



**FACULTAD DE INGENIERIA, ARQUITECTURA Y
URBANISMO**

**ESCUELA ACADÉMICO PROFESIONAL DE INGENIERÍA
DE SISTEMAS**

TESIS

**RECONOCIMIENTO DE IMPUREZAS DE
AISLADORES ELÉCTRICOS EN IMÁGENES
DIGITALES**

**PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO
DE SISTEMAS**

Autores:

**Bach. Villegas Vega Myguel Angel Mohamet
Bach. Chapoñan Santisteban Yilmer Justiniano**

Asesor:

Mg. Mejia Cabrera Heber Ivan

Línea de Investigación:

Tecnologías de la Información

**Pimentel – Perú
2018**

RECONOCIMIENTO DE IMPUREZAS DE AISLADORES ELECTRICOS EN IMÁGENES DIGITALES

Aprobación de la Tesis

Mg. Tuesta Monteza Victor Alexci
Presidente del jurado de tesis

Ing. Fuentes Adrianzen Denny John
Secretario del jurado de tesis

Mg. Mejia Cabrera Heber Ivan
Vocal del jurado de tesis



DEDICATORIA

A nuestros familiares que nos apoyaron económicamente, por su infinita comprensión de nuestro amor por esta carrera, con los tiempos de trabajo y alejamiento de ellos y por su amor que nos brindan cada día.

A mi tía Katty, para que sepa que, si yo pude terminar una tesis, la cual, en un primer momento, no sabía qué hacer. Ella también puede lograr vencer todo lo que está pasando. Te quiero mucho.

Myguel Villegas.

A nuestros amigos que siempre nos apoyaron, nos dieron palabras de aliento cuando creíamos que todo estaba perdido.

A nuestros docentes de la Universidad que gracias a ellos nos cautivó este gran tema del procesamiento de imágenes digitales.

Y a pesar de que no sabíamos mucho al respecto, ellos nos apoyaron y ayudaron a aclarar muchas dudas encontradas en el camino.



AGRADECIMIENTO

A Dios

Por darnos la oportunidad de vivir cada día.

A nuestras familias

Por su comprensión y apoyo.

A nuestros docentes de la USS

Por su dedicación y todas las atenciones recibidas.

Myguel Villegas – Yilmer Chapañan



Resumen

Este trabajo se enfoca en los algoritmos y técnicas empleadas para el reconocimiento de impurezas en aisladores eléctricos. Y se desarrolló con el objetivo de poder determinar a tiempo, cuando es que un aislador eléctrico necesita su respectivo mantenimiento preventivo; para impedir cortocircuitos, fugas a tierra y arcos eléctricos. La implementación se propuso en un laboratorio de investigación, limitándose a procesar 200 diferentes imágenes. Las técnicas usadas en la presente investigación fueron las de observación y entrevista a ingenieros eléctricos que nos explicó los principales problemas con respecto a estos aisladores eléctricos en media y alta tensión.

Para poder realizar este trabajo se usaron diferentes algoritmos en cada una de las etapas del procesamiento de imágenes digitales: Pre – Procesamiento (Filtros de Mediana y Filtro Blur Normalizado), Segmentación (Otsu), Extracción de características (Gabor y LBP) y Clasificación (SVM, KNN, Red Neuronal Probabilística y clasificación semiautomático).

Debido a que se usaron 2 algoritmos en el pre procesamiento, un algoritmo en la etapa de segmentación y 2 algoritmos en la etapa de Extracción, se obtuvieron 2 grupos (Gabor y LBP) para cada grupo de algoritmos de Pre Procesamiento.

Después de que se clasificaron todas las imágenes en cada uno de los grupos, se obtuvo un porcentaje en cuanto a rendimiento de 100% con SVM y KNN en base a la clasificación semiautomático con los algoritmos (Filtro Mediana, Otsu, Gabor), seguido de un 80% con todos los algoritmos de clasificación con los algoritmos (Filtro Normalizado, Otsu, LBP). El mejor resultado con el algoritmo de redes neuronales fue de 80% con Filtro Normalizado, Otsu y LBP.



Palabras clave: reconocimiento de imágenes, algoritmos, aisladores eléctricos, impurezas, mantenimiento preventivo.

Abstract

This work focuses on the algorithms and techniques used for the recognition of impurities in electrical insulators. And it was developed with the aim of being able to determine a time, when it is that an electrical insulator needs its respective preventive maintenance; to avoid short circuits, ground leaks and electric arcs. The implementation was proposed in a research laboratory, limiting to recognize 200 different images. The techniques used in the present investigation were the observation and interview of electrical engineers who did not explain the main problems with respect to these electrical insulators in medium and high voltage.

In order to perform this work, different algorithms were used in each of the stages of the digital image processing: Pre - Processing, Segmentation (Otsu), Extraction of characteristics (Gabor and LBP) and SVM Classification, KNN and Neural Probabilistic Network.

Due to the use of two algorithms in the processing, one algorithm in the segmentation stage and two algorithms in the extraction stage, two groups (Gabor and LBP) were obtained for each group of Pre Processing algorithms.

After all the images were classified in each of the groups, a percentage was obtained in terms of 100% performance with SVM and KNN based on the semiautomatico classification with the algorithms (Medium Filter, Otsu, Gabor), followed by 80% with all the classification algorithms with the algorithms (Normalized Filter, Otsu, LBP). The



best result with the neural network algorithm was 80% with Normalized Filter, Otsu and LBP.

Key Words: image recognition algorithms, electrical insulators, impurities, preventive maintenance



INDICE

CAPÍTULO I: PROBLEMA DE INVESTIGACIÓN.....	17
1.1. Situación Problemática.....	17
1.2. Formulación del Problema	20
1.3. Delimitación de la Investigación	20
1.4. Justificación e Importancia de la Investigación.....	21
1.4.1. A nivel Institucional:	21
1.4.2. A nivel de investigación:	21
1.5. Limitaciones de la Investigación	22
1.6. Objetivos de la Investigación.....	22
1.6.1. Objetivo general	22
1.6.2. Objetivos específicos	22
CAPÍTULO II: MARCO TEÓRICO	23
2.1. Antecedentes de Estudios:	23
2.1.1. A nivel Internacional:	23
2.1.2. A nivel Nacional:.....	25
2.1.3. A nivel Local:	26
2.2. Estado del arte	27
2.3. Base teórica científicas.....	29
2.3.1. Algoritmos de Pre Procesamiento:	29
2.3.2. Algoritmos de Segmentación:	41
2.3.3. Algoritmos de Extracción de características:	44
2.3.4. Algoritmos de Clasificación:	49
2.4. Definición de la terminología	72
2.4.1. Algoritmo:	72
2.4.2. Impurezas:	73
2.4.3. Aislador Eléctrico:.....	73
2.4.4. Imágenes digitales:	74
CAPÍTULO III: MARCO METODOLÓGICO.....	75



3.1.	Tipo y Diseño de Investigación	75
3.1.1.	Tipo de investigación:	75
3.1.2.	Diseño de la investigación:	75
3.2.	Población y Muestra.....	75
3.2.1.	Población:.....	75
3.2.2.	Muestra:.....	76
3.3.	Hipótesis.....	76
3.4.	Variables.....	76
3.4.1.	Técnicas de Procesamiento Digital de Imágenes.....	76
3.4.2.	Reconocimiento de impurezas.....	76
3.5.	Operacionalización:.....	77
3.6.	Métodos, técnicas e instrumentos de recolección de datos	77
3.6.1.	Métodos de investigación	77
3.6.2.	Técnicas de recolección de datos	78
3.6.3.	Instrumentos de recolección de datos.....	78
3.7.	Procedimiento para la recolección de datos	79
3.8.	Análisis Estadístico e Interpretación de los datos.....	79
3.8.1.	Sensibilidad:.....	80
3.8.2.	Especificidad:	80
3.8.3.	Exactitud:	80
3.8.4.	Precisión:	80
3.8.5.	Medida F:.....	80
3.9.	Criterios éticos	81
3.10.	Criterios de rigor científico	81
<i>CAPITULO IV: ANALISIS E INTERPRETACIÓN DE LOS RESULTADOS</i>		<i>82</i>
4.1.	Resultados en tablas y gráficos	82
4.1.1.	Resultados Generales:.....	85
<i>CAPÍTULO V: PROPUESTA DE INVESTIGACION.....</i>		<i>103</i>
5.1.	Diseño del Sistema	104



5.1.1. Pre Procesamiento:	105
5.1.2. Segmentación:	106
5.1.3. Extracción de Características:.....	107
5.1.4. Clasificación:	108
5.2. Adquisición y base de imágenes.....	109
5.3. Realización de Objetivos específicos	110
<i>CAPITULO VI: CONCLUSIONES Y RECOMENDACIONES</i>	<i>161</i>
6.1. Conclusiones	161
6.2. Recomendaciones	163
<i>Bibliografía.....</i>	<i>164</i>
<i>ANEXOS.....</i>	<i>167</i>



Índice de figuras

Figura 1 (a) Imagen original para analizar el resultado de los diferentes tipos de filtros estudiados: (b) Imagen de la figura 1(a) corrompida con ruido blanco Gaussiano de varianza 6: (c) Resultado de aplicar el filtro de la mediana. Fuente: Visión por computador imágenes digitales y aplicaciones, Pajares y De la Cruz, RA-MA Editorial. 30

Figura 2 (a) Resultado de filtrar la imagen de la figura 1 con la media aritmética; (b) con la media geométrica; (c) con la media armónica. Fuente: Visión por computador imágenes digitales y aplicaciones, Pajares y De la Cruz, RA-MA Editorial. 33

Figura 3(a) Imagen original de la figura 1(a) corrompida con ruido de tipo sal; (b) Resultado de aplicar el filtro de mínimos con mascara 3x3: (c) Imagen original de la figura1(a) corrompida con ruido de tipo pimienta; (d) Resultado de aplicar el filtro de máximos con máscara de 3x3. Fuente: Visión por computador imágenes digitales y aplicaciones, Pajares y De la Cruz, RA-MA Editorial..... 34

Figura 4 (a) Resultado de aplicar el filtro contra armónico a la imagen con ruido de tipo sal de la figura 3(a) con máscara de 3x3 y orden $R=-3$: (b) Resultado de utilizar el mismo filtro con la misma dimensión de mascara y orden $R=3$ a la imagen corrompida con ruido de pimienta dada en la figura 3(c). Fuente: Visión por computador imágenes digitales y aplicaciones, Pajares y De la Cruz, RA-MA Editorial..... 34

Figura 5 (a) Resultado de aplicar el filtro medio Y_p a la imagen con ruido de tipo sal de la figura 3(a) con máscara de 3x3 y orden $P=-3$; (b) Resultado de utilizar el mismo filtro con la misma dimensión de mascara y orden $P=3$ a la imagen corrompida con ruido de pimienta dada en la figura 3 (c). Fuente: Visión por computador imágenes digitales y aplicaciones, Pajares y De la Cruz, RA-MA Editorial..... 35

Figura 6: Máscara de procesado espacial con $P=5$ 40

Figura 7: Máscara de filtro paso alto de 3 X 3 40

Figura 8: Una red neuronal artificial es un grupo interconectado de nodos similar a la vasta red de neuronas en un cerebro biológico. Cada nodo circular representa una neurona artificial y cada flecha representa una conexión desde la salida de una neurona a la entrada de otra. 50



Figura 34: Imagen de Entrada 180 x 180 - Muestra Paso Bajo. Fuente: Propia. ... 138

Figura 35: Matriz resultante con algoritmo LBP. Fuente: Propia. 139

Figura 36: Matriz de entrada (LBP). Fuente: Propia. 140

Figura 37: Knn, evalúa sus vecinos más cercanos..... 142

Figura 38: Caso linealmente separable. Fuente Opencv..... 144

Figura 39: Búsqueda del hiperplano óptimo para maximizar el margen. Fuente Opencv
..... 145

Figura 40: Grafico que representa una neurona, la cual tiene datos de entrada, capas
ocultas y el dato de salida. Fuente Opencv 147

Figura 41: Red neuronal..... 148



INDICE DE TABLAS

Tabla 1: Tipos de causa de falla CIER. Fuente: COES	17
Tabla 2: Cantidad de imágenes utilizadas para la investigación. Fuente: Propia.	83
Tabla 3: Resultados con diferentes algoritmos de clasificación. Fuente: Propia.	85
Tabla 4: Indicadores de los algoritmos Filtro de Mediana y Gabor. Fuente: Propia.	92
Tabla 5: Indicadores de rendimiento de los algoritmos Filtro de Mediana y Gabor. Fuente: Propia.	93
Tabla 6: Indicadores de los algoritmos Filtro de Mediana y LBP. Fuente: Propia. ...	94
Tabla 7: Indicadores de rendimiento de los algoritmos Filtro de Mediana y LBP. Fuente: Propia.	95
Tabla 8: Indicadores de los algoritmos Filtro Normalizado y Gabor. Fuente: Propia.	97
Tabla 9: Indicadores de rendimiento de los algoritmos Filtro Normalizado y Gabor. Fuente: Propia.	98
Tabla 10: Indicadores de los algoritmos Filtro Normalizado y LBP. Fuente: Propia.	100
Tabla 11: Indicadores de rendimiento de los algoritmos Filtro Normalizado y LBP. Fuente: Propia.	102
Tabla 12: Imágenes usadas y fases usadas en cada una de las etapas. Fuente: Propia.	104
Tabla 13: Identificar patrones en aisladores. Fuente: Propia.	111
Tabla 14: Selección de algoritmos para esta investigación. Fuente: Propia.	113
Tabla 15: Análisis de las características de las imágenes de aisladores limpios y sucios en el algoritmo Gabor. Fuente: Propia.	131
Tabla 16: Costo del Proyecto. Fuente: Propia.	151
Tabla 17: Costo de desarrollo del proyecto. Fuente: Propia.	152
Tabla 18: Costo de Recursos Humanos. Fuente: Propia.	153
Tabla 19: Costo de Útiles de Escritorio. Fuente: Propia.	153
Tabla 20: Costos de Hardware. Fuente: Propia.	154
Tabla 21: Depreciación de Hardware. Fuente: Propia.	155
Tabla 22: Costos de Licencias de Software. Fuente: Propia.	155



Tabla 23: Costos de Materiales de Escritorio. Fuente: Propia.....	156
Tabla 24: Inversión Total. Fuente: Propia.....	156
Tabla 25: Beneficios. Fuente: Electronorte.....	157

Introducción

La presente investigación se refiere al tema de reconocer impurezas en aisladores eléctricos de media tensión, las impurezas son partículas de cualquier material que al tener contacto con estos aisladores produciría una aglomeración y un mal aislamiento de estos aisladores. Su principal función de los aisladores es la de aislar el cable pelado de los postes de luz, para así evitar que la corriente fluya por estos, pudiendo provocar accidentes contra las personas que pudieran tocarlos. Los cables que pasen por estos aisladores, debido al alto voltaje que pase por ellos, generan un campo electromagnético provocando así una imantación y atracción de partículas, generalmente metalizadas.

Debido a esta problemática lo que se hacía comúnmente con estos aisladores; era realizarse un estudio previo de que aisladores requerían un mantenimiento preventivo con mayor urgencia que otras secciones. Según las normas técnicas de Osinergmin estos mantenimientos deberían darse con una frecuencia de tiempo de una vez al año. Pero estos mantenimientos algunas veces podrían retrasarse bastante tiempo debido a que no se cuenta con personal y tiempo para realizar estos análisis previos, ocasionando así que estos mantenimientos se produzcan algunas veces, una vez cada dos años.

La excesiva aglomeración de partículas en los aisladores produce un mal aislamiento en los aisladores pudiendo provocar arcos eléctricos, fugas a tierra o cortocircuitos.



La investigación de esta problemática generó una ayuda a nivel institucional, porque las empresas que se encargan de distribuir energía eléctrica, pudieron obtener estos estudios previos para determinar que aisladores están sucios a tiempo. Así como también generó una ayuda en el campo de reconocimiento de imágenes, ya que se usaron diversos algoritmos al reconocer forma, color y textura de cada aislador y posteriormente se usó otros algoritmos para clasificarlos. Y finalmente a nivel social, porque se les brindaría un mejor servicio a la población.

La investigación lo que quiere realizar es un reconocimiento de impurezas en los aisladores eléctricos con el uso de la técnica de reconocimiento de imágenes, utilizando así diversos algoritmos en las etapas de pre-procesamiento, procesamiento, caracterización y clasificación de esta técnica.

Para esto nos basamos principalmente en 3 papers para poder elaborar un plan de proceso de los algoritmos a utilizar. En el paper sobre identificar defectos de la superficie de la madera basados en su textura, se usó una metodología híbrida que combinó GLCM para la extracción de características y redes de Retro propagación para el proceso de clasificación, logrando una precisión de 92.67%.

En el paper sobre el reconocimiento de defectos de óxido en el recubrimiento de puente de acero automatizada basada en características de color y textura, se usó detectar óxido en el acero basado en el color y textura, usando la transformada de Fourier, filtros Gauss, filtro de paso bajo para el proceso de



pre-procesamiento, caracterización y el algoritmo de k-means para el proceso de clasificación. Finalmente, en el paper sobre el reconocimiento de la moneda mediante la rotación invariante de las regiones binarias basadas en imágenes de magnitudes gradiente, lo que se quiso lograr fue determinar la veracidad de una moneda; a través de magnitud RFR-Gradiente para la extracción de características,

RFR-GM fue extraído de gradientes magnitudes en las imágenes de la moneda local y RFR utilizando magnitud diferencia local de transformación para aumentar la precisión del reconocimiento de la moneda.

Con esta investigación se quiere lograr con los algoritmos usados una precisión del reconocimiento de las impurezas en los aisladores eléctricos de más del 80%.

CAPÍTULO I: PROBLEMA DE INVESTIGACIÓN

1.1. Situación Problemática

Según un estudio realizado por las Naciones Unidas; la población mundial entre los años del 2010 y 2014, creció a un ritmo del 1.2 % anual, frente al 1.5% al que lo hacía en 1994. El crecimiento de la población genera mayor demanda de energía eléctrica, esto, se puede hallar en el portal web de la International Energy Agency (IEA), donde se muestra que, entre los años del 2009 al 2013, países como Estados Unidos produjo un total de energía eléctrica de 4170 a 4290 Teravatios por hora (TWh). China produjo un crecimiento de 3740 a 5440 TWh y Perú obtuvo una creciente de 32.93 a 43.37 TWh.

(COES, 2015) Muestra diferentes tipos de causas de falla de la energía eléctrica en las líneas de transmisión en el año 2015 de Perú, entre los que hubo cortes de energía eléctrica por tiempos prolongados.

Tabla 1: Tipos de causa de falla CIER. Fuente: COES

Tipos de causa de falla CIER	%	Numero de Fallas
NO IDE	28,87%	140
FNA	52,16%	253
OTR	11,75%	57
FEC	5,77%	28
EXT	1,24%	6
FEP	0,21%	1
TOTAL	100%	485



Es por ello que se necesita que, en el proceso de distribución de la energía a los usuarios finales, no haya problemas de ningún tipo; para esto, las empresas que, prestan estos servicios necesitan tener mantenimientos apropiados a sus equipos, materiales, entre otros. (Venero, 2011) En sus normas técnicas establece que, los aisladores eléctricos deben tener mantenimiento como mínimo una vez al año.

Entre estos materiales necesarios para la transmisión de la energía eléctrica tenemos al aislador eléctrico, que es un componente esencial cuando hablamos de transmitir electricidad, este se encuentra soportando corrientes de 220 kv a 380 kv y su principal función es la de aislar el paso de la electricidad a los postes de mediana tensión y torres de alta tensión.

(Cano, 2005) En su boletín técnico anual afirma que el problema surge cuando estos aisladores al estar expuestos a la intemperie, tienden a llenarse de impurezas inorgánicas como humos, vapores químicos – fungicidas, polvo y ceniza; orgánicas, como bacterias, microorganismos, esporas, vegetales, gérmenes, polen y polinia; acuosas, como neblina, niebla, vapor de agua, rocío, llovizna y lluvia. Así mismo, se detalla los efectos de los depósitos contaminantes sobre los aisladores entre los que se tiene, excesivas corrientes de fuga, flameos continuos, perforaciones, corrosión y radio interferencias.

Para solucionar el problema se está proponiendo utilizar técnicas y algoritmos de procesamiento de imágenes digitales tomadas a estos aisladores eléctricos y reconocer estas impurezas adheridas basadas en su textura lisa y



color, según el tipo de material que son de vidrio o porcelana (Santana, Osorio, Mari, & Sanín, 2012).

Hasta donde se ha indagado, no se han encontrado investigaciones referidas al reconocimiento de impurezas en aisladores eléctricos, utilizando procesamiento de imágenes digitales, pero existen investigaciones relacionadas a técnicas en base a textura como (Ozturk & Akdemir, 2015) que en su investigación comparación de algoritmos de detección de borde para el análisis de textura en la producción de vidrio, enfrentó el problema de detección de imperfecciones en vidrios como: ralladuras, grietas, arañazos y burbujas, utilizando procesamiento de imágenes. Se usaron diferentes algoritmos de detección de bordes, pero finalmente se decidió usar el algoritmo de bordes de Laplacian of Gaussian (LoG) por obtener mejores resultados.

(YongHua & W, 2014) En su investigación sobre la identificación de defectos de la superficie de madera basados en textura, uso técnicas de procesamiento de imágenes para detectar imperfecciones, utilizó un método híbrido el cual combina Tamura textura y GLCM, los resultados fueron positivos porque se obtuvo una precisión de detección mayor al 90%.

(Shen, Chen, & Chang, 2013) En su investigación reconocimiento de defectos óxido recubrimiento puente de acero automatizada basada en el color y textura, enfrentó el problema de detección de defectos en un puente de acero como: el óxido, utilizando procesamiento de imágenes, emplearon la metodología RUDERM, la cual tiene ventaja en ambientes con iluminación no uniforme, el tiempo de procesamiento fue corto, es ideal para emplear en tiempo



real, se aplicó en puentes de diferentes colores y la precisión que se obtuvo fue mayor al 90%.

Basados en las investigaciones anteriores donde se utilizaron técnicas y algoritmos para procesar imágenes de madera, vidrio y metal se obtuvo en su mayoría resultados aceptables, es por ello que en el presente trabajo de investigación se pretende identificar de manera automática impurezas en los aisladores eléctricos mediante imágenes digitales, haciendo uso de técnicas y algoritmos de procesamiento digital de imágenes propuesta por otros investigadores.

1.2. Formulación del Problema

¿Cómo identificar en forma automática impurezas adheridas en imágenes digitales de aisladores eléctricos?

1.3. Delimitación de la Investigación

El presente proyecto se desarrolló en la universidad Señor de Sipán ubicada en el distrito de Pimentel, provincia de Chiclayo, departamento de Lambayeque - Perú. Los interesados en esta investigación fueron los estudiantes de la escuela académica de ingeniería de sistemas; Myguel Angel Mohamet Villegas Vega y Yilmer Justiniano Chapoñan Santisteban, con el asesoramiento del Ing. Heber Ivan Mejía Cabrera. Logrando desarrollarlo en un periodo de 6 meses.

1.4. Justificación e Importancia de la Investigación

Con el presente proyecto se espera resolver el principal problema, de detectar impurezas en imágenes digitales de aisladores eléctricos de manera automática; para que se pueda determinar cuándo los aisladores eléctricos requerirán un mantenimiento preventivo. Todo esto con el objetivo de evitar el acumulo de impurezas en los aisladores y así; impedir excesivas corrientes de fuga, flameos continuos, perforaciones, corrosión y radio interferencias.

