador de Lentes para UCH_EYEGLASSES"

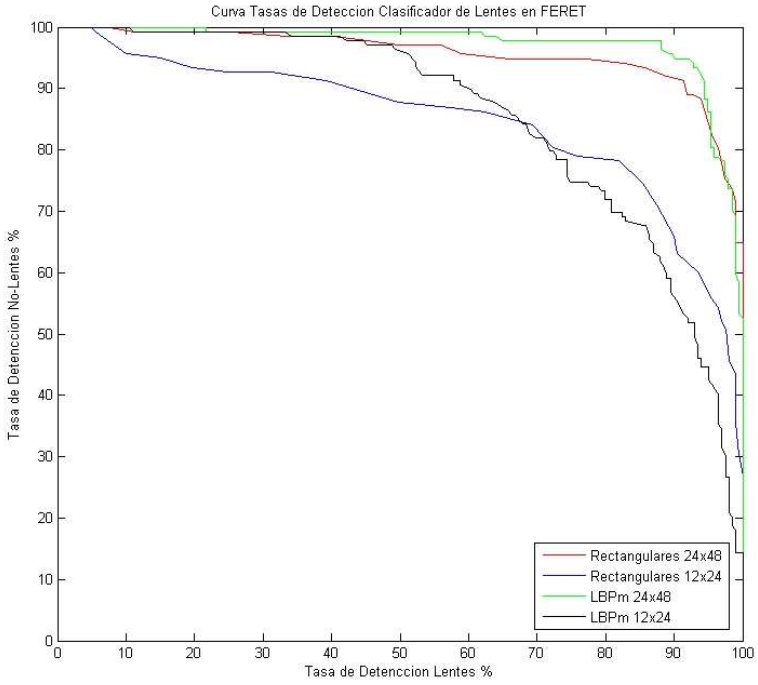


Figura 43 "Curva Tasa de Detección Clasificador de Lentes para FERET"

Eligiendo el punto umbral igual a 2 desde la curva para la base UCHEYEGLASSES. Por tratarse de la base de datos con ambientes no controlados más similar a lo que se en la realidad se podría enfrentar. La tasa de detección del clasificador para los ejemplos positivos es de 91% y para los ejemplos negativos del 80%. Estos resultados son buenos si se toma en cuenta las enormes diferencias que existen entre distintos tipos de lentes. A continuación en la Figura 44 se pueden observar ejemplos de clasificación de la clase positiva y negativa.

Figura 44 "Ejemplos Clasificación Lentes"

Es importante destacar que la detección de lentes depende mucho de cómo se haya desempeñado la detección de los ojos. Lamentablemente en muchos casos el que la gente lleve lentes influyo negativamente en la detección de los ojos. Se tomaran como caso de evaluación solo las imágenes donde los ojos se han detectado de manera correcta. A continuación en la Figura 45 se muestran ejemplos de imágenes en donde los ojos han sido detectados de manera incorrecta o bien no se han detectado debido a que el individuo posee lentes. A futuro, ingresar más imágenes de personas con lentes dentro del detector de ojos podría significar un aumento en la eficiencia de este y del detector de lentes.

Figura 45 "Ejemplos Ojos Mal o no Detectados"

5.3.3. Clasificador de Barba

Al igual que el clasificador de Lentes los resultados del clasificador de Barba serán evaluados con la curva de la tasa de detección y para las dos distintas características. En este caso se tiene un antes y un después del detector de boca. Como se puede apreciar en la Figura 46, y como antes se predijo, el clasificador aumenta enormemente su desempeño al utilizar la información de la boca para hacer la clasificación. Sin embargo se observa que las características rectangulares se despeñan de mejor manera, con o sin la información de la boca, que las características mLBP. Esto puede deberse a que las barbas son una clase tan general que el numero de clasificadores creados cuando se entrena con características mLBP es muy bajo. A futuro podría probarse como resulta la clasificación usando ambas características juntas.

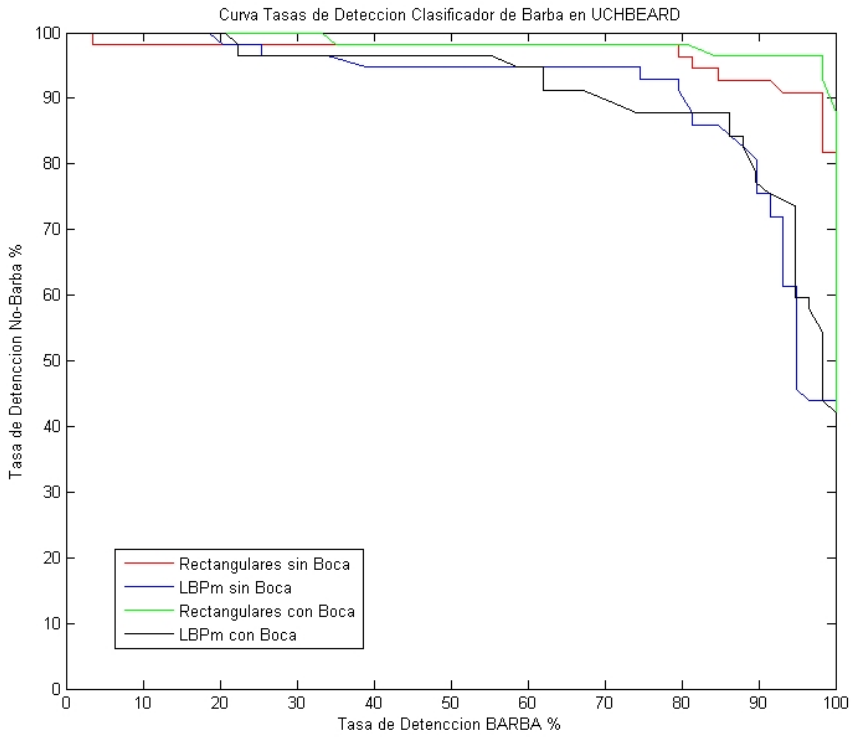


Figura 46 "Curva Tasa Detección Clasificador de Barba"

