

Instituto Tecnológico de León

“Detección de dos de los defectos presentados en
piezas inyectadas aplicando máquinas de visión y
sistemas inteligentes.”

Tesis

Que presenta:

Blanca Dolores Ruiz López

Para obtener el grado de

Maestro (a) en Ciencias de la Computación

Con la Dirección de:

Dra. María Del Rosario Baltazar Flores

Revisores

MC. Miguel Ángel Casillas Araiza

Dr. Raúl Santiago Montero

Dr. Juan Francisco Mosiño

León, Guanajuato. Enero 2023.



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO
NACIONAL DE MÉXICO

Instituto Tecnológico de León
División de Estudios de Posgrado e Investigación

León, Guanajuato, 11/enero/2023
OFICIO No. DEPI-006-2023

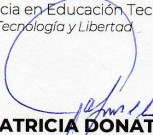
**C. BLANCA DOLORES RUIZ LÓPEZ
PRESENTE**

De acuerdo al fallo emitido por la Comisión Revisora, integrada por los: **Dra. María del Rosario Baltazar Flores**, **M.C. Miguel Angel Casillas Araiza**, **Dr. Raúl Santiago Montero**, **Dr. Juan Francisco Mosiño**, y considerando que cubre todos los requisitos establecidos en los Lineamientos Generales para la Operación del Posgrado del Tecnológico Nacional de México, **se autoriza la impresión** del trabajo de tesis titulado: **"Detección de dos de los defectos presenta<dos en piezas inyectadas aplicando máquinas de visión y sistemas inteligentes"**.

Lo que hacemos de su conocimiento para los efectos y fines correspondientes.

ATENTAMENTE

Excelencia en Educación Tecnológica®
Ciencia Tecnología y Libertad


LIC. PATRICIA DONATO JIMÉNEZ
SUBDIRECTORA ACADÉMICA

C.c.p. Expediente


JMCV/CLDG



Av. Tecnológico s/n Fracc. Industrial Julián de Obregón C.P.37290 León, Guanajuato.
Tel.: 477 710 5200 e-mail: tecleon@leon.tecnm.mx tecnm.mx | leon.tecnm.mx



2023
**Francisco
VILLA**

León, Guanajuato., a 11 de enero del 2023.

C. ING. LUIS ROBERTO GALLEGOS MUÑOZ
JEFE DE SERVICIOS ESCOLARES
P R E S E N T E

Por este medio hacemos de su conocimiento que la tesis titulada “**Detección de dos de los defectos presentados en piezas inyectadas aplicando máquinas de visión y sistemas inteligentes**”, ha sido leída y aprobada por los miembros del Comité Tutorial para su evaluación por el jurado del acto de examen de grado al alumno (a) **C. Blanca Dolores Ruiz López**, con número de control **M14241007** como parte de los requisitos para obtener el grado de Maestro(a) en Ciencias de la Computación (MCCOM-2011-05).

Sin otro particular por el momento, quedamos de Usted.

A T E N T A M E N T E
COMITÉ TUTORIAL



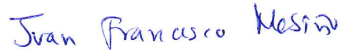
Dra. María del Rosario Baltazar Flores
DIRECTORA



MC. Miguel Ángel Casillas Araiza
REVISOR



Dr. Raúl Santiago Montero
REVISOR



Dr. Juan Francisco Mosiño
REVISOR

DECLARACION DE AUTENTICIDAD Y DE NO PLAGIO

Yo, **Blanca Dolores Ruiz López** identificado con No. Control **M14241007**, alumno (a) del programa de la **Maestría en Ciencias de la Computación**, autor(a) de la Tesis titulada: "**Detección de dos de los defectos presentados en piezas inyectadas aplicando máquinas de visión y sistemas inteligentes**." DECLARO QUE:

1.- El presente trabajo de investigación, tema de la tesis presentada para la obtención del título de **MAESTRO (A) EN CIENCIAS DE LA COMPUTACIÓN** es original, siendo resultado de mi trabajo personal, el cual no he copiado de otro trabajo de investigación, ni utilizado ideas, fórmulas, ni citas completas "stricto sensu", así como ilustraciones, fotografías u otros materiales audiovisuales, obtenidas de cualquier tesis, obra, artículo, memoria, etc. en su versión digital o impresa.

2.- Declaro que el trabajo de investigación que pongo a consideración para evaluación no ha sido presentado anteriormente para obtener algún grado académico o título, ni ha sido publicado en sitio alguno.

3.- Declaro que las pruebas o experimentos derivados de esta investigación fueron realizados bajo el consentimiento de los involucrados y con fines estrictamente académicos conforme a criterios éticos de confidencialidad.

Soy consciente de que el hecho de no respetar los derechos de autor y hacer plagio, es objeto de sanciones universitarias y/o legales por lo que asumo cualquier responsabilidad que pudiera derivarse de irregularidades de la tesis, así como de los derechos sobre la obra presentada.

Asimismo, me hago responsable ante el Tecnológico Nacional de México/Instituto Tecnológico de León o terceros, de cualquier irregularidad o daño que pudiera ocasionar por el incumplimiento de lo declarado.

De identificarse falsificación, plagio, fraude, o que el trabajo de investigación haya sido publicado anteriormente; asumo las consecuencias y sanciones que de mi acción se deriven, responsabilizándome por todas las cargas pecuniarias o legales que se deriven de ello sometiéndome a las normas establecidas en los Lineamientos y Disposiciones de la Operación de Estudios de Posgrado en el Tecnológico Nacional de México.

León, Guanajuato a 16 del mes de enero de 2023



Blanca Dolores Ruiz López

Nombre y firma del autor (a)

ACUERDO PARA USO DE OBRA (TESIS DE GRADO)

A QUIEN CORRESPONDA

PRESENTE

Por medio del presente escrito, **Blanca Dolores Ruiz López** en lo sucesivo el(AUTOR) hace constar que es titular intelectual de la obra denominada: "**Detección de dos de los defectos presentados en piezas inyectadas aplicando máquinas de visión y sistemas inteligentes**", (en lo sucesivo la OBRA) en virtud de lo cual autoriza al Tecnológico Nacional de México/Instituto Tecnológico de León (en lo sucesivo TECN/IT León) para que efectúe resguardo físico y/o electrónico mediante copia digital o impresa para asegurar su disponibilidad, divulgación, comunicación pública, distribución, transmisión, reproducción, así como digitalización de la misma con fines académicos y sin fines de lucro como parte del Repositorio Institucional del TECN/ITLeón.

De igual manera, es deseo del AUTOR establecer que esta autorización es voluntaria y gratuita, y que de acuerdo a lo señalado en la Ley Federal del Derecho de Autor y la Ley de Propiedad Industrial el TECN/IT León cuenta con mi autorización para la utilización de la información antes señalada, estableciendo que se utilizará única y exclusivamente para los fines antes señalados. El AUTOR autoriza al TECN /IT León a utilizar la obra en los términos y condiciones aquí expresados, sin que ello implique se le conceda licencia o autorización alguna o algún tipo de derecho distinto al mencionada respecto a la "propiedad intelectual" de la misma OBRA; incluyendo todo tipo de derechos patrimoniales sobre obras y creaciones protegidas por derechos de autor y demás formas de propiedad intelectual reconocida o que lleguen a reconocer las leyes correspondientes. Al reutilizar, reproducir, transmitir y/o distribuir la OBRA se deberá reconocer y dar créditos de autoría de la obra intelectual en los términos especificados por el propio autor, y el no hacerlo implica el término de uso de esta licencia para los fines estipulados. Nada de esta licencia menoscaba o restringe los derechos patrimoniales y morales del AUTOR.

De la misma manera, se hace manifiesto que el contenido académico, literario, la edición y en general de cualquier parte de la OBRA son responsabilidad de AUTOR, por lo que se deslinda al (TECN/ITLeón) por cualquier violación a los derechos de autor y/o propiedad intelectual, así como cualquier responsabilidad relacionada con la misma frente a terceros. Finalmente, el AUTOR manifiesta que estará depositando la versión final de su documento de Tesis, OBRA, y cuenta con los derechos morales y patrimoniales correspondientes para otorgar la presente autorización de uso.

En la ciudad de León, del estado de Guanajuato a los 16 días del mes de enero de 2023.

Atentamente,



Blanca Dolores Ruiz López

Dedicatoria

Dedico mi trabajo:

A mi hija, quiero que llegues mas lejos de lo que tu mamá ha llegado.

A mis padres que siempre me han apoyado y amado.

A mis hermanas que amo y espero verlas crecer profesionalmente.

A mis abuelitos quiero que siempre estén orgullosos de mi.

A mis suegros que han sido un par de padres más para mi.

A todos aquellos que forman parte de mi familia y sin duda alguna vez escuche palabras de aliento.

A mis seres queridos que hoy no están con nosotros.

A mi Morenita que siempre confió en mi y me animo tanto.

Agradecimientos

Agradezco:

A Dios por darme fortaleza para cumplir mis metas.

A mi esposo por todo el amor, apoyo y por ayudarme a crecer siempre de manera personal y profesional.

A mis maestros y asesores que me acompañaron en el camino.

Al Ing. Resendiz por permitirme creer y desarrollarme en su área de trabajo.

A mis jefes que me apoyaron para poder concluir con esta meta.

Publicaciones

- **Classification of defects in injected parts through: moments of image and multilayer perceptron.**

International Conference on Advanced Research in Technologies, Information, Innovation and Sustainability (ARTIIS 2022).

Blanca Ruiz, Rosario Baltazar, Raul Santiago, Miguel Casillas, Francisco Mosiño.

Instituto Tecnológico de León.

Santiago de Compostela, Spain, 14th of September 2022.

ISBN: 978-3-031-20319-0. Volumen: 1675. Páginas: 558-571.

Tabla de Contenido

Tabla de Contenido	XIII
Contenido de Tablas	XVII
Contenido de Figuras	XIX
1. Estado del Arte	9
1.1. Detección de rebaba, hoyos y manchas	13
1.2. Momentos invariantes de Hu	15
1.3. Redes Neuronales	17
1.3.1. Perceptron multicapa	18
1.4. Algoritmos genéticos	19
1.4.1. Evolución diferencial	20
2. Marco Teórico	23
2.1. Procesamiento de imágenes	23
2.1.1. Intensidad de color en una imagen	23
2.1.2. Ruido en imágenes	24
2.1.3. Segmentación de imágenes	24
2.1.4. Adición de imágenes	24
2.2. Descriptores de una imagen.	25
2.3. Funciones Momento.	25
2.3.1. Momentos Geométricos de una imagen.	26
2.4. Momentos Invariantes de Hu	27

2.5. Redes Neuronales.	29
2.5.1. La arquitectura de las Redes Neuronales Artificiales.	30
2.5.2. Perceptron multicapa.	31
2.6. Algoritmos Genéticos.	35
2.6.1. Evolución diferencial.	36
2.7. Diagrama del proyecto	39
2.7.1. Paso 1.	39
2.7.2. Paso 2.	39
2.7.3. Paso 3.	40
2.7.4. Paso 4.	41
2.7.5. Paso 5.	41
2.7.6. Paso 6.	42
3. Desarrollo	43
3.1. Modelo del problema	43
3.1.1. Defectos principales	44
3.2. Recolección de la base de datos.	45
3.3. Pruebas de procesamiento de imágenes.	53
3.4. Obtención de vectores característicos	65
3.5. Entrenamiento de la red neuronal	65
3.5.1. Entrenamiento de MLP	65
3.5.2. Entrenamiento DE-MLP	66
3.6. Clasificación de las imágenes.	66
3.7. Prototipo del sistema	67
4. Resultados	69
4.1. Resultados para la cara frontal.	69
4.2. Resultados para la cara inferior.	75
4.3. Resultados para la cara superior.	81
4.4. Resultados para la cara lateral.	86

5. Conclusiones y Trabajo a Futuro	91
5.1. Trabajo a futuro.	93
Bibliografía	95
A. Anexo I: Diseño propuesto.	99
B. Anexo II: Código de red neuronal MLP.	105
C. Anexo III: Código de red neuronal MLP-DE.	113

Contenido de Tablas

3.1. Dimensión de las bases de datos.	47
3.2. Distribución de base de datos correspondiente a piezas con manchas y agujeros.	56
4.1. Base de datos de la cara frontal para experimentación en las redes neuronales MLP y DE-MLP.	70
4.2. Estadísticos del porcentaje de clasificación de MLP en la cara frontal con manchas o huecos y rebabas.	72
4.3. Estadísticos del porcentaje de clasificación de DE-MLP de la cara frontal con manchas o huecos y rebabas.	75
4.4. Base de datos de la cara inferior para experimentación en las redes neuronales MLP y DE-MLP.	75
4.5. Estadísticos del porcentaje de clasificación de MLP en la cara inferior con manchas o huecos y rebabas.	78
4.6. Estadísticos del porcentaje de clasificación de DE-MLP de la cara inferior con manchas o huecos y rebabas.	79
4.7. Base de datos de la cara superior para experimentación en las redes neuronales MLP y DE-MLP.	81
4.8. Estadísticas del porcentaje de clasificación de MLP en la cara superior con manchas o huecos y rebabas.	83
4.9. Estadísticos del porcentaje de clasificación de DE-MLP de la cara superior con manchas o huecos y rebabas.	85

4.10. Base de datos de la cara lateral para experimentación en las redes neuronales MLP y DE-MLP.	86
4.11. Estadísticas del porcentaje de clasificación de MLP en la cara lateral con manchas o huecos y rebabas.	87
4.12. Estadísticos del porcentaje de clasificación de DE-MLP de la cara lateral con manchas o huecos y rebabas.	87

Contenido de Figuras

1.1. Características del sensor de triangulación láser, [Barrero, 2021]	14
1.2. Esquema explicativo del funcionamiento de un sistema de triangulación láser, [Barrero, 2021]	14
1.3. Precisión del algoritmo de reconocimiento de dígitos mediante Hu y Flusser, [Gómez and Gutierrez, 2011]	16
1.4. Precisión de los dígitos 6 y 9 luego de la corrección, [Gómez and Gutierrez, 2011]	16
1.5. Porcentaje de clasificación correcta por textura para PM, [Pajarez and Moreno, 2001]	18
1.6. Calidad media y mejor de la población a lo largo de las generaciones, con 1 propiedad y RS126, [Rivas, 2022]	21
2.1. Red monocapa recurrente, donde (a) conexiones hacia adelante, (c) conexiones laterales y (d) conexiones autorecurrentes. [Flores and Fernández, 2008]	31
2.2. Red multicapa de propagación hacia adelante, donde (a) conexiones hacia adelante. [Flores and Fernández, 2008]	31
2.3. Red multicapa de propagación hacia atrás, donde (a) conexiones hacia adelante, (b) conexiones hacia atrás, (c) conexiones laterales y (d) conexiones autorecurrentes. [Flores and Fernández, 2008]	32
2.4. Diagrama de la red MLP.	33
2.5. Proceso de algoritmo genético mediante selección natural.	36
2.6. Diagrama de flujo genérico de las operaciones llevadas a cabo por el algoritmo evolución diferencial, [Cuevas et al., 2016]	37

2.7. Diagrama general del proyecto.	39
2.8. Diseño del sistema de caída y traslado.	40
2.9. Diseño interno de cabinas de imagen.	40
2.10. Diagrama de proceso en la cabina de imágenes.	41
3.1. Impactos porcentuales de los defectos en el periodo 2020-2021.	44
3.2. 80-20 del impacto de los defectos en el acumulado del 2020-2021.	45
3.3. Estructura para la toma de imágenes.	46
3.4. Imagen de cada una de las caras de las Tee.	47
3.5. Ejemplo muestra de la base de datos sin defecto de la cara frontal.	48
3.6. Ejemplo muestra de la base de datos rebaba presentada en la cara frontal.	48
3.7. Ejemplo muestra de la base de datos huecos presentados en cara frontal.	48
3.8. Ejemplo muestra de la base de datos manchas presentadas en cara frontal.	49
3.9. Ejemplo muestra de la base de datos sin defecto en la cara inferior	49
3.10. Ejemplo muestra de la base de datos con rebaba en la cara inferior.	49
3.11. Ejemplo muestra de la base de datos con huecos en la cara inferior.	50
3.12. Ejemplo muestra de la base de datos con manchas en la cara inferior.	50
3.13. Ejemplo muestra de la base de datos sin defecto en la cara lateral.	50
3.14. Ejemplo muestra de la base de datos con rebaba en la cara lateral.	51
3.15. Ejemplo muestra de la base de datos con hoyos en la cara lateral.	51
3.16. Ejemplo muestra de la base de datos con manchas en la cara lateral.	51
3.17. Ejemplo muestra de la base de datos sin defecto en la cara superior.	52
3.18. Ejemplo muestra de la base de datos con rebaba en la cara superior.	52
3.19. Ejemplo muestra de la base de datos con huecos en la cara superior.	52
3.20. Ejemplo muestra de la base de datos con manchas en la cara superior.	53
3.21. Ejemplo de imagen original con fondo negro.	54
3.22. Ejemplo de la figura 3.21 con intensidad de color.	54
3.23. Operaciones con canal de intensidad.	55

3.24. Binarización y Detección de borde de la imagen.	56
3.25. Ejemplos de la Tee en fondo verde.	57
3.26. Procesamiento de la cara frontal para rebaba.	57
3.27. Procesamiento de la cara frontal para manchas y hueco.	58
3.28. Tratamiento final de la Tee sin defectos para la cara frontal.	58
3.29. Tratamiento final de la Tee con defectos para la cara frontal.	59
3.30. Procesamiento de la cara inferior para rebaba.	59
3.31. Procesamiento de la cara inferior para manchas y hueco.	60
3.32. Tratamiento final de la Tee sin defectos para la cara inferior.	60
3.33. Tratamiento final de la Tee con defectos para la cara inferior.	61
3.34. Procesamiento de la cara lateral para rebaba.	61
3.35. Procesamiento de la cara lateral para manchas y hueco.	62
3.36. Tratamiento final de la Tee sin defectos para la cara lateral.	62
3.37. Tratamiento final de la Tee con defectos para la cara lateral.	63
3.38. Procesamiento de la cara superior para rebaba.	63
3.39. Procesamiento de la cara superior para manchas y hueco.	63
3.40. Tratamiento final de la Tee sin defectos para la cara Superior.	64
3.41. Tratamiento final de la Tee con defectos para la cara lateralsuperior.	64
3.42. Diseño de la red neuronal.	66
3.43. Prototipo del diseño para la detección de defectos en la cara frontal.	67
4.1. Resultados de clasificación de MLP de los datos de entrenamiento cara frontal con manchas o huecos.	70
4.2. Resultados de clasificación de MLP de los datos de prueba cara frontal con manchas o huecos.	71
4.3. Resultados de clasificación de MLP de los datos de entrenamiento cara frontal con rebaba.	71
4.4. Resultados de clasificación de MLP de los datos de prueba cara frontal con rebaba.	72

4.5. Resultados de clasificación de DE-MLP de los datos de entrenamiento	
cara frontal con manchas o huecos, Fuente propia.	73
4.6. Resultados de clasificación de DE-MLP de los datos de prueba cara	
frontal con manchas o huecos, Fuente propia.	73
4.7. Resultados de clasificación de DE-MLP de los datos de entrenamiento	
cara frontal con rebaba, Fuente propia.	74
4.8. Resultados de clasificación de DE-MLP de los datos de prueba cara	
frontal con rebaba, Fuente propia.	74
4.9. Resultados de clasificación de MLP de los datos de entrenamiento	
cara inferior con manchas o huecos, Fuente propia.	76
4.10. Resultados de clasificación de MLP de los datos de prueba cara infe-	
rior con manchas o huecos, Fuente propia.	76
4.11. Resultados de clasificación de MLP de los datos de entrenamiento	
cara inferior con rebaba, Fuente propia.	77
4.12. Resultados de clasificación de MLP de los datos de prueba cara infe-	
rior con rebaba, Fuente propia.	77
4.13. Resultados de clasificación de DE-MLP de los datos de entrenamiento	
cara inferior con manchas o huecos, Fuente propia.	77
4.14. Resultados de clasificación de DE-MLP de los datos de prueba cara	
inferior con manchas o huecos, Fuente propia.	78
4.15. Resultados de clasificación de DE-MLP de los datos de entrenamiento	
cara inferior con rebaba, Fuente propia.	79
4.16. Resultados de clasificación de DE-MLP de los datos de prueba cara	
inferior con rebaba, Fuente propia.	80
4.17. Resultados de clasificación de MLP de los datos de entrenamiento	
cara superior con manchas o huecos, Fuente propia.	82
4.18. Resultados de clasificación de MLP de los datos de prueba cara su-	
perior con manchas o huecos, Fuente propia.	82
4.19. Resultados de clasificación de MLP de los datos de entrenamiento	
cara superior con rebaba, Fuente propia.	82

4.20. Resultados de clasificación de MLP de los datos de prueba cara superior con rebaba, Fuente propia.	83
4.21. Resultados de clasificación de DE-MLP de los datos de entrenamiento cara superior con manchas o huecos, Fuente propia.	84
4.22. Resultados de clasificación de DE-MLP de los datos de prueba cara superior con manchas o huecos, Fuente propia.	84
4.23. Resultados de clasificación de DE-MLP de los datos de entrenamiento cara superior con rebaba, Fuente propia.	84
4.24. Resultados de clasificación de DE-MLP de los datos de entrenamiento cara superior con rebaba, Fuente propia.	85
4.25. Resultados de clasificación de MLP de los datos de entrenamiento cara lateral con manchas o huecos, Fuente propia.	88
4.26. Resultados de clasificación de MLP de los datos de prueba cara lateral con manchas o huecos, Fuente propia.	88
4.27. Resultados de clasificación de MLP de los datos de entrenamiento cara lateral con rebaba, Fuente propia.	89
4.28. Resultados de clasificación de MLP de los datos de prueba cara lateral con rebaba, Fuente propia.	89
4.29. Resultados de clasificación de DE-MLP de los datos de entrenamiento cara lateral con manchas o huecos, Fuente propia.	89
4.30. Resultados de clasificación de DE-MLP de los datos de prueba cara lateral con manchas o huecos, Fuente propia.	90
4.31. Resultados de clasificación de DE-MLP de los datos de entrenamiento cara lateral con rebaba, Fuente propia.	90
4.32. Resultados de clasificación de DE-MLP de los datos de prueba cara lateral con rebaba, Fuente propia.	90
A.1. Diseño propuesto para la implementación del proyecto de detección de defectos en la Tee de 110x110mm.	99
A.2. Cara superior del diseño propuesto.	100

A.3. Cara inferior del diseño propuesto.	100
A.4. Cara lateral derecha del diseño propuesto.	101
A.5. Cara lateral izquierda del diseño propuesto.	101
A.6. Cara frontal del diseño propuesto.	102
A.7. Cara posterior del diseño propuesto.	102
A.8. Cabina para toma de imágenes del diseño propuesto.	103
A.9. Colocación de cabinas para toma de imágenes del diseño propuesto.	103

Resumen

El control de calidad en la industria es un factor de bastante importancia para cumplir con las especificaciones deseadas del producto que ofrecen, esto se logra muchas veces con apoyo del personal de trabajo. sin embargo, podemos encontrar algunas aplicaciones tecnológicas para lograr el mismo. En esta tesis se describe un desarrollo aplicado al control de calidad de piezas inyectadas mediante máquinas de visión, La pieza es inyectada y expulsada por la máquina lo que genera un proceso de caída el cual implica un diseño más complejo de la máquina de visión añadido a esto la pieza debe ser captada por todas sus caras, un total de seis caras de las cuales son cuatro diferentes. Obtenemos un conjunto de imágenes en un ambiente previamente establecido de las cuatro diferentes caras y los dos tipos de piezas encontradas: defectuosas y no defectuosas, seguido de un procesamiento de imágenes donde obtenemos el vector característico de cada una de ellas, este se ingresa a la red neuronal perceptrón multicapa donde se espera la clasificación de las imágenes correspondiente al tipo de pieza, una vez realizado presentamos su desempeño.

Introducción

La calidad en la industria siempre ha tenido gran importancia independientemente del producto o servicio que se ofrece, si bien comúnmente se cuenta con personal encargado de la examinación del producto hay defectos que se pasan por alto. Cuando hablamos de la producción de conexiones sanitarias algún proceso puede producir defectos en la fabricación. como ya mencionamos, aunque pasen por el área de calidad debido a la cantidad de producción algunas suelen no detectarse. El factor humano que puede perder de vista defectos y la velocidad de producción que para la Tee esta entre 4 piezas por minuto donde se debe revisar cada pieza retirar la rebaba en caso de ser posible o separarla de las piezas que cumplen los estándares son factores que incrementan la fallas en la calidad del producto. Estos factores generan costos extras de producción.

La detección de defectos al producir la Tee sanitaria cementada 110x110mm se realiza comúnmente de manera visual, en este proyecto se propone la aplicación de máquinas de visión y sistemas inteligentes para realizarlo de manera automática.

La visión artificial simula la capacidad de observación de los seres vivos para ver una escena, entenderla y actuar sobre esta, obteniendo la imagen a través de un dispositivo como lo es una cámara digital, escáner, telescopio, satélites entre otros, es una disciplina que involucra otras ramas de la inteligencia artificial para así poder tener un reconocimiento de lo que capta e interpretarlo para efectuar una acción sobre él [Galindo, 2012].

Los sistemas inteligentes son el apoyo de las máquinas de visión no solo para la detección de lo que se desea obtener sino también para la clasificación basada en paráme-

tros establecidos y algunas veces para realizar acciones sobre estos [Iván García, 2015], en nuestro caso los sistemas inteligentes nos serán de apoyo para la detección de los defectos y tomar acciones clasificándolo en tipos como:

- Agujero: defecto por falta de material
- Rebaba: defecto por daño en molde
- Quemadura: defecto por temperatura alta
- Grumos: defecto por temperatura baja
- Puntos de colores: defecto por Material contaminado

Lo cual ayuda sin duda al operador a no perder tiempo identificando el problema y este podrá tratarlo directamente, mejorando el tiempo de respuesta a los problemas que afectan la calidad.

La calidad en la industria siempre ha tenido gran importancia independientemente del producto o servicio que se ofrece, si bien comúnmente se cuenta con personal encargado de la examinación del producto esta aun así suele verse afectada después de las revisiones, pues hay defectos que se pasan por alto. Cuando hablamos de la producción de conexiones sanitarias algunos defectos suelen afectarlas y como ya mencionamos, aunque pasen por el área de calidad debido a la cantidad de producción algunas suelen no detectarse lo cual genera costos extra.

La detección de defectos al producir la Tee sanitaria cementada 110x110mm se realiza comúnmente de manera visual, en este proyecto se propone la aplicación de máquinas de visión y sistemas inteligentes para realizarlo de manera automática.

La visión artificial simula la capacidad de observación de los seres vivos para ver una escena, entenderla y actuar sobre esta, obteniendo la imagen a través de un dispositivo como lo es una cámara digital, escáner, telescopio, satélites entre otros, es una disciplina que involucra otras ramas de la inteligencia artificial para así poder tener un reconocimiento de lo que capta e interpretarlo para efectuar una acción sobre él [Galindo, 2012].

Planteamiento del Problema

La fabricación de cualquier producto siempre se busca que sea de la forma más económica posible sin afectar la calidad de este, el realizar productos de calidad desde la primera instancia en cualquier industria nos refleja un ahorro de recursos y esto es beneficioso de manera general para el planeta.

El producir conexiones sanitarias defectuosas tiene impactos económicos directos en la industria y el detectarla a destiempo hace que sean más grandes, el uso de personal humano para verificar que cada una de las piezas tenga la calidad deseada no es realmente efectivo pues el cansancio del personal es un factor de primer impacto para generar distracciones que lleven a una errónea clasificación de conexiones, adicional al ser demasiadas piezas producidas suelen dejar pasar piezas defectuosas sin darse cuenta lo cual prolonga el tiempo de identificación de lo que genera este defecto y esto nos lleva a que se siga produciendo con los defectos que impacta en números directamente en la producción.

Y ¿A qué nos referimos con que impacta en números?

Ciertamente es claro que cuando se produce una pieza defectuosa o una conexión scrap éste es materia prima desperdiciada que implica un reproceso de esta y para esto se necesita una doble inversión podemos observar directamente el impacto económico de esta forma, pero no es el único si lo pensamos bien al perder tiempo en la detección se invierten más recursos claro está que el tiempo es valioso y este se va perdiendo pero además podemos mencionar algunos otros de gran importancia como lo sería el desperdicio de trabajo del personal humano, energía eléctrica y otros recursos necesarios como el agua. Ahí observamos claramente un desperdicio, pero esto no se queda aquí además no se cumple con el programa de producción establecido, no se satisface las necesidades del cliente, no se mejora un costo en el producto, no se facilita la tarea del personal humano pues al no realizarla correctamente es necesario invertir un doble esfuerzo, se aumenta el número de horas muertas, se eleva el porcentaje de scrap, entre otros y ahora sí, podemos ver cuál es todo el impacto que se genera de la detección a destiempo de las conexiones scrap además todos esos

parámetros que podemos ayudar a reducir si planteamos una mejora.

Para el apoyo a la mejora nos planteamos la siguiente cuestión ¿Un sistema de máquina de visión podrá identificar los defectos presentados en la Tee como manchas, rebabas o quemaduras ? el fin de esto es identificar el producto que cumple o aquel que no cumple con los estándares, este consta de cámaras que captan la forma de la conexión y actúan sobre esta según lo establecido, esto se aplicará directamente en la planta de fabricación de conexiones Wavin león con las Tee Sanitarias cementadas de 110x110mm.

Hipótesis

Es posible la detección de dos defectos en la Tee cementada de 110x110mm de manera automática a través de máquinas de visión y sistemas inteligentes.

Objetivos Generales

Detectar dos tipos de defectos en la Tee cementada de 110x110mm a través de máquinas de visión y sistemas inteligentes en una línea de producción para la reducción del índice de scrap en la planta Wavin León.