1.4.1. A nivel Institucional:

Esta investigación ayudaría a las compañías de energía eléctrica porque a través del software que se propondrá, realizará un análisis con mayor rapidez y eficiencia de los aisladores eléctricos mediante imágenes digitales en tendidos eléctricos determinando así cuáles de estos componentes requieren o no mantenimiento. Evitando y/o reduciendo cortes inesperados de fluido eléctrico que pueden generar inconformidad a la población demandante.

1.4.2. A nivel de investigación:

Actualmente, en cuanto a investigaciones de reconocimiento de imágenes digitales de aisladores eléctricos, no se han encontrado estudios realizados, mucho menos en ambientes no controlados. Es por eso que, se propone efectuar esta investigación analizando y comparando técnicas y algoritmos más eficientes en reconocimiento de patrones (textura, color, etc.)



en imágenes digitales haciendo uso de la tecnología procesamiento de imágenes, que solucione la problemática planteada y nos permita desarrollar un software que sea capaz de identificar las impurezas de los aisladores eléctricos ayudando así a reducir tiempos y costos.

1.5. Limitaciones de la Investigación

Una de las principales dificultades que se encontró al desarrollar esta investigación fue la de no disponer de una base de datos referentes a aisladores eléctricos de media o alta tensión; para lo que se dispuso, elaborar nosotros mismos una base de datos de imágenes tomadas por los mismos investigadores con la ayuda de una cámara fotográfica.

1.6. Objetivos de la Investigación

1.6.1. Objetivo general

Identificar impurezas adheridas de manera automática en imágenes digitales de aisladores eléctricos.

1.6.2. Objetivos específicos

- a) Identificar patrones en los aisladores eléctricos (Textura, color).
- b) Seleccionar algoritmos para el reconocimiento de impurezas en el aislador.
- c) Implementar algoritmos seleccionados que permitan realizar un análisis efectivo de la imagen.
- d) Evaluar la efectividad de los algoritmos.
- e) Realizar la evaluación económica de la propuesta.



CAPÍTULO II: MARCO TEÓRICO

2.1. Antecedentes de Estudios:

2.1.1. A nivel Internacional:

Aristondo (2010). En su investigación *algoritmo de reconocimiento de forma y color para una plataforma robótica* desarrollo un sistema de control de un robot para que se capaz de forma autónoma recoger pelotas basados en su color y forma. Para el proceso de recogida de la pelota se utilizó el algoritmo de filtros de Kalman, para el color se hizo un comparación de modelos de colores RGB, YUV y HSL siendo estos dos últimos aceptables en cuanto a resultados porque en la mayoría de pruebas con iluminación adecuada e iluminación inadecuada se identificó de manera eficiente con un 71.1% con iluminación inadecuada y subió aún 87.4% de efectividad con iluminación adecuada, es decir en la mayoría de pelotas según los colores que se le ordenaba recoger en cuanto a resultados fueron tolerables y para la detección de la forma se utilizó el algoritmo de Shen y Castan, la cual estaba implementado en un librería MIL(Matrix Imaging Library).

En esta investigación se logró percibir que el algoritmo de detección de bordes Shen y Castan fue de gran importancia para lograr resultados positivos, si bien se sabe que existen bastantes algoritmos para la detección de bordes algunos se comportan de mejora manera dependiendo a las circunstancias en las que se encuentren.



Xiaorong, Ke, Xiong, Lupin, Zhibu (2014) En su investigación *reconocimiento de expresiones faciales de secuencia de imágenes usando técnicas de árbol de decisiones* se quiso lograr reconocer expresiones faciales. Para realizar este estudio se usaron métodos de modelo de apariencia Activa (AAM), aplicación piramidal de Lukas – Kanade y árboles de decisiones; se pudo apreciar que estos métodos son más efectivos que los usados en la red Bayesiana. También se pudo apreciar que la técnica de Árboles de Decisiones es más efectiva que la clasificación SVM (Support Vector Machine). Con esta investigación se obtuvieron resultados del 89.37% de efectividad que supero al 86.3 % si se hubiera usado la red Bayesiana.

En esta investigación se logró apreciar que los métodos usados fueron mejores que la red Bayesiana y que los arboles de decisiones fueron más efectivos que la clasificación SVM.

Saban, Bayram (2015) En su investigación *comparación de algoritmos de detección de borde para el análisis de textura en la producción de vidrio* se usaron diferentes algoritmos de detección de bordes, pero finalmente se decidió usar el algoritmo de bordes de LoG (Laplacian of Gaussian) por obtenerse mejores resultados. Se obtuvieron resultados del 76.52% usando el algoritmo de LoG y se comprobó ser mejor que los otros. Finalmente, los resultados obtenidos no fueron tan exitosos como los filtros de análisis de textura como el filtro de Gabor o la transformada de Wavelet,



pero en comparación a los otros, se obtiene una sencillez en el uso y un trabajo mucho más rápido.

En esta investigación se reconoció que usando el algoritmo de bordes de LoG se obtuvieron mejores resultados al compararlo con los demás algoritmos usados.

Portilla, Menéndez, Martínez (2016). En su investigación “Implementación del Algoritmo de Otsu sobre FPGA.”, implementaron un sistema para la detección del umbral óptimo en imágenes en escala de grises con un tamaño de 256x256 píxeles siguiendo el algoritmo propuesto por Otsu. El sistema propuesto es una implementación híbrida hardware-software, con la finalidad de utilizar menos recurso computacional, utilizar el método en tiempo real y evitar el empleo de divisores y casi la totalidad de multiplicadores. Se utilizó el lenguaje VHDL para el algoritmo Otsu, en cuanto a los resultados fueron satisfactorios ya que el objetivo principal era utilizar menos recurso computacional para utilizarlo en el desarrollo de las siguientes etapas de procesamiento de imágenes.

En esta investigación se logró detectar que el algoritmo de Otsu es el más utilizado en el procesamiento de imágenes digitales.

2.1.2. A nivel Nacional:

Gavidia (2014). En su trabajo de investigación *segmentación de imágenes médica mediante algoritmos de colonia de hormigas* plantea el uso de un método no-tradicional para la segmentación de imágenes



médicas haciendo énfasis en resonancias magnéticas cerebrales basado en colonias de hormigas artificiales”. La investigación buscó combinar dos algoritmos de colonia de hormigas de diferentes campos en que se aplicaron para segmentar resonancias electromagnéticas cerebrales para medir la efectividad y calidad de su aplicación, para lograr este objetivo se siguieron una secuencia de pasos básicos de los algoritmos originales, que incluían asignar hormigas a píxeles y construir rutas con valores de escala de grises bajos así como el depósito de feromonas dependiendo de la calidad de la ruta obtenida en el algoritmo de Malisia y la utilización de Max-Min Ant System para obtener particiones óptimas tomando en cuenta la vecindad de los píxeles en el algoritmo de Ouadfel, los resultados experimentales que la investigación mostro similitudes altas para ciertas segmentaciones mientras que para otros la segmentación si bien fue inferior se mantuvo en niveles aceptables.

En esta investigación se puede identificar claramente que el proceso de segmentación es vital en el reconocimiento de imágenes ya que probablemente sea un punto crítico para lograr un análisis efectivo en las imágenes digitales. Por lo tanto es necesario lograr una buena segmentación de las imágenes en el reconocimiento de patrones de imágenes para que el resto de procesos se logre un resultado positivo.

2.1.3. A nivel Local:

Montaño, Huaman (2013) En su trabajo de investigación *desarrollo de*



un software de monitoreo para el proceso de germinación de plantones usando patrones de reconocimiento de imágenes se analizaron las características de los plantones según el tiempo y el tipo de semilla germinada. Obteniendo características del tamaño que ocupa el plantón dentro de la celda de la bandeja y el color verde predominante de esta celda. Al realizar este estudio se integraron múltiples técnicas, desarrollando un procedimiento sólido y robusto que genero mejores resultados. Entre estas metodologías tenemos K – Means, Canny, Dilatación, Erosión, CSS (Curvatura, Escala, Espacio) y Cortes. Usando estas técnicas se optimizaron los tiempos de trabajo y la documentación de reportes que antes se hacían de forma manual. Evaluando los resultados, se aprecia una reducción del 49.8 % en los tiempos de monitoreo de las bandejas del vivero a comparación del proceso manual. Así mismo, se corroboró un índice de acierto del software de 95.71%.

2.2. Estado del arte

Saban, Bayram (2015) En su investigación *comparación de algoritmos de detección de borde para el análisis de textura en la producción de vidrio* se usaron diferentes algoritmos de detección de bordes, pero finalmente se decidió usar el algoritmo de bordes de LoG (Laplacian of Gaussian) por obtenerse mejores resultados. Se obtuvieron resultados del 76.52% usando el algoritmo de LoG y se comprobó ser mejor que los otros. Finalmente, los resultados



obtenidos no fueron tan exitosos como los filtros de análisis de textura como el filtro de Gabor o la transformada de Wavelet, pero en comparación a los otros, se obtiene una sencillez en el uso y un trabajo mucho más rápido.

Esta investigación se reconoció que usando el algoritmo de bordes de LoG se obtuvieron mejores resultados al compararlo con los demás algoritmos usados.

YongHua, Jin-Cong (2014). En su investigación sobre la identificación de defectos de la superficie de madera basados en textura. El objetivo fue aplicar reconocimiento de imágenes para detectar tres defectos comunes en la madera como nudos muertos, postes y nudos de vida de la madera, para ello se utilizó una metodología híbrida la cual combinaba Tamura textura y el método GLCM para obtener un mejor resultado, se investigó profundamente sobre la segmentación de imágenes, se utilizó GLCM para la extracción de características y la parte de clasificación se utilizó redes de retro propagación (BP), se implementó en Matlab 7.0, se seleccionó 300 muestras imágenes de madera con defectos, 16 tipos de madera blanda , 34 tipos de madera dura, de los resultados se obtuvo una precisión mayor al 90%, resultados aceptables.

Shen, Chen, Chang (2013). En su investigación reconocimiento de defectos óxido recubrimiento puente de acero automatizada basada en el color y la textura. El objetivo fue detectar defectos en el acero como el óxido haciendo uso del procesamiento de imágenes, se utilizó el método RUDERM la cual tiene una ventaja en el manejo de la iluminación no uniforme, se utilizó transformada de Fourier, filtros Gauss, filtro de paso bajo, Algoritmo de k-means. En la etapa



de prueba se utilizaron 140 imágenes de óxido artificiales normales y 560 imágenes artificiales de iluminación no uniforme, este método es ideal para utilizar en tiempo real ya que el tiempo de procesamiento es corto.

Kim, Ho, Man (2014). En su trabajo de investigación reconocimiento de la moneda mediante la rotación invariante región binaria basada en imágenes patrones basados en magnitudes gradiente el objetivo fue determinar si una moneda era falsa a través de algoritmos de textura, el método propuesto fue magnitudes RFR-gradiente(RFR-GM), se utilizó Magnitud RFR-gradiente para la extracción de características y RFR utilizando magnitud diferencia local de transformación para aumentar la precisión del reconocimiento de la moneda. El método propuesto fue muy robusto frente a la rotación y frente a ruidos de la moneda (abrasión, oxidación, y polvo), el método propuesto fue mejor, dicha investigación fue implementada en Matlab.

2.3. Base teórica científicas

2.3.1. Algoritmos de Pre Procesamiento:

2.3.1.1. Filtrado de la mediana

Pajares, De la Cruz (2002) definen que la mediana M de un conjunto de valores es tal que la unidad de los valores del conjunto es menor que M y la mitad de los valores son mayores que M . Con el objeto de realizar un filtrado de la mediana en el entorno de vecindad, ordenamos las intensidades de la vecindad como se ha explicado anteriormente, determinamos la mediana y asignamos esta última a la intensidad de pixel. Por ejemplo, en un entorno



de vecindad 3x3 la mediana es el quinto valor más grande, en un entorno de 5x5 la mediana es el décimo tercer valor más grande, y así sucesivamente. Cuando varios pixeles en el entorno de vecindad tienen el mismo valor, se agrupan de la siguiente forma: por ejemplo, supongamos que un entorno de vecindad 3x3 tiene los valores (20, 30, 10, 50, 50, 60, 50, 50, 30). Estos valores se ordenarán como sigue, (10, 20, 30, 30, 50, 50, 50, 50, 60) obteniéndose el valor 50 de la mediana. La principal función del filtrado de la mediana es hacer que los puntos con intensidades muy distintas se hagan muy parecidos a sus vecinos, eliminando así los picos de intensidad que aparezcan aislados en el área de la máscara de filtro.

En la figura 1(a) se muestra una imagen original, que sirve de base para el estudio de los diferentes tipos de filtros usados a lo largo del capítulo. En la figura 1(b) se muestra la imagen de la figura 1(a) corrompida con ruido blanco (media nula) Gaussiano de varianza 6 y en (c) el resultado tras aplicar el filtro de la mediana.

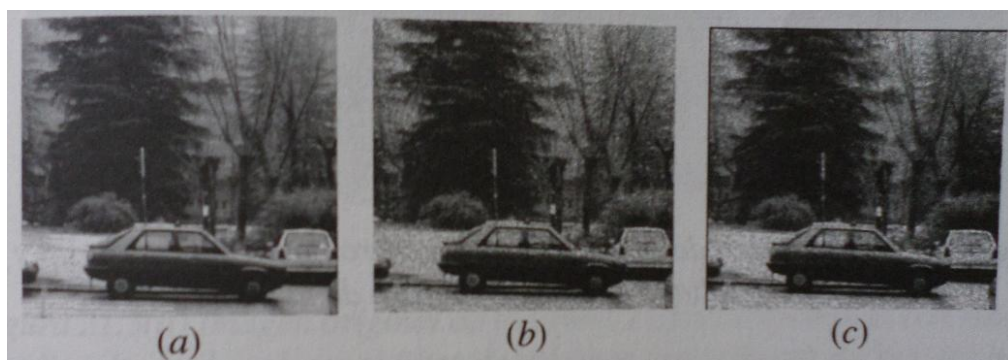


Figura 1 (a) Imagen original para analizar el resultado de los diferentes tipos de filtros estudiados: (b) Imagen de la figura 1(a) corrompida con ruido blanco Gaussiano de varianza



6: (c) Resultado de aplicar el filtro de la mediana. Fuente: *Visión por computador imágenes digitales y aplicaciones*, Pajares y De la Cruz, RA-MA Editorial.

2.3.1.2. Filtrado de medias

Pajares, De la Cruz (2002) definen que los filtros de las medias funcionan mediante la definición de algún tipo de promediado sobre el entorno de vecindad $n \times n$ de la ventana. El más básico es el filtro de la media aritmética, que calcula la media aritmética de los pixeles en la ventana, como sigue,

$$Ma = \frac{1}{nm} \sum_{(x,y) \in W} f(x,y) \quad \text{Ecuación (1)}$$

Donde nm es el número de pixeles en la ventana W de dimensión $n \times m$. el filtro de la media aritmética suaviza las variaciones locales dentro de una imagen, de modo que esencialmente es un filtro paso bajo. Se puede implementar mediante una máscara de convolución donde todos los coeficientes de la máscara son $1/nm$.

El *filtro media geométrica* trabaja mejor con ruido Gaussiano y retiene mejor los detalles de la información que el filtro media aritmética. Está definido como el producto de los valores de los pixeles dentro de la ventana, elevados a la potencia $1/nm$:

$$Mg = \prod_{(x,y) \in W} [f(x,y)]^{1/nm} \quad \text{Ecuación (2)}$$



Este filtro falla con ruido del tipo sal y pimienta.

El *filtro de la media armónica* falla con ruido de pimienta, pero trabaja bien con ruido de tipo sal. Está definido como sigue,

$$Mar = \frac{nm}{\sum_{(x,y) \in W} \frac{1}{f(x,y)}} \quad \text{Ecuación (3)}$$

Este filtro trabaja bien con ruido Gaussiano, manteniendo los detalles de la imagen mejor que el filtro de la media aritmética.

En la figura 2(a) se muestra el resultado de filtrar la imagen de la figura 1(b) con la media aritmética. En la figura 2(b) con la media geométrica; y en (c) con la media armónica. Obsérvese que los mejores resultados se obtienen con la media aritmética.

El *filtro medio contra-armónico* trabaja bien para imágenes conteniendo ruido del tipo sal o pimienta dependiendo del orden R del filtro:

$$Mca = \frac{\sum_{(x,y) \in W} f(x,y)^{R+1}}{\sum_{(x,y) \in W} f(x,y)^R} \quad \text{Ecuación (4)}$$

Donde W es la ventana bajo consideración de dimensión n x m. Para valores de R negativos elimina el ruido sal, mientras para valores positivos de R, elimina el ruido de tipo pimienta.



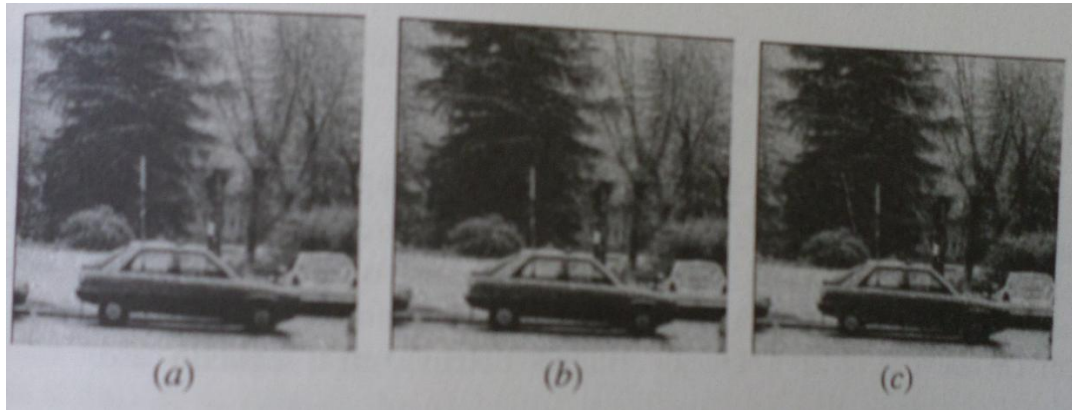


Figura 2 (a) Resultado de filtrar la imagen de la figura 1 con la media aritmética; (b) con la media geométrica; (c) con la media armónica. Fuente: *Visión por computador imágenes digitales y aplicaciones, Pajares y De la Cruz, RA-MA Editorial.*

En la figura 4(a) se muestra el resultado de aplicar el filtro contra armónico a la imagen con ruido de tipo sal de la figura 3(a) con máscara de 3x3 y orden $R = -3$. En la figura 4(b) aparece el resultado de usar el mismo filtro con la misma dimensión de máscara y orden $R = 3$ a la imagen corrompida con ruido de pimienta dada en la figura 3(c). Se observa que el mejor resultado se obtiene en el segundo caso.



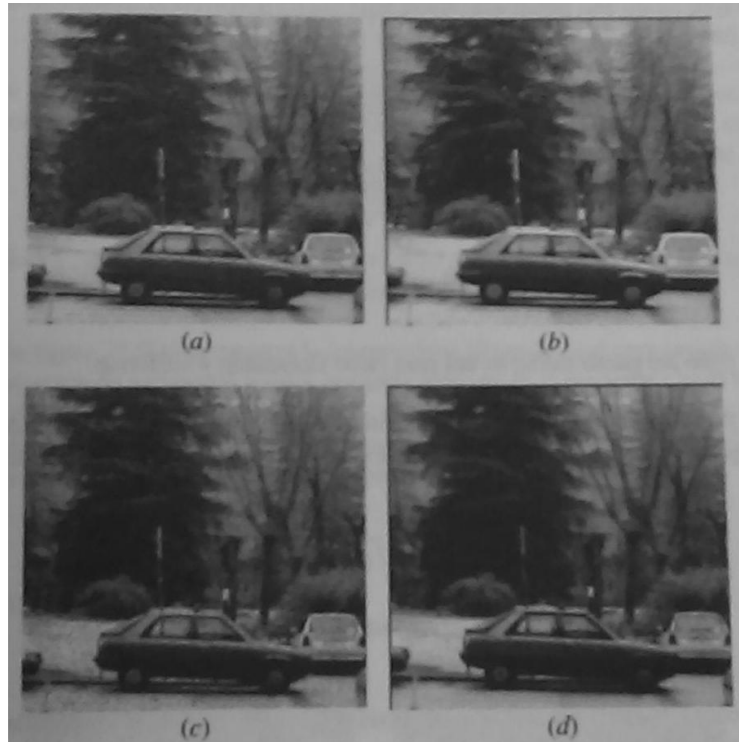


Figura 3(a) Imagen original de la figura 1(a) corrompida con ruido de tipo sal; (b) Resultado de aplicar el filtro de mínimos con máscara 3x3; (c) Imagen original de la figura1(a) corrompida con ruido de tipo pimienta; (d) Resultado de aplicar el filtro de máximos con máscara de 3x3. Fuente: *Visión por computador imágenes digitales y aplicaciones*, Pajares y De la Cruz, RA-MA Editorial.

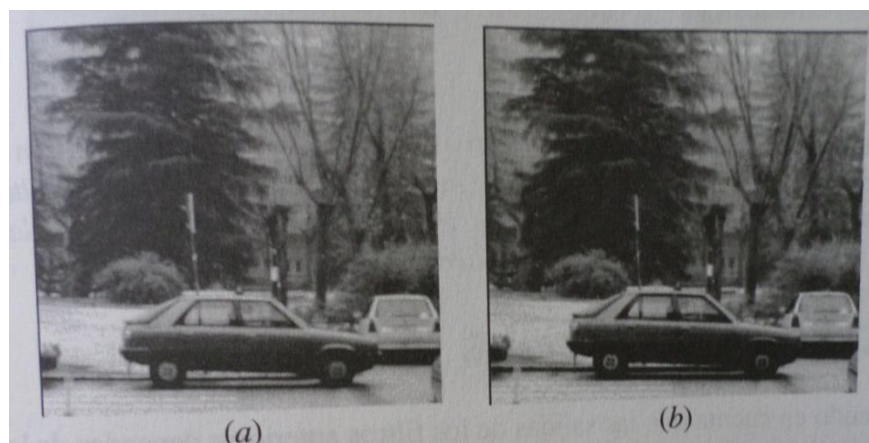


Figura 4 (a) Resultado de aplicar el filtro contra armónico a la imagen con ruido de tipo sal de la figura 3(a) con máscara de 3x3 y orden $R=-3$; (b) Resultado de utilizar el mismo filtro con la misma dimensión de máscara y orden $R=3$ a la imagen corrompida con ruido de pimienta



dada en la figura 3(c). Fuente: *Visión por computador imágenes digitales y aplicaciones*, Pajares y De la Cruz, RA-MA Editorial.

El filtro medio MY_p está definido como sigue:

$$MY_p = \left[\sum_{(x,y) \in W} \frac{f(x,y)^p}{nm} \right]^{1/p} \quad \text{Ecuación (5)}$$

Este filtro elimina el ruido de sal para valores negativos de P y el ruido de pimienta para valores positivos de P.

En la figura 5(a) se muestra el resultado de aplicar el filtro medio MY_p a la imagen con ruido de tipo sal de la figura 3(a) con máscara de 3x3 y orden P = -3. En la figura 5(b) aparece el resultado de utilizar el mismo filtro con la misma dimensión de mascara y orden P = 3 a la imagen corrompida con ruido de pimienta dada en la figura 3(c). De nuevo se observa el mejor resultado obtenido en el segundo caso.

