UNIVERSIDAD DE CASTILLA-LA MANCHA
ESCUELA SUPERIOR DE INFORMÁTICA

GRADO EN INGENIERÍA INFORMÁTICA

TRABAJO FIN DE GRADO

Detección de Lesiones en Mamografías y Ecografías para el Diagnóstico del Cáncer de Mama

M. Gema Díaz Pizarro

Febrero, 2014

UNIVERSIDAD DE CASTILLA-LA MANCHA
ESCUELA SUPERIOR DE INFORMÁTICA

DEPTO. DE ING. ELÉCTRICA, ELECTRÓNICA,
AUTOMÁTICA Y COMUNICACIONES

TRABAJO FIN DE GRADO

Detección de Lesiones en Mamografías y Ecografías para el Diagnóstico del Cáncer de Mama

Autora: M. Gema Díaz Pizarro

Directora: Noelia Vállez Enano

Codirector: Jesús Salido Tercero

Febrero, 2014

Detección de Lesiones en Mamografías y Ecografías para el Diagnóstico del Cáncer de Mama
© M. Gema Díaz Pizarro, 2014

En virtud del Artículo 7 del Real Decreto Legislativo 1/1996, de 12 de abril por el que se aprueba el Texto Refundido de la Ley de Propiedad Intelectual, modificado por la Ley 23/2006 de 7 de julio, este TFG se considera una obra en colaboración entre las diferentes partes. Por tanto la propiedad intelectual de este TFG, productos y de su difusión será compartida entre el alumno y los directores.

TRIBUNAL:

Presidente: _____

Vocal: _____

Secretario: _____

FECHA DE DEFENSA: _____

CALIFICACIÓN: _____

PRESIDENTE	VOCAL	SECRETARIO
Fdo.:	Fdo.:	Fdo.:

A mi padre,

Resumen

Los nuevos avances tecnológicos y la implantación de sistemas informatizados en clínicas y hospitales ofrecen una gran cantidad de posibilidades en diagnóstico automático de enfermedades.

Puede haber muchas diferencias entre las imágenes médicas a analizar debido a las variaciones existentes entre unas pruebas y otras, así como a las distintas fisionomías de los pacientes. Distinguir lesiones en pacientes donde el tejido de la mama tiene un color similar al de la lesión presenta mayor dificultad. Si a este hecho se une el pequeño tamaño de algunas lesiones y el cansancio de los radiólogos al examinar imágenes continuamente, pueden producirse fallos en el diagnóstico.

El trabajo actual tiene como objetivo la construcción de una herramienta de ayuda para la detección del cáncer de mama, a través del análisis de mamografías y ecografías.

Abstract

New technological developments and implementation of information systems for clinics and hospitals offer a lot of possibilities in disease automated diagnostic.

There may be many differences between medical images to analyze due to the variations between some images and others, as well as the various physiognomies of patients. Lesions distinctions have a higher difficulty in patients where the breast tissue has a color similar to the lesions. If we add to this fact the small size of some lesions and fatigue of radiologists to review images continuosly, diagnosis failures may result.

Current work is aimed at building a tool to help detect breast cancer, through the analysis of mammograms and ultrasound scans.

AGRADECIMIENTOS

En primer lugar, quería agradecer a la directora de este TFG, Dña. Noelia Vállez Enano, su paciencia y dedicación, por ayudarme siempre y dedicar parte de su tiempo a resolver todas mis dudas. Me siento muy afortunada de haberla tenido como directora. Le deseo mucho ánimo con su Tesis Doctoral, sin duda está realizando un gran trabajo. A Dña. Gloria Bueno García y D. Jesús Salido Tercero por brindarme la posibilidad de realizar este proyecto y por la colaboración que han prestado.

Dar las gracias a toda mi familia, soy lo que soy gracias a ellos. A mi padre que, a pesar de dejarnos antes de tiempo, no me cabe la menor duda de que ha estado y va a estar conmigo siempre. Espero que desde algún sitio se acuerde siempre de mí tanto como yo de él. De igual manera, gracias a mi madre. No tengo palabras para agradecerle todo lo que ha hecho. Espero que se sienta muy orgullosa de mí, al menos tanto como yo lo estoy de ella.

Agradecer a mis hermanos, Alfonso y José Miguel, sus continuas muestras de apoyo y confianza. Ellos saben lo importantes que son para mí. Le deseo mucha suerte a Alfonso en la nueva etapa que ha comenzado en Inglaterra, estoy segura de que le va a ir muy bien. De igual modo, gracias a Elena a la que considero una hermana más. Les deseo lo mejor a ella y a José Miguel en la espera de su primer hijo/a, por lo que estamos todos muy ilusionados.

Agradecer a mis compañeros y compañeras más cercanos de la carrera los buenos momentos pasados durante mi etapa en la Universidad. Personas como Silvia, María, Fátima, Laura, Raúl, Javier, Jesús, Rocío, Ana M. y muchas más sin duda han hecho de estos años los mejores años de mi vida. Tengo la certeza de que me llevo amigos/as para siempre.

A mis amigas gracias por su apoyo y preocupación a lo largo de estos años. Por su paciencia y su comprensión en los momentos en los que no pude estar con ellas porque tenía que estudiar o que hacer proyecto. Agradezco su confianza en que todo me iba a salir bien.

A todos los que forman parte del grupo de investigación VISILAB, que tantas cosas me ha aportado y del que me llevo muy buenos recuerdos.

Por último, gracias a todas aquellas personas que han pasado por mi vida en estos años y han influido en mí ya que todas, en mayor o menor medida, tienen un hueco en mi memoria.

ÍNDICE GENERAL

1 Introducción **1**

1.1 Introducción 1

1.2 Diagnóstico del Cáncer de Mama 1

1.3 Clasificadores 2

1.4 Estructura del Documento 3

2 Objetivos **5**

2.1 Objetivos y Enfoque 5

2.1.1 Objetivos específicos 6

2.2 Condicionantes 7

2.2.1 Medios software 7

2.2.2 Medios hardware 10

3 Antecedentes **11**

3.1 Anatomía y Exploración de la Mama 11

3.1.1 Anatomía de la mama 11

3.1.2 Tipos de Tejido Mamario 12

3.1.3 Lesiones del Tejido Mamario 13

3.1.4 Técnicas de Exploración 14

3.2 Imagen Digital 18

3.3 Regiones de Interés 19

3.4 Reconocimiento de Patrones 19

3.4.1 Fundamento Matemático 20

3.5 Sistema básico de reconocimiento 21
3.5.1 Descripción de los sistemas de reconocimiento de patrones 21
3.5.2 Selección de variables 22
3.6 Características de la imagen 23
3.6.1 Haar 23
3.6.2 Texturas 25
3.7 Clasificadores 27
3.7.1 Clasificadores débiles 27
3.7.2 Clasificadores Fuertes 30
3.7.3 Cascada de Clasificadores de Viola y Jones 33

4 Método de trabajo **39**
4.1 Método de Desarrollo de Sistemas Dinámicos (DSDM) 39
4.1.1 Ciclo de Vida 40
4.1.2 Artefactos 41
4.2 Proceso Unificado de Desarrollo (PUD) 42
4.2.1 Fases del Proceso Unificado de Desarrollo 44
4.2.2 Artefactos 45

5 Evolución y Costes **47**
5.1 Evolución del Proyecto 47
5.1.1 Prototipo 1 47
5.1.2 Prototipo 2 47
5.1.3 Prototipo 3 48
5.1.4 Prototipo 4 48
5.1.5 Prototipo 5 49
5.1.6 Prototipo 6 49
5.1.7 Prototipo 7 49
5.1.8 Prototipo 8 50
5.1.9 Prototipo 9 50

5.1.10 Prototipo 10 50
5.2 Estimación de costes 51
5.2.1 Costes de Personal 52
5.2.2 Costes de *Software* 55
5.2.3 Costes de *Hardware* 55

6 Artefactos Generados **57**
6.1 Estudio de Viabilidad 57
6.2 Diagramas de Casos de Uso 58
6.3 Diagrama de Clases 61
6.4 Diagramas de Secuencia 62
6.5 Diseño de la Interfaz Gráfica 62
6.6 Planificación 63

7 Desarrollo de la Aplicación **67**
7.1 Integración algoritmos-interfaz 67
7.2 Herencia y Polimorfismo 68
7.3 Programación multihilo 69
7.3.1 *C++ Windows Threads* 69
7.3.2 Trabajo en segundo plano para la clase *Aplicación* 70
7.4 Modelo Arquitectónico 71
7.5 Patrones de diseño 72
7.5.1 Fabricación parametrizada 72
7.5.2 Patrón Proxy 73
7.6 Pruebas de la Aplicación 73
7.6.1 Diseño de casos de prueba 74

8 Resultados **77**
8.1 Conjuntos de imágenes 77
8.1.1 Ecografías 77

8.1.2 Mamografías 78
8.2 Estimación del error de clasificación 79
8.3 Preparación de los datos de entrenamiento y pruebas 80
8.4 Resultados de los clasificadores 80
8.4.1 Matrices de confusión 81
8.4.2 Curvas ROC 83
8.5 Tiempos Computacionales 86

9 Conclusiones y Propuestas **89**
9.1 Objetivos Alcanzados 89
9.2 Propuestas 90
9.3 Conclusión Personal 90

A Fichero XML **93**
A.1 Estructura del fichero XML generado 93
A.1.1 Ejemplo fichero XML 93

B Manual de la Aplicación **97**
B.1 Instalación 97
B.2 Manual de Usuario 97
B.2.1 Extracción de características 97
B.2.2 Entrenamiento de los Algoritmos 98
B.2.3 CAD 99

C Resultados detallados de los clasificadores **105**

D Curvas ROC de los clasificadores **109**
D.1 Resultados del análisis de la curva ROC 109

E Contenido del CD **115**

Bibliografía **117**

ÍNDICE DE FIGURAS

2.1 Hallazgos más comunes que se pueden encontrar en mamografías y ecografías 6

3.1 Anatomía de la mama 12

3.2 Comparación entre los diferentes tipos de tejidos 13

3.3 Mamógrafo 15

3.4 Proyecciones mamográficas más comunes 16

3.5 Ecógrafo 17

3.6 Frontera de decisión que separa dos regiones 20

3.7 Esquema general de un sistema de reconocimiento de patrones 21

3.8 Procedimiento de selección de variables para problemas de clasificación supervisada 23

3.9 Conjunto base de características Haar 24

3.10 Suma del área rectangular sombreada en la imagen integral 25

3.11 Algunos tipos de texturas 25

3.12 Texturas diferentes con el mismo histograma 26

3.13 Ángulos para calcular la matriz de co-ocurrencia 26

3.14 Estructura de un árbol de decisión 28

3.15 Estructura del clasificador *Stump* 29

3.16 Tipos de funciones Kernel 31

3.17 Representación gráfica de SVM 32

3.18 Cascada de clasificadores 34

3.19 Estructura de cascada con metaetapas 37

3.20 Representación del cambio de umbral del SVM 38

4.1 Ciclo de Vida DSDM 40
4.2 Mapa Conceptual del Proceso Unificado de Desarrollo 45
4.3 Fases del Proceso Unificado de Desarrollo 46

5.1 Extracción de las imágenes para el entrenamiento de los clasificadores 51
5.2 Entrenamiento de los clasificadores 52
5.3 Detección de lesiones 53
5.4 Resultado de un analizador de código 53

6.1 Diagrama de casos de uso relacionado con la extracción de muestras 58
6.2 Diagrama de Casos de Uso relacionado con el Entrenamiento del Clasificador 59
6.3 Diagrama de Casos de Uso relacionado con la Detección de Lesiones . . . 60
6.4 Diagrama de Clases del Sistema 61
6.5 Diagrama de secuencia del caso de uso: *Marcar lesiones* 62
6.6 Diagrama de secuencia del caso de uso: *Entrenar clasificador* 63
6.7 Diagrama de secuencia del caso de uso: *Ejecutar algoritmo* 64
6.8 Boceto de la interfaz gráfica referente a la *Extracción de Muestras* 64
6.9 Boceto de la interfaz gráfica referente a las *Pruebas de los Algoritmos* . . . 65
6.10 Boceto de la interfaz gráfica referente al *CAD* 65
6.11 Diagrama de Gantt con los tiempos previstos 66
6.12 Diagrama de Gantt con los tiempos reales 66

7.1 Ejemplos de Herencia implementados en la aplicación 68

8.1 Ecografía con información de la paciente 78
8.2 Mamografía 79
8.3 Representación del espacio de una curva ROC 84
8.4 Representación gráfica de los tiempos computacionales 87

B.1 Inicio de la aplicación 98
B.2 Extracción de muestras 99
B.3 Extracción de muestras 100

B.4 Extracción de muestras 101
B.5 Pruebas Algoritmos 101
B.6 Pruebas Algoritmos 102
B.7 Pruebas Algoritmos 102
B.8 CAD 103
B.9 CAD 103

D.1 Curvas ROC para las ecografías utilizando características Haar. 110
D.2 Curvas ROC para las ecografías utilizando características estadísticas de textura. 111
D.3 Curvas ROC para las mamografías utilizando características Haar. 112
D.4 Curvas ROC para las mamografías utilizando características estadísticas de textura. 113

ÍNDICE DE TABLAS

5.1 Cálculo de la variable FAE 54
5.2 Material Software 55
5.3 Material *Hardware* 56

8.1 Número de imágenes para cada conjunto y tipo 81
8.2 Matriz de confusión para dos clases. VP - verdaderos positivos, VN - verdaderos negativos, FP - falsos positivos y FN - falsos negativos. 81
8.3 Matrices de confusión para ecografías con las características Haar 82
8.4 Matrices de confusión para ecografías con las características estadísticas de textura 83
8.5 Matrices de confusión para mamografías con las características Haar 84
8.6 Matrices de confusión para mamografías con las características estadísticas de textura 85
8.7 AUC de las curvas ROC de los clasificadores utilizando las características estadísticas de textura con las ecografías 85
8.8 AUC de las curvas ROC de los clasificadores utilizando las características estadísticas de textura con las mamografías 86
8.9 AUC de las curvas ROC de los clasificadores utilizando las características Haar con las ecografías 86
8.10 AUC de las curvas ROC de los clasificadores utilizando las características Haar con las mamografías 87

C.1 Resultado de los clasificadores con las características estadísticas de textura para las ecografías. 105
C.2 Resultado de los clasificadores con las características estadísticas de textura para las mamografías. 106

C.3 Resultado de los clasificadores con las características Haar para las ecografías. 106

C.4 Resultado de los clasificadores con las características Haar para las mamografías. 107

ÍNDICE DE ALGORITMOS

1 Cascada de Clasificadores 35

2 Algoritmo de entrenamiento para construir una clasificador en cascada utilizando *AdaBoost* y SVM 36

CAPÍTULO 1

INTRODUCCIÓN

1.1 INTRODUCCIÓN

El cáncer de mama es la primera causa de muerte por cáncer entre las mujeres en todo el mundo. Se estima que el riesgo de padecer cáncer de mama es de 1 de cada 8 mujeres [5]. No obstante, cada año disponemos de mayor información para diagnosticar precozmente y tratar el cáncer de mama, lo que ha permitido que la supervivencia global a los 5 años del diagnóstico de este tumor sea del 82,8 % en nuestro país, por encima de la media europea y similar a los países con las mejores cifras de supervivencia [6].

El cáncer de mama es una de las pocas enfermedades cancerosas que tarda en presentar síntomas y que puede ser detectada antes de que lo haga. La probabilidad de curación del cáncer de mama desciende a medida que el tumor crece con el paso del tiempo. La mamografía permite detectar lesiones en la mama hasta dos años antes de que sean palpables y cuando aún no han invadido en profundidad ni se han diseminado a los ganglios y a otros órganos [7]. Durante los últimos años se ha visto que la detección precoz de este tipo de cáncer por medio del estudio y del uso de mamografías (*screening*[1]) junto con pruebas complementarias como la ecografía o la resonancia magnética disminuía la mortalidad un 20-30 %. No es de extrañar, por tanto, que dentro del campo de estudio de la Visión por Computador se estén realizando grandes esfuerzos en desarrollar detectores de lesiones para este tipo de cáncer.

1.2 DIAGNÓSTICO DEL CÁNCER DE MAMA

La mamografía es, hoy en día, el método más efectivo para la detección del cáncer de mama antes de que este sea palpable. De hecho, es la única prueba diagnóstica que cuenta con la aprobación de la *Food and Drug Administration* (FDA) para la tarea de *screening*. Se

[1] El *screening* es el método de recogida de un gran número de mamografías con el objetivo de detectar una enfermedad sin que se tengan indicios de esta. Su objetivo es la detección del cáncer de mama de manera temprana para poder actuar antes y reducir sus efectos.

trata de un tipo específico de imagen para examinar el pecho que utiliza bajas dosis de rayos X ofreciendo imágenes de alta calidad (con resoluciones de alrededor de 3300x4000 *pixels* y una profundidad de color de 2^{12} niveles de gris).

La ecografía mamaria se utiliza para estudiar las mamas bilateralmente en toda su extensión. A través de ella se pueden analizar nódulos o tumores de forma diferencial palpados en la exploración clínica o visualizados a través de una mamografía. Con la mamografía se manifiesta si existe alguna lesión de tipo benigna o maligna. En caso de que la lesión sea benigna, la ecografía permite analizarla para saber si es sólido (fibroadenoma) o líquido (quiste), dato que no se puede hallar a través de la mamografía.

Existen dos tipos de exámenes que se realizan utilizando las mamografías: la lectura y el diagnóstico. La lectura se utiliza con pacientes que no presentan síntomas de padecer ningún tipo de lesión en la mama, mientras que el diagnóstico tiene como objetivo examinar a un paciente que ha presentado anomalías con anterioridad. Un método muy recomendado es la doble lectura de una mamografía por dos radiólogos para evitar casos de cáncer no detectados, aunque eso implicaría una carga de trabajo y unos costes elevados.

Con la ayuda de sistemas de CAD (del inglés *Computer Aided Detection/Diagnosis*) ya no serían necesarios dos radiólogos para examinar las mamografías, sino uno. El objetivo de estos sistemas es servir de referente y aportar una segunda opinión para los profesionales dedicados a la búsqueda de lesiones. Para la construcción de este tipo de sistemas se pueden aplicar diferentes métodos de clasificación y combinaciones de estos. Uno de los más empleados en los últimos años ha sido el método de clasificación propuesto por Viola and Jones [45], habiendo sido ampliamente utilizado para la detección en multitud de problemas.

Este método de clasificación en cascada es muy adecuado para la construcción de este tipo de sistemas por su alta precisión, por su diseño rápido, eficiente y simple y porque en cada etapa no se necesitan todas las características sino que se seleccionan las más importantes a través de la combinación de clasificadores débiles mediante un algoritmo de *boosting* [25].

1.3 CLASIFICADORES

Decimos que un clasificador es una función F(x) que asigna una clase a una muestra. Las consideraciones a tener en cuenta a la hora de crear un clasificador son: la capacidad de acierto, rapidez, claridad, tiempo de aprendizaje y falsos positivos. Los clasificadores pueden ser más o menos complejos: los *Weak Classifiers* son los más sencillos y los *Strong Classifiers* los más complejos. Esto se verá con más detalle en el apartado 3.7.

Los *Weak Classifiers* o clasificadores "débiles" deben ser moderadamente eficaces, es decir, que además de tratarse de un clasificador sencillo y rápido de construir y ejecutar debe

obtener una capacidad de acierto algo superior a un clasificador aleatorio. Un clasificador es considerado como un *Strong Classifier* cuando su función de decisión es compleja, necesitando tiempo y otros recursos para su entrenamiento y ejecución o cuando está formado a partir de la combinación de un conjunto finito de clasificadores débiles, ponderados en función del peso asociado a cada uno, dando como resultado un clasificador significativamente más efectivo y complejo.

El uso de clasificadores débiles reduce eficazmente la complejidad y el tiempo de cálculo, pero presenta problemas como una baja capacidad de acierto y una alta tasa de falsos positivos. Para mejorar estos inconvenientes, Viola y Jones propusieron un clasificador basado en cascada de varios clasificadores *AdaBoost*.

La creación del clasificador se basa en un sistema de reconocimiento de patrones que funciona en dos modos distintos: entrenamiento y reconocimiento. Durante el entrenamiento se diseña el extractor de características que representa los patrones de entrada y se entrena el clasificador tomando un conjunto de datos de entrenamiento. Una vez que el clasificador está entrenado, se toma como dato de entrada el vector de características de un patrón desconocido que será asignado a una de las clases o categorías (Sección 3.5).

El presente trabajo está motivado por la necesidad de desarrollar y comparar diferentes métodos de detección con el objetivo de integrar en los sistemas CAD las que mejores resultados ofrezcan.

1.4 ESTRUCTURA DEL DOCUMENTO

Este trabajo se encuentra dividido en 7 capítulos y 3 anexos como se muestra a continuación:

- **Capítulo 1. Introducción.** Realiza una introducción al tema y muestra la estructura del documento.

- **Capítulo 2. Objetivos.** Detalla cuáles son los objetivos generales y específicos y los condicionantes que han influido en el desarrollo de este trabajo.

- **Capítulo 3. Antecedentes.** Se presentan los fundamentos y bases teóricas. Se detallan las características de la mama y los distintos tipos de lesiones que se pueden hallar en las mismas. Asimismo, se describe la imagen digital, los patrones, cuáles son las técnicas de reconocimiento de objetos, mostrando especial interés en los clasificadores en cascada, y cómo tratar las características o descriptores de una imagen, tanto las características Haar como las estadísticas de textura.

- **Capítulo 4. Método de trabajo.** Describe la metodología de desarrollo *software* utilizada.
- **Capítulo 5. Evolución y costes.** Se describen de manera detallada los resultados obtenidos en cada una de las iteraciones que se han llevado a cabo. Además, en este capítulo se muestran los costes para llevar a cabo el desarrollo de la aplicación.
- **Capítulo 6. Artefactos generados.** Se presentan los artefactos obtenidos según la metodología descrita en el Capítulo 4.
- **Capítulo 7. Desarrollo de la aplicación.** Este capítulo describe las diferentes decisiones de diseño tomadas durante el proceso de desarrollo.
- **Capítulo 8. Resultados.** Se realiza una comparativa entre las diferentes variaciones del clasificador en cascada y cómo afectan estas en la detección de lesiones en ecografías y mamografías.
- **Capítulo 9. Conclusiones y propuestas futuras.** En él se exponen las diferentes conclusiones obtenidas a partir del diseño, de la construcción y del uso de la aplicación además de una serie de propuestas para futuras ampliaciones.
- **Apéndice A.** Se muestra la estructura del fichero XML generado durante el entrenamiento del clasificador.
- **Apéndice B.** Instalación y manual de la aplicación.
- **Apéndice C.** Curvas ROC de los clasificadores.
- **Apéndice C.** Contenido del CD.

CAPÍTULO 2

OBJETIVOS

2.1 OBJETIVOS Y ENFOQUE

El objetivo del trabajo es la implementación y comparación de métodos computacionales de clasificación automática orientados a la detección de lesiones de mama. Para ello, se debe realizar un análisis de imágenes diagnósticas de distintas modalidades. Las dos modalidades más utilizadas para la detección y el diagnóstico de lesiones mamarias son la mamografía seguida de la ecografía. Las lesiones típicas a buscar en las mamografías se clasifican en: microcalcificaciones, masas y distorsiones arquitecturales, pudiendo ser estas a su vez benignas o malignas y suponer, en algunos casos, un indicador de la existencia de una malignidad. Con respecto a las lesiones en ecografías, las lesiones a buscar son las masas ya que las pequeñas microcalcificaciones son imperceptibles debido al ruido de la propia imagen.

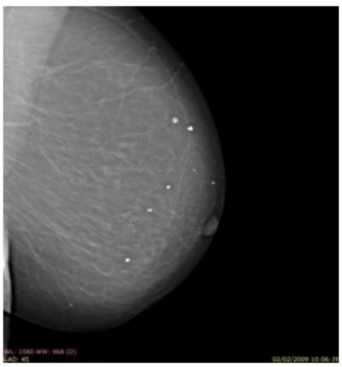
En la Fig. 2.1 se pueden observar los distintos tipos de lesiones sobre los que se centra este trabajo.

Por tanto, el principal objetivo de este Trabajo Fin de Grado es generar una aplicación para detectar de forma rápida regiones de interés en mamografías y ecografías para el diagnóstico del cáncer de mama, además de la integración del proceso completo de detección en una única herramienta.

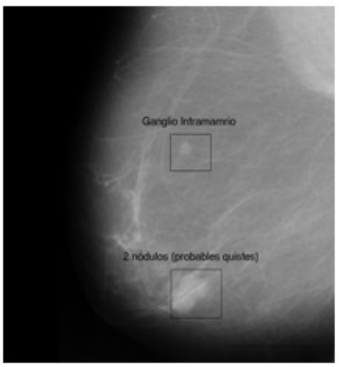
Se deben analizar, implementar y comparar distintas técnicas de inteligencia artificial para la detección de las regiones de interés (lesiones en la mama tales como nódulos, microcalcificaciones y distorsiones) [3]. Los resultados deben servir para la obtención de una detección fiable dando lugar a una herramienta objetiva y útil para los médicos.

La aplicación podrá ser utilizada a su vez para otro tipo de tareas de detección como la detección de caras, detección de personas o la detección de otros tipos de lesiones cancerosas.

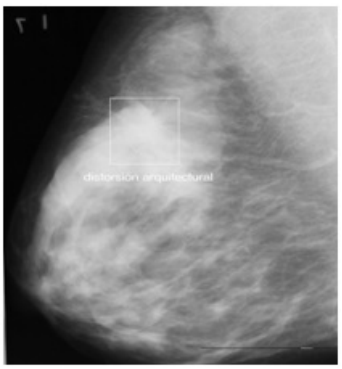
Todas las partes que conlleva el proceso de detección podrán ser integradas en una sola herramienta. Se podrán ir añadiendo de forma sencilla más variantes de los distintos algoritmos de clasificación, etapas de procesado previas a la detección. Además, la aplicación



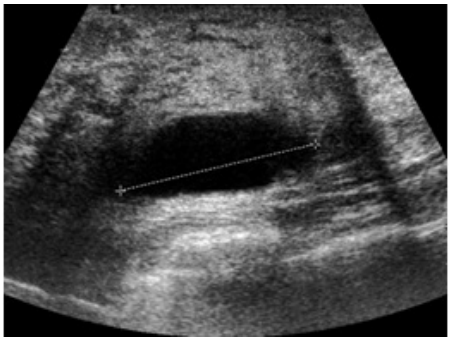
(a) Mamografía con varias microcalcificaciones



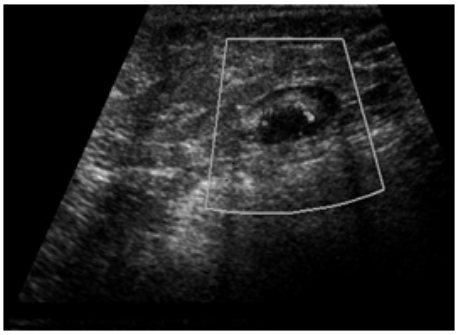
(b) Mamografía con dos masas



(c) Mamografía con una distorsión arquitectural



(d) Ecografía con lesión benigna



(e) Ecografía con lesión maligna

Figura 2.1: Hallazgos más comunes que se pueden encontrar en mamografías y ecografías

debe proveer de un instrumento para la creación de conjuntos de datos de entrenamiento y test para los algoritmos.

2.1.1 Objetivos específicos

La aplicación debe estar dividida en tres partes para la integración de todo el proceso de detección de lesiones. En la primera parte, se debe poder extraer una serie de zonas de imágenes para crear los conjuntos de entrenamiento. La segunda, debe ofrecer lo necesario para entrenar un cierto clasificador en función de una serie de parámetros introducidos por el usuario a través de la interfaz. Y, por último, la tercera, debe dar soporte para el análisis de una imagen con un clasificador previamente entrenado con el objetivo de detectar alguna anomalía.

El objetivo principal de esta aplicación es, por tanto, facilitar el proceso de detección de lesiones, desde la preparación de los datos hasta la detección. Entre las funcionalidades desarrolladas están:

- Extraer zonas de interés de un conjunto de imágenes para obtener el conjunto de entrenamiento de los clasificadores.

- Mostrar información acerca de las zonas seleccionadas como coordenadas, anchura, altura y signo (positivo o negativo en función de si la zona contiene o no un hallazgo).
- Entrenar el clasificador en función de una serie de parámetros introducidos por el usuario mediante la interfaz.
- Ver el estado de todo el proceso de entrenamiento.
- Generar un fichero con el clasificador resultante de este entrenamiento.
- Analizar una imagen en función de un clasificador elegido por el usuario y mostrar si esa imagen contiene o no algún tipo de anomalía.

2.2 CONDICIONANTES

En esta sección se describen las limitaciones y condicionantes a considerar para la resolución del problema, todo lo relacionado con los medios software y hardware que han influido en el desarrollo de este trabajo.

2.2.1 Medios software

Lenguaje de construcción

El lenguaje que se ha utilizado para el desarrollo de esta aplicación es C++.

Los principales motivos de esta elección son los siguientes:

- Es más rápido y el trabajo con imágenes requiere gran capacidad de cálculo.
- Posibilita la obtención de un desarrollo bajo el paradigma orientado a objetos.
- Posibilita la utilización de la librería de Visión por Computador OpenCV.

Además, en C++ existen dos tipos de ficheros principalmente: los de cabecera (*.h* o *.hpp*) y los ficheros fuentes (*.cpp*). La diferencia entre ellos es puramente conceptual, el compilador no distingue entre uno u otro, de eso se encarga el programador.

Las ventajas de dividir el código fuente son:

- *Compilación más eficiente:* teniendo todo el código en un sólo fichero, al realizar cualquier modificación y volver a compilar habría que recompilar todo el código, sin embargo, si la implementación está dividida en varios ficheros, al realizar una modificación en cualquiera de ellos y compilar, sólo sería necesario recompilar la clase que se ha modificado y no todo el código.

- *Organización:* Mejora la estructura del sistema de archivos, y reduce los tiempos de búsqueda al estar el código más ordenado.

- *Facilitar la reutilización:* Al estar más dividido el código, se mejora la reutilización de partes del mismo.

- *Compartir código en varios proyectos:* Si varios proyectos necesitan utilizar el mismo código, es posible incluir el mismo en ambos proyectos y si se modifica este código, se modifica automáticamente en ambos proyectos.

- *Dividir tareas entre programadores:* Si se desea que varios programadores trabajen a la vez sobre un mismo proyecto, se puede asignar un fichero a cada uno y luego juntarlos.

Librería OpenCV

OpenCV es una librería de Visión por Computador publicada bajo la licencia de dominio BSD1 (*Berkeley Software Distribution*) [10], permitiendo así el libre desarrollo de aplicaciones comerciales o de investigación. En la actualidad está considerada como la principal librería de funciones para Visión por Computador en tiempo real, y está en constante desarrollo.

La librería está desarrollada en lenguaje C y C++ y puede ser utilizada en diferentes entornos operativos como Windows, MacOS y Linux. Tiene como un objetivo prioritario maximizar el rendimiento, optimizando el consumo de los recursos computacionales, algo que resulta crítico en aplicaciones de Visión por Computador.

Para la implementación de los algoritmos de clasificación se optó por codificar los algoritmos de nuevo en lugar de realizar una modificación del código fuente de OpenCV porque su modificación suponía rehacer tanto código como la codificación de los algoritmos desde el principio, ya que la cascada de Viola&Jones sólo soporta etapas con su tipo de algoritmos de *boosting* y la ampliación de etapas con otro de tipo de clasificador era costoso.

En este trabajo, OpenCV se utiliza únicamente para el manejo de imágenes, la persistencia y la creación de los SVM (*Support Vector Machines*).

Microsoft Visual Studio 2010

El entorno de desarrollo integrado utilizado ha sido *Microsoft Visual Studio* versión 2010 de *Microsoft*. Esta herramienta permite crear multitud de aplicaciones para diversas plataformas. Además, las herramientas de administración del ciclo de vida de Visual Studio permiten entregar software en ciclos más cortos y con mayor agilidad, a la vez que se aumenta la calidad del software.

La interfaz gráfica ha sido construida utilizando la tecnología Windows Forms. Windows Forms es un subconjunto de la clase .NET Framework Class Library que permite el desarrollo de aplicaciones de escritorio. Incluye clases base, interfaces, enumeraciones y controles gráficos diversos.

Control de versiones

Para el manejo del control de versiones se ha utilizado la herramienta *Subversion* a través del cliente *TortoiseSVN* que utiliza una base de datos central que contiene todos los archivos y sus distintas versiones. Esta base de datos se denomina *repositorio*. El repositorio se encuentra en un servidor de archivos ejecutando el servidor de *Subversion*. Cada desarrollador tiene una copia propia de trabajo en su ordenador local. Un usuario puede descargarse la última versión del repositorio y trabajar sobre ella sin perjudicar a nadie y cuando decida aplicar los cambios que ha realizado puede actualizar las modificaciones que ha realizado en el repositorio.

Es una herramienta muy útil para trabajos en grupo y/o, como en este caso, para manejar las distintas versiones que se van creando durante el desarrollo de un proyecto.

LaTeXy *TeXnicCenter*

Para la realización de la parte documental de este trabajo se ha empleado el sistema de composición de textos LaTeX.

LaTeXestá constituido por un gran conjunto de macros de TeX con el objetivo de facilitar el uso del lenguaje de composición tipográfica. Es una herramienta de gran utilidad para la producción de artículos académicos, tesis y libros técnicos, dada la alta calidad tipográfica de los documentos realizados con este sistema.

El desarrollo del documento se prepara iterando en tres fases: edición, procesado y previsualización. La fase de edición consume muy pocos recursos y se puede llevar a cabo con cualquier editor ASCII aunque existen entornos especializados.

El proceso es universal, es decir, es igual en todos los sistemas operativos.

Doxygen

Para la generación de documentación a partir de los comentarios del código fuente se ha utilizado la herramienta *Doxygen*. El nombre de *Doxygen* es un acrónimo de dox(document) gen(generator). Este generador automático soporta los lenguajes C, C++, Java, Objective-C, Python, IDL, VHDL y algo para PHP, C# y D. La documentación generada, tanto en HTML (*HyperText Markup Language*) como en PDF (*Portable Document Format*) se encuentra en la documentación electrónica del trabajo (Ver Anexo E).

Sistema Operativo

El sistema operativo que se ha utilizado es *Windows* 7. Se optó por este sistema por ser el más utilizado en PCs de escritorio ya que no había ninguna restricción en este aspecto.

2.2.2 Medios hardware

Los medios hardware utilizados para la realización de este proyecto han sido los siguientes:

- Portátil con procesador Intel(R) Core 2 Duo 2.00GHz, 2.00 GB de RAM.
- Ordenador de sobremesa con procesador Intel(R) Xeon(R) CPU ES-2620 2.00GHz, 16.00 GB de RAM.