Objetivos Específicos

- Realizar estudio amplio del estado del arte de las técnicas para detectar defectos en piezas inyectadas y de esta manera efectuar la toma de imágenes para la base de datos estableciendo los tipos de defectos para identificar a cuál corresponde cada una de las imágenes
- Realizar el estadístico para la definición de los dos defectos a detectar.
- Determinar la técnica de procesamiento de imágenes más adecuada a las necesidades de clasificación, desarrollar el sistema preliminar de detección de

errores y su arreglo óptico

- Implementar el sistema automatizado de la detección de los defectos en el proceso de inyección de conexiones sanitarias y difundir los resultados de investigación en medios académicos especializados.

Justificación

El personal humano es el pilar de toda la producción generada en la industria, es el personal de primera mano por el que pasa el producto y toda la cadena industrial depende de ellos es por eso que mientras su trabajo sea lo más cómodo posible siempre lo harán de la mejor manera con lo que tendremos consecuencias positivas, las jornadas laborales suelen ser largas y monótonas si añadimos la preocupación de que tu trabajo se esté dando de manera correcta se ve reflejado un cansancio no solo físico si no mental por parte del colaborador, el ayudar a mejorar esto facilitaría el trabajo del personal dando como resultados un colaborador feliz con lo que hace que sin duda tendrá un mejor estado anímico y físico, al ser ellos el pilar de la producción también tendremos un impacto en números ahora este impacto será positivo, nos favorecerá en el ahorro de todos los recursos invertidos: agua, energía eléctrica, materia prima, etc. Se reducirá una inversión de esfuerzo laboral por el capital humano, y ciertamente nos acercaremos al cumplimiento de la producción.

Bien ahora con esto podríamos decir que los errores en la detección de defectos son inevitables, sin embargo estos errores pueden ser reducidos o eliminados, si bien suena difícil la inspección al 100 % del producto no es algo imposible, recordemos que el cliente puede perder la confianza al obtener un solo producto de mala calidad, para evitar esto y lograr la disminución o eliminación de defectos contamos con apoyo de nuevos entornos como lo es el poka-yoke implementando tres funciones principales:

- Parada: las funciones son paradas en la detección de un defecto
- Control: las piezas defectuosas no pueden pasar al próximo proceso.

- Aviso: alarmas y señales de ocurrencia de defectos.

Todo esto lo podemos apoyar de los mecanismos poka-yoke: alarmas y detección de errores, conmutadores de límite, contadores, tacos de guía de diferentes tamaños, etc [Hirano., 1991].

La tecnología de procesos es uno de los soportes para lograr la sostenibilidad de la naturaleza implementando la tecnología más novedosa y buscando que no sea desechada rápidamente además hace un aporte al crecimiento económico y mejora la capacidad productiva si se aplica para realizar acciones automáticas como la detección de defectos inmediatos en la producción de la Tee cementada de 110x110mm podemos hacer una comparación de los gastos de recursos antes y después de la implementación del sistema y conocer la cantidad de conexiones buenas producidas en la misma máquina previamente a la aplicación del sistema e incluso su deferencia con la producción posterior con esto observaremos las ventajas de la aplicación además el ahorro de recursos generado y estudiaremos sus áreas de oportunidad.

La primer forma de determinar los dos defectos que serán seleccionados para la detección de ellos es observar cual de estos tiene un mayor impacto monetario para esto se contemplan siete defectos posibles.

- Apariencia : Piezas con manchas o marcas fuera de los estándares de calidad.
- Deformación: Pieza con deformidades.
- Pieza incompleta: Pieza con agujeros o incompleta.
- Rebabas: Pieza con material excedente a su forma.
- Suciedad o aceite: Pieza con aceite de las maquinarias o manchas de suciedad.
- Veteado: Pieza con rayas o vetas de color marrón.

La pérdida económica se logra determinar con ayuda del historial de las pérdidas surgidas en los años 2020 y 2021, en relación dinero-defecto logramos llegar a los

defectos con mayor impacto monetario, los cuatro defectos ya presentados son aquellos que mayor impacto económico tienen donde los dos más altos son la apariencia y las rebabas que presentan las piezas.

Capítulo 1

Estado del Arte

La búsqueda de la calidad para cualquier producto o servicio es algo que se encuentra presente desde mucho tiempo atrás, los egipcios son un ejemplo claro de esto pues en la construcción de las pirámides ellos aplicaron sistemas de control de calidad. La fabricación de productos artesanos en la edad media también nos muestra un ejemplo de la búsqueda de la calidad pues ellos diseñaban y desarrollaban sus productos asegurándose que la calidad final de este fuera aceptable a sus estándares, más adelante en el siglo XVIII es el comienzo de la producción en masa de bienes donde el proceso de fabricación se integraba por varias personas pero se necesitaba buscar un estándar de calidad y es ahí donde se implementó la producción siguiendo medidas específicas que hoy conocemos como metrología, también fueron aplicadas normas adecuadas al proceso que se conocen como normalización con esto se disminuyeron los defectos en la producción reduciendo también las variaciones que se encontraban en el proceso, todo este modelo de producción abrió camino a la revolución industrial donde Frederick W. Taylor propuso separar el proceso en líneas que desarrollan las actividades por secciones implementando con esto inspectores de calidad los cuales se encargaban de separar el artículo defectuoso de los que se encontraban en buen estado esto ayudó a mejorar los precios de su producto pues eran de una mejor calidad [san Miguel., 2009]. Si observamos bien ya se encontraban implementados los sistemas para el control de la calidad y con el paso de los

años se fueron puliendo para que esta fuera lo mejor alcanzada posible, si bien estos eran de forma muy manual ya se buscaba reducir los defectos en la producción, pero, aunque en la actualidad en muchas industrias esta tarea se sigue realizando de manera rudimentaria esto es propenso altamente a la no detección de productos defectuosos pues el error humano es un factor importante y que podemos encontrar interactuando directamente en algunos casos, la necesidad de reducir estos errores de detección y el trabajo laboral nos lleva a la aplicación de sistemas para el control de la calidad automatizados en la industria es aquí donde se abren camino y se comienzan a implementar.

El diseño de sistemas autónomos e inteligentes es aplicado para la capacitación, soporte, control de producción, etc., hasta el control de calidad que consiste en sistemas de calidad con aplicaciones de inteligencia artificial basados en aprendizaje automático y nos ayudan a garantizar la calidad adecuada para las necesidades de la planta [Åsa Fast-Berglund, 2020], algunas de las aplicaciones de estos es la detección de defectos en superficies mediante segmentación de umbral gradiente híbrido y registro de imágenes, La superficie planteada para esta aplicación tenía patrones especiales que son establecidos pero surgían errores de detección cuando eran separados por un borde es por eso que se aplicó un sistema que implementa dos métodos: segmentación por umbral gradiente para la detección de bordes que aplica desde uno a más valores de umbral para convertir la intensidad gris de una imagen a una con valores binarios, aplicando un filtro de frecuencia cuando se trata de procesar superficies metálicas y se elimina la influencia del ruido, una vez hecho esto se presentan operadores de convolución que nos darán las características del borde y mostrarán el vector gradiente de la intensidad del gris de la imagen. El segundo método consiste en el registro y diferenciación de imágenes donde se introduce una imagen de prueba que es comparada con una de referencia y obtenemos las características de cada una de ellas, el registro de imágenes nos ayuda a obtener las coincidencias y diferencias entre la imagen de prueba contra la imagen de referencia con esto se obtiene el registro de la imagen para la detección de defectos [Guangzhong Cao, 2018].

La aplicación de procesamiento de imágenes para control de la calidad en diver-

Los procesos de manufactura es amplia, esta implica el cambio en la naturaleza de las imágenes para el fin necesario, dividiendo cada una de ellas en miles de cuadros pequeños llamados píxeles ubicados de la forma (x, y) con un rango de 0 a 255 y realizando una conversión a umbral blanco y negro de esta para obtener una imagen binaria. Encontramos dos principales tipos de procesamiento de imágenes [\[Muangklang, 2019\]](#):

- Procesamiento de imágenes analógicas: alteración de imágenes con medios eléctricos.
- Procesamiento de imágenes digitales: modifica una imagen dando otra versión de la misma

Y un ejemplo del empleo de este en sistemas de la búsqueda de la calidad es el sistema de detección y clasificación de defectos en frutas mediante el procesamiento de imágenes que como vemos ya implica una clasificación, para este proyecto se establecieron los tipos de defectos más comunes que se encontraban en las frutas y se propusieron 5 clases con defectos y una 6 que especifica la clase sin defectos, se tomaron varias imágenes de las 6 clases establecidas a estas se les aplicaron diversos filtros para la obtención de diferentes características y después se cargó la información a una red neuronal perceptrón multicapa que se entrenó con el algoritmo retropropagación para obtener la clasificación de los defectos [\[Tellez, 2003\]](#), otra aplicación de estas tecnologías se ve en la búsqueda de la calidad en la porcelana, donde los defectos son preestablecidos por el personal de conocimiento óptimo de estos como: asimetrías, curvas, bordes deformados, color degradado, fugas de esmalte de huellas de retoque, descamación, fisuras, grietas, dentadura, arañazos, etc. y posteriormente se realiza el procesamiento habitual a las imágenes obtenidas de estos: cambio a escala de grises, redimensionamiento, etc. Enseguida se ingresan a un algoritmo de aprendizaje profundo que trabaja sus capas con el siguiente orden jerárquico $\text{pixel} \rightarrow \text{borde} \rightarrow \text{textura} \rightarrow \text{motivo} \rightarrow \text{parte} \rightarrow \text{objeto}$, una vez que se cuenta con la información y se procesa pasa al segundo paso el cual consiste en la recuperación y el reconocimiento logrando un dominio específico del defecto

capturado de esa manera se puede efectuar el tercer paso el cual consiste en la decisión y reutilización del material de ser posible, su aplicación en línea se presenta de la siguiente manera:

1. El producto llega al sistema de inspección
2. El sensor detecta su presencia y hace llegar una señal al sistema de visión artificial para su activación.
3. Se ilumina el producto.
4. El sistema de visión artificial recibe la imagen
5. El algoritmo de visión artificial se ejecuta realiza el proceso del sistema y analiza la imagen recibida según lo mencionado anteriormente.
6. El sistema de visión envía la señal a un robot que actúa como desviador del producto defectuoso
7. El operador humano revisa el producto rechazado, las posibles causas y aplica acciones correctivas a estas.

este sistema permite realizar el proceso en tiempo real y a alta velocidad obteniendo ventajas que no se encuentran en la forma manual [\[Adriana Birlutiu, 2017\]](#).

Finalmente nos encontramos con una aplicación directa en las tuberías sanitarias de PVC esto mediante la caracterización acústica en ellas para la detección de grietas aplicando el análisis y dominio de las frecuencias producidas en este desarrollo, se aplicó una teoría de comparación entre la presión que se transmitía a la tubería y la que se revivía de esta, aplicando ondas acústicas, el modelo propone la propagación de ondas planas a través de la pared del tubo que se miden como niveles de presión sonora en decibelios, con esto se puede establecer una pérdida de señal que estará dada por grietas en la tubería, es decir la tubería con algún defecto o grieta producirá una pérdida en la presión sonora de la onda acústica, esta aplicación esta propuesta para sistemas de alcantarillado ya instalados en ciudades o industrias, sin

embargo fue probado en muestras de tubería que no se encontraban en los sistemas de alcantarillado y los resultados nos reflejaron que la frecuencia acústica tuvo una gran pérdida de presión sonora cuando se trataba de una tubería agrietada, con lo que se podría intuir que la detección entre una y otra sería clara [Patil, 2018].

Retomando los defectos que generan un impacto principal ya están seleccionados (rebaba, hoyos y manchas) es posible profundizar las técnicas ya aplicadas para estos tipos en diferentes ámbitos y alcanzar una mejor visión del panorama.

1.1. Detección de rebaba, hoyos y manchas

La rebaba y huecos se localiza en un amplio rango de productos si nos referimos por ejemplo a los frenos de disco una variación de altura es considerada como rebaba si es al exterior o como hueco si es al interior y entre las técnicas de detección encontramos aquella que es mediante sensores de triangulación láser como los mostrados en la figura 1.1 estos son diseñados para medir superficies amplias en un solo escaneo, proyectando una línea estrecha de luz sobre la superficie que aparenta tener una distorsión producida por la perspectiva de la cámara como se ve en el diseño de la figura 1.2 de ahí podemos obtener los componentes de las dimensiones del objeto escaneado, con apoyo de la librería Common Vision Blox de Stemmer imaging que nos permite el control de los sensores donde se desarrollan tres algoritmos para el análisis de diferentes zonas: esquinas, planas y especiales es decir aquellas que por su particularidad no son esquinas o planas, como podría serlo un chafalán. [Barrero, 2021] con relación a las técnicas implementadas para la detección de manchas uno muy específico es el análisis de objetos traslucidos mediante la visión por computadora, para el estudio del objeto se genera un ambiente controlado minimizando la iluminación exterior para lograr un mejor control de la imagen de la cual se extraen características básicas como lo es el perímetro, la orientación, longitud, bordes, esquinas, dilatación y algoritmos de erosión, la clasificación la ejecuta a través de estadísticos y medición de objetos [Barrero, 2019].

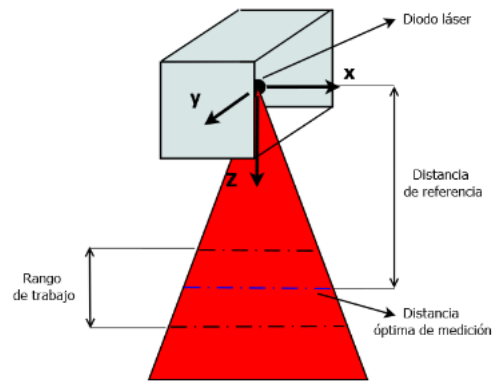


Figura 1.1: Características del sensor de triangulación láser, [Barrero, 2021](#)

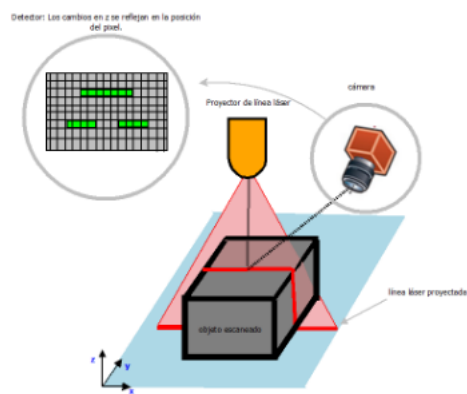


Figura 1.2: Esquema explicativo del funcionamiento de un sistema de triangulación láser, [Barrero, 2021](#)

1.2. Momentos invariantes de Hu

Los momentos invariantes de Hu que nos aportan la descripción de las imágenes son utilizados por [Ochoa et al., 2013] para clasificar objetos rígidos en conjunto con la herramienta para reconocimiento de patrones K-means, entre los objetos a clasificar se realizaron las siguientes pruebas:

- Prueba 1: Clasificación entre llaves y monedas.
- Prueba 2: Clasificación entre pernos y tuercas.
- Prueba 3: Clasificación entre herramientas de trabajo que incluían pinzas, dos martillos, dos cucharas y dos serruchos.
- Prueba 4: Clasificación de número que incluían el uno, tres y cinco.
- Prueba 5: Clasificación de figura que comprende estrellas, círculos, rectángulos y triángulos.

Las pruebas obtienen un porcentaje promedio de clasificación correcta del 97.76 % y un porcentaje de error del 2.24 % [Ochoa et al., 2013].

Otra de las aplicaciones de los momentos de Hu esta en el reconocimiento de dígitos que tiene aplicación en el apoyo de personas invidentes donde el reconocimiento se realiza a través de un celular, los momentos de Hu toman un papel importante debido a que es necesario que el algoritmo no se vea afectado por la rotación o escalamientos del dígito, es decir que las características del número no se vean afectadas en cuanto a cambio de posiciones o tamaños [Gómez and Gutierrez, 2011]. Para la experimentación [Gómez and Gutierrez, 2011] implementa dos tipos de vectores descriptores los momentos de Hu ya mencionados y adicional los descriptores de Flusser, en la figura 1.3 podemos observar los resultados de ambas experimentaciones. Se detecta una confusión en la clasificación del 6 y el 9 debido a la simetría entre ambos números y la invariación a la rotación que presentan sus vectores descriptores, se agrega adicional momentos complejos para solucionar el problema llegando a los resultados de la figura 1.4.

*	Matriz de covarianza semi-compartida	
Caracter	Hu	Flusser
0	99.34 %	99.21 %
1	100.00 %	100.00 %
2	100.00 %	100.00 %
3	100.00 %	100.00 %
4	100.00 %	100.00 %
5	97.97 %	99.32 %
6	91.65 %	86.96 %
7	100.00 %	100.00 %
8	85.53 %	92.97 %
9	89.52 %	81.06 %
Total	96.38 %	95.83 %

Figura 1.3: Precisión del algoritmo de reconocimiento de dígitos mediante Hu y Flusser, [Gómez and Gutierrez, 2011](#)

*	Matriz de covarianza semi-compartida	
Caracter	Hu	Flusser
6	99.47 %	99.48 %
9	99.49 %	96.72 %

Figura 1.4: Precisión de los dígitos 6 y 9 luego de la corrección, [Gómez and Gutierrez, 2011](#)

1.3. Redes Neuronales

Alan Turing estudio por primera vez en 1936 el cerebro como un forma de ver el mundo de la computación [Rivas and Mazón, 2017], sin embargo no fue hasta 1943 donde Warren McCulloch y Walter Pitts un neurofisiologo y un matemático respectivamente entablaron los primeros fundamentos de la computación neuronal, lanzando un teoría acerca de la forma de trabajar de la neuronas [Matich, 2001] y modelaron una red neuronal simple mediante circuitos eléctricos asimismo crearon un modelo computacional para redes neuronales basadas en matemática y algoritmos denominados lógica de umbral [Steemit, 2019]. En 1949 Donal Hebb explicó los procesos del aprendizaje desde el punto de vista psicologico diciendo que el aprendizaje ocurría cuando ciertos cambios en una neurona eran activados [Matich, 2001] también argumento que si dos nervios se disparan al mismo tiempo mejora su conexión [Steemit, 2019], en 1950 se logro simular una red neuronal hipotética realizada por Nthaniel Rochester perteneciente a los laboratorios de IBM y a aunque esta fallo, en 1956 el proyecto de investigación de verano de Dartmouth sobre inteligencia artificial proporciono un gran impulso al ámbito [Steemit, 2019]. Frank Rodenblatt comenzó en 1957 el desarrollo del Perceptron un método capaz de aprender una serie de patrones y relacionar los patrones similares aunque no se encontraran en el entrenamiento a pesar de esto tenia limitaciones debido a su incapacidad de clasificar clases no separables linealmente [Matich, 2001], entre 1959 y 1960 Bernard Widrow y Marcian Hoff crearon a Adaline un sistema lineal y adaptivo en la que se implementaban dos capas, llamada Madaline usadas para el reconocimiento de voz y la eliminación de ruido y eco de las línea telefónicas [Rivas and Mazón, 2017]. 1969 fue un año grave para las redes neuronales ya que Marvin Minsky y Seymour Papert probaron matemáticamente que Perceptron no era capaz de resolver problemas relativamente simples como aquellos de clases no separables linealmente, más delante se propuso capas adicionales para la solución de este probema, Paul Werbos desarrollo en 1974 la idea básica del algoritmo de aprendizaje backpropagation o propagación hacia atrás [Matich, 2001], no obstante en 1986 David Rumelhart y G.

	Perceptrón Multicapa	
	Textura	RGB
c_1 (agua)	95%	91%
c_2 (rocas)	83%	82%
c_3 (arbustos)	86%	81%
c_4 (hierbas secas)	97%	96%

Figura 1.5: Porcentaje de clasificación correcta por textura para

PM, [Pajarez and Moreno, 2001]

Hinton redescubrieron el back propagation [Matich, 2001] y como resultado se obtuvo aunque un aprendizaje mas lento un resultado más preciso [Rivas and Mazón, 2017]. Partiendo de este suceso las redes neuronales han logrado múltiples avances y aplicaciones en la clasificación de información, gestión bibliotecaria, diseño de interfase, filtrado de información, búsqueda incompleta, descubrimiento de información , entre otros [de Moya Anegón et al., 1998].

1.3.1. Perceptron multicapa

Perceptron multicapa es un red neuronal que ha tenido aplicaciones en el reconocimiento de patrones, en la biología para el estudio de cerebros u obtención del modelo de retina, en el medio ambiente para el pronóstico del tiempo, finanzas pre- visión de la evolución e los precios, identificación de firmas , aplicaciones militares, clasificación de señales, creación de armas inteligentes hasta la detección de fugas hidráulicas [Vidal, 2015] no obstante las aplicaciones de nuestro interés son aquellas que se desarrollan en la clasificación y en la búsqueda de ejemplos más particulares encontramos la clasificación de texturas encontradas en las estructuras naturales en imágenes aéreas y de satélite, se obtienen las propiedades a través de la homogeneidad, correlación, entropía y de la transformada de fourier que nos entrega la magnitud máxima, magnitud media , energía de la magnitud y varianza de la magnitud finalmente de los filtros de gabor modificando la frecuencia y orientaciones, estos vectores obtenidos son ingresados a PM donde se llega a los resultados de la figura 1.5 [Pajarez and Moreno, 2001].

Perceptron multicapa como ya se mencionó también se aplica en el reconocimiento de patrones, como lo es la clasificación de patrones en conductas adictivas que incluye el consumo de drogas en este estudio el vector característico se compone de datos que describen al adicto, sus resultados en algunos test y conductas del mismo y estos incluyen la edad, puntuación en el test de depresión de Beck , uso de alguna droga en los últimos meses(cocaína o heroína), historial del uso de drogas, número de tratamientos previos, raza, tiempo del tratamiento , lugar del tratamiento y tiempo de recaída del sujeto por debajo o encima de los 5 meses como resultado de ingresar estos vectores a la red con diferentes muestras y diferentes características prioritarias logra una eficacia promedio del 74.5 % superando la mayoría de los estudios encontrados con anterioridad que estaban por debajo del 70 % [\[Vidal, 2015\]](#).

1.4. Algoritmos genéticos

En 1970 [\[Gajawada, 2012\]](#) John H. Holland, sus colegas y sus estudiantes de la universidad de Michigan desarrollaron la teoría de los algoritmos genéticos en un proyecto que deseaba explicar el proceso de adaptación natural de los sistemas y diseñar sistemas artificiales que conservaran los mecanismos más importantes de los sistemas naturales [\[alvarez et al., 2010\]](#). Marco Dorigo en 1992 propuso la optimización de colonias de hormigas , Kennedy y Eberhart en 1995 introdujeron la optimización de enjambres de partículas, por su parte Storn y Price establecieron en 1996 el algoritmo de evolución diferencial, en 2002 K.M. Passino estableció el algoritmo de optimización de alimentación bacteriana y en 2003 X.L. Li propone el algoritmo del enjambre de peces artificiales poco mas adelante en 2005 Karaboga introduce el algoritmo de colonia artificial de abejas, todos estos algoritmos fueron desarrollados a través del comportamiento de diversas especies naturales para lograr una optimización en la resolución de problemas complejos [\[Gajawada, 2012\]](#).

1.4.1. Evolución diferencial

Evolución diferencial es un algoritmo genético que nos apoya en la optimización de la selección de pesos de la red neuronal, se buscaron aplicaciones similares donde encontramos el uso de evolución diferencial con los algoritmos de reconocimiento de patrones k-NN y mínima distancia para la predicción de la estructura secundaria de proteínas a través de la evolución de características de aminoácidos, Las proteínas son moléculas encargadas de una gran variedad de funciones en el cuerpo como [Rivas, 2022](#):

- Enzimas: Actúan como catalizadores biológicos en reacciones químicas.
- Estructurales: Dan estructura y soporte a la célula.
- De defensa: Protegen al organismo contra invasiones externas.
- Reguladoras: Controlan actividades fisiológicas o celulares , entre ellas están las hormonas.
- Transporte : Enlazan y llevan sustancias de un lugar a otro.

Las proteínas están formadas por aminoácidos que se van uniendo unos a otros mediante enlaces [Rivas, 2022](#). Los aminoácidos se clasifican por su radical de la siguiente forma [Rivas, 2022](#):

- No polares: Contiene radicales que lo hacen hidrófobos.
- Polares: Los radicales de este grupo tienen elementos que pueden realizar enlaces de puente de hidrógeno con agua y otras moléculas.
- Ácidos: Los elementos de su radical hacen que sean donadores de protones por lo que tienen carga negativa
- Básicos: Estos aceptan protones y tiene carga positiva.

La estructura secundaria de las proteínas son las estructuras regulares que se forman de manera local a lo largo de la cadena y se debe principalmente a los puentes de

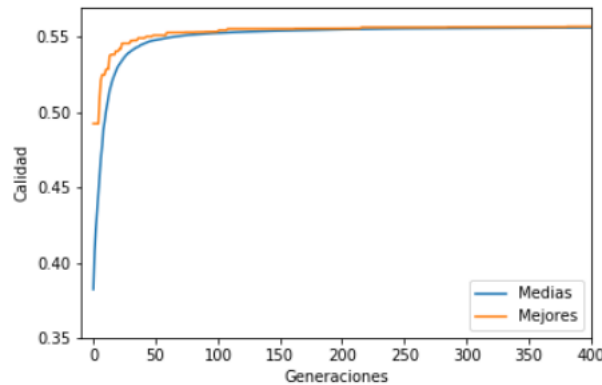


Figura 1.6: Calidad media y mejor de la población a lo largo de las generaciones, con 1 propiedad y RS126, [Rivas, 2022]

hidrógeno entre aminoácidos cercanos [Rivas, 2022]. Se pretende que a través de las características de los aminoácidos se perdiga la estructura secundaria de la proteína. Evolución diferencial logra el mejor porcentaje de clasificación correcto alrededor de 55.57% y este converge alrededor de la generación 200, a pesar de que se ejecutó durante 1500 generaciones en la figura 1.6 mostrada solo vemos 400 por su rápida convergencia.

Otra aplicación de evolución diferencial es el ajuste de parámetros de una carga estocástica, la carga es representada por un horno de arco eléctrico conocido como Electric Arc furnace o EAF este es una carga no lineal y altamente variable que origina problemas de calidad de energía, para encontrar los parámetros adecuados A (suma de las caídas de tensión del ánodo y del cátodo), B (caída de tensión por unidad de longitud), C_a, C_b, D_a, D_b (C y D son constantes cuyos valores determinan la diferencia entre el aumento y disminución de las partes actuales de la característica tensión-corriente), l_o (longitud del arco de referencia) y r (señal de ruido blanco) se usa evolución diferencial en las señales reales de tensión y corriente, una vez que se cuenta con estos parámetros se simula el modelo matemático, obteniendo una señal real y una simulada que gracias al error=0.0018761 que es bastante bajo las señales se superponen bastante bien [es Murillo Arbel aez and Medina, 2019].

En conclusión, a lo investigado se cuenta con diversas fuentes de apoyo de sistemas de

calidad y de detección automática de defectos, aunque no se encontró directamente una aplicación en conexiones sanitarias, los métodos empleados en diversos ámbitos y en la producción de tuberías realizaron grandes aportes a la investigación de las posibles aplicaciones y a los probables métodos que podrán ser aplicados al proyecto propuesto.

Capítulo 2

Marco Teórico

2.1. Procesamiento de imágenes

Una imagen está construida mediante un arreglo bidimensional de píxeles cada uno con una intensidad luminosa, estas pueden estar constituidas en una escala de grises o también en una combinación de tres colores: rojo, verde y azul (RGB) [Maestre, 2005]. Podemos definir un mapa de color como una matriz de $n \times 3$ donde los renglones son una tripleta de colores, el primer renglón como el valor mínimo del eje de color al último renglón como el máximo, cuando se definen diferentes distribuciones de intensidad de los tres colores son creados diferentes mapas de color [Maestre, 2005]. El tratamiento o procesamiento de imágenes juega un gran papel en la visión por computadora ya que de este depende desde el realce de atributos del objeto de interés hasta la calidad de los datos obtenidos para el entrenamiento de nuestro sistema, este se puede realizar aplicando diversos métodos.

2.1.1. Intensidad de color en una imagen

La intensidad de color de una imagen es la cantidad de color que tiene es decir el nivel de color o nivel de croma o saturación [Grob, 1990].

2.1.2. Ruido en imágenes

Las imágenes suelen tener una cantidad de ruido que se ve representada en algunos de sus píxeles cuando estos toman un nivel de gris diferente al de sus vecinos, esto se puede deber al medio por el cual fueron capturadas, pero es posible reducirlo y tratarlo con métodos de filtrado, encontramos principalmente tres tipos de ruido: El Gaussiano el cual es aquel que produce pequeñas variaciones en la imagen, el impulsional donde el valor que toma el píxel no se relaciona con el valor ideal, si no con el valor del ruido con valores muy altos o bajos que se presentan como puntos blancos o negros y finalmente el multiplicativo donde la imagen se obtiene a través de la multiplicación de dos señales [Maestre, 2005].

2.1.3. Segmentación de imágenes

La binarización de una imagen se obtiene cuando se comparan los niveles de gris que presenta la misma con un valor predeterminado el cual se establece como umbral donde si el nivel de gris de la imagen es menor al umbral a dicho píxel es asignado el valor de 0 es decir el píxel pasa a negro y si es mayor al umbral es asignado a 1 es decir el píxel pasa a blanco logrando tener ahora una imagen en blanco y negro [Maestre, 2005].

2.1.4. Adición de imágenes

En las imágenes también es posible realizar operaciones aritméticas entre las cuales se encuentran la suma, resta, multiplicación y división, estas operaciones son posibles de realizar cuando las imágenes tienen el mismo tamaño [Maestre, 2005].