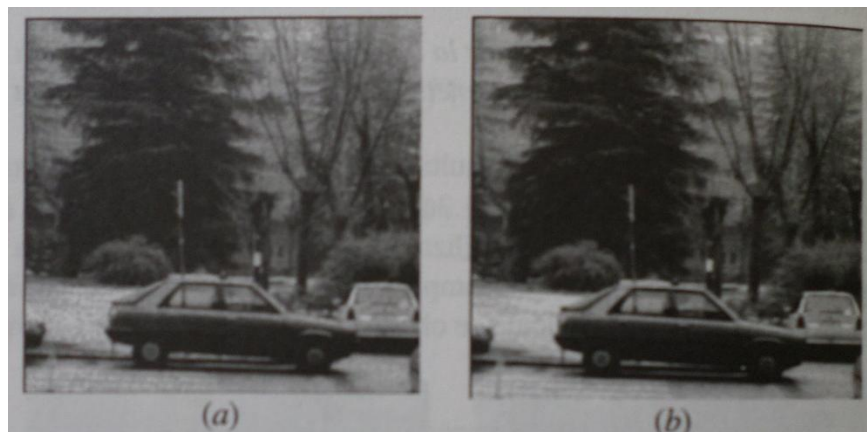


Figura 5 (a) Resultado de aplicar el filtro medio Y_p a la imagen con ruido de tipo sal de la figura 3(a) con máscara de 3x3 y orden $P=-3$; (b) Resultado de utilizar el mismo filtro con la misma dimensión de mascara y orden $P=3$ a la imagen corrompida con ruido de pimienta dada en la figura 3 (c). Fuente: *Visión por computador imágenes digitales y aplicaciones*, Pajares y De la Cruz, RA-MA Editorial.



2.3.1.3. Filtrado de Wiener

Pajares, De la Cruz (2002) definen que sean R_f y R_n las matrices de correlación de f y n definidas respectivamente por las ecuaciones.

$$R_n = E\{nn^T\} \quad y \quad R_f = E\{ff^T\} \quad \text{Ecuación (6)}$$

Donde $E\{\}$ indica el valor esperado de la operación, y f y n serán definidas posteriormente. El elemento ij -ésimo de R_f está dado por $E\{f_i f_j\}$ que es la correlación entre los elementos i y j de f . igualmente, el elemento ij -ésimo de R_n da la correlación entre los dos elementos correspondientes en n . puesto que los elementos de f y n son reales. $E\{f_i f_j\} = E\{f_j f_i\}$, $E\{n_i n_j\} = E\{n_j n_i\}$, y se sigue que R_f y R_n son matrices reales simétricas. Para muchas funciones de imagen la correlación entre pixeles (esto es elementos de f o n) no se extiende más allá de una distancia de 20 a 30 pixeles en la imagen, de modo que una matriz de correlación típica tiene una banda de elementos distintos de cero sobre la diagonal principal y ceros en las regiones inferior izquierda y superior derecha. Basándose en esta suposición de que la correlación entre cualesquiera dos pixeles es una función de la distancia entre los pixeles y no de su posición, R_f y R_n pueden aproximarse por matrices circulares por bloques y por tanto pueden diagonalizarse por la matriz W mediante el procedimiento descrito en el Apéndice m sobre matrices circulares por bloques (Andrews y Hunt 1977). Utilizando A y B para designar matrices se obtiene:

$$R_f = WAW^{-1} \quad y \quad R_n = WBW^{-1} \quad \text{Ecuación (7)}$$



Así como los elementos de la matriz diagonal D en $H = WDW^{-1}$ corresponden a la transformada de Fourier de los elementos bloque de H , los elementos de A y B son las transformadas de los elementos de correlación en R_f y R_n respectivamente. Las transformadas de Fourier de esas correlaciones se denominan *espectros de potencia o densidad espectral* de $f_e(x, y)$ y $n_e(x, y)$, respectivamente, y se denotan como $S_f(u, v)$ y $S_n(u, v)$ a partir de ahora.

Definiendo

$$Q^T Q = R_f^{-1} R_n \quad \text{Ecuación (8)}$$

Y sustituyendo esta expresión en $\hat{f} = (H^T H + \gamma Q^T Q)^{-1} H^T g$ se obtiene,

$$\hat{f} = (H^T H + \gamma R_f^{-1} R_n)^{-1} H^T g \quad \text{Ecuación (9)}$$

Utilizando la ecuación (7) se obtiene,

$$\hat{f} = (WD^*DW^{-1} + \gamma WA^{-1}BW^{-1})^{-1} WD^*W^{-1}g \quad \text{Ecuación (10)}$$

Multiplicando ambos lados por W^{-1} y realizando alguna manipulación de matrices la ecuación anterior se reduce a,

$$W^{-1}\hat{f} = (D^*D + \gamma A^{-1}B)^{-1}D^*W^{-1}g \quad \text{Ecuación (11)}$$

Teniendo en cuenta el significado de los elementos de A y B , que las matrices dentro del paréntesis son diagonales y haciendo uso de los conceptos desarrollados en el Apéndice M sobre efectos de la diagonalización en el



modelo de degradación, podemos escribir los elementos de la ecuación anterior de la forma,

$$\hat{F}(u, v) = \left[\frac{H^*(u, v)}{|H(u, v)|^2 + \gamma[S_\eta(u, v)/S_f(u, v)]} \right] G(u, v) \tag{Ecuación (12)}$$

$$= \left[\frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + \gamma[S_\eta(u, v)/S_f(u, v)]} \right] G(u, v)$$

Para $u, v = 0, 1, 2, \dots, N-1$ donde $|H(u, v)|^2 = H^*(u, v)H(u, v)$ y se supone que $M = N$.

Cuando $\gamma = 1$, el término dentro de los corchetes externos en (12) se reduce al también denominado *filtro de Wiener*. Si γ es variable la expresión anterior se denomina *filtro de Wiener paramétrico*. En ausencia de ruido, $S_\eta(u, v) = 0$ y el filtro de Wiener se reduce a un filtro ideal inverso. No obstante, cuando $\gamma = 1$, el uso de (12) no proporciona una solución óptima, porque como se ha mencionado en dicha sección, γ se debe ajustar para satisfacer la restricción $|g - H\hat{f}|^2 = |n|^2$. Se puede demostrar, no obstante, que la solución obtenida con $\gamma = 1$ es óptima en el sentido de que minimiza la siguiente cantidad $E \{ [f(x, y) - \hat{f}(x, y)]^2 \}$. Claramente esto es un criterio estadístico que trata a f y \hat{f} como variables aleatorias.

Cuando $S_f(u, v)$ y $S_\eta(u, v)$ son desconocidas (un problema muy frecuente en la práctica) la ecuación (12) puede aproximarse por,



$$\hat{F}(u, v) = \left[\frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + K} \right] G(u, v) \quad \text{Ecuación (13)}$$

Donde K es una constante a determinar mediante ensayo y error.

2.3.1.4. Filtro Blur Normalizado

El filtro Blur normalizado es un filtro de los muchos usados de los filtros de paso bajo. Este filtro es un filtro de promediado. Por ello, los coeficientes de la máscara de la figura 8 serán todos positivos; las diferentes implementaciones darán más o menos importancia al pixel central respecto de los vecinos, pero el comportamiento global es similar. Es práctica habitual que los coeficientes se normalicen, de forma que su suma sea igual a uno. De este modo, la ganancia del filtro para frecuencias espaciales nulas es unitaria, de forma que la componente continua de la imagen no se ve alterada.

El normalizado es un filtro paso bajo que atenúa las componentes de medias-bajas frecuencias y dejan intactas las bajas en función de la frecuencia de corte que se elija. Se usan también para eliminar todo lo que no sean variaciones suaves de nivel de gris.

Este tipo de filtrado tiene como objeto reducir el ruido de adquisición cuando sólo se disponga de una única imagen. El inconveniente de esta técnica es el suavizado de las transiciones (debido al recorte de componentes espectrales de alta frecuencia) de forma que el aspecto final



de la imagen ser a de un cierto desenfoque, tanto mayor cuanto más grande sea el filtro.

Los filtros de paso bajo son usados para suavizar la imagen, es decir equilibran los valores de los pixeles de la imagen.

λ_1	λ_2	λ_3	λ_4	λ_5
λ_6	λ_7	λ_8	λ_9	λ_{10}
λ_{11}	λ_{12}	λ	λ_{13}	λ_{14}
λ_{15}	λ_{16}	λ_{17}	λ_{18}	λ_{19}
λ_{20}	λ_{21}	λ_{22}	λ_{23}	λ_{24}

Figura 6: Máscara de procesamiento espacial con P=5

2.3.1.5. Filtro Paso Alto

En este caso los coeficientes de la máscara serán positivos y negativos y típicamente su suma será nula, de forma que la respuesta del filtro en frecuencias espaciales cero sea nula. Un ejemplo de filtro paso alto es la máscara siguiente:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & 1 \end{bmatrix}$$

Figura 7: Máscara de filtro paso alto de 3 X 3



Esta máscara corresponde, concretamente, a la aproximación discreta de la laplaciana de una función escalar. Es por tanto un filtro de segunda derivada.

Los filtros paso alto atenúan las componentes de baja frecuencia y dejan intactas las de medias-altas en función de la frecuencia de corte que se elija. Se usan para quedar con las propiedades de la imagen en los que los niveles de gris varían bruscamente, por bordes de la imagen

2.3.2. Algoritmos de Segmentación:

Estos algoritmos se pueden dividir en una de estas dos propiedades según los valores del nivel de gris: discontinuidad o similitud entre los niveles de gris de los pixeles vecinos.

Discontinuidad. Divide la imagen basándose en los cambios violentos de niveles de gris: detección de puntos aislados, detección de líneas y detección de bordes.

Similitud. Divide la imagen basándose en la búsqueda de zonas que tengan valores equivalentes: crecimiento de región y umbralización. En el crecimiento por regiones, es un procedimiento que junta los pixeles o subregiones de la imagen en regiones más grandes basándose en un criterio predefinido. Umbralización es un método para distinguir un objeto del fondo de la imagen con una simple binarización.

Entre estas dos propiedades, elegimos por similitud debido a que las imágenes a procesar tienen valores similares, y en esta elegimos umbralización debido a que las imágenes de los aisladores casi siempre tendrán un fondo claro

debido a que son tomadas en altura y el fondo será el cielo. Entre los algoritmos que ejecutamos en las pruebas con la librería Balu, tenemos los siguientes:

2.3.2.1. Algoritmo Balu

Este algoritmo se encuentra en la librería Balu hecha para el programa Matlab. Este algoritmo recibe de entrada una imagen a color (RGB). Elige un valor por defecto de threshold de -0.05 . El rango de este valor es entre 1 y -1 . Si este valor es positivo, es usado para dilatar la segmentación. De lo contrario, si es negativo, la segmentación se erosionaría. También, determina una imagen binaria del objeto, determina una imagen binaria del borde del objeto y por último eleva el contraste de la imagen binaria resultante.

2.3.2.2. Algoritmo Otsu

La umbralización es una técnica de segmentación ampliamente utilizada en las aplicaciones industriales. Se emplea cuando hay una clara diferencia entre los objetos a extraer respecto del fondo de la escena. Los principios que rigen son la similitud entre los píxeles pertenecientes a un objeto y sus diferencias respecto al resto. Por tanto, la escena debe caracterizarse por un fondo uniforme y por objetos parecidos.

Al aplicar un umbral, T , la imagen en escala de grises, $f(x,y)$, quedará binarizada; etiquetando con '1' los píxeles correspondientes al objeto y con '0' aquellos que son del fondo. Por ejemplo, si los objetos son claros respecto del fondo, se aplicará:



$$g(x, y) = \begin{cases} 1 \leftrightarrow f(x, y) > T \\ 0 \leftrightarrow f(x, y) \leq T \end{cases}$$

En el caso de que los objetos sean oscuros respecto del fondo, la asignación sería a la inversa:

$$g(x, y) = \begin{cases} 1 \leftrightarrow f(x, y) < T \\ 0 \leftrightarrow f(x, y) \geq T \end{cases}$$

El umbral puede depender de $f(x, y)$, de alguna propiedad local del píxel, $p(x, y)$, y hasta de su propia posición:

$$T = T(f(x, y), p(x, y), x, y)$$

Si el umbral sólo depende de $f(x, y)$ se dice que es un umbral global; en el caso de que además dependa de $p(x, y)$, por ejemplo, el valor medio de los píxeles vecinos, el umbral es denominado local; y si depende también de la posición (x, y) del píxel, se denominará dinámico.

La mayoría de las técnicas de umbralización se basan en estadísticas sobre el histograma unidimensional (el que vimos hasta ahora) de una imagen. También se utiliza la matriz de co-ocurrencia de una imagen. Para localizar los umbrales se pueden usar procedimientos paramétricos y no paramétricos. En los paramétricos, la distribución de los niveles de gris de una clase de objeto lleva a encontrar los umbrales. En los procedimientos no paramétricos, los umbrales se obtienen de forma óptima de acuerdo a algún criterio. En particular, el método de Otsu, que es el objetivo de este apunte, selecciona el umbral óptimo, maximiza la varianza entre clases (between-class variance) mediante una búsqueda de forma completa.

Si bien hay distintos métodos para encontrar un umbral, muchos de ellos no nos entregan resultados óptimos cuando se procesan imágenes del mundo



real debido a la existencia de ruido, histogramas planos o una iluminación inapropiada. Por otro lado, el método de Otsu fue uno de los métodos más destacados de elección de umbral para imágenes del mundo real. No obstante, como dijimos, este método usa una búsqueda rigurosa para calcular el criterio para incrementar la varianza entre las clases. A medida que el número de clases de una imagen se intensifica, el algoritmo de Otsu precisa más tiempo que lo normal para elegir un umbral multinivel apropiado. Para definir el umbral de una imagen eficazmente, vamos a proponer una varianza entre clases modificada para el método de Otsu. Esta modificación del método disminuirá considerablemente el tiempo de cálculo.

La importancia del algoritmo de Otsu reside en que se realiza de forma automática, es decir, no necesita una verificación humana, ni una data anterior de la imagen antes de su debido proceso.

Finalmente decidimos usar el algoritmo de Otsu debido que nos dio mejores resultados en las pruebas con la librería Balu en Matlab. Ver capítulo 5

2.3.3. Algoritmos de Extracción de características:

Debido a que el presente trabajo de investigación, se apoyara en extraer características basadas en la textura, a su vez, se detectó que los patrones principales en los aisladores eléctricos son textura lisa y textura rugosa (Tabla 12 Identificar patrones en aisladores), dicha característica se repetía en todos los aisladores eléctricos. Existen investigaciones similares (Tabla 13 Selección de algoritmos, se muestra algoritmos de extracción basado en textura) a esta, que se basaron también en la textura; es por eso que se pasó a documentar



los siguientes algoritmos para tener una base teórica fundamentada.

La textura es otro descriptor importante y muy utilizado en el reconocimiento, nos describe que tan homogénea será la imagen. Los descriptores de textura se basan siempre en una vecindad, ya que la textura se define para regiones y no para píxeles individuales. Es difícil encontrar un solo descriptor de la textura, ya que existen varios problemas asociados a ellos: el detector perfecto debería ser insensible a rotaciones y a escalamientos.

2.3.3.1. Filtros de Gabor

Los filtros de Gabor mostraron poseer propiedades de localización óptima en el dominio espacial, siendo esto algo deseable en los problemas de segmentación de imágenes texturadas.

Sea $I(x; y)$ la imagen de entrada, la cual está formada por N texturas t_1, t_2, \dots, t_N , con $N > 2$. Cuando decimos que la imagen está formada por varias texturas, nos referimos a que la imagen de entrada posee un mapa de texturas subyacente, el que se podría definir como mapa $I(x, y) = i$, si en la posición $(x; y)$ de la imagen hay un píxel proveniente de la textura t_i .

A la imagen de entrada se le aplican k filtros, donde cada uno de ellos se compone de un pre filtro de Gabor $h_j(x, y)$, un operador de magnitud $|-|$, y un pos filtro Gaussiano $g_{pj}(x; y)$.

Típicamente, el número de filtros (k) es menor al de texturas (N). El pre filtro de Gabor j tiene como impulso de salida $h_j(x; y)$ (3.1). Luego de esto, se utiliza un clasificador para obtener la segmentación definitiva.



Las funciones de Gabor son una completa conjunto de bases dada por:

$$f(x, y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp\left[-\frac{1}{2}\left[\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right]\right] \exp(2\pi j u_0 x) \quad (3.1)$$

Un banco de filtros autónomos similar puede obtenerse a través de la dilatación apropiada y la rotación de $f(x, y)$ a través de la generación de la función.

$$f_{pq}(x, y) = \alpha^{-p} f(x', y')$$

Donde:

$$\begin{aligned} X' &= \alpha^{-p} f(x \cos\theta_q + y \sin\theta_q) \\ &= \alpha^{-p} f(-x \sin\theta_q + y \cos\theta_q) \\ &= \alpha > 1; p = 1, 2, \dots, S; q = 1, 2, \dots, L. \end{aligned}$$

Hay subíndices enteros p y q represente el índice de escala (dilatación) y orientación (rotación), respectivamente. S es el número total de escalas y L es el número total de orientaciones en el banco de filtros de Gabor auto-similar. Para cada orientación q , el ángulo θ_q está dada por.

$$\theta_q = \frac{\pi(q-1)}{L}, q = 1, 2, \dots, L.$$



2.3.3.2. Patrones Locales Binarios (LBP)

LBP es un operador de textura simple pero muy eficiente que etiqueta los píxeles de una imagen por vecindad de umbral de cada pixel con el valor del píxel central, y considera el resultado como un número binario. Debido a su poder de discriminación y la simplicidad de cálculo, este operador de textura se ha convertido en un método popular que se usa en varios tipos de aplicaciones. Puede ser visto como un enfoque unificador de los modelos tradicionalmente divergentes del análisis de texturas: los estadísticos versus los estructurales. Quizá la propiedad más importante del operador, para aplicaciones del mundo real, es su robustez frente a cambios, en una escala de grises monótona, causados, por ejemplo, por las variaciones de iluminación y frente a rotaciones, en el caso de utilizar códigos circulares. Otra característica importante es su simplicidad computacional, lo que nos permite analizar las imágenes en tiempo real.

LBP proporciona un operador de análisis de textura que se define como una medida de la textura en una escala de grises invariante, derivado de una definición general de textura mediante vecinos locales.

La forma actual del operador LBP es muy diferente de su versión básica: la definición original se extiende a un conjunto de vecinos arbitrarios circulares, y se han desarrollado nuevas versiones del mismo, sin embargo, la idea principal es la misma: un código binario



que describe el patrón de la textura local que es construido por el umbral de un conjunto de vecinos por el valor de gris de su centro

Definimos T , Textura, como la distribución conjunta de los niveles de gris de $P + 1 (P > 0)$ pixeles de una imagen. $T = t (g_c, g_0, g_1, \dots, g_{p-1})$

Donde g_c se corresponde con el nivel de gris del pixel central del muestreo de una matriz $n \times n$ y $g_p (p = 1, 2, \dots, p - 1)$, corresponden a los valores de gris de los pixeles P equidistantes en un círculo de radio $R (R > 0)$ que forman un conjunto circular y simétrico de los vecinos con respecto al centro.

$T = t (g_c, g_0 - g_c, g_1 - g_c, \dots, g_{p-1} - g_c)$, asumiendo que la diferencia es independiente la distribución puede ser factorizada:

$T \approx t (g_c) t (g_0 - g_c, g_1 - g_c, \dots, g_{p-1} - g_c)$, como $t (g_c)$, describe la luminosidad general de una imagen, que no está relacionada a la textura de la imagen local, puede ser ignorada y quedando: $T \approx t (g_0 - g_c, g_1 - g_c, \dots, g_{p-1} - g_c)$.

Aunque invariante frente a cambios de escala de grises, las diferencias se ven afectados por el cambio de escala. Para lograr invariancia con respecto a cualquier transformación monótona de la escala de grises, solo es necesario considerar los signos de las diferencias:

$T \approx t (s (g_0 - g_c), s (g_1 - g_c), \dots, s (g_{p-1} - g_c))$, donde:

$$S(x) = 1, x \geq 0; 0, x < 0$$

Ahora, un peso 2^p es asignado a cada signo $s (g_p - g_c)$ transformando la diferencia en el muestreo en un único código LBP:



$$LBP_{P,R} = \sum_{p=0}^{p=1} s(g_p - g_c)2^p$$

2.3.4. Algoritmos de Clasificación:

Todos los algoritmos descritos acá son los encontrados en nuestros antecedentes. Finalmente usamos redes neuronales, KNN y SVM por mostrar mejores resultados en nuestros antecedentes.

2.3.4.1. Redes neuronales:

Una red neuronal está compuesta por varios nodos conectados entre sí, su forma de conectarse y la forma en la que trata la información de entrada, son las características que distinguen a las redes existentes.

La neurona tiene como función, dar una salida a las entradas que tenga. Cada neurona tiene n conexiones de entrada y solo una salida, a cada valor de entrada E_i , que proviene de una neurona distinta, se le multiplica por un peso P_i para poder ponderar la importancia de los valores provenientes de las diversas neuronas. En función del valor resultante de la sumatoria de todas las entradas, incluyendo el valor de la función $Q()$, único de cada neurona, esta dará un valor de salida u otro.



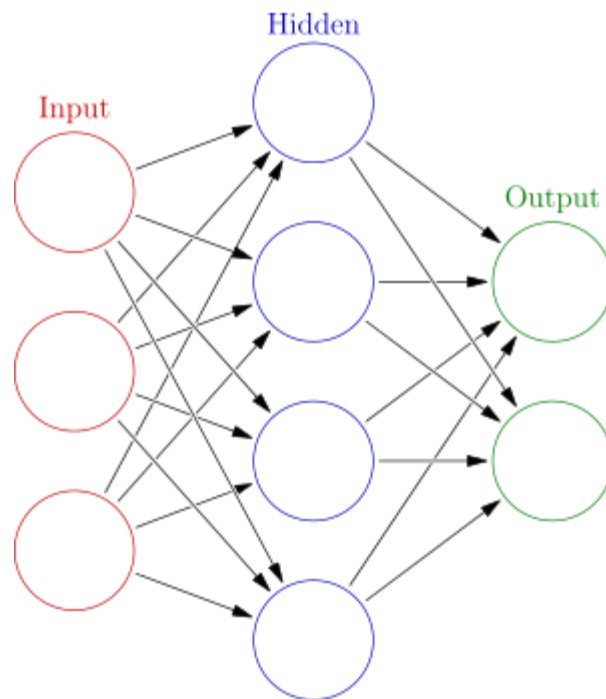


Figura 8: Una red neuronal artificial es un grupo interconectado de nodos similar a la vasta red de neuronas en un cerebro biológico. Cada nodo circular representa una neurona artificial y cada flecha representa una conexión desde la salida de una neurona a la entrada de otra.

2.3.4.2. Redes Bayesianas

Pajares, De la Cruz (2002) definen que la teoría de la decisión de Bayes es un método estadístico clásico en clasificación de patrones. Se basa en el supuesto de que el problema de la decisión se enfoca en términos probabilísticos y que todas las probabilidades relevantes resultan conocidas.

Antes de proceder a la generalización del desarrollo de este método, vamos a ver un ejemplo. Consideremos el problema de diseñar un clasificador para separar dos clases de frutas en una cinta transportadora, melocotones M o albaricoques A . Alguien que



observe la cinta tiene muy difícil predecir qué tipo de fruta aparecerá en la cinta, por lo que la secuencia de tipos de frutas se presenta aleatoria. Usando la terminología de la teoría de la decisión, diremos que según aparece cada tipo de fruta, pertenecerá a uno de los dos estados: o es un melocotón o es un albaricoque. Sea y el estado que designa la pertenencia a uno de los dos tipos de fruta, con $y = M$ e $y = A$ para melocotón y albaricoque respectivamente. Puesto que la pertenencia a uno de los dos estados es impredecible, consideramos y una variable aleatoria.