Si para el mejor clasificador o sea el entrenado con características rectangulares que usa la información de la boca, se toma un umbral igual a 1, su Tasa de Detección para los ejemplos de la clase positiva corresponde a un 97% y para la clase negativa a un 98%. Resultados que son muy positivos para la clasificación de barba, si se piensa en la diversidad del problema. Estudios en bases de datos de prueba con mayor cantidad de ejemplos, podrían formar curvas de mejor calidad.

A continuación, en la Figura 47 se pueden observar ejemplos de detección positiva y negativa en la base de datos de prueba.

Figura 47 "Ejemplos Clasificación de Barba"

5.3.4. Clasificador de Bigotes

Muy parecido al clasificador de barba, en la clasificación de bigotes también existió un antes y un después del detector de bocas. Sin embargo, recortar un área tan específica como la de los bigotes cuando no se ocupa la información de la posición de la boca hace que los resultados fueran excesivamente malos. Por esto ni siquiera se hace una comparación con el antes del detector de bocas.

Siguiendo la regla general, la evaluación de los resultados de la clasificación de bigotes se hará a través de la curva de la tasa de detección, para las dos características estudiadas. También se compara el uso de imágenes de ejemplo de 24× 48 pixeles que parece un tanto excesivo para el área del bigote y de 12× 24 pixeles que parece el tamaño óptimo a usar, dado que las bocas son de al menos 18× 36 pixeles.

Como se puede apreciar en la Figura 48 la clasificación usando 12× 24 y características Rectangulares resulto sorpresivamente buena y muy por sobre el resto. Se piensa que al intentar clasificar esta característica con 24× 48 pixeles se sobredimensiona el problema ya que tiene un área de interés tan reducida. Es por esto que las características más pequeñas funcionan mejor en este caso. Al igual que lo visto en la clasificación de Barba y a diferencia del clasificador de Lentes las características rectangulares obtienen un mejor desempeño.

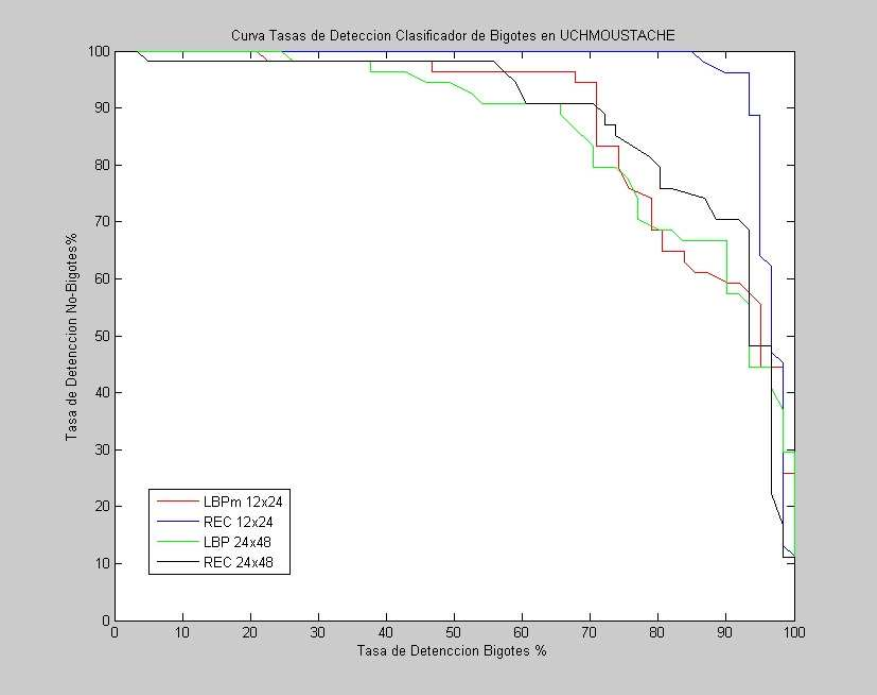


Figura 48 "Curva Tasa Detección Clasificador de Bigotes"

Eligiendo para este clasificador el punto de operación igual a 1, la tasa de detección para la clase positiva es de 96% y para la clase negativa de 93%. Como se puede apreciar, los resultados son tan buenos nuevamente como los del detector de lentes o barba. Esto se debe a que se introdujo la detección de la boca con el objetivo de obtener un mejor entrenamiento y posterior clasificación. Los resultados confirman que es un muy buen método y que se podría utilizar perfectamente en una aplicación real. En la Figura 49 se puede observar ejemplos de la clasificación de bigotes y no bigotes.