Podemos agregar un escalar a una imagen, esto da como resultado una aclaración de la imagen y se hace sumando el escalar a cada uno de los píxeles de la imagen donde si el resultado es mayor a 255 se toma como valor resultante 255 [Iván García, 2015]. La suma de dos imágenes se ejecuta sumando un píxel más un píxel que se encuentra en

la misma ubicación de las imágenes y se representa en la forma [\[Iván García, 2015\]](#):

$$A(x, y) + B(x, y) = R(x, y) \quad (2.1)$$

2.2. Descriptores de una imagen.

Descriptores de una imagen. Los descriptores son aquellos patrones o vectores que nos permiten caracterizar y obtener las propiedades de una imagen, como su iluminación, color, forma o textura [\[Zambrano, 2018\]](#)

2.3. Funciones Momento.

Los momentos que se calculan a partir de una imagen representan un conjunto de características globales de la forma de la imagen que nos entregan información acerca de las diversas geometrías que la componen, estos tienen una gran cantidad de aplicaciones en el análisis de imágenes como el reconocimiento de patrones, clasificación de objetos, estimación de las posiciones e incluso en codificación de imágenes, visión por computadora, etc [\[Quitl, 2010\]](#). Las propiedades que nos entregan los momentos de una imagen describen valores estadísticos y mecánicos de la imagen, los momentos de orden cero, uno y dos de la función de densidad probabilística (PDF) representan la probabilidad total, la esperanza y la varianza respectivamente. En los momentos de distribución espacial de una masa, podemos encontrar la masa total, posición del centroide y los momentos de inercia [\[Quitl, 2010\]](#). Si vemos una imagen como una distribución de intensidades en dos dimensiones, los momentos geométricos de los valores del pixel con respecto a sus posiciones en el plano, la imagen nos puede dar información del área total de la misma, las coordenadas del centroide y su orientación esto nos puede dar vectores de descriptores invariantes a la traslación, rotación, intensidad y escalamiento [\[Quitl, 2010\]](#).

2.3.1. Momentos Geométricos de una imagen.

El momento(m,n) de probabilidad de densidad conjunta p(x,y) se define[1,3]:

$$M(m, n) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^m y^n p(x, y) dx dy \quad (2.2)$$

donde m,n=0,1,2,3,.. y representa el orden del momento[2] Los momentos geométricos de diferentes ordenes representan diferentes características espaciales de la distribución de intensidades de la imagen[2] El momento de orden cero m_00 representan la intensidad total de una función imagen y en el caso de una imagen binaria nos entrega el área[Quitl, 2010].

$$m_00 = \int \int_{\zeta} f(x, y) dx dy \quad (2.3)$$

Los momentos de primer orden m_10 y m_01 nos entregan la intensidad alrededor del eje x y el eje y respectivamente en la imagen , el centroide o centro de gravedad de la imagen[1,3] (x_0, y_0) se define como[Quitl, 2010]

$$x_0 = \frac{m_10}{m_00}, y_0 = \frac{m_01}{m_00} \quad (2.4)$$

Los momentos calculados con respecto a la intensidad del centroide son llamados momentos centrales y están definidos como[Zambrano, 2018, Lara, 2012]

$$U(m, n) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x - \bar{x})^m (y - \bar{y})^n p(x, y) dx dy \quad (2.5)$$

Donde \bar{x} y \bar{y} son los promedios marginales de P(x,y)[Zambrano, 2018] si F(j,k) representa una función imagen discreta o la distribución de intensidades de una imagen digital reemplazamos la sumatoria continua por una sumatoria espacial teniendo el momento espacial geométrico de una imagen como[Zambrano, 2018, Lara, 2012]:

$$M_u(m, n) = \sum_{j=1}^J \sum_{k=1}^K (x_j)^m (y_k)^n F(j, k) \quad (2.6)$$

Donde F(j,k) es el valor de intensidad de un pixel en las coordenadas (j,k), los momentos espaciales de bajo orden se representan como[Zambrano, 2018]

$$M_u(0,0) = \sum_{j=1}^J \sum_{k=1}^K F(j,k) \quad (2.7)$$

$$M_u(0,1) = \sum_{j=1}^J \sum_{k=1}^K (y_k)^n F(j,k) \quad (2.8)$$

$$M_u(1,0) = \sum_{j=1}^J \sum_{k=1}^K (x_j)^m F(j,k) \quad (2.9)$$

Donde ?? es denominada superficie de imagen, [2.7](#) momento fila de primer orden, [2.8](#) momento columna de primer orden.

Los momentos de inercia de segundo orden de la imagen se definen [Zambrano, 2018](#):

$$U(0,2) = \sum_{j=1}^J \sum_{k=1}^K (x - \bar{x})^m (y - \bar{y})^n p(x,y) dx dy \quad (2.10)$$

$$U(2,0) = \sum_{j=1}^J \sum_{k=1}^K (x - \bar{x})^m (y - \bar{y})^n p(x,y) dx dy \quad (2.11)$$

$$U(1,1) = \sum_{j=1}^J \sum_{k=1}^K (x - \bar{x})^m (y - \bar{y})^n p(x,y) dx dy \quad (2.12)$$

Donde [2.9](#) momento de inercia de fila, [2.10](#) momento de inercia de columna, [2.11](#) momento de inercia fila-columna.

2.4. Momentos Invariantes de Hu

Los momentos invariantes nos apoyan a reconocer los objetos aunque no se encuentren en la misma posición y parten de los momentos geométricos ordinarios , Hu propuso la normalización de los momentos centrales a través de [Zambrano, 2018](#), [Lara, 2012](#):

$$V(m,n) = \frac{U(m,n)}{[M(0,0)]^\alpha} \quad (2.13)$$

Donde:

$$\infty = \frac{m+n}{2} + 1 \quad (2.14)$$

para $m+n= 2,3,\dots$

esto nos llevo a siete momentos espaciales que se describen como:

$$h_1 = V(2, 0) + V(0, 2) \quad (2.15)$$

$$h_2 = [V(2, 0) - V(0, 2)]^2 + 4[V(1, 1)]^2 \quad (2.16)$$

$$h_3 = [V(3, 0) - 3V(1, 2)]^2 + [V(0, 3) - 3V(2, 1)]^2 \quad (2.17)$$

$$h_4 = [V(3, 0) - V(1, 2)]^2 + [V(0, 3) - V(2, 1)]^2 \quad (2.18)$$

$$\begin{aligned} h_5 = & [V(3, 0) - 3V(1, 2)][V(3, 0) + V(1, 2)][[V(3, 0) + V(1, 2)]^2 \\ & - 3[V(0, 3) + V(2, 1)]^2 + 3V(2, 1) - V(0, 3)][V(0, 3) + V(2, 1)] \\ & [3[V(3, 0) + V(1, 2)]^2 - [V(0, 3) + V(2, 1)]^2] \end{aligned} \quad (2.19)$$

$$\begin{aligned} h_6 = & [V(2, 0) - V(0, 2)][[V(3, 0) + V(1, 2)]^2 - [[V(3, 0) + V(1, 2)]^2 \\ & - 4V(1, 1)[V(3, 0) + V(1, 2)][V(0, 3) + V(2, 1)] \end{aligned} \quad (2.20)$$

$$\begin{aligned} h_7 = & [3V(2, 1) - V(0, 3)][V(3, 0) + V(1, 2)][[V(3, 0) + V(1, 2)]^2 \\ & - 3[V(0, 3) + V(2, 1)]^2] + [3V(1, 2) - V(3, 0)][V(0, 3) + V(2, 1)] \\ & [3[V(3, 0) + V(1, 2)]^2 - [V(0, 3) + V(2, 1)]^2] \end{aligned} \quad (2.21)$$

este conjunto de momentos h_1, h_2, \dots, h_7 son invariantes a la traslación rotación y escala [Quitl, 2010, Lara, 2012]

2.5. Redes Neuronales.

Las redes neuronales biológicas son un conjunto de células nerviosas entrelazadas de manera natural para conformar el sistema neuronal en el cerebro humano, las células neuronales tienen un proceso de comunicación con otras células neuronales a diferencia del resto de las células en el cuerpo humano estas son capaces de recibir, procesar y transmitir información importante a nuestro organismo, este proceso es la llave del aprendizaje humano ya que al interactuar con el exterior y comenzar la estimulación de los sentidos genera un intercambio de señales electroquímicas que apoyan al aprendizaje [Restrepo et al., 2021], Las redes neuronales artificiales (RNA) son modelos computacionales basados en los principios de aprendizaje del cerebro humano [Flores and Fernández, 2008]. Las RNA están formadas por procesadores elementales llamados neuronas artificiales que realizan una serie de cálculos a partir de un vector de entrada o bien a partir de la entrega de información de otras neuronas a través de esto nos proporcionan una salida, teniendo así tres tipos de neuronas [Flores and Fernández, 2008]:

- Neuronas de entrada: Reciben señales o vectores de información desde el entorno.
- Neuronas ocultas: Reciben estímulos y emiten salidas dentro del sistema, en ellas se genera el procesamiento de la información.
- Neuronas de salida: Envían la señal fuera del sistema una vez que se termina el procesamiento de la información.

Las neuronas artificiales se caracterizan por los siguientes elementos [Flores and Fernández, 2008]:

- Un valor de activación inicial.
- Entradas a la neurona (X_i), con sus respectivos pesos asociados V_{ij} .
- Una función de propagación.
- Una función de activación (f).

- Una función de salida (F).
- Una señal de salida que transmite.
- Una regla de aprendizaje que determina la forma de la actualización de los pesos de la red.

2.5.1. La arquitectura de las Redes Neuronales Artificiales.

La arquitectura de las RNA se refiere a la organización y disposición de las neuronas en la red firmando capas de procesadores interconectados esta depende de 4 parámetros principales: el número de capas del sistema, el número de neurona por capa, el grado de conectividad de as neuronas y el tipo e conexiones, y se pueden clasificar en diferentes criterios, para este caso en particular nos interesan dos criterios [\[Flores and Fernández, 2008\]](#):

- Según su estructura en capas:
 - Redes monocapa: Red compuesta por una única capa de neuronas entre las que se establecen conexiones laterales, figura [2.1](#).
 - Redes multicapa: Red compuesta por varias capas de neuronas (de entrada, ocultas y de salida). La capa a la que pertenece la neurona se distingue según el origen de las señales que recibe y el destino de la señal que entrega.
- Según el flujo de datos en la red:
 - Redes unidireccionales o de propagación hacia adelante(feedforward): En las que ninguna salida neuronal es entrada de unidades de la misma capa o de capas precedentes. La información circula en un único sentido , desde las neuronas de entrada hacia las neuronas de salida de la red, figura [2.2](#).
 - Redes de propagación hacia atrás(feedback): Red en las que las salidas de las neuronas pueden servir de entradas a unidades del mismo nivel o de niveles previos, figura [2.3](#).

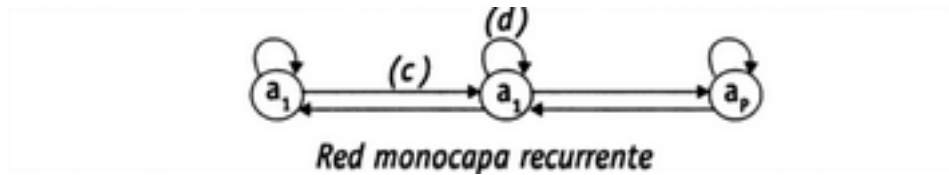


Figura 2.1: Red monocapa recurrente, donde (a) conexiones hacia adelante, (c) conexiones laterales y (d) conexiones autorecurrentes. [Flores and Fernández, 2008]

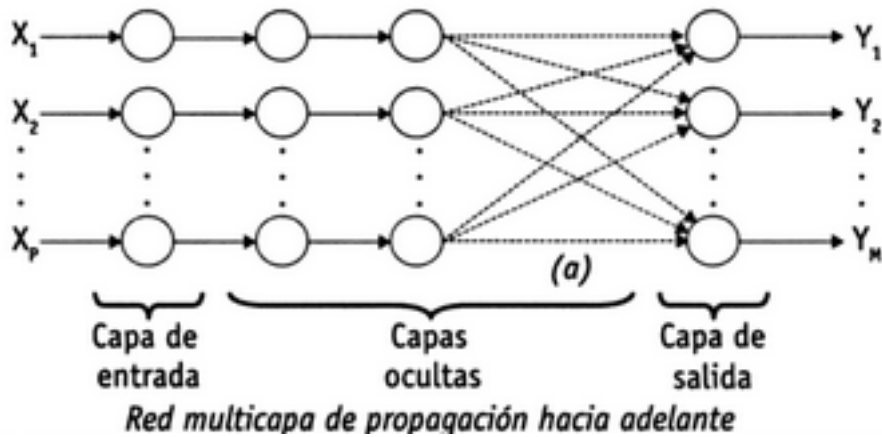


Figura 2.2: Red multicapa de propagación hacia adelante, donde (a) conexiones hacia adelante. [Flores and Fernández, 2008]

2.5.2. Perceptron multicapa.

Perceptrón multicapa (MLP) es un modelo de red neuronal con "propagación hacia adelante", que se compone de la capa de entrada, una o varias capas intermedias u ocultas y una capa de salida [Flores and Fernández, 2008].

El proceso de aprendizaje de MLP se basa en encontrar una función que represente correctamente los patrones de aprendizaje además de llevar a cabo un proceso de generalización que permita tratar individuos desconocidos, esto realizando el ajuste de un vector de pesos V a partir de la información de la muestra. Teniendo el conjunto de patrones muestrales $(x_1, y_1), (x_2, y_2), \dots, (x_p, y_{1p})$ y una función de error $\epsilon(W, X, Y)$ el entrenamiento se basa en la búsqueda de n conjunto de pesos que disminuyan el error de aprendizaje $E(V)$ [Flores and Fernández, 2008]:

$$\min_W E(W) = \min_W \sum_{i=1}^p \epsilon(W, x_i, y_i) \quad (2.22)$$

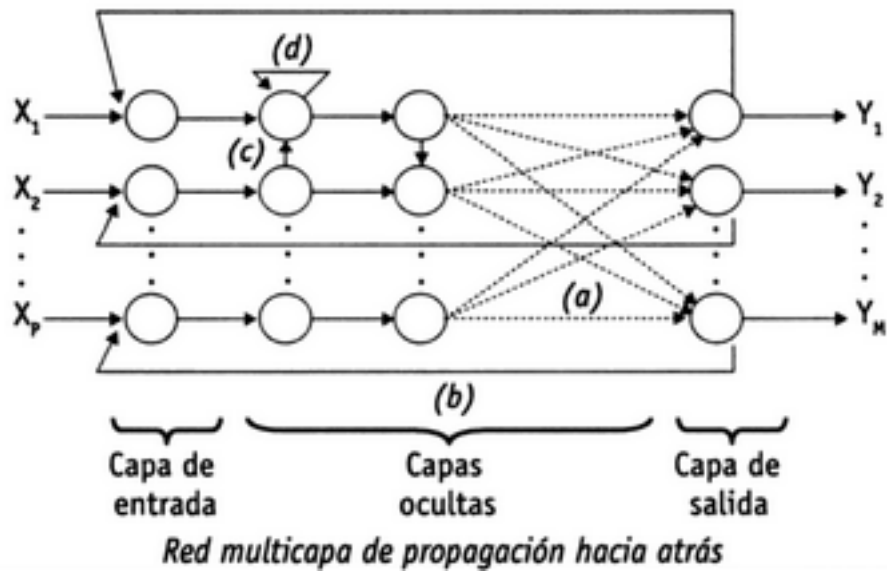


Figura 2.3: Red multicapa de propagación hacia atrás, donde (a) conexiones hacia adelante, (b) conexiones hacia atrás, (c) conexiones laterales y (d) conexiones autorecurrentes. [Flores and Fernández, 2008]

Desarrollo de Perceptron Multicapa.

Perceptron multicapa esta compuesta de su bias es decir el parámetro de ajuste que permite a las RNA ubicar correctamente la frontera de decisión en el hiperplano de soluciones [Restrepo et al., 2021], sus vectores de pesos (V_{ij} y W_{ij}) y sus 3 capas [CYTED and CONICIT, 1999] y [Arahal et al., 2006]:

- Capa de entrada $X_i]_{(i=1)}^n$
- Capa oculta $Z_j]_{(j=1)}^L$
- Capa de salida $Y_k]_{(k=1)}^m$

Podemos especificar las etapas de MLP como.

- Etapa 1: Feedforward.

En la etapa 1 definimos los vectores de pesos V_{ij} y W_{ij} y asignamos los vectores de entrada X_i es decir aquellos valores que nos entrega nuestro entorno a evaluar adicionando el bias, también es necesario contar con la función de

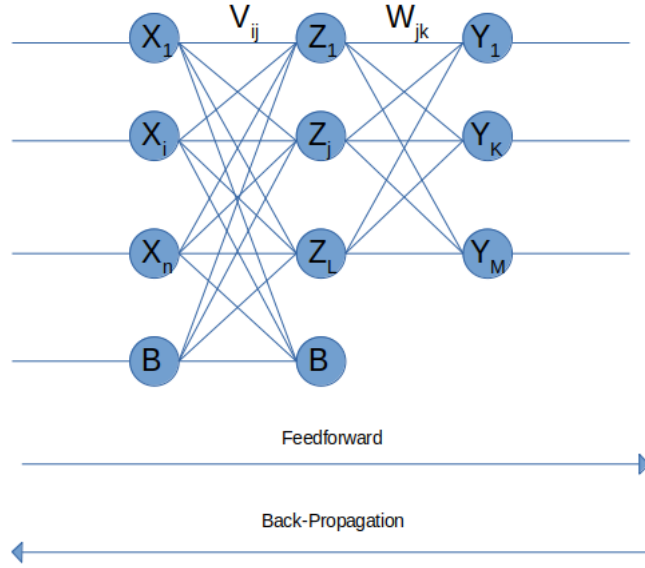


Figura 2.4: Diagrama de la red MLP.

activación $f(x)$ y su derivada $f'(x)$, la función de activación mas común es la función sigmoide [2.23](#) [\[Ana, 2022\]](#) y [\[Vicente et al., 2019\]](#).

$$f(x) = \frac{1}{1 + \exp(-x)} \quad (2.23)$$

La derivada de esta función es [2.24](#) [\[Ana, 2022\]](#) y [\[Vicente et al., 2019\]](#).

$$f'(x) = f(x)[1 - f(x)] \quad (2.24)$$

Ya que contamos con estos valores calculamos los vectores de entrada a la capa intermedia Z (Z_{in}) con la ecuación [2.25](#) y los pesos V [\[Ana, 2022\]](#) y [\[Vicente et al., 2019\]](#).

$$Z_{inj} = \sum_{i=1}^n V_{ij} X_i + V_{oj} \quad 1 \leq j \leq l \quad (2.25)$$

Para calcular los vectores de salida que entregara la capa intermedia utilizamos la función [2.26](#) [\[Ana, 2022\]](#) y [\[Vicente et al., 2019\]](#).

$$Z_j = f(z_{inj}) \quad 1 \leq j \leq l \quad (2.26)$$

El siguiente paso consiste en calcular los vectores de entrada de la capa de salida esto se realiza con el vector de pesos W y el vector de salida de la capa intermedia Z aplicando la función [2.27](#) [Ana, 2022](#) y [Vicente et al., 2019](#).

$$Y_{ink} = \left[\sum_{j=1}^l W_{jk} Z_j \right] + W_{ok} \quad 1 \leq k \leq m \quad (2.27)$$

La capa de salida nos entregara un valor calculado con la función [2.28](#), este valor de salida es aquel que nos apoya a realizar el cálculo del error por patron generado hasta este punto además identifica la clasificación que esta asignando la RNA sin embargo no sera tomado en cuenta hasta completar la regla de ajuste de pesos [Ana, 2022](#) y [Vicente et al., 2019](#).

$$Y_k = f(Y_{ink}) \quad 1 \leq k \leq m \quad (2.28)$$

A continuación teniendo Y_k podemos calcular el error cuadrático que presenta para cada patrón con la ecuación [2.29](#) [Ana, 2022](#) y [Vicente et al., 2019](#).

$$E^{p=1} = (t_p - Y_p)^2 \quad (2.29)$$

- Etapa 2: Back-propagation.

En a etapa 2 calculamos los vectores que nos apoyaran a ajustar los pesos en la siguiente etapa y comenzamos nuestros cálculos invirtiendo las capas es decir de la capa de salida a la intermedia y luego de la capa intermedia a la entrada [Ana, 2022](#) y [Vicente et al., 2019](#).

Para la capa de salida a intermedia [Ana, 2022](#) y [Vicente et al., 2019](#):

$$\delta_k = (t_k - y_k) F'(y_{ink}) \quad 1 \leq k \leq m \quad (2.30)$$

$$\Delta W_{jk} = \alpha \delta_k \quad 1 \leq j \leq l \quad 1 \leq k \leq m \quad (2.31)$$

donde α es a taza de aprendizaje [Ana, 2022](#) y [Vicente et al., 2019](#).

Para la capa intermedia a entrada [Ana, 2022](#) y [Vicente et al., 2019](#).

$$\delta_{inj} = \sum_{k=1}^m \delta_k W_{jk} \quad 1 \leq j \leq l \quad (2.32)$$

$$\delta_{,j} = \delta_{inj} F'(Z_{inj}) \quad 1 \leq j \leq l \quad (2.33)$$

$$\Delta V_{ij} = \alpha \delta_{,j}' X_i \quad 1 \leq i \leq n \quad 1 \leq j \leq l \quad (2.34)$$

$$\Delta V_{oj} = \alpha \delta_{,j} \quad 1 \leq j \leq l \quad (2.35)$$

- Etapa 3: Ajuste de pesos.

La ultima etapa de MLP es la etapa 3 en esta nos encargamos de ajustar los pesos con apoyo de los valores de ajuste calculados en la etapa 2 (ΔW_{jk} , ΔV_{ij} y ΔV_{oj}) con las ecuaciones [2.36](#) y [2.37](#) [Ana, 2022](#) y [Vicente et al., 2019](#).

$$V_{ij(new)} = V - ij(old) + \Delta V_{ij} \quad 0 \leq i \leq n \quad 1 \leq j \leq l \quad (2.36)$$

$$V_{oj} = W_{jk(old)} + \Delta W_{jk} \quad 0 \leq j \leq l \quad 1 \leq k \leq m \quad (2.37)$$

2.6. Algoritmos Genéticos.

Los algoritmos genéticos son técnicas de optimización basadas en la de selección natural y la genética, donde aplican los siguientes principios [Cerrolaza and Annicchiarico, 1996](#).

1. Los procesos de evolución operan sobre los cromosomas estos codifican las estructuras de los seres vivos.
2. La selección natural nos permite que se reproduzcan los cromosomas que codifican estructuras mas exitosas.

3. La reproducción ocurre cuando la evolución toma lugar a través de las mutaciones de los cromosomas.

La finalidad del algoritmo genético es encontrar la mejor solución a través de la búsqueda en una población inicial de posibles soluciones que van evolucionando y mejorando en cada iteración del algoritmo, estas iteraciones son nombradas comúnmente como generaciones donde la última de ellas incluye a la mejor solución de nuestro problema de optimización, podemos definir a cada posible solución como individuo un proceso rápido y sencillo de los algoritmos genéticos lo podemos ver en la figura [2.5](#)

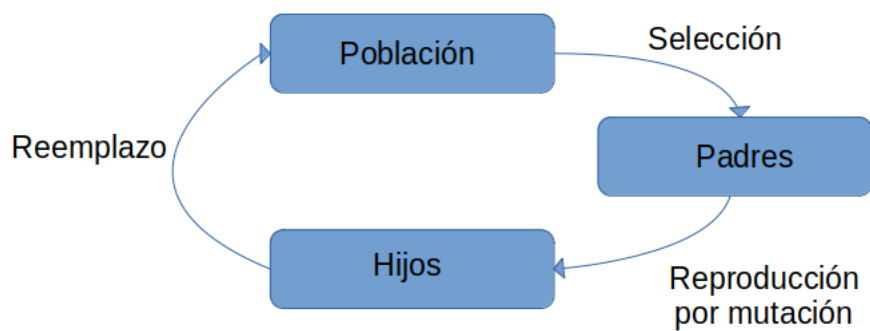


Figura 2.5: Proceso de algoritmo genético mediante selección natural.

2.6.1. Evolución diferencial.

Este algoritmo pertenece a los algoritmos evolutivos es poblacional y funciona con problemas de optimización continua [Cuevas et al., 2016](#).

Este algoritmo necesita las siguientes variables de entrada [Cuevas et al., 2016](#) y [Espinal et al., 2011](#):

- Dimensión del problema (dim)
- Dimensión de la población (N)

- Esquema de mutación.
- Probabilidad de cruce $0 \leq CR \leq 1$

y nos entrega la siguiente variable de salida [Cuevas et al., 2016] y [Espinal et al., 2011]:

- Aproximación a la mejor solución del problema.

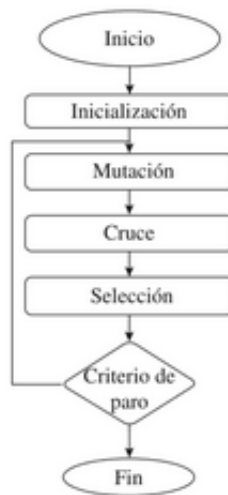


Figura 2.6: Diagrama de flujo genérico de las operaciones llevadas a cabo por el algoritmo evolución diferencial, [Cuevas et al., 2016]

Desarrollo de Evolución Diferencial.

Podemos describir el desarrollo de Evolución diferencial (DE) en cuatro pasos [Cuevas et al., 2016] y [Espinal et al., 2011]:

Paso 1: El primer paso para el desarrollo del algoritmo de evolución diferencial es la generación de los N individuos de la primera población, estos deben tener una distribución de probabilidad uniforme dentro del espacio de búsqueda.

Paso 2: Evaluar cada individuo con la función fitness.

Paso 3: Hacer hasta llegar a un criterio de paro

3.1 Para $i=1$ hasta N :

- Generar un vector mutado en base al esquema de muta seleccionado donde $\bar{X}_m \in \mathbb{R}^{dim}$
- Generar un vector de prueba \bar{X}_t que depende de \bar{X}_m y \bar{X}_i donde \bar{X}_i es el vector actual de mi población.

$$X_{tj} = \begin{cases} X_{mj} & \text{si } rand() < R \\ X_{ij} & \text{encaso contrario} \end{cases} \quad (2.38)$$

- Si $f(x_t) \leq F(x_i)$ entonces $x_i = x_t$

Paso 4: Comprobar condición de paro del paso 3.

Esquemas de mutación para Evolución Diferencial.

Tenemos 5 posibles esquemas de muta para DE [\[Cuevas et al., 2016\]](#) y [\[Espinal et al., 2011\]](#):

1. DE/rand/1

$$\bar{X}_m = \bar{X}_{r1} + F(\bar{X}_{r2} - \bar{X}_{r3})$$

donde $0 \leq F \leq 2, \bar{X}_{r1} \neq \bar{X}_{r2} \neq \bar{X}_{r3} \neq \bar{X}_i$

2. DE/rand/2

$$\bar{X}_m = \bar{X}_{r5} + F(\bar{X}_{r1} - \bar{X}_{r2} - \bar{X}_{r3} - \bar{X}_{r4})$$

donde $\bar{X}_{r1} \neq \bar{X}_{r2} \neq \bar{X}_{r3} \neq \bar{X}_{r4} \neq \bar{X}_{r5} \neq \bar{X}_i$

3. DE/best/1

$$\bar{X}_m = \bar{X}_{best} + F(\bar{X}_{r1} - \bar{X}_{r2})$$

donde $\bar{X}_{r1} \neq \bar{X}_{r2} \neq \bar{X}_i$

4. DE/best/2

$$\bar{X}_m = \bar{X}_{best} + F(\bar{X}_{r1} - \bar{X}_{r2} - \bar{X}_{r3} - \bar{X}_{r4})$$

5. DE/rand to best /1

$$\bar{X}_m = \bar{X}_i + F_1(\bar{X}_{best} - \bar{X}_{r1}) + F_2(\bar{X}_{r2} - \bar{X}_{r3})$$

donde $0 \leq F_1, F_2 \leq 2$

2.7. Diagrama del proyecto

El diagrama general del proyecto es descrito en la figura 2.7 y sigue los pasos mostrados a continuación.

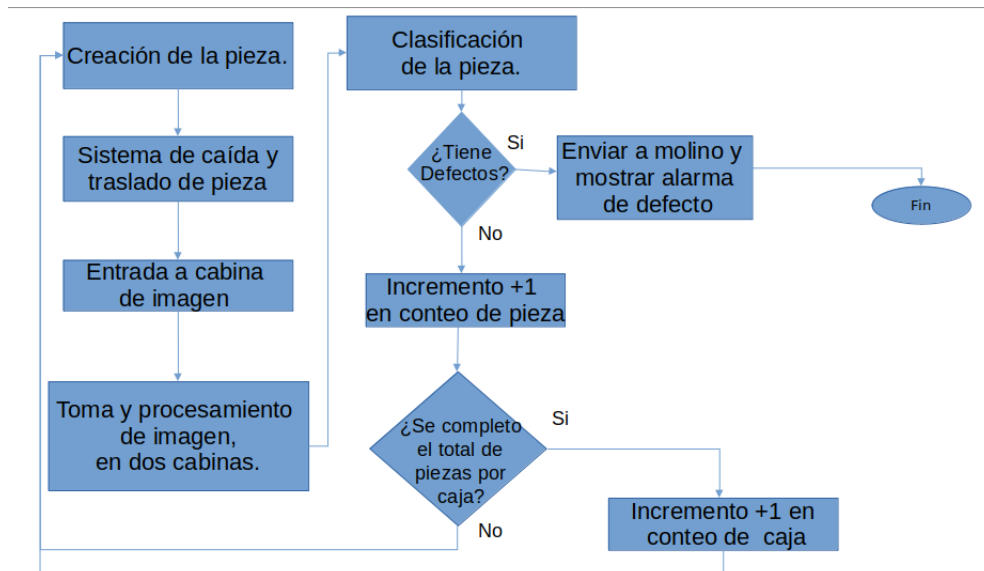


Figura 2.7: Diagrama general del proyecto.