Si la cosecha de melocotones ha sido más o menos igual que la de albaricoques, diremos que la siguiente fruta que va a aparecer tiene la misma probabilidad de ser melocotón o albaricoque. De forma más directa, suponemos que existe alguna *probabilidad a priori* $P(y=M)$ de que la siguiente fruta sea un melocotón y alguna *probabilidad a priori* de que sea un albaricoque $P(y=A)$. Esas probabilidades a priori reflejan nuestro conocimiento a priori sobre cuál es la probabilidad de ver aparecer un melocotón o un albaricoque antes de que aparezca la próxima pieza de fruta. Ambas probabilidades $P(y=M)$ y $P(y=A)$ son no negativas y su suma es la unidad.

Supongamos por un instante que estamos forzados a tomar una decisión sobre el tipo de fruta que aparecerá a continuación. La única información son esas probabilidades a priori, por tanto si tenemos que tomar una decisión, parece razonable usar la siguiente



regla de decisión, decidir M si $P(y=M) > P(y=A)$, en caso contrario decidir A.

No obstante, la mayoría de las veces no se toma una decisión con esa poca información. En el ejemplo podríamos usar la componente de color rojo x como una información más. Así, diferentes muestras de frutas darán diferentes valores de x , resulta natural expresar esta variabilidad en términos probabilísticos, en ese sentido, consideremos a x una variable aleatoria continua cuya distribución depende del tipo de fruta. Sean $p(x / y=A)$ y $p(x / y=M)$ las funciones de *densidad de probabilidad condicionales* para x dado que el tipo de fruta sea A o M. supongamos que conocemos tanto las probabilidades a priori como estas últimas funciones de densidad de probabilidad. Suponer además que medimos la componente de color rojo de un albaricoque y descubrimos que vale x , la pregunta será ¿cómo influye esta medida sobre nuestra actitud con relación al tipo de fruta de que se trata? La respuesta nos la proporciona la *regla de Bayes*:

$$p\left(y = \frac{A}{x}\right) = \frac{p(x/y = A)P(y = A)}{p(x)} \quad \text{Ecuación (14)}$$

$$p\left(y = \frac{M}{x}\right) = \frac{p(x/y = M)P(y = M)}{p(x)} \quad \text{Ecuación (15)}$$



Donde

$$p(x) = p(x/y = A)P(y = A) + p(x/y = M)P(y = M)$$

Ecuación (16)

La regla de Bayes muestra como la observación del valor x cambia las probabilidades a priori a las *probabilidades a posteriori* $p(y = A / x)$ y $p(y = M / x)$.

La ecuación (45) es una constante de normalización, que no necesita ser obtenida a la hora de tomar una decisión, puesto que la regla de decisión es una comparación de las magnitudes relativas de las probabilidades a posteriori. Una vez que se determinan esas probabilidades a posteriori, la siguiente regla (función discriminante) se utiliza para clasificar x.

$$fd(x) = \begin{cases} A & \text{si } p(x/y = A)P(y = A) > p(x/y = M)P(y = M) \\ M & \text{de otro modo} \end{cases}$$

Ecuación (17)

Si ahora ampliamos el número de tipos de frutas hasta J y el número de atributos dados por el vector de atributos $x = \{x_1, x_2, \dots, x_d\}$ estamos ante un problema de clasificación general. Usando el teorema de Bayes y considerando que tanto las probabilidades a priori $P(y = c_j)$ como las densidades condicionales para cada clase $p(x / y = c_j)$ son conocidas o



se pueden de que esa observación pertenezca a una determinada clase. Estas probabilidades, llamadas probabilidades a posteriori pueden usarse para construir una regla discriminante

$$p\left(y = \frac{c_j}{x}\right) = \frac{p(x/y = c_j)P(y = c_j)}{p(x)} \quad \text{Ecuación (18)}$$

Donde

$$p(x) = \sum_{j=1}^c p(x/y = c_j)P(y = c_j) \quad \text{Ecuación (19)}$$

Para una mejor comprensión vamos a redefinir la ecuación

$$L(y, f(x, w)) = \begin{cases} 0 & \text{si } y = f(x, w) \\ 1 & \text{si } y \neq f(x, w) \end{cases} \text{ como}$$

$$L(y_i, f_j(x, w)) = \begin{cases} 0 & \text{si } y_i = f_j(x, w) \\ 1 & \text{si } y_i \neq f_j(x, w) \end{cases} \quad \text{Ecuación (20)}$$

Con $i, j = 1, 2, \dots, c$. A partir de $R(w) = \int L(f(x, w))p(x)dx$, el riesgo condicional esperado correspondiente a esta función de perdidas resulta ser

$$R(y_i/x) = \sum_{j=i}^c L(y_i, f_j(x, w))P(y_j/x)$$

$$= \sum_{j \neq i} P(y_j/x) = 1 - P(y_i/x) \quad \text{Ecuación (21)}$$



Y $P(y_j/x)$ es la probabilidad condicional de que la clasificación sea correcta. A partir de (50) minimizar el riesgo condicional significa maximizar $P(y_j/x)$, por lo que la regla de decisión considerando el error mínimo resulta

$$x \in c_i \text{ si } P(y = c_i/x) > P(y = c_j/x) \quad \forall i \neq j, \quad i, j = 1, 2, \dots, c \tag{22}$$

Ecuación (22)

Fijándose en el Segundo término de la expresión (47) del teorema de Bayes y habiendo eliminado el término no discriminante $p(x)$, se tiene una forma alternativa de clasificar un vector de atributos x :

$$x \in c_i \text{ si } P(x/y = c_i)P(y = c_i) > P(x/y = c_j)P(y = c_j) \quad \forall i \neq j, \quad i, j = 1, 2, \dots, c \tag{23}$$

Ecuación (23)

Generalmente las distribuciones de densidad de probabilidad se eligen Normales o Gaussianas para la mayoría de los casos prácticos en reconocimiento de patrones (ver apéndice I). un caso especial surge cuando las probabilidades a priori son iguales para todas las clases, ya que en esta situación la distancia de Mahalanobis se puede utilizar como función discriminante mediante la siguiente regla de decisión a partir de (52) y teniendo en cuenta el signo negativo en el término exponencial de la función de densidad de probabilidad Normal, así

$$x \in c_i \text{ si } d_M^2(x, m_i) < d_M^2(x, m_j) \quad \forall i \neq j, \quad i, j = 1, 2, \dots, c \tag{24}$$

Ecuación (24)



Donde m_i , m_j son los vectores media de las clases c_i y c_j respectivamente.

En el supuesto de que las matrices de covarianza sean la identidad, la distancia de Mahalanobis al cuadrado resulta ser la distancia Euclidea al cuadrado, en cuyo caso tendríamos,

$$d_E^2(x, m_i) = (x - m_i)^t(x - m_i) = x^t x + 2x^t m_i + m_i^t m_i \quad \text{Ecuación (25)}$$

En la expresión anterior el termino $x^t x$ no discrimina, ya que se repite en todas las clases, de forma que puede despreciarse. Ahora, si se cambia de signo y se divide por 2 en la ecuación (54) se obtiene la siguiente función discriminante,

$$f d_i(x) = x^t m_i - \frac{1}{2} m_i^t m_i \quad \text{Ecuación (26)}$$

Como el resultado de (54) es una cantidad positive, al haber eliminado el termino $x^t x$ y cambiado de signo los restantes, se deduce que la distancia Euclidea al cuadrado mínima hace la expresión (55) máxima. Una aplicación de esta distancia puede encontrarse en Chim y col. (1999) para reconocimiento óptico de caracteres mediante momentos invariantes.

A veces puede resultar de interés conocer el grado de separabilidad entre dos clases c_i y c_j . Esto se puede medir utilizando la media y la



matriz de covarianza de las clases mediante lo que se conoce como *divergencia* (Jensen, 1982).

$$Diverg_{ij} = 0.5Tr[(C_i - C_j)(C_j^{-1} - C_i^{-1})] \quad \text{Ecuación (27)}$$

$$+0.5Tr[(C_i^{-1} - C_j^{-1})(m_i - m_j)(m_i - m_j)']$$

Donde $Tr[.]$ Es la traza de una matriz (suma de los elementos de la diagonal principal), C_i y C_j son las matrices de covarianza de las clases c_i y c_j y m_i y m_j son los vectores media de las clases c_i y c_j , finalmente el superíndice t indica vector transpuesto.

2.3.4.3. Clustering

Pajares, De la Cruz (2002) definen que el problema del agrupamiento es el de separar un conjunto de datos en un cierto número de grupos (“clusters” en inglés) basándose en alguna medida de similitud. El objetivo es encontrar un conjunto de grupos para los cuales, las muestras entro de un grupo son más similares entre sí que con las muestras de diferentes grupos. Generalmente, también se genera un prototipo local que caracteriza y representa a los miembros de un determinado grupo. Las estructuras de los datos son inferidas luego analizando los grupos resultantes y/o sus prototipos. Obsérvese que la tarea del agrupamiento puede estar fuera del aprendizaje predictivo, puesto que el objetivo es agrupar los datos principalmente



y no caracterizar los datos futuros. No obstante, muchos de los métodos usados para cuantización vectorial, que si es un método predictivo, se usan en agrupamientos.

Lagunas variantes del algoritmo AGL se utilizan a menudo para agrupamientos bajo el nombre de *k-means* o *c-means* que veremos más tarde, donde *k* o *c* indica el número de grupos. El análisis de grupos difiere de CV en que la medida de similitud para el agrupamiento se elige subjetivamente basada en su capacidad para crear grupos de interés.

Los algoritmos de agrupamiento se pueden dividir en las siguientes categorías principales:

- 1) *Secuenciales*. Estos algoritmos producen un agrupamiento simple. En muchos de ellos los vectores de características se presentan al algoritmo una vez o unas pocas veces. El resultado final depende generalmente del orden en el que se presentan los vectores al algoritmo.
- 2) *Jerárquicos*. Que a su vez pueden ser de *aglomeración* o de *división*. Los de *aglomeración* producen una secuencia de agrupamientos en un orden decreciente de grupos *m* en cada paso previo. Por el contrario, los de *división* producen un número de grupos creciente *m* en cada paso como consecuencia de la división de un simple grupo en dos.
- 3) *Basados en la optimización de una función de coste*. En estos



algoritmos se evalúa el agrupamiento basándose en una función de coste J que es optimizada. Generalmente el número de grupos se mantiene constante. En esta categoría se incluyen lo que nosotros denominamos algoritmos puros, donde un vector dado pertenece exclusivamente a un determinado grupo. El algoritmo más famoso de esta categoría es el Isodata. En contraposición los anteriores están los algoritmos basados en la lógica *difusa* donde un vector pertenece a más de un grupo, con un determinado grado de pertenencia. También podemos destacar los algoritmos probabilistas, que siguen el criterio de clasificación Bayesiano, de forma que cada vector x se asigna al grupo cuya probabilidad a posteriori sea máxima.

Existen otras categorías de algoritmos de agrupamiento tales como *algoritmos genéticos*, *métodos de relajación estocástica*, algoritmos basados en *transformaciones morfológicas*, basados en *grafos*, etc. Así como abundantes variantes de los mencionados anteriormente. Todos ellos se pueden encontrar en Theodoridis y Koutroumbas (1999).

A continuación, vamos a estudiar algunos de los más usados en reconocimiento de patrones dentro de las diferentes categorías expuestas anteriormente. Previamente, estudiaremos las definiciones básicas relativas a los agrupamientos.



Sea $X = \{x_1, x_2, \dots, x_n\}$ un conjunto de muestras o datos. Definimos un agrupamiento R con m grupos de X , como la partición de X en m conjuntos (grupos) R_1, R_2, \dots, R_m cumpliendo las tres condiciones siguientes:

- a) $R_i \neq \emptyset, i = 1, \dots, m$
- b) $\cup_{i=1}^m R_i = X$
- c) $R_i \cap R_j = \emptyset, i \neq j, i, j = 1, \dots, m$

Los vectores contenidos en un grupo R_i son similares entre si y menos similares a los vectores de otro grupo. El concepto de similitud depende de la medida usada (ver apéndice J). Bajo la definición precedente, cada vector pertenece a un único grupo. Una definición alternativa es la proporcionada en términos de los *conjuntos difusos*. Un agrupamiento difuso de X en m grupos se caracteriza por m funciones u_j donde

$$u_j: X \rightarrow [0,1], \quad j = 1, \dots, m \tag{Ecuación (28)}$$

Y

$$\sum_{j=1}^m u_j(x) = 1, \quad i = 1, 2, \dots, n \quad 0 < \sum_{j=1}^n u_j(x_i) < n, \quad j = 1, 2, \dots, m \tag{Ecuación (29)}$$



Las u_j son las denominadas *funciones miembro*. El significado de estas funciones miembro es que cada vector x pertenece a más de un grupo simultáneamente hasta un cierto grado, que se cuantifica por el correspondiente valor u_j en el intervalo $[0,1]$. Los valores próximos a la unidad muestran un alto grado de pertenencia en el grupo correspondiente, mientras que los próximos a cero muestran un bajo grado de pertenencia. Los valores de esas funciones de pertenencia son indicativos de la estructura del conjunto de datos, en el sentido de que, si una función miembro tiene valores próximos a uno para dos vectores de X , esto es x_k, x_n , entonces estos vectores son considerados similares entre sí.

La condición de la derecha en (57) garantiza que no hay casos triviales en el sentido de que existan grupos que no compartan ningún vector. Esto es análogo a la condición $R_i = \emptyset$ en la definición anterior.

La definición de agrupamiento en m conjuntos distintos R_i , dada anteriormente, puede verse como un caso especial del agrupamiento difuso si definimos las funciones miembro difusas u_j tomando valores en $\{0,1\}$, esto es, para ser o bien 1 o bien 0. En este sentido, cada vector de datos pertenece exclusivamente a un grupo y las funciones miembro se llaman ahora *funciones características* (Klir 1995).



2.3.4.4. KNN

En este algoritmo la clasificación de nuevos ejemplos se realiza buscando el conjunto de los k ejemplos más cercanos de entre un conjunto de ejemplos etiquetados previamente guardados y seleccionando la clase más frecuente de entre sus etiquetas. La generalización se pospone hasta el momento de la clasificación de nuevos ejemplos (por esta razón es por la que en ocasiones el llamado aprendizaje perezoso, en inglés, lazy learning)

Una parte muy importante de este método es la definición de la medida de distancia (o similitud) apropiada para el problema a tratar. Esta debería tener en cuenta la importancia relativa de cada atributo y ser eficiente computacionalmente. El tipo de combinación para escoger el resultado de entre los k ejemplos más cercanos y el valor de la propia k también son cuestiones a decidir de entre alternativas.

Este algoritmo, en su forma más simple, guarda en memoria todos los ejemplos durante el proceso de entrenamiento y la clasificación de nuevos ejemplos se basan en las clases de los k ejemplos más cercanos (por esta razón se le conoce también como basado en memoria, en ejemplos, en instancias o en casos). Para obtener el conjunto de los k vecinos más cercanos, se calcula la



distancia entre el ejemplo a clasificar $x = (x_1, \dots, x_m)$ y todos los ejemplos guardados

$x_i = (x_{i1}, \dots, x_{im})$. Una de las distancias más utilizadas es la euclídea.

$$de(x, x_i) = \sqrt{\sum_{j=1}^m (x_j - x_{ij})^2} \quad \text{Ecuación (30)}$$

2.3.4.5. Árboles de Decisión

Un árbol de decisión es una forma de representar reglas de clasificación inherentes a los datos, con una estructura en árbol n-ario que particiona los datos de manera recursiva. Cada rama de un árbol de decisión representa una regla que decide entre una conjunción de valores de un atributo básico (nodos internos) o realiza una predicción de la clase (nodos terminales).

Este algoritmo está pensado para trabajar con atributos nominales. El conjunto de entrenamiento queda definido por $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$ donde cada componente x corresponde a $x_i = (x_{i1}, \dots, x_{im})$. Donde m corresponde al número de atributos de los ejemplos de entrenamiento; y el conjunto de atributos por $A = (a_1, \dots, a_m)$. Donde $dom(a)$ corresponde al conjunto de todos los posibles valores del atributo a_j , y para cualquier valor de un ejemplo de entrenamiento $x_{ij} \in dom(a_j)$



El proceso de construcción de un árbol es un proceso iterativo, en el que, en cada iteración, se selecciona el atributo que mejor particiona el conjunto de entrenamiento. Para realizar este proceso, tenemos que mirar la bondad de las particiones que genera cada uno de los atributos y, en segundo paso, seleccionar el mejor. La partición del atributo a genera $| \text{dom}(a) |$ conjuntos, que corresponde al número de elementos del conjunto. Existen diversas medidas para mirar la bondad de la partición. Una básica consiste en asignar a cada conjunto de la partición la clase mayoritaria del mismo y contar cuantos quedan bien clasificados y dividirlo por el número de ejemplos. Una vez calculadas las bondades de todos los atributos, escogemos el mejor.

Cada conjunto de la mejor partición pasara a ser un nuevo nodo del árbol. A este nodo se llegará a través de una regla del tipo atributo = valor. Si todos los ejemplos del conjunto han quedado bien clasificados, lo convertimos en nodo terminal con la clase de los ejemplos. En caso contrario, lo convertimos en nodo interno y aplicamos una nueva iteración al conjunto (“reducido”) eliminando el atributo que ha generado la partición. En caso de no quedar atributos, lo convertiríamos en nodo terminal asignando la clase mayoritaria.

Para realizar el test, exploramos el árbol en función de los valores de los atributos del ejemplo de test y las reglas del árbol hasta



llegar al nodo terminal, y damos como predicción la clase del nodo terminal al que llegamos.

Este algoritmo tiene la gran ventaja de la facilidad de interpretación del modelo de aprendizaje. Un árbol de decisión nos da la información clara de toma de decisiones y de la importancia de los diferentes atributos involucrados. Por esta razón se ha utilizado mucho en diversos campos, como el financiero.

2.3.4.6. Random Forest

Es una combinación de árboles predictores tal que cada árbol depende de los valores de un vector aleatorio probado independientemente y con la misma distribución para cada uno de estos.

1. Aleatoriamente se crea (seleccionando con reemplazado) el conjunto de entrenamiento de igual tamaño que el conjunto original. Al seleccionarse aleatoriamente con reemplazo no todos los datos de conjunto general estarán en el conjunto de entrenamiento. La probabilidad de que un dato particular esté en el conjunto de entrenamiento es aproximada a 66 %.

2. Los datos que no forman parte del conjunto de entrenamiento forman el conjunto de validación o out of bag data (OOB data)



3. En cada punto de división del árbol o nodo, la búsqueda de la mejor variable para dividir los datos no se realiza sobre todas las variables sino sobre un subconjunto, m , de las mismas. La elección del subconjunto de variables se realiza de forma aleatoria.

4. Se busca la mejor división de los datos de entrenamiento teniendo en cuenta solo al m variables seleccionadas. Para esta tarea se debe implementar una función objetivo. Habitualmente ésta es la entropía o el índice de Gini.

5. Los anteriores procesos son repetidos varias veces, de forma que se tienen un conjunto de árboles de decisión entrenados sobre diferentes conjuntos de datos y de atributos.

6. Una vez el algoritmo entrenado, la evaluación de cada nueva entrada es realizado con el conjunto de árboles. La categoría final de la clase (clasificación) es realizada por el voto mayoritario del conjunto de árboles, y en caso de regresión por el valor promedio de los resultados.

Los datos OOB se usan para determinar la impureza en los nodos terminales. La suma de estas impurezas determina la impureza del árbol. Cada registro del conjunto global de datos estará in bag para algunos árboles del random forest, y out of the bag para otros árboles. Es probable que cualquier par de registros no compartan un patrón idéntico sobre en qué árboles están in bag y en cuales están out of the bag.



Para medir el error de random forest se suele utilizar la técnica denominada out-the-bag error. Para cada árbol se utiliza el conjunto de objetos no seleccionados por su muestra bootstrap de entrenamiento para ser clasificados con dicho árbol. Promediando sobre el conjunto de árboles de random forest se puede estimar el error del algoritmo.

Por otro lado, random forest puede ser paralelizado eficazmente puesto que cada árbol puede construirse de manera independiente a los otros árboles. Esta característica lo hace deseable para entornos paralelos.

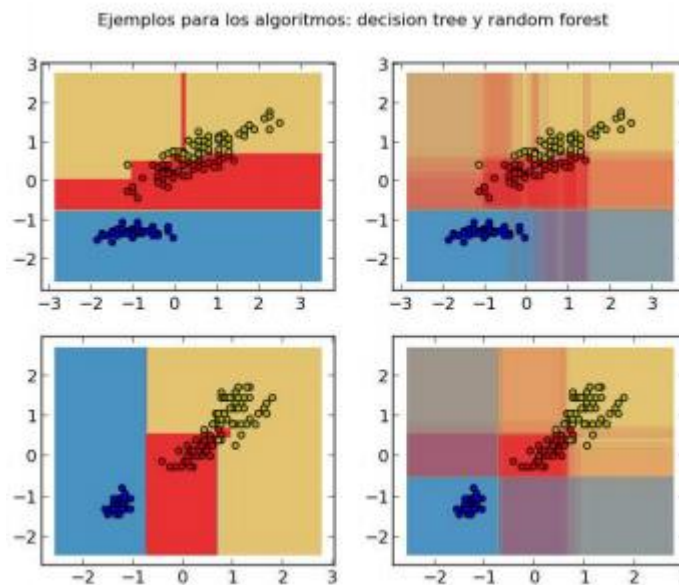


Figura 9: Fuente: <http://www.wae.ciemat.es>

El algoritmo random forest puede ser utilizado para medir la proximidad de los datos. Esto significa que el algoritmo puede ser utilizado para hacer aprendizaje no supervisado.



Dados dos elementos (i, j) , la proximidad entre ambos puede ser medida como la fracción de árboles en los cuales los elementos i y j están en el mismo nodo terminal. Si se hace esta operación para todos los pares de elementos, se formará una matriz de proximidad entre los elementos. Esta matriz de proximidad puede ser utilizada tanto para aprendizaje no supervisado como para explorar la estructura de los datos.

2.3.4.7. Máquinas de soporte vectorial (SVM)

Las máquinas de vectores de soporte (SVM por sus siglas en inglés Support Vector Machines) (Vapnik, 1995) son una de las técnicas de clasificación controladas más robustas. Se trata de un algoritmo clasificador biclase (esto le permite clasificar en dos clases), aunque la mayoría de implementaciones del mismo, (como, LIBSVM, LIBLINEAR) soportan clasificación multi-clase, es decir de más de 2 capas. Al igual que cualquier clasificador controlado biclase, el objetivo de un algoritmo SVM es producir un límite de decisión (lineal o no) en el espacio de características de manera que las observaciones siguientes se clasifiquen de forma automática en uno de los dos grupos definidos por dicho límite (también llamada como hiperplano). La singularidad de SVM es que intenta generar dicho hiperplano de manera que amplíe su separación con cada uno de los



grupos, figura 1. Por ello se dice que este algoritmo clasificador es uno de margen máximo.

Si nosotros consideramos elementos en un espacio D – dimensional, i.e. $^{\circ} D$, el hiperplano esta señalado por un vector $\theta = (\theta_0, \theta_1, \dots, \theta_D)$, de manera que su ecuación es:

$$\theta_0 + \sum_{i=1}^D \theta_i X_i = 0 \quad (1)$$

Así, dado un elemento $X = (X_1, \dots, X_D) \in {}^{\circ} D$, un clasificador determina a una clase o a otra en función de si $\theta_0 + \sum_{i=1}^D \theta_i X_i > 0$ o $\theta_0 + \sum_{i=1}^D \theta_i X_i < 0$. Sin embargo, en el caso de SVM esta determinación se realiza en función de si $\theta_0 + \sum_{i=1}^D \theta_i X_i > \gamma$ o $\theta_0 + \sum_{i=1}^D \theta_i X_i < \gamma$ con $\gamma > 0$.