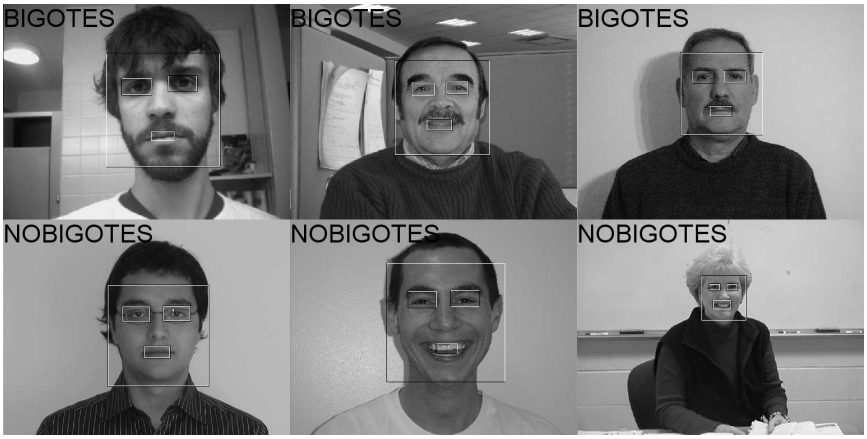


Figura 49 "Ejemplos Clasificación de Bigotes"

5.3.5. Clasificador de Edades

A diferencia de los clasificadores antes mostrados la nueva arquitectura propuesta en la Figura 36 para el clasificador de edad no hace imposible el que se pueda validar con el grafico de la tasa de detección. Esto se debe a que los resultados son entregados por intervalos. Sin embargo, cada clasificador por separado se debe probar en sus respectivas bases de datos para obtener los umbrales que determinaran los intervalos de decisión. Con estos tres umbrales ya elegidos se pudo proceder al post-entrenamiento explicado en la sección 4.2. Eligiendo todas las peores clasificaciones y calculando los intervalos en los cuales se hacen las clasificaciones con porcentajes de seguridad.

Luego de calculados todos los intervalos con las bases de datos de entrenamiento se procede a clasificar la base de evaluación. Se pueden encontrar resultados como los mostrados en la Figura 50, donde tres distintos ejemplos cercanos a las fronteras son clasificados con porcentajes de seguridad.

Figura 50 "Ejemplos Clasificación de Edad"

5.4. Análisis de Resultados

5.4.1. ¿Son los clasificadores AdaBoost un buen método de detección y clasificación de características faciales?

En este trabajo se han visto resultados para la detección y clasificación de características faciales bastante distintos. Se ha podido comprobar que la detección de bocas obtiene muy

buenos resultados si se compara con lo obtenido en [24]. Además por lo visto en [4] sé sabe que los clasificadores Adaboost también son muy buenos para la detección de ojos. Así, se puede afirmar que los clasificadores con estructura Adaboost tienen muy buenos resultados si se les desea usar para la detección de características faciales. Quedaría como objetivo a futuro, desarrollar la detección también de la nariz. Con ojos, nariz y boca, se podría hacer una estimación de pose con mayor exactitud. También queda como objetivo a futuro, el que, las bocas puedan ser detectadas sin el uso de la posición de los ojos, así, se anularía la dependencia que se tiene a estos. Sin embargo, se puede afirmar con seguridad debido a los resultados que la detección de bocas alcanzada en el presente trabajo es tremendamente buena.

En seguida, cuando se trata de clasificar las caras por sus características visibles como lentes, barba y bigotes, vemos que los resultados son bastante buenos. Se piensa que estos resultados pueden mejorar enormemente teniendo bases de datos más representativas de los conjuntos de entrenamiento.

Queda pendiente el estudiar otros métodos para la clasificación de barba y bigotes. Es posible por ejemplo que la detección de piel en el área de interés de ciertamente un resultado bastante bueno. Sin embargo, cuando se trata de la clasificación de lentes, se hace difícil pensar en otros métodos debido a la gran variedad y tipos de estos.

Se destaca además que luego de diseñar el detector de boca los resultados tanto de barba como de bigotes cambiaron positivamente. Se concluye entonces que fue una buena idea la construcción de este. Ambos resultados rectifican que al clasificar este tipo de características, características rectangulares poseen mejor desempeño que las características LBPm. No así cuando se trata de clasificar los lentes.

Con respecto a la clasificación por edad, si bien se logro determinar no con mayor exactitud pero si con mayor validez las edades de las personas. Debido a los resultados obtenidos en esta misma gama de clasificadores, como los son los de género y raza [4], es posible decir que los

clasificadores Adaboost hacen un muy buen trabajo. Primero debido a que son clasificadores basados en las imágenes. Segundo debido a su alta velocidad si se le compara con otros métodos como Support Vector Machine [4]. Tanto clasificadores de Barba, Bigotes y Lentes tienen similares características que el clasificador de genero desarrollado en [4] por lo que los tiempos de detección no difieren en demasía. Pueden ser mayores, en el caso del clasificador de Barba y Lentes, ya que la imagen que se analiza es de mayor resolución. O menores para el clasificador de bigotes donde la imagen tiene la mitad de la resolución. Por otro lado, para el clasificador de edad, al tener que ocupar 3 clasificadores distintos, la clasificación demora el triple de lo normal. Siempre que se usaran características REC los tiempos de detección fueron mucho menores.

Si se reflexiona el que una persona siempre podrá contestar la pregunta ¿tiene o no ese sujeto lentes?, ¿estaba el llevando bigotes? con 100% de seguridad, salvo algunos casos extremos. Por otro lado la respuesta de estas mismas preguntas pero para el rango de edad de las personas siempre será un poco más difícil e indeterminada.

Finalmente se puede afirmar que los clasificadores Adaboost son una buena opción para la detección y clasificación de características faciales, por lo que, adoptar esta técnica y incluso mejorar bases de datos y entrenamientos sería una buena inversión.