2.7.1. Paso 1.

La pieza es fabricada a través de la máquina inyectora, esta consta de un molde al cual se inyecta material a la temperatura adecuada donde al ser llenado se le permite un tiempo de enfriamiento con ayuda del sistema de enfriamiento y en seguida es abierto para la caída de la pieza formada.

2.7.2. Paso 2.

Sistema de caída y traslado de la pieza, una vez que la pieza cae este se encarga del control de caída y traslado de la pieza, el diseño del mismo podemos observarlo en la imagen 2.8, en el anexo A.1 podemos encontrar el diseño propuesto desde sus diferentes vistas.

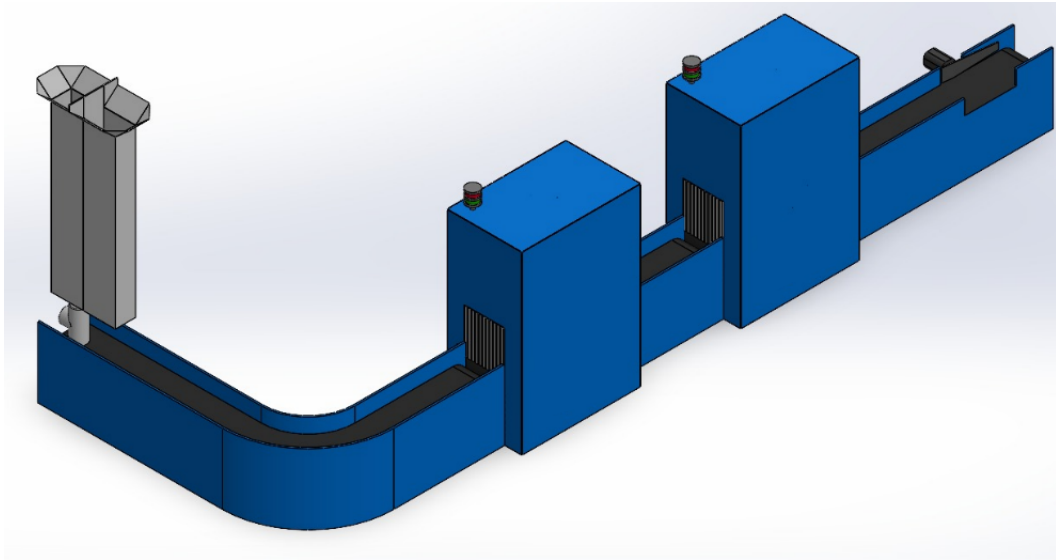


Figura 2.8: Diseño del sistema de caída y traslado.

2.7.3. Paso 3.

Entrada a cabina de imagen. Las piezas siguen la guía de traslado para entrar a dos cabinas según el diseño mostrado [2.9](#).

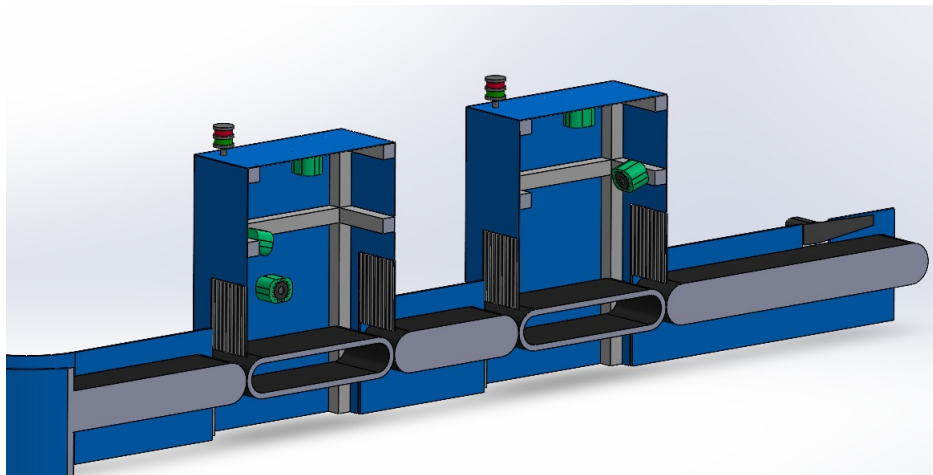


Figura 2.9: Diseño interno de cabinas de imagen.

2.7.4. Paso 4.

Toma y procesamiento de imágenes en dos cabinas, una vez la pieza esta dentro de las cabinas son tomadas las imágenes de sus caras para la realización del procesamiento de imágenes.

Este procesamiento se desarrolla siguiendo el diagrama de la figura 2.10 en donde comienza por realizar la captura de la imagen seguido de su procesamiento que se desglosa en dos tipos según el defecto a detectar el que esta mostrado ya sea el procesamiento realizado para aquellas piezas que contienen manchas y huecos o el que corresponde a las piezas que tienen rebaba, ambos fueron aplicados a las piezas que no contenían defectos.

Teniendo el procesamiento de imagen por defecto este nos entrega un vector característico por medio de los momentos de Hu para cada uno, con el cual se entrena la red para que clasifique imágenes y podamos analizar los resultados.

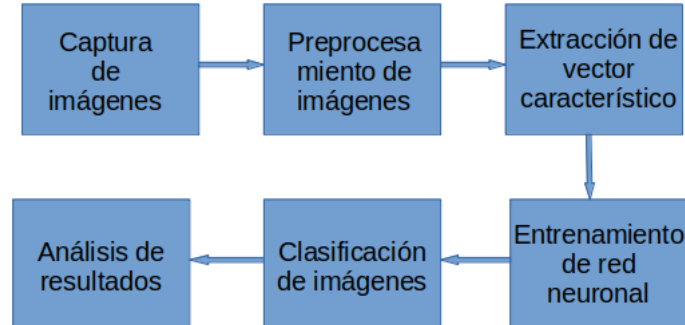


Figura 2.10: Diagrama de proceso en la cabina de imágenes.

2.7.5. Paso 5.

Clasificación de la pieza, Una vez que nuestra red esta entrenada y las imágenes fueron tomadas en la cabina podemos identificar si la pieza tiene o no defectos y tomar una decisión.

2.7.6. Paso 6.

¿Tiene defectos?, si en base a la clasificación la pieza tiene defectos esta será enviada al sistema de recuperación de material y enviará una alarma para señalar que las piezas fabricadas comienzan a tener defectos. en cuanto a las piezas no defectuosas seguirán el camino de embalaje pero incrementando el conteo de piezas en la caja y de llegar al limite incrementando el conteo en cajas de piezas.

Capítulo 3

Desarrollo

3.1. Modelo del problema

Los procesos de calidad de las piezas en la planta Wavin León son hasta el momento bastante rústicos para la producción en masa que se maneja, las conexiones pasan de manera manual y visual por el personal operativo donde ellos realizan la separación de las piezas o el ajuste para que sea aceptable, esto provoca una fatiga constante en el personal y la alta probabilidad de un desacierto al separar las piezas.

De manera general toda aquella pieza que de manera visual o física no cumple con los estándares de calidad es considerada defectuosa, podemos encontrar una variedad amplia de defectos que se pueden presentar en las piezas los cuales suelen agruparse en 6 categorías

- Apariencia: Defecto visual en el que la pieza puede tener manchas o marcas.
- Deformación: Defecto físico en el cual se presentan deformidades en el plástico que conforma la pieza.
- Pieza incompleta: Defecto físico en el cual la pieza presenta agujeros o se encuentra incompleta.
- Rebaba: Defecto físico en el cual la pieza contiene excedentes de material.

- Suciedad: Defecto visual en el que se presentan manchas de polvo o aceite, entre otros.
- Veteado: Defecto visual en el cual se presentan rayas o vetas color amarillo a marrón.

Un problema adicional se ve en el conteo de piezas esto suele ser un problema cuando el empaqueo contiene un número diferente de piezas al establecido.

3.1.1. Defectos principales

De las 6 categorías de defectos se busca detectar aquellos que tienen un mayor impacto en la producción de la pieza para lograr esto se observan las tendencias de las pérdidas por defecto generadas en los años 2020 y 2021 donde se obtiene un acumulado de ambos que se puede ver en la figura 3.1 si bien ya es visual los principales defectos que se deberían atacar, se decide realizar el 80-20 de ellos para tener una mayor visibilidad del por que debemos resolver la detección de esos defectos.

Impacto por defecto de porcentualmente.

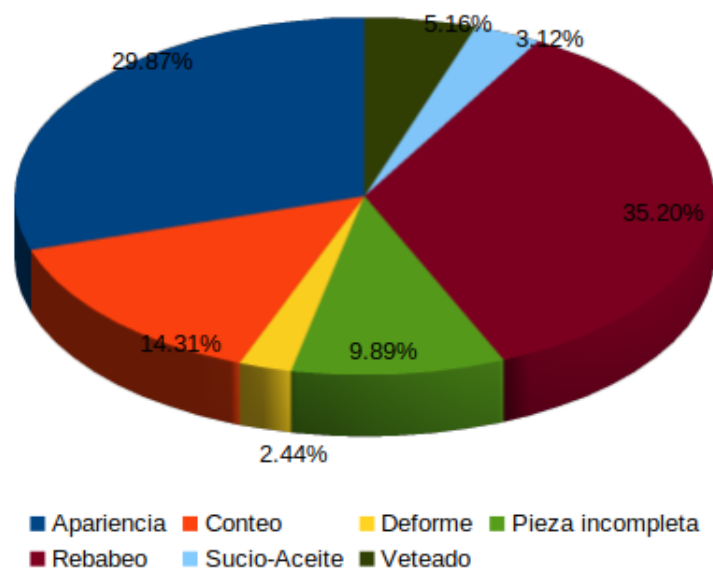


Figura 3.1: Impactos porcentuales de los defectos en el periodo 2020-2021.

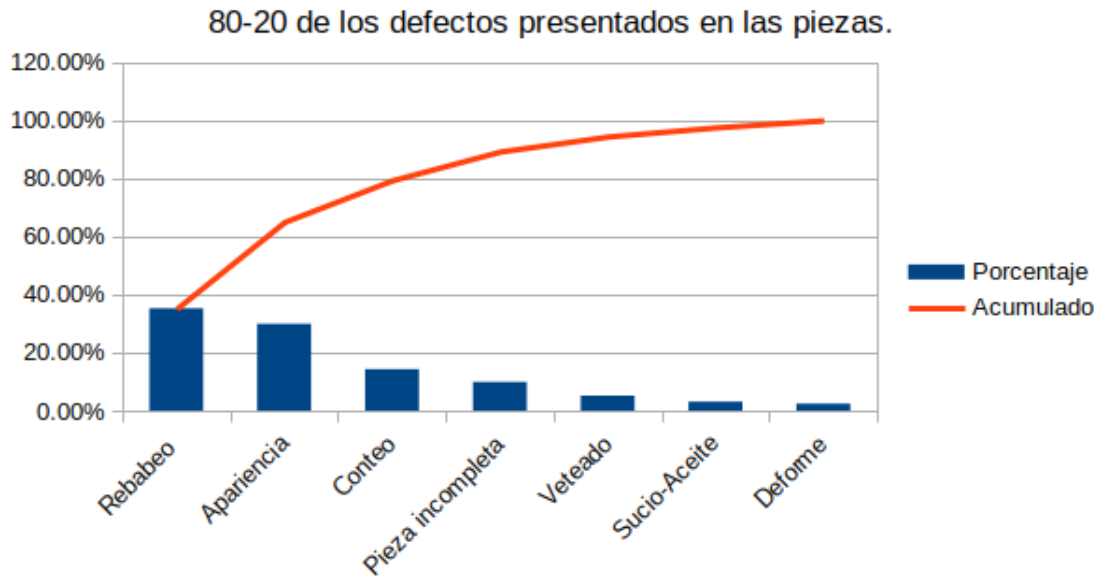


Figura 3.2: 80-20 del impacto de los defectos en el acumulado del 2020-2021.

El 80-20 mostrado en la figura [3.2](#) nos muestra que debemos atacar los problemas que conforman el 80 % de los defectos, ya que este es el área de mayor rechazo en piezas, los defectos con mayor impacto son seleccionados, rebaba con un impacto del 35.20 %, Apariencia con un impacto del 29.87 %, conteo con 14.31 % y para lograr superar el 80 % también es contemplado las piezas incompletas con un 9.89 %. De esta forma determinamos que los defectos a atacar serán 3 y un problema adicional el cual se refiere al conteo de piezas al empaquetar.

3.2. Recolección de la base de datos.

Una vez que logramos identificar los defectos que se desean detectar podemos recolectar las imágenes, es necesario contar con una estructura fija donde las imágenes contengan los mismos estándares. Se desarrolla una estructura donde las piezas puedan ser iluminadas desde la parte inferior o superior como se muestra en la imagen [3.3](#).



Figura 3.3: Estructura para la toma de imágenes.

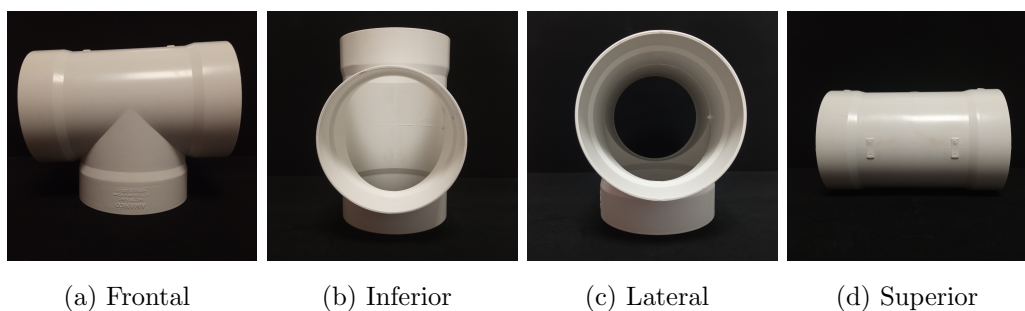


Figura 3.4: Imagen de cada una de las caras de las Tee.

Logrando una estandarización de las imágenes es posible realizar la base de datos de las piezas, conocemos los 3 defectos a capturar y se establece el número de caras de la pieza a tratar, la pieza cuenta con 6 caras donde 2 pares de ellas tienen la misma forma dando así un total de 4 caras diferentes a capturar, en las figuras [3.4](#) logramos observar las 4 caras diferentes que conforman una pieza. Tomando en cuenta que cada una cuenta con 4 tipos de pieza piezas no defectuosas, piezas con manchas o mala apariencia, con rebaba y con deformaciones o huecos, están conformadas de las siguientes caras frontal, inferior, lateral y superior donde se tendrán un total de 16 bases de datos, las bases de datos se componen de la siguiente forma.

Tabla 3.1: Dimensión de las bases de datos.

	Manchas		Rebaba		Huecos	
	Defectuosa	Correcta	Defectuosa	Correcta	Defectuosa	Correcta
Frontal	103.00	206.00	103.00	206.00	206.00	527.00
Inferior	38.00	100.00	62.00	100.00	126.00	226.00
Lateral	86.00	400.00	76.00	400.00	429.00	163.00
Superior	22.00	98.00	28.00	98.00	86.00	140.00

Se realiza la captura de imágenes para del desarrollo de la base de datos según cada especificación como se ve en los ejemplos de las figuras de la [3.5](#) a la [3.20](#).

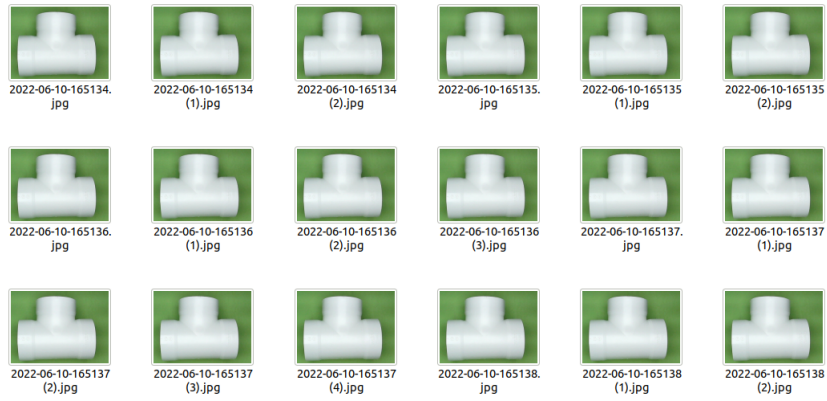


Figura 3.5: Ejemplo muestra de la base de datos sin defecto de la cara frontal.

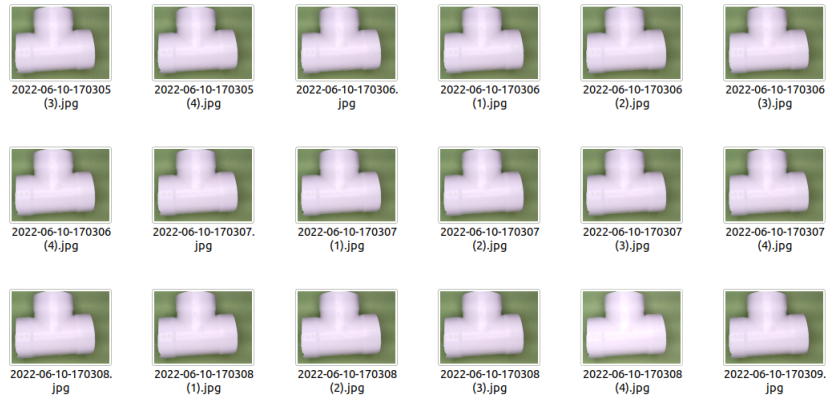


Figura 3.6: Ejemplo muestra de la base de datos rebaba presentada en la cara frontal.

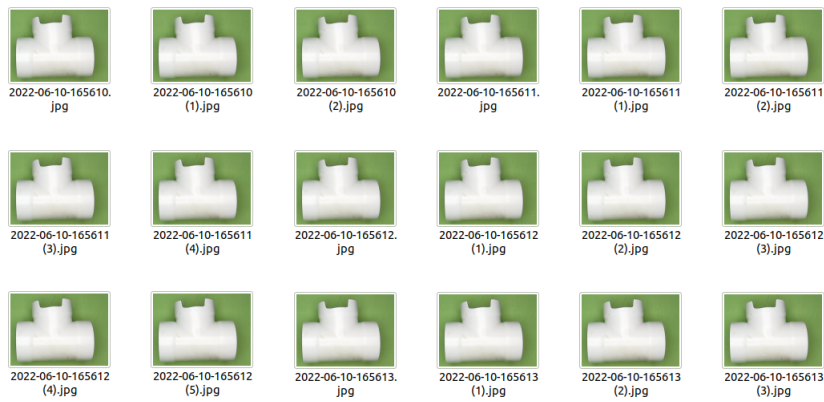


Figura 3.7: Ejemplo muestra de la base de datos huecos presentados en cara frontal.

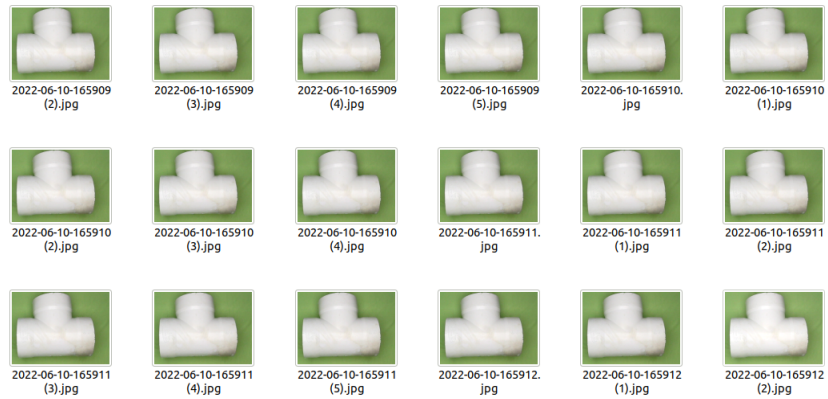


Figura 3.8: Ejemplo muestra de la base de datos manchas presentadas en cara frontal.

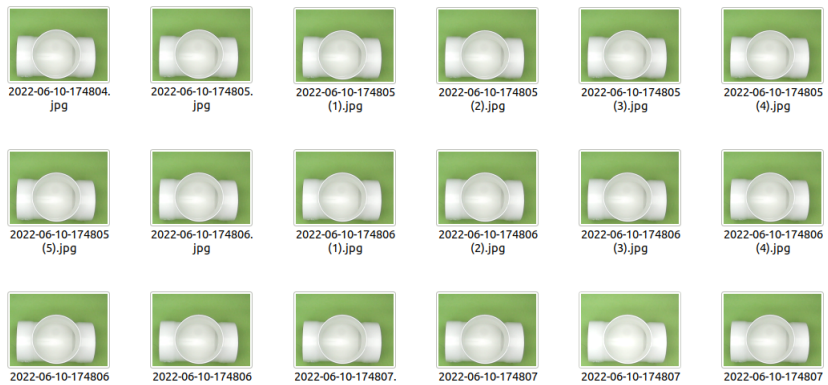


Figura 3.9: Ejemplo muestra de la base de datos sin defecto en la cara inferior .

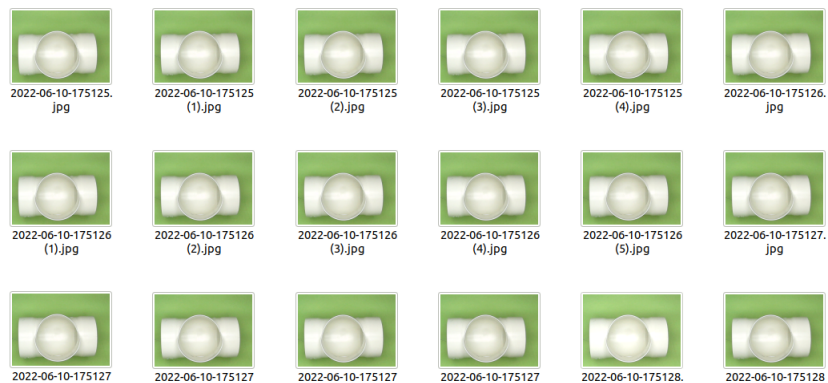


Figura 3.10: Ejemplo muestra de la base de datos con rebaba en la cara inferior.

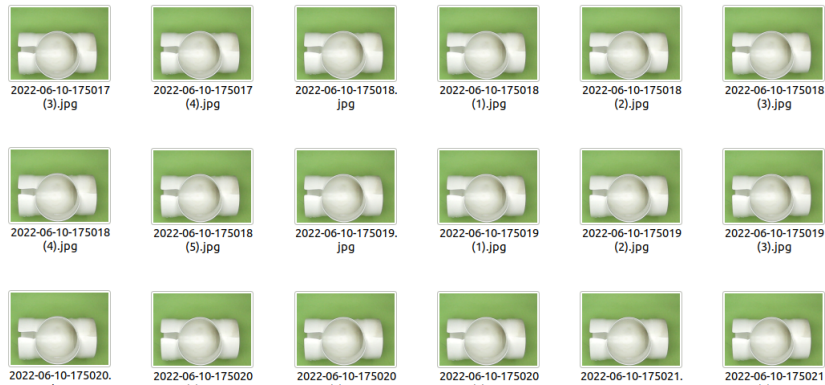


Figura 3.11: Ejemplo muestra de la base de datos con huecos en la cara inferior.

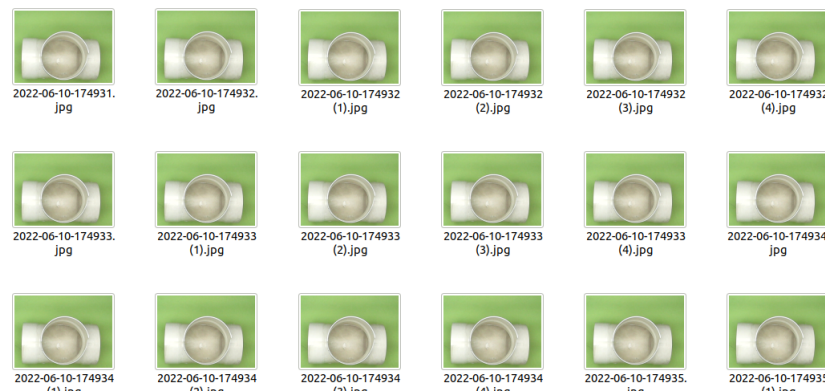


Figura 3.12: Ejemplo muestra de la base de datos con manchas en la cara inferior.

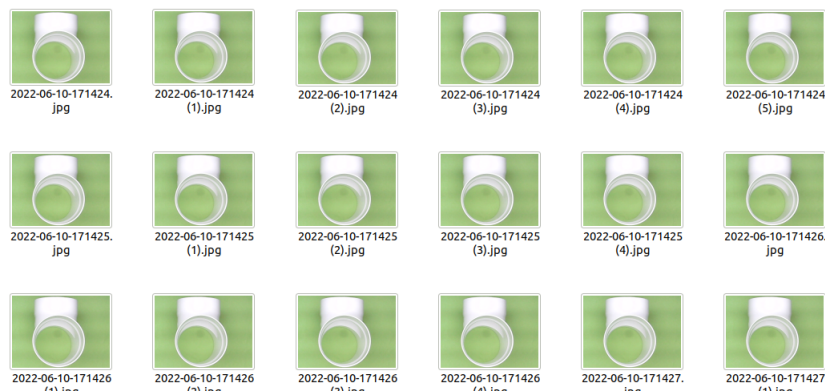


Figura 3.13: Ejemplo muestra de la base de datos sin defecto en la cara lateral.

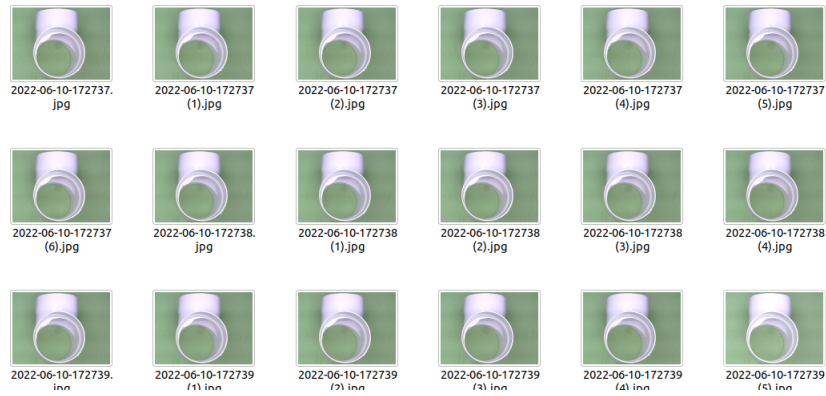


Figura 3.14: Ejemplo muestra de la base de datos con rebaba en la cara lateral.

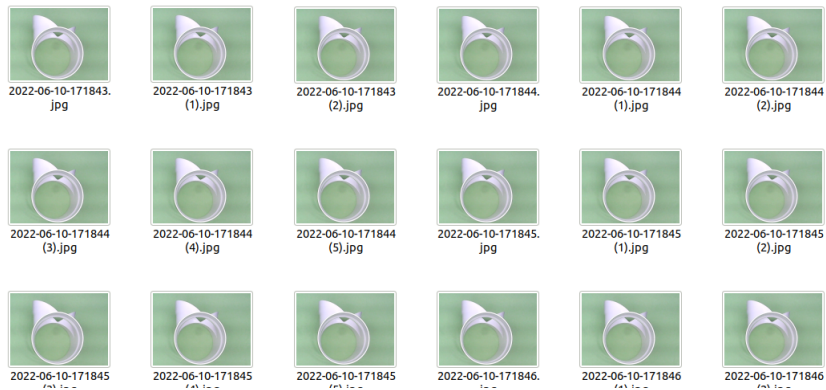


Figura 3.15: Ejemplo muestra de la base de datos con hoyos en la cara lateral.

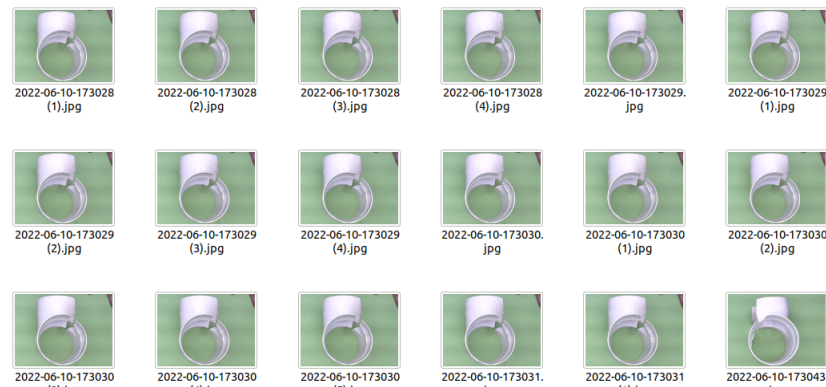


Figura 3.16: Ejemplo muestra de la base de datos con manchas en la cara lateral.

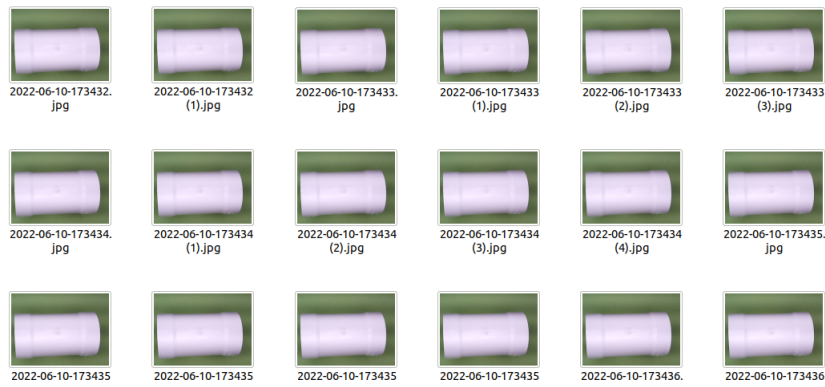


Figura 3.17: Ejemplo muestra de la base de datos sin defecto en la cara superior.