CAPÍTULO 3

ESTADO DEL ARTE

En este capítulo se presentan los fundamentos y bases teóricas del presente Trabajo Fin de Grado. Se detallan las características de la mama y los distintos tipos de lesiones que se pueden hallar en las mismas.

Asimismo se describe la imagen digital, las texturas, los patrones, cuáles son las técnicas de reconocimiento de objetos y cómo tratar las características o descriptores de una imagen tanto las características Haar como las estadísticas de textura.

3.1 ANATOMÍA Y EXPLORACIÓN DE LA MAMA

En la mujer, se define la mama como la estructura anatómica donde se encuentra la glándula mamaria. La glándula mamaria es un parénquima glandular compuesto de alveolos y conductos del estroma de soporte. Cada célula alveolar actúa como una unidad de secreción, produciendo leche completa, sintetizando y transportando desde el plasma sanguíneo todos los componentes de la leche: las proteínas, grasas, hidratos de carbono, sales minerales, anticuerpos y agua.

Las glándulas mamarias se encuentran presentes en ambos sexos. En el hombre permanecen rudimentarias durante toda la vida, sin embargo en la mujer se mantienen poco desarrolladas hasta antes de la pubertad, momento en el que comienza el proceso de maduración. Su máximo desarrollo se produce durante el embarazo y especialmente en la etapa posterior al parto, la lactancia.

3.1.1 Anatomía de la mama

Las mamas están situadas en la parte anterior del tórax y pueden expandirse de forma variable por su cara lateral. Su forma depende de características personales, genéticas y en la misma mujer de acuerdo a la edad y maternidad. La masa de la mama está constituida en gran

medida por tejido glandular y adiposo. Durante el período de embarazo y lactancia el tamaño de la mama aumenta por causa del crecimiento del tejido glandular.

En la Figura 3.1 se puede ver que la base de la glándula mamaria se extiende desde la tercera costilla hasta la séptima, desde el borde externo del esternón hasta la línea axilar media.

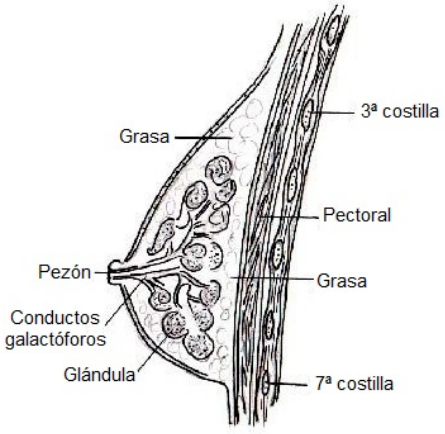


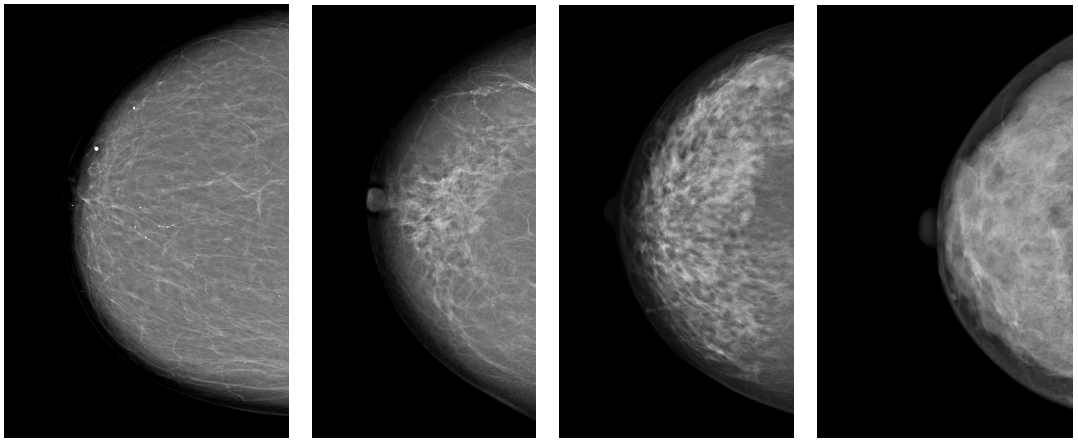
Figura 3.1: Anatomía de la mama

3.1.2 Tipos de Tejido Mamario

El *American College of Radiology* (ACR) *Breast Imaging Reporting and Data System* especifica que la densidad mamaria se clasifica en cuatro tipos dependiendo de la cantidad de tejido mamario (parénquima) presente (Fig. 3.2) [12]:

- Predominantemente adiposo: El tejido mamario presente es menor del 25 %, siendo el resto grasa.
- Fibroglandular: Presenta entre el 25 % y el 50 % de tejido mamario.
- Heterogéneamente denso: Entre el 50 % y el 75 % de la mama es tejido.
- Extremadamente denso: La mama contiene más del 75 % de tejido mamario.

El tipo de tejido se considera un factor determinante para la detección de lesiones. Puesto que el tejido mamario es más denso que la grasa, los rayos X no pasan con facilidad a través de él dando lugar a zonas más blancas y opacas fáciles de confundir con las lesiones. Por tanto, el nivel de dificultad para la detección es menor en los tejidos predominantemente adiposos y mayor en los extremadamente densos [44].



(a) Predominantemente adiposo (b) Fibroglandular (c) Heterogéneamente denso (d) Extremadamente denso

Figura 3.2: Comparación entre los diferentes tipos de tejido

3.1.3 Lesiones del Tejido Mamario

La interpretación automática de lesiones en la mama sigue siendo un problema difícil y sin una solución generalizada. La dificultad del problema reside en:

- la complejidad de las imágenes y de las lesiones mamográficas a estudiar,
- los componentes perceptibles de la mama en una mamografía en el tipo de tejido mamario y
- la variación significativa de las glándulas mamarias y las características de las lesiones entre los distintos pacientes.

Es relativamente sencillo identificar masas mamográficas sobre tejido adiposo, pero es extremadamente difícil hacerlo sobre tejido denso, debido a las pequeñas diferencias de densidad. La mayor parte de los errores que se producen en la detección de lesiones en mamografías se debe a la alta densidad existente en la mamografía analizada, al uso de una técnica poco apropiada, o a un error u omisión en la lectura de la misma.

Los distintos tipos de lesiones que podemos encontrar dentro de las mamografías son [11]:

- Masas o nódulos. Se denomina nódulo a la lesión de pequeño tamaño, concreta y que se puede individualizar de cuanto le rodea, mientras que una masa es una lesión que tiene un gran volumen. Las masas mamográficas también se pueden confundir con tejido denso debido a que también son radio opacas. En cuanto a las ecografías, las masas suelen aparecer como una zona hipoecogénica (en relación con la ecogenicidad[1] de la grasa) lo que se traduce en zonas más oscuras.

[1] Se denomina ecogenicidad a la capacidad de los tejidos de reflejar (hacer eco de) ondas de ultrasonido.

- Microcalcificaciones. Son minúsculos depósitos de calcio que aparecen en la mamografía como puntos blancos brillantes. Pueden aparecer tanto aislados como en grupo. Pequeños grupos de microcalcificaciones irregulares dentro de una mamografía son un signo temprano de un posible cáncer de mama maligno. Su diminuto tamaño propicia fallos de localización para el diagnóstico por parte de los radiólogos. Son las lesiones más frecuentes encontradas en mamografías, seguidas de las masas y las distorsiones arquitecturales. En el caso de las ecografías, las microcalcificaciones son prácticamente indetectables.

- Distorsiones. Es una alteración del tejido glandular con la producción de radiaciones finas y espiculadas. Suelen ser indicador de alguna malignidad subyacente.

- Casos especiales. Son lesiones que presentan un aspecto ecográfico específico: microquistes agrupados, quiste complicado, lesión cutánea, ganglio intramamario, cuerpo extraño o adenopatía axilar [3].

3.1.4 Técnicas de Exploración

Las técnicas utilizadas para el diagnóstico del cáncer de mama van desde la autoexploración hasta los sistemas CADs [41] [46].

1. Autoexploración y examen clínico. A menos que el paciente se encuentre en alto riesgo de padecer cáncer, debido a antecedentes familiares u otros factores que puedan influir, la ACS (*American Cancer Society*) recomienda realizarse un examen clínico de mama cada tres años hasta los 40 años y, posteriormente, un examen clínico por año. El médico examinará las mamas para detectar bultos o cualquier tipo de cambio que pueda ser sospechoso. También analizará si se ha producido un crecimiento de los ganglios linfáticos de debajo de la axila.

2. Mamografía. Es un procedimiento radiológico mediante rayos X que utiliza una serie de radiografías del tejido mamario. Hoy en día es considerada como la mejor técnica de obtención de imágenes antes de que el paciente o el médico las puedan palpar.

   En las mamografías se utiliza información visual que permite conocer la estructura interna de los senos para estudiar los tejidos y buscar lesiones en base a la apariencia, consistencia, tamaño y región en donde se localicen.

   Para la obtención de las distintas proyecciones radiográficas de una mama, los profesionales sanitarios utilizan un mamógrafo. Se trata de un equipo de rayos X que consta de un generador de corriente y una columna con un brazo giratorio, con un tubo de rayos X, un compresor y un receptor de imagen. Este equipo debe ser capaz de producir

una imagen que identifique las estructuras de las glándulas mamarias, tales como vasos sanguíneos, tejido glandular o grasa, entre otras, para poder visualizar y detectar fases tempranas de lesiones (Fig. 3.3).

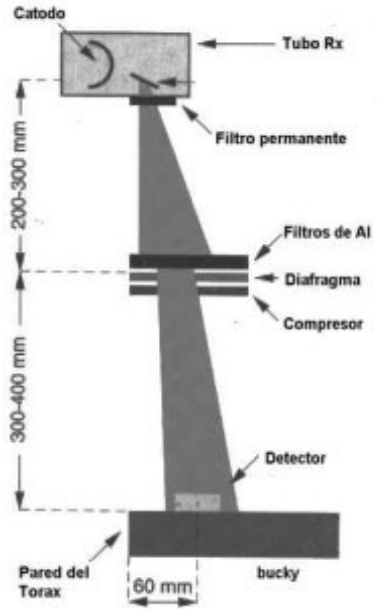


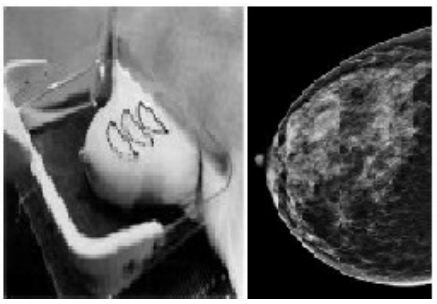
Figura 3.3: Mamógrafo

Un factor importante a la hora de hacer la realización de una mamografía es la correcta colocación de la paciente. Toda la mama debe estar dentro del mamógrafo para evitar que se queden fuera posibles zonas con lesiones. También es necesario ejercer la presión adecuada para que no aparezcan zonas borrosas en la imagen que dificulten la localización de lesiones.

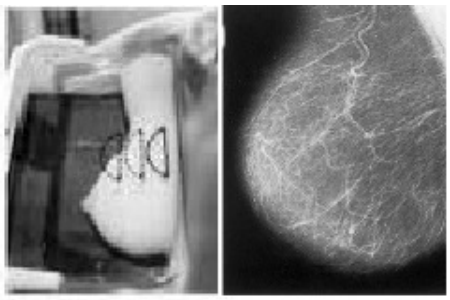
Las principales proyecciones que se puede hacer sobre la mamografía son la CC (Cráneo-Caudal) y la MLO (Medio-Lateral Oblicua). Es posible que sean necesarias proyecciones adicionales y particulares como la lateral (ML), las magnificadas, las axilares o los conos de compresión localizada (Fig. 3.4)

Sin embargo, no son fáciles de detectar debido a las pequeñas diferencias de densidad en la imagen, como se ha comentado anteriormente, hecho que provoca que el profesional encargado de detectar esas lesiones pueda pasar por alto algunas de ellas.

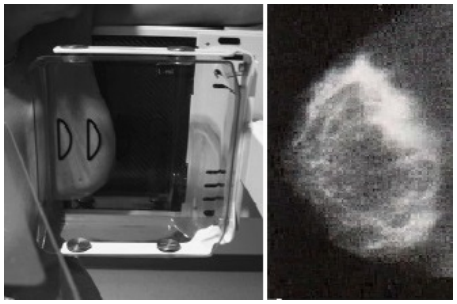
3. Ecografía. Se considera un método secundario para detección del cáncer de mama, ya que evalúa anomalías observadas previamente en mamografías o durante un examen clínico, sin embargo, se trata una herramienta de detección segura puesto que no conlleva un procedimiento invasivo. Puede determinar si una región de interés es un quiste o un tejido sólido ya que hace más evidente la composición y la forma de las masas. En la práctica no se utiliza de forma prioritaria debido a que tiene un alto índice



(a) Proyección Cráneo-Caudal (CC)



(b) Proyección Medio-Lateral Oblicua (MLO)



(c) Proyección Medio-Lateral (ML)

Figura 3.4: Proyecciones mamográficas más comunes

de falsos positivos detectados, es decir, encuentra regiones de interés que realmente no lo son, y a la escasa visualización de las microcalcificaciones.

La herramienta para la producción de ecografías es el ecógrafo. Este a través de los cristales de su transductor genera una onda de ultrasonidos que viaja por el interior de los tejidos sobre los que incide. Esta onda se atenúa como consecuencia de la absorción y se refleja y se refracta debido a la diferencia de impedancias acústicas en la etapa que se denomina interfase y dependiendo de la densidad que presentan los diferentes tejidos que componen los órganos.

Durante la interfase, se producen ondas debido a la reflexión. Estas ondas, o ecos, se recogen por otros transductores que convierten las señales acústicas en señales eléctricas para su procesamiento y composición de la imagen (Fig. 3.5).

4. Resonancia magnética nuclear (RM). Esta técnica utiliza un imán y ondas de radio para obtener imágenes del interior de la mama. La ventaja que aporta con respecto a las demás técnicas utilizadas es que puede detectar tumores demasiado pequeños para ser localizados en los exámenes físicos o difíciles de ver en las mamografías convencionales. La resonancia magnética no sustituye a las mamografías, sino que actúa como un método complementario de las mismas.

   No es recomendable usar este método habitualmente en pacientes con riesgo de padecer lesiones de mama ya que tiene un alto índice de falsos positivos, que puede conducir

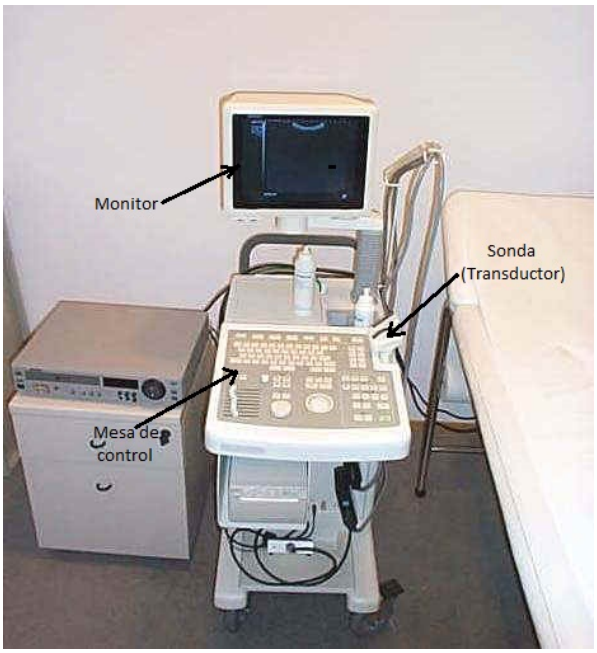


Figura 3.5: Ecógrafo

a diversos errores. Además es una prueba costosa, no tiene disposición inmediata y requiere la interpretación de un radiólogo con experiencia.

5. Sistemas CAD. En los métodos tradicionales, un radiólogo tiene que analizar las radiografías, por tanto, la tasa de aciertos va a depender de la destreza y de la experiencia que tenga dicho radiólogo. En la detección y el diagnóstico asistido por computador se escanea la mamografía después de que un radiólogo la haya revisado y se identifican áreas altamente sospechosas, hecho que permite que el profesional se centre en puntos específicos que pudo pasar por alto. Todo esto propicia un ahorro de tiempo y dinero además de una mayor eficiencia en la detección.

   En la actualidad, existen tres sistemas comerciales tipo CAD que han sido admitidos por la FDA (*Food and Drug Administration, USA*) para asistir en la detección de irregularidades en mamografías. Estos sistemas son [40]:

   - *R2 Technology's Image Checker:* Primer sistema aprobado por la FDA. Usado para la detección de masas y calcificaciones.
   - *Mamoreader:* Apto para la detección de masas, calcificaciones, distorsiones arquitecturales y asimetrías. Fue desarrollado por *Intelligent System's Software, Inc*.
   - *Second look:* Orientado a la detección de masas y calcificaciones. Fue desarrollado por *CADx Medical Systems*.

   Por otro lado, los sistemas CADx (*Computer-Aided Diagnosis*) son aplicaciones utilizadas para la asistencia en el diagnóstico de lesiones. Se considera que los sistemas CADx

desarrollan tareas mucho más complejas que los CAD's pues, además de localizar lesiones, deben especificarlas y determinar su grado de malignidad o benignidad. Por causa de esta complejidad, aún no existen sistemas CADx comerciales, sólo existen artículos desarrollados en el área de investigación.

3.2 IMAGEN DIGITAL

Una de las cosas más importantes en las que se centra el presente trabajo es el tratamiento de las imágenes digitales que son utilizadas como entrada del sistema CAD. Estas imágenes pueden haber sido obtenidas directamente en formato digital mediante equipos que soporten esta funcionalidad o a través del escaneado de imágenes radiológicas impresas.

La imagen digital es almacenada mediante una matriz. Esta matriz es un conjunto de valores dispuestos en filas y columnas donde cada valor corresponde a una situación específica en la imagen, es decir, representa el brillo o intensidad en esa situación. Cada elemento de esta matriz es un valor que define el brillo del píxel. Este valor es relativo y determina el contraste de la imagen.

El rango de valores sobre los que un sistema puede dar respuesta se llama *rango de escala de grises* o *rango dinámico*. Se describe como el número de niveles de gris que se puede representar en él. El número máximo de niveles de gris que puede representar un sistema viene dado por el rango numérico de cada píxel o profundidad de bits.

Las mamografías utilizadas en este trabajo han sido obtenida mediante un proceso de digitalización obteniendo imágenes de 2^8 niveles de gris.

Las imágenes digitales a color se rigen por los mismos conceptos de muestreo, cuantificación y resolución que las imágenes en escala de grises. Sin embargo, en lugar de utilizar un único valor de intensidad para el nivel de gris, se emplean tres componentes para cuantificar los píxeles, que tendrán distinto significado en función del modelo o espacio de color utilizado. El formato RGB más común utiliza 256 niveles de intensidad para cada uno de los componentes.

Las ecografías siguen el formato RGB de 8 bits por canal y píxel, es decir, $(8_r, 8_g, 8_b)$. En este caso, el color se utiliza únicamente para las anotaciones por lo que se obtiene como imagen final una imagen en escala de grises de 8 bits haciendo la media de los canales y recortando la zona donde aparece la mama.

3.3 REGIONES DE INTERÉS

Llamamos Región de Interés - ROI (del inglés *Region of Interest*) a una región de una mamografía o ecografía que se considera esencial extraer porque contiene información relevante. Para obtener las características de una ROI se utiliza una ventana que la contenga.

A partir de la información extraída de esa ventana se calculan las características de la ROI.

Un tumor, analizado al nivel de detalle de una ROI, presenta un tamaño bastante homogéneo, con características similares a las que pueden encontrarse en un tejido sano. Debido a esto, la precisión del clasificador disminuye si el tamaño de la ventana es mucho menor que el tamaño del tumor. Por el contrario, si la ventana de características es ligeramente más grande que el tamaño del tumor, el clasificador va a ser más eficaz puesto que es posible recoger características representativas del tumor que lo diferencian del tejido sano. [42]

3.4 RECONOCIMIENTO DE PATRONES

Según [36] el reconocimiento de patrones es la ciencia que se encarga de agrupar, en una misma región del espacio, conjuntos de patrones (muestras) que pertenecen a la misma clase.

Con el reconocimiento se obtiene como resultado la asociación de una serie de elementos u objetos a clases o prototipos. Esta asociación se realiza en base a las características o propiedades de los objetos. A partir de las características extraídas de las regiones de una imagen se pueden reconocer los objetos presentes en ellas.

En este trabajo, se aplica el reconocimiento de patrones para detectar distintos tipos de lesiones que pueden darse en mamografías y ecografías.

No existen reglas exactas para descubrir el mejor conjunto de características para agrupar y distinguir las muestras por lo que deben ser seleccionadas con detenimiento. A este hecho se le suma que las clases pueden no estar correctamente definidas y que la información sobre los objetos puede ser incompleta.

El algoritmo o la función que se encarga de asignar las muestras a su correspondiente clase se conoce con el nombre de clasificador.

En este trabajo, para realizar el entrenamiento del clasificador, se diferencia entre las regiones de la imagen que contienen lesión y las regiones de la imagen de tejido mamario sin lesión.

3.4.1 Fundamento Matemático

Reconocimiento mediante funciones discriminantes

Se trata de una técnica estadística multivariante que se encarga de analizar diferencias significativas entre grupos de objetos respecto a un conjunto de variables medidas sobre los mismos para, en el caso de que existan, explicar en qué sentido se dan y proporcionar procedimientos de clasificación sistemática de nuevas observaciones de origen desconocido en uno de los grupos analizados. [4]

Elementos

- *Función discriminante, f(x):*
  Mide la relevancia de la clase c para el patrón x.
- *Región de decisión, R:*
  Es el conjunto de todos los puntos del espacio que el clasificador asigna a la clase c.
- *Frontera de decisión:*
  Es la franja que separa las regiones de decisión (Fig. 3.6).

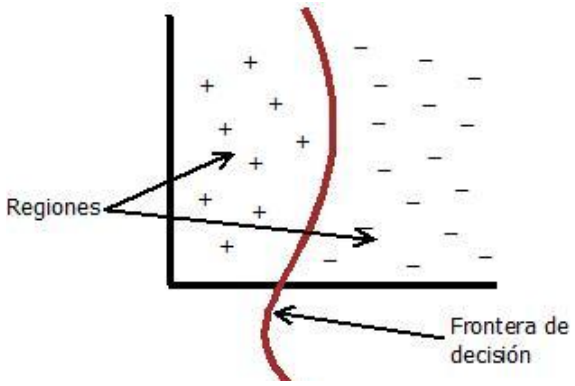


Figura 3.6: Frontera de decisión que separa dos regiones.

- *Clasificador:*
  Seleccionan la clase en base a la función discriminante *f(x)*.
- *Conjunto de entrenamiento.*
  Conjunto de pares patrón, etiqueta.
  x_k, c_k, $k = 1, \ldots, K$
- *Algoritmo de entrenamiento.*
  Conjunto de reglas de ajuste de los parámetros de las funciones discriminantes $f_i(x, c_i)$.
- *Pruebas.*

- Conjunto de patrones etiquetados NO utilizados durante el entrenamiento.
- Sirven para evaluar el rendimiento del clasificador.

Es deseable obtener clasificadores con alto grado de generalización, que es la capacidad para clasificar correctamente patrones no utilizados durante el entrenamiento.

3.5 SISTEMA BÁSICO DE RECONOCIMIENTO

3.5.1 Descripción de los sistemas de reconocimiento de patrones

El esquema de un sistema de reconocimiento de patrones consta de varias etapas relacionadas entre sí, es decir, los resultados de una etapa pueden ocasionar cambios en los parámetros de etapas posteriores (Fig. 3.7).

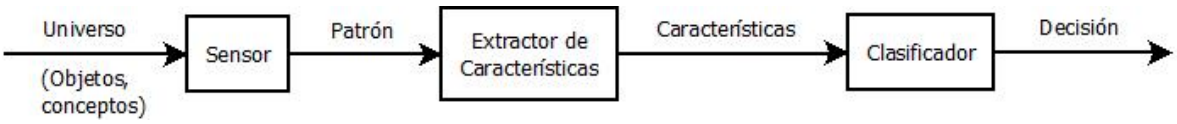


Figura 3.7: Esquema general de un sistema de reconocimiento de patrones

1. La finalidad que tiene el *Sensor* es proporcionar una representación factible de los elementos del universo que son objeto de clasificación. Determina límites en rendimiento del sistema. Ha de ser capaz de transformar variables físicas o químicas en magnitudes eléctricas.

2. El *Extractor de Características* transforma la información observada en valores numéricos o simbólicos, utiliza al patrón de representación para extraer la información discriminatoria y elimina la información que no es de interés para el estudio. Por tanto, elimina información redundante e irrelevante, reduce las dimensiones del problema y maximiza los rasgos discriminantes.

3. La toma de decisiones del sistema se hace en la etapa de *Clasificación*. Se basa en las características extraídas para clasificar la medición. Su trabajo es asignar los patrones de clase desconocida a la categoría apropiada. [22]

   La clasificación puede ser de dos tipos:

   - Supervisada. Se usa un conjunto de aprendizaje etiquetado que sirve para entrenar al sistema.
   - No supervisada. El sistema no tiene el conjunto de muestras etiquetado para aprender a clasificar la información a priori, sino que se basa en cálculos estadísticos para clasificar los patrones. Este es el caso de las técnicas de *clustering*.

Un sistema de reconocimiento de patrones funciona, como se explica en la sección anterior, en dos modos distintos: entrenamiento y reconocimiento.

- Durante el *modo de entrenamiento* se realiza el diseño del extractor de características que representa los patrones de entrada y entrena al clasificador tomando un conjunto de datos de entrenamiento con el fin de minimizar el número de patrones mal identificados.
- Teniendo el clasificador ya entrenado, durante el *modo de reconocimiento* se toma como dato de entrada el vector de características de un patrón desconocido que será asignado a una de las clases o categorías. [2]

Dado un patrón, representado por un vector de n características

$$\mathbf{x} = (x(1), x(2), ..., x(n)) \tag{3.1}$$

se asigna a una de las c clases o categorías $C(i)$.

3.5.2 Selección de variables

Consiste en determinar cuál es el conjunto de características más apropiado para describir los objetos. Los tipos de selección de variables se dividen en dos en función de su finalidad:

- *Para la clasificación:* La selección de las características más acertadas de entre el total de características que describen a los objetos. Se hace para: mejorar la clasificación y/o aumentar la velocidad de procesamiento.
- *Para la representación:* Decidir qué características representan mejor a cierto tipo de objetos [13].

Existen dos estrategias fundamentales para llevar a cabo la selección de variables:

- *Aproximación directa o wrapper:* La selección de características se hace usando información de la función de clasificación. Cada subconjunto de variables candidato es evaluado mediante el modelo clasificatorio inducido, con el paradigma y las variables seleccionadas junto con la variable clase. Al final, se eligen las variables que proporcionan los mejores resultados de clasificación.
- *Aproximación indirecta o filter:* La selección se hace con un criterio independiente del clasificador. Este tipo de aproximación establece una medida indirecta de bondad de la selección de variables, normalmente se trata de un ranking entre las variables

predictoras teniendo en cuenta un criterio fijado de antemano de relevancia entre una variable predictora y la variable clase. Las variables predictoras se ordenan en función de dicha medida de relevancia seleccionándose las x primeras que se deseen para inducir con ellas el modelo clasificador.

La Figura 3.8 presenta el diagrama básico relativo al procedimiento de selección de subconjuntos de variables para problemas de clasificación supervisada [38]. La función objetivo va a depender de la estrategia de selección seguida.

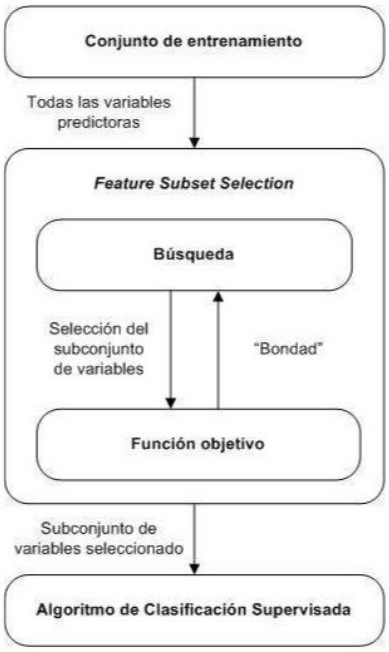


Figura 3.8: Procedimiento de selección de variables para problemas de clasificación supervisada

3.6 CARACTERÍSTICAS DE LA IMAGEN

La eficiencia de un proceso de detección aumenta de forma significativa si este proceso está basado en la detección de las características más significativas del tipo de objeto que debe ser reconocido.

3.6.1 Haar

Las características Haar son muy utilizadas en los algoritmos de clasificación en cascada, las cuales examinan los contrastes entre regiones de una misma imagen [19].

Una característica Haar examina regiones rectangulares adyacentes en un lugar concreto dentro de una ventana de detección, suma las intensidades de los píxeles dentro de estas

regiones y calcula la diferencia entre ellas. Luego, estas diferencias se utilizan para clasificar las regiones candidatas de una imagen. El número de características Haar existentes dentro de una ventana de detección es muy grande ya que se calculan con todos los posibles tamaños de rectángulo y todos los desplazamientos a lo largo de la ventana. Por tanto, para conseguir una clasificación rápida, se deben rechazar la mayoría de estas características y centrarse en un conjunto pequeño con las características más significativas para la clasificación.

En la Figura 3.9 podemos ver el conjunto base de características Haar formado por tres tipos de características: utilizando dos, tres y cuatro rectángulos. Para el caso de dos rectángulos, el valor representa la diferencia entre la suma de los píxeles dentro de las dos regiones rectangulares, horizontal o verticalmente adyacentes, que tienen el mismo tamaño y forma. Las características de tres rectángulos calculan la suma dentro de los dos rectángulos exteriores y a este valor se le resta la suma en el rectángulo central. Por último, las características de cuatro rectángulos calculan la diferencia entre pares diagonales de rectángulos.

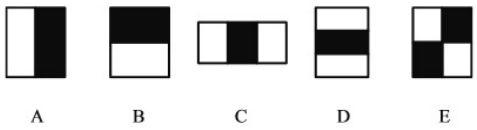


Figura 3.9: Conjunto base de características Haar [34]

(A) y (B) muestran las características de dos rectángulos en sentido horizontal y vertical respectivamente, (C) y (D) características de tres rectángulos en sentido horizontal y vertical respectivamente, y (E) características de cuatro rectángulos.

Una de las contribuciones de Viola y Jones fue la utilización de tablas de áreas sumadas, que llamaron imágenes integrales [31]. Dichas imágenes pueden describirse como tablas de búsqueda de dos dimensiones que mantienen la estructura de una matriz y el tamaño de la imagen original. Cada elemento de la imagen integral contiene la suma de todos los píxeles situados en la región de arriba a la izquierda de la imagen original (en relación con la posición del elemento). Esto permite calcular la suma de las áreas rectangulares en la imagen, en cualquier posición o escala, utilizando sólo cuatro operaciones de búsqueda:

$$sum = I(C) + I(A) - I(B) - I(D) \tag{3.2}$$

donde los puntos A, B, C y D pertenecen a la imagen integral I como se puede ver en la Figura 3.10.

Cada característica Haar puede necesitar más de cuatro operaciones de búsqueda, en función de su definición. Las características de dos rectángulos de Viola y Jones necesitan

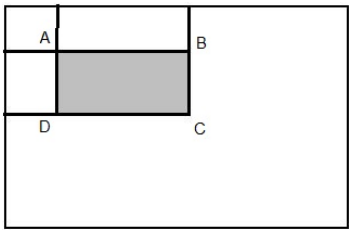


Figura 3.10: Suma del área rectangular sombreada en la imagen integral

seis operaciones de búsqueda, las de tres rectángulos necesitan ocho, y las características de 4 rectángulos requieren 9.

3.6.2 Texturas

Al examinar ciertos tipos de materiales se puede observar cierta homogeneidad en su aspecto, como si este siguiese algún tipo de patrón. Estos patrones visuales que se aprecian se conocen con el nombre de texturas (Fig. 3.11). La textura puede variar en función del objeto y de la resolución de la imagen. La visualización de los patrones depende de la resolución de la imagen, pues un patrón visto con una determinada resolución puede ser totalmente distinto al de otra resolución mayor o menor. En el estudio de texturas suele aplicarse un análisis estadístico de orden uno o superior [18].

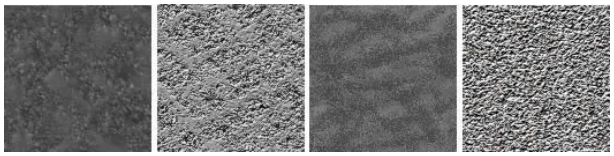


Figura 3.11: Algunos tipos de texturas

El análisis de texturas tiene es de fundamental importancia en el área de la visión por computador y el procesamiento de imagen. En las últimas décadas se propusieron muchos métodos para la extracción de características y la clasificación. El análisis de texturas dispone de numerosas aplicaciones en imagen médica además de en otras áreas. Los métodos existentes se dividen en: métodos estadísticos, métodos de procesamiento de señal, métodos estructurales y métodos basados en modelos [35].

Análisis estadístico de texturas

El análisis estadístico de texturas estudia las diferentes propiedades de la imagen. Los estadísticos pueden ser de primer orden, segundo orden y órdenes superiores en función de los puntos definidos durante el análisis. En este caso, el análisis está orientado al estudio de los niveles de gris de la imagen a través de estadísticos de primer y segundo orden.

Estadísticos de primer orden

Los estadísticos de primer orden se calculan a partir del histograma de la imagen [12]. El histograma representa la distribución de los niveles de color, o nivel de gris en este caso, de la imagen.

Estadísticos de segundo orden

Todas las características calculadas en base a los estadísticos de primer orden se mantienen invariantes ante cualquier permutación de los valores de gris de los *pixels*. Este tipo de estadísticos no permiten clasificar texturas diferentes que poseen distribuciones espaciales de color iguales ya que las características no dependen de la orientación ni de la colocación (Fig. 3.12). Debido a esto, además de utilizar estadísticos de primer orden, se utilizan estadísticos de segundo orden que a su vez detallan la distribución espacial de la textura [29].

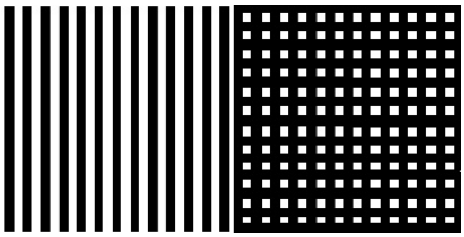


Figura 3.12: Texturas diferentes con el mismo histograma

Del mismo modo que los estadísticos de primer orden se basan en los histogramas, los de segundo orden se basan en las matrices de co-ocurrencia de los valores de los *pixels* [12]. La matriz de co-ocurrencia con ángulo a y distancia d refleja la frecuencia de aparición de un píxel de referencia con un píxel vecino situado a una distancia d en la dirección dada por el ángulo a (Fig. 3.13).