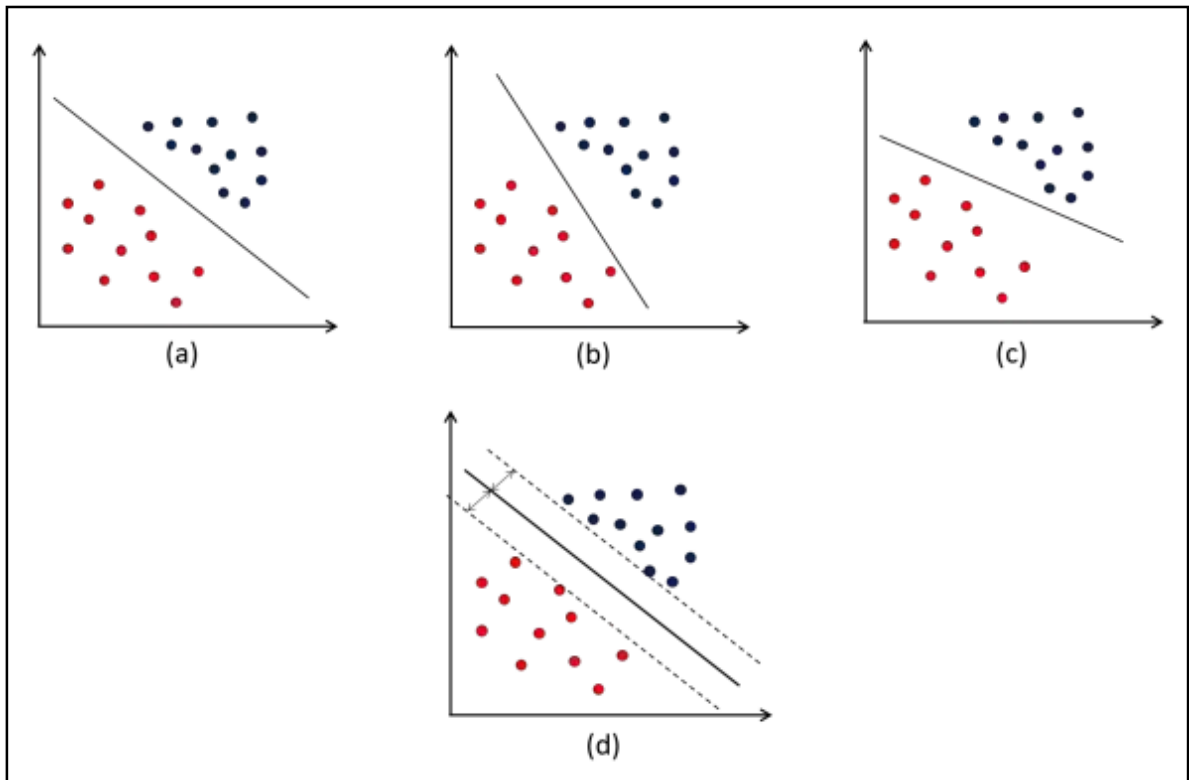


Figura 10: Ejemplos de hiperplanos que separan dos clases diferentes en un espacio bidimensional. (a) - (c) Diferentes hiperplanos. (d) Hiperplano de margen máximo

Para deducir el hiperplano durante la etapa de entrenamiento es necesario reducir la función de coste:

$$J(\theta) = C \sum_{i=1}^N [y_i \text{cost}_1(\theta^T x_i) + (1 - y_i) \text{cost}_0(\theta^T x_i)] + \frac{1}{2} \sum_{j=1}^D \theta_j^2, \quad (2)$$

Donde $\text{cost}_0(z)$ y $\text{cost}_1(z)$ son funciones de coste en función de si $y_i = 1$ o $y_i = 0$, respectivamente que tienen un aspecto parecido al mostrado en la figura 2.



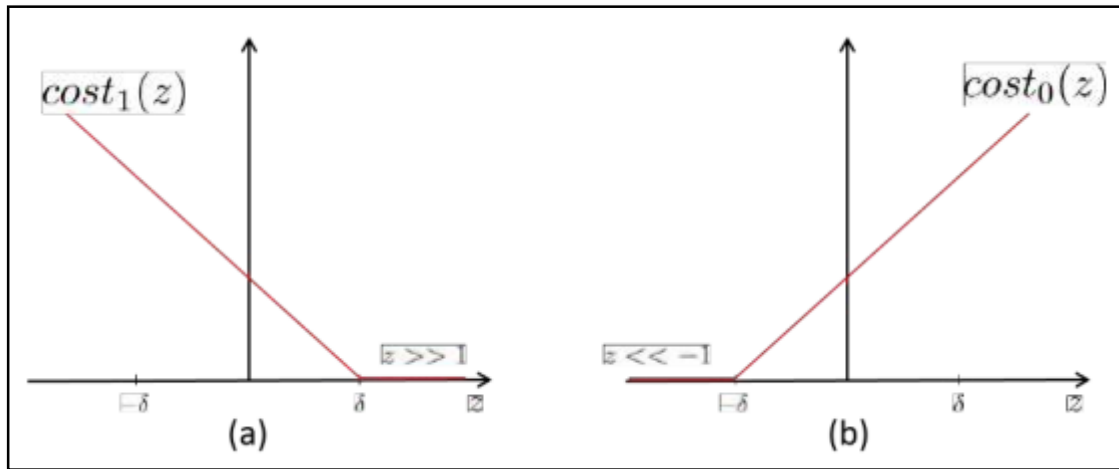


Figura 11: Ilustración de las funciones cost 0 y cost 1

Otra de las singularidades de las SVM es su capacidad para conseguir límites de decisión no lineales, figura 3 $\{x_1, x_1^2, \dots, x_D, x_1^2\}$ ampliando la dimensión del espacio de características mediante el uso de kernels. Por ejemplo, mapeando un grupo de D características $\{X_1, \dots, X_D\}$ en, por ejemplo, uno de 2D características $\{X_1, X_1^2, \dots, X_D, X_1^2\}$. Esto es muy útil cuando las clases de los elementos que se van a clasificar **no son linealmente disgregables en el espacio original**. Este mapeado se lleva a cabo con las denominadas funciones kernel o, sencillamente, kernels, $k(x, y)$, que se realizan de manera que el valor que retornan sea más diminuto cuanto más lejos esté y de x . En la práctica, tanto los elementos x como y serán los elementos del grupo de entrenamiento. En definitiva, los vectores que establecen el hiperplano se calculan como una composición lineal de imágenes del kernel k con parámetros α_i :

$$\sum_i \alpha_i k(x_i, x) = cte \quad (3)$$



Algunos de los *kernels* más usados son el lineal (ecuación 4), polinomial (ecuación 5), gaussiano, también conocido como *radial basis function* (ecuación 6) o sigmoïdal (ecuación 7)

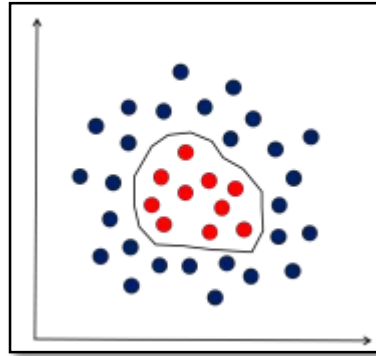


Figura 12: Ejemplo de frontera de decisión no lineal

$$k(x_i, x_j) = x_i \cdot x_j \quad (4)$$

$$k(x_i, x_j) = (x_i \cdot x_j)^d \quad (5)$$

$$k(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|), \gamma > 0^2 \quad (6)$$

$$k(x_i, x_j) = \tanh(kx_i \cdot x_j + c), k > 0, c < 0 \quad (7)$$

2.4. Definición de la terminología

2.4.1. Algoritmo:

Corona, Ancona (2011) Se denomina algoritmo al conjunto de pasos ordenados y finitos que permiten resolver un problema o tarea específica. Los algoritmos son independientes del lenguaje de programación y de la computadora que se vaya a emplear para ejecutarlo. Todo algoritmo debe ser:

1. *Finito*. en tamaño o número de instrucciones (tiene un primer y un último paso) y tiempo de ejecución (debe terminar en algún momento). Por lo tanto, debe



- tener un punto particular de inicio y fin.
2. *Preciso.* debe tener un orden entre los pasos.
 3. *Definido.* No debe ser ambiguo (dobles interpretaciones); si se ejecuta el mismo algoritmo el resultado siempre será el mismo, sin importar las entradas proporcionadas.
 4. *General.* Debe tolerar cambios que se puedan presentar en la definición del problema.

2.4.2. Impurezas:

Una impureza es una sustancia dentro de un limitado volumen de líquido, gas o sólido, que difieren de la composición química de los materiales o compuestos.

Las impurezas son, ya sea natural o añadidos, durante la síntesis de una sustancia química o producto comercial. Durante la producción, las impurezas pueden ser a propósito, accidentalmente, inevitablemente, cierto o añadido en el fondo.

2.4.3. Aislador Eléctrico:

Un aislador eléctrico es aquel dispositivo hecho de cerámica, vidrio o materiales compuestos; cuya principal función es la de aislar el cable pelado por el que pasa la corriente eléctrica de los postes (Media Tensión) y torres (Alta Tensión).



2.4.4. Imágenes digitales:

Las imágenes digitales son representadas a partir de una matriz numérica en binario, estas imágenes se encuentran formadas por miles y millones de pixeles, donde cada pixel es representado por bits, estos bits luego son interpretados por el computador para crear una imagen para su visualización o impresión. Estas imágenes pueden obtenerse de varias formas, como la conversión analógica-digital como los escáneres y las cámaras digitales, o directamente mediante programas informáticos.

CAPÍTULO III: MARCO METODOLÓGICO

3.1. Tipo y Diseño de Investigación

3.1.1. Tipo de investigación:

El tipo de investigación de la presente es tecnológica porque el software a desarrollar será beneficioso para el campo del reconocimiento de imágenes digitales en ambientes no controlados.

3.1.2. Diseño de la investigación:

El tipo de la investigación es cuasi experimental ya que la propuesta de este presente trabajo de investigación realizara pruebas con las imágenes sin contar con un grupo de control establecido. Se tomarán todas las imágenes y se les procesara para comparar y analizar resultados.

3.2. Población y Muestra

3.2.1. Población:

En los siguientes trabajos de investigación de YongHua, Jin-Cong “Identificación de defectos se la superficie en madera basados en textura”, se utilizaron 300 imágenes para la etapa de prueba y se obtuvieron resultados positivos.

Shen, Chen, Chang, en su investigación “Reconocimiento de defectos óxido recubrimiento puente de acero automatizada basada en el color y la textura”, en la etapa de prueba se utilizaron 140 imágenes de óxido artificiales normales y 560 imágenes artificiales de iluminación no uniforme.



Es por ello que la población serán 250 imágenes debido a anteriores trabajos mencionados en la que se obtuvieron buenos resultados. La población serán imágenes tomadas por una cámara debido a que no se cuenta con una base de datos de aisladores eléctricos.

3.2.2. Muestra:

La muestra es de 200 imágenes, las cuales fueron seleccionadas como las más óptimas para el proceso.

3.3. Hipótesis

Utilizando técnicas de procesamiento de imágenes se podrá identificar las adherencias de impurezas en imágenes de aisladores eléctricos.

3.4. Variables

3.4.1. Técnicas de Procesamiento Digital de Imágenes.

3.4.2. Reconocimiento de impurezas.

3.5. Operacionalización:

VARIABLES	DIMENSIONES	INDICADORES	TECNICAS E INSTRUMENTOS DE RECOLECCION DE DATOS
TECNICAS DE PROCESAMIENTO DIGITAL DE IMAGENES	Matriz de confusión	Verdadero informativo (VI)	Observación
		Falso informativo (FI)	
		Verdadero No – informativo (VNI)	
		Falso No – informativo (FNI)	
RECONOCIMIENTO DE IMPUREZAS	Rendimiento	Sensibilidad	Observación
		Especificidad	
		Exactitud	
		Precisión	
		Medida F	

3.6. Métodos, técnicas e instrumentos de recolección de datos

3.6.1. Métodos de investigación

3.6.1.1. Experimentación

Usaremos este método porque nos permite manipular las variables en función que nos permite la reunión de datos, conociendo los efectos de los estímulos recibidos y creados para su apreciación.



Este método nos exige seleccionar grupos pareados de sujetos, someterlos a tratamientos distintos, controlar las variables y comprobar si las diferencias observadas son determinantes.

3.6.2. Técnicas de recolección de datos

Las técnicas empleadas en la investigación fueron:

- a) **Observación:** Utilizamos esta técnica debido a que, para poder reconocer las impurezas en los aisladores eléctricos, los tuvimos que visualizar.

3.6.3. Instrumentos de recolección de datos

La recolección de la información se realizó a través de los siguientes instrumentos:

- a) **Observación:** Este método consiste en adquirir a través del sentido de la vista información de lo que ocurre en el medio exterior en tiempo y lugar determinado.
- b) **Entrevista:** Es el dialogo entre 2 o más personas en la que el entrevistador realiza preguntas al entrevistado que son de interés propia para obtener información pertinente.
- a. **Lapicero:** Instrumento utilizado para escribir información útil para la investigación.
- b. **Papel:** Material de escritorio que sirve para registrar información importante para la investigación.
- c. **Ordenador:** Instrumento para ordenar y procesar la información relevante para la investigación.



- d. Grabador MP3: Dispositivo útil para grabar audios en las entrevistas realizadas.
- e. Memorias USB: Dispositivo de almacenamiento para guardar información sobre la investigación.

3.7. Procedimiento para la recolección de datos

Observación: Se obtuvieron las imágenes de los aisladores con la ayuda de una cámara que con la función zoom pudo acercarse lo suficiente para captar buenas fotografías. Las imágenes tomadas fueron seleccionadas para poder determinar cuales estaban óptimas, es decir que no estén borrosas, ni tomadas a contra luz, ni muy oscuras; para luego realizar el procesamiento de reconocimiento de imágenes.

Entrevista: Se elaboró un cuestionario con preguntas pertinentes con el objetivo de obtener más información sobre el problema de esta investigación. Luego, se solicitó una reunión con el entrevistado a las 4 p.m. de un día martes y se le realizaron las preguntas sobre el tema. Esta entrevista por medio de un dispositivo móvil.

3.8. Análisis Estadístico e Interpretación de los datos

Posadas, A. (2013) usa las siguientes formulas cuando se trabaja con pruebas que pueden ser valores verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos. Y con el fin de medir el rendimiento de los algoritmos, la determinación de errores y tratamiento de datos usados se usaron las siguientes formulas:



3.8.1. Sensibilidad:

Es la proporción de muestras correctamente diagnosticadas para cada tipo de deficiencia nutricional.

$$\frac{\sum \text{Verdadero Inform.}}{\sum \text{Verdadero Inform.} + \sum \text{Falso No Inform.}}$$

3.8.2. Especificidad:

Nos indica la capacidad de nuestro estimador para dar como casos negativos los casos realmente sanos; proporción de sanos correctamente identificados.

$$\frac{\sum \text{Verdadero No Inform.}}{\sum \text{Verdadero No Inform.} + \sum \text{Falso Inform.}}$$

3.8.3. Exactitud:

Se refiere a cuán cerca del valor real se encuentra el valor medido.

$$\frac{\sum \text{Verdadero Inform.} + \sum \text{Verdadero No Inform.}}{\sum \text{Verdadero Inform.} + \sum \text{Falso Inform.} + \sum \text{Verdadero No Inform.} + \sum \text{Falso No Inform.}}$$

3.8.4. Precisión:

Es la proporción de muestras correctamente diagnosticadas del total de muestras evaluadas.

$$\frac{\sum \text{Verdadero Inform.}}{\sum \text{Verdadero Inform.} + \sum \text{Falso Inform.}}$$

3.8.5. Medida F:

Nos permite corregir el error de la Distancia, en los casos en los que la revocación (R) y la precisión (P) se compensan, dando así, una medida de precisión al algoritmo.

$$2 * \frac{\text{Prec} * \text{Sen}}{\text{Prec} + \text{Sen}}$$



3.9. Criterios éticos

Social: El presente proyecto ayudaría a reducir los cortes inesperados de energía eléctrica, generando un apoyo a las empresas que suministran este tipo de energía y brindando un mejor servicio a la población.

Responsabilidad: Este proyecto de investigación se le considera de manera responsable porque prevendría cortes de energía eléctrica y evitaría electrocuciones a las personas, debido a que se realizaría un mantenimiento preventivo a los aisladores y de este modo, se aseguraría un alto índice de aislamiento.

3.10. Criterios de rigor científico

Tecnológico: Este proyecto ayudaría a la escuela de ingeniería de sistemas y también a la escuela de mecánica eléctrica, debido a que actualmente no existe una investigación que pueda reconocer impurezas en los aisladores eléctricos de media o de alta tensión.

Población apropiada: Todas las imágenes de la población serán fotografías tomadas de aisladores eléctricos de diferentes ángulos. Se contarán con un máximo de 3 fotografías de un mismo aislador, esto con la finalidad de que el proceso de reconocimiento de imágenes pueda detectar y ser más certero en los resultados que se obtengan.

CAPITULO IV: ANALISIS E INTERPRETACIÓN DE LOS RESULTADOS

4.1. Resultados en tablas y gráficos

Para lograr los resultados se consideraron 200 porciones de imágenes (150 para entrenamiento y 50 para clasificación) de aisladores, entre las cuales se clasificaron según su nivel de suciedad, obteniendo: sucios y limpios.



Figura 13: Ejemplo de Imágenes Tomadas



El número de imágenes utilizadas para cada clasificación según la suciedad ha sido la siguiente:

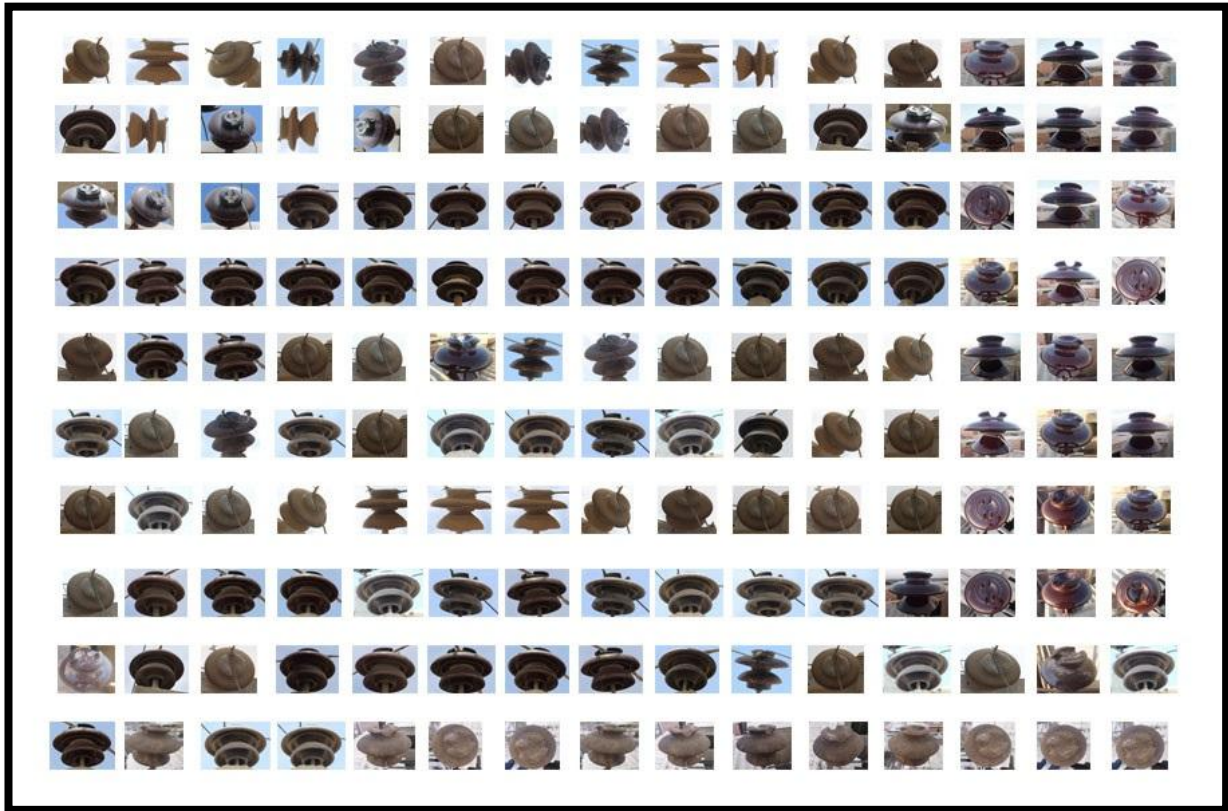


Figura 14: Imágenes de aisladores eléctricos utilizadas en el entrenamiento para los clasificadores.

Fuente: Propia.

Tabla 2: Cantidad de imágenes utilizadas para la investigación. Fuente: Propia.

Imágenes	Entrenamiento	Clasificación	Total - Estado
Sucios	113	30	143
Limpios	37	20	57
TOTAL	150	50	200



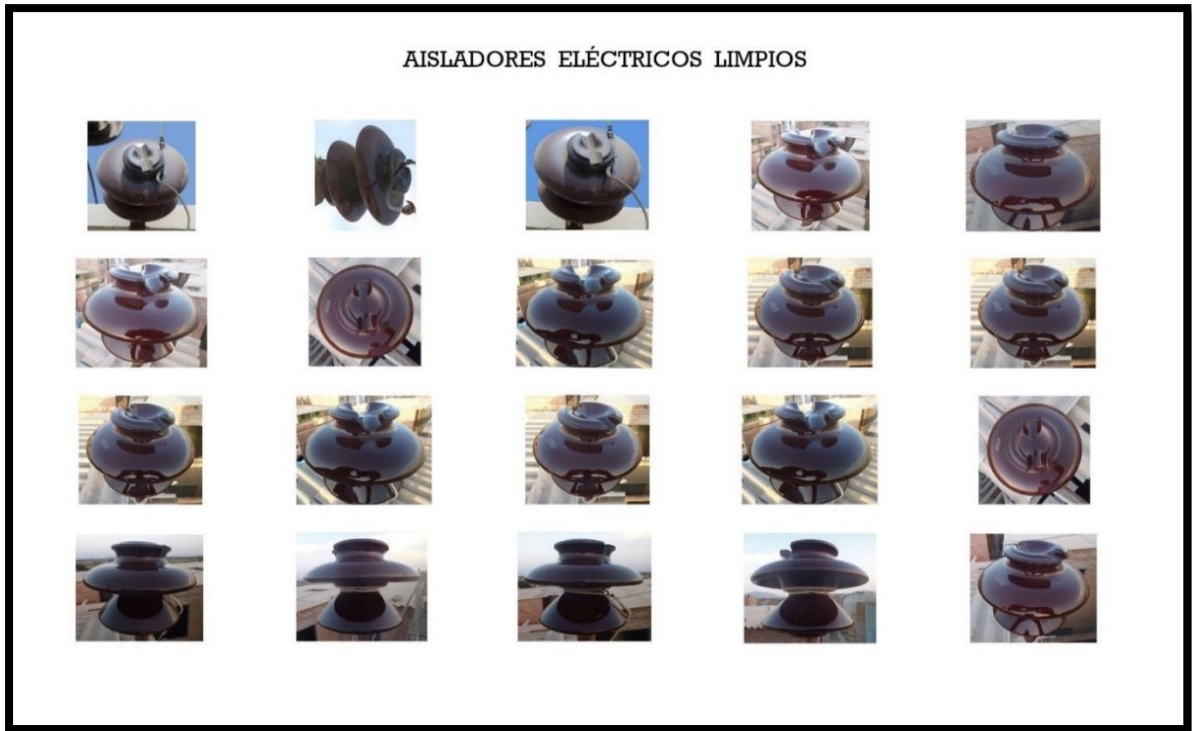


Figura 15: Imágenes de aisladores eléctricos en estado limpio, utilizados para realizar el test en los clasificadores. Fuente: Propia.