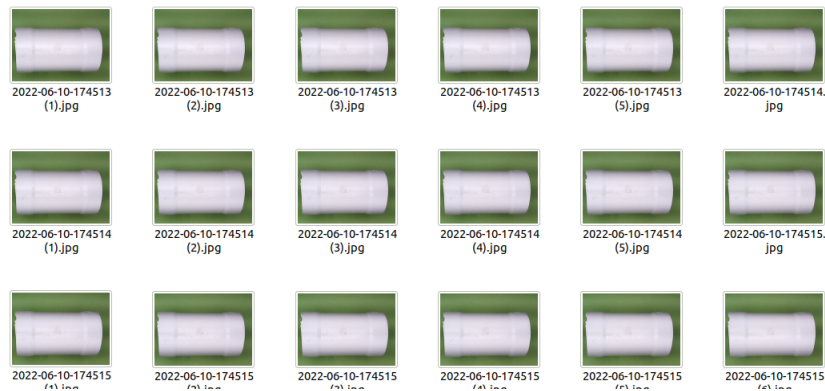


Figura 3.18: Ejemplo muestra de la base de datos con rebaba en la cara superior.

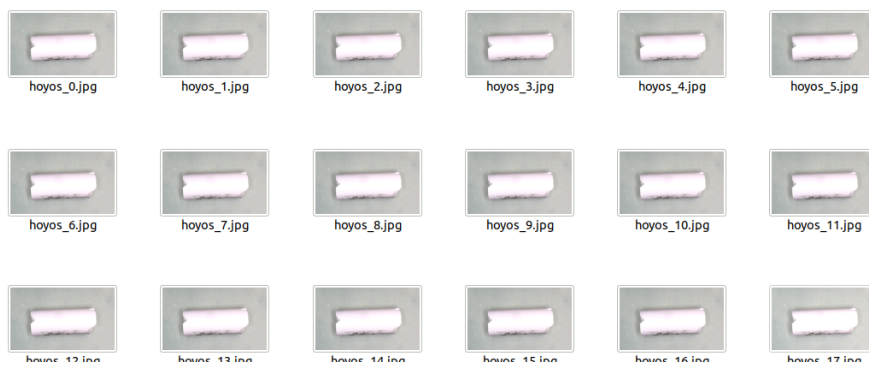


Figura 3.19: Ejemplo muestra de la base de datos con huecos en la cara superior.

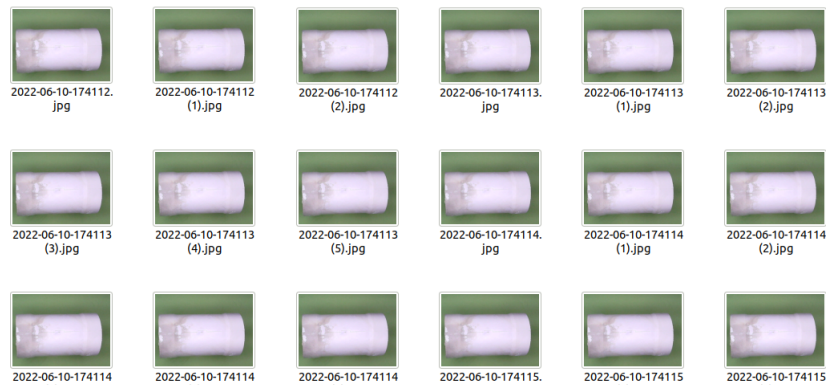


Figura 3.20: Ejemplo muestra de la base de datos con manchas en la cara superior.

3.3. Pruebas de procesamiento de imágenes.

Antes de seleccionar los métodos a aplicar a las imágenes fue necesario realizar una amplia variedad de pruebas donde se busca resaltar la pieza junto con sus características del resto del fondo. En una primera instancia se aplicó un fondo negro a la pieza y el procesamiento utilizando solo intensidades de colores a diversas imágenes con un conjunto de operaciones de imágenes para observar sus resultados algunos ejemplos de esto se muestran a continuación.

Tenemos la imagen original de la cara frontal de la Tee [3.21](#) que es tomada como ejemplo para representar los procesamientos.

Los códigos se desarrollaron en lenguaje Python, comenzamos resaltando el color según la intensidad de canales donde se obtienen las imágenes [3.22](#) para verde, azul y rojo para rojo lo logramos aplicando el siguiente código:

```
I1 = imread("IMG_20211119_172233.jpg");
iRed = I1(:, :, 1);
iGreen = I1(:, :, 2);
iBlue = I1(:, :, 3);
```

Posteriormente generamos operaciones aritméticas entre los canales obtenidos como se ven en las figuras [3.23](#) donde se hace la adición de las imágenes con la intensidad de canal azul y rojo, adición del azul y verde, adición del verde y rojo. Se desarrollo también algunas sustracciones como sustracción de los canales de intensidad rojo menos azul, sustracción de los canales de intensidad rojo menos verde y sustracción



Figura 3.21: Ejemplo de imagen original con fondo negro.

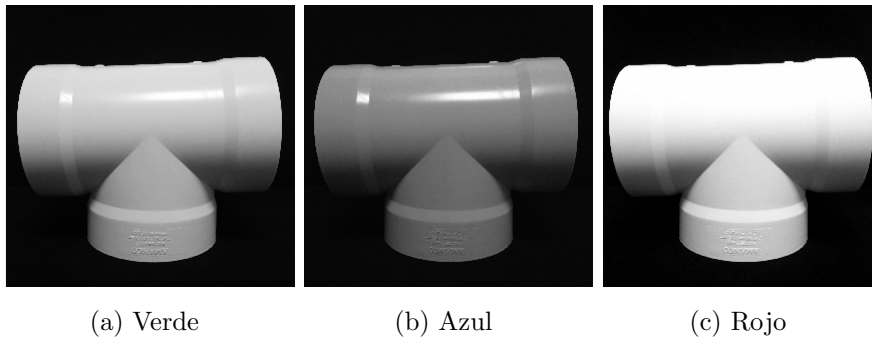


Figura 3.22: Ejemplo de la figura 3.21 con intensidad de color.

de los canales de intensidad verde menos rojo. Esto se genero mediante el código que se muestra a continuación:

```
Inew1 = iBlue+iGreen ;  
Inew2 = iBlue+iRed ;  
Inew3 = iGreen+iRed ;  
  
Inew4 = iRed-iBlue ;  
Inew5 = iRed-iGreen ;  
Inew6 = iGreen-iRed ;
```

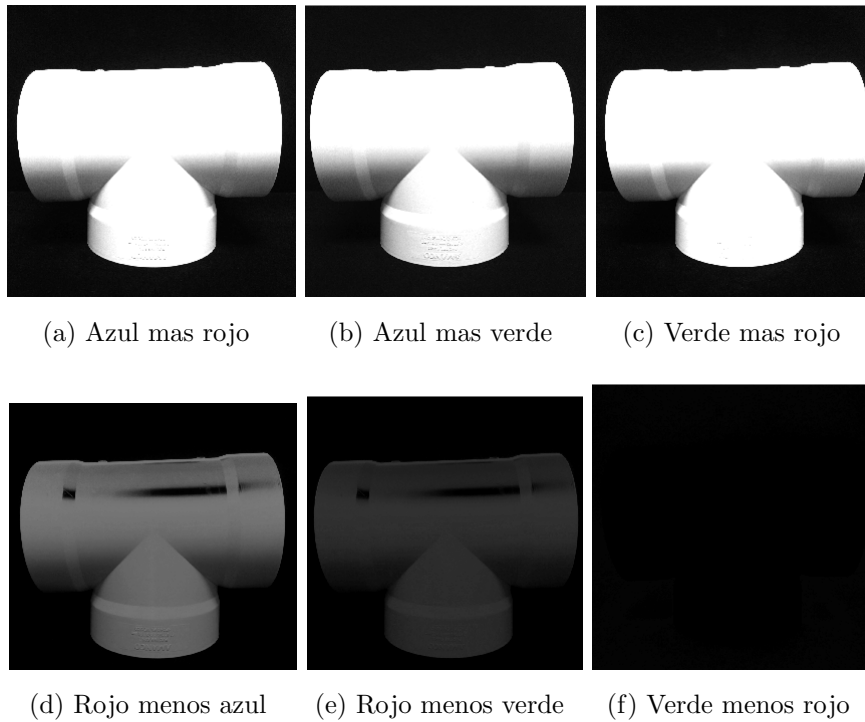



Figura 3.23: Operaciones con canal de intensidad.

Se realizaron pruebas de segmentación a través de la binarización de la imagen como se muestra en la figura [3.24](#) y la detección de bordes de la pieza como se muestra en [3.24](#) por medio del siguiente código:

```
I1gray = rgb2gray(I1);
i1BW = imbinarize(I1gray,0.1);
I1umborde = edge(i1BW, 'Canny');
```

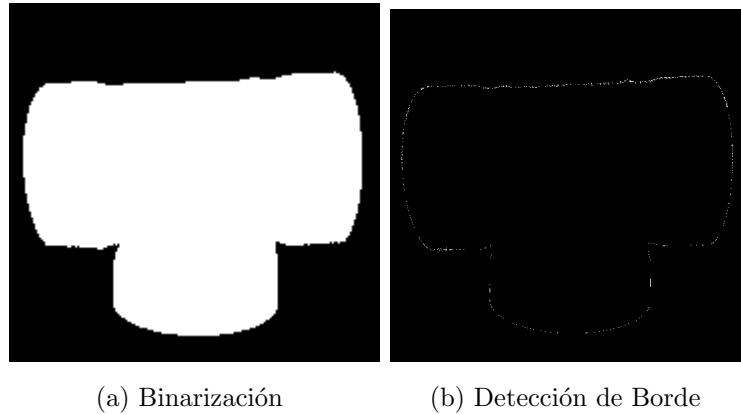


Figura 3.24: Binarización y Detección de borde de la imagen.

Tabla 3.2: Distribución de base de datos correspondiente a piezas con manchas y agujeros.

	Manchas y huecos		Rebabas	
	Defectuosas	Correcta	Defectuosas	Correcta
Frontal	206.00	206.00	206.00	527.00
Inferior	100.00	100.00	126.00	226.00
Lateral	162.00	400.00	429.00	163.00
Superior	50.00	98.00	86.00	140.00

De los primeros resultados se comenta que es posible mejorar la toma de la imagen si el fondo se reemplaza, es cambiado a un fondo verde donde tenemos la Tee sin defectos con fondo verde, Tee con rebaba fondo verde, Tee con manchas fondo verde y Tee con huecos fondo verde como se muestra en [3.25](#). Mediante el análisis de la variedad de procesamientos aplicados se llegó a la conclusión de que los huecos y manchas se pueden tratar en un solo procesamiento por lo que es posible unir las bases de datos para el tratamiento de esos defectos donde la base de datos se ve como la tabla [3.2](#). Teniendo en cuenta que los huecos y manchas fueron unificados el procesamiento de estas disminuye donde: Se aplica un procesamiento para las Manchas y huecos en cara frontal, cara inferior, cara lateral y cara superior respectivamente otros mas para aquellas con rebabas en las mismas 4 caras teniendo un

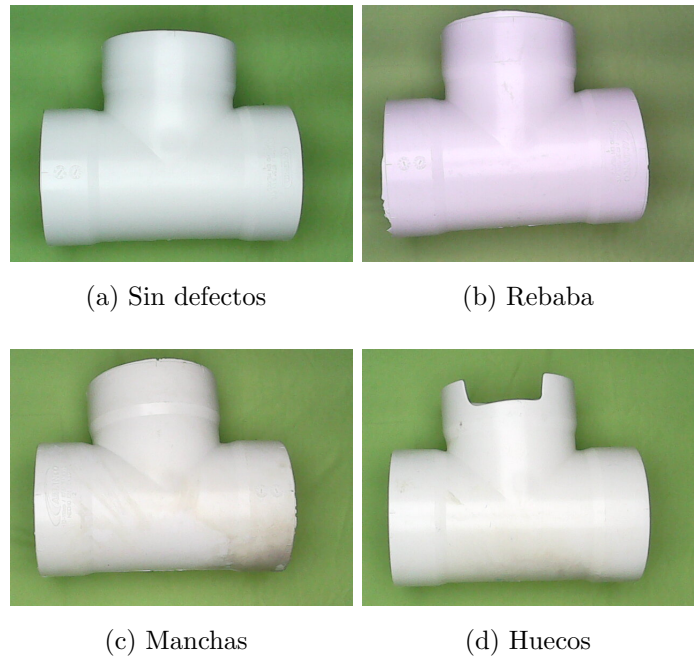


Figura 3.25: Ejemplos de la Tee en fondo verde.

total de 8 procesamientos diferentes.

- Procesamiento final para la cara frontal.

El procesamiento de la cara frontal se divide de dos maneras, la primera figura [3.26](#) que constituye a las rebabas trata la imagen sin defectos y aquellas que contienen rebabas de la misma forma, comienza separando el canal rojo de la imagen con el valor 0, donde posteriormente aplica un filtro de desenfoque gaussiano dándole un kernel de 3 por 3 y finalmente un valor de 0 para las desviaciones estándar del kernel en ambos ejes, seguido con una binarización mediante threshold con un umbral de 110 y un filtro morfológico.

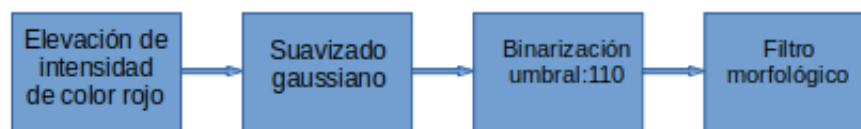


Figura 3.26: Procesamiento de la cara frontal para rebaba.

Las manchas que corresponde al segundo tratamiento figura [3.27](#) pasan por un proceso mas sencillo como se muestra en el código una vez que separamos el canal rojo, se realiza una binarización con un umbral de 160 y termina con un filtro morfológico.

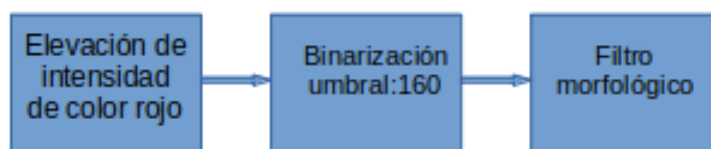


Figura 3.27: Procesamiento de la cara frontal para manchas y hueco.

```

rojo=original[:, :, 0]
Grojo= cv2.GaussianBlur(rojo ,(3 ,3) ,0)

t, Rebaba = cv2.threshold(Grojo , 110 ,255, cv2.THRESH_BINARY)
Rebaba= cv.morphologyEx(Rebabahoyos , cv.MORPHOPEN, kernel)

t, manchashoyos = cv2.threshold(Grojo , 160 ,255, cv2.THRESH_BINARY)
manchashoyos= cv.morphologyEx(manchas , cv.MORPHOPEN, kernel)
  
```

En la figura [3.28](#) podemos observar como se ve la imagen con la aplicación de ambos procesamientos cuando no se tienen defectos y en la [3.29](#) cuando los tiene.

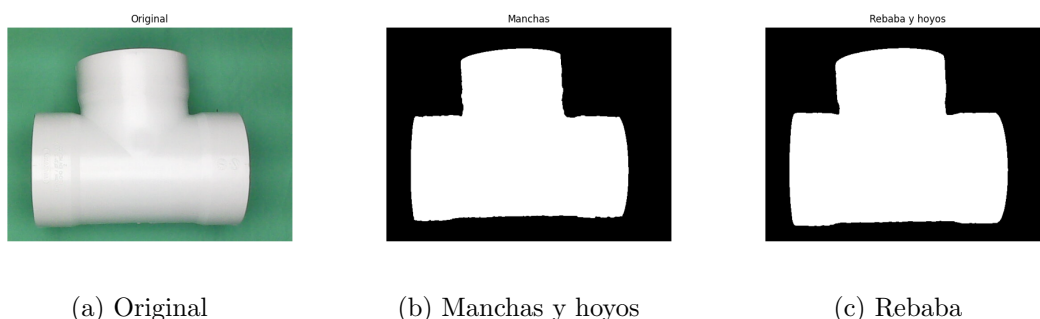


Figura 3.28: Tratamiento final de la Tee sin defectos para la cara frontal.

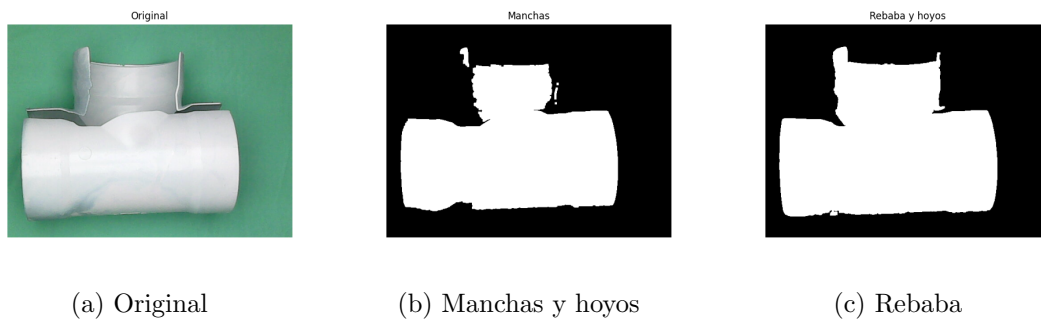


Figura 3.29: Tratamiento final de la Tee con defectos para la cara frontal.

- Procesamiento final para la cara Inferior.

Para la cara inferior de la pieza se realizan también dos procesamientos el correspondiente a la rebaba mostrado en la figura 3.30 donde comenzamos separando el canal rojo de la imagen al que se le aplica un filtro gaussiano con un kernel de 3 por 3 y un 0 para las desviaciones estándar a la imagen obtenida se le aplican dos binarizaciones por separado una con umbral de 20 y otra con umbral de 160 que posteriormente son adheridas una sobre otra a esta nueva imagen se le aplica un filtro gaussiano con los mismos parámetros que el anterior sucesivo de un filtro morfológico nuevamente una binarización con un umbral de 10 donde finalmente logramos obtener las imagen.

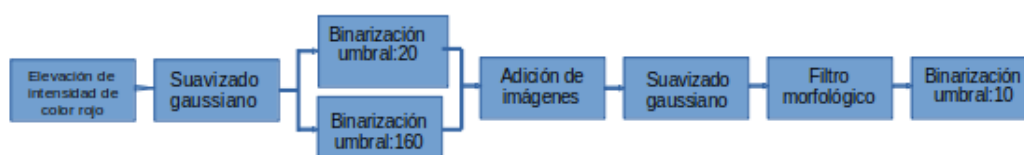


Figura 3.30: Procesamiento de la cara inferior para rebaba.

A diferencia del tratamiento de la rebaba el aplicado para las manchas y huecos es mas sencillo como se ve en la figura 3.31, ya que los primeros dos pasos de este en cuanto la separación del canal rojo y el filtro gaussiano son los mismos solo continua con una binarización de umbral de 180 y finalmente el filtro morfológico.

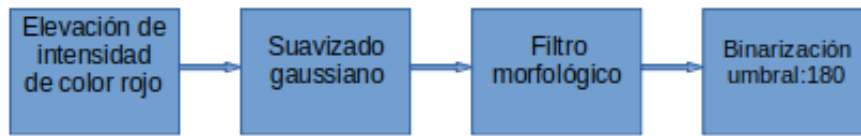


Figura 3.31: Procesamiento de la cara inferior para manchas y hueco.

```

rojo=original[:, :, 0]
Grojo= cv2.GaussianBlur(rojo ,(3,3),0)

t, Rebaba = cv2.threshold(Grojo, 20,255,cv2.THRESH_BINARY_INV)
t, Rebaba2 = cv2.threshold(Grojo, 160,255, cv2.THRESH_BINARY)
Rebaba2=Rebaba+Rebaba2
Rebaba2=cv2.GaussianBlur(Rebaba2,(3,3),0)
Rebaba2=cv.morphologyEx(Rebaba2, cv.MORPH_OPEN, kernel)
t, Rebaba2=cv2.threshold(Rebaba2,10,255,cv2.THRESH_BINARY)

t, manchas=cv2.threshold(Grojo,180,255,cv2.THRESH_BINARY)
manchas=cv.morphologyEx(manchas, cv.MORPH_OPEN, kernel)
  
```

Nuevamente encontramos ambos ejemplos en las figuras [3.32](#) y [3.33](#).

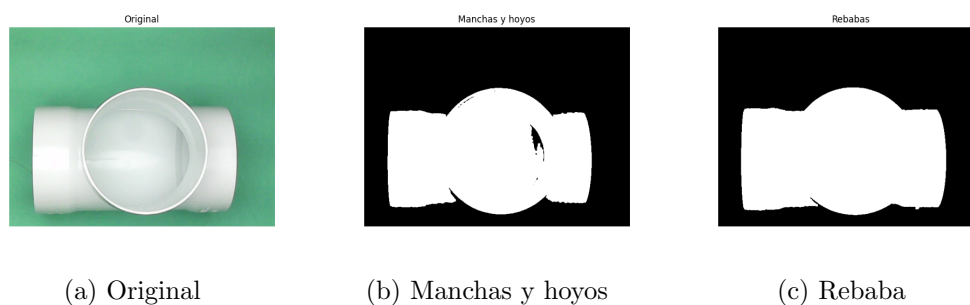


Figura 3.32: Tratamiento final de la Tee sin defectos para la cara inferior.

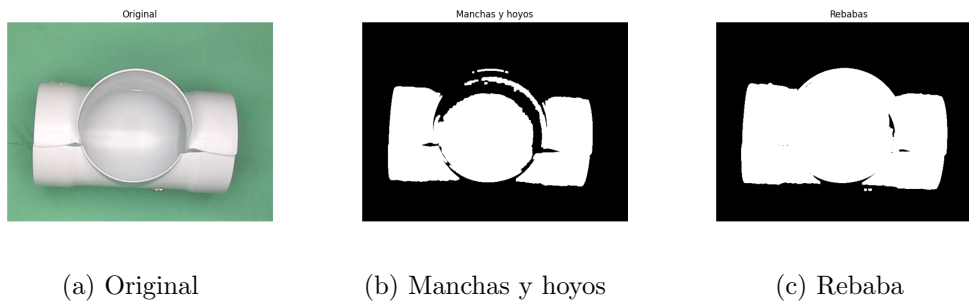


Figura 3.33: Tratamiento final de la Tee con defectos para la cara inferior.

- Procesamiento final para la cara Lateral.

Se realizan dos procesamientos para la cara lateral, figura [3.34](#) el correspondiente a la rebaba donde comenzamos separando el canal rojo de la imagen al que se le aplica dos filtros gaussiano uno tras otro con un kernel de 3 por 3 y un 0 para las desviaciones estándar, a la imagen obtenida se le aplican dos binarizaciones por separado una con umbral de 180 y otra con umbral de 230 que posteriormente son adheridas una sobre otra a esta nueva imagen se le aplica un filtro gaussiano con los mismos parámetros que el anterior y nuevamente una binarización con un umbral de 10 para llegar a la imagen final.

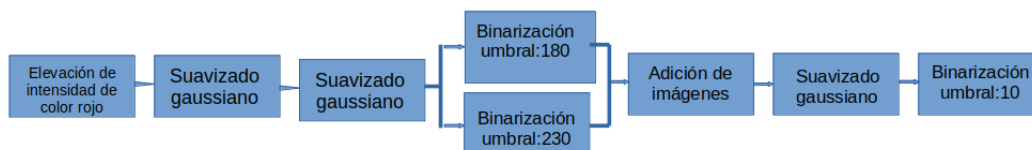


Figura 3.34: Procesamiento de la cara lateral para rebaba.

Para las machas se sigue el mismo procedimiento figura [3.35](#) sin embargo difiere en los umbrales una vez pasamos los dos filtros gaussianos la binarización de la imagen con diferentes umbrales toman el valor de 210 y 240 seguido de la adhesión de estas imágenes donde nuevamente llega al filtro gaussiano repitiendo los parámetros finalmente la misma binarización el umbral de 10.

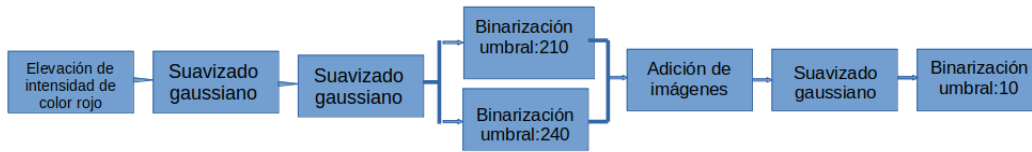


Figura 3.35: Procesamiento de la cara lateral para manchas y hueco.

```

rojo=original[:, :, 0]

Grojo= cv2.GaussianBlur(rojo,(3,3),0)
Grojo2= cv2.GaussianBlur(Grojo,(3,3),0)

t,Rebaba=cv2.threshold(Grojo,180,255,cv2.THRESH_BINARY)
t,Rebaba2=cv2.threshold(Grojo,230,255,cv2.THRESH_BINARY)
Rebaba2=Rebaba+Rebaba2
Rebaba2= cv2.GaussianBlur(Rebaba2,(3,3),0)
t, Rebaba2=cv2.threshold(Rebaba2,10,255,cv2.THRESH_BINARY)

t,manchas=cv2.threshold(Grojo2,210,255,cv2.THRESH_BINARY)
t,manchas2=cv2.threshold(Grojo2,240,255,cv2.THRESH_BINARY)
manchas2=manchas+manchas2
manchas2= cv2.GaussianBlur(manchas2,(3,3),0)
t,manchas2=cv2.threshold(manchas2,10,255,cv2.THRESH_BINARY)
  
```

Teniendo ambos ejemplos de los resultados en las figuras [3.36](#) y [3.37](#).

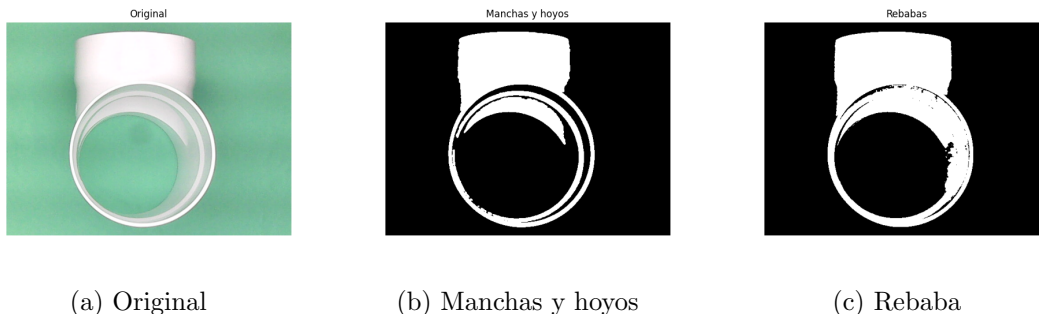


Figura 3.36: Tratamiento final de la Tee sin defectos para la cara lateral.

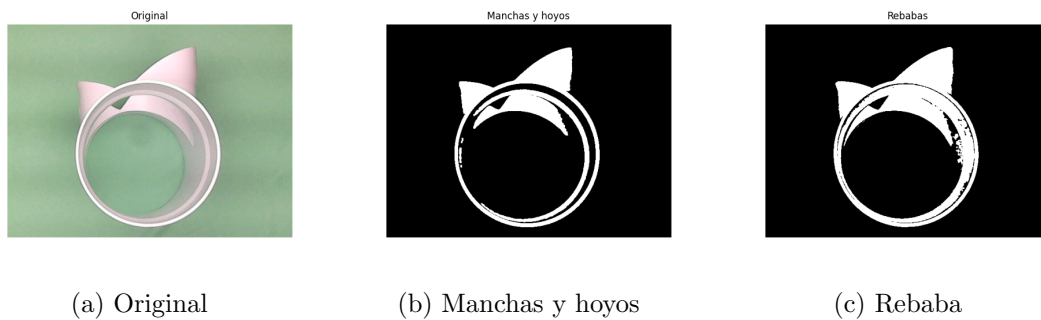


Figura 3.37: Tratamiento final de la Tee con defectos para la cara lateral.

- Procesamiento final para la cara Superior.

La Rebaba en la cara superior se trata según la figura 3.38 comenzando con una diferencia de canal donde al azul le restamos el verde, seguimos con un filtro gaussiano con los mismos parámetros continuando con una binarización umbral de 130 finalmente un filtro morfológico.



Figura 3.38: Procesamiento de la cara superior para rebaba.

Las manchas para la cara superior se trata según la figura 3.39 realiza mediante una resta de imágenes del canal rojo menos el azul , seguido de un filtro gaussiano y una umbralización con umbral de 110 finalmente un filtro morfológico.



Figura 3.39: Procesamiento de la cara superior para manchas y hueco.

```

roaz=original[:, :, 0] - original[:, :, 2] %rojo menos azul
azve=original[:, :, 2] - original[:, :, 1] % azul- verde

Gazve= cv2.GaussianBlur(azve, (3, 3), 0)
Groaz= cv2.GaussianBlur(roaz, (3, 3), 0)

t, Rebaba = cv2.threshold(azve, 130, 255, cv2.THRESH_BINARY_INV)
Rebaba= cv.morphologyEx(Rebaba, cv.MORPHOPEN, kernel)

t, manchas= cv2.threshold(Groaz, 110, 255, cv2.THRESH_BINARY_INV)
manchas= cv.morphologyEx(manchas, cv.MORPHOPEN, kernel)

```

Teniendo los resultados de ambos ejemplos en las figuras [3.40](#) y [3.41](#).