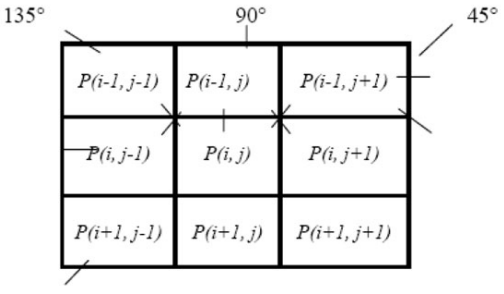


Figura 3.13: Ángulos para calcular la matriz de co-ocurrencia

En este caso se emplearon las distancias $d = 1,2,5$ y los ángulos a=0º, 45º, 90º, 135º para cada una de las distancias. En total, para cada imagen, se calculan 13 características de primer orden y 19 para cada una de las matrices de co-ocurrencia determinadas con lo que se obtienen 241 descriptores.

3.7 CLASIFICADORES

Un clasificador es una función *f(x)* que asigna una clase a una muestra:

$$f(x) = X \rightarrow C \tag{3.3}$$

siendo X el conjunto de muestras (imágenes o sus características) y C el conjunto de etiquetas o clases de X.

Los problemas de detección de dos clases se caracterizan por tener una clase positiva (1) y una clase negativa (0).

Las consideraciones a tener en cuenta a la hora de crear un clasificador son: [33]

- *Capacidad de acierto* - Mide el número de decisiones correctas sobre el número total de muestras.
- *Rapidez* - Tiempo empleado en llevar a cabo la clasificación.
- *Claridad* - Agrupación entendible para los humanos.
- *Tiempo de aprendizaje* - Tiempo para obtener la información necesaria a partir de un conjunto de datos de entrenamiento.
- *Falsos positivos* - En el caso de un sistema CAD, se refiere a la detección de lesiones en una imagen cuando en realidad no lo son.

Los clasificadores pueden ser más o menos complejos. A los clasificadores más sencillos se les suele llamar *Weak Classifiers*, siendo los clasificadores más complejos conocidos como *Strong Classifiers*.

3.7.1 Clasificadores débiles

Los clasificadores débiles (o *weak classifiers* en inglés) deben ser moderadamente eficaces, es decir, que además de tratarse de un clasificador sencillo y rápido de construir y ejecutar debe obtener una capacidad de acierto algo superior a un clasificador aleatorio. En el caso de tener dos clases los aciertos deben suponer más del 50 % o el error menos del 50 %, ya que:

$$Error(\%) = 100\% - Aciertos(\%) \tag{3.4}$$

Como veremos en la sección 3.7.2 se puede obtener un clasificador más preciso a partir de la combinación de muchos clasificadores “débiles”.

Árboles de decisión

En el campo de la medicina, lo ideal sería que los profesionales médicos pudieran contar con el apoyo de una herramienta que les permitiera analizar los datos obtenidos de cada uno de sus pacientes para poder hallar, en función de casos que ya han sido tratados, el diagnóstico más acertado así como el tratamiento a seguir, lo cual representaría una gran ayuda para el médico. Una herramienta alternativa para la predicción y clasificación de grandes cantidades de datos que es utilizada ampliamente en el campo de la inteligencia artificial son los árboles de decisión.

Los árboles de decisión son una técnica inductiva que está basada en una estructura de árbol que clasifica una serie de datos iniciales en determinados grupos o clases. Los nodos representan atributos y las ramas representan los posibles valores del atributo. A partir de estos datos iniciales, un árbol de decisión podrá clasificar nuevos datos que le sean presentados. Cuando el algoritmo recibe un dato, éste lo va procesando por cada nodo del árbol y "decide" la rama del nodo por la que continuar en función del valor del atributo seleccionado por el nodo y un valor umbral calculado.

La Figura 3.14 muestra la estructura que tienen los árboles de decisión:

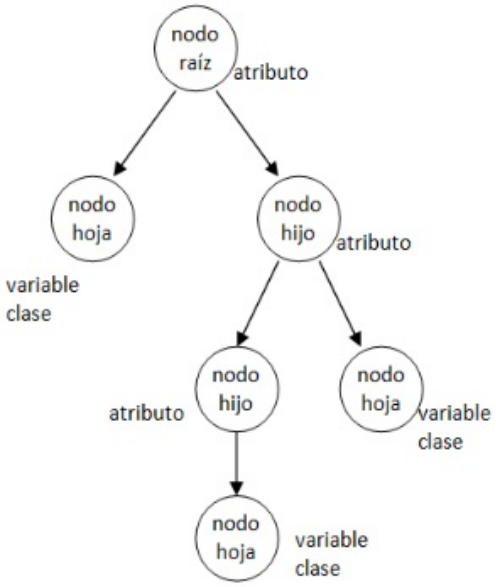


Figura 3.14: Estructura de un árbol de decisión[23]

Decision *Stumps*

Un *Stump* es un pequeño árbol de decisión que consta únicamente del nodo raíz y dos hojas (Fig. 3.15).

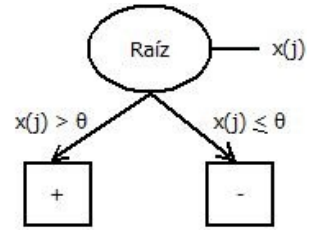


Figura 3.15: Estructura del clasificador *Stump*

En este caso, *f(x)* es una función que asigna una determinada clase a *x* en función de un cierto umbral θ.

Cada muestra *x* tiene una serie de características *x(j)*. *f(x)* asignará una muestra a una u otra clase en función de si cierta *x(j)* es mayor o menor que un umbral θ. Por tanto:

$$f(x(j),\theta) = \begin{cases} 1 & \text{si } x(j) > \theta \\ 0 & \text{caso contrario} \end{cases} \tag{3.5}$$

El error de este clasificador será:

$$error = \frac{n^o\ fallos}{n^o\ total\ muestras} \tag{3.6}$$

en caso de que todas las muestras tengan la misma importancia.

Si la penalización del clasificador es más alta en unos casos que en otros, habrá que asignar pesos y el error vendrá dado por la expresión:

$$error = \frac{\sum_{i \in fallos} w_i}{\sum_{i=1}^{n} w_i} = \frac{\sum pesos\ de\ fallos}{\sum pesos\ totales} \tag{3.7}$$

De entre todos los *Stumps* posibles, el que mejor clasifique los patrones dados será el que obtenga menor error en base a una serie de pesos, es decir, el par de (j*, θ*), característica y umbral que minimice dicho error, de forma que:

$$(j^*, \theta^*) := arg\ min_{j,\theta} \frac{\sum_{i=1}^{n} w_i \cdot eval\{c_i \neq f(x_i(j),\theta)\}}{\sum_{i=1}^{n} w_i} \tag{3.8}$$

siendo w_i el peso de cada una de las muestras, x_i la muestra en cuestión, $f(x_i(j),\theta)$ la etiqueta asignada por el *Stump* a la muestra x_i con la característica j y el umbral θ y c_i la etiqueta real de la muestra x_i.

Además, *eval* es una función que vale 1 o 0 en base a si se cumple o no la condición.

$$eval \begin{cases} eval\{verdadero\} = 1 \\ eval\{falso\} = 0 \end{cases} \tag{3.9}$$

La cascada básica propuesta por Viola y Jones utiliza los *Stumps* como clasificadores ligeros básicos del algoritmo *AdaBoost*, como se verá más adelante.

3.7.2 Clasificadores Fuertes

Decimos que un clasificador es fuerte (o *Strong Classifier* en inglés) cuando su función de decisión es compleja, necesitando tiempo y otros recursos para su entrenamiento y ejecución o cuando está formado a partir de la combinación de un conjunto finito de clasificadores débiles, ponderados en función del peso asociado a cada uno, dando como resultado un clasificador significativamente más efectivo y complejo.

Support Vector Machines (SVM)

Los algoritmos de aprendizaje automático han experimentado un importante avance a través de las técnicas basadas en *kernel* como máquinas de soporte vectorial, máquinas de punto de Bayes, análisis de componentes principales del *kernel* y procesos gausianos.

Las máquinas de vectores de soporte (en inglés *Support Vector Machines*, SVM) son un grupo de métodos de aprendizaje supervisado que se pueden aplicar a la clasificación. Dado un conjunto de muestras de entrenamiento, podemos etiquetar las clases y entrenar un SVM para construir un modelo que prediga la clase de una nueva muestra.

Un SVM constituye un hiperplano o conjunto de hiperplanos en un espacio de dimensionalidad muy alta (o incluso infinita). El algoritmo SVM se basa en encontrar el hiperplano que da la mayor distancia mínima de los ejemplos de entrenamiento. Cuanto mejor sea la separación entre las clases, mejor será la clasificación obtenida. A la vez que minimiza el error de clasificación en el entrenamiento, maximiza el margen para mejorar la generalización del clasificador y generaliza de manera eficaz con pocas muestras de entrenamiento. Este algoritmo presenta un buen rendimiento al generalizar en problemas de clasificación.

Estas máquinas representan una extensión de los modelos no lineales del algoritmo desarrollado por Vladimir Vapnik y Alexey Chervonenkis [27].

Funciones *Kernel*

Con el clasificador hiperplanar se utilizan productos escalares sobre el espacio de características, si se hace la sustitución (*kernel trick*):

$$\langle x, x' \rangle = k(x, x') \tag{3.10}$$

La función k encierra tanto un producto escalar como una traslación de vectores (*varphi-image*) a un nuevo espacio de características, puede que de mayor dimensionalidad que el original.

A continuación se detallan las distintas funciones kernel existentes y su representación en la Figura 3.16:

- **Gausiano:**

$$k(x,x') = \exp(-\frac{\|x-x'\|^2}{2\sigma^2}) \tag{3.11}$$

- **Polinomial:**

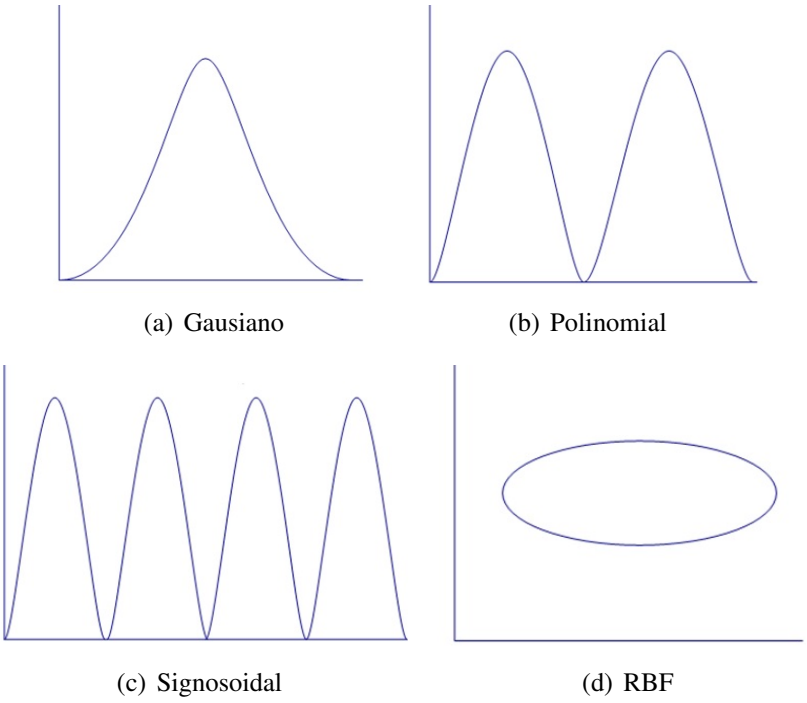
$$k(x,x') = \langle x,x' \rangle^d, d \in \mathbb{N} \tag{3.12}$$

- **Signosoidal:**

$$k(x,x') = \tanh(\kappa\langle x,x' \rangle + \zeta)\kappa > 0, \zeta > 0 \tag{3.13}$$

- **RBF:**

$$k(x,x') = f(d(x,x')), d : métrica, f : \Re^+ \to \Re \tag{3.14}$$



(a) Gausiano (b) Polinomial

(c) Signosoidal (d) RBF

Figura 3.16: Tipos de funciones Kernel

Construcción de un SVM

Partimos de un conjunto de datos de entrenamiento x_i, c_i con:

$$i = 1,...,l, c_i \in \{0,1\},\ x_i \in R^d \tag{3.15}$$

entonces existe un hiperplano, como el de la Fig. 3.17, que separa los datos de etiquetas positivas y negativas, tales que:

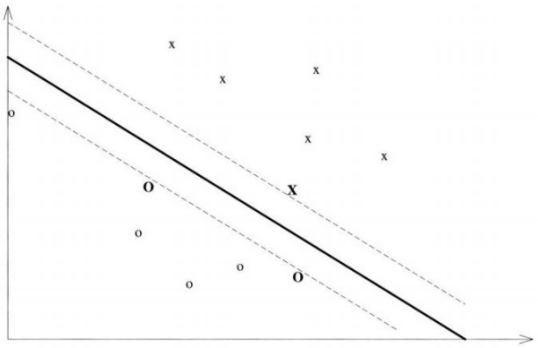


Figura 3.17: Representación gráfica de SVM [1]

Con la construcción de una función clasificadora se pretende:

- Minimizar el error en la separación de los objetos dados.
- Maximizar el margen de separación, es decir, mejorar la generalización del clasificador.

AdaBoost

AdaBoost (*Adaptive Boosting*) está compuesto por varios clasificadores ligeros que forman uno más potente. Cada clasificador ligero realiza la agrupación para una sola dimensión en un vector de características.

El clasificador final es una combinación lineal de los ligeros:

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + ... + \alpha_n f_n(x) \tag{3.16}$$

siendo $f_n(x)$ un clasificador débil y α_n el peso asociado a ese clasificador sencillo n.

El algoritmo para construir $F(x)$ mediante *AdaBoost* es el siguiente:

ENTRADA: Conjunto de muestras etiquetadas.
$(x_1, c_1), (x_2, c_2), ..., (x_n, c_n)$ donde $x_i \in$ X, $c_i \in$ C = {0, 1}

SALIDA: $F(x)$.

Paso 1. Se calculan los pesos iniciales de cada muestra:

$$w_{1,i} = \frac{1}{n} \tag{3.17}$$

Paso 2. Durante T iteraciones (t = 1, ..., T), siendo T el número final de clasificadores débiles del clasificador *AdaBoost*.

1. Normalizar los pesos para que la suma de todos ellos sea 1 y estos sigan una función de distribución de probabilidad.

$$w_{t,i} = \frac{w_{t,i}}{\sum_{k=1}^{n} w_{t,k}} \tag{3.18}$$

2. Desde $j = 1$ hasta $j = n^o$ características.
   (a) Obtener el *Stump* $f_{t,j}$ con la característica j y los pesos w_t.
   (b) Calcular el error de $f_{t,j}$

$$e_{t,j} \leftarrow \sum_{i=1}^{n} w_{t,i} \cdot \mid f_{t,j}(x_i) - c_i \mid \tag{3.19}$$

3. Elegir el f_t con el menor error e_t.
4. Actualizar los pesos.

$$w_{t+1,i} \leftarrow w_{t,i} \left(\frac{e_t}{1 - e_t} \right)^{1-|f_t(x_i)-c_i|} \tag{3.20}$$

5. Hallar el peso del *Stump*.

$$\alpha_t \leftarrow \log \frac{1 - e_t}{e_t} \tag{3.21}$$

Paso 3. Obtener $F(x)$.

$$F(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t f_t(x) \geq \left(\frac{1}{2}\right) \sum_{t=1}^{T} \alpha_t \\ 0 & \text{c.c.} \end{cases} \tag{3.22}$$

3.7.3 Cascada de Clasificadores de Viola y Jones

Para algunos clasificadores ligeros, este método puede reducir eficazmente la complejidad y tiempo de cálculo, pero la capacidad de acierto obtenida es baja y aún tiene una tasa alta de falsos positivos [16].

Para solucionar este problema, Viola y Jones propusieron un clasificador basado en una cascada de varios clasificadores *AdaBoost*. En cada etapa de la cascada se entrena un clasificador para detectar casi todos los objetos buscados, o muestras positivas, mientras se elimina una fracción de muestras negativas. Asumiendo que haya L capas en este clasificador, que las tasas de detección y de falsos positivos de cada etapa sean d_i y f_i respectivamente, y

que las etapas sean independientes, las tasas de detección y de falsos positivos de la cascada completa se obtienen a partir de las k etapas [14]:

$$D(x) = \prod_{i=1}^{L} d_i \qquad F(x) = \prod_{i=1}^{L} f_i \tag{3.23}$$

Si se tiene en cuenta que todas las etapas mantienen los mismos valores de f_i y d_i, en el caso de tener una cascada con 25 etapas, $d = 0,999$ y $f = 0,5$, se obtiene una $D = 0,999^{25} \approx 0,98$ y una $F = 0,5^{25} \approx 3 \cdot 10^{-8}$. En este ejemplo se aprecia que la tasa de detección o verdaderos positivos decrece muy lentamente, mientras que la tasa de falsos positivos lo hace rápidamente.

El principio básico del algoritmo de detección de Viola&Jones consiste en dividir una imagen en subventanas y escanear cada una de ellas en diferentes posiciones y distintas escalas [45]. Por tanto, los autores proponen el método basado en una cascada de clasificadores para reducir el tiempo de clasificación manteniendo los requisitos de rendimiento del clasificador. En cada etapa, el clasificador tiene que detectar si la subventana que se analiza es positiva o negativa. Cuando una de las subventanas se clasifica como negativa, esta es descartada inmediatamente. Si se considera una posible subventana como positiva, esta pasa a la siguiente etapa del clasificador. Se considera que una subventana contiene una zona positiva si y sólo si pasa a través de todas las etapas del detector sin que sea descartada. [21]

Como se puede apreciar en la Figura 3.18, se van eliminando muestras en cada etapa (la mitad de las negativas) manteniéndose un 99 % de las positivas. Por tanto, los FP (falsos positivos) disminuyen de forma más rápida mientras que la detección lo hace más lentamente.

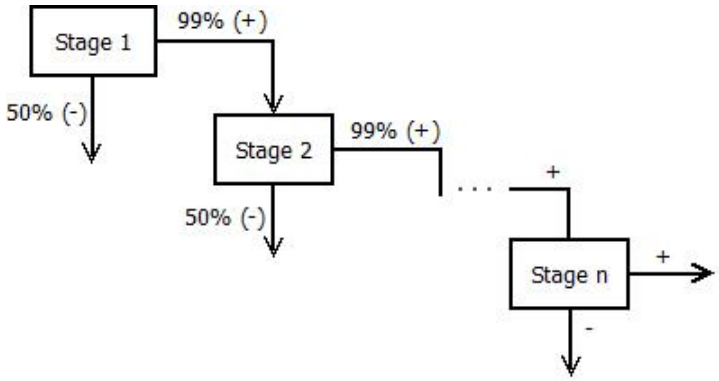


Figura 3.18: Cascada de clasificadores

Las ventajas que ofrece la clasificación en cascada son:

- Alta precisión.
- Diseño rápido, eficiente y simple. La cascada detecta de forma rápida y sin necesidad de muchas operaciones las muestras negativas más fáciles de clasificar.

- En cada etapa no se necesitan todas las características, sino que se seleccionan las más importantes mediante la combinación de varios clasificadores ligeros utilizando un algoritmo de *boosting*.

Algoritmo 1: Cascada de Clasificadores

Data: P = Conjunto de muestras positivas
N = Conjunto de muestras negativas
f = Tasa máxima de FP por etapa
d = Tasa mínima de detección por etapa
F = Tasa máxima de FP global
Result: Cascada de Clasificadores (vector con los H(x))
inicio;
$F_0 = 1$; $D_0 = 1$; i = 0;
while $F_i > F_{target}$ **do**
    i = i + 1; $n_i = 0$; $F_i = F_{i-1}$;
    while $F_i > f \times F_{i-1}$ **do**
        n_i++;
        Obtener un *AdaBoost* con n_i características usando las muestras {P, N}.
        Evaluar la cascada completa con la nueva etapa sobre el conjunto de validación determinando F_i y D_i.
        while $D_i < d \times D_{i-1}$ **do**
            Decrementar el umbral del *AdaBoost* de la última etapa.
    N'={};
    if $F_i > F$ **then**
        Añadir a N' las imágenes de N que la cascada clasifica como positivas incorrectamente.
        N = N'

Cascada de Cheng y Jhan

El algoritmo de clasificación en cascada de Cheng y Jhan propone una cascada de clasificadores combinando *AdaBoost* y SVM, modificando el algoritmo de entrenamiento del clasificador de Viola&Jones para hacerlo apto para la construcción de un clasificador en cascada *AdaBoost*-SVM y poder obtener mayor tasa de detección y menor tasa de falsos positivos utilizando, además, un número de etapas menor. [15]

El clasificador *AdaBoost* en las etapas iniciales podría alcanzar los objetivos preestablecidos con un número pequeño de clasificadores débiles; pero a medida que avanzan las etapas, las muestras del conjunto de entrenamiento restante son menos y más similares. El clasificador *AdaBoost* en las etapas finales necesitaría una combinación lineal de más clasificadores

débiles para alcanzar los objetivos propuestos, lo que provocaría sobreajuste y mayor tiempo de ejecución. Este problema se soluciona sustituyendo el clasificador *AdaBoost* en las etapas finales de la cascada aplicando el algoritmo SVM. Para ello, se considera un número de clasificadores débiles umbral n_{th} a partir del cual *AdaBoost* será sustituido por SVM.

El algoritmo, propuesto por Cheng y Jhan, utilizado para la construcción del clasificador en cascada *AdaBoost*-SVM es el siguiente:

Algoritmo 2: Algoritmo de entrenamiento para construir una clasificador en cascada utilizando *AdaBoost* y SVM

Data: P = Conjunto de muestras positivas
N = Conjunto de muestras negativas
f = Tasa máxima de FP por etapa
d = Tasa mínima de detección por etapa
F = Tasa máxima de FP global
n_{th} = el máximo número aceptable de clasificadores débiles
Result: Clasificador en cascada Adaboost-SVM
inicio;
$F_0 = 1$; $D_0 = 1$; i = 0;
while $F_i > F_{target}$ **do**
    i = i + 1; $n_i = 0$; $F_i = F_{i-1}$;
    while $F_i > f \times F_{i-1}$ **do**
        n_i++;
        Obtener un *AdaBoost* con n_i características usando las muestras {P, N}.
        Evaluar la cascada completa con la nueva etapa sobre el conjunto de validación determinando F_i y D_i.
        while $D_i < d \times D_{i-1}$ **do**
            Decrementar el umbral del AdaBoost de la última etapa.
    if $n_i \geq n_{th}$ **then**
        Usar P y N para entrenar un clasificador con las n_i características elegidas por *AdaBoost* usando SVM. Evaluar el clasificador de cascada actual en el conjunto de validación para determinar F_i y D_i.
    else
    $\bar{N} = \{\}$;
    if $F_i > F$ **then**
        Añadir a N' las imágenes de N que la cascada clasifica como positivas incorrectamente.
        N = N'

El algoritmo de clasificación de Cheng y Jhan utiliza un SVM con kernel de tipo RBF, con los parámetros $C = 10$ y $\sigma^2 = 0{,}5$.

Cascada de Chen y Chen

Para resolver el problema del gran número de clasificadores débiles en las últimas etapas de la cascada, Chen y Chen proponen un enfoque nuevo para la cascada que puede reducir eficazmente el número de clasificadores débiles usados por el clasificador *AdaBoost*, pero requiere cálculos más complicados. En esta variante de la cascada se añaden a la cascada original algunos clasificadores metaetapa para utilizar la información entre etapas y aprender nuevos límites de clasificación para mejorar el rendimiento de la detección (Fig. 3.19).

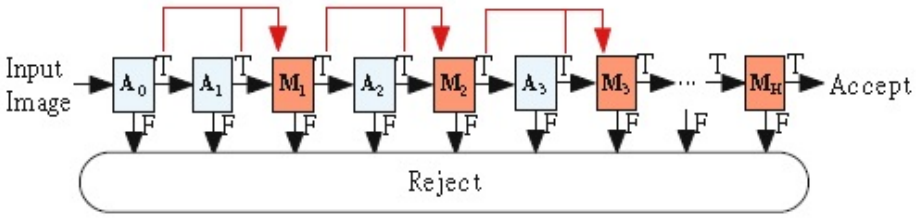


Figura 3.19: Estructura de cascada con metaetapas [16]

En la Figura 3.19, A_k y M_k representan las etapas *AdaBoost* y las metaetapas respectivamente. En este caso, cada metaetapa es un clasificador en un espacio bidimensional. El vector de entrada de la primera metaetapa M_1 es un vector bidimensional $(Conf(A_0), Conf(A_1))$, donde $Conf(A_i)$ es el grado de confianza de la etapa i de *AdaBoost*. El vector de entrada de las otras metaetapas $M_i(i = 2, ..., H)$ es también un vector bidimensional $(Conf(M_{i-1}), Conf(A_i))$ que representa el valor de confianza de las dos etapas previas de la cascada y donde $Conf(M_i)$ es el grado de confianza de la metaetapa i.

Del mismo modo que en una etapa *AdaBoost*, una metaetapa se diseña con el objetivo de cumplir los ratios predefinidos de acierto de muestras positivas y negativas.

Las metaetapas introducidas requieren poco tiempo computacional ya que sólo llevan a cabo una clasificación bidimiensional. Sin embargo, ayuda a rechazar más muestras negativas durante el entrenamiento de la cascada completa. Durante la implementación, normalmente se fija como objetivo de cada metaetapa una tasa de detección del 100 %, lo que hace que todas las muestras sean clasificadas correctamente, eligiéndose de entre todos los posibles el clasificador con la tasa de eliminación de muestras negativas más alta bajo esta condición [16]. Este criterio no influye en la última decisión del clasificador previo *AdaBoost* sobre los datos positivos, pero ayuda a la eliminación de más muestras negativas. Por tanto, añadiendo las metaetapas, puede reducirse el número total de etapas *AdaBoost* requeridas fijando los mismos objetivos lo que se traduce en un aumento de la tasa de detección bajo la misma tasa de falsos positivos.

Clasificador de Metaetapa

El método de clasificación usado en las metaetapas puede ser arbitrario, pero Chen y Chen proponen utilizar SVM como clasificador metaetapa debido a su alta capacidad de generalización y a su eficiencia durante la evaluación. Para entrenar el clasificador de cada metaetapa, se aplica una validación cruzada de tres grupos para seleccionar el mejor parámetro de penalización *C* del SVM con kernel lineal. Finalmente, se obtiene un hiperplano que delimita los datos de entrenamiento positivos y negativos.

Para lograr el objetivo de la metaetapa, hay que mover el hiperplano a lo largo de su dirección normal, aplicando diferentes umbrales, para encontrar el que tiene la tasa más alta de eliminación de datos de entrenamiento negativos manteniendo el 100 % de las muestras positivas detectadas correctamente (Fig. 3.20).

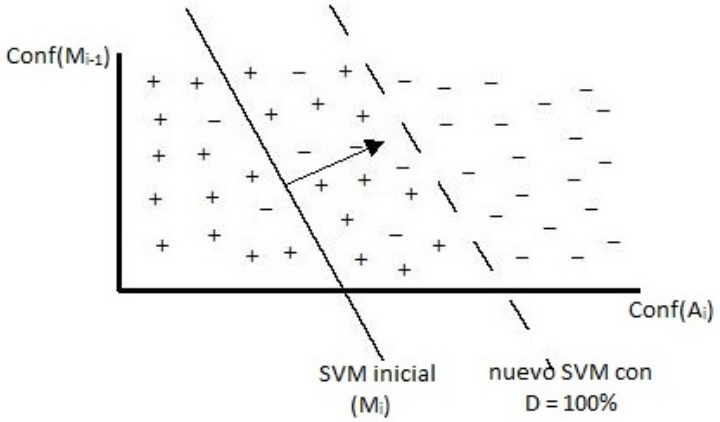


Figura 3.20: Representación del cambio de umbral del SVM

Cada metaetapa contiene el grado de confianza de todas las etapas previas, ya que cada metaetapa, exceptuando la primera, utiliza como entrada el grado de confianza de su metaetapa previa más cercana. Por tanto, la información de etapas anteriores se propagará de forma iterativa según se vaya avanzando en las metaetapas del clasificador.

CAPÍTULO 4

MÉTODO DE TRABAJO

Este capítulo describe el método de trabajo que se ha utilizado para la realización del presente trabajo. Los aspectos relacionados con la Ingeniería del *Software*, que ofrecen técnicas para exponer la metodología de desarrollo utilizada junto con las diferentes iteraciones que se han realizado y la planificación que se ha seguido para su consecución.

Para la realización del presente Trabajo, se ha seguido un desarrollo en base a los prototipos incrementales de DSDM (en español *Método de Desarrollo de Sistemas Dinámicos*) y se ha completado con diagramas basados en el PUD (*Proceso Unificado de Desarrollo*).

4.1 MÉTODO DE DESARROLLO DE SISTEMAS DINÁMICOS (DSDM)

DSDM es un método que proporciona un *framework* para el desarrollo ágil de *software*. Está basado en la continua implicación del usuario en un desarrollo iterativo e incremental que se adapte a los posibles cambios en los requisitos para obtener un sistema capaz de abarcar las necesidades de la empresa en tiempo y presupuesto [17].

DSDM apareció en enero de 1994 con el fin de crear una metodología RAD (en inglés *Rapid Application Development*) unificada. Fue pensado con el fin de completar lo que le faltaba al método RAD, al proporcionar una estructura que tome en cuenta el ciclo de desarrollo completo.

Esta metodología se caracteriza por:

- Participación del usuario
- Desarrollo iterativo y creciente
- Frecuencia de entrega mejorada
- Pruebas integradas en cada fase

- La aceptación de los productos entregados depende directamente del cumplimiento de los requisitos

4.1.1 Ciclo de Vida

DSDM consta de 3 fases: fase del preproyecto, fase del ciclo de vida del proyecto, y fase del postproyecto (Fig. 4.1).

- Fase del Preproyecto: Cosas que son necesarias antes de comenzar el proyecto.
- Fase del Ciclo de Vida del Proyecto: El proyecto se lleva a cabo. Esta fase se divide a su vez en 5 etapas:
    1. Estudio de Viabilidad
    2. Estudio del Negocio
    3. La iteración del Modelo Funcional
    4. Diseño e Iteración de la Estructura
    5. Implementación
- Fase del Postproyecto: Cosas que son necesarias una vez que el proyecto haya sido completado.

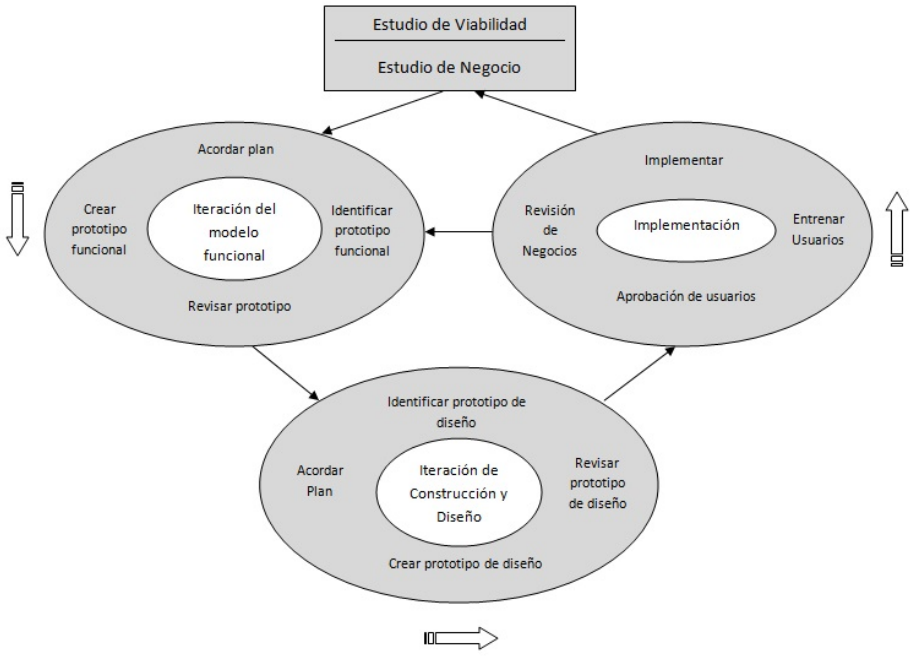


Figura 4.1: Ciclo de Vida DSDM

4.1.2 Artefactos

Describiremos los artefactos obtenidos a través de la metodología DSDM mediante las distintas fases que lo forman [43]:

Fase 1. Estudio de Viabilidad

- **Informe de Viabilidad:** Permite decidir que opción tomar, además de proporcionar información para saber si es conveniente seguir con el estudio de viabilidad o no.
- **Boceto del plan para el desarrollo:** Es el resultado de la primera planificación para el proyecto. En esta etapa se establecen los plazos e hitos importantes para varias fases de trabajo, o entregas clave.
- **Enumeración de riesgos:** El registro de riesgos se comienza al inicio del proyecto, ya que debe estar presente durante todo el desarrollo del mismo.
- **Prototipo rápido:** Artefacto opcional, se utiliza en caso de no conocer lo suficiente el negocio o la tecnología.

Fase 2. Estudio del Negocio

- **Descripción de los procesos de negocio y especificación de casos de uso:** La identificación de los casos de uso implica una mayor participación por parte del cliente.
- **Documento de especificación de requisitos *software* (SRS):** Especificaciones de los requisitos a bajo nivel: diagramas ER, modelos de negocio objetivos, etc.
- **Definición de la arquitectura del sistema:** Describe la conexión entre los componentes hardware y *software* utilizables. Se proporciona una descripción tanto de la arquitectura técnica como de la funcional.
- **Boceto del plan de prototipado:** Conjunto de requisitos que se tratarán en la siguiente fase.
- **Lista priorizada de requisitos:** Detalla lo que debe hacer la solución propuesta.

Fase 3. Iteración del Modelo Funcional

- **Modelo Funcional:** Define lo que la solución va a hacer sin entrar en detalle de cómo van a funcionar aspectos no funcionales tales como la seguridad o el rendimiento. Se desarrolla a lo largo de todo el proyecto, profundizando su contenido en cada iteración.

- **Casos de prueba:** Pruebas a realizar.
- **Funciones prioritarias:** Conjunto de prioridades de las funciones que se entregan al final de cada iteración.
- **Resumen de los documentos de prototipos funcionales:** Reúne los comentarios de los usuarios sobre el incremento presente. Servirá de artefacto de entrega para las siguientes iteraciones.
- **Requisitos no funcionales:** Serie de requisitos que se considerarán en la siguiente fase.
- **Análisis de riesgos de desarrollo superior:** De gran importancia en esta fase, puesto que en la siguiente, los errores serán más difíciles de encontrar y corregir.
- **Plan de *timebox*:** Describe el plan para los *timebox* individuales dentro del modelo funcional, de diseño y de construcción de cada iteración.