5.4.2. Comparación con Métodos Existentes

Siempre resulta conveniente el comparar el trabajo desarrollado con otros métodos, o bien con el mismo método desarrollado por otras personas o laboratorios. El principal problema en este trabajo es que fue muy difícil encontrar otros métodos para algunos de los detector o clasificadores. Así, se hará una comparación para los cuales se pudo encontrar algo de estado del arte.

En cuanto a la detección de boca se puede ver que el resultado en [1] es de un 86% de bocas detectadas. Usando métodos basados en la detección de piel y la geometría de las caras. Sin embargo, debido al ambiente sumamente controlado al cual se pone a prueba, se hace

imposible hacer una comparación precisa del método con el desarrollado en el presente trabajo. Queda claro, que el método desarrollado en este trabajo obtiene resultados excelentes tanto en ambientes controlados como en ambientes no controlados, con tasas de detección cercanas al 100% y errores que son comparables al error que comete una persona al marcar la boca manualmente.

Para la clasificación de Lentes se ha encontrado dos trabajos importantes [25] y [26]. En [25] se ve como usando método similar basado en Adaboost se han encontrado resultados un poco mejores, con una tasa de detección superior al 98%. Hace suponer el que las bases de entrenamiento ocupadas en [25] fueron obtenidas de mejor manera. También se desarrolla en dicho trabajo una comparación de este método con SVM. En esta comparación se ve al igual que en [4] que la clasificación de características faciales usando clasificadores Adaboost da iguales o mejores resultados. Además con velocidades sorprendentemente mayores.

Cabe destacar que en [25] se han ocupado para el entrenamiento y la evaluación no las mismas imágenes, pero sí, procedentes de las mismas bases de datos con ambientes controlados. En general esto podría influir en los resultados. Como se vio en 5.3.2 la clasificación de lentes obtiene mejores resultados al ser probada en bases de datos con ambientes controlados. En [26] vemos como con modelos PCA se alcanzan sumamente altos porcentajes de detección de lentes. Sin embargo también se observa que las imágenes de prueba tienen una alta resolución y son de ambiente muy controlado. Esto se debe a que el objetivo era la localización y extracción de los lentes de la imagen y no solo la clasificación de estos.

Finalmente, para la clasificación de edades se ha encontrado también dos trabajos [27] y [28]. En [27] se propone distinguir las edades de los individuos en 3 clases: bebes, jóvenes-adultos y adultos mayores. Por lo que se evita de sobremanera las difíciles fronteras abordadas en el presente trabajo. Sus resultados son buenos, sin embargo, hacen uso de las características geométricas de las caras. Esto restringe mucho el problema al tipo de personas usadas en el experimento.

Por otro lado en [28] se abordan al igual que en el presente trabajo 4 clases de clasificación: bebes, jóvenes medios, adultos jóvenes y adultos mayores. El problema se aborda eso sí y al igual que en [27] usando la geometría de las caras. Para luego hacer una clasificación

con respecto a esta. Primero se extrae la información de la posición de los ojos, nariz y boca, para luego hacer una clasificación. Los resultado obtenidos para la clasificación en su bases de evaluaciones alcanzan el 81.58%, lo cual lo hace bastante efectivo. Sin embargo se remite a usar ejemplos solamente de personas de origen asiático, lo cual puede afectar en la geometría de las caras.

Dejando de lado este último punto se hace importante notar que en ambos trabajos la discriminación de imágenes de bebes alcanza niveles cercanos al 100%. En el futuro hacer un previa discriminación de los beses podría dar buenos resultados. Es importante también recalcar que en [27] se ocupan clasificadores basados en la geometría, lo cual, según lo estudiado previamente, los hace menos robusto que lo que se intenta hacer en el presente trabajo. En [28] en cambio, se extraen las distancias entre ojos, boca y nariz (información de la geometría facial). Luego esta información es usada para entrenar una red neuronal, lo cual lo hace más robusto y confiable. Sería interesante a futuro investigar si las características a entrenar podrían ser las mencionadas anteriormente usando los ya creados detectores de ojos y boca.

En cuanto a los otros clasificadores no se ha encontrado a la fecha ningún estudio que puedan dar similares resultados para hacer las comparaciones. Por sus excelentes resultados se induce que son buenos métodos para las tareas que fueron propuestos. Tanto el clasificador de Barba como de Bigotes tiene resultados muy positivos en bases de datos con ambientes no controlados que permiten ocuparlos con eficacia en cualquier tipo de aplicación.

6. Comentarios y Conclusiones

En este trabajo de memoria se estudio el problema de la detección y clasificación de características faciales usando algoritmos estadísticos. Cada uno de estos fue tratado en forma independiente, de manera que se podrían implementar como bloques separados en sistemas distintos.

La elección de Adaboost como algoritmo de aprendizaje estadístico se hizo en base a los buenos resultados presentados en distintos trabajos. Queda claro que Adaboost funciona con muy buenos resultados tanto para la detección de características faciales, como para la clasificación de estas. En imágenes con ambientes no controlados y distintas características. Por esto además no se incluyo el estudio de la clasificación con otros tipos de algoritmos.

Cuando se utilizan algoritmos basados en las imágenes como Adaboost las bases de datos son de suma importancia y deben ser lo suficientemente representativas de las clases que se desea evaluar. Para los casos de clasificación de características faciales las bases ocupadas en algunos de los clasificadores no fueron lo suficientemente numerosas ni variadas. Es por esto que en un futuro armar más amplias y mejores bases de datos mejoraría los resultados.