Figura 16: Imágenes de aisladores eléctricos en estado sucio, utilizados para realizar el test en los clasificadores. Fuente: Propia.

Las pruebas fueron llevadas en un equipo con procesador Intel Core I5 – 4440, 16 GB RAM DDR3, HDD a 5700 rpm y grafica Intel HD4600.

4.1.1. Resultados Generales:

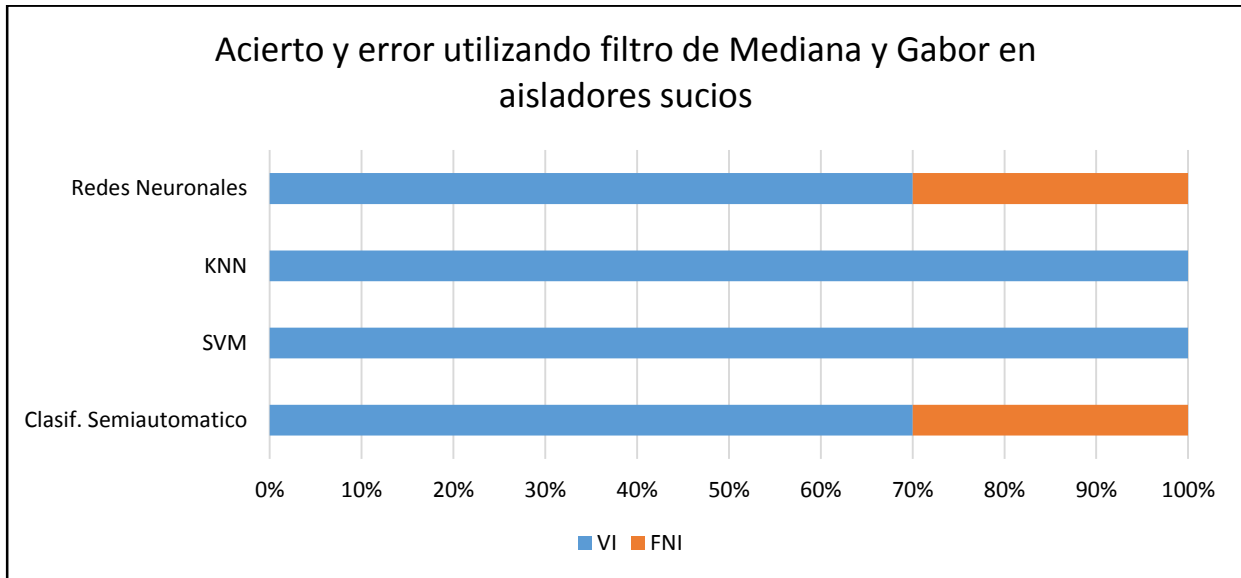
Estos resultados se obtuvieron al usar el algoritmo en segmentación Otsu, por considerarse el mejor según nuestros antecedentes.

Tabla 3: Resultados con diferentes algoritmos de clasificación, evaluación según la matriz de confusión con el indicador de sensibilidad. Fuente: Propia.

Algoritmos de Pre Procesamiento	Algoritmos de Segmentación	Algoritmos de Extracción	Algoritmos de Clasificación			
			Clasif. Semiautomático	SVM	KNN	Redes Neuronales
Filtro Mediana	Otsu	Gabor	70.0%	100.0%	100.0%	70.0%
		LBP	80.0%	80.0%	80.0%	76.7%
Gabor		70.0%	70.0%	70.0%	70.0%	
LBP		80.0%	80.0%	80.0%	80.0%	
Filtro Normalizado						

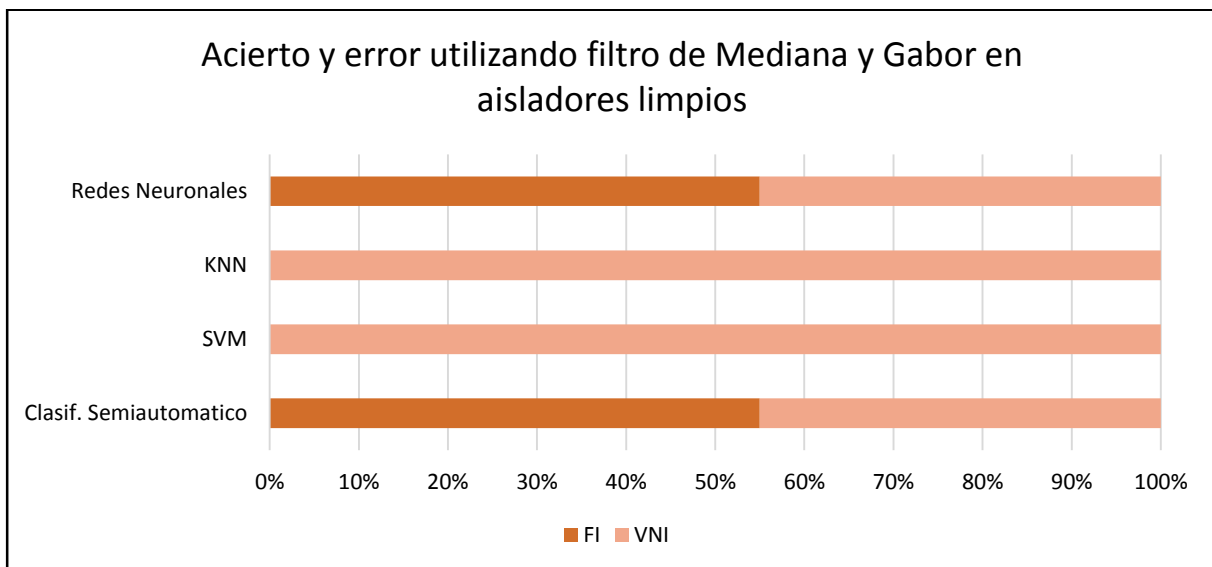


Gráfico 1: Acierto y error utilizando Filtro de Mediana y Gabor en aisladores sucios. Fuente: Propia.



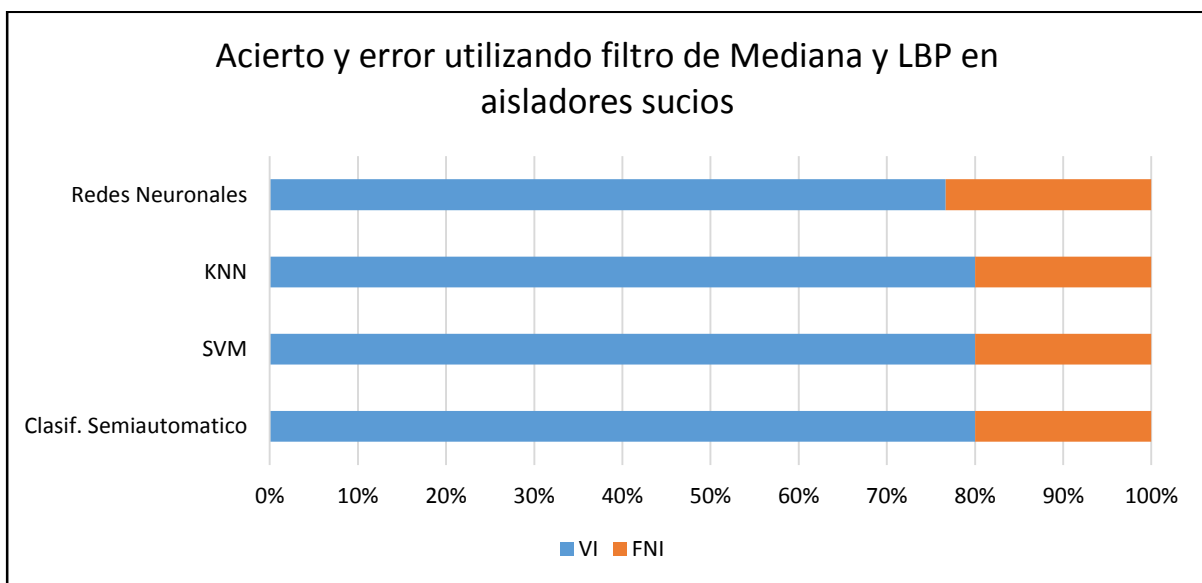
Utilizando filtro de Mediana y Gabor, la clasificación en aisladores sucios fue de un 70% de aciertos y 30% de error en los clasificadores semiautomáticos y redes neuronales, y casi el 100% de acierto en los clasificadores kNN y SVM.

Gráfico 2: Acierto y error utilizando Filtro de Mediana y Gabor en aisladores limpios. Fuente: Propia.



Utilizando filtro de Mediana y Gabor, la clasificación en aisladores limpios fue de casi 55% de aciertos y 45% de error en los clasificadores semiautomáticos y redes neuronales, mientras que los clasificadores KNN y SVM no tuvieron aciertos.

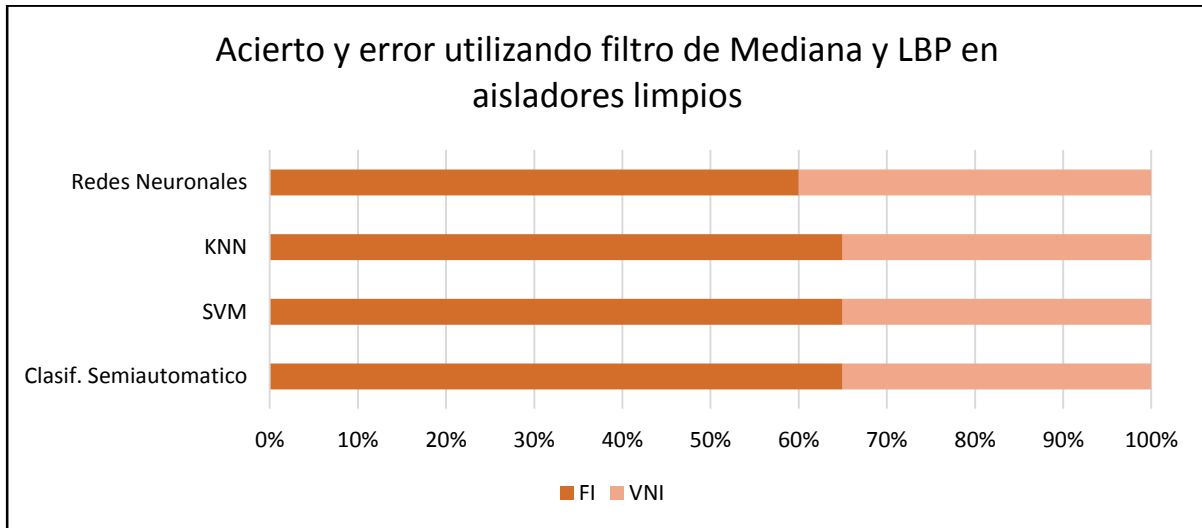
Gráfico 1: Acierto y error utilizando filtro de Mediana y LBP en aisladores sucios. Fuente: Propia.



El los algoritmos filtro de Mediana y LBP, la clasificación en aisladores sucios fue de casi 80% de aciertos y 20% de error en los clasificadores semiautomático, KNN y SVM y más de 75% de aciertos en el clasificador de redes neuronales.

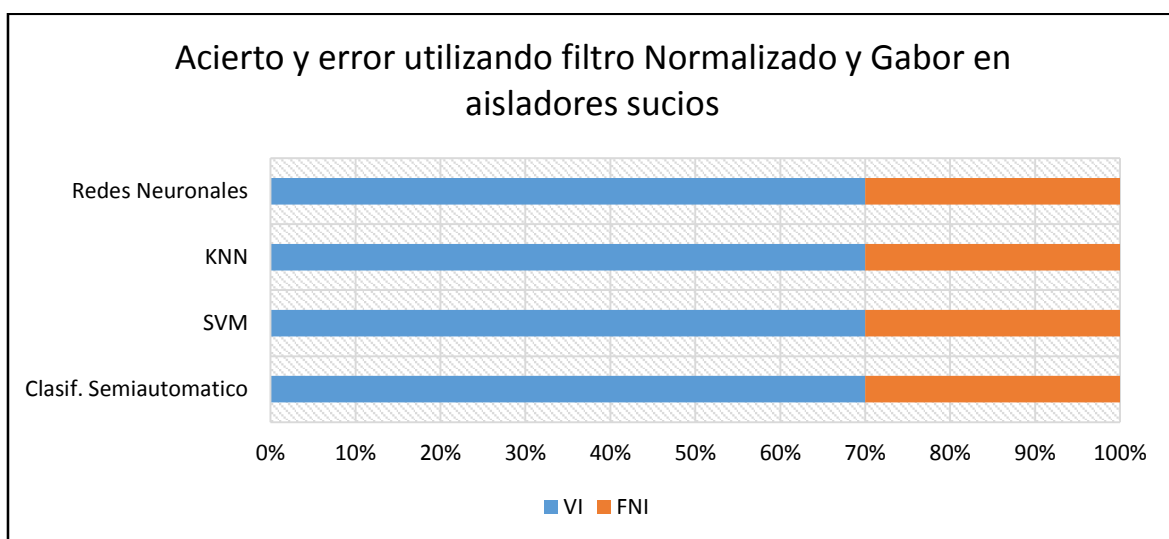


Gráfico 2: Acierto y error utilizando filtro de Mediana y LBP en aisladores limpios. Fuente: Propia.



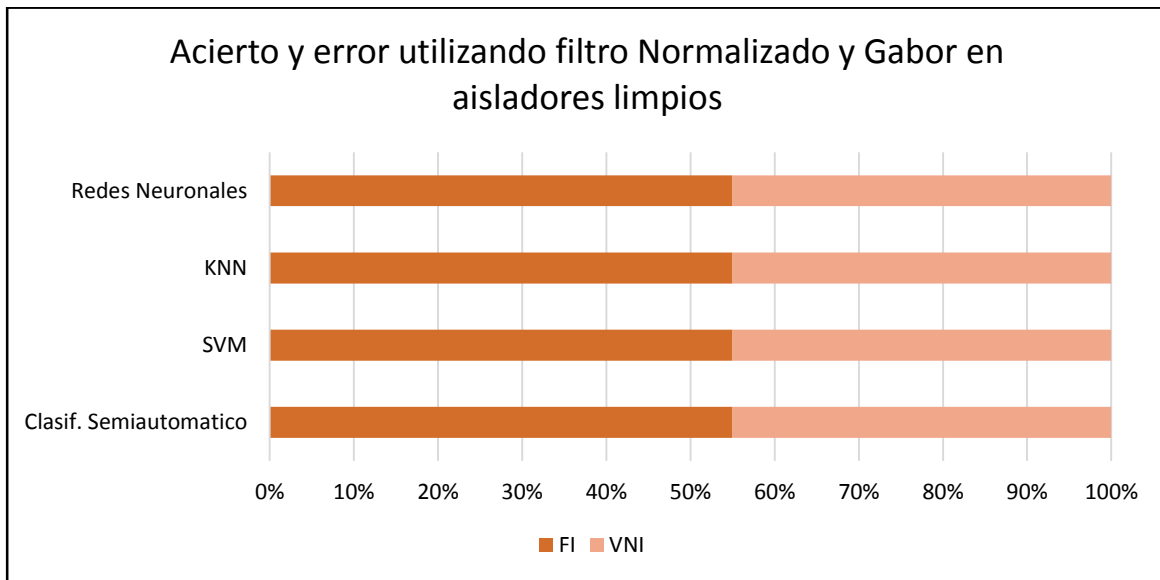
Al utilizar algoritmos filtro de Mediana y LBP, la clasificación en aisladores limpios fue de casi 65% de aciertos y 35% de error en los clasificadores KNN, SVM y semiautomático, y 60% de aciertos en redes neuronales.

Gráfico 5: Acierto y error utilizando filtro Normalizado y Gabor en aisladores sucios. Fuente: Propia.



Al aplicar filtro normalizado y Gabor, los resultados en la clasificación en el caso de los aisladores sucios, mostro una tasa de acierto de 70% en todos los clasificadores y un 30% margen de error.

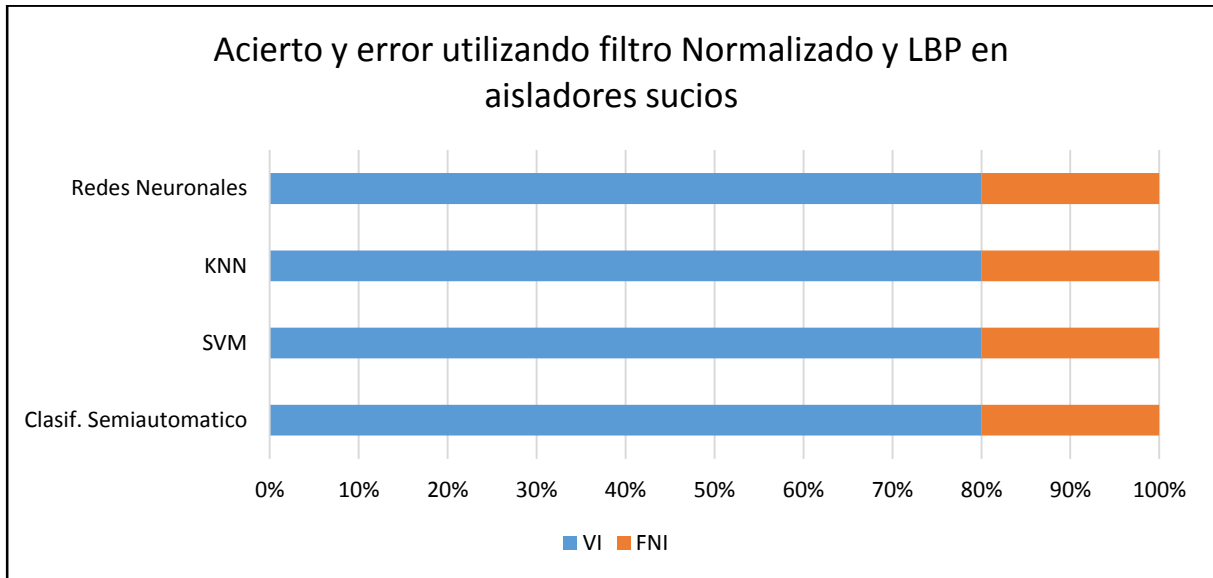
Gráfico 3: Acierto y error utilizando filtro Normalizado y Gabor en aisladores limpios. Fuente: Propia.



Utilizando filtro normalizado y Gabor, los resultados en la clasificación en los aisladores limpios, mostro el 55% de aciertos en todos los clasificadores y un 45% margen de error.

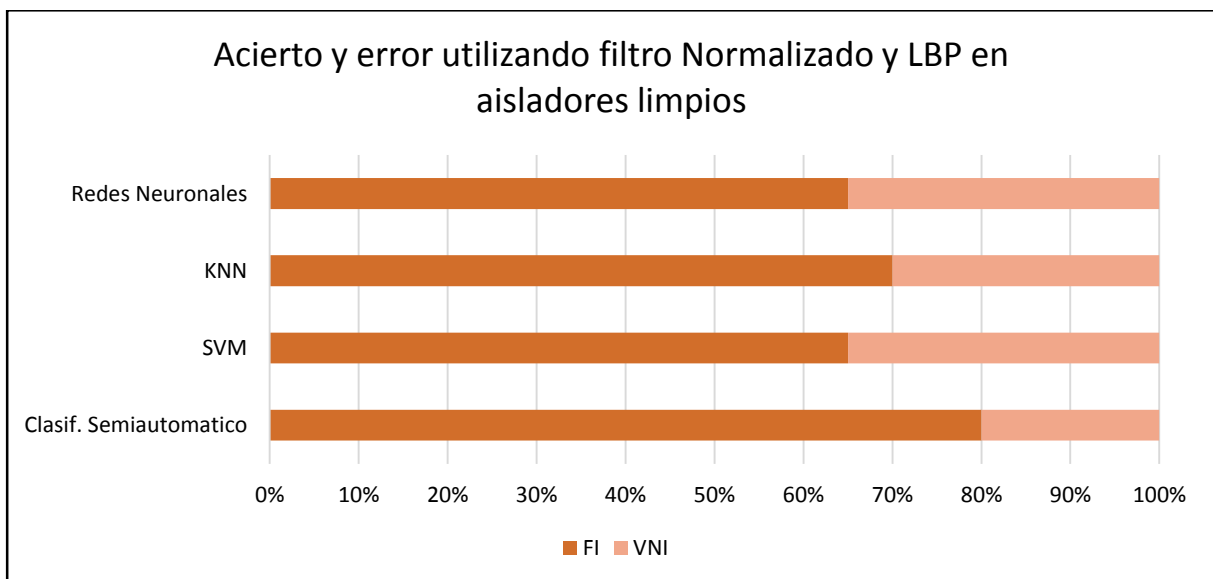


Gráfico 4: Acierto y error utilizando filtro Normalizado y LBP en aisladores sucios. Fuente: Propia.



Utilizando filtro normalizado y LBP, los resultados en la clasificación en los aisladores sucios, mostro el 80% de aciertos en todos los clasificadores con un 20% de desaciertos.

Gráfico 5: Acierto y error utilizando filtro Normalizado y LBP en aisladores limpios. Fuente: Propia.



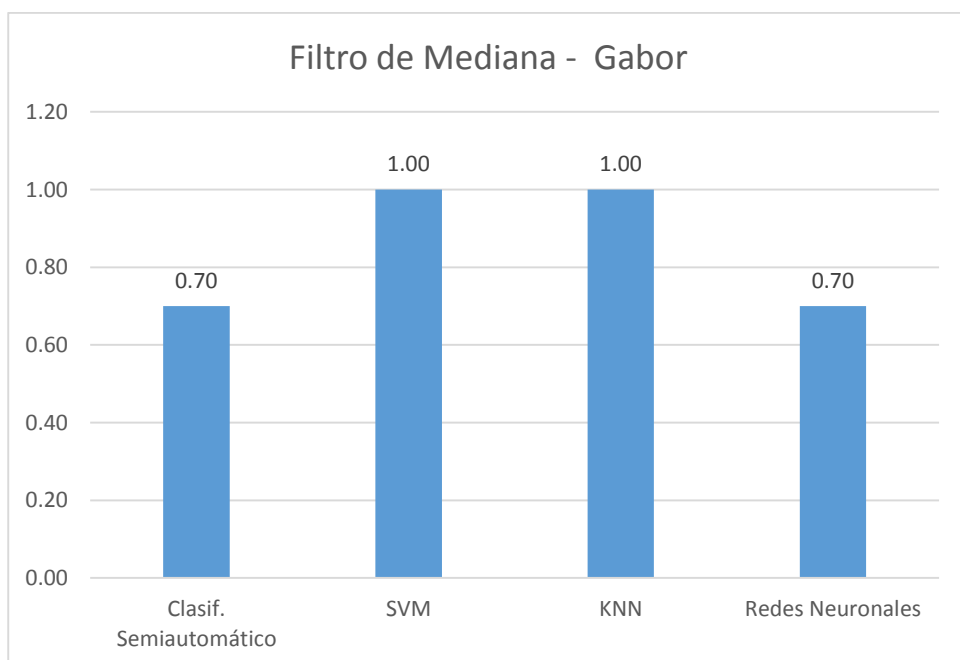
Utilizando filtro normalizado y LBP, los resultados en la clasificación en los aisladores limpios, tanto en redes neuronales, KNN, SVM y el semiautomático arrojo una tasa de aciertos del 65%, 70%, 65% y 80% respectivamente, mientras que un margen de error de 35%, 30%, 35% y el 20% respectivamente.

Los resultados presentados en los gráficos anteriormente demuestran que en cuanto a efectos satisfactorios los algoritmos filtro normalizado y el extractor de características LBP o patrones locales binarios son más efectivos en comparación con los demás algoritmos.