Fase 4. Diseño e Iteración de la Estructura

- **Plan de *timebox*:** Describe el plan para los *timebox* individuales dentro del modelo funcional, de diseño y construcción de cada iteración.
- **Sistema Probado:** Indica cuando el sistema está listo para ser migrado a uso operativo. Se debe entregar toda la documentación y los registros de prueba de la iteración.

Fase 5. Implementación

- **Sistema entregado:** Puesta en marcha del sistema probado.
- **Manual de usuario:** Instrucciones de uso del sistema.
- **Informe de revisión del proyecto:** Resume el resultado del proyecto. En función de este informe, se establece el curso del desarrollo adicional.

4.2 PROCESO UNIFICADO DE DESARROLLO (PUD)

El Proceso Unificado de Desarrollo (PUD) [28] es una metodología de desarrollo *software* extensible que puede ser adaptada a las necesidades de organizaciones o proyectos específicos. El PUD define un “conjunto de actividades necesarias para transformar los requisitos de usuario en un sistema *software*”.

El PUD surge como una evolución del Proceso Unificado de Rational (RUP), algunas de sus características identificativas son que está basado en componentes *software* que se conectan mediante interfaces y que usa el lenguaje de modelado UML (*Unified Modelling Language*, Lenguaje Unificado de Modelado) para especificar los diagramas y demás artefactos que definen al sistema junto con el código ejecutable.

Con el objetivo principal de facilitar la creación de *software* de calidad que satisfaga las necesidades de los usuarios, el PUD proporciona un conjunto de métodos, técnicas y herramientas para la consecución de tal fin [39].

Las características que definen al PUD son:

- **Dirigido por casos de uso.** Para comenzar a desarrollar un sistema, la primera función que hay que realizar es identificar las funcionalidades que los usuarios necesitan para su interacción con el sistema. Con el término usuario se hace referencia no sólo a usuarios humanos, sino a otros sistemas que interactúan con el sistema a desarrollar.

  A estas funcionalidades les llamamos requisitos funcionales y se representan a través de los casos de uso. Cada caso representa un escenario de uso, que define la interacción del usuario con dicha funcionalidad. Por tanto los casos de uso van a ser una herramienta fundamental para guiar el proceso de desarrollo desde la especificación de requisitos hasta las etapas finales de las pruebas.

  Al conjunto de todos los casos de uso que presenta un sistema se le denomina modelo de casos de uso, y se representa a través del diagrama de casos de uso que especifica UML.

  La elección de los casos de uso que deben desarrollarse primero condiciona notablemente la arquitectura y la propia evolución del desarrollo. La arquitectura del sistema y los casos de uso maduran conforme avanza el ciclo de vida del sistema.

- **Centrado en la arquitectura.** La arquitectura de un sistema *software* se describe a través de sus diferentes vistas. El concepto de arquitectura *software* involucra los aspectos estáticos y dinámicos más significativos del sistema. La arquitectura surge de las necesidades de la empresa y está influenciada por factores como la plataforma de *software* en la que se ejecutará, la disponibilidad de componentes reutilizables o los requisitos no funcionales.

  El PUD asume que no existe un modelo único que cubra todos los aspectos del sistema y por tanto pueden existir múltiples modelos y vistas que definan la arquitectura *software* de un sistema.

  Arquitectura y casos de uso deben evolucionar en paralelo. Los casos de uso deben guiarse por la arquitectura y esta debe proveer espacio para la realización de todos los casos de uso, actualmente y en el futuro.

- **Iterativo e incremental.** El PUD está formado por una serie de iteraciones o mini-proyectos. Cada iteración se va a centrar en un conjunto de casos de uso elegidos cuidadosamente y va a ofrecer como resultado un incremento del producto desarrollado que añade o mejora las funcionalidades del sistema en desarrollo, por tanto va a suponer un refinamiento progresivo del sistema. El flujo de trabajo que va a seguir es el siguiente:

  - *Requisitos:* Se reúnen los requisitos funcionales del sistema.
  - *Análisis:* Se identifican y especifican los casos de uso.
  - *Diseño:* Se crea un diseño utilizando la arquitectura seleccionada como referente para tratar de dotar al sistema de las funcionalidades representadas por los casos de uso identificados en la etapa previa.
  - *Implementación:* Se traducen a código los artefactos creados durante la etapa de diseño.
  - *Pruebas:* Se verifica que el sistema cumple las especificaciones detalladas en los casos de uso.

A continuación se muestra un resumen gráfico del Proceso Unificado de Desarrollo:

4.2.1 Fases del Proceso Unificado de Desarrollo

El PUD divide el proceso de desarrollo de un proyecto en cuatro etapas, que a su vez se dividen en iteraciones. Cada iteración contempla alguno de los flujos de trabajo y cada fase concluye con un hito, donde se deben alcanzar una serie de objetivos propuestos. De este modo, las fases del PUD son:

1. **Inicio:** Se obtiene el modelo de casos de uso simplificado, se determinan los riesgos potenciales y se realiza una evaluación aproximada del proyecto.
2. **Elaboración:** Se mejora el modelo de casos de uso de la fase anterior y se diseña la arquitectura del sistema. Se desarrollan los casos de uso más críticos que se establecieron durante la fase de inicio.
3. **Construcción:** Se desarrollan todos los casos de uso y se prueba el *software*.
4. **Transición:** Se produce la entrega e instalación del *software* a los usuarios.

En la Figura 4.3 se muestra un gráfico que representa cada una de las fases dentro de un ciclo:

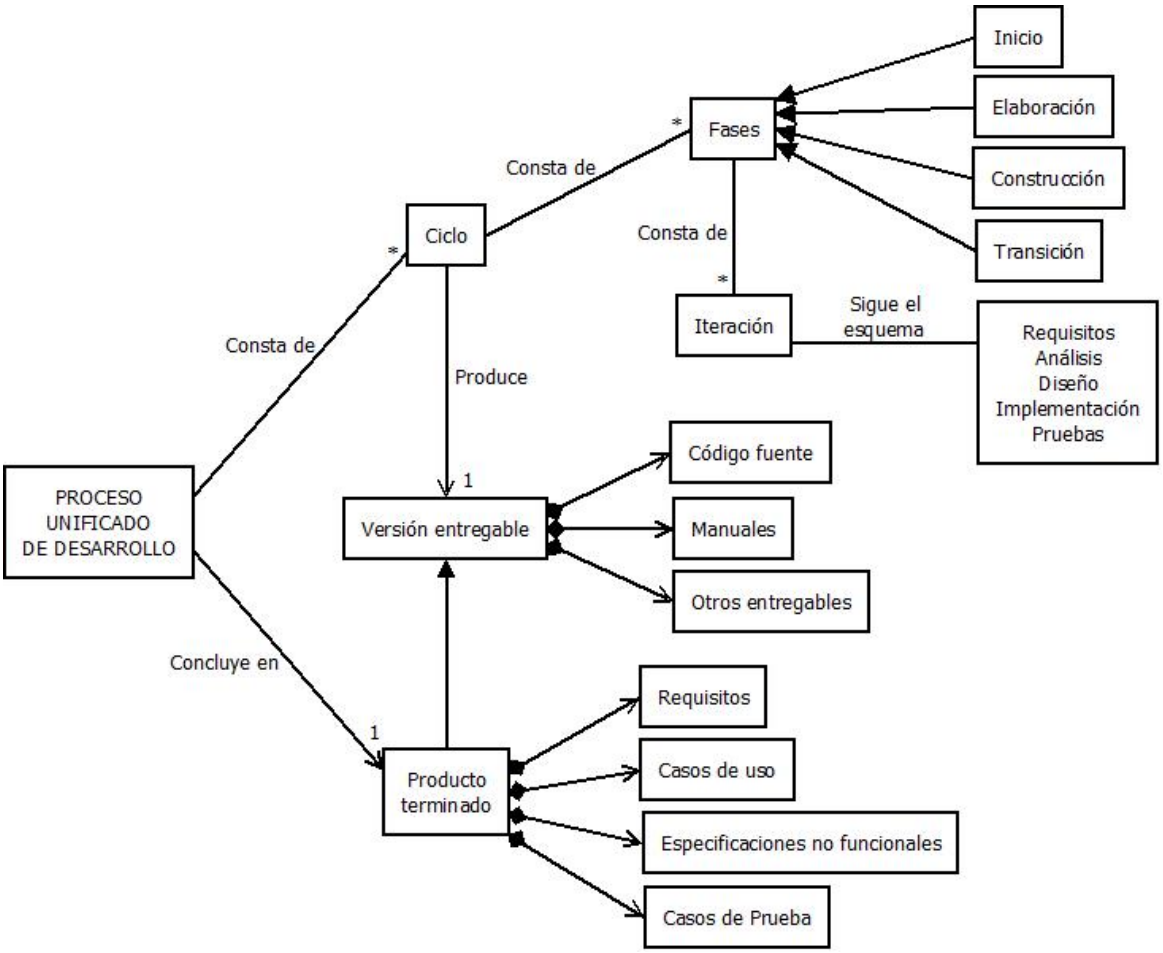


Figura 4.2: Mapa Conceptual del Proceso Unificado de Desarrollo

4.2.2 Artefactos

Durante el desarrollo *software* se construyen diferentes artefactos, de acuerdo con el PUD. Estos artefactos representan al sistema desde distintos puntos de vista y a diferentes niveles de abstracción. Los artefactos se generan a partir de los siguientes modelos [37]:

- **Modelo de Análisis.** Determina con mayor detalle los casos de uso y asigna las diferentes funcionalidades del sistema a un conjunto de clases.

- **Modelo de Diseño.** Constituye el diagrama de clases y de paquetes en forma de subsistemas, clases e interfaces.

- **Modelo de Implementación.** Especifica los componentes o ficheros físicos que contiene el sistema y su relación con las clases.

- **Modelo de Despliegue.** Representa el sistema mediante nodos o equipos, las relaciones entre nodos y las relaciones entre los diferentes componentes y los nodos.

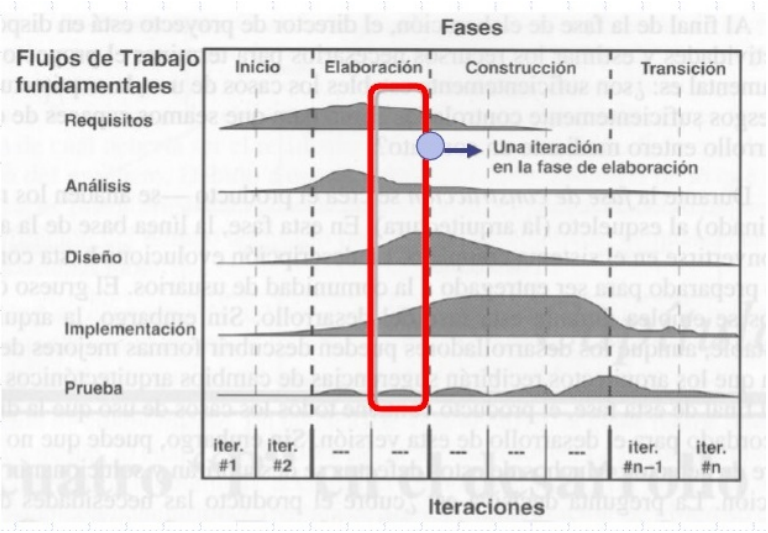


Figura 4.3: Fases del Proceso Unificado de Desarrollo [26]

- **Modelo de Prueba.** Constituye la especificación de los casos de prueba que verifican los casos de uso. En este modelo incluye casos de prueba que sirven para comprobar que se cumplen los requisitos determinados por el cliente.

CAPÍTULO 5

EVOLUCIÓN Y COSTES

5.1 EVOLUCIÓN DEL PROYECTO

Siguiendo la metodología de desarrollo DSDM, en esta sección se expone la planificación de los prototipos que se pretenden llevar a cabo para la realización de la herramienta propuesta en este Trabajo Fin de Grado, así como los diferentes flujos de trabajo que se desarrollan en cada una de ellos. El proyecto se ha estructurado en los prototipos que se describen a continuación.

5.1.1 Prototipo 1

Objetivos: Creación de la carga de imágenes.

Alternativas: Utilizar OpenCV u otras librerías para C/C++.

Riesgos: Soportar distintos formatos de imagen.

Resultados: Teniendo como entrada la ruta donde se encuentran las imágenes, devuelve como salida un ArrayList con las mismas.

5.1.2 Prototipo 2

Objetivos: Cálculo de las características Haar y de las estadísticas de textura.

Resultados: A partir de una matriz Mat de OpenCV con la imagen y del tipo de características elegidas para realizar el cálculo, devuelve como salida un vector con las características de la imagen.

5.1.3 Prototipo 3

Objetivos: Implementación de las clases relacionadas con los clasificadores débiles de la cascada de Viola&Jones.

Iteración 1

Objetivos: Implementación de las clases *WeakParams* y *WeakClassifier*.

Resultados: Clase con los parámetros de los clasificadores débiles (pesos con los que se debe entrenar, imágenes de entrenamiento, índice de la característica a utilizar y tipo de clasificador débil) y clase abstracta de la que heredarán los diferentes clasificadores débiles implementados y que establece los métodos y parámetros genéricos.

Iteración 2

Objetivos: Implementación de la clase *Stump*.

Resultados: Clase para el clasificador débil *Stump*.

5.1.4 Prototipo 4

Objetivos: Implementación de las clases relacionadas con los clasificadores de etapa de la cascada de Viola&Jones.

Iteración 1

Objetivos: Implementación de las clases *StageParams* y *StageClassifier*

Resultados: Clase con los parámetros de la etapa y clase abstracta de la que heredarán los diferentes clasificadores de etapa implementados y que establece los métodos y parámetros genéricos.

Iteración 2

Objetivos: Implementación de *AdaBoost*.

Resultados: Clasificador de etapa de *AdaBoost*.

5.1.5 Prototipo 5

Objetivos: Implementación de las clases principales de la cascada de Viola&Jones.

Iteración 1

Objetivos: Implementación de las clases *CascadeParams* y *Cascade*

Resultados: Clase con los parámetros de la cascada y clase abstracta de la que heredarán las diferentes cascadas implementadas y que establece los métodos y parámetros genéricos.

Iteración 2

Objetivos: Implementación de la clase *VJCascade*.

Resultados: Implementación de la cascada propuesta por Viola y Jones. Hereda de la clase abstracta *Cascade*.

5.1.6 Prototipo 6

Objetivos: Implementación del algoritmo Chen&Chen.

Iteración 1

Objetivos: Implementación del clasificador de etapa basado en SVM de Chen&Chen, *ChenSVM*.

Resultados: Clasificador SVM de meta-etapa para la clase posterior de la cascada Chen&Chen, *ChenCascade*.

Iteración 2

Objetivos: Implementación de la clase principal con el código de la cascada de Chen&Chen, *ChenCascade*.

Resultados: Implementación de la cascada propuesta por Chen y Chen. Hereda de la clase abstracta Cascade.

5.1.7 Prototipo 7

Objetivos: Implementación del algoritmo de Cheng y Jhan.

Iteración 1

Objetivos: Implementación del clasificador de etapa basado en SVM de Cheng y Jhan, *ChengSVM*.

Resultados: Clasificador de SVM de etapa para la clase posterior de la cascada Cheng&Jhan, *ChengCascade*.

Iteración 2

Objetivos: Implementación de *ChengCascade*.

Resultados: Implementación de la cascada propuesta por Cheng y Jhan. Hereda de la clase abstracta Cascade.

5.1.8 Prototipo 8

Objetivos: Diseño de la interfaz de usuario.

Alternativas: Realizar la interfaz con Microsoft Expression Blend 4 o con Windows Forms.

Resultados: Se optó por Windows Forms por compatibilidad con C++ y para conseguir una mejor integración de interfaz y algoritmos. Se puede observar el diseño final de la interfaz de usuario en las Figuras 5.1, 5.2 y 5.3.

5.1.9 Prototipo 9

Objetivos: Guardar resultados en un fichero y visualización de los mismos en la interfaz.

Resultados: El entrenamiento de las cascadas devuelve un fichero XML donde se recogen todas las variables que forman la cascada. Se puede ver un ejemplo del fichero generado en el Apéndice A.

5.1.10 Prototipo 10

Objetivos: Generación de la presente documentación.

Resultados: En esta iteración es donde se concluye la aplicación. En ella se ha generado la presente memoria del Trabajo Fin de Grado (en adelante TFG) y se ha obtenido la aplicación final, contenida en el CD, para que pueda ser utilizada en cualquier computadora. Además, se ha generado el manual de usuario en el que se explica de manera detallada cómo usar la aplicación. Este manual se puede ver en el Apéndice B.

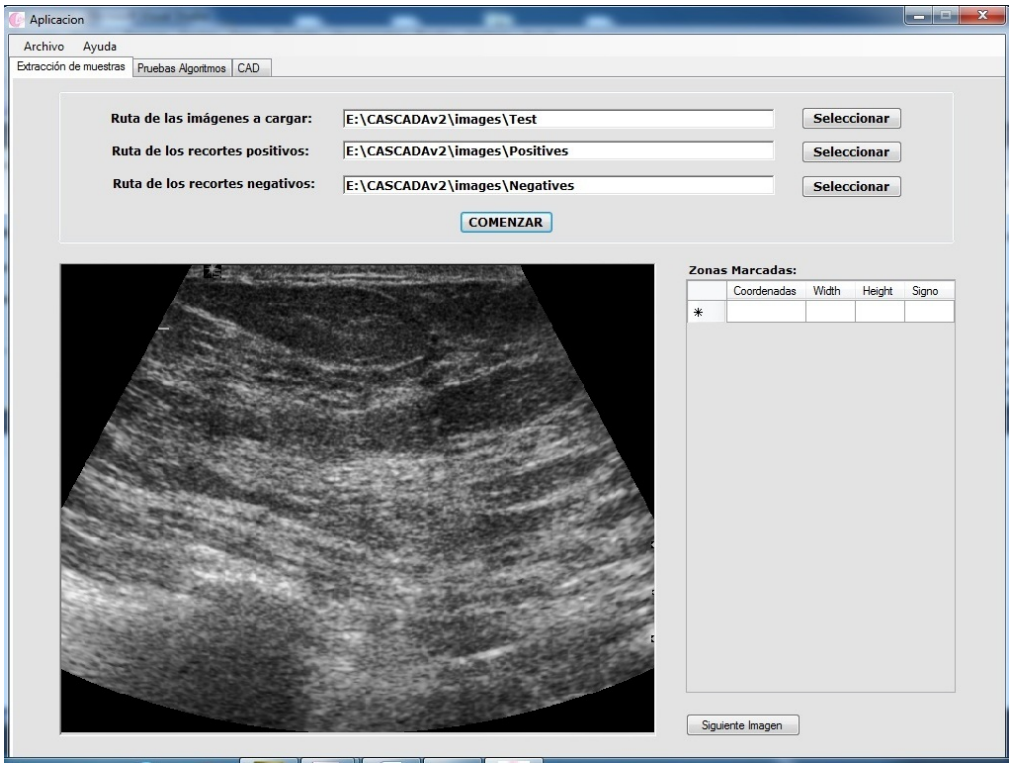


Figura 5.1: Extracción de las imágenes para el entrenamiento de los clasificadores

5.2 ESTIMACIÓN DE COSTES

Un factor fundamental a la hora de construir una aplicación es llevar a cabo una buena estimación del coste que supone el desarrollo de la misma. Para realizar esta estimación hay que tener en cuenta costes de personal, el coste de las instalaciones y los costes de software y de *hardware*.

Una mala realización de esta estimación puede suponer la negativa del cliente ante la aceptación del desarrollo de la aplicación si se sobrestima el coste o las pérdidas para la empresa en caso de infravalorarlo.

No es una tarea sencilla realizar una estimación del coste de la implementación de sistemas como el que se desarrolla en este proyecto. La dificultad fundamental radica en el hecho de que está sujeto a la evaluación médica para que sea fiable y utilizable. No obstante, en términos generales, se puede realizar una estimación del gasto de personal, de fungible e inventariable necesarios. No se incluyen cifras salariales ya que el coste es dependiente de los cambios en el mercado.

En esta sección se detallan los costes de personal, de *hardware* y software para el desarrollo del presente trabajo.

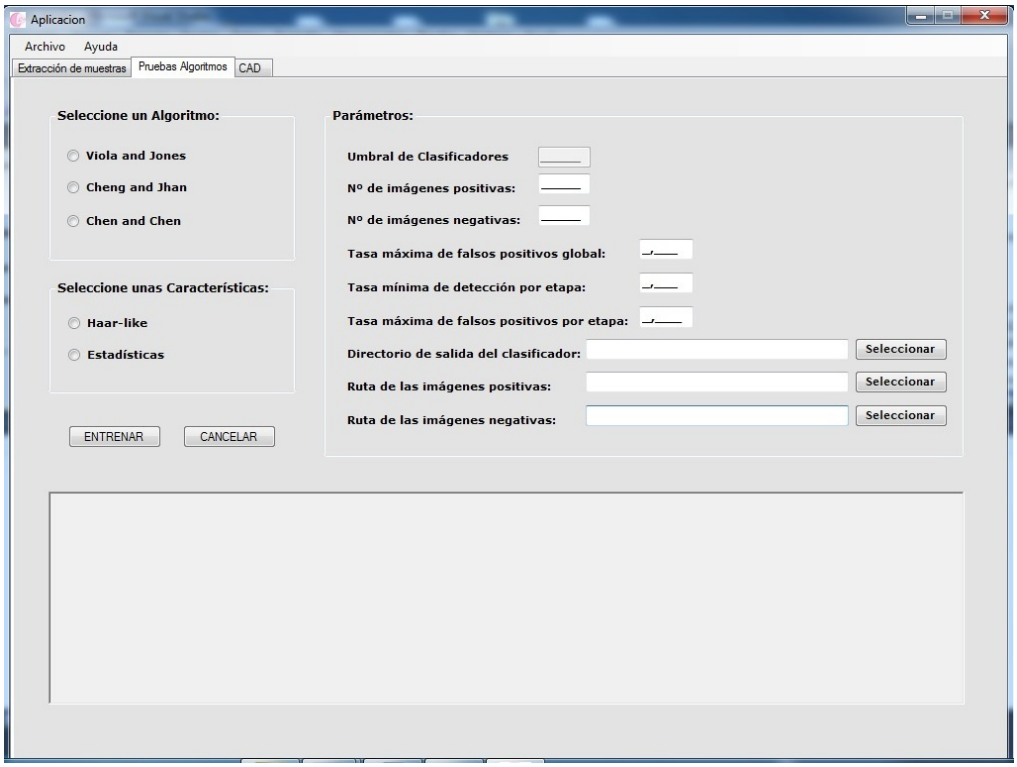


Figura 5.2: Entrenamiento de los clasificadores

5.2.1 Costes de Personal

Para realizar la estimación de costes el método empleado ha sido COCOMO [8]. Este modelo forma parte del grupo de los modelos algorítmicos que tratan de establecer una relación matemática la cual permite estimar el esfuerzo y el tiempo requerido para el desarrollo de un producto.

Este modelo está basado en modelos de estimaciones matemáticas, está orientado al producto final y no a fases intermedias y se basa en la cantidad de líneas de código del proyecto.

En primer lugar se debe calcular el esfuerzo, para lo que necesitamos hallar la variable KLDC (kilo de líneas de código). Esta variable se ha obtenido mediante un analizador de código para ver exactamente las líneas de código eliminando el código autogenerado en la interfaz (Fig. 5.4).

Por tanto, el KLDC es 4,171.

Se ha utilizado el tipo de modelo orgánico COCOMO ya que el proyecto no supera las 50 KLDC, y es el más adecuado para este caso. Además, se ha seleccionado COCOMO intermedio ya que este ofrece, generalmente, mejor estimación que el modelo básico.

Los coeficientes para el modelo orgánico son: $a = 3,2$ $b = 1,05$ $c = 2,5$ y $d = 0,38$.

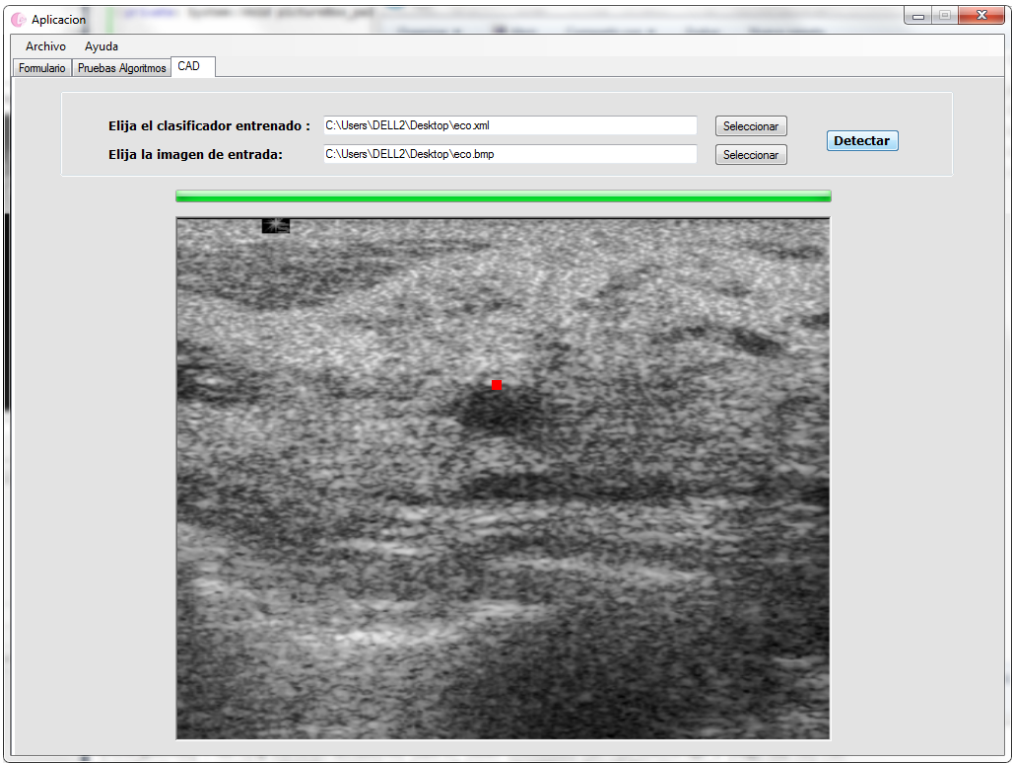


Figura 5.3: Detección de lesiones

El siguiente paso es calcular la variable FAE, la cual representa el factor de ajuste del esfuerzo. Esta se calcula como el producto de los valores seleccionados para 15 atributos del proyecto. En la Tabla 5.1 se muestran sombreados los valores seleccionados.

Por tanto, tenemos que la variable FAE, en este caso, es igual a:

$$FAE = 1,15 \cdot 1,15 \cdot 1 \cdot 1 \cdot 1,07 \cdot 0,86 \cdot 0,82 \cdot 0,70 \cdot 1 \cdot 0,95 \cdot 1 \cdot 0,91 \cdot 1,08 = 0,65 \qquad (5.1)$$

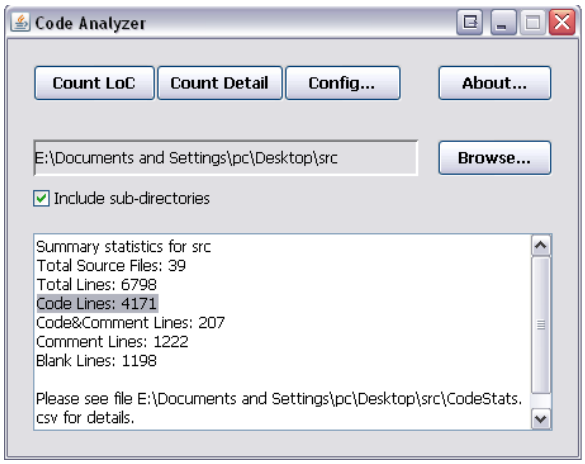


Figura 5.4: Resultado de un analizador de código

CONDUCTORES DE COSTE						
	Muy bajo	*Bajo*	*Nominal*	*Alto*	*Muy alto*	*Extr. alto*
Fiabilidad requerida del software	0,75	0,88	1.00	1,15	1,40	-
Tamaño de la base de datos	-	0,94	1.00	1,08	1,16	-
Complejidad del producto	0,70	0,85	1.00	1,15	1,30	1,65
Restricciones del tiempo de ejecución	-	-	1.00	1,11	1,30	1,66
Restricciones del almacenamiento principal	-	-	1.00	1,06	1,21	1,56
Volatilidad de la máquina virtual	-	0,87	1.00	1,15	1,30	-
Tiempo de respuesta del ordenador	-	0,87	1.00	1,07	1,15	-
Capacidad del analista	1,46	1,19	1.00	0,86	0,71	-
Experiencia en la aplicación	1,29	1,13	1.00	0,91	0,82	-
Capacidad de los programadores	1,42	1,17	1.00	0,86	0,70	-
Experiencia en S.O. utilizado	1,21	1,10	1.00	0,90	-	-
Experiencia en el lenguaje de programación	1,14	1,07	1.00	0,95	-	-
Prácticas de programación modernas	1,24	1,10	1.00	0,91	0,82	-
Utilización de herramientas software	1,24	1,10	1.00	0,91	0,83	-
Limitaciones de planificación del proyecto	1,23	1,08	1.00	1,04	1,10	-

Tabla 5.1: Cálculo de la variable FAE

A continuación calculamos el esfuerzo de desarrollo E:

$$E = a \cdot KLDC^{b} \cdot FAE = 3,2 \cdot 4,171^{1,05} \cdot 0,65 = 9,3 \text{ personas-mes} \tag{5.2}$$

El tiempo de desarrollo en base al esfuerzo que se obtiene es de:

$$T = c \cdot E \cdot d = 2,5 \cdot 9,3 \cdot 0,38 = 5,8 \text{ meses} \tag{5.3}$$

Por último, el personal requerido es:

$$P_{req} = \frac{Esfuerzo}{Tiempo} = \frac{9,3}{5,8} = 1,6 \text{ personal requerido} \tag{5.4}$$

De los resultados se deduce que el desarrollo de la aplicación necesita un trabajador a tiempo completo y otro a media jornada para ser llevada a cabo en unos 6 meses. En caso de contarse con un único trabajador, el desarrollo requiere algo más de 9 meses. Como se

ha comentado anteriormente, esta estimación no tiene encuentra el estudio previo para la realización del trabajo ni la generación de documentación.

El desarrollo real de la aplicación ha sido de 10 meses. A esto hay que añadir 3 meses de estudio previo y desarrollo de pequeños prototipos para valorar las distintas alternativas a seguir, un mes de documentación (aunque parte de ella se realizó durante el estudio previo) y un mes de vacaciones al superarse el año. Por ello, el gasto de personal para los 15 meses, y teniendo en cuenta que el coste anual de un trabajador capacitado para llevar a cabo este tipo de proyectos es de 22500€ a 29600€[1], supone entre 28125€ y 37000€.

5.2.2 Costes de *Software*

Aquí se debe contabilizar el coste de la adquisición de material utilizado, además de pequeño material de oficina y las licencias *software* utilizadas. El *software* necesario se puede ver en la Tabla 5.2.

Material Software	Descripción	Coste de licencias
Visual Studio 2010 – Professional Edition	Entorno de desarrollo.	549€
Windows 7 Professional	Sistema operativo	133,84€
OpenCV2.4.3	Librería de Visión por Computador	Gratis
CMake2.8	Programa usado para compilar OpenCV	Gratis
Visual Paradigm - Professional	Programa usado para la realización de diagramas de desarrollo	510€
TeXnicCenter	Programa utilizado para escribir la documentación	Gratis
Doxygen	Generador de documentación	Gratis
TortoiseSVN	Programa para el control de versiones	Gratis
Balsamiq Mockups	Programa para la generación de bocetos.	58€
	TOTAL	1.250,84€

Tabla 5.2: Material Software

5.2.3 Costes de *Hardware*

Aquí se tiene en cuenta todo el *hardware*, material físico, que se ha utilizado para la realización de este TFG (Tabla 5.3).

[1]Tabla de retribuciones de la UCLM. Rango de salarios entre las categoría Tercera-O-II y Segunda-O-II

Material Hardware	Coste
Ordenador con procesador Intel Core 2 Duo 2.00GHz, 2.00 GB de RAM	750€
Ordenador con procesador Intel Xeon 2GHz, 16.00 GB de RAM	2.500€
TOTAL	3.250€

Tabla 5.3: Material *Hardware*

CAPÍTULO 6

ARTEFACTOS GENERADOS

En este capítulo se presentan los artefactos obtenidos según la planificación descrita en el Capítulo 4. Se describen de manera detallada los resultados obtenidos en cada una de las iteraciones que se han llevado a cabo para la consecución del presente TFG.

6.1 ESTUDIO DE VIABILIDAD

Al comienzo de la realización de este trabajo se plantearon las posibles alternativas existentes para la consecución del mismo. Se analizaron las herramientas con las que se contaba y la forma en la que se podía abordar el problema de entrenar un clasificador para luego aplicarlo a la detección.

Se estudió la opción de la modificación de código de OpenCV, pero se decidió no utilizarlo porque su modificación suponía rehacer tanto código como codificar los algoritmos de nuevo, ya que la cascada Viola&Jones de OpenCV sólo soporta etapas con su tipo de algoritmos de Boosting y añadir etapas con otro tipo de clasificadores era costoso. Por este motivo, se optó por implementar los algoritmos desde cero.

Por otro lado, el uso de las características estadísticas de textura y la necesidad de que el código resultante pueda ser integrado junto con otros algoritmos de procesamiento de imagen también hacían poco viable el uso de la cascada de OpenCV.

En esta fase también se estudió que la metodología de desarrollo DSDM fuera el enfoque más adecuado ya que el equipo de desarrollo cuenta sólo con tres participantes.

Una vez realizado el estudio de viabilidad, se realiza el análisis del mismo a través de un *diagrama de casos de uso*, con la definición de los usuarios participantes y cada uno de los *casos de uso*.

6.2 DIAGRAMAS DE CASOS DE USO

Los *diagramas de casos de uso* dan una visión global de las funcionalidades que ofrece el sistema. Su objetivo es proporcionar una vista conceptual de la aplicación que se va a construir. Esta vista engloba, de manera general, las funcionalidades externas que ofrece la aplicación al usuario.

A la hora de plantear la estructura de la aplicación, se dividió el diagrama de casos de uso en tres: un diagrama para la funcionalidad que implementa el formulario de la aplicación donde se marcan las lesiones encontradas, un diagrama para el entrenamiento del clasificador en función de los parámetros elegidos y otro para la detección de lesiones. Las Figuras 6.1, 6.2 y 6.3 muestran los diagramas de casos de uso identificados.