El uso de la creación semi-automática de bases de datos propuesto en [4], usado y modificados en este trabajo, entrego una excelente fuente para la creación de bases de datos. Siempre y cuando dichas bases tuvieran la información correcta (marcadas con el punto central de los ojos y la boca). De otro modo, se uso el marcado manual de imágenes para obtener las bases de datos.

Los entrenamientos realizados para cada uno de estos clasificadores son también un paso de suma importancia y como ya se ha mencionado, va de la mano con la obtención del área de interés en la formación de las bases de datos. Estos se deben llevar a cabo en múltiples ocasiones tratando de

maximizar sus resultados. Queda claro que el sistema de entrenamiento de clasificadores Adaboost realizado en [2] y utilizado en el presente trabajo es de gran efectividad y podría utilizarse para distintas aplicaciones.

Cada uno de los clasificadores por separado cumplió con el objetivo de clasificación propuesto. Muy buenos resultados se obtuvieron para la detección de bocas, las cuales fueron probadas en 2 bases de datos, 1 con ambientes controlados (BioID) y una con ambientes no controlados (UChileDB). Si bien se reportaron mejores resultados en la base de datos con ambientes controlados, como era de esperarse, los resultados para la detección de bocas en la base de datos con ambientes no controlados son muy buenos. Estos resultados dan gran seguridad que el detector puede ocuparse en otras aplicaciones.

Buenos resultados se obtiene de la clasificación de lentes, que fue probada sobre la base de datos FERET y UCHEYEGLASSES. La primera por contener una gama alta de ejemplos de la clase lentes y imágenes de personas con y sin lentes. Y la segunda por poseer ejemplos con ambientes no controlados. En ambas bases de datos los resultados fueron buenos. Sin embargo, mejores resultados fueron presentados en [25] sobre la base de datos FERET, usando el mismo método.

Se debe además tomar en cuenta que el resultado del clasificador de lentes está acompañado de la correcta detección de los ojos. Es común el que, debido a que las imágenes de ejemplo poseían lentes, la detección de los ojos no tuviera la precisión necesaria para hacer una eficaz clasificación. Aun cuando se quito los resultados en que el detector de ojos se desempeñaba de muy mala forma. Se propone así a futuro hacer una prueba del clasificador con la información de los ojos marcados y no con los ojos detectados. Aunque los resultados pueden mejorar como en [4], en el caso del clasificador de género, el ocupar el clasificador de esta forma no tiene sentido si se desea implementar en un sistema de interfaz de usuario automático.

Tanto los resultados del clasificador de Barba como de Bigotes fueron muy buenos, si se considera lo diverso y distinto de los grupos de clasificación. Sin embargo, queda pendiente a futuro el estudio de otros métodos. Se piensa quizás que un simple estudio del porcentaje de piel detectada en la misma

área que se busca clasificar, podría obtener igual o mejores resultados. Además, el estudio de estos mismos clasificadores sobre bases de datos con más ejemplos entregaría una mejor percepción del funcionamiento de ambos clasificadores.

Es también claro que el objetivo de la detección de bocas, que fue mejorar los resultados del clasificador de barba y bigotes fue plenamente cumplido. Ambos clasificadores mejoraron con enormidad los resultados. Debido a que no se encontró ningún trabajo posterior sobre la clasificación de estas características, se hace imposible la correcta validación de los resultados.

Importante es también concluir que tanto las características Rectangulares como las LBPm funcionan de mejor manera dependiendo del problema abordado, por lo tanto es siempre muy importante el estudio de ambas características al momento de desarrollar una aplicación. Junto con que no siempre es mejor entrenar los clasificadores o detectores con las imágenes de alta resolución. Quedo ejemplificado en el detector de bocas y el clasificador de bigotes donde imágenes de menor resolución reportaron resultados mejores y más robustos.

Para la clasificación de edad, y pese a que se encontraron sendos trabajos con resultados más que alentadores, se considera que la solución propuesta en este trabajo define un resultado más confiable. Además robusto ya que no ocupa información de la geometría de la cara. Sin embargo, por lo visto en ambos trabajos, queda claro que la separación de algunas clases especificas es lo suficientemente efectiva como para ser aplicada en un principio en el bloque de clasificación.

Finalmente, se considera que la implementación de clasificadores y detectores de características faciales usando aprendizaje estadístico fue exitosa. Se observa como la detección de características faciales obtiene resultados asombrosos y con mucha mejor velocidad de proceso que otros métodos también basados en las imágenes. Así también, en el ámbito de la clasificación de las características faciales se ven resultados muy positivos. Sin embargo, se cree posible el poder encontrar resultados aun mejores, por ejemplo, ampliando la variedad en las bases de datos de los entrenamientos. A pesar de

esto el presente trabajo da seguridad para sostener que los clasificadores Adaboost entregan muy buenos resultados para los objetivos propuestos.