4.1.1.1. Algoritmo de Pre Procesamiento Filtro Mediana:

4.1.1.1.1. Algoritmo de Extracción Gabor:

Gráfico 9: Valores porcentuales de los algoritmos Filtro de Mediana y filtro de Gabor, basado en el indicador de sensibilidad. Fuente: Propia.

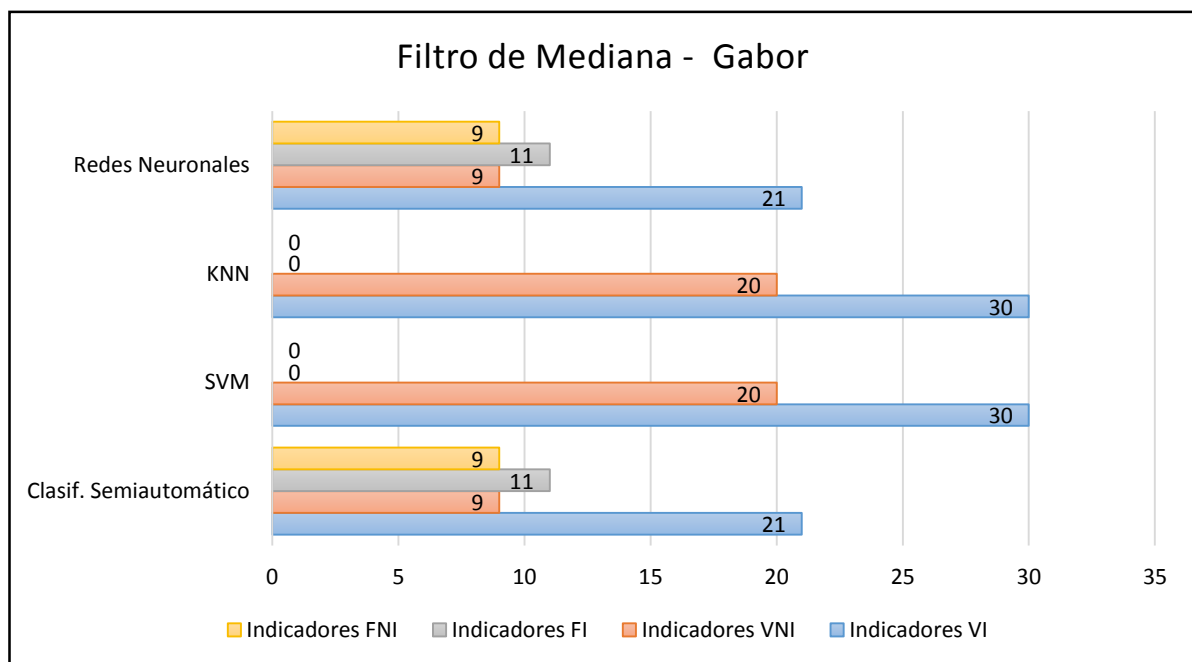


En el grafico podemos ver que cuando usamos el algoritmo de Filtro de Mediana en Pre procesamiento con el algoritmo Gabor en extracción, se obtienen mejores resultados en el algoritmo clasificador de SVM y KNN seguido por Redes Neuronales y Clasificación semiautomático.

Tabla 4: Cuadro de confusión de Filtro de Mediana y filtro de Gabor. Fuente: Propia.

Algoritmos de Extracción	Algoritmos de Clasificación	Indicadores			
		VI	VNI	FI	FNI
Gabor	Clasif. Semiautomático	21	9	11	9
	SVM	30	20	0	0
	KNN	30	20	0	0
	Redes Neuronales	21	9	11	9

Gráfico 10: Valores de los indicadores de los algoritmos Filtro de Mediana y Gabor. Fuente: Propia.

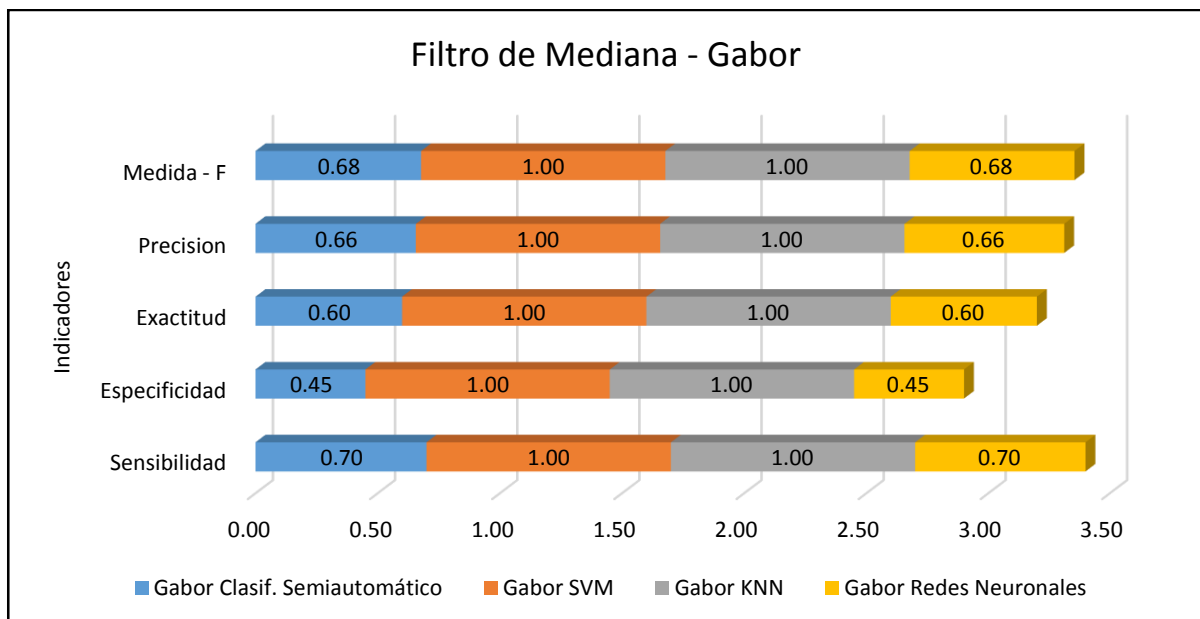


Los resultados de la tabla muestran que los clasificadores KNN y SVM fueron más efectivos al momento de clasificar los aisladores tanto sucios como limpios.

Tabla 5: Indicadores de rendimiento de los algoritmos Filtro de Mediana y Gabor. Fuente: Propia.

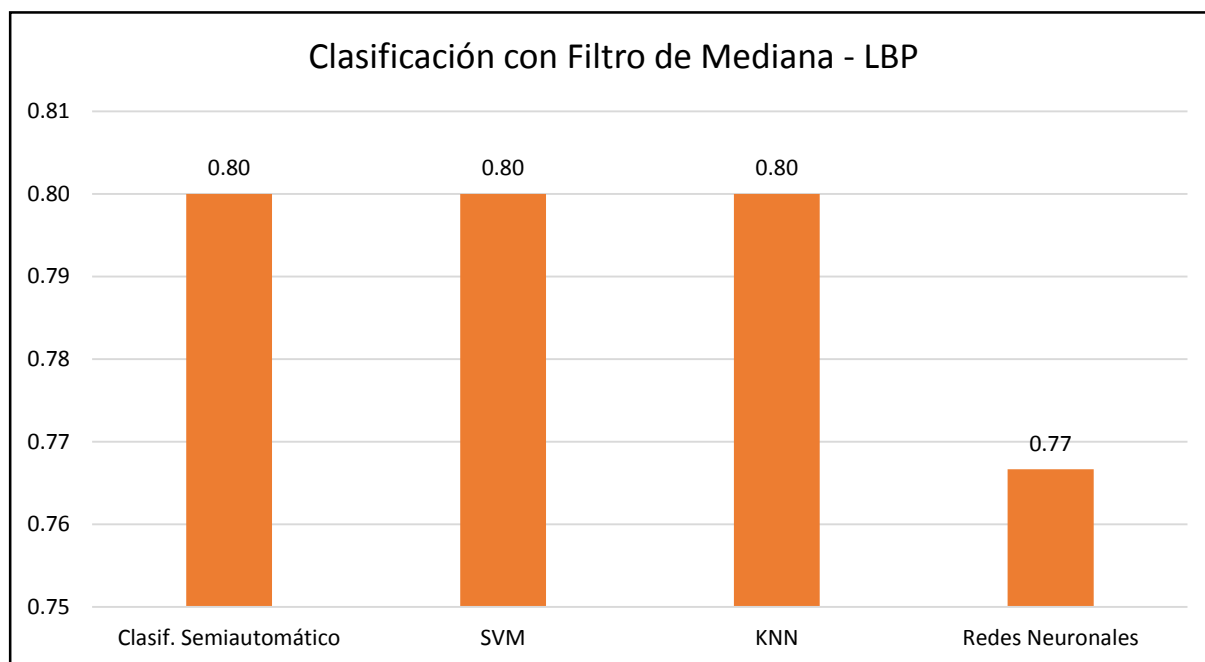
Algoritmos de Extracción	Algoritmos de Clasificación	Indicadores				
		Sensibilidad	Especificidad	Exactitud	Precisión	Medida - F
Gabor	Clasif. Semiautomático	0.700	0.450	0.600	0.656	0.677
	SVM	1.000	1.000	1.000	1.000	1.000
	KNN	1.000	1.000	1.000	1.000	1.000
	Redes Neuronales	0.700	0.450	0.600	0.656	0.677

Gráfico 11: Valores de los indicadores de rendimiento de los algoritmos Filtro de Mediana y Gabor. Fuente: Propia.



4.1.1.1.2. Algoritmo de Extracción LBP:

Gráfico 12: Valores porcentuales de los algoritmos Filtro de Mediana y LBP. Fuente: Propia.



En el gráfico vemos que los resultados basados en el indicador de sensibilidad son similares en los algoritmos SVM, KNN y semiautomático, estas alcanzan tasas altas a diferencia del clasificador redes neuronales.

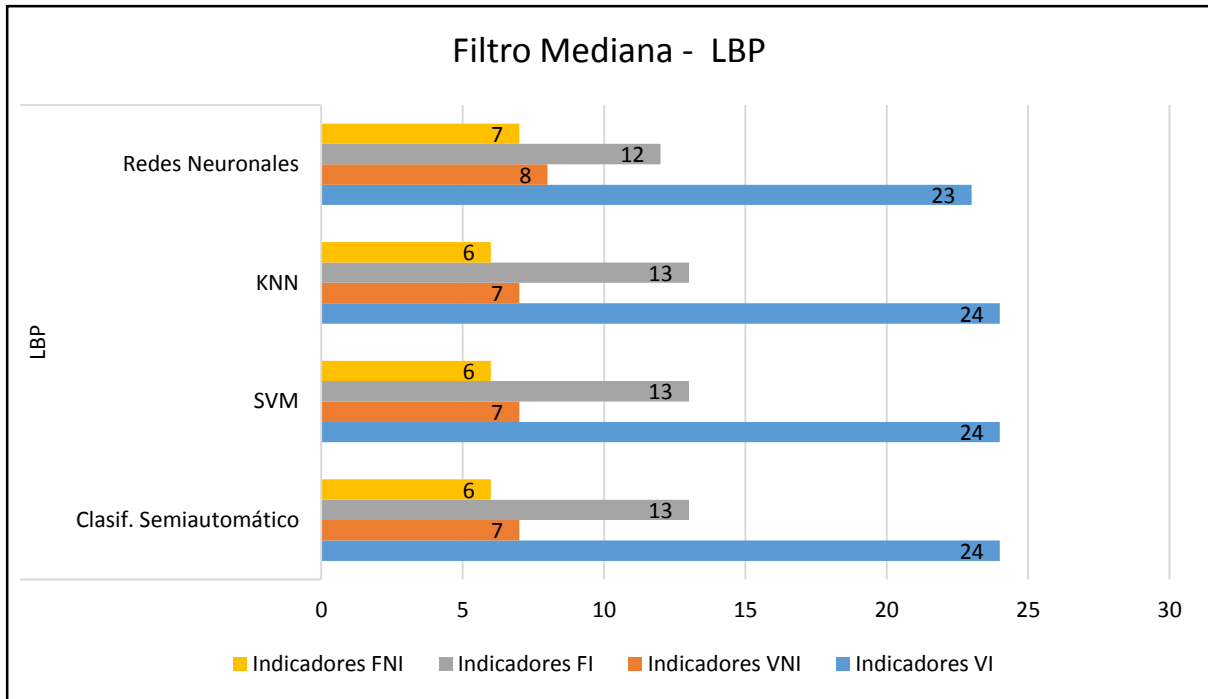
Tabla 6: Matriz de confusión de Filtro de Mediana y LBP. Fuente: Propia

Algoritmos de Extracción	Algoritmos de Clasificación	Indicadores			
		VI	VNI	FI	FNI
LBP	Clasif. Semiautomático	24	7	13	6
	SVM	24	7	13	6
	KNN	24	7	13	6
	Redes Neuronales	23	8	12	7



Gráfico 13: Valores de los indicadores de los algoritmos Filtro de Mediana y LBP.

Fuente: Propia.



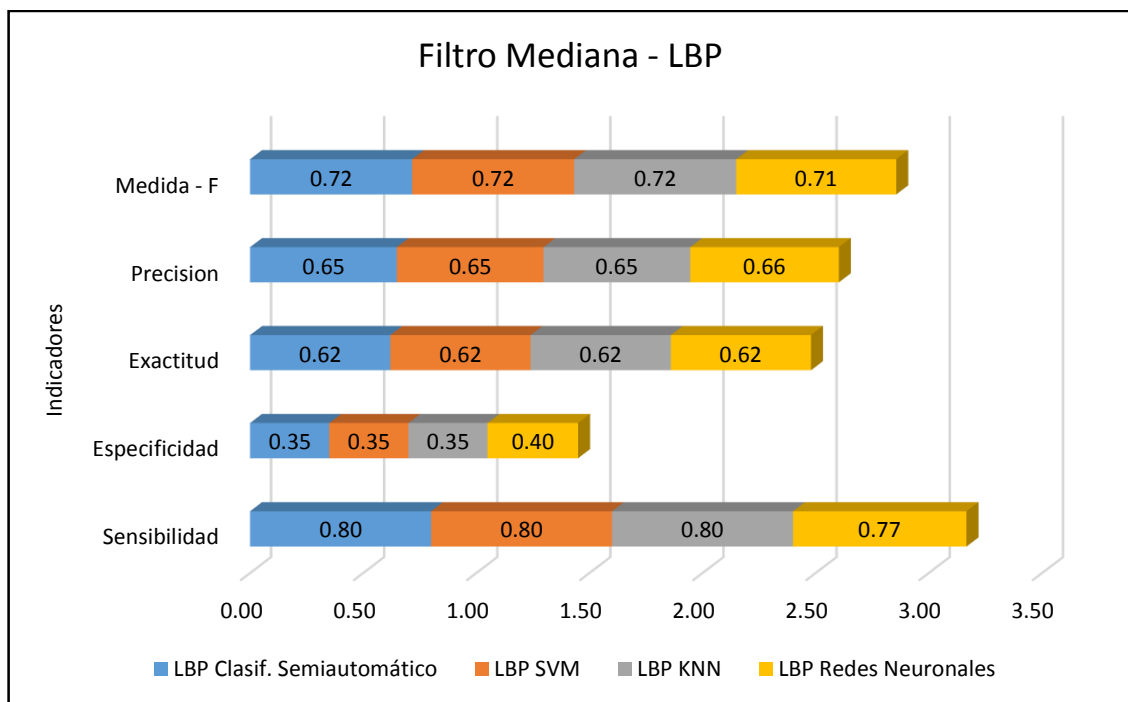
Los resultados de los clasificadores utilizando el algoritmo de LBP en extracción de características muestra un mejor rendimiento al detectar 24 aisladores sucios en los clasificadores como knn, svm y semiautomático de 30 aisladores sucios, mientras que 13 aisladores limpios en los clasificadores knn, svm y semiautomático de 20 aisladores limpios.



Tabla 7: Indicadores de rendimiento de los algoritmos Filtro de Mediana y LBP. Fuente: Propia.

Algoritmos de Extracción	Algoritmos de Clasificación	Indicadores				
		Sensibilidad	Especificidad	Exactitud	Precisión	Medida - F
LBP	Clasif. Semiautomático	0.800	0.350	0.620	0.649	0.716
	SVM	0.800	0.350	0.620	0.649	0.716
	KNN	0.800	0.350	0.620	0.649	0.716
	Redes Neuronales	0.767	0.400	0.620	0.657	0.708

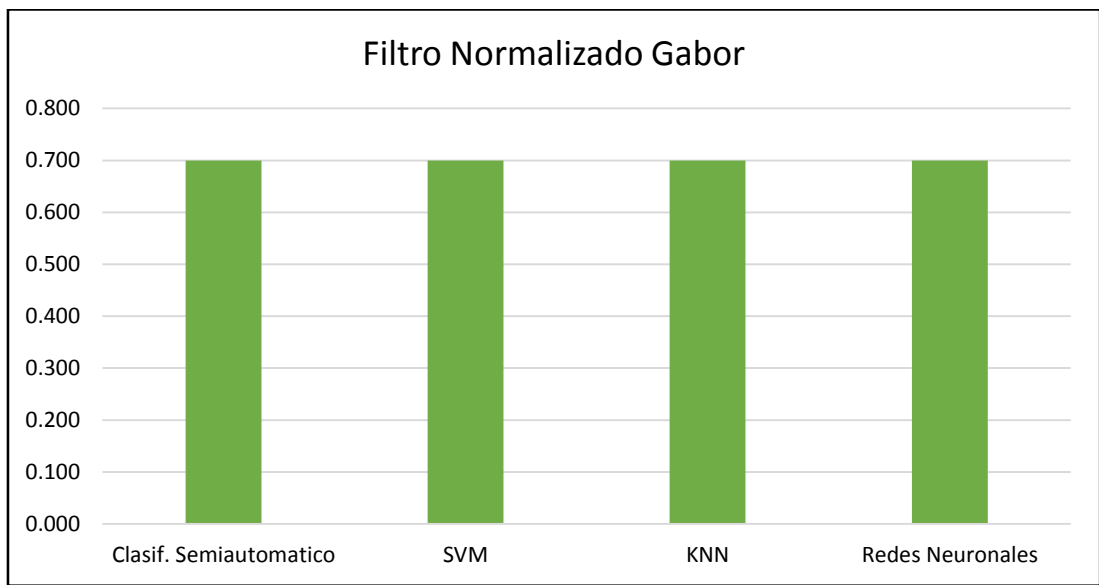
Gráfico 14: Valores de los indicadores de rendimiento de los algoritmos Filtro de Mediana y LBP. Fuente: Propia.



4.1.1.2. Algoritmo de Pre Procesamiento Filtro Normalizado

4.1.1.2.1. Algoritmo de Extracción Gabor:

Gráfico 15: Valores porcentuales de los algoritmos Filtro Normalizado y Gabor basado en el indicador de sensibilidad. Fuente: Propia.



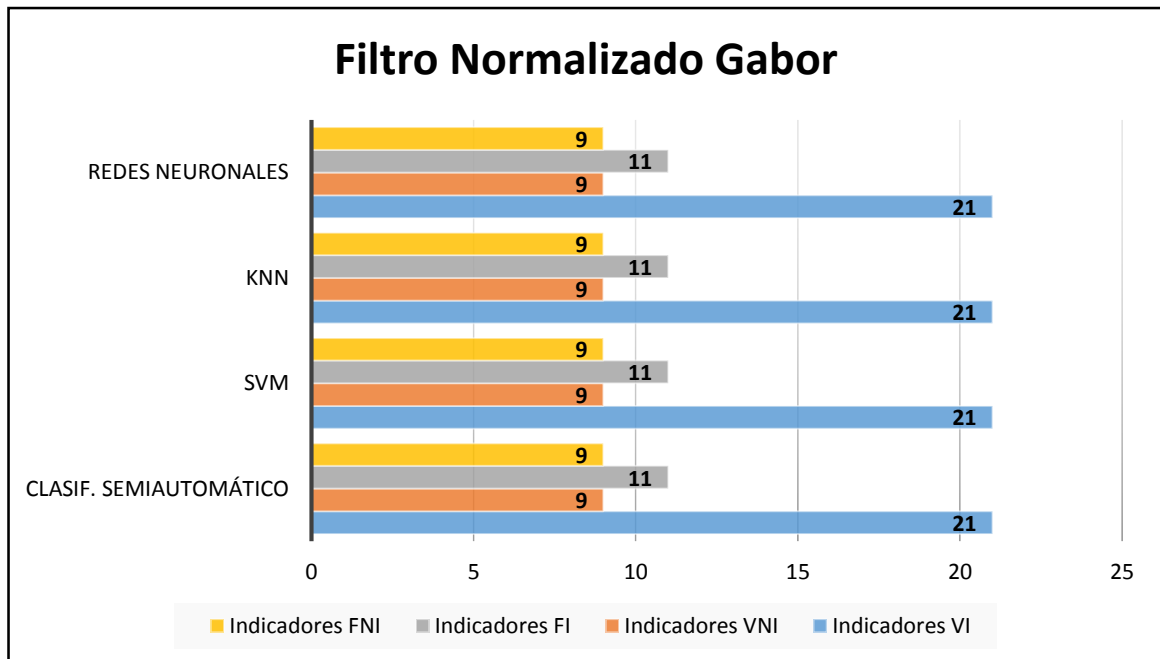
En el gráfico podemos ver que existe similitud en los resultados en todos los algoritmos de clasificación.

Tabla 8: Matriz de confusión de los algoritmos Filtro Normalizado y Gabor. Fuente: Propia.

Algoritmos de Extracción	Algoritmos de Clasificación	Indicadores			
		VI	VNI	FI	FNI
Gabor	Clasif. Semiautomático	21	9	11	9
	SVM	21	9	11	9
	KNN	21	9	11	9
	Redes Neuronales	21	9	11	9



Gráfico 16: Valores de los indicadores de los algoritmos Filtro Normalizado y Gabor. Fuente: Propia.



Utilizando filtro normalizado y Gabor, los clasificadores no muestra que se detectaron 21 aisladores sucios de 30 aisladores sucios, mientras que se clasifico 11 aisladores limpios de 20 aisladores limpios.



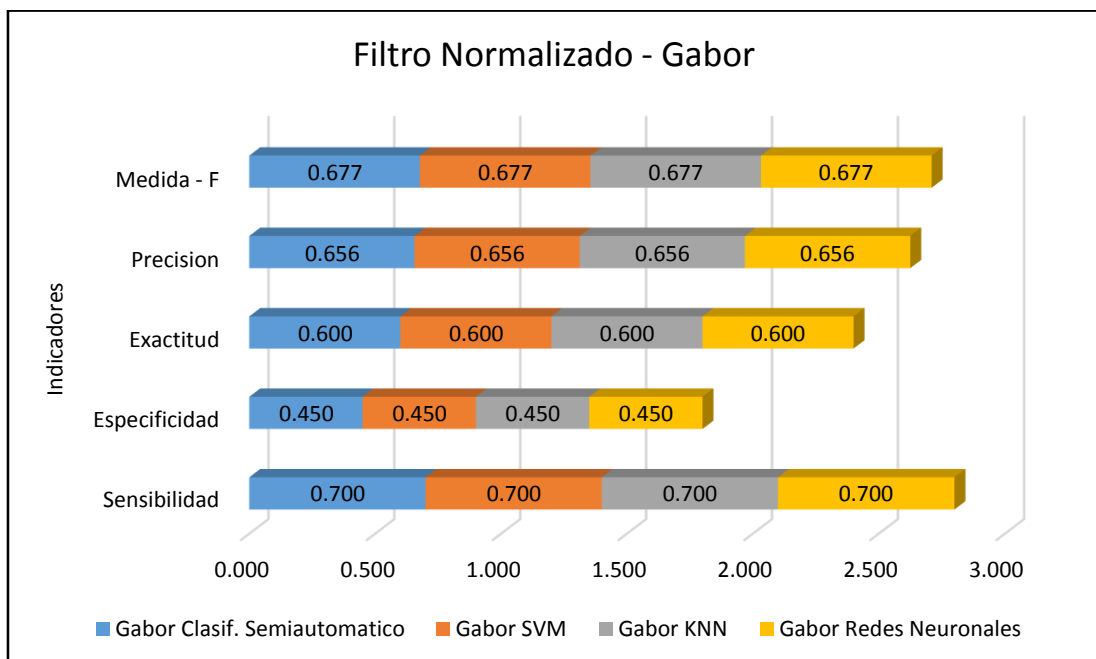
Tabla 9: Indicadores de rendimiento de los algoritmos Filtro Normalizado y Gabor. Fuente:

Propia.