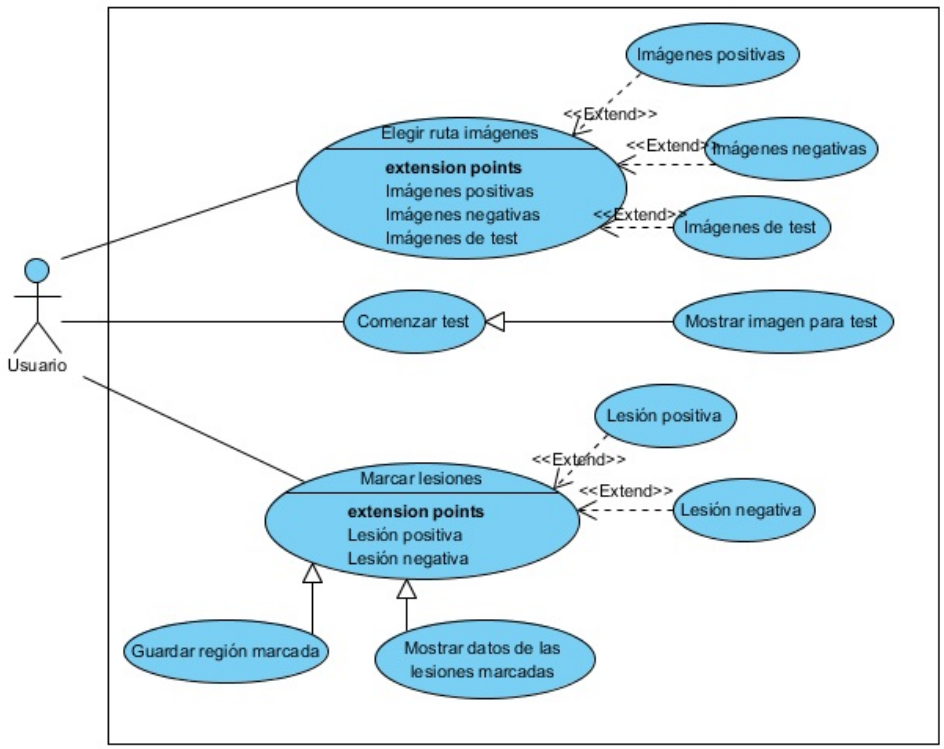


Figura 6.1: Diagrama de casos de uso relacionado con la extracción de muestras

Descripción de los Casos de Uso

Los diagramas de casos de uso suelen ir acompañados de una descripción de los mismos para su mejor comprensión. Esta descripción especifica lo que representa cada diagrama de Caso de Uso y proporciona información complementaria. A continuación se detallan los casos de uso de este TFG.

1. *Elegir ruta imágenes positivas:* Este caso de uso se encarga de abrir un cuadro de diálogo para seleccionar la ruta donde se encuentran las imágenes positivas y copiar

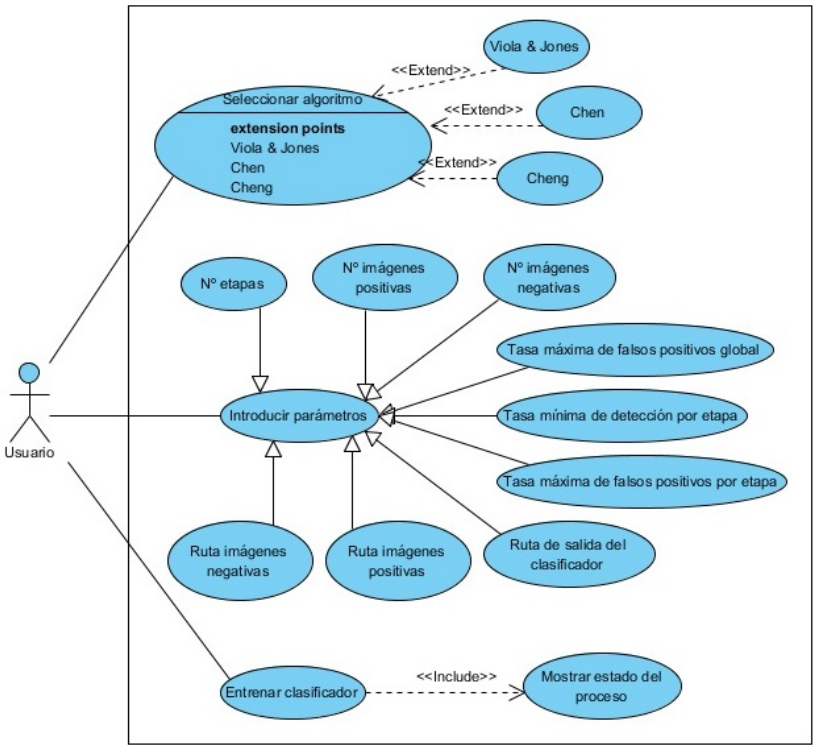


Figura 6.2: Diagrama de Casos de Uso relacionado con el Entrenamiento del Clasificador

la ruta en pantalla para que el usuario pueda comprobar que ha seleccionado la ruta correcta.

2. *Elegir ruta imágenes negativas:* Se encarga de abrir un cuadro de diálogo para seleccionar la ruta donde se encuentran las imágenes negativas y copiar la ruta en pantalla para que el usuario pueda comprobar que ha seleccionado la ruta correcta.

3. *Elegir imágenes para test:* Se encarga de abrir un cuadro de diálogo para seleccionar la ruta donde se encuentran las imágenes que se quieren analizar para añadir a una lista, vacía o ya existente, posibles regiones que el usuario considere que son positivas o negativas.

4. *Comenzar test:* Se encarga de mostrar la primera imagen contenida en la ruta seleccionada en el caso de uso anterior.

5. *Marcar lesiones:* Se encarga de recoger cada una de las lesiones que el usuario distingue en una imagen determinada. Cada vez que el usuario marca una región en una imagen, se abre un cuadro de diálogo para que especifique si esa zona que ha marcado es positiva o negativa. La región marcada se guarda en la carpeta correspondiente según su signo. Además se muestran en pantalla los datos de cada región marcada, a través de las coordenadas de su esquina superior izquierda, anchura, altura y signo.

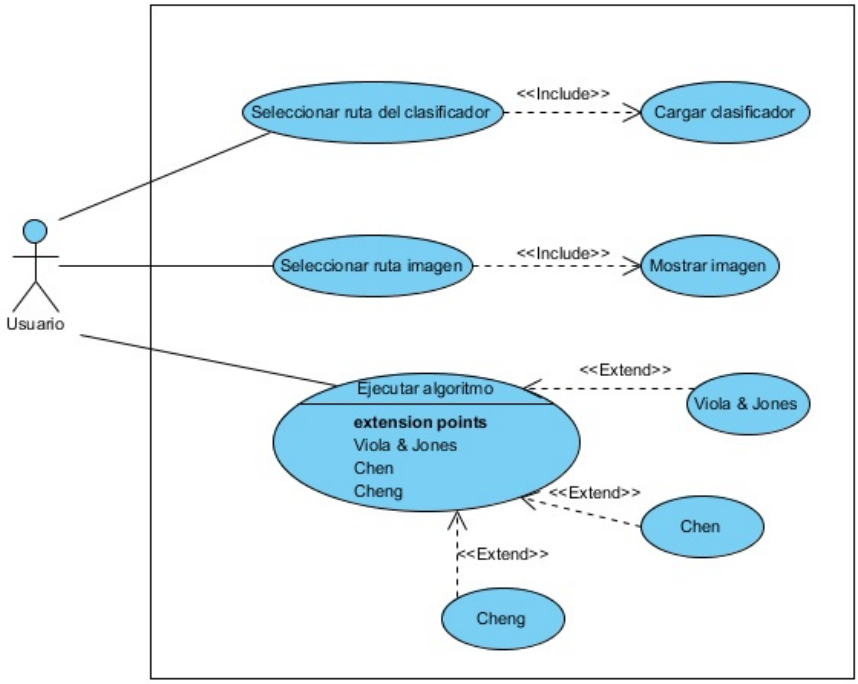


Figura 6.3: Diagrama de Casos de Uso relacionado con la Detección de Lesiones

6. *Seleccionar algoritmo:* El usuario debe seleccionar el algoritmo con el que quiere realizar el entrenamiento.

7. *Introducir parámetros:* Se encarga de procesar los parámetros que especifica el usuario. Estos parámetros son: número máximo de clasificadores débiles (sólo en el caso de Cheng&Jhan), número de imágenes positivas, número de imágenes negativas, tasa máxima de falsos positivos global, tasa mínima de detección por etapa, tasa máxima de falsos positivos por etapa, ruta del directorio de salida del clasificador y rutas de donde se encuentran las imágenes positivas y negativas respectivamente.

8. *Entrenar el clasificador:* Se encarga de aplicar un algoritmo de entrenamiento con unos determinados parámetros en función de las selecciones previas que haya hecho el usuario a través de los dos casos de uso anteriores.

9. *Seleccionar ruta del clasificador:* Abre un cuadro de diálogo para seleccionar la ruta donde se encuentra el archivo XML del clasificador.

10. *Seleccionar ruta imagen:* Abre un cuadro de diálogo para seleccionar la ruta de la imagen sobre la que se quiere aplicar el clasificador. Muestra la imagen en pantalla.

11. *Ejecución de algoritmos:* Se encarga de pintar sobre la imagen zonas que el clasificador considera que son positivas y, por tanto, que contienen lesión, en función del algoritmo que se haya utilizado: *Viola&Jones*, *Chen&Chen* o *Cheng&Jhan*.

6.3 DIAGRAMA DE CLASES

Los diagramas de clases tienen como objetivo detallar la estructura de un sistema mostrando sus clases, sus atributos y las relaciones entre clases. A partir de ellos se crea el diseño conceptual del sistema y se describen los componentes encargados del funcionamiento de éste y las relaciones entre los distintos componentes. Este tipo de diagramas definen, por tanto, una vista de bloques del sistema ofreciendo una base que facilita el desarrollo al programador.

En la Figura 6.4 se presenta el Diagrama de Clases de la aplicación con las clases que componen el sistema y sus relaciones. Estas clases están divididas en dos capas: dominio y presentación.

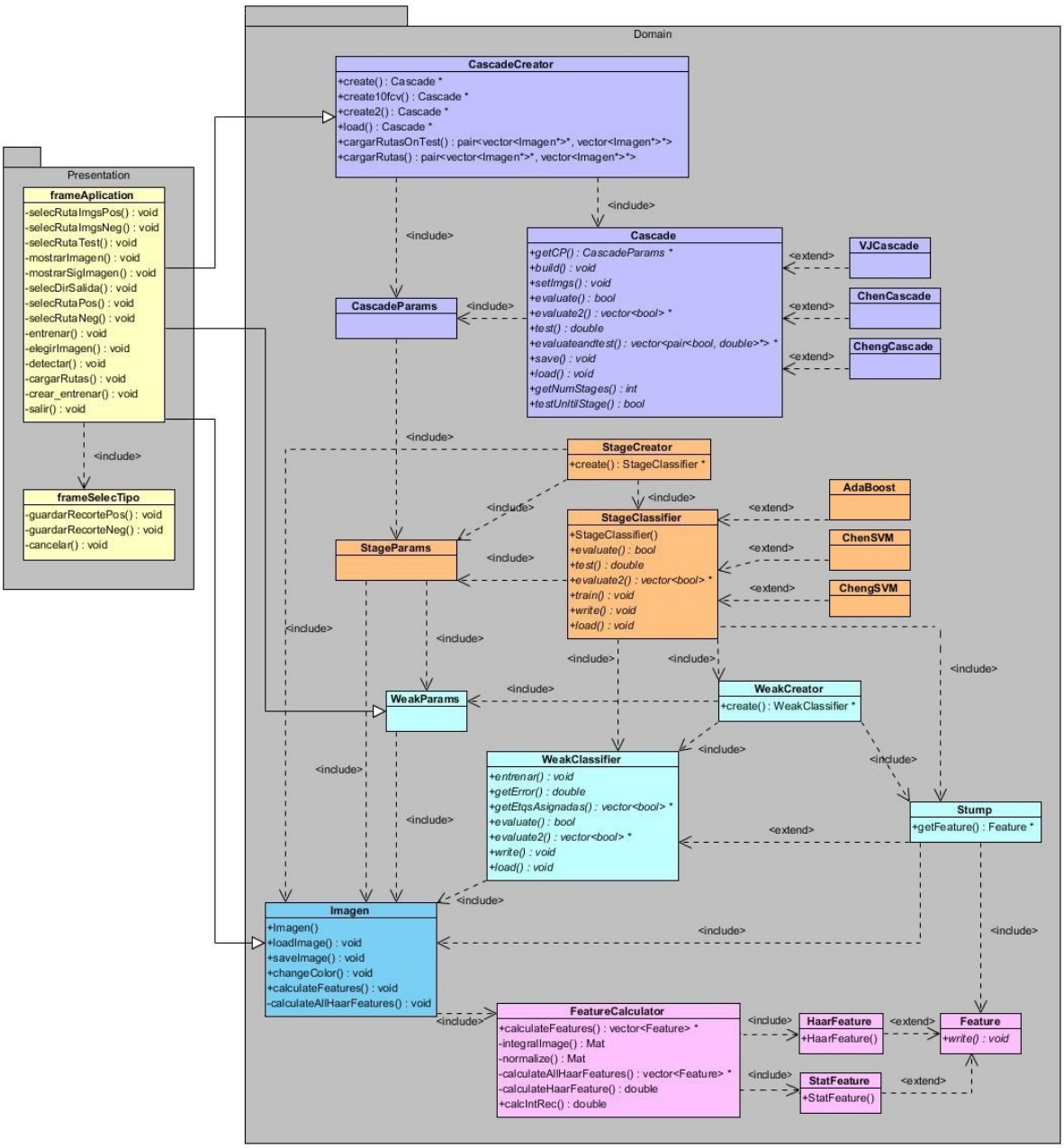


Figura 6.4: Diagrama de Clases del Sistema

6.4 DIAGRAMAS DE SECUENCIA

En un *diagrama de secuencia* se especifican las interacciones ordenadas entre objetos de forma temporal. Muestra los objetos y usuarios que se encuentran en un determinado escenario y la secuencia de mensajes intercambiados entre dichos objetos y/o usuarios para llevar a cabo la funcionalidad que la aplicación debe realizar en ese escenario. El objetivo de estos diagramas es validar las funcionalidades de los *casos de uso* y obtener una primera aproximación de la aplicación que se va a desarrollar.

Cada *caso de uso* puede estar formado por uno o más escenarios. Cada escenario representa un posible camino en la ejecución de la funcionalidad especificada en su respectivo *caso de uso*.

Las Figuras 6.5, 6.6 y 6.7 muestran algunos de los diagramas de secuencia que se utilizaron a lo largo del desarrollo de la aplicación.

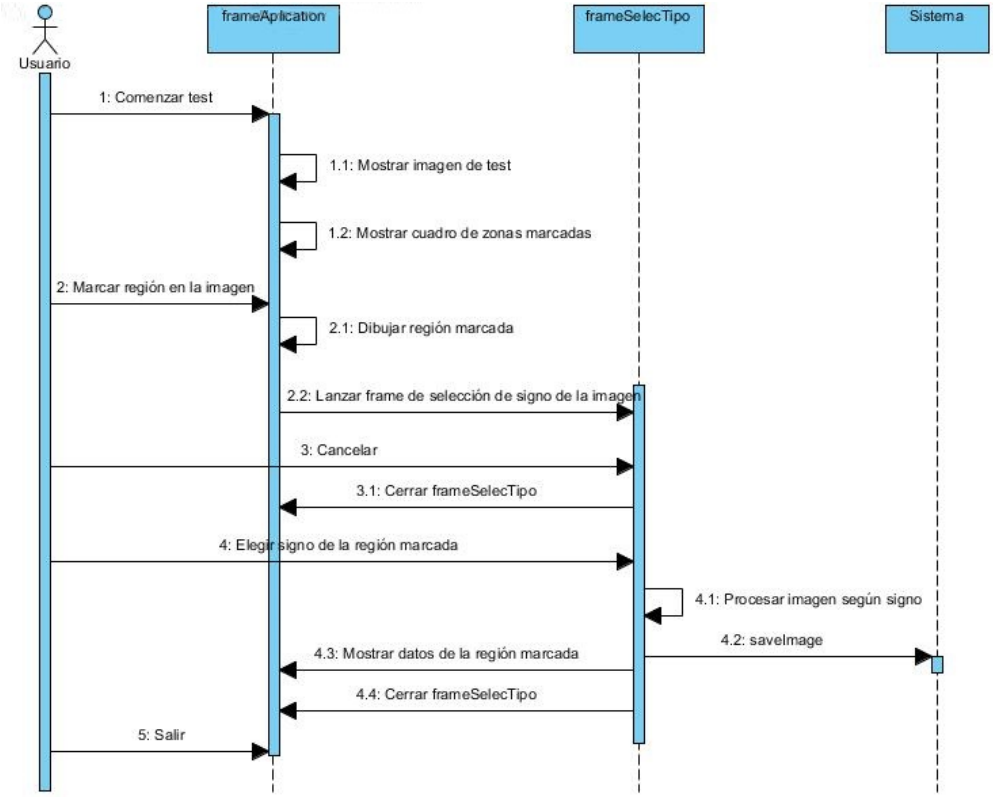


Figura 6.5: Diagrama de secuencia del caso de uso: *Marcar lesiones*

6.5 DISEÑO DE LA INTERFAZ GRÁFICA

A medida que ha ido avanzando el análisis y diseño de la aplicación, se han construido algunos bocetos, como los de la Figuras 6.8, 6.9 y 6.10 que han servido de referente para construir la interfaz gráfica de la aplicación.

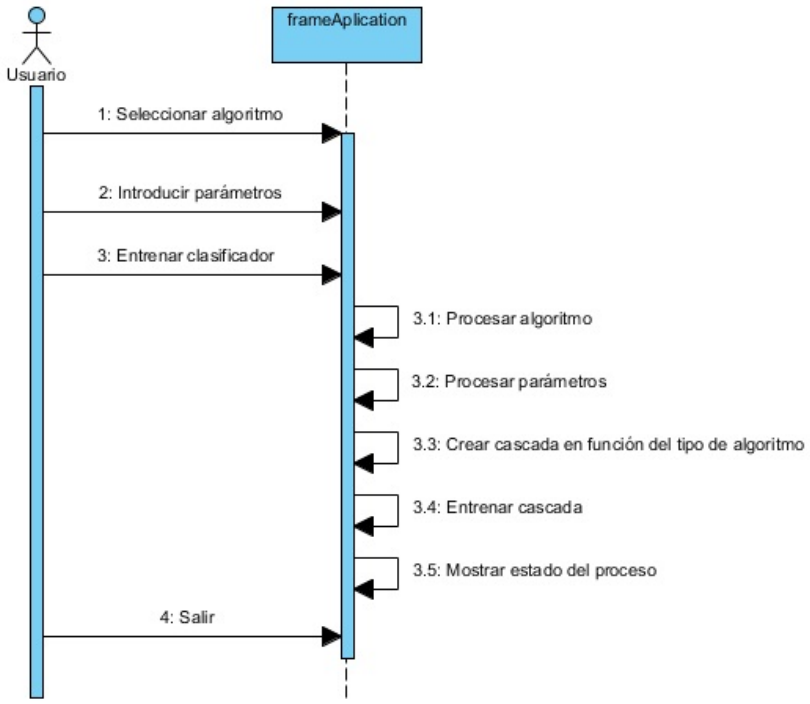


Figura 6.6: Diagrama de secuencia del caso de uso: *Entrenar clasificador*

6.6 PLANIFICACIÓN

Examinando cada uno de los prototipos definidos en el capítulo anterior y valorando la complejidad de cada una, se obtuvo una previsión de tiempo durante la planificación del TFG a través del diagrama de Gantt[1] mostrado en la Figura 6.11.

En la Figura 6.12 se ve el reparto temporal final en el que se ha realizado este trabajo.

[1]El objetivo del diagrama de Gantt es planificar el tiempo de dedicación previsto para diferentes tareas o actividades a lo largo de un tiempo total determinado

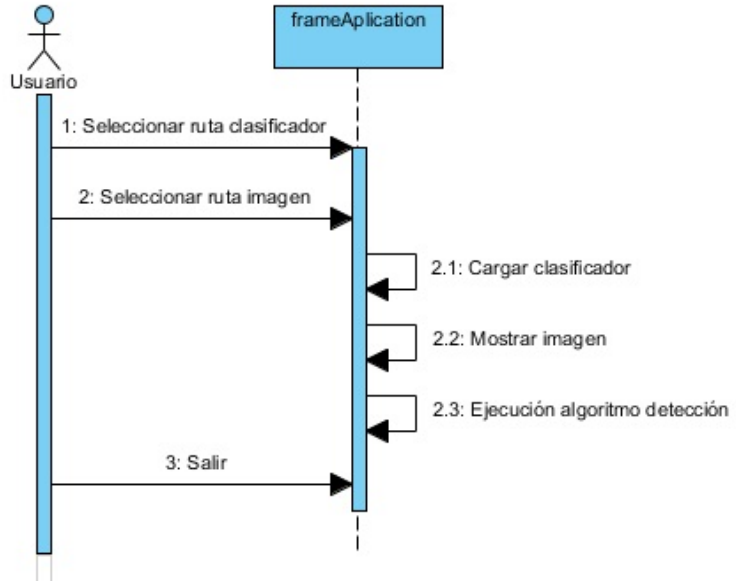


Figura 6.7: Diagrama de secuencia del caso de uso: *Ejecutar algoritmo*

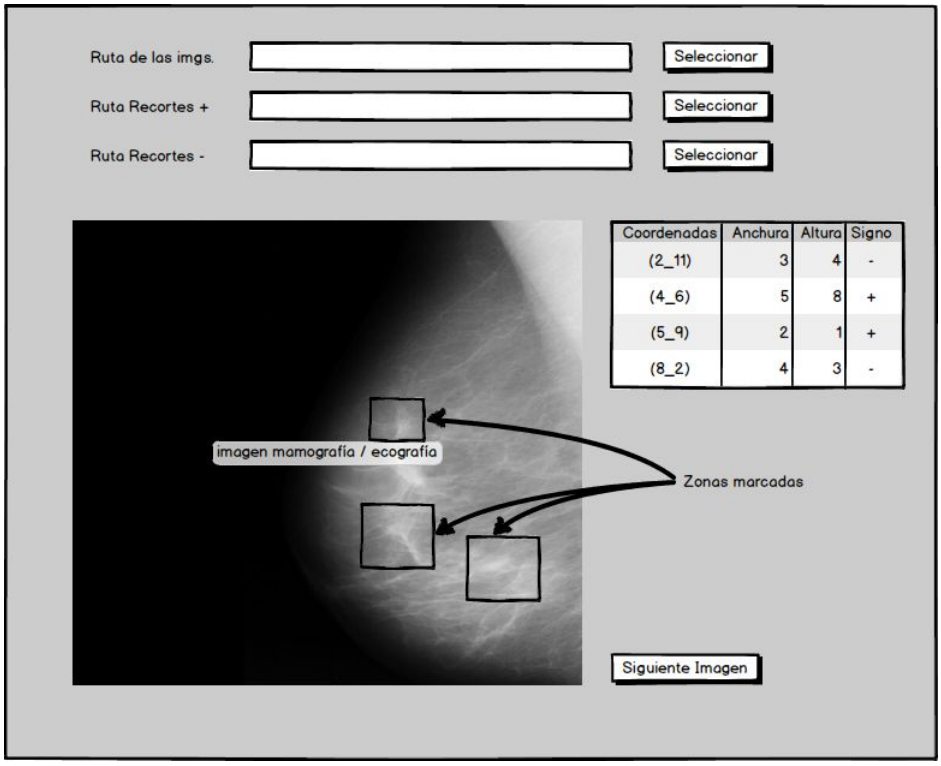


Figura 6.8: Boceto de la interfaz gráfica referente a la *Extracción de Muestras*

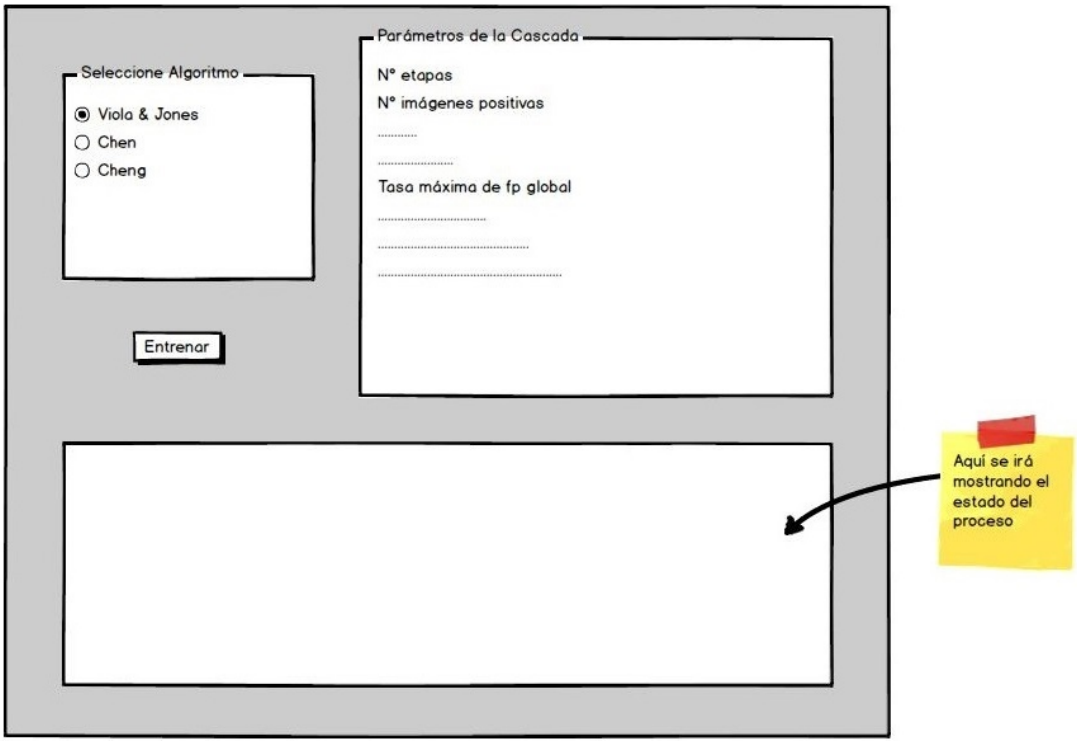


Figura 6.9: Boceto de la interfaz gráfica referente a la *Pruebas de los Algoritmos*

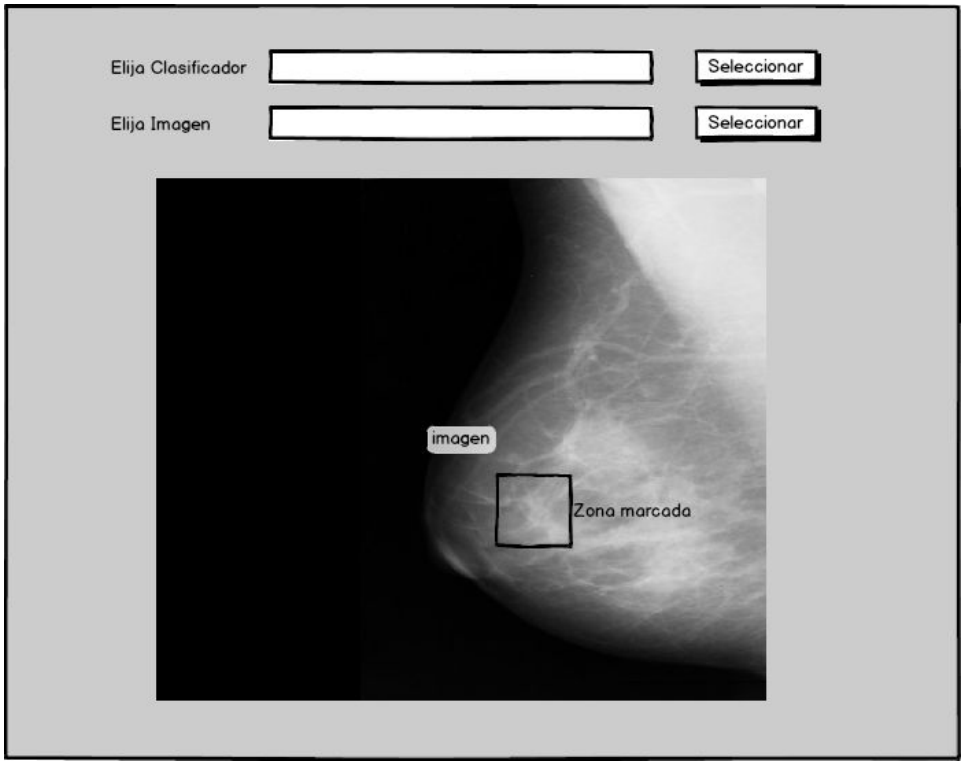


Figura 6.10: Boceto de la interfaz gráfica referente al *CAD*

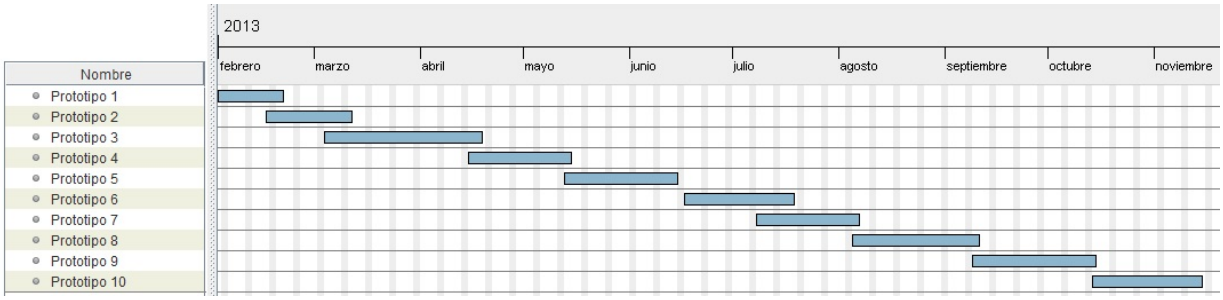


Figura 6.11: Diagrama de Gantt con los tiempos previstos

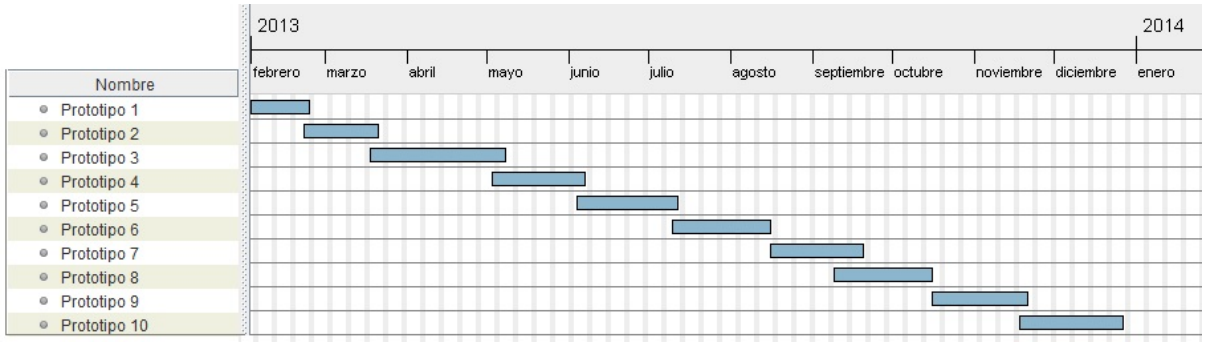


Figura 6.12: Diagrama de Gantt reales

CAPÍTULO 7

DESARROLLO DE LA APLICACIÓN

7.1 INTEGRACIÓN ALGORITMOS-INTERFAZ

Una vez implementados los algoritmos y teniendo una primera aproximación de lo que es la interfaz de usuario final, la manera de vincular ambas partes es añadir en las clases de la interfaz las cabeceras de los archivos de los algoritmos que se requieren en cada caso. Pero, en caso de que dos clases dependan entre sí, se originarían problemas de inclusiones cíclicas ya que una clase A incluiría a una clase B y B a su vez incluiría a A que de nuevo incluiría a B y así sucesivamente. Esto sucede entre las clases principales de las capas de presentación y dominio ya que es necesario enviar los parámetros desde la presentación al dominio y recibir de este el estado del proceso de ejecución. Para resolver este problema, se ha utilizado una clase auxiliar (*Consola.h*) que actúa de intermediaria. A continuación se muestra un pequeño extracto de ella:

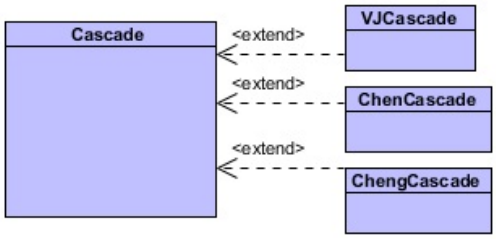
```
1  public ref class Consola{
2  public:
3        static System::Windows::Forms::RichTextBox^  rtbConsola;
4        static void AppendText(System::String^ text){
5          rtbConsola->AppendText(text+"\n");
6          rtbConsola->ScrollToCaret();
7          rtbConsola->Refresh();
8          }
9       //...
10       static void ChangeLast(System::String^ s){
11        array<System::String^>^ tempArray = gcnew
              array<System::String^>( rtbConsola->Lines->Length );
12        for(int i=0; i<tempArray->Length; i++)
13          tempArray[i] = rtbConsola->Lines[i];
14        tempArray[rtbConsola->Lines->Length-1] = gcnew
              System::String(s);
15        rtbConsola->Lines = tempArray;
16              rtbConsola->Refresh();
17         }
```

```
18 };
```

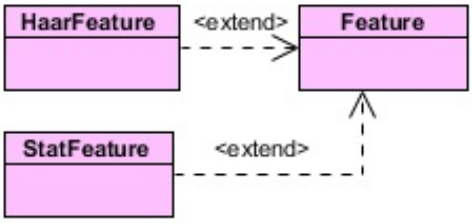
7.2 HERENCIA Y POLIMORFISMO

La herencia permite definir clases (subclases) a partir de otra clase genérica (superclase). La subclase posee todas las propiedades de la superclase, además de las suyas propias. La herencia favorece la reutilización de código, genera código más fiable y robusto y disminuye el coste de mantenimiento.

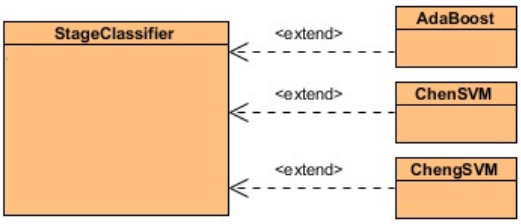
En la Figura 7.1 se pueden ver ejemplos de herencia aplicada a este trabajo:



(a) Herencia de la clase *Cascade*



(b) Herencia de la clase *Feature*



(c) Herencia de la clase *StageClassifier*

Figura 7.1: Ejemplos de Herencia implementados en la aplicación

En el primer ejemplo se puede observar que las clases *VJCascade*, *ChenCascade* y *ChengCascade* heredan de la clase *Cascade*. Por tanto poseen las propiedades de la clase *Cascade*, además de las suyas propias.