Anexo A

FORMATOS DE ARCHIVOS

Formato XML

Formato del archivo xml generado por la interfaz de usuario para la marcación de imágenes:

```
Directory
|
\--- Picture
|
\---- Face1
\---- Face2
\---- Face3
\--- ...
\--- ...
\--- Picture
|
\---- Face1
\--- ...
\--- ...
\--- ...
```

Estructura en forma de tags del archivo XML:

```
<directory absolute_path="" relative_path="">
<picture name="">
<face>
<left_eye x="" y="">
<right_eye x="" y="">
<noose_tip x="" y="">
<mouth_center x="" y="">
</face>
<face>
<left_eye x="" y="">
<right_eye x="" y="">
<noose_tip x="" y="">
<mouth_center x="" y="">
</face>
...
</picture>
</directory>
```

Formato GND

Formato del archivo generado desde el XML respectivo para cada una de las imágenes marcadas:

name= name.jpg
view= Frontal
id= number
left_eye_coords= x y
right_eye_coords= x y
nose_tip_coords= x y
mouse_center_coords= x y
view=Frontal

Anexo B

RUTINAS

Rutina en Linux XML → GND

Rutina en Linux que convierte archivo XML en un archivo por imagen con extensión GND

```
cat $1 |dos2unix| grep -v directory | grep -v xml | Gawk -F"\"| |<" \
'{
    if ($2=="picture"){
        gndfilename=$4 ".gnd";
        printf("name="$4"."$7"\n") > gndfilename;
    }
    if ($2=="face"){
        printf("view="$4"\n") >> gndfilename;
        printf("id= 123123\n") >> gndfilename;
    }
    if ($2=="left_eye"){
        printf("left_eye_coords= "$4" "$7"\n") >> gndfilename;
    }
    if ($2=="right_eye"){
        printf("right_eye_coords= "$4" "$7"\n") >> gndfilename;
    }
    if ($2=="noose_tip"){
        printf("nose_tip_coords= "$4" "$7"\n") >> gndfilename;
    }
    if ($2=="mouth_center"){
        printf("mouse_center_coords= "$4" "$7"\n") >> gndfilename;
    }
}'
```

Modificación Recorte de Cara

Se presenta en seguida, las modificaciones hechas en las rutinas de Matlab para el recorte de las caras usando como punto de referencia el punto central de la boca. Estas se presentan a modo de ejemplo para la rutina utilizada para recortar el área de la barba, similar se utilizo para recortar el área de los bigotes:

```
function recortar_cara_barba_cnboca(im_name,gnd_name,out_name,ang,rep)

% Programa para recortar caras, version 7 03.03.2008
% Mauricio A. Correa P. Email : Mauricio.Knight@Gmail.com
% Oscar Sanhueza R. Email : Osanhuez@Ing.uchile.cl

% Utiliza funcion generar_pares_aleatorios()

% Leyendo imagen y datos

a = imread(strtrim(im_name));
[caras,dd]=leer_caras(gnd_name);

if dd>0
for bb=1:dd

    a = imread(strtrim(im_name));

    x_ojo_iz2=caras(bb,1);
    y_ojo_iz2=caras(bb,2);
    x_ojo_de2=caras(bb,3);
    y_ojo_de2=caras(bb,4);
    x_boca=caras(bb,5);
    y_boca=caras(bb,6);

while 1

% Enderezando la imagen (alineacion de los ojos), calculando  nuevas
% coordenadas de los ojos y la boca.

siz=size(a);
alp=atan((y_ojo_iz2-y_ojo_de2)/(x_ojo_iz2-x_ojo_de2));
a=imrotate(a,alp*360/(2*pi),'bilinear');


if (alp>=0)
% ojo izquierdo
diagi=sqrt(x_ojo_iz2*x_ojo_iz2+y_ojo_iz2*y_ojo_iz2);
angi=atan(x_ojo_iz2/y_ojo_iz2)+alp;
y_ojo_iz=siz(2)*sin(alp)+diagi*cos(angi);
```

```
x_ojo_iz=diagi*sin(angi);
% ojo derecho
diagd=sqrt(x_ojo_de2*x_ojo_de2+y_ojo_de2*y_ojo_de2);
angd=atan(x_ojo_de2/y_ojo_de2)+alp;
y_ojo_de=siz(2)*sin(alp)+diagd*cos(angd);
x_ojo_de=diagd*sin(angd);
%boca
diagb=sqrt(x_boca*x_boca+y_boca*y_boca);
angb=atan(x_boca/y_boca)+alp;
y_boca=siz(2)*sin(alp)+diagb*cos(angb);
x_boca=diagb*sin(angb);

end
if (alp<0)
% ojo izquierdo
diagi=sqrt(x_ojo_iz2*x_ojo_iz2+y_ojo_iz2*y_ojo_iz2);
angi=atan(y_ojo_iz2/x_ojo_iz2)-alp;
x_ojo_iz=siz(1)*sin(-alp)+diagi*cos(angi);
y_ojo_iz=diagi*sin(angi);
% ojo derecho
diagd=sqrt(x_ojo_de2*x_ojo_de2+y_ojo_de2*y_ojo_de2);
angd=atan(y_ojo_de2/x_ojo_de2)-alp;
x_ojo_de=siz(1)*sin(-alp)+diagd*cos(angd);
y_ojo_de=diagd*sin(angd);
%boca
diagb=sqrt(x_boca*x_boca+y_boca*y_boca);
angb=atan(y_boca/x_boca)-alp;
x_boca=siz(1)*sin(-alp)+diagb*cos(angb);
y_boca=diagb*sin(angb);

end


d = abs(x_ojo_iz-x_ojo_de);  % distancia entre los ojos
xp=(x_ojo_iz+x_ojo_de)/2;  % Punto medio entre los ojos
yp=(y_ojo_iz+y_ojo_de)/2; % Altura de los ojos

%generando repeticiones con sus respectivas variaciones
for i=1:1:rep

T=generar_pares_aleatorios(1);
x_ojo_iza=x_ojo_iz+T(1);
y_ojo_iza=y_ojo_iz+T(2);
x_ojo_dea=x_ojo_de+T(3);
y_ojo_dea=y_ojo_de+T(4);
x_bocan=x_boca+T(5)*0.01*d;
y_bocan=y_boca+T(6)*0.01*d;

d = abs(x_ojo_iza-x_ojo_dea);
xmin = x_bocan - 1.0*d;
ymin = y_bocan + 0.2*d;
width = 2*d ;
height= 1*d;

In = imcrop(a,[xmin ymin width height]);
In = imresize(In,[24 48],'bilinear');
imwrite(In,strcat(out_name,'_',int2str(bb),'_',int2str(i),'.png'),'png');

end
break;
end
end
end
```