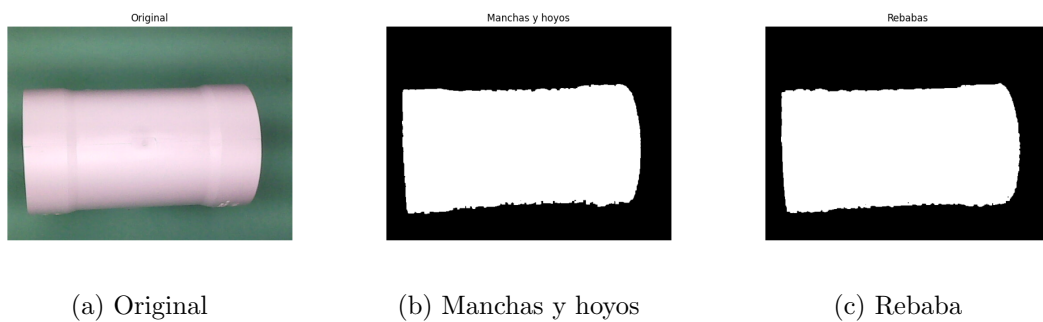


Figura 3.40: Tratamiento final de la Tee sin defectos para la cara Superior.

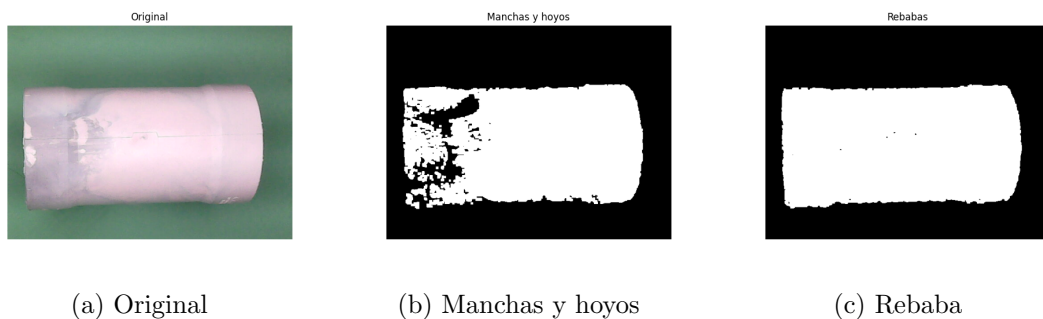


Figura 3.41: Tratamiento final de la Tee con defectos para la cara lateralsuperior.

3.4. Obtención de vectores característicos

Los vectores característicos, se obtienen a través de los momentos de Hu mediante el código mostrado a continuación, se indica a partir de cual imagen obtendrá los momentos y se asignan a un vector que es acumulado en una lista para finalmente ser exportada en un excel.

```
M=cv2.moments(cntm)
hum=cv2.HuMoments(M)
```

3.5. Entrenamiento de la red neuronal

Una vez que tenemos los vectores característicos en una base de datos podemos entrenar la red estos vectores son ingresados de manera individual siguiendo el proceso de entrenamiento que al final nos entregara los pesos calculados.

3.5.1. Entrenamiento de MLP

Se propone que MLP nos apoye en la separación de las imágenes para lograr una clasificación entre las partes que cumplen con los requisitos, estas se etiquetan como no defectuosas y las que no cumplen con los requisitos, que se etiquetan como defectuosas. La red neuronal se entrena a partir de los vectores obtenidos de las imágenes, estos vectores se etiquetan según su clase. La red neuronal es particular para cada una de las bases de datos esta se constituye como se muestra en la imagen [3.42](#), consta de 7 entradas que corresponden al vector característico de cada imagen, seguido de la capa oculta que cuenta con X neuronas totalmente conectadas que generaran un vector de pesos y finalmente la capa de salida que cuenta con 2 neuronas ,nuestros parámetros se establecen de la siguiente forma: NG=1000 o error=0.0001, experimentos=33 y 2 neuronas en la capa de salida ya que son las clases de piezas que tenemos. La red ser a entrenada con el 50% de los individuos y posteriormente probada con el otro 50%, en el anexo [B](#) podemos encontrar el código correspondiente a la red de MLP.

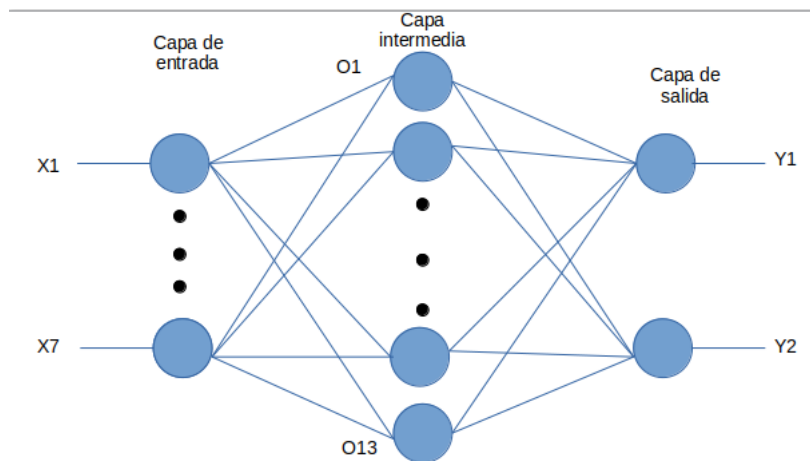


Figura 3.42: Diseño de la red neuronal.

3.5.2. Entrenamiento DE-MLP

La red neuronal DE-MLP se define con la misma estructura que MLP sin embargo N toma el valor de 20 para los vectores de pesos que posteriormente entrarán en el proceso de mutación, este proceso de mutación en el cual se busca seleccionar aquel con el mínimo error tiene por nombre best rand 2 de esta forma el mejor vector según el algoritmo genético es aquel que al final de las generaciones tiene el menor error en la clasificación de las imágenes, en el anexo [C](#) podemos encontrar el código correspondiente a la red de DE-MLP.

3.6. Clasificación de las imágenes.

Al lograr entrenar la red y obtener los valores de los pesos correspondientes, podemos ingresar nuestros vectores característicos para realizar la clasificación, de esta forma la red le asigna un valor a la salida dependiendo de la imagen de entrada según la clase obtenida y estos son analizados para sus resultados.

3.7. Prototipo del sistema

Debido al tiempo de desarrollo la implementación no fue completada, para observar el desempeño se desarrollo un prototipo mostrado en la figura 3.43, donde además de la detección de defectos pudimos agregar el contador de piezas con los componentes que ya se contaba que nos dará el apoyo para matar la perdida por conteo



Figura 3.43: Prototipo del diseño para la detección de defectos en la cara frontal.

Capítulo 4

Resultados

A lo largo del desarrollo del proyecto de tesis presentado se llevo a cabo las pruebas de clasificación propuestas, en esta sección se presentaran los resultados obtenidos según los porcentajes de clasificación de cada una y sus estadísticos obtenidos para su analisis posterior.

Los experimentos consisten en 33 subexperimentos donde se pretende la identificación de los defectos definidos en base al 80-20 de los defectos acumulados en el 2020-2021 que nos muestra la Figura 3.2, una vez definido que los defectos a considerar serán las manchas, huecos y rebabas, según las primeras pruebas de procesamiento de imágenes se definió unificar manchas y huecos en un conjunto y rebabas en otro quedando la base de datos a analizar para cada experimento como la mostrada en la tabla 3.2 donde se toman 50 % para pruebas y 50 % para entrenamiento, las imágenes son procesadas según lo establecido en la sección 3.3, obtenemos sus vectores característicos como se muestra en la sección 3.4 y así introducidos a cada red que fue entrenada según los parámetros de la sección 3.5.

4.1. Resultados para la cara frontal.

La base de datos de la cara frontal se separo para la experimentación según la tabla 4.1, cada una de las imágenes fue procesada según los dos conjuntos de defectos y lo indicado en la sección 3.3 como procesamiento final de la cara frontal, una vez

Tabla 4.1: Base de datos de la cara frontal para experimentación en las redes neuronales MLP y DE-MLP.

	Prueba		Entrenamiento	
	Defectuosas	Correctas	Defectuosas	Correctas
Manchas y hueco	103	103	103	103
Rebaba	103	264	103	263

realizado es posible obtener los vectores característicos a través de los momentos de Hu como se muestra en la sección 3.4 y así cargados a las redes MLP y DE-MLP con los parámetros mostrados en la sección 3.5 para obtener los resultados.

Las figuras 4.1 a la 4.4 comprende los resultados del porcentaje de clasificación de cada experimento para la cara frontal con MLP que son desglosados en detección de rebabas entrenamiento y prueba así como detección de manchas o huecos entrenamiento y prueba, teniendo los resultados de cada uno de ellos logramos obtener los estadísticos mostrados en la tabla 4.1. Los mismos experimentos pero con la aplicación de DE-MLP se muestran en las figuras 4.5 a la 4.8 con esta se obtienen los estadísticos de la tabla 4.2.

Resultados de clasificación MLP de los datos de entrenamiento cara frontal con manchas o huecos.

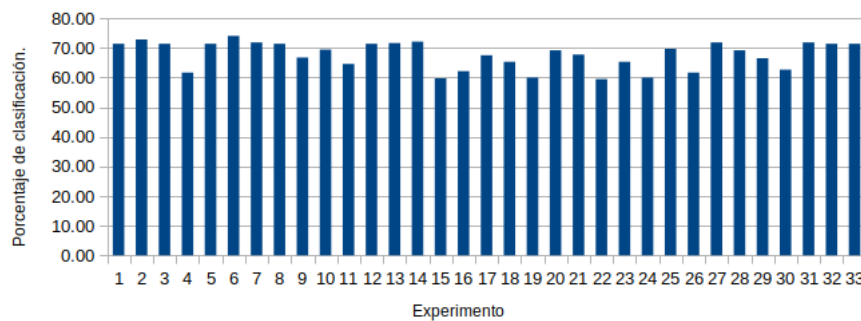


Figura 4.1: Resultados de clasificación de MLP de los datos de entrenamiento cara frontal con manchas o huecos.

Resultados de clasificación MLP de los datos de prueba cara frontal con manchas o huecos.

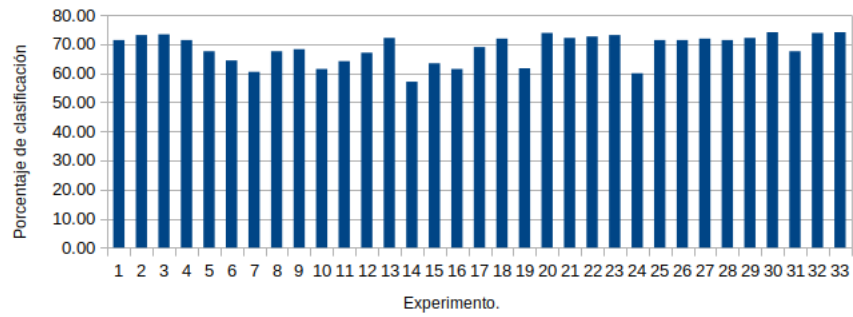


Figura 4.2: Resultados de clasificación de MLP de los datos de prueba cara frontal con manchas o huecos.

Resultados de clasificación MLP de los datos de entrenamiento cara frontal con rebaba.

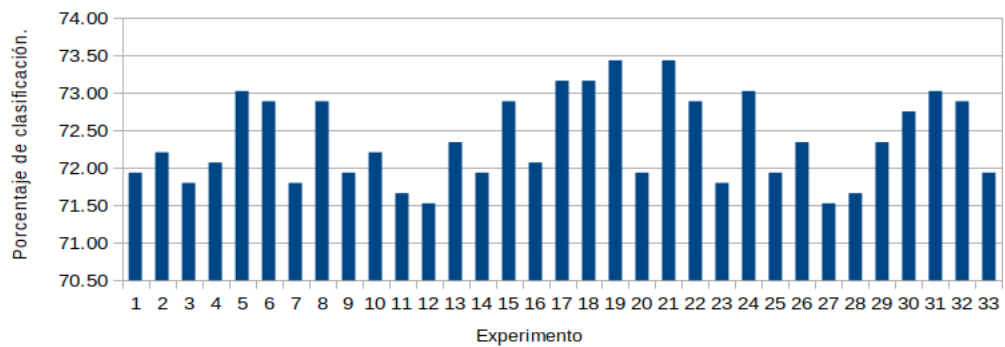


Figura 4.3: Resultados de clasificación de MLP de los datos de entrenamiento cara frontal con rebaba.

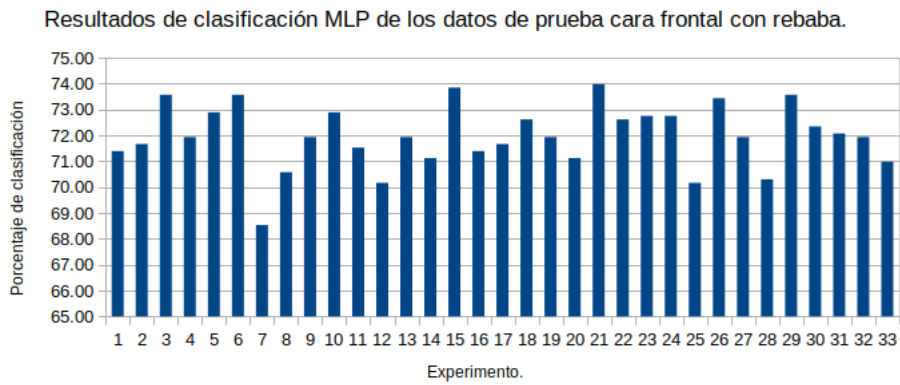


Figura 4.4: Resultados de clasificación de MLP de los datos de prueba cara frontal con rebaba.

Tabla 4.2: Estadísticos del porcentaje de clasificación de MLP en la cara frontal con manchas o huecos y rebabas.

	Manchas y huecos		Rebabas	
	Entrenamiento	Prueba	Entrenamiento	Prueba
Máximo	74.03	74.03	73.43	73.98
Mínimo	59.47	57.04	71.53	68.53
Promedio	67.69	68.62	72.38	71.93
Mediana	69.17	71.36	72.21	71.93
Moda	71.36	71.36	71.93	71.93

Resultados de clasificación MLP-DE de los datos de entrenamiento cara frontal con manchas o huecos.

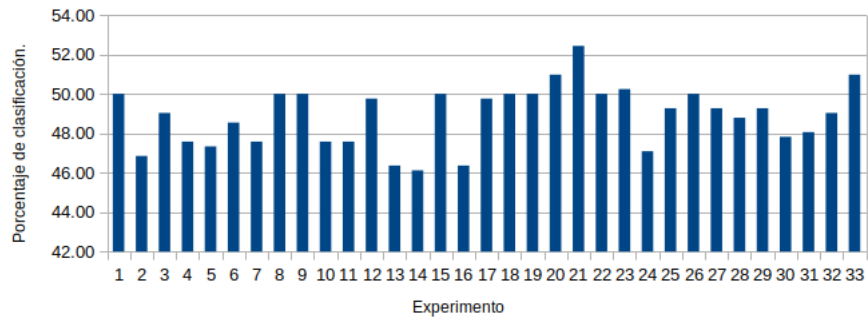


Figura 4.5: Resultados de clasificación de DE-MLP de los datos de entrenamiento cara frontal con manchas o huecos, Fuente propia.

Resultados de clasificación MLP-DE de los datos de prueba cara frontal con manchas o huecos.

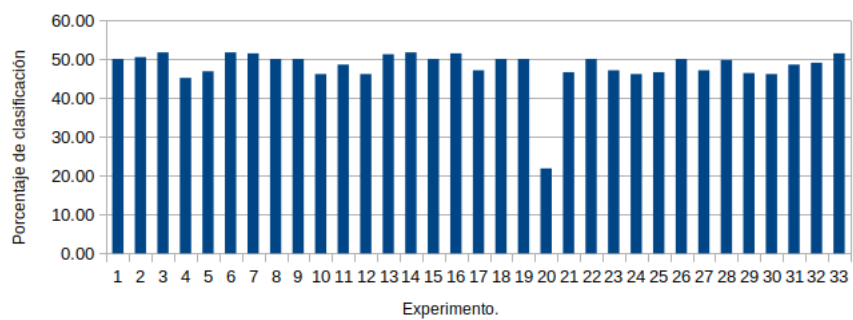


Figura 4.6: Resultados de clasificación de DE-MLP de los datos de prueba cara frontal con manchas o huecos, Fuente propia.

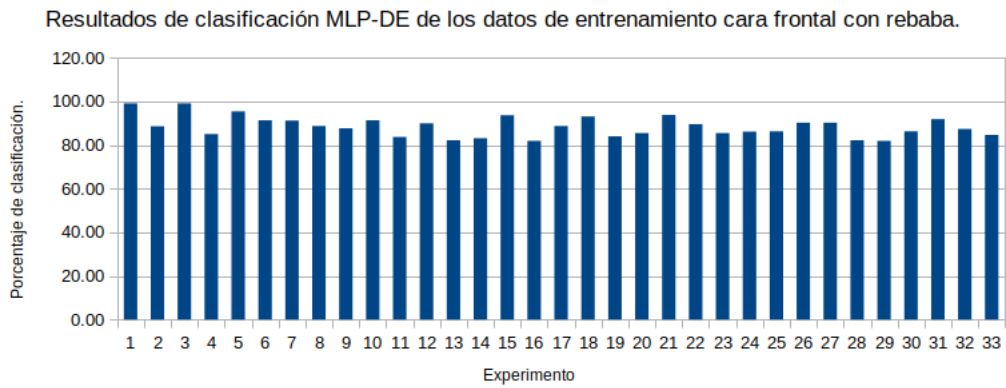


Figura 4.7: Resultados de clasificación de DE-MLP de los datos de entrenamiento cara frontal con rebaba, Fuente propia.

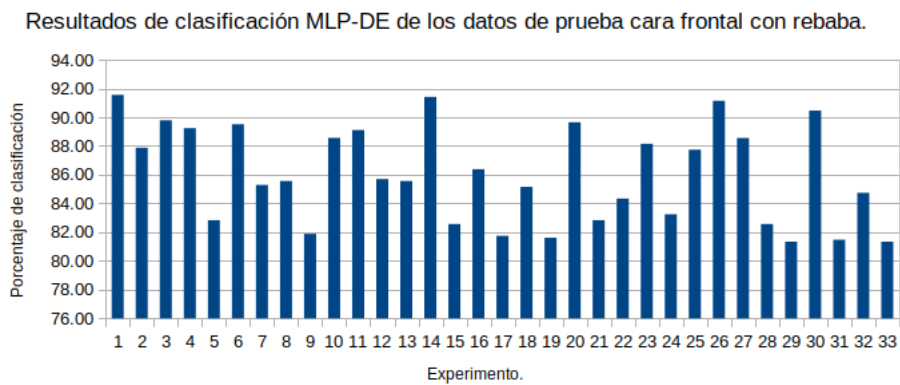


Figura 4.8: Resultados de clasificación de DE-MLP de los datos de prueba cara frontal con rebaba, Fuente propia.

Tabla 4.3: Estadísticos del porcentaje de clasificación de DE-MLP de la cara frontal con manchas o huecos y rebabas.

	Manchas o huecos		Rebabas	
	Entrenamiento	Prueba	Entrenamiento	Prueba
Máximo	52.43	51.70	99.05	91.55
Mínimo	46.12	21.84	81.88	81.34
Promedio	48.90	48.07	88.42	86.03
Mediana	49.27	49.76	88.56	85.56
Moda	50.00	50.00	99.05	82.83

4.2. Resultados para la cara inferior.

La base de datos de la cara inferior se separo para la experimentación según la tabla [4.4](#), cada una de las imágenes fue procesada según los dos conjuntos de defectos y lo indicado en la sección [3.3](#) como procesamiento final de la cara inferior, una vez realizado es posible obtener los vectores característicos a través de los momentos de Hu como se muestra en la sección [3.4](#) y así cargados a las redes MLP y DE-MLP con los parámetros mostrados en la sección [3.5](#) para obtener los resultados.

Tabla 4.4: Base de datos de la cara inferior para experimentación en las redes neuronales MLP y DE-MLP.

	Prueba		Entrenamiento	
	Defectuosas	Correctas	Defectuosas	Correctas
Manchas y hueco	50	50	50	50
Rebaba	63	113	63	113

Las figuras [4.9](#) a la [4.12](#) nos representan los resultados del porcentaje de clasificación de cada experimento para MLP en la cara inferior, que consiste en entrenamiento y prueba para la detección de defectos con rebabas y la detección de manchas o huecos, teniendo los resultados de cada uno de ellos logramos obtener los estadísticos mos-

trados en la tabla 4.3, Para las figuras 4.13 a la 4.16 sus experimentos consisten en lo mismo sin embargo es aplicado DE-MLP y la tabla de estadísticos la encontramos en la 4.4.

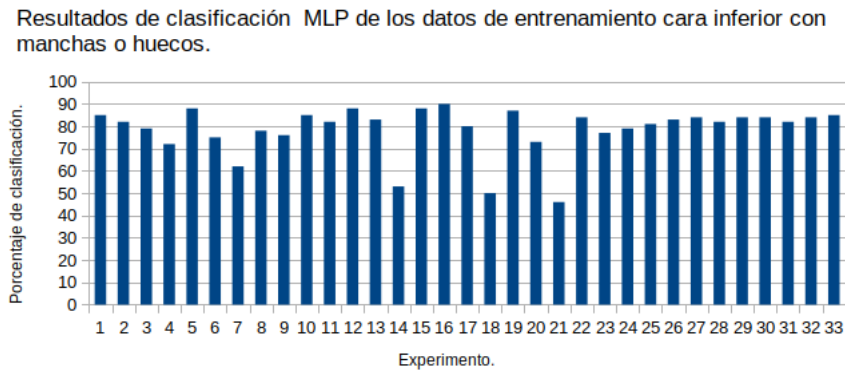


Figura 4.9: Resultados de clasificación de MLP de los datos de entrenamiento cara inferior con manchas o huecos, Fuente propia.

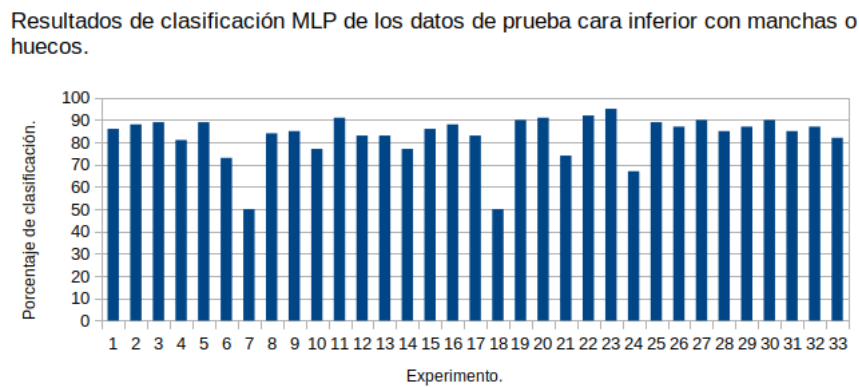


Figura 4.10: Resultados de clasificación de MLP de los datos de prueba cara inferior con manchas o huecos, Fuente propia.

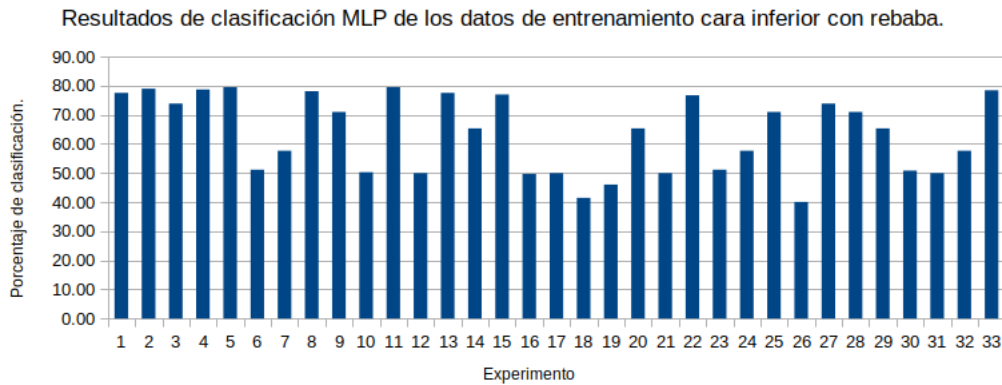


Figura 4.11: Resultados de clasificación de MLP de los datos de entrenamiento cara inferior con rebaba, Fuente propia.

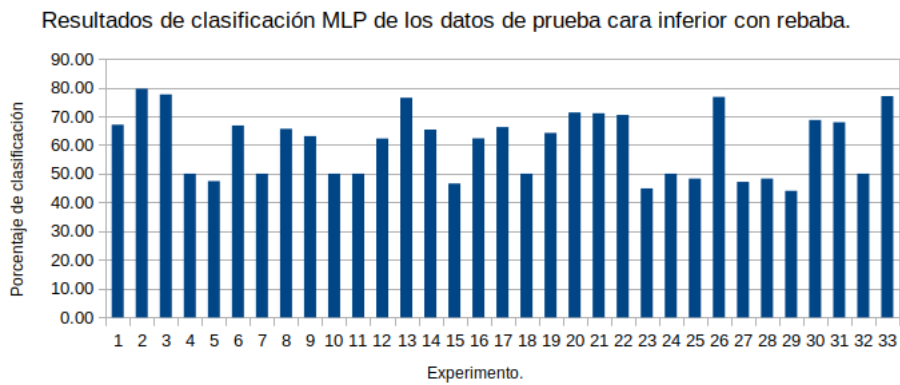


Figura 4.12: Resultados de clasificación de MLP de los datos de prueba cara inferior con rebaba, Fuente propia.



Figura 4.13: Resultados de clasificación de DE-MLP de los datos de entrenamiento cara inferior con manchas o huecos, Fuente propia.

Tabla 4.5: Estadísticos del porcentaje de clasificación de MLP en la cara inferior con manchas o huecos y rebabas.

	Manchas y huecos		Rebaba	
	Entrenamiento	Prueba	Entrenamiento	Prueba
Máximo	90	95	79.55	44.03
Mínimo	46	50	40.06	44.03
Promedio	78.52	82.85	63.41	60.49
Mediana	82	86	65.34	63.07
Moda	84	89	50	50

Resultados de clasificación MLP-DE de los datos de prueba cara inferior con manchas o huecos.

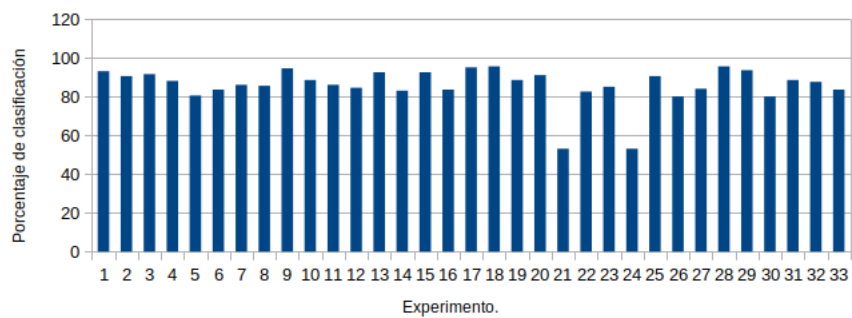


Figura 4.14: Resultados de clasificación de DE-MLP de los datos de prueba cara inferior con manchas o huecos, Fuente propia.

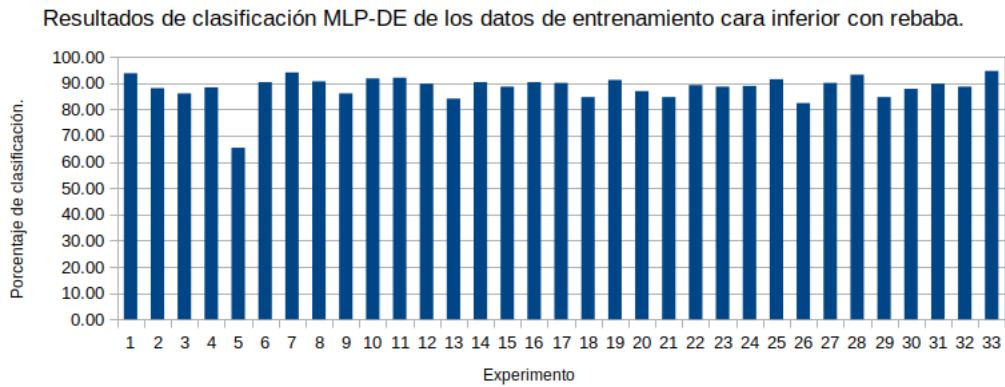


Figura 4.15: Resultados de clasificación de DE-MLP de los datos de entrenamiento cara inferior con rebaba, Fuente propia.

Tabla 4.6: Estadísticos del porcentaje de clasificación de DE-MLP de la cara inferior con manchas o huecos y rebabas.

	Manchas y huecos		Rebabas	
	Entrenamiento	Prueba	Entrenamiento	Prueba
Máximo	95.00	95.50	94.60	95.17
Mínimo	80.50	53.00	65.34	84.09
Promedio	86.39	85.76	88.38	89.34
Mediana	87.00	87.50	89.20	88.64
Moda	83.00	83.50	90.34	88.35

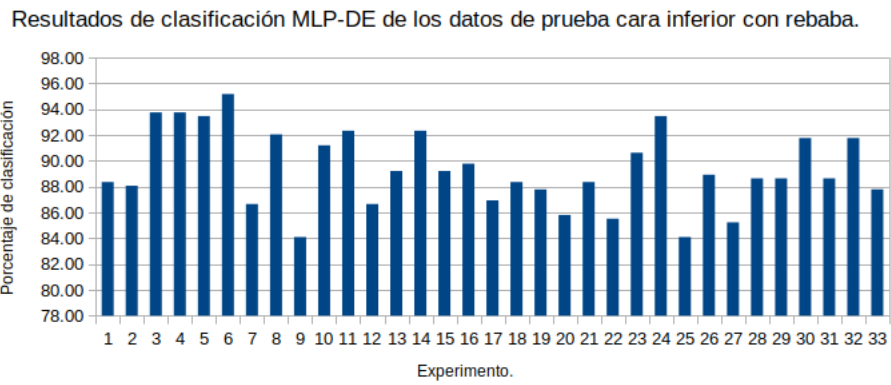


Figura 4.16: Resultados de clasificación de DE-MLP de los datos de prueba cara inferior con rebaba, Fuente propia.