Por otro lado, el polimorfismo es la capacidad que poseen los lenguajes orientados a objetos de ofrecer acceso a métodos de clases diferentes bajo la misma orden. Por tanto, una llamada a un mismo método puede tener comportamientos distintos según sea la clase del objeto a través del que se llama.

En este trabajo se utiliza el polimorfismo, por ejemplo, al crear un clasificador. El clasificador devuelto es del tipo de la clase padre aunque haya sido creado con un tipo de cascada

en concreto, por tanto, cuando este clasificador es utilizado, bien para el entrenamiento o bien para evaluar una imagen, el método utilizado es el de la clase hija.

7.3 PROGRAMACIÓN MULTIHILO

Un hilo de ejecución, en sistemas operativos, es una característica que permite a una aplicación realizar varias tareas a la vez. Los distintos hilos de ejecución utilizan una serie de recursos conjuntamente como: el espacio de memoria, los archivos abiertos y la situación de la autenticación, entre otras.

Los hilos disponen de un estado de ejecución y pueden sincronizarse entre ellos para evitar problemas de compartición de recursos. Cada hilo está asociado a una tarea específica y determinada como forma de aumentar la eficiencia de uso del procesador.

En este trabajo se ha utilizado la programación multihilo en las clases *AdaBoost*, para mejorar el tiempo de entrenamiento del algoritmo, y *Aplicación*, para poder ejecutar los algoritmos de entrenamiento y la detección de lesiones en segundo plano y poder actualizar la salida de la aplicación en la interfaz de forma segura mediante dos enfoques diferentes: *C++ Windows Threads* y *BackgroundWorker* respectivamente.

7.3.1 *C++ Windows Threads*

Para mejorar el rendimiento del entrenamiento del clasificador *AdaBosst*, la función *CreateThread* crea una serie de subprocesos que calculan varios clasificadores *Stump* al mismo tiempo ya que una función puede lanzar múltiples hilos de ejecución dentro de ella. Todos los hilos de ejecución creados deben especificar la dirección inicial del código que van a ejecutar. La dirección de inicio típicamente es el nombre de una función definida en el código del programa. Esta función toma un sólo parámetro y devuelve un valor *DWORD*. En este caso, dicha función y la generación de los diferentes hilos de ejecución quedan como:

```
1   DWORD WINAPI trainWeak( LPVOID lpParam ) {
2     WeakClassifier* wc = (WeakClassifier*)lpParam;
3     wc->entrenar();
4     return 0;
5   }
6   //...
7   void AdaBoost::train(){
8     //...
9     for( int iter=0; iter<iters; iter++ ){
10          WeakClassifier* pDataArray[MAX_THREADS];
11          DWORD   dwThreadIdArray[MAX_THREADS];
12          HANDLE  hThreadArray[MAX_THREADS];
```

```
13            for( int thread=0; thread<MAX_THREADS; thread++ ){
14              int n = iter*MAX_THREADS+thread;
15              if(n<d){
16                pDataArray[thread] = (WeakClassifier*)
                      HeapAlloc(GetProcessHeap(),
                      HEAP_ZERO_MEMORY, sizeof(WeakClassifier));
17                if( pDataArray[thread] == NULL ) ExitProcess(2);
18                //...
19                WeakClassifier* hj = WeakCreator::create(wp);
20                pDataArray[thread]=hj;
21                hThreadArray[thread] = CreateThread(NULL, 0,
                      trainWeak, pDataArray[thread], 0,
                      &dwThreadIdArray[thread]);
22                if (hThreadArray[thread] == NULL) ExitProcess(3);
23              }else{
24                pDataArray[thread] = NULL;
25                hThreadArray[thread] = NULL;
26              }
27            }
28          }
29          WaitForMultipleObjects(MAX_THREADS, hThreadArray, TRUE,
              INFINITE);
30          //...
31      }
```

7.3.2 Trabajo en segundo plano para la clase *Aplicación*

La clase *BackgroundWorker* se utiliza para ejecutar una operación en un subproceso dedicado e independiente en *C++ Windows Forms*. La interfaz puede dejar de responder en operaciones que requieren mucho tiempo. La clase *BackgroundWorker* ofrece una solución en caso de necesitar una interfaz rápida cuando, por el contrario, se producen grandes retrasos debidos a tales operaciones. Además, esta clase permite la actualización de la información del proceso de entrenamiento de forma segura durante el acceso al cuadro de texto de la aplicación.

Para ejecutar una operación que requiera mucho tiempo, se debe crear un objeto *BackgroundWorker* y realizar oyentes de eventos que creen informes del progreso de la operación. La clase *BackgroundWorker* forma parte del cuadro de herramientas proporcionado por *Windows Forms*.

Para preparar una operación en segundo plano, hay que agregar un controlador de eventos al evento *DoWork*. A través de este controlador de eventos se ejecuta la operación que consume mucho tiempo. Para iniciar esta operación se llama a *RunWorkerAsync*. El evento

ProgressChanged se utiliza para recibir notificaciones de actualizaciones del progreso y el evento *RunWorkerCompleted* para recibir un aviso cuando finalice la operación.

El intercambio de datos con la interfaz se realiza a través de los eventos *ProgressChanged* y *RunWorkerCompleted*.

7.4 MODELO ARQUITECTÓNICO

Antes de comenzar a desarrollar una aplicación *software* de gran dimensión o con una amplia cantidad de funcionalidades, es necesario realizar un diseño previo. Existe una gran cantidad de diagramas que sirven de ayuda para modelar el diseño de la aplicación. Estos diagramas sirven de apoyo para que el programador entienda lo que quiere construir y suponen un punto de apoyo para las futuras modificaciones que el propio programador quiera hacer u otro desarrollador diferente.

A continuación se muestra el modelo arquitectónico del sistema, seguido de los diagramas de clases y de secuencia de cada caso de uso, para que sirvan de apoyo para una mejor comprensión de la interacción de los usuarios con cada caso de uso. Se va a realizar una vista arquitectónica del sistema a muy alto nivel de abstracción, pues cuanto antes se establezcan las bases del diseño menos problemas se tendrán durante el desarrollo. La primera visión arquitectónica se debe ir desarrollando a lo largo de las iteraciones en las que pueden surgir nuevas ideas con las que el diseño se va ampliando hasta completar su estructura final.

En la construcción de cualquier sistema *software* es necesario dividir el código en un conjunto de capas, este diseño se conoce con el nombre de *multicapa*. Cada una de estas capas tiene un conjunto de responsabilidades bien definidas que la diferencian de las demás capas.

Las principales ventajas que aporta esta forma de diseño son [30]:

- Simplifica la reutilización. Al realizar una división de la aplicación en diferentes capas, agrupando el código en función de unas determinadas funcionalidades similares, es posible reutilizar alguna de estas capas por separado en otras aplicaciones donde se necesite volver a implementar algo similar.
- Mejora la distribución de los recursos humanos. Se divide el trabajo en distintas áreas de conocimiento.
- Facilita el mantenimiento. Si se parte de un buen diseño, a la hora de modificar o añadir nuevas funcionalidades a las ya existentes se harán cambios sobre una determinada capa y no sobre la aplicación entera.

Este trabajo se encuentra dividido en dos capas:

- **Capa de presentación:** Esta capa reúne todos los aspectos del *software* que tienen que ver con las interfaces y la interacción con el usuario. Estos aspectos incluyen, entre otras cosas, el diseño y el manejo de las ventanas, el formato de las salidas, los menús, gráficos y elementos multimedia en general.

- **Capa de dominio:** En esta capa se desarrollan los aspectos del *software* que automatizan o apoyan los procesos de negocio que llevan a cabo los usuarios. Estos aspectos incluyen las tareas que forman parte de los procesos y las reglas y restricciones que se aplican. Esta capa también recibe el nombre de Capa Lógica de la Aplicación.

Estas capas son totalmente independientes dejando posible la ejecución por lotes de los algoritmos a través de un pequeño método principal.

La capa de persistencia es manejada por cada objeto en sí. Los clasificadores son almacenados como archivos XML (*Extensible Markup Language*). En estos archivos, cada clasificador escribe sus parámetros de configuración dejando que cada etapa escriba los suyos de forma anidadas. Así mismo, cada etapa invoca al método de escritura de los clasificadores débiles para que escriban sus parámetros. De esta forma, y gracias al polimorfismo (Sección 7.2) es posible, por ejemplo, añadir nuevos tipos de clasificadores de etapa o débiles sin necesidad de modificar el código de almacenamiento de resultados de la cascada o de la etapa. El Anexo A muestra un ejemplo de fichero XML generado.

A continuación se presentan los diagramas de clases y de secuencia. Estos diagramas están construidos con el Lenguaje Unificado de Modelado (UML). [9]

7.5 PATRONES DE DISEÑO

7.5.1 Fabricación parametrizada

Partiendo de una superclase con varias subclases, decimos que la fabricación parametrizada es el método a través del cual se crea una instancia de una subclase a través de un parámetro identificativo que se le pasa al constructor de la superclase.

Como se ha visto anteriormente, la construcción de cierto tipo de cascada basada en unas características concretas se realiza a través de la selección de una serie de parámetros. Por ello, el patrón de fabricación parametrizada ha sido elegido para la creación de los diferentes objetos del sistema.

A continuación se muestra un ejemplo de utilización de fabricación parametrizada en la clase *CascadeCreator*:

```
1  static Cascade* create(CascadeParams* cp)
2  {
3    //...
4    switch(cp->getCascadeType()){
5    case VIOLAJONES:
6      return new VJCascade(cp,imagenesp,imagenesn);
7    case CHEN:
8      return new ChenCascade(cp,imagenesp,imagenesn);
9    case CHENG:
10     return new ChengCascade(cp,imagenesp,imagenesn);
11   default:
12     Consola::AppendText("Not implemented");
13     return NULL;
14   }
15 }
```

En este ejemplo el parámetro identificativo es la variable *type*.

7.5.2 Patrón Proxy

El propósito del patrón Proxy es proporcionar un representante o sustituto de otro objeto para controlar el acceso a este. Con la utilización de este patrón se consigue minimizar el coste de crear e inicializar un objeto hasta que es realmente necesario.

Las clases que implementan este patrón ofrecen una serie de funcionalidades por medio de un conjunto de métodos estáticos. Este tipo de clases no pueden ser instanciadas.

En este trabajo, las clases *CascadeCreator*, *FeatureCalculator*, *StageCreator* y *WeakCreator* utilizan la fabricación parametrizada y el patrón Proxy.

7.6 PRUEBAS DE LA APLICACIÓN

El desarrollo de sistemas de *software* implica realizar una serie de tareas (desde la primera a la última etapa del ciclo de vida) en las que se puede cometer una serie de errores. Por ello, se debe incorporar una actividad que garantice la calidad del *software* final obtenido. Esta actividad se lleva a cabo en la etapa de prueba del *software*. La idea es incluir una serie de casos de prueba que cumplan:

- El caso de prueba debe ser un programa para descubrir un error.
- Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar un error no descubierto hasta entonces.

- Una prueba tiene éxito si descubre un error no detectado hasta entonces.

Sin embargo, es necesario tener en cuenta que las pruebas no pueden asegurar la ausencia de errores pero sí demostrar que existen defectos en el *software*.

7.6.1 Diseño de casos de prueba

Se deben diseñar pruebas con la mayor probabilidad de encontrar el mayor número de errores y que conlleven el mínimo esfuerzo y el mínimo gasto de tiempo. Cualquier producto *software* se puede probar de dos formas:

- Pruebas de caja negra: Las pruebas se realizan de forma que se comprueba que cada función sea operativa.
- Pruebas de caja blanca: Se desarrollan pruebas para asegurar que la operación interna se ajusta a las especificaciones además de comprobar el funcionamiento de los componentes internos.

En las pruebas de la caja negra, se presente demostrar que las funciones del *software* son operativas, es decir, que la entrada a cada función es tratada correctamente obteniéndose la salida esperada.

En las prueba de caja blanca, por el contrario, se realiza un examen minucioso de todos los detalles, comprobando los caminos lógicos del programa, los bucles y las condiciones además de examinar el estado del programa en varios puntos.

Por tanto, la realización de unas pruebas de caja blanca exhaustivas daría lugar a programas libre de errores. Sin embargo, realizar tal cantidad de pruebas prolongaría el desarrollo del *software*, por lo que se suele realizar el estudio de los caminos lógicos importantes.

Para realizar las pruebas de la aplicación desarrollada en este trabajo se siguieron ambos enfoques:

- Pruebas de caja negra: Se ha desarrollado un conjunto de imágenes sintético a partir de 10 pequeñas matrices de 7x5 pixels de las cuales 6 imágenes estaban etiquetadas como negativas y 4 como positivas. Con este pequeño conjunto se calcularon las salidas que se debían obtener durante el desarrollo de cada función de la aplicación comparándola siempre con la salida obtenida por esta. Una vez que la funcionalidad es la deseada, se utiliza un conjunto de prueba con imágenes reales para comprobar que, aunque no es posible calcular las salidas a mano, estas siguen siendo coherentes. Cabe destacar que durante las pruebas con el conjunto real de datos se han encontrado algunos problemas adicionales tales como:

  - Fugas de memoria. El uso de punteros clásicos en C++ hace que el código resultante sea más rápido pero, por contra, el programador debe encargarse de liberar la memoria de forma manual cuando esta ya no sea necesaria.
  - División por 0 no controlada en el estudio del algoritmo *AdaBoost*. Cuando se ejecutó el código con el conjunto de imágenes reales se detectaron una serie de resultados erróneos del tipo *#IND*. Estos resultados se debían a la división por 0 durante el cálculo de los coeficientes α_i de *AdaBoost* cuando el error de clasificación de uno de los clasificadores débiles era 0. Este caso especial no aparecía contemplado en la bibliografía previamente consultada y se solucionó dividiendo por la precisión de la máquina ($DBL_EPSILON$) cuando el error era 0.

  Por otro lado, la idea de utilizar un número diferente de imágenes positivas y negativas, así como imágenes de ancho y alto diferentes, intenta encontrar problemas producidos al intercambiar erróneamente estas variables. Este tipo de pruebas también se han realizado sobre las funcionalidades de la interfaz de usuario de la aplicación.

- Pruebas de caja blanca: Debido a la complejidad de los algoritmos y el tiempo necesario para la realización de este tipo de pruebas, las pruebas de caja blanca se han realizado mediante el seguimiento del código sobre el caso sintético con el depurador de Visual Studio. Se ha comprobado que el código siga el flujo deseado en cada prototipo de la aplicación y que los valores de las diferentes variables y de las condiciones de los bucles sean siempre correctos.

CAPÍTULO 8

RESULTADOS

El resultado principal de este trabajo es una comparativa entre las diferentes variaciones del clasificador en cascada y cómo afectan estas en la detección de lesiones en ecografías y mamografías. En las secciones siguientes se analizan en detalle la comparativa mencionada anteriormente.

8.1 CONJUNTOS DE IMÁGENES

Para el estudio de los diferentes clasificadores se han utilizado dos conjuntos de imágenes de imágenes de la mama correspondientes a pruebas ecográficas y radiológicas respectivamente.

8.1.1 Ecografías

El conjunto de ecografías utilizado está formado por 194 imágenes obtenidas en colaboración con los médicos del servicio de radiología del Hospital General Universitario de Ciudad Real. A pesar de que el formato de estas imágenes es DICOM [20], todas han sido previamente recortadas y convertidas al formato PGM (*Portable Graymap*) ya que los ecógrafos incluyen información personal de las pacientes dentro de la propia imagen (Fig. 8.1).

Mientras que las imágenes originales tienen un tamaño de 800x600 pixels, las imágenes extraídas tienen un tamaño de alrededor de 500x500 pixels.

Por otro lado, todas las imágenes de este conjunto fueron marcadas por los radiólogos guardándose como resultado el contorno de las imágenes marcadas y la benignidad o malignidad de estas junto a otra información adicional.

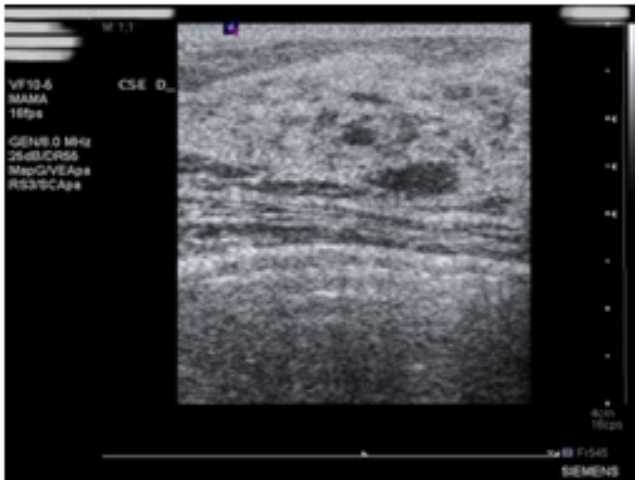


Figura 8.1: Ecografía con información de la paciente

8.1.2 Mamografías

El conjunto de mamografías utilizado es la base de datos mini-MIAS perteneciente a la Mammographic Image Analysis Society [32]. Está compuesto por 322 mamografías en papel que fueron digitalizadas, recortadas y centradas dando lugar a imágenes de 1024x1024 pixels (Fig. 8.2). Además de las imágenes, los autores proveen información adicional acerca de las lesiones encontradas en cada caso. Esta información viene organizada de la siguiente forma:

```
mdb001    G    CIRC    B    535    425    197

mdb002    G    CIRC    B    522    280    69

mdb003    D    NORM

mdb004    D    NORM

mdb005    F    CIRC    B    477    133    30

mdb005    F    CIRC    B    500    168    26

mdb006    F    NORM

mdb007    G    NORM

...
```

La primera columna indica el nombre del archivo de la imagen, la segunda el tipo de lesión encontrada, la tercera si la lesión es benigna o maligna y las tres últimas hacen referencia a las coordenadas (X,Y) del centro de la lesión y el radio de esta respectivamente.

8.2 ESTIMACIÓN DEL ERROR DE CLASIFICACIÓN

La mejor forma de comparar dos clasificadores es a través de su error. Los ratios de detección y falsos positivos están relacionados con el error de clasificación de forma que:

$$error = (1 - D_r) + FP_r \tag{8.1}$$

es decir, el error es la suma de los ratios de detecciones perdidas $(1 - D_r)$ y de falsos positivos (FP_r). Dado un conjunto de datos X con sus respectivas etiquetas (C) la forma más natural para calcular la estimación del error es ejecutar el clasificador *f(x)* sobre un conjunto de datos y hallar la proporción de muestras mal clasificadas (N_{error}) sobre el número total (n).

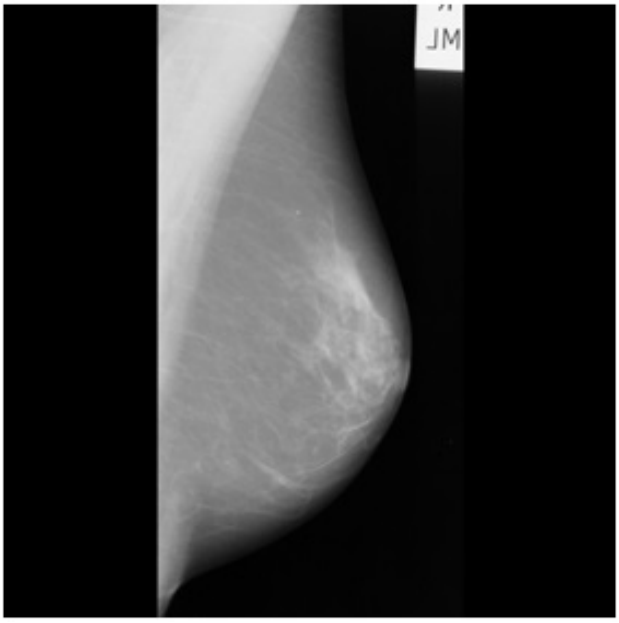


Figura 8.2: Mamografía

Es deseable contar con dos conjuntos de datos, uno para entrenar el clasificador y otro para estimar el error y que éstos contengan un elevado número de muestras. En la mayoría de las ocasiones no es posible obtener dos conjuntos de datos lo suficientemente grandes y se utilizan otros métodos para estimar el error. A continuación se detallan los métodos más utilizados para la estimación del error [22]:

Re-sustitución: Se diseña el clasificador entrenándolo sobre todas las muestras disponibles y se prueba con ellas también. En este caso, al probar el clasificador sobre el mismo conjunto que se utilizó para su creación la estimación del error demasiado optimista. Normalmente, este tipo de clasificadores no generaliza bien.

Hold-out: El conjunto de datos X se divide en dos partes, una para entrenar y otra para estimar el error. En este caso, se produce una estimación pesimista del error, sobre todo si no se dispone de gran cantidad de muestras.

Validación cruzada: Conocido también como *k-fold cross-validation* en inglés. Se elige un entero *k* y se divide X en *k* partes (*folds*), a ser posible de igual tamaño. Una vez dividido el conjunto de datos, se utiliza una de las partes para probar el comportamiento del clasificador y el resto, *k-1*, se utilizan como conjunto de entrenamiento. Este procedimiento es repetido *k* veces utilizando un conjunto de prueba diferente en cada caso y obteniéndose el error del clasificador como la media de los obtenidos en las *k* repeticiones. Cuando se utiliza *k=n* el método es llamado *leave-one-out*.

Por simplicidad y por la buena estimación que se obtiene del error se han realizado los entrenamientos y las pruebas de los clasificadores mediante una validación cruzada de 5 grupos (*5fcv*). Aunque para las pruebas se utilice uno de estos métodos es necesario tener en cuenta que el clasificador final se debe construir con todas las muestras disponibles.

8.3 PREPARACIÓN DE LOS DATOS DE ENTRENAMIENTO Y PRUEBAS

Una vez seleccionado el método de entrenamiento y prueba, es necesario preparar los dos conjuntos de imágenes de forma que se extraigan las regiones de interés además de una serie de regiones sin lesión que servirán como entrada a los clasificadores. En este punto, se debe recordar que la extracción de las características es más efectiva cuando se realiza sobre la parte concreta de la lesión y no sobre toda la imagen.

Las imágenes que contienen lesión se han extraído a partir de las coordenadas marcadas por los radiólogos en el caso de las ecografías y a partir de la información adjuntada por las mamografías del conjunto mini-MIAS.

Las imágenes negativas han sido extraídas de forma aleatoria del resto de zonas de las imágenes donde no había ninguna lesión.

La Tabla 8.1 muestra el número de imágenes obtenido para cada conjunto y para cada tipo.

Finalmente, las imágenes obtenidas para la comparativa han sido reescaladadas para poder obtener las mismas características Haar en todas ellas (70x70 pixels en el caso de las mamografías y 30x30 pixels en el caso de las ecografías).

8.4 RESULTADOS DE LOS CLASIFICADORES

Para el entrenamiento y las pruebas de los clasificadores se han establecido, además del método de validación cruzada de 5 grupos, los siguientes parámetros:

	Ecografías		Mamografías	
	Sin lesión	Con lesión	Sin lesión	Con lesión
fold 1	100	7	300	24
fold 2	100	8	300	23
fold 3	100	7	300	23
fold 4	100	8	300	24
fold 5	100	7	300	23
Total	500	37	1500	117

Tabla 8.1: Número de imágenes para cada conjunto y tipo

- Ratio mínimo de detección por etapa = *0,99*
- Ratio máximo de falsos positivos por etapa = *0,5*
- Ratio máximo de falsos positivos global = *0,01*

El Anexo C muestra los resultados detallados obtenidos por los clasificadores sobre cada grupo de validación. Para facilitar la comprensión de los datos obtenido se han obtenido las *matrices de confusión* de cada clasificador.

8.4.1 Matrices de confusión

Una matriz de confusión, también conocida como tabla de contingencia o matriz de error, es una tabla que permite la visualización, en este caso, de los resultados obtenidos por un clasificador. Cada fila de la matriz representa las muestras de una cierta clase, mientras que las columnas representan las muestras que han sido asignadas a una clase por el clasificador.

Cuando se trabaja con problemas de dos clases, la matriz de confusión consta de dos filas y dos columnas que representan el número de falsos positivos, falsos negativos, verdaderos positivos (detecciones) y verdaderos negativos (Tabla 8.2).

		Clases Asignadas	
		Positiva	Negativa
Clases Reales	Positiva	VP	FN
	Negativa	FP	VN

Tabla 8.2: Matriz de confusión para dos clases. VP - verdaderos positivos, VN - verdaderos negativos, FP - falsos positivos y FN - falsos negativos.

En las Tablas 8.3, 8.4, 8.5 y 8.6 se muestran las matrices de confusión de los resultados anteriores.

En vista de los resultados anteriores se pueden calcular los ratios de detección y falsos positivos para los conjuntos de imágenes. A continuación se muestran estos datos:

(a) Viola&Jones

		Clases Asignadas	
		Positiva	Negativa
Clases Reales	Positiva	34	3
	Negativa	6	494

(b) Chen&Chen

		Clases Asignadas	
		Positiva	Negativa
Clases Reales	Positiva	32	5
	Negativa	5	495

(c) Cheng&Jhan

		Clases Asignadas	
		Positiva	Negativa
Clases Reales	Positiva	17	20
	Negativa	5	495

Tabla 8.3: Matrices de confusión para ecografías con las características Haar

- Ecografías.
  - Haar
    - * Chen&Chen Dr=0,92 y FPr=0,012
    - * Cheng&Jhan Dr=0,85 y FPr=0,01
    - * Viola&Jones Dr=0,46 y FPr=0,01
  - Estadísticos de textura
    - * Chen&Chen Dr=0,78 y FPr=0,03
    - * Cheng&Jhan Dr=0,22 y FPr=0,002
    - * Viola&Jones Dr=0,35 y FPr=0,034
- Mamografías.
  - Haar
    - * Chen&Chen Dr=0,56 y FPr=0,07
    - * Cheng&Jhan Dr=0,25 y FPr=0,013
    - * Viola&Jones Dr=0,28 y FPr=0,02
  - Estadísticos de textura
    - * Chen&Chen Dr=0,56 y FPr=0,037
    - * Cheng&Jhan Dr=0,25 y FPr=0,02
    - * Viola&Jones Dr=0,03 y FPr=0,005

Para decidir qué clasificador obtiene mejores resultados en cada caso es necesario tener en cuenta que se busca un ratio de detección alto junto a un ratio de falsos positivos bajo. Por

(a) Viola&Jones

		Clases Asignadas	
		Positiva	Negativa
Clases Reales	Positiva	29	8
	Negativa	15	485

(b) Chen&Chen

		Clases Asignadas	
		Positiva	Negativa
Clases Reales	Positiva	8	29
	Negativa	1	499

(c) Cheng&Jhan

		Clases Asignadas	
		Positiva	Negativa
Clases Reales	Positiva	13	24
	Negativa	17	483

Tabla 8.4: Matrices de confusión para ecografías con las características estadísticas de textura

lo tanto, se puede observar que el mejor clasificador para ecografías es Viola&Jones junto a las características Haar, detectando correctamente el 92 % de las muestras con lesión y con sólo un 1,2 % de falsos positivos. En el caso de las mamografías, el mejor clasificador es Viola&Jones junto a las características estadísticas de textura, detectando el 56 % de las lesiones y dando un 3,7 % de falsos positivos.

Aunque se pueden observar grandes diferencias entre estos resultados obtenidos, los datos solo muestran un punto específico del régimen de funcionamiento de los clasificadores, par (D_r, FP_r), siendo necesario algún otro mecanismo para analizar estos resultados.

8.4.2 Curvas ROC

Las curvas ROC (del inglés Receiver Operating Characteristic) es la representación de la relación entre el ratio de verdaderos positivos (D_r) frente a la razón o ratio de falsos positivos (FP_R) según se varía el umbral de discriminación de los clasificadores [24]. En el caso de los clasificadores en cascada el umbral que se varía es el de la última etapa obtenida [45].

El análisis de la curva ROC, proporciona información para la selección del mejor clasificador independientemente del coste de la distribución de las dos clases sobre las que se decide. La curva ROC es también independiente de la distribución de las clases en la población (en el caso de la detección, la probabilidad de que exista el patrón buscado).

En la Figura 8.3 se puede ver la representación del espacio de una curva ROC. En ella, la diagonal que pasa por el punto B representa la curva obtenida por un clasificador que decida la salida de forma aleatoria (probabilidad(clase)=0.5). Si la curva pasa por el punto C el clasificador es peor que el clasificador aleatorio mientras que la del que pasa por el punto A

(a) Viola&Jones

		Clases Asignadas	
		Positiva	Negativa
Clases Reales	Positiva	64	53
	Negativa	102	1398

(b) Chen&Chen

		Clases Asignadas	
		Positiva	Negativa
Clases Reales	Positiva	29	88
	Negativa	20	1480

(c) Cheng&Jhan

		Clases Asignadas	
		Positiva	Negativa
Clases Reales	Positiva	33	84
	Negativa	31	1469

Tabla 8.5: Matrices de confusión para mamografías con las características Haar

es mejor. A su vez, el clasificador que obtenga una curva ROC que pase por el punto C será mejor que el que obtuvo una que pasaba por el A. Por ello, en una curva ROC el punto de operación ideal es la esquina superior izquierda donde la $D_r = 1$ y la $FP_r = 0$.

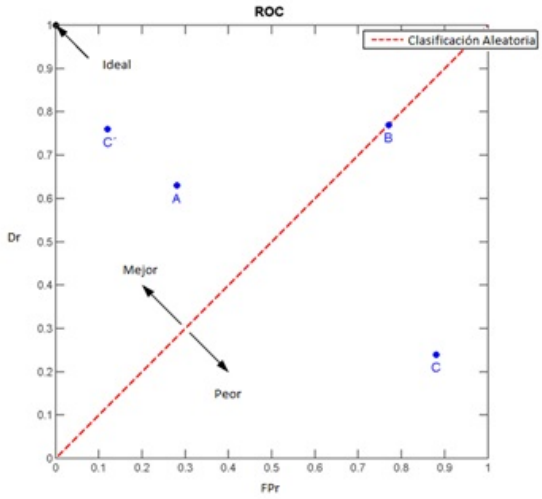


Figura 8.3: Representación del espacio de una curva ROC

En base a lo anterior, es posible obtener y comparar el área bajo la curva (cuyo acrónimo en inglés es AUC) de los distintos clasificadores como medida de lo bien o mal que realizan el trabajo de clasificación bajo todos los puntos de régimen de funcionamiento.

Para este caso, al utilizar validación cruzada de 5 grupos, el AUC resultante es la media de las 5 AUC de las curvas ROC obtenidas de cada grupo de validación. Las Tablas 8.7, 8.8, 8.9 y 8.10 muestran las AUC de las curvas ROC obtenidas (Ver Anexo D).

(a) Viola&Jones

		Clases Asignadas	
		Positiva	Negativa
Clases Reales	Positiva	66	51
	Negativa	56	1444

(b) Chen&Chen

		Clases Asignadas	
		Positiva	Negativa
Clases Reales	Positiva	29	88
	Negativa	30	1470

(c) Cheng&Jhan

		Clases Asignadas	
		Positiva	Negativa
Clases Reales	Positiva	4	113
	Negativa	7	1493

Tabla 8.6: Matrices de confusión para mamografías con las características estadísticas de textura

Algoritmo	fold	AUC	Media
Chen&Chen	1	0,849	
Chen&Chen	2	0,788	
Chen&Chen	3	0,904	0,880
Chen&Chen	4	0,935	
Chen&Chen	5	0,925	
Cheng&Jhan	1	0,2	
Cheng&Jhan	2	0,707	
Cheng&Jhan	3	0,153	0,436
Cheng&Jhan	4	0,969	
Cheng&Jhan	5	0,15	
Viola&Jones	1	0,906	
Viola&Jones	2	0,814	
Viola&Jones	3	0,936	0,921
Viola&Jones	4	0,969	
Viola&Jones	5	0,98	

Tabla 8.7: AUC de las curvas ROC de los clasificadores utilizando las características estadísticas de textura con las ecografías

Atendiendo a los resultados proporcionados por las curvas ROC, se puede observar que el mejor clasificador para ecografías es el de Viola&Jones con las características Haar obteniendo un AUC = 0,95. Para las mamografías, el clasificador en cascada que mejores resultados obtiene es el de Viola&Jones utilizando las características estadísticas de textura y obteniendo un AUC = 0,88.

Una vez comparados los resultados y conocidos ya los clasificadores en cascada junto con los parámetros que mejores resultados ofrecen para los dos conjuntos de imágenes, es posible crear los detectores con todas las imágenes disponibles.

Algoritmo	fold	AUC	Media
Chen&Chen	1	0,852	
Chen&Chen	2	0,859	
Chen&Chen	3	0,850	0,870
Chen&Chen	4	0,893	
Chen&Chen	5	0,895	
Cheng&Jhan	1	0,763	
Cheng&Jhan	2	0,771	
Cheng&Jhan	3	0,837	0,815
Cheng&Jhan	4	0,817	
Cheng&Jhan	5	0,888	
Viola&Jones	1	0,879	
Viola&Jones	2	0,835	
Viola&Jones	3	0,897	0,882
Viola&Jones	4	0,879	
Viola&Jones	5	0,919	

Tabla 8.8: AUC de las curvas ROC de los clasificadores utilizando las características estadísticas de textura con las mamografías

Algoritmo	fold	AUC	Media
Chen&Chen	1	0,923	
Chen&Chen	2	0,924	
Chen&Chen	3	0,927	0,952
Chen&Chen	4	0,995	
Chen&Chen	5	0,990	
Cheng&Jhan	1	0,923	
Cheng&Jhan	2	0,924	
Cheng&Jhan	3	0,919	0,770
Cheng&Jhan	4	0,429	
Cheng&Jhan	5	0,655	
Viola&Jones	1	0,923	
Viola&Jones	2	0,924	
Viola&Jones	3	0,927	0,952
Viola&Jones	4	0,999	
Viola&Jones	5	0,989	

Tabla 8.9: AUC de las curvas ROC de los clasificadores utilizando las características Haar con las ecografías

8.5 TIEMPOS COMPUTACIONALES

Los tiempos de ejecución de los algoritmos han sido medidos durante todo el proceso de obtención de los resultados descritos en este capítulo. Todos los tiempos fueron obtenidos con un ordenador Intel Xeon con 2.00GHz y 16GB de RAM y con una precisión de milisegundos. La Figura 8.4 muestra dichos tiempos de ejecución organizados por tipo de imagen, de características y de clasificador.