Extracción de NO-Bocas

Se presenta en seguida, las modificaciones hechas en las rutinas de Matlab para el recorte de las no bocas, usando como punto de referencia la boca marcada:

```
function recortar_no_boca2(im_name,gnd_name,out_name_ent,out_name_val,ang)

% Programa para recortar no bocas, version 1 03.03.2008
% Oscar Sanhueza

% Utiliza funcion generar_pares_aleatorios()

% Leyendo imagen y datos

a = imread(strtrim(im_name));
[caras,dd]=leer_caras(gnd_name);
```

```
if dd > 0
for bb=1:dd

    a = imread(strtrim(im_name));

    x_ojo_iz2=caras(bb,1);
    y_ojo_iz2=caras(bb,2);
    x_ojo_de2=caras(bb,3);
    y_ojo_de2=caras(bb,4);
    x_boca=caras(bb,5);
    y_boca=caras(bb,6);

while 1

% Enderezando la imagen (alineacion de los ojos), calculando  nuevas
% coordenadas de los ojos y la boca.

siz=size(a);
alp=atan((y_ojo_iz2-y_ojo_de2)/(x_ojo_iz2-x_ojo_de2));
a=imrotate(a,alp*360/(2*pi),'bilinear');

if (alp>=0)
% ojo izquierdo
diagi=sqrt(x_ojo_iz2*x_ojo_iz2+y_ojo_iz2*y_ojo_iz2);
angi=atan(x_ojo_iz2/y_ojo_iz2)+alp;
y_ojo_iz=siz(2)*sin(alp)+diagi*cos(angi);
x_ojo_iz=diagi*sin(angi);
% ojo derecho
diagd=sqrt(x_ojo_de2*x_ojo_de2+y_ojo_de2*y_ojo_de2);
angd=atan(x_ojo_de2/y_ojo_de2)+alp;
y_ojo_de=siz(2)*sin(alp)+diagd*cos(angd);
x_ojo_de=diagd*sin(angd);
%boca
diagb=sqrt(x_boca*x_boca+y_boca*y_boca);
angb=atan(x_boca/y_boca)+alp;
y_boca=siz(2)*sin(alp)+diagb*cos(angb);
x_boca=diagb*sin(angb);

end
if (alp<0)
% ojo izquierdo
diagi=sqrt(x_ojo_iz2*x_ojo_iz2+y_ojo_iz2*y_ojo_iz2);
angi=atan(y_ojo_iz2/x_ojo_iz2)-alp;
x_ojo_iz=siz(1)*sin(-alp)+diagi*cos(angi);
y_ojo_iz=diagi*sin(angi);
% ojo derecho
diagd=sqrt(x_ojo_de2*x_ojo_de2+y_ojo_de2*y_ojo_de2);
angd=atan(y_ojo_de2/x_ojo_de2)-alp;
x_ojo_de=siz(1)*sin(-alp)+diagd*cos(angd);
y_ojo_de=diagd*sin(angd);
%boca
diagb=sqrt(x_boca*x_boca+y_boca*y_boca);
angb=atan(y_boca/x_boca)-alp;
x_boca=siz(1)*sin(-alp)+diagb*cos(angb);
y_boca=diagb*sin(angb);

end

d = abs(x_ojo_iz-x_ojo_de);  % distancia entre los ojos



%Creado arreglo con 8 distintas distancias para bocas
dif = [[-1 1;-1.5 0;-1 -1;0 1.5;0 -1.5;1 1;1.5 0;1 -1]];
%creando arreglo al azar entre 1 y 8 sin repeticion
no=randxyC(8,1,9);


% Extrayendo no bocas
for i=1:1:8

dif2=dif(no(i),:);
T=generar_pares_aleatorios(1);
%GENERANDO PEQUEGNAS VARIACIONES
x_ojo_iza=x_ojo_iz+T(1);
y_ojo_iza=y_ojo_iz+T(2);
x_ojo_dea=x_ojo_de+T(3);
y_ojo_dea=y_ojo_de+T(4);
x_bocan=x_boca+dif2(1)*0.35*d+(randxy(0.01,0.02)*d);
y_bocan=y_boca+dif2(2)*0.25*d+(randxy(0.01,0.02)*d);

d = abs(x_ojo_iza-x_ojo_dea);
xmin = x_bocan-0.5*d;
```

```
ymin = y_bocan-0.25*d;
width = floor(1*d);
height= floor(0.5*d);

In = imcrop(a,[xmin ymin width height]);
auxsize=size(In);
auxsize=size(auxsize);
if auxsize(2)==3
    In = rgb2gray(imresize(In,[18 36],'bilinear'));
else
    In = imresize(In,[18 36],'bilinear');
end

%Escribiendo la imagen en el pat de Entrenamiento y luego en el pat Evaluacion
if i<=4
imwrite(In,strcat(out_name_ent,'_',int2str(bb),'_',int2str(no(i)),'.png'),'png');
else
imwrite(In,strcat(out_name_val,'_',int2str(bb),'_',int2str(no(i)),'.png'),'png');
end

end




break;
end
end
end
end
```

Referencias

[1] K. Sobottka y I.Pitas, ”Face Localization and Facil Feature Extraction Based on shape and Color Information”, en Septiembre 1996.