4.3. Resultados para la cara superior.

La base de datos de la cara superior se separo para la experimentación según la tabla ??, cada una de las imágenes fue procesada según los dos conjuntos de defectos y lo indicado en la sección 3.3 como procesamiento final de la cara superior, una vez realizado es posible obtener los vectores característicos a través de los momentos de Hu como se muestra en la sección 3.4 y así cargados a las redes MLP y DE-MLP con los parámetros mostrados en la sección 3.5 para obtener los resultados.

Tabla 4.7: Base de datos de la cara superior para experimentación en las redes neuronales MLP y DE-MLP.

	Prueba		Entrenamiento	
	Defectuosas	Correctas	Defectuosas	Correctas
Manchas y hueco	25	49	15	49
Rebaba	43	70	43	70

Las gráficas de resultados de la cara superior se representan en las figuras 4.17 a la 4.20 para MLP y 4.21 a la 4.24 para DE-MLP nos entregan los resultados del porcentaje de clasificación de cada experimento para la cara superior, que consiste en entrenamiento y prueba para la detección de rebabas y nuevamente entrenamiento y prueba para la detección de manchas o huecos, obteniendo los estadísticos mostrados en la tabla 4.5 y 4.6 respectivamente.

Resultados de clasificación MLP de los datos de entrenamiento cara superior con manchas o huecos.

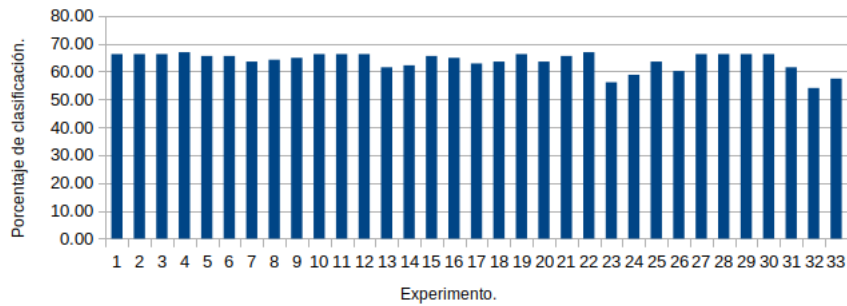


Figura 4.17: Resultados de clasificación de MLP de los datos de entrenamiento cara superior con manchas o huecos, Fuente propia.

Resultados de clasificación MLP de los datos de prueba cara superior con manchas o huecos.

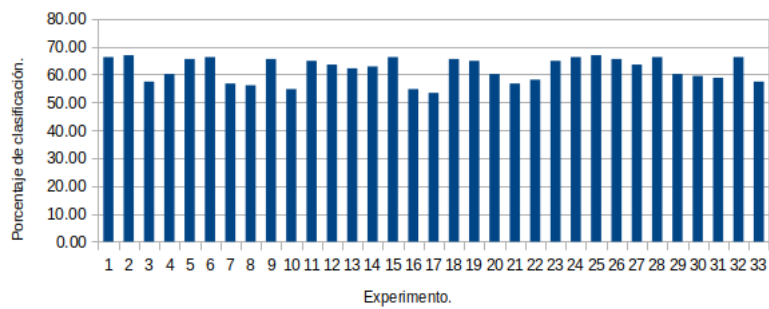


Figura 4.18: Resultados de clasificación de MLP de los datos de prueba cara superior con manchas o huecos, Fuente propia.

Resultados de clasificación MLP-DE de los datos de entrenamiento cara superior con rebaba.

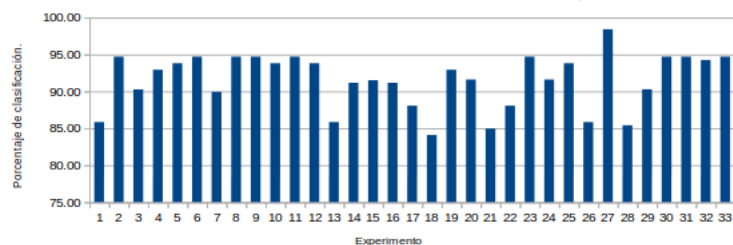


Figura 4.19: Resultados de clasificación de MLP de los datos de entrenamiento cara superior con rebaba, Fuente propia.

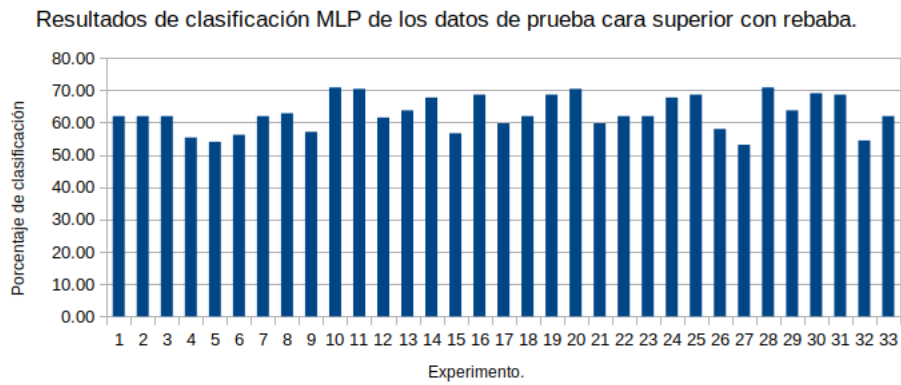


Figura 4.20: Resultados de clasificación de MLP de los datos de prueba cara superior con rebaba, Fuente propia.

Tabla 4.8: Estadísticas del porcentaje de clasificación de MLP en la cara superior con manchas o huecos y rebabas.

	Manchas y huecos		Rebabas	
	Entrenamiento	Prueba	Entrenamiento	Prueba
Máximo	66.89	66.89	70.80	70.80
Mínimo	54.05	53.38	57.52	53.10
Promedio	63.84	61.94	60.93	62.81
Mediana	65.54	63.51	61.95	61.95
Moda	66.22	66.22	61.95	61.95

Resultados de clasificación MLP-DE de los datos de entrenamiento cara superior con manchas o huecos.

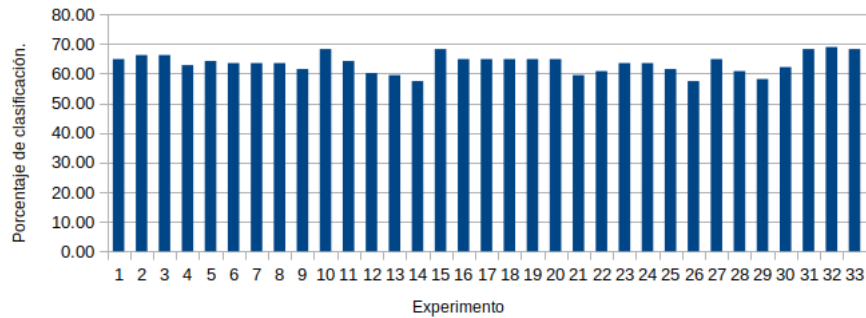


Figura 4.21: Resultados de clasificación de DE-MLP de los datos de entrenamiento cara superior con manchas o huecos, Fuente propia.

Resultados de clasificación MLP-DE de los datos de entrenamiento de la cara superior con manchas y huecos.



Figura 4.22: Resultados de clasificación de DE-MLP de los datos de prueba cara superior con manchas o huecos, Fuente propia.

Resultados de clasificación MLP-DE de los datos de entrenamiento cara superior con rebaba.

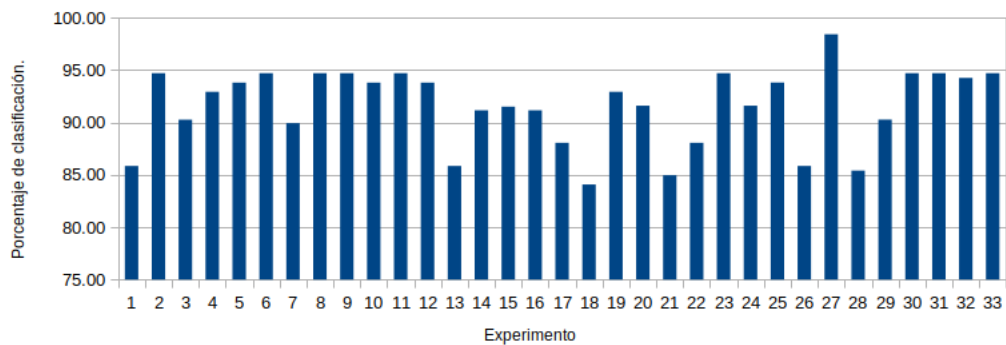


Figura 4.23: Resultados de clasificación de DE-MLP de los datos de entrenamiento cara superior con rebaba, Fuente propia.

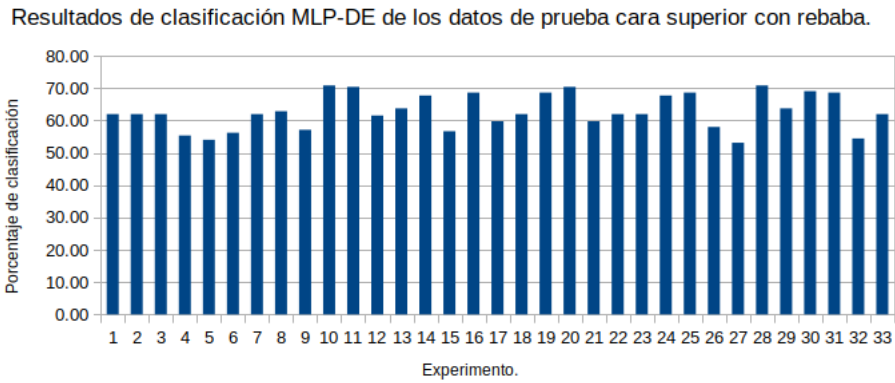


Figura 4.24: Resultados de clasificación de DE-MLP de los datos de entrenamiento cara superior con rebaba, Fuente propia.

Tabla 4.9: Estadísticos del porcentaje de clasificación de DE-MLP de la cara superior con manchas o huecos y rebabas.

	Manchas y huecos		Rebabas	
	Entrenamiento	Prueba	Entrenamiento	Prueba
Máximo	68.92	66.22	98.41	96.02
Mínimo	57.43	58.11	84.07	83.19
Promedio	63.51	63.02	91.56	90.12
Mediana	63.51	63.51	92.92	90.71
Moda	64.86	64.86	94.69	85.84

4.4. Resultados para la cara lateral.

La base de datos de la cara lateral se separo para la experimentación según la tabla [4.10](#), cada una de las imágenes fue procesada según los dos conjuntos de defectos y lo indicado en la sección [3.3](#) como procesamiento final de la cara lateral, una vez realizado es posible obtener los vectores característicos a través de los momentos de Hu como se muestra en la sección [3.4](#) y así cargados a las redes MLP y DE-MLP con los parámetros mostrados en la sección [3.5](#) para obtener los resultados.

Tabla 4.10: Base de datos de la cara lateral para experimentación en las redes neuronales MLP y DE-MLP.

	Prueba		Entrenamiento	
	Defectuosas	Correctas	Defectuosas	Correctas
Manchas y hueco	81	200	81	200
Rebaba	215	82	214	81

En las graficas [4.25](#) a la [4.28](#) tenemos los porcentajes de claificación para MLP cuando es aplicado a la cara lateral y en las [4.29](#) a la [4.32](#) para la misma cara con DE-MLP, sus experimento consisten en la detección de rebabas y la detección de manchas o huecos en prueba y entrenamiento para cada uno, logrando los estadísticos mostrados en la tabla 4.7 y 4.8 respectivamente.

Tabla 4.11: Estadísticas del porcentaje de clasificación de MLP en la cara lateral con manchas o huecos y rebabas.

	Manchas y huecos		Rebabas	
	Entrenamiento	Prueba	Entrenamiento	Prueba
Máximo	77.87	79.56	79.54	86.83
Mínimo	67.57	68.41	62.63	62.63
Promedio	71.40	72.50	70.68	72.45
Mediana	71.79	72.47	71.17	72.24
Moda	72.64	72.30	71.17	71.17

Tabla 4.12: Estadísticos del porcentaje de clasificación de DE-MLP de la cara lateral con manchas o huecos y rebabas.

	Manchas y huecos		Rebaba	
	Entrenamiento	Prueba	Entrenamiento	Prueba
Máximo	97.97	97.64	88.25	88.25
Mínimo	72.63	76.35	71.17	71.17
Promedio	83.10	85.14	72.72	78.89
Mediana	84.79	84.46	71.17	71.17
Moda	72.63	84.63	71.17	71.17

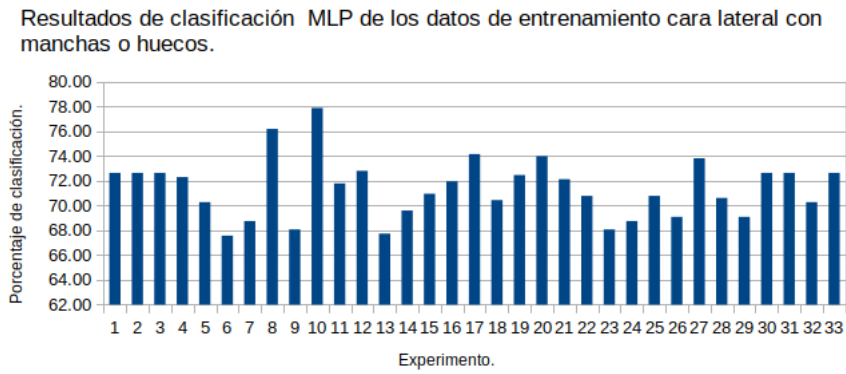


Figura 4.25: Resultados de clasificación de MLP de los datos de entrenamiento cara lateral con manchas o huecos, Fuente propia.

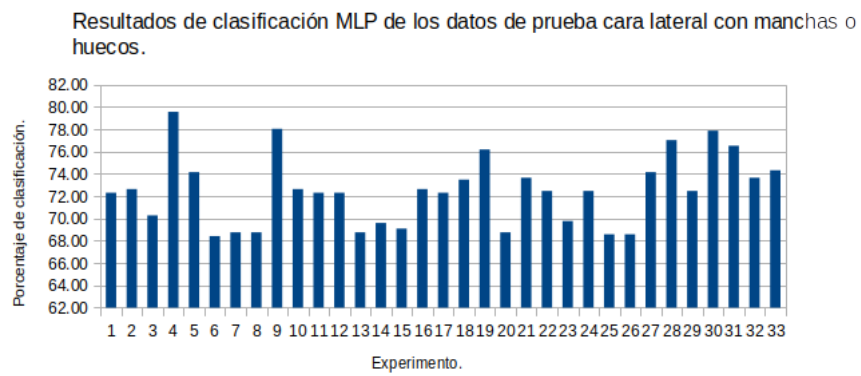


Figura 4.26: Resultados de clasificación de MLP de los datos de prueba cara lateral con manchas o huecos, Fuente propia.

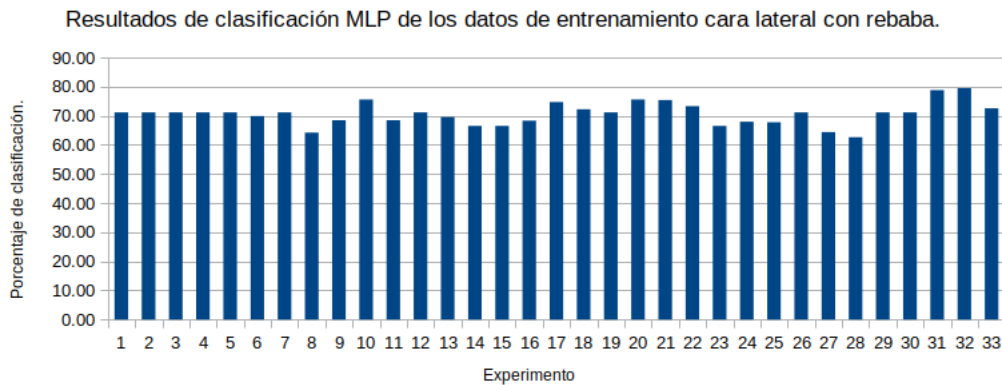


Figura 4.27: Resultados de clasificación de MLP de los datos de entrenamiento cara lateral con rebaba, Fuente propia.

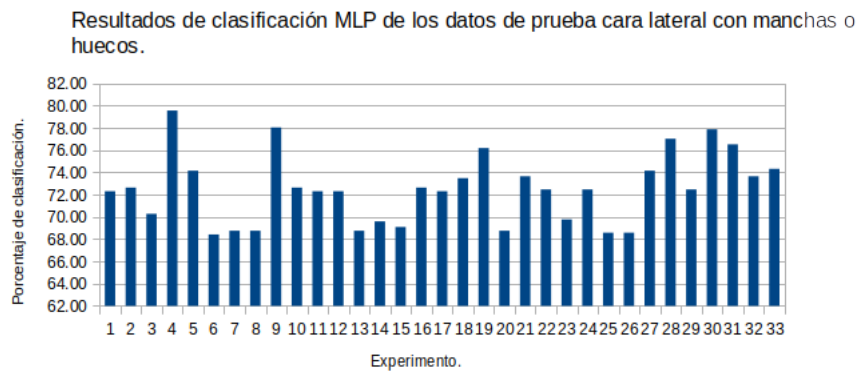


Figura 4.28: Resultados de clasificación de MLP de los datos de prueba cara lateral con rebaba, Fuente propia.

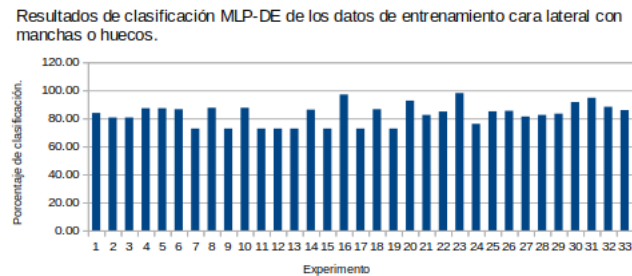


Figura 4.29: Resultados de clasificación de DE-MLP de los datos de entrenamiento cara lateral con manchas o huecos, Fuente propia.

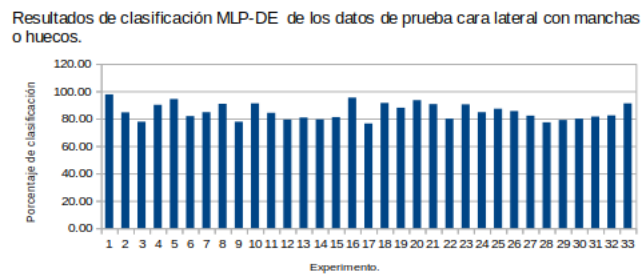


Figura 4.30: Resultados de clasificación de DE-MLP de los datos de prueba cara lateral con manchas o huecos, Fuente propia.

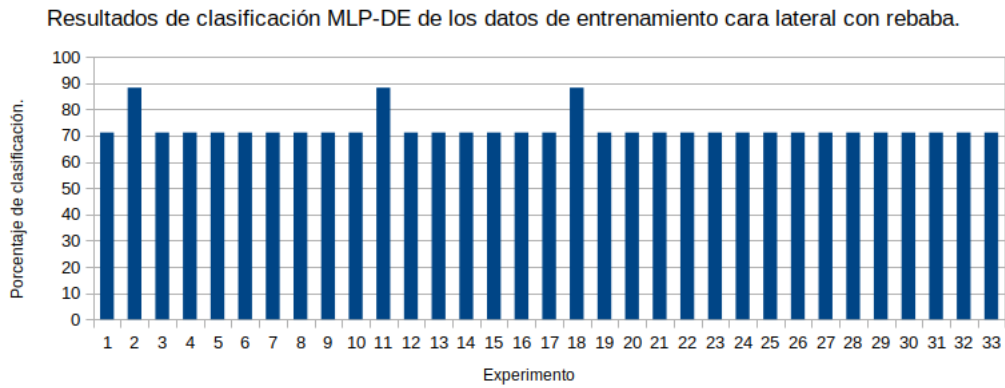


Figura 4.31: Resultados de clasificación de DE-MLP de los datos de entrenamiento cara lateral con rebaba, Fuente propia.

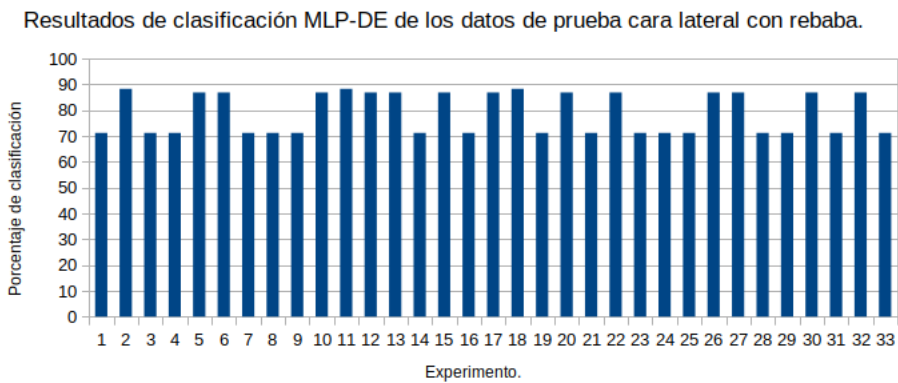


Figura 4.32: Resultados de clasificación de DE-MLP de los datos de prueba cara lateral con rebaba, Fuente propia.

Capítulo 5

Conclusiones y Trabajo a Futuro

En este trabajo se presento una aplicación industrial de las máquinas de visión para la detección de defectos en las conexiones sanitarias Tee de 110x110mm en ellas se aplico diversos métodos de procesamiento de imagen para la mejor separación de los objetos de interés, después los momentos de Hu para la obtención de los vectores característicos de estas imágenes donde finalmente fueron ingresadas a dos redes neuronales artificiales la Perceptron Multicapa (MLP) y una combinación de la misma con el algoritmo genético evolutivo Evolución Diferencial (ED), Este proceso fue aplicado para las 4 diferentes caras de la pieza y sus dos conjuntos de defectos en los que fueron separados uno que incluía las rebabas, otro más para las manchas y huecos, a continuación analizaremos los resultados estadísticos de estos experimentos, para la cara frontal en la aplicación de MLP sus estadísticos nos entregaron el porcentaje de clasificación mínimo de los experimentos donde ambos resultaron bastante bajos no superando el 80 % de porcentaje de clasificación por lo que podemos decir que 2 de cada 10 piezas son mal clasificadas, la mediana esta entre 69 a 73 % es decir al menos la mitad de los resultados esta por debajo de ese porcentaje. En cuanto a DE-MLP para el conjunto de manchas y huecos son resultados de estadísticos malos que no superan el 53 % , para la detección de rebabas vemos mejores resultados con mínimos por encima del 80 % y máximos del 99.05 % una moda encima del 80 % y mediana encima del 85 % es decir que la mayoría de los experimentos

esta por encima del 80 % de clasificación correcta. Para la cara inferior MLP para ambos conjuntos de defectos los cambios entre el porcentaje de clasificación mínimo y el máximo son altamente significativos por lo que son descartados para el análisis por lo que nos basamos en el resto de los estadísticos en el promedio observamos que no supera el 83 % las medianas y las modas para ambos conjuntos tienen menos diferencias cuando hablamos de manchas y huecos la moda oscila alrededor del 85 % y la mediana igual es decir que al menos 8 de cada 10 piezas en la mayoría de los experimentos son clasificadas correctamente, en cambio para rebaba la mediana tenemos un 65 % y la moda 50 % para ambas con una tendencia a que la mitad de las piezas por experimento sea clasificada incorrectamente, con DE-MLP se ven para ambos conjuntos mejores resultados a los anteriores los máximos porcentajes de clasificación que se encuentran alrededor del 95 % y el resto de sus estadísticos excluyendo el mínimo que se encuentran alrededor del 85 % alcanzando incluso el 90 %, estos resultados muestran mejorías para este algoritmo donde la mayoría de los experimentos muestran ventaja sobre MLP. Respecto a la cara Superior con MLP se observó que todos los estadísticos de ambos conjuntos de defectos se encuentran sin superar el 71 % de clasificación correcta, desde este punto podemos ver que en los mejores casos 7 de 10 piezas son clasificadas correctamente sin embargo es poco factible que esto se repita constantemente o que los resultados tengan una mejoría. El conjunto de manchas y huecos en DE-MLP muestra el mismo patrón estadístico de resultados que para MLP sin lograr desarrollar un mejor desempeño en cambio el conjunto de defectos por rebaba sí muestra mejores resultados con sus porcentajes de clasificación mínimos arriba del 83 % o máximos arriba del 96 % y la mayoría de sus experimentos según el promedio, la mediana y la moda logran porcentajes de clasificación arriba del 85 % incluso alcanzando el 94 %. Finalmente la cara Lateral los estadísticos de MLP tienen pocas diferencias entre sí los resultados del porcentaje de clasificación para ambos conjuntos de defectos rondan alrededor del 70-75 % con poca diferencia en los mínimos que están abajo del 70 % y en el máximo de prueba para las rebabas que es de 86.83 %, en cuanto a la moda y la mediana nos indica que en la mayoría de los experimentos al menos 7 de 10 piezas

se clasificaron de manera adecuada, cuando aplicamos DE-MLP se observan muy uniformes para el conjunto de rebaba con lo que venia mostrando el algoritmo de MLP con resultados alrededor del 70 % de porcentaje de clasificación correcto sin embargo el conjunto de defectos de manchas y huecos muestra una mejoría en el desempeño alcanzando incluso en sus máximos el 97.97 % de clasificación correcta pero un mínimo de hasta 73.63 % que cuando es analizado con la moda del mismo valor podemos observar que es un valor recurrente en los experimentos a pesar de eso vemos que sus medianas por encima del 84 % lo que nos deja ver que la mitad de los experimentos están por encima de ese valor. Si lo definimos de forma general los resultados obtenidos no podrían ser aplicados de manera industrial ya que los errores en las clasificaciones deben ser mínimos comparados con los resultados obtenidos, es importante mencionar si bien hay caras con conjuntos de defectos y métodos de los aplicados muy específicos que muestran resultados aceptables la pieza es evaluada en producción a todas sus caras y no solo a una, lo que descartaría la aplicación de estos métodos industrialmente para todas las caras de la pieza y que recomendaría la aplicación de un método de clasificación y extracción de vectores característicos diferentes trabajando en conjunto con los que obtuvieron un mejor resultado en este desarrollo.

5.1. Trabajo a futuro.

Se pretende incorporar un conjunto de trabajos a esta investigación, comenzando con el desarrollo del prototipo físico a pie de máquina en donde adicional se desean probar la obtención de los vectores característicos a través de otros métodos así como algunas otras opciones de clasificación algunos ejemplos son los métodos de reconocimiento de patrones o RNA diversas, asimismo es necesario complementar el desarrollo no solo con la clasificación de defectos en piezas si no también con la diferenciación de piezas, ya que son producidas no solo Tee's si no también codos, Yee, entre otras conexiones sanitarias que se desean clasificar con apoyo de la visión artificial .

Bibliografía

- [Adriana Birlutiu, 2017] Adriana Birlutiu, Adrian Burlacu, M. K. D. O. (2017). Defect detection in porcelain industry based on deep learning techniques. *XIX Simposio internacional sobre algoritmos simbólicos y numéricos para la informática científica (SYNASC)*,.
- [alvarez et al., 2010] alvarez, J., Hurtado, S., and trujillo, H. (2010). Algoritmos geneticos.
- [Ana, 2022] Ana (2022). *IEEE Conference Publication*.
- [Arahal et al., 2006] Arahal, M., Berenguel, M., and Rodriguez, F. (2006). *Técnicas de prediccion con aplicaciones en ingenieria*. Universidad de Sevilla.
- [Barrero, 2019] Barrero, R. C. (2019). Análisis de objetos transúcidos usando técnicas de visión por computador.
- [Barrero, 2021] Barrero, R. C. (2021). Estación para la detección de defectos superficiales en discos de freno mediante triangulación láser.
- [Cerrolaza and Annicchiarico, 1996] Cerrolaza, M. and Annicchiarico, W. (1996). *Algoritmos de optimización estructural basados en simulación genética*. Universidad central de Venezuela.
- [Cuevas et al., 2016] Cuevas, E., Osuna, J., Oliva, D., and Navarro, M. (2016). *Optimización algoritmos programados con matlab*. Alfaomega.

- [CYTED and CONICIT, 1999] CYTED and CONICIT (1999). *Aplicaciones de las redes de neuronas en supervisión, diagnosis y control de procesos*. Equinoccio.
- [de Moya Anegón et al., 1998] de Moya Anegón, F., Solana, V. H., and Bote, V. G. (1998). La aplicación de redes neuronales artificiales (rna):a la recuperación de la información.
- [Espinal et al., 2011] Espinal, A., Ornelas, M., Sotelo, M., Baltazar, R., Carpio, M., and Puga, H. (2011). Comparison of pso and de for training neural networks. *Mexican International Conference on Artificial Intelligence*.
- [Flores and Fernández, 2008] Flores, R. and Fernández, J. (2008). *Las Redes neuronales Artificiales: Fundamentos teóricos y aplicaciones prácticas*. Netbiblo.
- [Gajawada, 2012] Gajawada, S. (2012). Postdoc : The human optimization.
- [Galindo, 2012] Galindo, C. (2012). Optimización de un sistema de visión artificial para la detección de defectos en el vidrio. *Studentlitteratur AB*.
- [Grob, 1990] Grob, B. (1990). *Televisión practica y sistemas de video*. Marcombo.
- [Guangzhong Cao, 2018] Guangzhong Cao, S. R. . Y. P. (2018). Large-complex-surface defect detection by hybrid gradient threshold segmentation and image registration. *IEEE Journals Magazine — IEEE Xplore*.
- [Gómez and Gutierrez, 2011] Gómez, S. and Gutierrez, S. (2011). Desarrollo de un sistema prototipo de reconocimiento de d igitos usando momentos invariantes.
- [Hirano., 1991] Hirano., H. (1991). *Poka-Yoke, Mejorando la calidad del producto evitando los defectos*. Productivity express.
- [Iván García, 2015] Iván García, V. C. (2015). La visión artificial y los campos de aplicación. *Revista digital UPEC Tierra infinita*.
- [Lara, 2012] Lara, C. P. (2012). Clasificación de objetos rígidos a partir de imágenes digitales, empleando los momentos invariantes de hu.