Algoritmo	fold	AUC	Media
Chen&Chen	1	0,783	
Chen&Chen	2	0,806	
Chen&Chen	3	0,891	0,831
Chen&Chen	4	0,828	
Chen&Chen	5	0,845	
Cheng&Jhan	1	0,494	
Cheng&Jhan	2	0,720	
Cheng&Jhan	3	0,750	0,629
Cheng&Jhan	4	0,633	
Cheng&Jhan	5	0,549	
Viola&Jones	1	0,730	
Viola&Jones	2	0,675	
Viola&Jones	3	0,888	0,792
Viola&Jones	4	0,844	
Viola&Jones	5	0,823	

Tabla 8.10: AUC de las curvas ROC de los clasificadores utilizando las características Haar con las mamografías

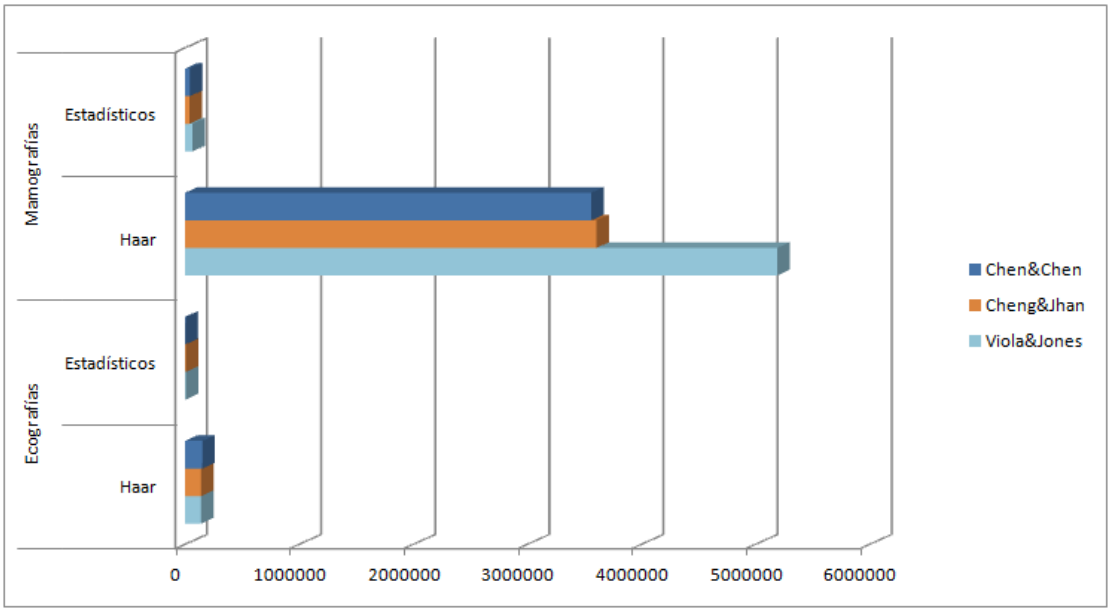


Figura 8.4: Representación gráfica de los tiempos computacionales

CAPÍTULO 9

CONCLUSIONES Y PROPUESTAS

En este capítulo se exponen las diferentes conclusiones obtenidas a partir del diseño, de la construcción y del uso de la aplicación además de una serie de propuestas para futuras ampliaciones.

9.1 OBJETIVOS ALCANZADOS

Este Trabajo Fin de Grado proporciona una herramienta para la ayuda en la investigación de detección de lesiones en ecografías y mamografías. Se ha integrado todo el proceso de clasificación de lesiones en una única herramienta a la vez que se han comparado tres métodos computacionales de clasificación automática enfocados a la detección del cáncer de mama junto con dos tipos de características diferentes: Haar y estadísticos de textura.

El estudio de estas técnicas es de gran utilidad porque representa un referente para los profesionales en la búsqueda de lesiones que muchas veces no son fáciles de distinguir por el ojo humano.

Aunque esté orientada a la detección de lesiones en ecografías y mamografías, se podrá utilizar, del mismo modo, para otros tipos de detección tales como detección de caras, detección de personas o la detección de otros tipos de lesiones cancerosas.

Atendiendo a los resultados proporcionados por las curvas ROC, se puede observar que el mejor clasificador para ecografías es el de Viola&Jones con las características Haar obteniendo un AUC = 0,95. Para las mamografías, el clasificador en cascada que mejores resultados obtiene es el de Viola&Jones utilizando las características estadísticas de textura y obteniendo un AUC = 0,88.

La herramienta generada permite la inclusión sencilla de nuevos algoritmos de detección o etapas de procesado previas para continuar mejorando los métodos propuestos. Asimismo, se debe destacar la facilidad de reutilización del código de la capa de dominio para su uso aparte en pruebas por lotes.

Finalmente, la primera pestaña de la herramienta desarrollada proporciona una interfaz sencilla para la preparación de los conjuntos de entrenamiento a partir de distintas bases de datos.

9.2 PROPUESTAS

En esta sección se describen las propuestas y posibles mejoras futuras para la ampliación de esta herramienta.

- Una de las ampliaciones propuestas es el uso del estándar DICOM para el manejo de imágenes ya que facilita el intercambio de archivos entre otros sistemas y la aplicación.
- Implementación y pruebas de otros algoritmos para obtener un resultado más preciso e íntegro.
- Otra mejora sería que, además de marcar automáticamente las lesiones de interés, se pudiera ofrecer una resolución sobre el tipo de lesión del que se trata.
- Por último, el sistema deberá contar con la aprobación de profesionales médicos para garantizar su correcto funcionamiento y la fiabilidad que ofrece.

9.3 CONCLUSIÓN PERSONAL

La realización del presente Trabajo Fin de Grado me ha permitido aplicar a un problema real la mayor parte de los conocimientos adquiridos durante mis años de estudio en la Escuela Superior de Informática.

Este trabajo me ha enseñado que la figura del Ingeniero Informático no se restringe solamente al área de la teoría de la información, de la ingeniería del software o de la teoría de autómatas, entre otras, el trabajo de un Ingeniero Informático es aplicable en multitud de áreas de conocimiento, como la medicina en este caso. La multitud de posibilidades que ofrecen estos estudios me ha permitido analizar cada uno de los problemas y elegir entre la solución más acertada en cada caso.

La realización de esta aplicación me ha planteado desafíos tales como adentrarme en el mundo de la Inteligencia Artificial y los sistemas de aprendizaje, antes desconocidos para mí, además de profundizar en la programación multihilo, de la que tenía conocimientos generales aprendidos en Sistemas Operativos y la aplicación de un nuevo lenguaje que no había utilizado anteriormente, C++.

En definitiva, estoy satisfecha de haber elegido la titulación de Ingeniería Informática como carrera universitaria, y espero poder seguir aplicando y mejorando mis conocimientos en los próximos años.

APÉNDICE A

FICHERO XML

A.1 ESTRUCTURA DEL FICHERO XML GENERADO

A continuación se muestra un ejemplo de fichero XML generado por la aplicación. Cabe destacar que la estructura que sigue es:

- Parámetros de la cascada.
- Lista de clasificadores de etapa.
  - Clasificador de etapa 0
    - Lista de clasificadores débiles.
      - Clasificador débil 0
      - ...
      - Clasificador débil K
  - ...
  - Clasificador de etapa N.
    - ...

A.1.1 Ejemplo fichero XML

```
<?xml version="1.0"?>
<opencv_storage>
<CascadeParams>
  <cascadeType>0</cascadeType>
  <maxStages>0</maxStages>
  <rutaSalida>
    "./ECO(final)/ViolaJones_0.99_0.01_-1/c1.xml"
  </rutaSalida>
```

```
9     <rutaPos>"./imagesECO/Positives"</rutaPos>
10    <rutaNeg>"./imagesECO/Negatives"</rutaNeg>
11    <numPos>37</numPos>
12    <numNeg>500</numNeg>
13    <rows>30</rows>
14    <columns>30</columns>
15    <ext>pgm</ext>
16    <globalMaxFalseRate>
17      1.0000000000000000e-002
18    </globalMaxFalseRate>
19    <caractType>0</caractType>
20    <haarinc>3</haarinc>
21    <StageParams>
22      <minHitRate>9.8999999999999999e-001</minHitRate>
23      <maxFalseAlarm>5.0000000000000000e-001</maxFalseAlarm>
24    </StageParams>
25  </CascadeParams>
26  <VJCascade>
27    <StageClassifiers>
28      <StageClassifier0>
29        <AdaBoost>
30          <threshold>3.2684587298444141e+000</threshold>
31          <weakclassifiers>
32            <weak0>
33              <alpha>4.0271138057285363e+000</alpha>
34              <Stump>
35                <umbral>-6.3896744391420370e+001</umbral>
36                <direction>-1</direction>
37                <ncaract>9116</ncaract>
38                <HaarFeature>
39                  <x>3</x>
40                  <y>4</y>
41                  <ww>21</ww>
42                  <wl>24</wl>
43                  <c>2</c>
44                </HaarFeature>
45              </Stump>
46            </weak0>
47            <weak1>
```

```
48              <alpha>3.2684587301420374e+000</alpha>
49              <Stump>
50                <umbral>-5.6896011332949556e+001</umbral>
51                <direction>-1</direction>
52                <ncaract>5039</ncaract>
53                <HaarFeature>
54                  <x>12</x>
55                  <y>4</y>
56                  <ww>24</ww>
57                  <wl>9</wl>
58                  <c>1</c>
59                </HaarFeature>
60              </Stump>
61            </weak1>
62          </weakclassifiers>
63        </AdaBoost>
64      </StageClassifier0>
65      <StageClassifier1>
66        <AdaBoost>
67          <threshold>2.4869046521760563e+000</threshold>
68          <weakclassifiers>
69            <weak0>
70              <alpha>2.4869046524506695e+000</alpha>
71              <Stump>
72                <umbral>-6.1830407320833473e+001</umbral>
73                <direction>-1</direction>
74                <ncaract>1328</ncaract>
75                <HaarFeature>
76                  <x>3</x>
77                  <y>8</y>
78                  <ww>18</ww>
79                  <wl>24</wl>
80                  <c>1</c>
81                </HaarFeature>
82              </Stump>
83            </weak0>
84            <weak1>
85              <alpha>2.6848668656465486e+000</alpha>
86              <Stump>
```

```
87                  <umbral>-2.9695262104246183e+000</umbral>
88                  <direction>-1</direction>
89                  <ncaract>26961</ncaract>
90                  <HaarFeature>
91                    <x>8</x>
92                    <y>23</y>
93                    <ww>6</ww>
94                    <wl>18</wl>
95                    <c>5</c>
96                  </HaarFeature>
97                </Stump>
98              </weak1>
99            </weakclassifiers>
100         </AdaBoost>
101       </StageClassifier1>
102     </StageClassifiers>
103 </VJCascade>
104 </opencv_storage>
```

APÉNDICE B

MANUAL DE LA APLICACIÓN

B.1 INSTALACIÓN

Para la instalación de la aplicación son necesarios los siguientes archivos:

- El ejecutable de la aplicación compilado para la arquitectura.
- Los archivos *.dll* compilados de OpenCV, o en caso de no tenerlos, se puede proceder a la descarga de la librería OpenCV en el ordenador donde se vaya a ejecutar el programa.
- Los archivos *.ico* con los iconos de la aplicación.

Teniendo los archivos mencionados, sólo sería necesario abrir el ejecutable (el archivo *.exe*) de la aplicación para que ésta funcione.

B.2 MANUAL DE USUARIO

B.2.1 Extracción de características

En primer lugar, cuando se ejecuta la aplicación, el usuario verá la siguiente pantalla (Fig. B.1):

Cabe destacar que en esta pantalla hay tres pestañas y en cualquier momento se puede seleccionar cualquiera de las tres dependiendo de cuál sea la funcionalidad que desea ejecutar.

En esta pantalla se deberá seleccionar la ruta de las imágenes a cargar, la de los recortes positivos y la de los recortes negativos. Una vez seleccionadas estas tres rutas se podrá comenzar con la extracción de muestras dándole al botón *COMENZAR*.

En el recuadro que en un principio aparecía en blanco, ahora debe aparecer la primera imagen de las contenidas en la ruta de las imágenes a cargar seleccionada previamente (Fig. B.2).

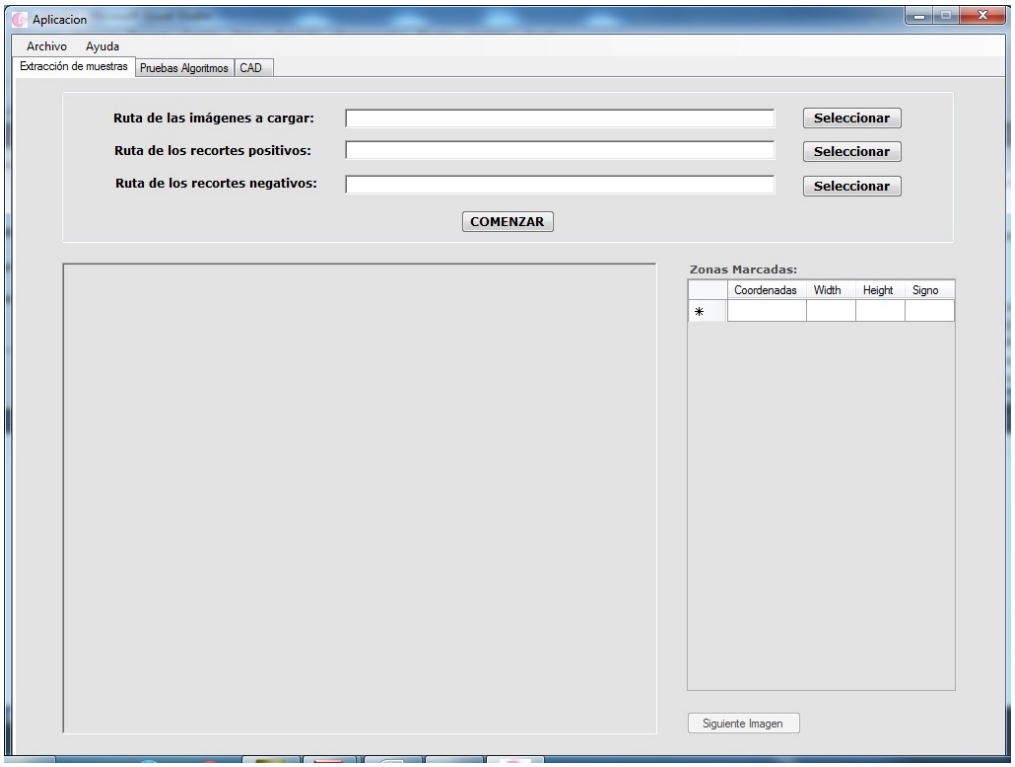


Figura B.1: Inicio de la aplicación

Una vez visualizada la imagen, si se detectan regiones que puedan ser positivas o negativas, se pueden seleccionar para añadirlas al repositorio de imágenes de entrenamiento. Para seleccionar una región en la imagen basta con pinchar con el ratón en un punto y arrastrarlo hasta que el recuadro que aparece contenga la región en cuestión. Cuando se levante el ratón, es decir, cuando se haya terminado de marcar la región que se deseaba, aparecerá un cuadro de diálogo donde se podrá seleccionar si se trata de un recorte positivo o negativo o si, de otro modo, se quiere cancelar la última acción y no guardar la última región que se había seleccionado (Fig. B.3).

Tanto si se selecciona que el recorte marcado es positivo como si es negativo, se guardará la subimagen seleccionada en su correspondiente ruta, de las marcadas al inicio y además aparecerá en la tabla de la derecha las características que definen a la subimagen, tales como: *coordenadas*, *anchura*, *altura* y *signo*.

Para pasar a extraer las muestras de otra imagen hay que pinchar en el botón *Siguiente imagen* que aparece abajo a la derecha.

B.2.2 Entrenamiento de los Algoritmos

En caso de que se desee entrenar el clasificador con cada uno de los algoritmos, habrá que seleccionar la pestaña *Pruebas Algoritmos* y aparecerá la siguiente pantalla (Fig. B.5):

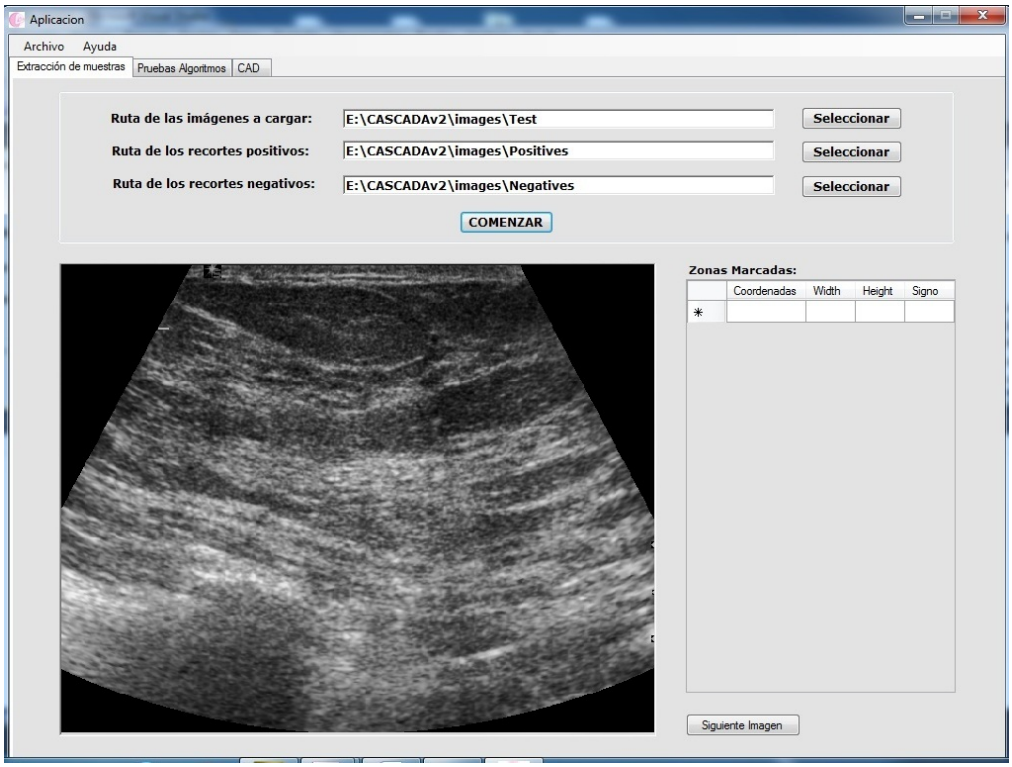


Figura B.2: Extracción de muestras

Se deberá seleccionar un algoritmo (*Viola&Jones*, *Chen&Chen* o *Cheng&Jhan*), unas características (*Haar-like* o *Estadísticas*) e introducir los siguientes parámetros: umbral de clasificadores (en caso de que se elija el algoritmo de *Cheng&Jhan*), número de imágenes positivas, número de imágenes negativas, tasa máxima de falsos positivos global, tasa mínima de detección por etapa, tasa máxima de falsos positivos por etapa, directorio de salida del clasificador, ruta de las imágenes positivas y ruta de las imágenes negativas.

Una vez completados todos estos parámetros se puede pulsar el botón *ENTRENAR* para que comience el entrenamiento del clasificador. Durante todo el entrenamiento se podrá ver en el cuadro de texto de abajo cuál es el estado del proceso (Fig. B.6).

Cuando se termina de ejecutar el algoritmo aparece en el cuadro de estado que se han guardado los resultados y el tiempo que se ha empleado en la ejecución como se puede ver en la figura B.7.

B.2.3 CAD

En caso de seleccionar la pestaña *CAD* para aplicar el sistema de detección de lesiones, el usuario verá como pantalla inicial la mostrada en la Figura B.8.

Se deberá seleccionar la ruta de un clasificador entrenado y de la imagen que se desea analizar. Para aplicar el clasificador elegido a esa imagen bastará con darle al botón *Detectar*.

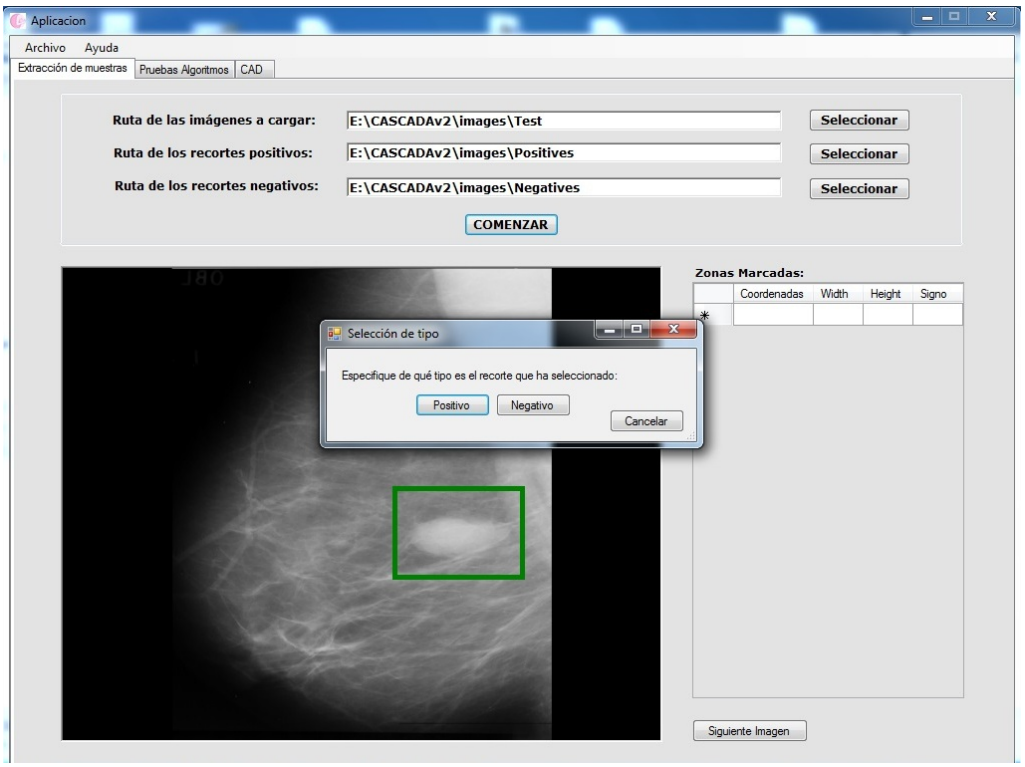


Figura B.3: Extracción de muestras

A continuación se podrá ver marcada la zona de la imagen donde existe una lesión, en caso de que la haya (Fig. B.8).

A lo largo del uso de la aplicación, se puede seleccionar en cualquier momento en el menú *Ayuda* la opción *Ver la ayuda*.

Si se desea terminar la aplicación se puede seleccionar en el menú *Archivo* la opción *Salir*.

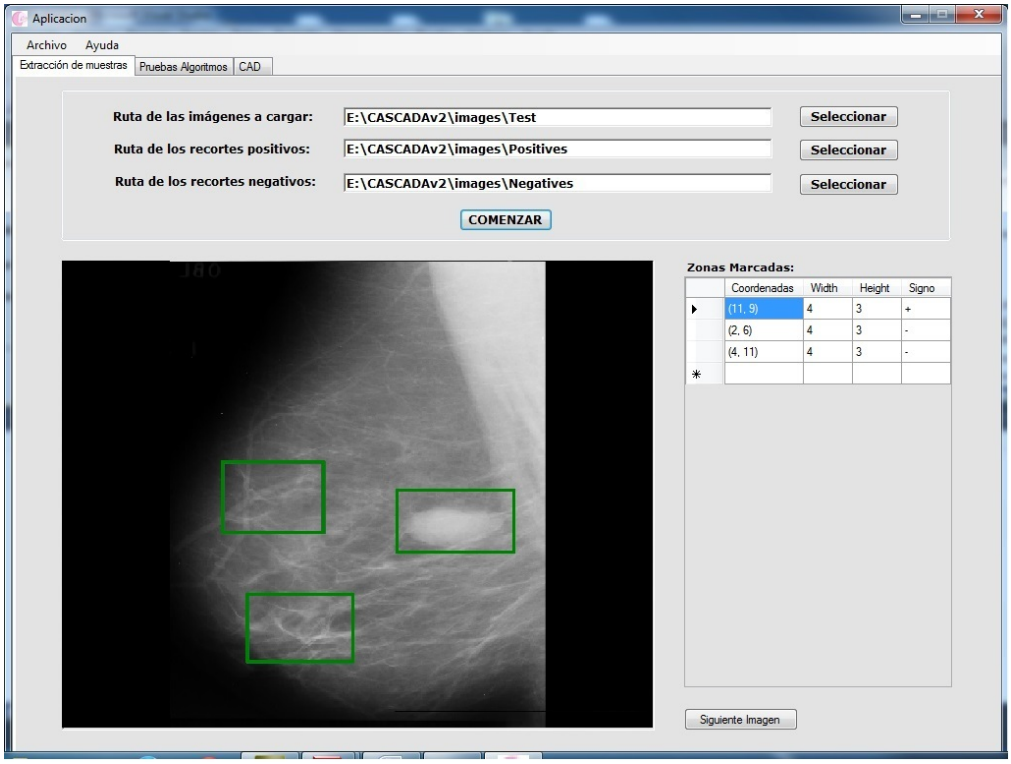


Figura B.4: Extracción de muestras

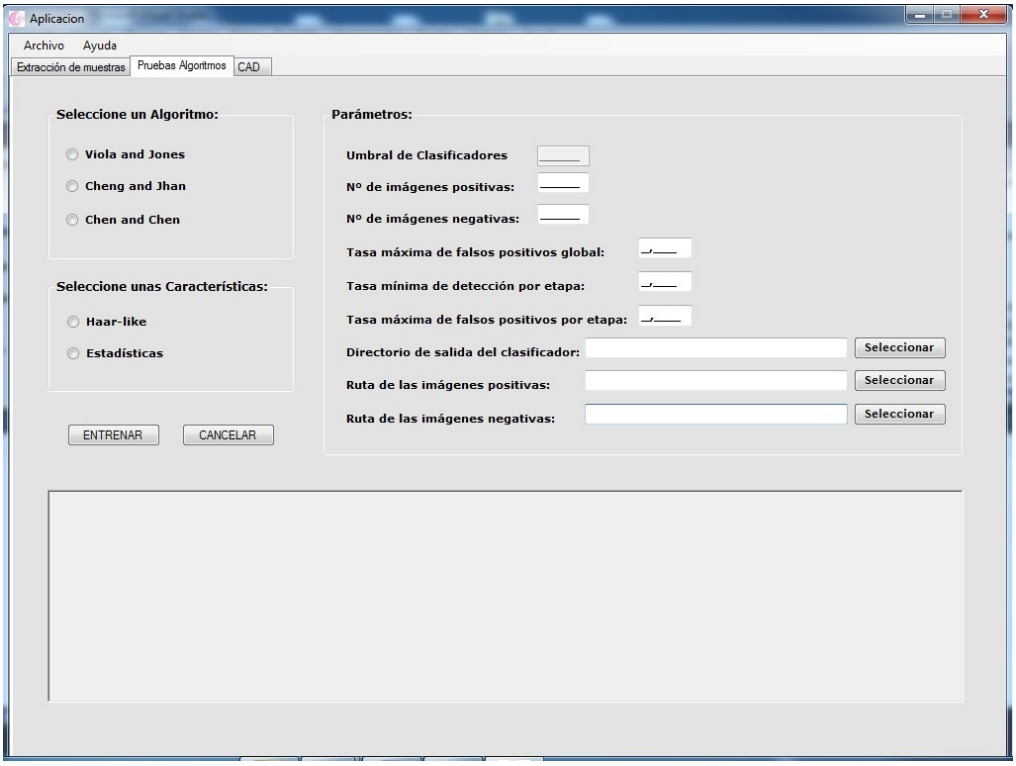


Figura B.5: Pruebas Algoritmos

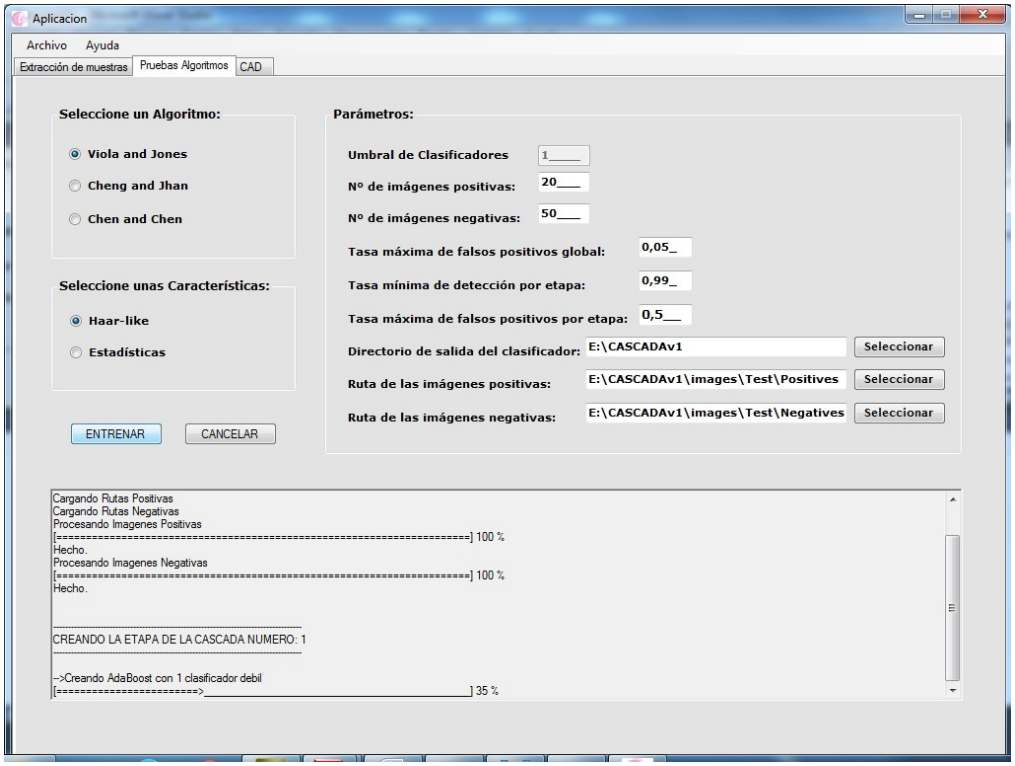


Figura B.6: Pruebas Algoritmos

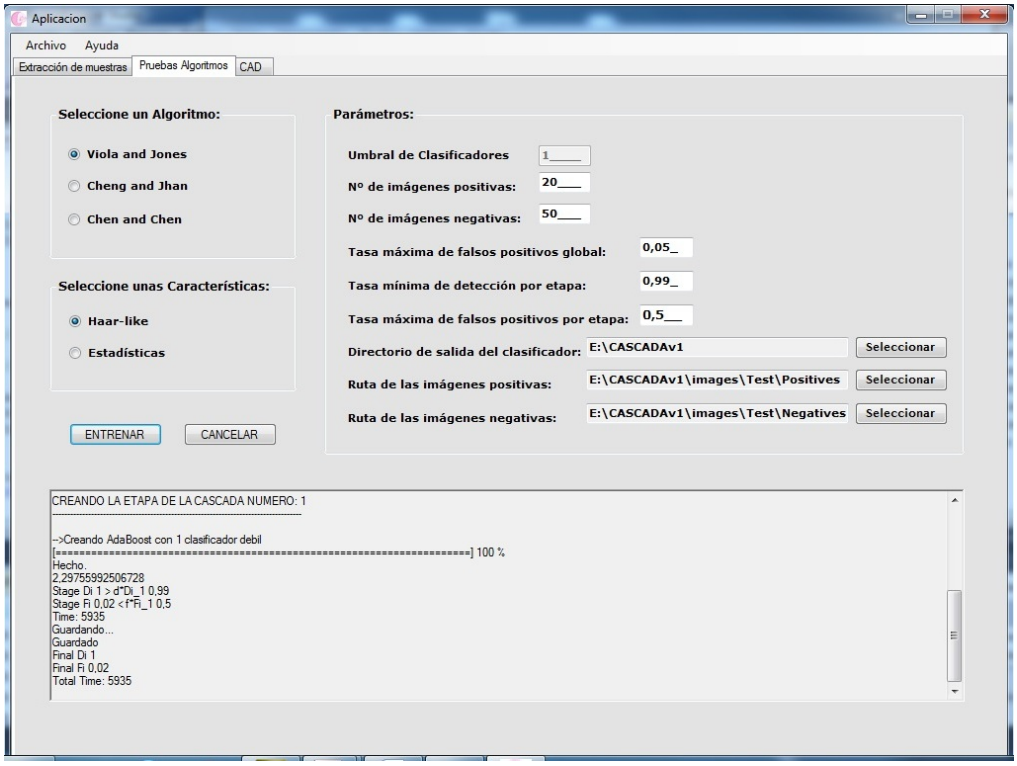


Figura B.7: Pruebas Algoritmos

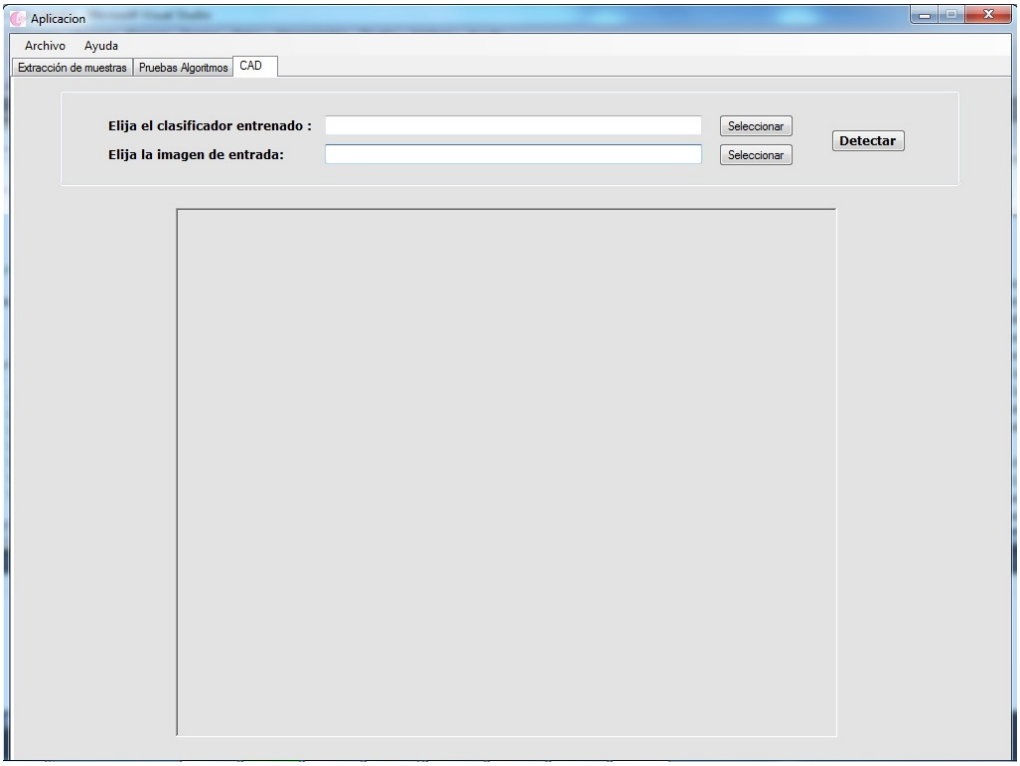


Figura B.8: CAD

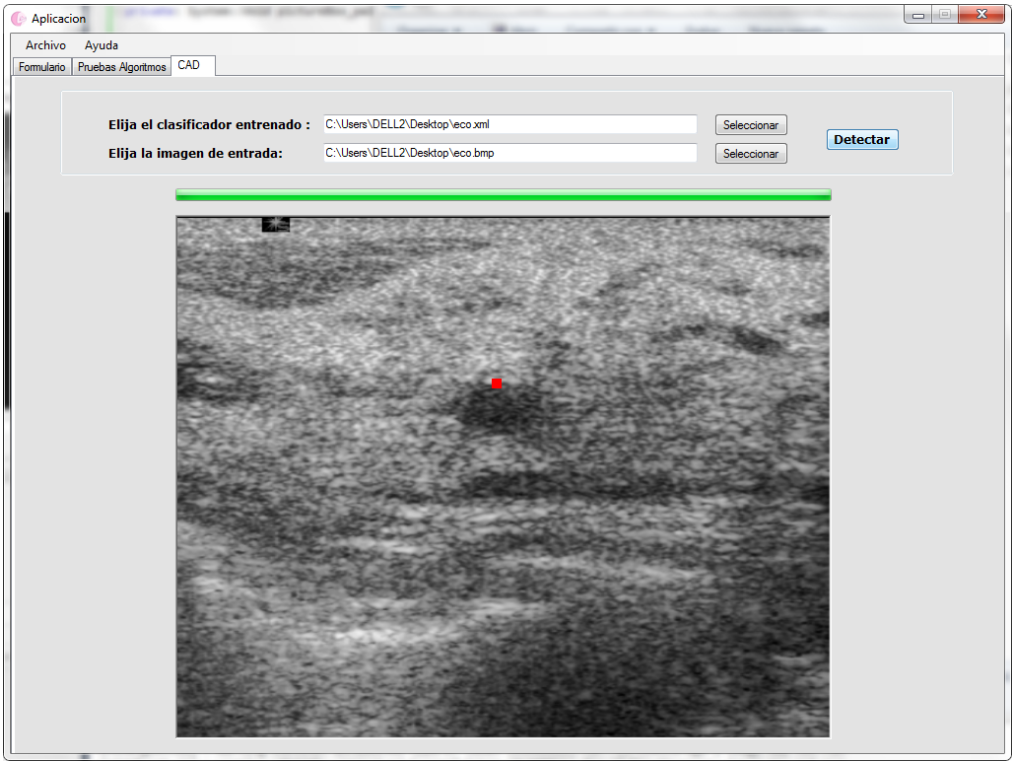


Figura B.9: CAD

APÉNDICE C

RESULTADOS DETALLADOS DE LOS CLASIFICADORES

A continuación se muestran los resultados obtenidos de las *5fcv* de los diferentes clasificadores. Las Tablas C.1 y C.2 muestran los resultados obtenidos por los clasificadores con las características estadísticas de textura para las imágenes de ecografía y mamografía respectivamente. De igual forma, las Tablas C.3 y C.4 muestran los resultados con las características Haar.