[2] R. Verschae, J. Ruiz-del-Solar, M. Correa, “A unified learning framework for object detection and classification sing nested cascades of boosted classifiers”. Machine Vision and Applications, pp. 85-103, 2008.

[3] Y. Freund, R. Schapire. “A decision-theoretic generalization of on-line learning and an application to boosting”. Journal of Computer and System Sciences, 55(1):119– 139, en Septiembre 1997.

[4] M. Correa, “Herramientas Computacionales para la Anotación, Indexación y Administración de Álbumes de Fotos Familiares”, Agosto 2006, Memoria para optar al Título de Ingeniero Civil Electricista, Universidad de Chile.

[5] Y. Freund, R. Schapire, ”A Short Introduction to Boosting”. Journal of Japanese Society for Artifial Intelligence, 14(5):771-780, en Septiembre 1999.

[6] R. Schapire, Y. Singer, “Improved boosting algorithms using confidence-rated predictions”. Machine Learnning. 37(3), 297–336 (1999).

[7] P. Viola, M. Jones, “Rapid object detection using a boosted cascade of simple Features”. IEEE Conf. on Computer Vision and Pattern Recognition, pp. 511–518 (2001).

[8] B. Fröba, A. Ernst, “Face detection with the modified census transform”. Sixth Int. Conf. on Face and Gesture Recognition, pp. 91–96 (2004).

[9] T. Ojala T, M. Pietikäinen, D. Harwood, “A comparative study of texture measures with classification based on featured distribution”. Pattern Recognition, 29(1): pp.51-59, 1996.

[10] H. Abdenour, “Learning and Recognizing Faces: From Still Images to Video Sequences”, Faculty Of Technology, Department of Electrical and Information Engineering, Infotech Oulu, University of Oulu, pp. 32-33, 2005.

[11] H. Francke, ”Diseño y Construcción de Interfaz Humano Robot Utilizando Gestos Realizados con las Manos”, Octubre 2007, Memoria para optar al Título de Ingeniero Civil Electricista, Universidad de Chile.

[12] C. Papageorgiou, M. Oren, and T. Poggio. “A general framework for object detection”, In International Conference on Computer Vision, 1998.

[13] Y. Freund and R. Schapire. “A decision-theoretic generalization of on-line learning and an application to Boosting”, Journal of Computer and System Sciences, 55(1):119–139, Agosto 1997.

[14] A.M. Martinez and R. Benavente. “The AR Face Database. CVC Technical Report #24”, Junio 1998. http://www-prima.inrialpes.fr/FGnet/

[15] Equinox Corporation - Human Identification at a Distance Database, http://www.equinoxsensors.com/products/HID.html

[16] The FG-NET Aging Database, Face and Gesture Recognition Research Network. http://www.fgnet.rsunit.com/

[17] W. Gao, B. Cao, S. Shan, D. Zhou, X. Zhang, D. Zhao, "The CAS-PEAL Large-Scale Chinese Face Database and Evaluation Protocols", Technical Report No. JDL_TR_04_FR_001, Joint Research & Development Laboratory, CAS, 2004.
http://www.jdl.ac.cn/peal/home.htm

[18] M. J. Lyons, S. Akamatsu, M. Kamachi, J. Gyoba, "Coding Facial Expressions with Gabor Wavelets", Third IEEE International Conference on Automatic Face and Gesture Recognition, Nara Japan, IEEE Computer Society, pp. 200-205, Abril 14-16 1998.

[19] M. Weber, "Frontal face dataset", California Institute of Technology, 1996. http://www.vision.caltech.edu/html-files/archive.html.

[20] N. A. Fox, B. A. O'Mullane, and R. B. Reilly, "VALID: A new practical audio-visual database, and comparative results", Proc. of the 5th International Conference on Audio- and Video-Based Biometric Person Authentication (AVBPA-2005).
http://ee.ucd.ie/validdb

[21] "Learn how to Draw Portraits", Marzo 2008.
http://drawsketch.about.com/b/2006/02/27/learn-how-to-draw-portraits.htm

[22] BioID Face Database. Available on March 2008 in: http://www.bioid.com/downloads/facedb/index.php

[23] P. J. Phillips, H. Wechsler, J. Huang and P. Rauss, "The FERET database and evaluation procedure for face recognition algorithms," Image and Vision Computing J., Vol. 16, no. 5, pp. 295-306, 1998.

24 P. Wilson, J. Fernandez, "Facial feature detection using Haar classifiers", J. Comput. Small Coll. 21,pp 127-133, Abril 2006.

25 B. WU, H. AI and R. LIU, "Glasses Detection by Boosting Simple Wavelet Features", Computer Science and Technology Department, Tsinghua University, Beijing.

26 W. Chenyu, L. Ce, S. Heung-Yeung, X. Ying-Qing and Z. Zhengyou, "Automatic Eyeglasses Removal from Face Images" The 5th Asian Conference on Computer Vision, Melbourne, Australia, Enero 23–25 2002.

27 Y. H. Kwon and N. da-Vitoria-Lobo, "Age Classification from Facial Images", Computer Vision and Image Understanding, Vol. 74, No. 1, pp. 1–21, Abril, 1999.

28 H. Wen-Bing, L. Cheng-Ping and C. Chun-Wen, "Classification of Age Groups Based on Facial Features", Tamkang Journal of Science and Engineering, Vol. 4, No. 3, pp. 183-192, 2001.