- [Maestre, 2005] Maestre, J. J. E. E. . L. E. P. (2005). *Fundamentos de procesamiento de imágenes*. Universidad Autonoma de Baja California.
- [Matich, 2001] Matich, D. J. (2001). *Redes neuronales: Conceptos básicos y aplicaciones*.
- [Muangklang, 2019] Muangklang, P. S. . E. (2019). Image processing for quality control in manufacturing process. *IEEE Conference Publication*.
- [Ochoa et al., 2013] Ochoa, J., Pérez, C., Toscano, J., and etc (2013). Clasificación de objetos rígidos a partir de imágenes digitales empleando los momentos invariantes de hu.
- [Pajarez and Moreno, 2001] Pajarez, G. and Moreno, V. (2001). *Clasificación de texturas mediante redes neuronales*.
- [Patil, 2018] Patil, M. S. . R. (2018). Acoustic characterization of pvc sewer pipes for crack detection using frequency domain analysis. *IEEE Conference Publication*.
- [Quitl, 2010] Quitl, C. T. (2010). Clasificación de objetos en movimiento usando momentos de jacobi-fourier y la mtf del sistema optico digital. *Inaqe*.
- [Restrepo et al., 2021] Restrepo, D., Vilorio, J., and Robles, C. (2021). *El camino a las redes neuronales artificiales*. Unimagdalena.
- [Rivas, 2022] Rivas, H. (2022). Evolución de características de aminoácidos para la predicción de la estructura secundaria de proteínas.
- [Rivas and Mazón, 2017] Rivas, W. and Mazón, B. (2017). *Redes neuronales Artificiales aplicadas al Reconocimiento de Patrones*.
- [san Miguel., 2009] san Miguel., P. (2009). *Calidad*. Paraninfo.
- [Steemit, 2019] Steemit (2019). Breve historia de las redes neuronales artificiales.
- [Tellez, 2003] Tellez, L. P. . J. (2003). Detección y clasificación de defectos en frutas mediante el procesamiento digital de imágenes. *Revista colombiana de física*.

- [Vicente et al., 2019] Vicente, J., Gonzáles, J., Parra, F., and Beltrán., M. (2019). *Métodos de Data Science aplicados a la economía y ala administración y direccion de empresas*. Uned.
- [Vidal, 2015] Vidal, M. (2015). *El uso del Perceptrón Multicapa para la clasificación de patrones en conductas adictivas*.
- [Zambrano, 2018] Zambrano, J. E. C. (2018). Implementación de un sistema de reconocimiento de formas en fpga. *UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ*.
- [Åsa Fast-Berglund, 2020] Åsa Fast-Berglund, O. S. . (2020). Increasing operational flexibility using industry 4.0 enabling technologies in final assembly. *IEEE Conference Publication*.
- [es Murillo Arbel aez and Medina, 2019] es Murillo Arbel aez, I. D. U. M. P. A. L. P. D. U. C. E. U. E. A. E. D. C. A. and Medina, J. E. O. (2019). Implementación de una metdología para ajustar los parámetros de una carga estocástica usando el algoritmo evolución diferencial.

Apéndice A

Anexo I: Diseño propuesto.

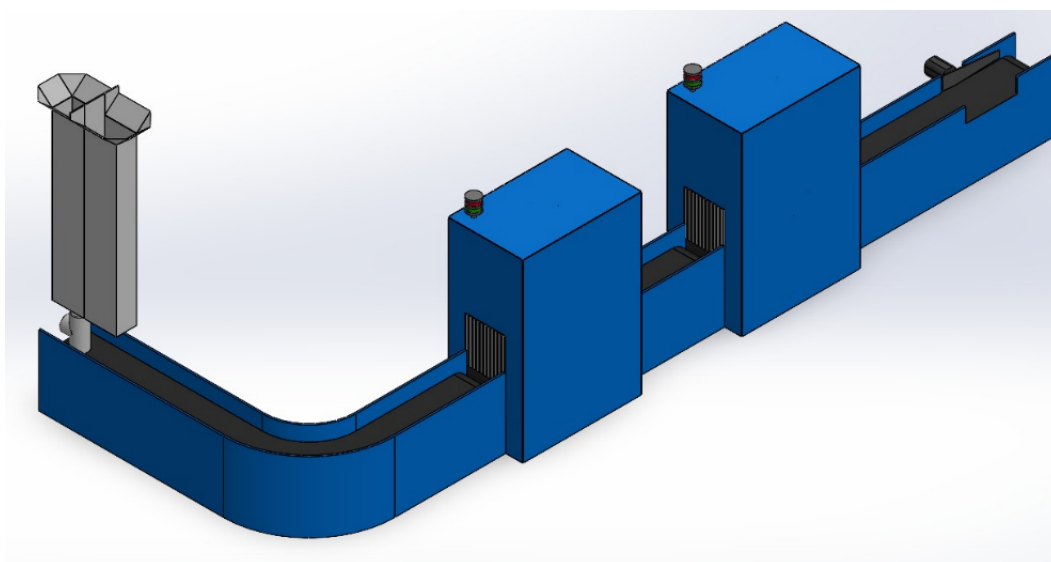


Figura A.1: Diseño propuesto para la implementación del proyecto de detección de defectos en la Tee de 110x110mm.

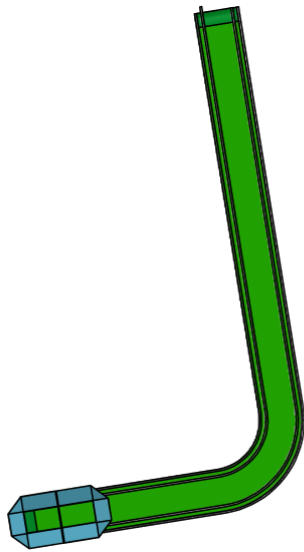


Figura A.2: Cara superior del diseño propuesto.

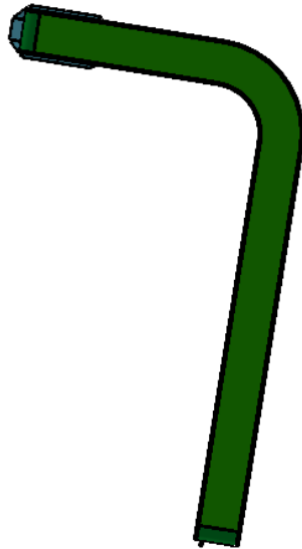


Figura A.3: Cara inferior del diseño propuesto.

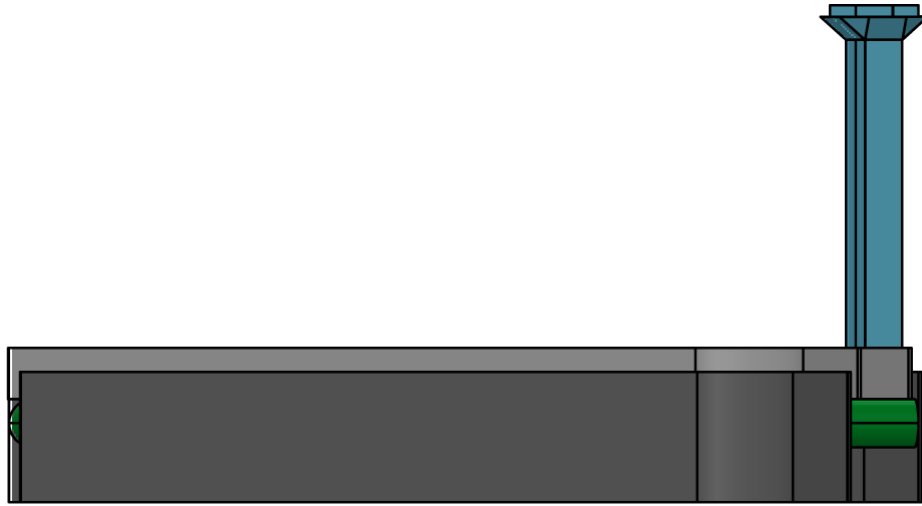


Figura A.4: Cara lateral derecha del diseño propuesto.

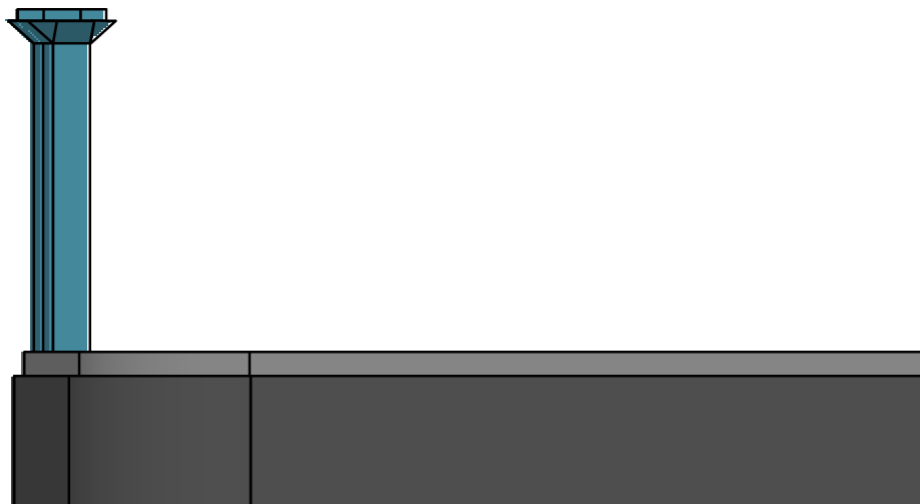


Figura A.5: Cara lateral izquierda del diseño propuesto.

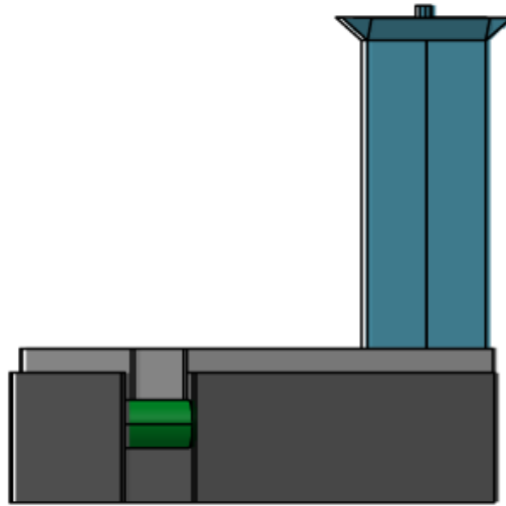


Figura A.6: Cara frontal del diseño propuesto.

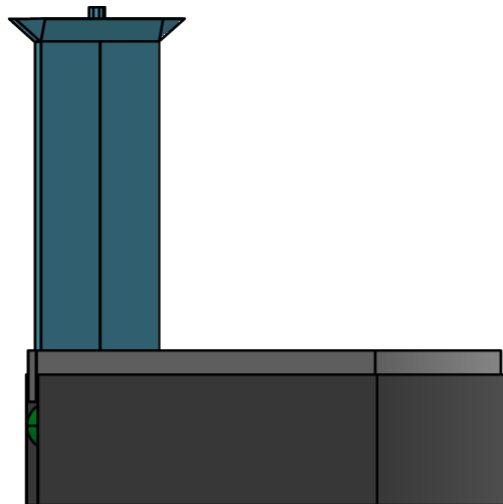


Figura A.7: Cara posterior del diseño propuesto.

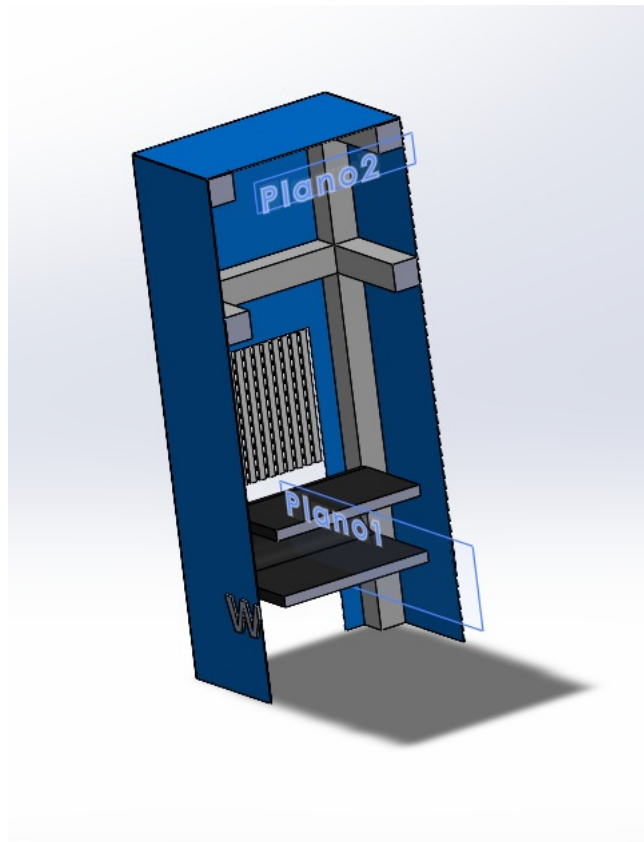


Figura A.8: Cabina para toma de imágenes del diseño propuesto.

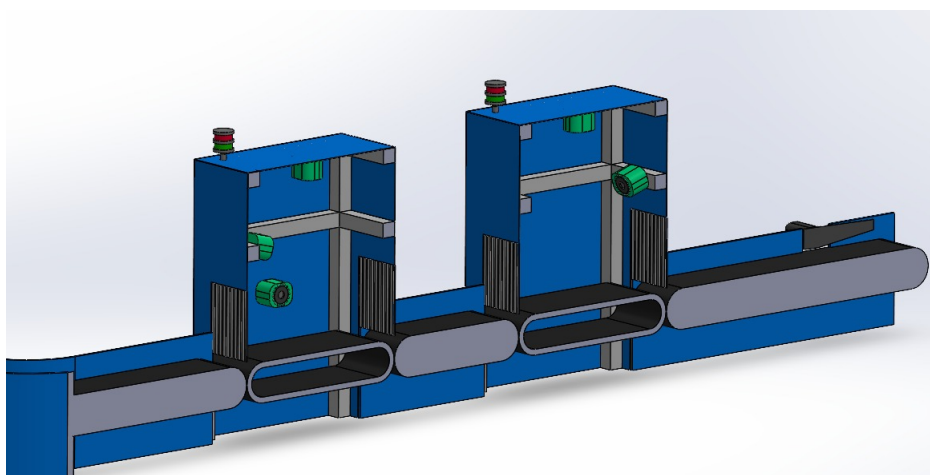


Figura A.9: Colocación de cabinas para toma de imágenes del diseño propuesto.

Apéndice B

Anexo II: Código de red neuronal MLP.

```
import math
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.model_selection import train_test_split
from math import exp
import matplotlib.pyplot as plt

NGF=1000
entrada=7
intermedia=13
salida=2
bias=1
generacion=206
expe=33

for x in range(expe):
    NG=1 #NG inicial
    errorp=0
```

```

errorg=1
vp=0
errorg_graficar=np.zeros ([NGF,1])
NG_graficar=np.zeros ([NGF,1])

v=np.random.rand(entrada+1,intermedia)
w=np.random.rand(intermedia+bias , salida)
v=(-1)+(v*2)
w=(-1)+(w*2)

iris = pd.read_csv(r"datamanchas.csv")
iris = iris.drop('Id', axis=1)
pred = pd.read_csv(r"/pred.csv")
pred = pred.drop('Id', axis=1)

X = np.array(iris.drop(['Species'], 1))
y = np.array(iris['Species'])
y_pred=np.array(pred['Species'])

X_normalizada=(X -X.min())/ (X.max()-X.min())

X_train , X_test , y_train , y_test=train_test_split(X_normalizada , y , test_size =

while ( errorg > 0.0001 and NG <= NGF ):

    errorg=0

    for g in range(generacion):

```

```

""" FEED-FORWARD """
#MATRIZ DE Zinj
zinj=np.zeros([intermedia,1])
for j in range(intermedia):
    for i in range(entrada+1):
        if (i==0):
            pbias=v[i][j]
        else:
            zinj[j][0]=(v[i][j]*X_train[g][i-1])+
            if (i==entrada):
                zinj[j][0]=zinj[j][0]+pbias

#MATRIZ DE Zj
zj=np.zeros([intermedia+bias,1])
for j in range(intermedia+1):
    if(j==0):
        zj[0][0]=1
    else:
        zj[j][0]=1/(1+exp(-zinj[j-1][0]))

#MATRIZ Yink k=3 qu son las salidas
yink=np.zeros([1,salida])
for k in range(salida):
    for j in range(intermedia+1):
        if (j==0):
            pbiasw=w[j][k]
        else:
            yink[0][k]=(w[j][k]*zj[j][0])+yink[0]
            if (j==entrada):
                yink[0][k]=+yink[0][k]+pbiasw

#MATRIZ DE YK
yk=np.zeros([1,salida])
for k in range(salida):
    yk[0][k]=1/(1+exp(-yink[0][k]))

```

```

t=np.zeros([1,salida])
if(y_train[g]=="ok"):
    t[0][0]=1
    t[0][1]=0
if(y_train[g]=="notok"):
    t[0][0]=0
    t[0][1]=1

for k in range(salida):
    errorp=math.pow((t[0][k]-yk[0][k]),2)+errorp

""" -----BACK PROPAGATION----- """

#Calculo de DELTAK
deltak=np.zeros([3,1])
for k in range(salida):
    deltak[k][0]=(t[0][k]-yk[0][k])*(yk[0][k]*(1-yk[0][k]))

#Calculo de DeltaWjk
alfa=1/(NG+1)
deltawjk=np.zeros([intermedia+1,salida])
for k in range(salida):
    for j in range(intermedia+1):
        deltawjk[j][k]=alfa*(deltak[k][0])*(zj[j][0])

#Calculo de Deltainj
deltainj=np.zeros([intermedia,1])
for j in range(intermedia):
    for k in range(salida):
        deltainj[j][0]=deltak[k][0]*w[j+1][k]+deltainj[j][0]

```

```

#Calculo Deltajprima
deltajp=np.zeros([intermedia,1])
for j in range(intermedia):
    deltajp[j][0]=deltainj[j][0]*(zj[j+1][0]*(1-zj[j+1][0]

#Calculo deltavij
deltavij=np.zeros([entrada+1,intermedia])
for i in range(entrada+1):
    for j in range(intermedia):
        if(i==0):
            deltavij[i][j]=alfa*deltajp[j][0]
        else:
            deltavij[i][j]=alfa*deltajp[j][0]*X_t

""" -----AJUSTE DE PESOS----- """
#Calculo vijnew
for i in range(entrada+1):
    for j in range(intermedia):
        v[i][j]=v[i][j]+deltavij[i][j]

#Calculo wjknew
for k in range(salida):
    for j in range(intermedia+1):
        w[j][k]=w[j][k]+deltawjk[j][k]

#Errorgeneracion
errorg=errorg+errorp
errorp=0

#Acumulado de error por generacion
if (g==generacion-1):
    errorg_graficar[NG-1][0]=errorg
    NG_graficar[NG-1][0]=NG

if (NG == NGF or (errorg <=0.0001 and g==generacion-1)):

```

```

        if(yk[0][0] > yk[0][1]):
            y_pred[g]="ok"
        if(yk[0][1] > yk[0][0]):
            y_pred[g]="notok"

    NG=NG+1

# Entrenamiento

porcentaje_train=accuracy_score(y_train , y_pred)*100
mc_train= confusion_matrix(y_train , y_pred)
print('Porcentaje de clasificaci n entrenamiento: \n', porcentaje_train,'%')

fig , ax = plt.subplots()
ax.plot(NG_graficar , errorg_graficar)
plt.xlabel("N mero de generaci n", size = 16,)
plt.ylabel("Error por generaci n", size = 16)
plt.title("Error por generacion", fontdict={'family': 'serif',
'color' : 'darkblue', 'weight': 'bold', 'size': 18})
plt.savefig(r"/home/lola/Escritorio/Maestria /Proyecto/4to semestre/imagenes/")
plt.close()

# Prueba
for g in range(generacion):

    """ FEED-FORWARD """
    #MATRIZ DE Zinj
    zinj=np.zeros([intermedia,1])
    for j in range(intermedia):
        for i in range(entrada+1):
            if (i==0):
                pbias=v[i][j]
            else:
                zinj[j][0]=(v[i][j]*X_test[g][i-1])+zinj[j][0]

```



```

        if (i==entrada):
            zinj[j][0]=zinj[j][0]+pbias

#MATRIZ DE Zj
zj=np.zeros([intermedia+bias,1])
for j in range(intermedia+1):
    if (j==0):
        zj[0][0]=1
    else:
        zj[j][0]=1/(1+exp(-zinj[j-1][0]))

#MATRIZ Yink k=3 qu son las salidas
yink=np.zeros([1,salida])
for k in range(salida):
    for j in range(intermedia+1):
        if (j==0):
            pbiasw=w[j][k]
        else:
            yink[0][k]=(w[j][k]*zj[j][0])+yink[0][k]
        if (j==entrada):
            yink[0][k]=+yink[0][k]+pbiasw

#MATRIZ DE YK
yk=np.zeros([1,salida])
for k in range(salida):
    yk[0][k]=1/(1+exp(-yink[0][k]))

t=np.zeros([1,salida])
if (y_test[g]=="ok"):
    t[0][0]=1
    t[0][1]=0
if (y_test[g]=="notok"):
    t[0][0]=0
    t[0][1]=1

```

```

#Errorpatron
for k in range(salida):
    errorp=math.pow((t [0][k]-yk [0][k]),2)+ errorp

    if(yk [0][0] > yk [0][1]):
        y_pred [g]="ok"
    if(yk [0][1] > yk [0][0]):
        y_pred [g]="notok"

# Prueba -----
porcentaje_test=accuracy_score(y_test , y_pred)*100
mc_test= confusion_matrix(y_test , y_pred)
print('\n \n Porcentaje de clasificaci n Prueba: \n', porcentaje_test,'%')

# Prueba -----

```

Apéndice C

Anexo III: Código de red neuronal MLP-DE.

```
from cProfile import label
from curses.ascii import CR
from pprint import pp
import numpy as np
import random
import matplotlib.pyplot as plt
import numpy
import statistics
import math
import pandas as pd
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.model_selection import train_test_split
from math import exp

N=20
NGF=10
NGI=0
CR=0.8
f=0.9
```

```

expf=11
pi=3.141592653589793
graf=np.zeros([1,expf])
entrada=7
intermedia=13
salida=2
bias=1
errorp=0
errorg=1
generacion=206
dim=((entrada+1)*(intermedia))+((intermedia+1)*(salida))
vectormejor=np.zeros([expf,dim])
errorminimoxg=np.zeros([NGF,1])
evaluacion=np.zeros([N,1])
NGG=np.zeros([NGF,1])

        iris = pd.read_csv(r"datamanchas.csv")
        iris = iris.drop('Id',axis=1) #borrar la columna ID
        pred = pd.read_csv(r"pred.csv")
        pred = pred.drop('Id',axis=1) #borrar la columna ID

X = np.array(iris.drop(['Species'], 1))
y = np.array(iris['Species'])
y_pred=np.array(pred['Species'])

X_normalizada=(X -X.min())/ (X.max()-X.min())

X_train , X_test , y_train , y_test = train_test_split(X_normalizada , y , test_size=0.5,

v=np.zeros([entrada+1,intermedia])
w=np.zeros([intermedia+bias , salida])

```

```

def ppavw(pp,n):
    registro=0
    for i in range (entrada+1):
        for j in range(intermedia):
            v[i][j]=pp[n][registro] # pp a v
            registro=registro+1

    for j in range(intermedia+1):
        for k in range (salida):
            w[j][k]=pp[n][registro] # pp a w
            registro=registro+1
    return v,w

###-----Feed-Forward-----#####

def FF(X_train , y_train , pp,n):
    #ERROR POR generacon sacarlo del ciclo
    errorg=0
    errorp=0
    generacion=157 #generacion=75 patrones
    v, w=ppavw(pp,n)
    for g in range(generacion):

        #MATRIZ DE Zinj
        zinj=np.zeros([intermedia ,1])
        for j in range(intermedia):
            for i in range(entrada+1):
                if (i==0):
                    pbias=v[i][j]
                else:
                    zinj[j][0]=(v[i][j]*X_train[g][i-1])+zinj[j][0]
                if (i==entrada):
                    zinj[j][0]=zinj[j][0]+pbias

        #MATRIZ DE Zj

```

```

zj=np.zeros ([intermedia+bias ,1])
for j in range(intermedia+1):
    if (j==0):
        zj [0][0]=1
    else :
        #zj [j][0]=1/(1+exp(-zinj [j -1][0]))
        zj [j][0]=max(0.0, -zinj [j -1][0])
#MATRIZ Yink k=3 qu son las salidas
yink=np.zeros ([1 , salida ])
for k in range(salida ):
    for j in range(intermedia+1):
        if (j==0):
            pbiasw=w[j ][k]
        else :
            yink [0][k]=(w[j ][k]*zj [j][0])+yink [0][k]
        if (j==entrada ):
            yink [0][k]=+yink [0][k]+pbiasw

#MATRIZ DE YK
yk=np.zeros ([1 , salida ])
for k in range(salida ):
    yk [0][k]=max(0.0, -yink [0][k])

t=np.zeros ([1 , salida ])
if (y_train [g]=="ok "):
    t [0][0]=1
    t [0][1]=0
if (y_train [g]=="notok "):
    t [0][0]=0
    t [0][1]=1

for k in range(salida ):
    errorp=math.pow((t [0][k]-yk [0][k]),2)+errorp

errorg=errorg+errorp

```

```

errorp=0
# Acumulado de error por generacion
if (g==generacion-1):
    errorg_graficar=errorg
    errorg=0
    NG_graficar=n

if(yk[0][0] > yk[0][1]):
    y_pred[g]='ok'
if(yk[0][1] > yk[0][0]):
    y_pred[g]='notok'

return y_train , y_pred , NG_graficar , errorg_graficar

def kfold(X_train , y_train):
    for experimento in range (expf):

        pp=np.zeros ([N,dim])
        pp=np.random.rand(N,dim)
        pp=(-1)+(pp*2)
        sumxinew=np.zeros ([N,1])
        for NGI in range (NGF):
            xinew=np.zeros ([N,dim])

            errorg_graficar=np.zeros ([N,1])
            NG_graficar=np.zeros ([N,1])
            for inicia in range(N):
                y_train , y_pred1 , NG_graficar [inicia] , evaluacion [inicia]=FF(X_train , y

            errorgexperimento=numpy.argmin(evaluacion)
            errorminimoxg [NGI]=evaluacion [errorgexperimento]

        for inicia in range (N):

```

```

xr1=np.zeros([1,dim])
xr2=np.zeros([1,dim])
xi=np.zeros([1,dim])
xm=np.zeros([1,dim])
xt=np.zeros([1,dim])
a1=0
fxi=0
fxt=0

posxb=numpy.argmin(evaluacion)
xbest=pp[posxb]
xi[0]=pp[incia]

while(xr1[0][dim-1]==0 or xr2[0][dim-1]==0):
    for j in range(2):
        a=random.randint(0,N-1)
        if(j==0 and incia!=a):
            xr1[0]=pp[a]
            a1=a
        elif(j>0 and incia!=a and a1!=a):
            xr2[0]=pp[a]

xm=xbest+(f*(xr1-xr2))

for j in range(dim):
    xtrand=random.uniform(0,1)
    xtrandj=np.round(1+(xtrand*(1-dim)))
    if(xtrand<=CR or xtrandj==j):
        xt[0][j]=xm[0][j]
    else:
        xt[0][j]=xi[0][j]

y_train , y_pred2 ,NG_graficar2 , fxi=FF(X_train , y_train , xi , 0)
y_train , y_pred3 ,NG_graficar3 , fxt=FF(X_train , y_train , xt , 0)

```



```

        if (fxt<fxi):
            xnew[inicia]=xt
        else:
            xnew[inicia]=xi
    NGG[NGI]=NGI
    pp=xnew

for i in range(N):
    train , pred ,NG_graficar4 ,sumxnew[i]=FF(X_train , y_train , xnew , i)

posminxg=numpy . argmin (sumxnew)
minxg=sumxnew [posminxg]
vectormejor [experimento]=xnew [posminxg]
graf [0] [experimento]=minxg
fig , ax = plt . subplots ()
ax . plot (NGG , errorminimoxg)
plt . xlabel (" N mero de generaci n" , size = 16 ,)
plt . ylabel (" Error cuadratico" , size = 16)
plt . title ('Error por generacion' , fontdict={'family' : 'serif' ,
'color' : 'darkblue' , 'weight' : 'bold' , 'size' : 18})
#plt . show ()
plt . savefig (r"ERROR"+str (x)+".jpg")
plt . close ()

return train , pred , graf , vectormejor

#entrenamiento
etiquetaent , predentrenamiento , grafent , mejorent=kfold (X_train , y_train)

#pruebaevaluacion
etiquetaprue , predprueba , grafprueb , mejorprueb=kfold (X_test , y_test)

```

```
def matriz(y_train , y_pred , tipo):
    porcentaje_train=accuracy_score(y_train , y_pred)*100
    mc_train= confusion_matrix(y_train , y_pred)
    print('Porcentaje de clasificaci n ',tipo,' \n', porcentaje_train,'%')
    print('Matriz de confusi n ' ,tipo,' \n',mc_train)
```