Algoritmo	fold	TestDr	TestFPr	Netapas
Chen&Chen	1	0,714	0,01	3
Chen&Chen	2	0	0	3
Chen&Chen	3	0	0	3
Chen&Chen	4	0,429	0	3
Chen&Chen	5	0	0	3
Cheng&Jhan	1	0,286	0,01	1
Cheng&Jhan	2	0	0,01	2
Cheng&Jhan	3	0,286	0,02	1
Cheng&Jhan	4	0,714	0,03	5
Cheng&Jhan	5	0,429	0	1
Viola&Jones	1	0,714	0,03	4
Viola&Jones	2	0,714	0,03	5
Viola&Jones	3	0,714	0,02	4
Viola&Jones	4	0,714	0,03	5
Viola&Jones	5	1	0,04	4

Tabla C.1: Resultado de los clasificadores con las características estadísticas de textura para las ecografías. *TestDr* - ratio de detección y *TestFPr* - ratio de falsos positivos.

Algoritmo	fold	TestDr	TestFPr	Netapas
Chen&Chen	1	0,174	0,017	3
Chen&Chen	2	0,522	0,067	3
Chen&Chen	3	0	0	3
Chen&Chen	4	0,130	0,003	3
Chen&Chen	5	0,435	0,017	3
Cheng&Jhan	1	0	0	2
Cheng&Jhan	2	0,130	0,017	2
Cheng&Jhan	3	0,043	0,007	3
Cheng&Jhan	4	0	0	2
Cheng&Jhan	5	0	0	3
Viola&Jones	1	0,565	0,063	6
Viola&Jones	2	0,435	0,09	6
Viola&Jones	3	0,696	0,073	6
Viola&Jones	4	0,609	0,057	6
Viola&Jones	5	0,522	0,05	6

Tabla C.2: Resultado de los clasificadores con las características estadísticas de textura para las mamografías. *TestDr* - ratio de detección y *TestFPr* - ratio de falsos positivos.

Algoritmo	fold	TestDr	TestFPr	Netapas
Chen&Chen	1	0,857	0,02	3
Chen&Chen	2	0,857	0,01	1
Chen&Chen	3	0,857	0	3
Chen&Chen	4	0,857	0	3
Chen&Chen	5	0,857	0,02	3
Cheng&Jhan	1	0,857	0,02	2
Cheng&Jhan	2	0,857	0,01	1
Cheng&Jhan	3	0,571	0,02	2
Cheng&Jhan	4	0	0	1
Cheng&Jhan	5	0	0	1
Viola&Jones	1	0,857	0,02	2
Viola&Jones	2	0,857	0,01	1
Viola&Jones	3	0,857	0	2
Viola&Jones	4	1	0,01	2
Viola&Jones	5	1	0,02	2

Tabla C.3: Resultado de los clasificadores con las características Haar para las ecografías. *TestDr* - ratio de detección y *TestFPr* - ratio de falsos positivos.

Algoritmo	fold	TestDr	TestFPr	Netapas
Chen&Chen	1	0,174	0,013	3
Chen&Chen	2	0,304	0,017	3
Chen&Chen	3	0,609	0,02	3
Chen&Chen	4	0,174	0,017	3
Chen&Chen	5	0	0	3
Cheng&Jhan	1	0,522	0,043	3
Cheng&Jhan	2	0,261	0,013	3
Cheng&Jhan	3	0,348	0,033	3
Cheng&Jhan	4	0,043	0	2
Cheng&Jhan	5	0,261	0,013	2
Viola&Jones	1	0,478	0,063	6
Viola&Jones	2	0,435	0,07	6
Viola&Jones	3	0,652	0,043	6
Viola&Jones	4	0,652	0,093	6
Viola&Jones	5	0,522	0,07	6

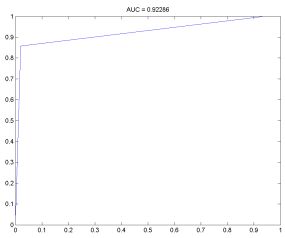
Tabla C.4: Resultado de los clasificadores con las características Haar para las mamografías. *TestDr* - ratio de detección y *TestFPr* - ratio de falsos positivos.

APÉNDICE D

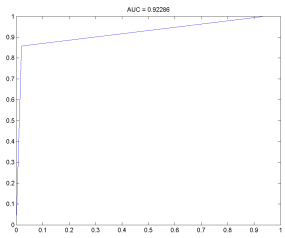
CURVAS ROC DE LOS CLASIFICADORES

D.1 RESULTADOS DEL ANÁLISIS DE LA CURVA ROC

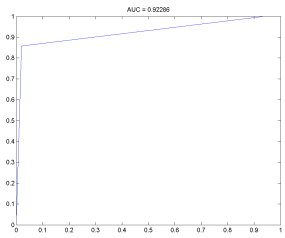
A continuación se muestran todas las curvas ROC halladas durante el proceso de entrenamiento y comparación de los clasificadores:



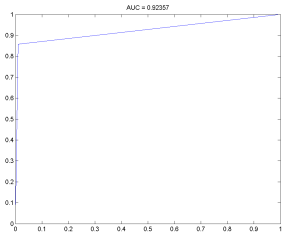
(a) Chen&Chen con *fold*=1



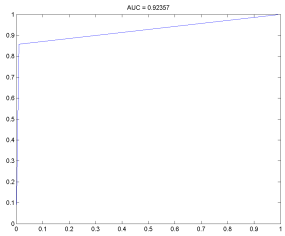
(b) Cheng&Jhan con *fold*=1



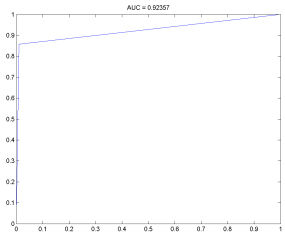
(c) Viola&Jones con *fold*=1



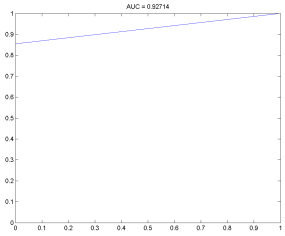
(d) Chen&Chen con *fold*=2



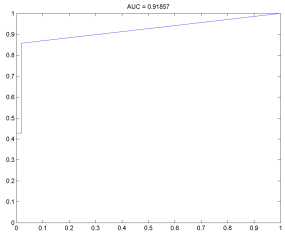
(e) Cheng&Jhan con *fold*=2



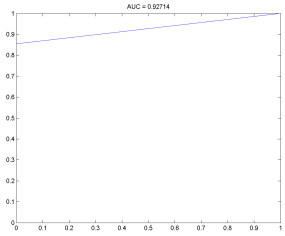
(f) Viola&Jones con *fold*=2



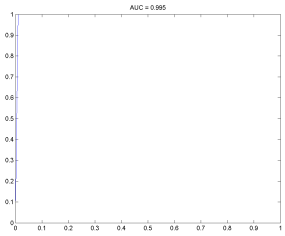
(g) Chen&Chen con *fold*=3



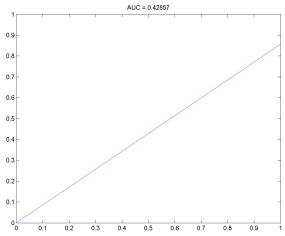
(h) Cheng&Jhan con *fold*=3



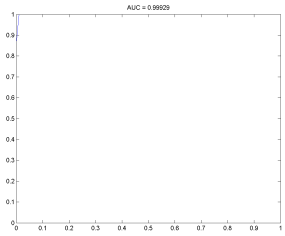
(i) Viola&Jones con *fold*=3



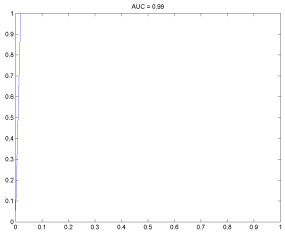
(j) Chen&Chen con *fold*=4



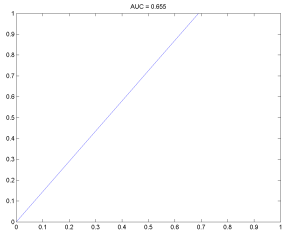
(k) Cheng&Jhan con *fold*=4



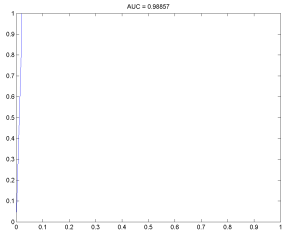
(l) Viola&Jones con *fold*=4



(m) Chen&Chen con *fold*=5

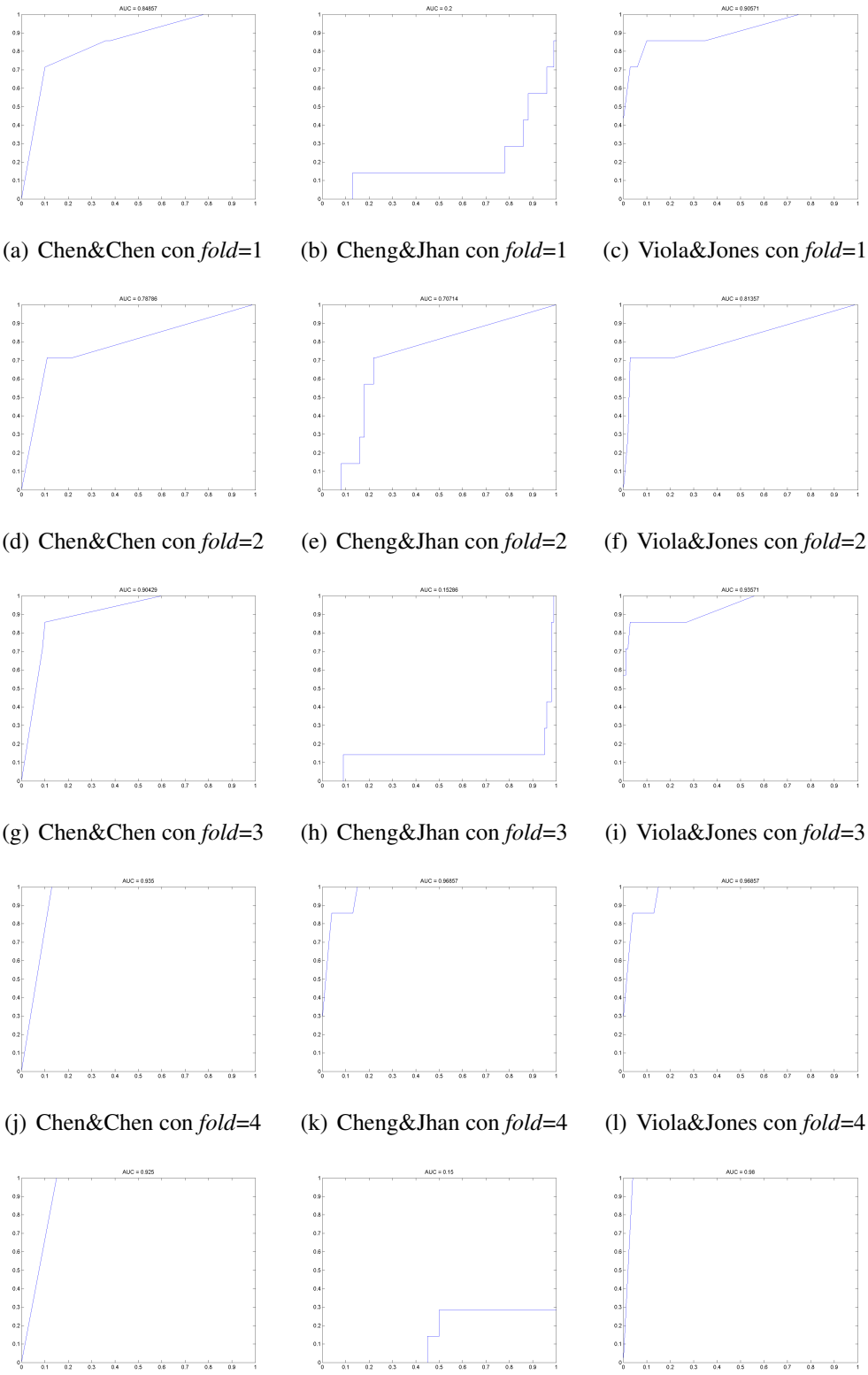


(n) Cheng&Jhan con *fold*=5



(o) Viola&Jones con *fold*=5

Figura D.1: Curvas ROC para las ecografías utilizando características Haar.



(a) Chen&Chen con *fold*=1 (b) Cheng&Jhan con *fold*=1 (c) Viola&Jones con *fold*=1

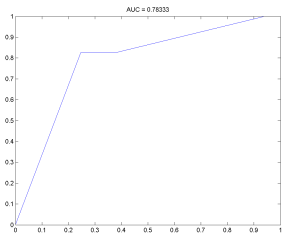
(d) Chen&Chen con *fold*=2 (e) Cheng&Jhan con *fold*=2 (f) Viola&Jones con *fold*=2

(g) Chen&Chen con *fold*=3 (h) Cheng&Jhan con *fold*=3 (i) Viola&Jones con *fold*=3

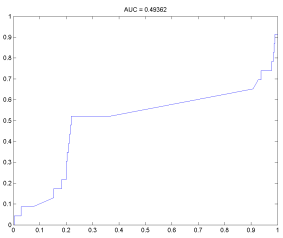
(j) Chen&Chen con *fold*=4 (k) Cheng&Jhan con *fold*=4 (l) Viola&Jones con *fold*=4

(m) Chen&Chen con *fold*=5 (n) Cheng&Jhan con *fold*=5 (o) Viola&Jones con *fold*=5

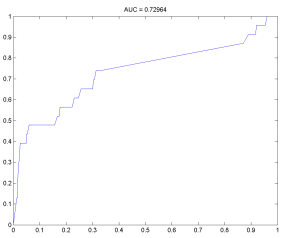
Figura D.2: Curvas ROC para las ecografías utilizando características estadísticas de textura.



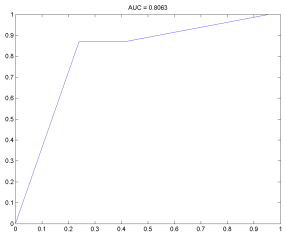
(a) Chen&Chen con *fold*=1



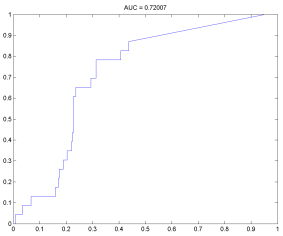
(b) Cheng&Jhan con *fold*=1



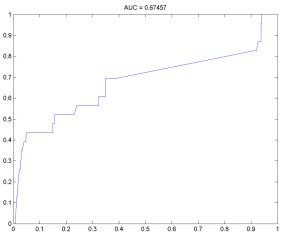
(c) Viola&Jones con *fold*=1



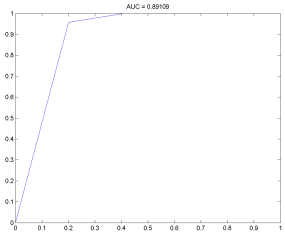
(d) Chen&Chen con *fold*=2



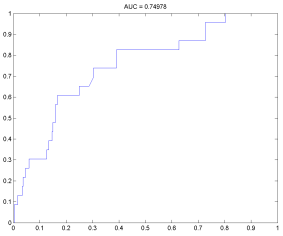
(e) Cheng&Jhan con *fold*=2



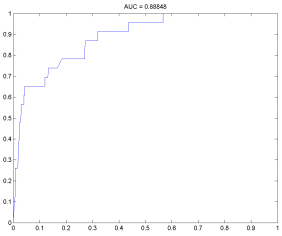
(f) Viola&Jones con *fold*=2



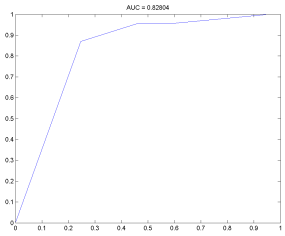
(g) Chen&Chen con *fold*=3



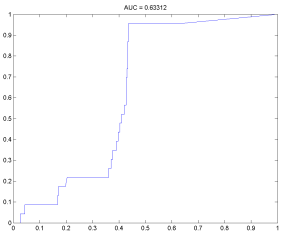
(h) Cheng&Jhan con *fold*=3



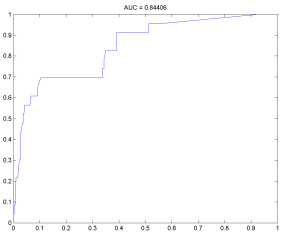
(i) Viola&Jones con *fold*=3



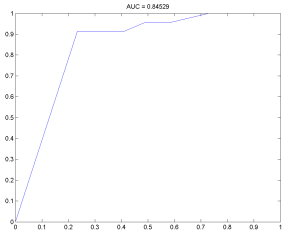
(j) Chen&Chen con *fold*=4



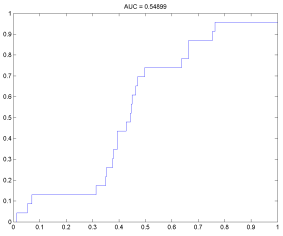
(k) Cheng&Jhan con *fold*=4



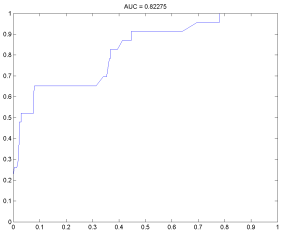
(l) Viola&Jones con *fold*=4



(m) Chen&Chen con *fold*=5

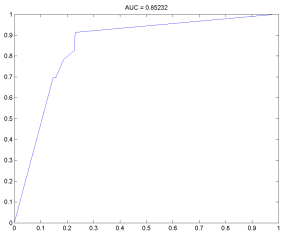


(n) Cheng&Jhan con *fold*=5

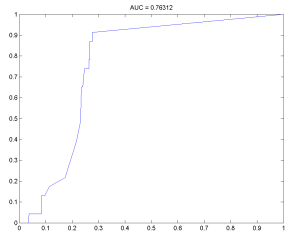


(o) Viola&Jones con *fold*=5

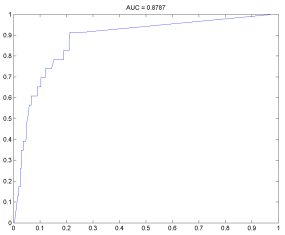
Figura D.3: Curvas ROC para las mamografías utilizando características Haar.



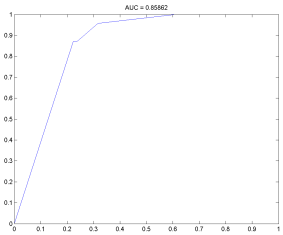
(a) Chen&Chen con *fold*=1



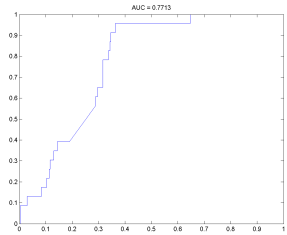
(b) Cheng&Jhan con *fold*=1



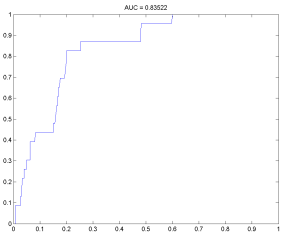
(c) Viola&Jones con *fold*=1



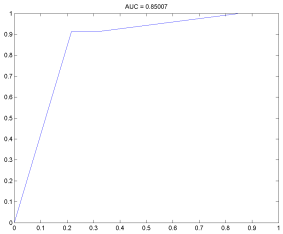
(d) Chen&Chen con *fold*=2



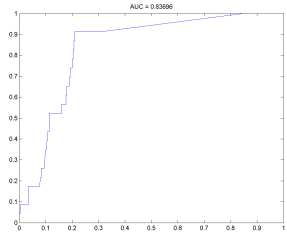
(e) Cheng&Jhan con *fold*=2



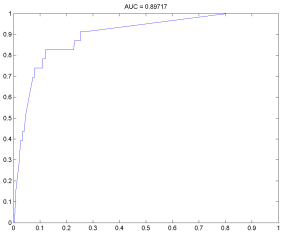
(f) Viola&Jones con *fold*=2



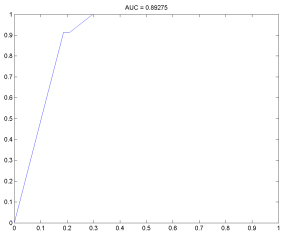
(g) Chen&Chen con *fold*=3



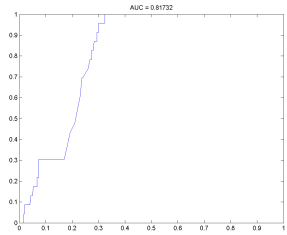
(h) Cheng&Jhan con *fold*=3



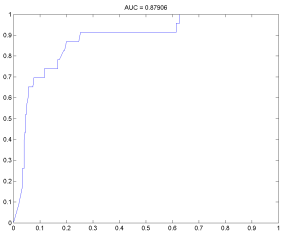
(i) Viola&Jones con *fold*=3



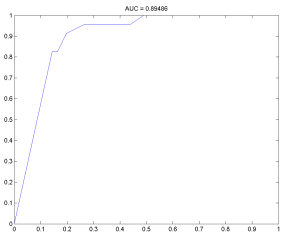
(j) Chen&Chen con *fold*=4



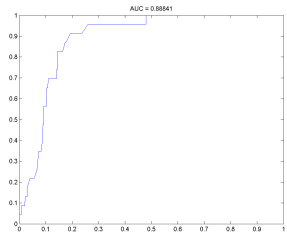
(k) Cheng&Jhan con *fold*=4



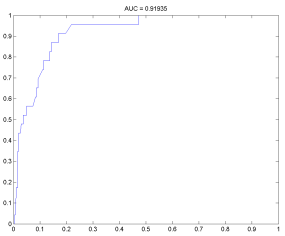
(l) Viola&Jones con *fold*=4



(m) Chen&Chen con *fold*=5



(n) Cheng&Jhan con *fold*=5



(o) Viola&Jones con *fold*=5

Figura D.4: Curvas ROC para las mamografías utilizando características estadísticas de textura.

APÉNDICE E

CONTENIDO DEL CD

El contenido del CD que acompaña este trabajo es el siguiente:

- Documentación del Trabajo Fin de Grado en formato PDF.
- Documentación generada por Doxygen en dos formatos:
  - HTML. Se accede mediante el archivo index.htlm.
  - PDF.
- Vídeos de demostración de la aplicación.

BIBLIOGRAFÍA

[1] S. Aguado Rodríguez y P. L. Antona Díaz. *Aplicación de la Técnica SVM en el Análisis Forense de Imágenes de Dispositivos Móviles*. Fac. de Informática - Depto. de Sistemas Informáticos y Computación. Universidad Complutense de Madrid, Archivo institucional e-prints complutense, 2012. http://eprints.ucm.es/16102OAI.

[2] L. Alonso Romero y T. Calonge Cano. Redes neuronales y reconocimiento de patrones, 2001. Imprenta Cardenal (Salamanca, España).

[3] American College of Radiology - (ACR). *BI-RADS Atlas. Breast Imaging Reporting and Data System. Mammography, Breast Ultrasound Breast and MR Imaging, 3ª edición*. 2013.

[4] J. A. Andrade Hernández, C. E. Díaz Arcos, K. P. Hernández Hernández, J. R. Peña Butrón y A. F. Solís Magallanes. Reconocimiento de patrones. http://maestrosilvestre.aprenderapensar.net/files/2011/05/01-Rec_Patrones.doc, [Última consulta: Abril 2013].

[5] Asociación Española contra el Cáncer (AECC). Diagnóstico precoz del cáncer de mama. http://www.aecc.es/SobreElCancer/CancerPorLocalizacion/CancerMama/Paginas/diagnosticoprecoz.aspx, [Última consulta: Marzo 2013].

[6] Asociación Española contra el Cáncer (AECC). Pronóstico (supervivencia y mortalidad). http://www.aecc.es/SobreElCancer/CancerPorLocalizacion/CancerMama/Paginas/pronostico.aspx, [Última consulta: Marzo 2013].

[7] Asociación Española contra el Cáncer (AECC). Incidencia. https://www.aecc.es/sobreelcancer/cancerporlocalizacion/cancermama/paginas/incidencia.aspx, [Última consulta: Octubre 2013].

[8] B. Boehm, C. Abts, A. W. Brown, S. Chulani, B. K. Clark, E. Horowitz, R. Madachy, D. J. Reifer y B. Steece. *Software Cost Estimation with COCOMO II*. Prentice-Hall, 2000.

[9] G. Booch, J. Rumbaugh, I. Jacobson, J. Molina y J. Martínez. *UML: el lenguaje unificado de modelado*. Addison-Wesley Object Technology Series. Pearson Educación, 2006.

[10] G. Bradski y A. Kaebler. *Learning OpenCV, Computer vision with the OpenCV library*. O'Reilly, 2008.

[11] G. Bueno, A. Martínez, J. M. Caballero, P. Cosías, A. Martín y A. Adán. Detección automática de lesiones en la mama. *XXV Jornadas de automática, Ciudad Real*, 2008.

[12] G. Bueno, N. Vállez, O. Déniz, P. Esteve, M. Rienda, M. Arias y C. Pastor. Automatic breast parenchymal density classification integrated into a CADe system. *International Journal of Computer Assisted Radiology and Surgery*, 6(3):309–318, 2011.

[13] J. A. Carrasco-Ochoa, J. F. M. Trinidad, J. S. Rodríguez y G. S. di Baja. *Proceedings on Pattern Recognition - 5th Mexican Conference, MCPR, Querétaro, Mexico*, volume 7914 of *Lecture Notes in Computer Science*. Springer, 2013.

[14] M. Castrillón Santana, O. Déniz Suárez, L. Antón Canalís y J. Lorenzo Navarro. Face and facial feature detection evaluation- performance evaluation of public domain haar detectors for face and facial feature detection. *VISAPP*, pp. 167–172, 2008.

[15] C.-W. Chang, D. M.-H. Chiang, A. W.-D. Wu y Y.-C. Chang. Identifying critical features of complex systems under prior known conditions. *Computers & Industrial Engineering*, 61(3):788–793, 2011.

[16] Y. Chen y C. Chen. A cascade of feed-forward classifiers for fast pedestrian detection. *ACCV*, pp. 905–914, 2007.

[17] G. Coleman y R. Verbruggen. A quality software process for rapid application development. *Software Quality Journal*, 7(2):107–122, 1998.

[18] A. de la Escalera Hueso. *Visión por computador: fundamentos y métodos*. Pearson, Prentice Hall, 2001.

[19] Departament d'Informàtica de Sistemes i Computadors - Universitat Politècnica de València. Reconocimiento facial. `http://web-sisop.disca.upv.es/imd/cursosAnteriors/2k8-2k9/videos/trabajandoConOpenCV/secciones/Interaccion.html`, [Última consulta: Diciembre 2012].

[20] DICOM. Digital imaging and communications in medicine. `http://medical.nema.org/`, [Última consulta: Diciembre 2013].

[21] A. Dragolici. *Detección, localización e identificación biométrica de caras en imágenes: métodos y evaluación en el marco NIST-MBGC. Tesis de Máster*. 2010.

[22] R. O. Duda, P. E. Hart y S. D. G. *Pattern Classification, 2ª edición*. Wiley-Interscience, Noviembre 2000.

[23] R. Erandi Barrientos Martínez, N. Cruz Ramírez, H. G. Acosta Mesa, I. Rabatte Suárez, M. d. C. Gogeascoechea Trejo, P. Pavón León y S. L. Blázquez Morales. Árboles de decisión como herramienta en el diagnóstico médico. *Revista médica de la Universidad Veracruzana*, 9(2):19–24, 2009.

[24] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[25] Y. Freund y R. E. Schapire. A short introduction to boosting. pp. 1401–1406, 1999.

[26] F. O. García, C. Bravo y I. García. *Tema 11. El Proceso Unificado de Desarrollo Software. Ingeniería del Software. Ingeniería del Software de Gestión*. Escuela Superior de Informática de Ciudad Real.

[27] O. Ivanciuc. SVM - support vector machines. `http://www.support-vector-machines.org/`, [Última consulta: Abril 2013].

[28] I. Jacobson, G. Booch y J. Rumbaugh. *El proceso unificado de desarrollo de software*. Addison Wesley, 2000.

[29] B. Jähne. *Digital image processing (6. ed.)*. Springer, 2005.

[30] C. Larman y B. Valle. *UML y patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado*. Editorial Alhambra S. A. (SP), 2003.

[31] Q. Li, U. F. Niaz y B. Merialdo. An improved algorithm on viola-jones object detector. In *CBMI 2012, 10th Workshop on Content-Based Multimedia Indexing, Annecy, France*, 06 2012.

[32] Mammographic Image Analysis Society. mini-MIAS database. `http://peipa.essex.ac.uk/info/mias.html`, [Última consulta: Diciembre 2013].

[33] R. Marín Morales y J. T. Palma Méndez. *Métodos de Inteligencia Artificial*. McGraw-Hill/Interamericana Esp, Aravaca, 2008.

[34] Y. Martínez Díaz, H. Méndez Vázquez y E. García Reyes. *Detección de rostros: una revisión crítica*. Memorias del XII Congreso Nacional de Matemática y Computación de la SCMC, COMPUMAT 2011 y el IX Congreso Nacional de Reconocimiento de Patrones, RECPAT 2011, Noviembre 2011.

[35] A. Materka y M. Strzelecki. Texture analysis methods. *A Review, Technical University of Lodz, Institute of Electronics, COST B11 report*, p. 33, Brussels 1998.

[36] M. N. Murty y V. S. Devi. *Pattern Recognition - An Algorithmic Approach*. Undergraduate Topics in Computer Science. Springer, 2011.

[37] M. Polo. *Manual de Ingeniería del Software II*. Escuela Superior de Informática de Ciudad Real, 2002.

[38] J. Ruiz Shulcloper, A. Guzmán Arenas y J. Martínez-Trinidad. *Enfoque Lógico Combinatorio al Reconocimiento de Patrones. Selección de variables y clasificación supervisada*. Departamento de Ciencias de la Computación e Inteligencia Artificial. Universidad del País VascoEuskal Herriko Unibertsitatea.

[39] J. Rumbaugh, I. Jacobson y G. B. Addison-Wesley. *El Proceso Unificado de Desarrollo Software*. Addison-Wesley, 2000.

[40] M. Sampat, M. Markey y A. Bovik. Computer-aided detection and diagnosis in mammography. *Handbook of Image and Video Processing Elsevier Academic Press*, pp. 1195–1217, 2005.

[41] J. J. B. Solano. Sistema automatizado de apoyo al diagnostico de lesiones en imagenes de mamografias. `http://www.portalesmedicos.com/publicaciones/articles/4031/3/`, [Última consulta: Febrero 2012].

[42] A. D. Sotolongo. Extracción de características para el análisis de imágenes de mamografías. `http://www.monografias.com/trabajos93/extraccion-caracteristicas-analisis-imagenes-mamografias/extraccion-caracteristicas-analisis-imagenes-mamografias2.shtml`, [Última consulta: Enero 2014].

[43] J. Stapleton. *DSDM, Dynamic Systems Development Method: The Method in Practice*. Addison-Wesley, 1997.

[44] N. Vállez, G. Bueno, O. Déniz, J. Dorado, J. A. Seoane, A. Pazos y C. Pastor. Breast density classification to reduce false positives in CADe systems. *Computer methods and programs in biomedicine*, Noviembre 2013.

[45] P. A. Viola y M. J. Jones. Rapid object detection using a boosted cascade of simple features. pp. 511–518, 2001.

[46] Your Breast Options. Mamografías y otras pruebas. `http://www.yourbreastoptions.com/es-ES/detectiondiagnosis/Paginas/mammogramsothertests.aspx`, [Última consulta: Febrero 2012].