Algoritmos de Extracción	Algoritmos de Clasificación	Indicadores				
		Sensibilidad	Especificidad	Exactitud	Precisión	Medida - F
Gabor	Clasif. Semiautomático	0.700	0.450	0.600	0.656	0.677
	SVM	0.700	0.450	0.600	0.656	0.677
	KNN	0.700	0.450	0.600	0.656	0.677
	Redes Neuronales	0.700	0.450	0.600	0.656	0.677

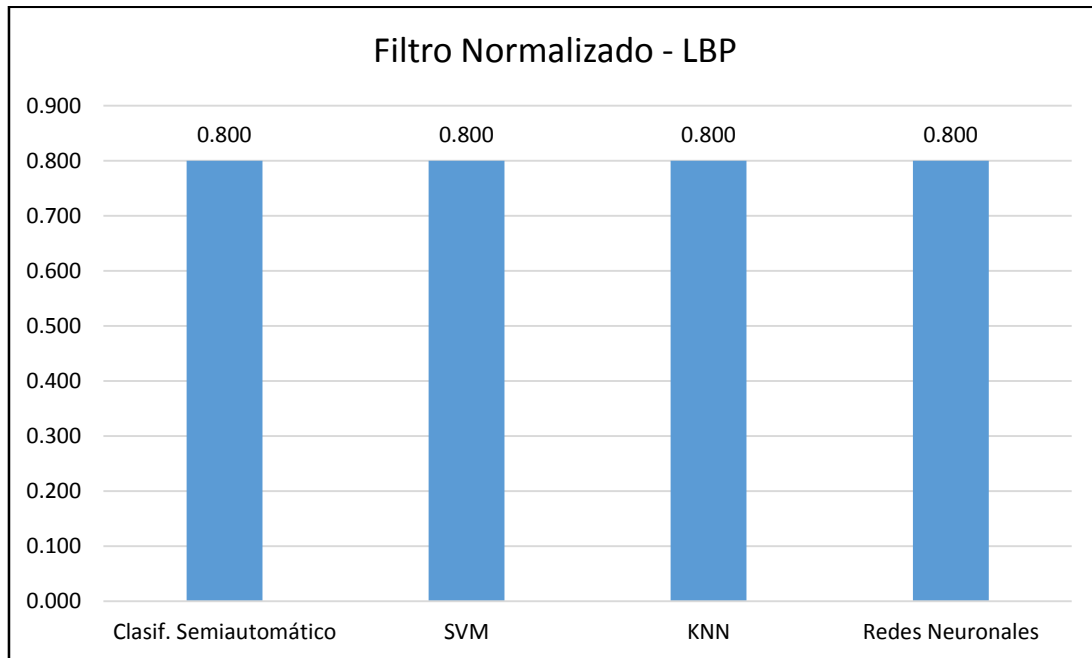
Gráfico 17: Valores de los indicadores de rendimiento de los algoritmos Filtro Normalizado y

Gabor. Fuente: Propia.



4.1.1.2.2. Algoritmo de Extracción LBP:

Gráfico 18: Valores porcentuales de los algoritmos Filtro Normalizado y LBP. Fuente: Propia.



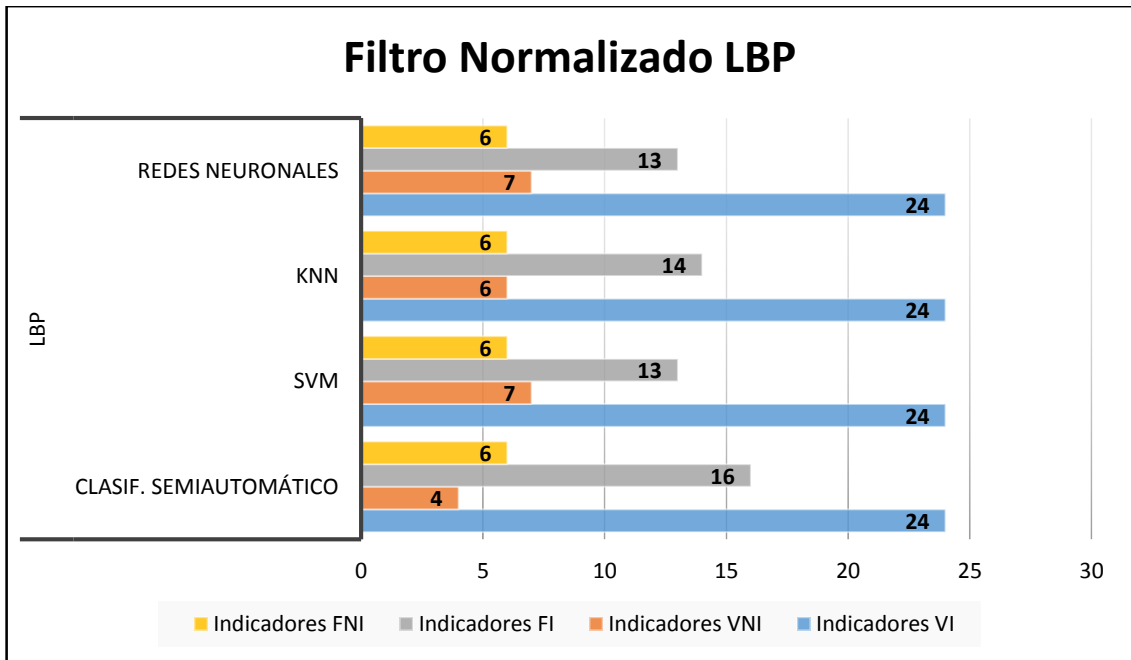
En el grafico vemos una igualdad en los resultados al usar todos los algoritmos de clasificación.

Tabla 10: Matriz de confusión, utilizando los algoritmos Filtro Normalizado y LBP. Fuente: Propia.

Algoritmos de Extracción	Algoritmos de Clasificación	Indicadores			
		VI	VNI	FI	FNI
LBP	Clasif. Semiautomático	24	4	16	6
	SVM	24	7	13	6
	KNN	24	6	14	6
	Redes Neuronales	24	7	13	6



Gráfico 19: Valores de la matriz de confusión en Filtro Normalizado y LBP. Fuente: Propia



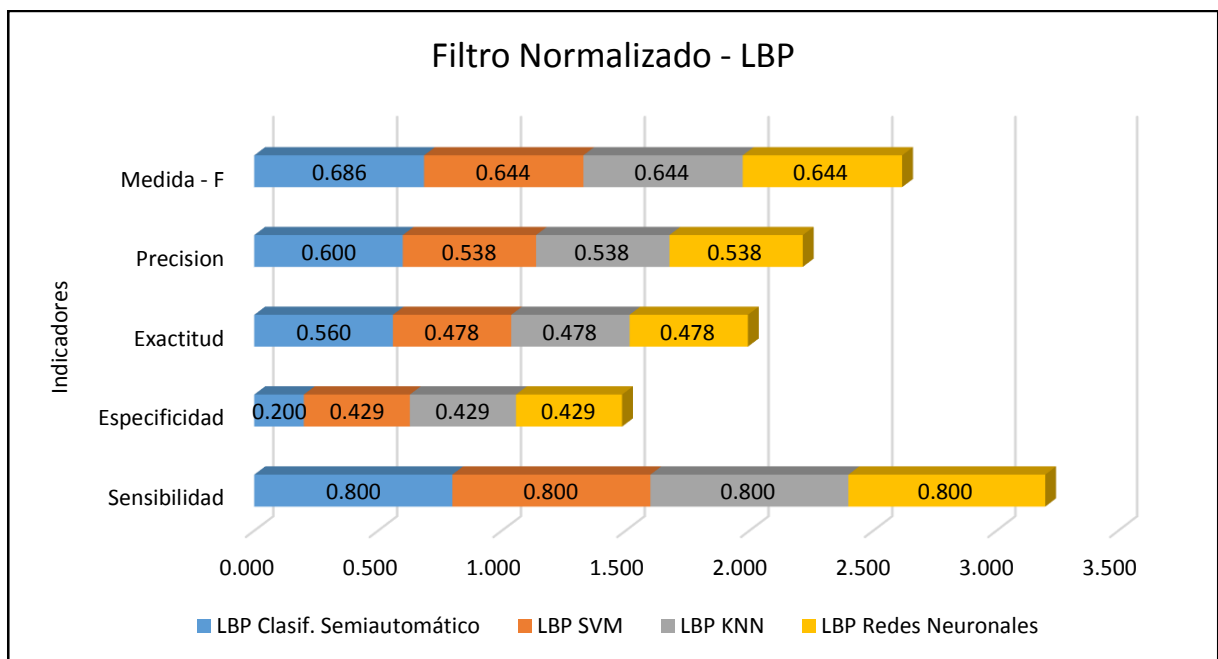
El gráfico muestra que utilizando filtro normalizado y LBP, los clasificadores catalogaron 24 aisladores sucios de 30 aisladores sucios. En el caso de los aisladores limpios la cual se ingresó 20 imágenes los clasificadores redes neuronales y SVM detecto 13 aisladores limpios, 14 aisladores knn y 16 aisladores el clasificador semiautomático de las 20 ingresadas.



Tabla 11: Indicadores de rendimiento de los algoritmos Filtro Normalizado y LBP. Fuente: Propia.

Algoritmos de Extracción	Algoritmos de Clasificación	Indicadores				
		Sensibilidad	Especificidad	Exactitud	Precisión	Medida - F
LBP	Clasif. Semiautomático	0.800	0.200	0.560	0.600	0.686
	SVM	0.800	0.429	0.478	0.538	0.644
	KNN	0.800	0.429	0.478	0.538	0.644
	Redes Neuronales	0.800	0.429	0.478	0.538	0.644

Gráfico 20: Valores de los indicadores de rendimiento de los algoritmos Filtro Normalizado y LBP. Fuente: Propia.



CAPÍTULO V: PROPUESTA DE INVESTIGACION

Existen diversos tipos de materiales a usar en la fabricación de aisladores eléctricos, como porcelana, vidrio y resina epóxica, sin embargo, en esta investigación se eligieron los hechos de porcelana.

(Gemini.udistrital. 2006) La porcelana es el material más utilizado en la fabricación de aisladores, está conformado por arcilla, feldespato y cuarzo o alúmina, la porcelana ofrece baja porosidad lo que ayuda a la baja absorción de agua, alta resistencia al calor y resistencia mecánica.



Figura 17: Aislador cerámico Tipo Pin. Fuente: Gemini.udistrital

En base a las condiciones encontradas en aisladores de porcelana al efectuar su mantenimiento, se han tomado las siguientes características físicas sobre los aisladores, con el objetivo de encontrar patrones fundamentales para ejecutar esta investigación.

5.1. Diseño del Sistema

En este proyecto tenemos 2 etapas importantes para poder clasificar las imágenes, mostramos estas etapas en la siguiente tabla:

Tabla 12: Imágenes usadas y fases usadas en cada una de las etapas. Fuente: Propia.

Etapa	Fases del procesamiento de imágenes	Imágenes usadas	
		Aisladores Limpios	Aisladores Sucios
Entrenamiento	Pre - procesamiento, segmentación, extracción	37	113
Clasificación	Pre - procesamiento, segmentación, extracción y clasificación	20	30

En la etapa de entrenamiento, las imágenes procesadas no realizan la fase de clasificación, debido a que estas imágenes solo son para entrenar al algoritmo clasificador. Por lo tanto, al finalizar la etapa de entrenamiento y recorrer las 150 imágenes en total, entre sucias y limpias; lo que hacemos es obtener una matriz con 150 filas que nos indica la cantidad de imágenes y generar un archivo en formato txt (se pueden crear 4 archivos txt diferentes según los algoritmos a usar en cada fase, 2 en pre – procesamiento y 2 en extracción) que guardara las



características de las 150 imágenes según los algoritmos usados hasta esa etapa. Así por ejemplo podemos tener un archivo txt con el siguiente nombre: FiltroNormalizado_Gabor.txt (indica que se han usado los algoritmos de filtro normalizado, algoritmo Otsu que usamos en segmentación y algoritmo Gabor en extracción).

Así también en la etapa de clasificación, cada imagen hará el mismo proceso de la etapa de entrenamiento. Pero también se obtendrá una matriz después de la fase de extracción, la cantidad de filas de esta matriz dependerá de las imágenes procesadas a clasificar. En esta etapa también se cargará el archivo txt según los algoritmos a usar en cada fase y se obtendrán 2 matrices de cada archivo txt; una matriz obtendrá los valores de característica de textura y la otra matriz obtendrá un valor de etiqueta si la imagen procesada fue sucia o limpia. La información de estas 2 matrices se le dará al algoritmo clasificador. En total cada algoritmo clasificador recibirá una matriz de las imágenes de entrenamiento (características), una matriz de las etiquetas de entrenamiento (si la imagen fue sucia o limpia) y una matriz de las imágenes de clasificación. Se explicará mejor el proceso en el objetivo de “Implementar algoritmos seleccionados que permitan realizar un análisis efectivo de la imagen”.

5.1.1. Pre Procesamiento:

- La imagen debe tener un fondo más claro que el del aislador.
- En la imagen, el aislador debe estar de forma vertical y ubicada en el centro.
- Después se aplicarán los algoritmos de realce, Filtro Normalizado y Filtro Mediana.



- Obtendremos 200 imágenes de salida con el Filtro Normalizado y 200 con el Filtro Mediana.
- Estas 200 imágenes serán divididas en 150 para la parte de entrenamiento del proceso y 50 imágenes que serán para la clasificación.

5.1.2. Segmentación:

- Aplicamos la segmentación con el algoritmo Otsu a cada una de las 200 imágenes en cada Filtro de Pre Procesamiento.
- Se eligió el algoritmo de Otsu, por haber obtenido mejores resultados con el programa Matlab, donde se hicieron pruebas antes que con el programa Qt Creator y Open CV. En las siguientes imágenes se puede apreciar que en la imagen segmentada por Otsu es mucho más limpia a nivel de los bordes, que la segmentada por Balu (Algoritmo del ToolBox del mismo nombre).



Figura 18: Imagen original de un aislador cerámico. Fuente: Propia.



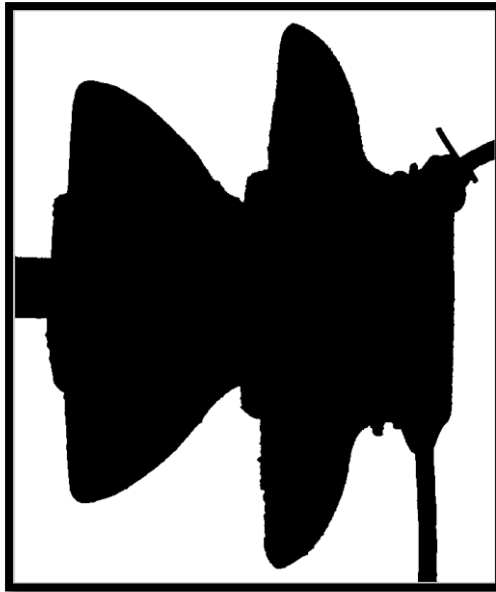


Figura 19: Imagen segmentada – Algoritmo Otsu. Fuente: Propia.

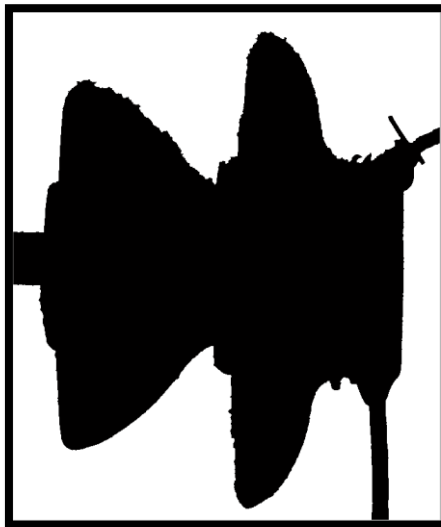


Figura 20: Imagen segmentada – Algoritmo Balu. Fuente: Propia.

- A cada imagen segmentada se le extrajo una porción de imagen de 180 por 180 pixeles.

5.1.3. Extracción de Características:

- A cada una de las porciones de las imágenes segmentadas se les aplico los Algoritmos de Gabor y LBP.

- La extracción de características se hizo con las imágenes en un tono de color de escala de grises.
- Antes de pasar a la etapa de clasificación se hizo un análisis mediante observación según el caso de algoritmo implementado:
 - o LBP: Se observó los valores del vector característico, y los pixeles entre 112 al 127 se repetían en mayor cantidad cuando el aislador estaba limpio, mientras que los valores 240 al 254 se repetían cuando el aislador se encontraba sucio (ver anexo 5).
 - o Gabor: Se observó los valores del vector característico y se pudo notar que la cantidad de contornos de una imagen de aislador limpio eran mayores a 31, mientras que la cantidad de contornos de una imagen de aislador sucio eran menores o iguales a 31. (Ver anexo 6).

Al hacer este análisis se pudo determinar una diferencia entre un aislador limpio de un sucio.

5.1.4. Clasificación:

- Las matrices obtenidas de la etapa de extracción de características fueron clasificadas.
- Se clasificaron las porciones de imágenes con los algoritmos de SVM, KNN, Redes Neuronales probabilísticas y con una clasificación semiautomática.

En la etapa de segmentación solo se utilizó un algoritmo, el cual es Otsu; debido a que estos son necesarios para segmentar la imagen e identificar a



los aisladores. Pero donde realmente se realiza el trabajo es en la etapa de extracción y clasificación, debido a que en estas etapas ya se está trabajando con las porciones de las imágenes segmentadas.

Si no se obtienen los resultados esperados es importante realizar algunos ajustes en alguna etapa del proceso para mejorarlos.

A continuación, se muestra una gráfica del proceso que se siguió:

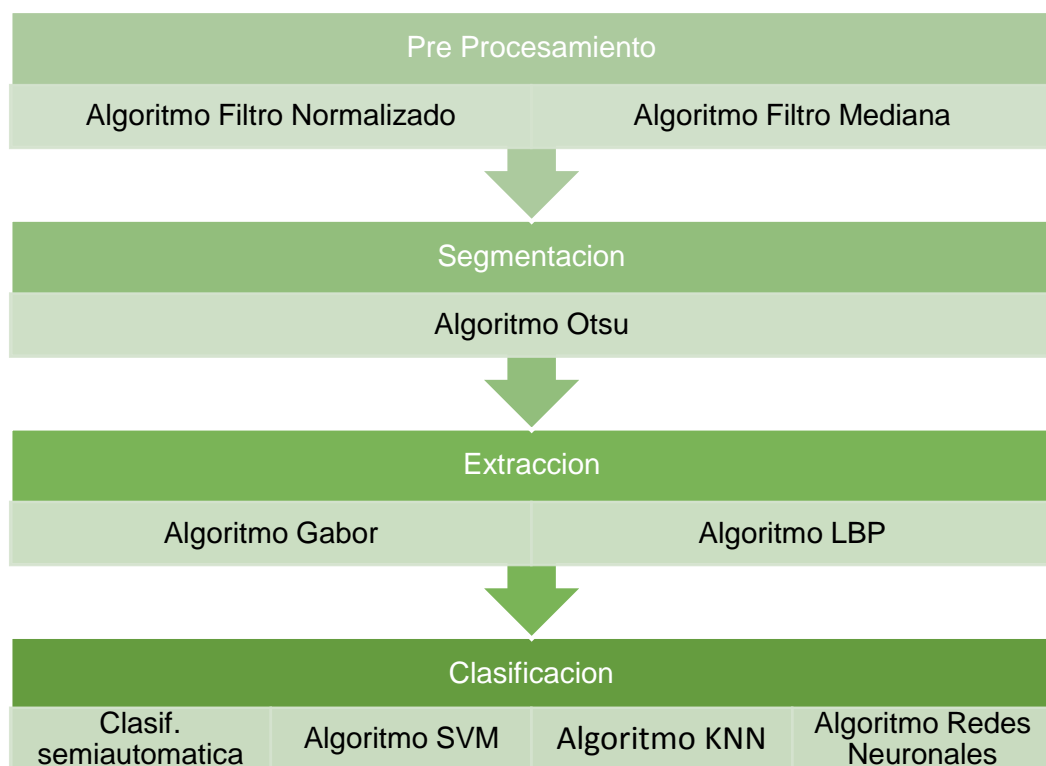


Figura 21: Diseño estructural del sistema en la presente investigación. Fuente: Propia.

5.2. Adquisición y base de imágenes

La adquisición de imágenes es uno de los procesos más importantes en el procesamiento de imágenes, ya que la calidad de las imágenes debe tener una calidad óptima que no afecte al rendimiento del sistema.

Las imágenes de los aisladores muy sucios y medio sucios se tomaron del distrito de Mórrope, provincia de Lambayeque, departamento de Lambayeque



- Perú. Y las imágenes de los aisladores limpios se tomaron en la ciudad de Chiclayo. La toma de las fotografías se hizo con una cámara Canon EOS Rebel T5 y con la ayuda de un lente Canon EF 75 – 300 mm f/4 – 5,6 III. La cual permitió tomas desde varios ángulos, Así, también se consideró que el clima debería ser un día nublado para impedir captar la luz de los rayos solares y posteriormente afectar el proceso de segmentación de las imágenes.

5.3. Realización de Objetivos específicos

a) Identificar patrones en los aisladores eléctricos (Textura, color).

(Ikinormas.Micodensa) Los aisladores serán de porcelana del tipo proceso en húmedo “wet process”, de altas propiedades aislantes, alta resistencia mecánica, alta inercia química, elevado punto de fusión, esmalte color café, porosidad nula, libre de defectos tales como grietas, calcinaciones, burbujas y estar completamente vitrificado. Los aisladores y sus aditamentos deben ser inmunes a la acción de la humedad, el humo, el polvo, el ozono, etc. y a los cambios rápidos de temperatura, en condiciones de trabajo.

Toda la superficie expuesta del aislador debe cubrirse con un vidriado de tipo compresión duro, liso, brillante, impermeable a la humedad que le permita mantenerse fácilmente libre de polvo o suciedades residuales ocasionadas por la contaminación ambiental por medio de lavado natural de las aguas lluvias.



Debido a esta descripción, resaltamos el color y la textura del aislador.



a) **Color:** Esmalte color café.

b) **Textura:** Porosidad nula, libre de defectos tales como grietas, calcinaciones, burbujas y estar completamente vitrificado.



Tabla 13: Identificar patrones en aisladores. Fuente: Propia.

IMÁGENES	CONDICION	DESCRIPCION	PATRONES PRINCIPALES
	<p>Imagen tomada en ambiente no controlado (aprox. 12 pm – 1 pm)</p>	<p>Aislador cubierto por polvo de color madera corpulento con pequeños grumos acumulados en la superficie. La forma del aislador se encuentra intacta a pesar del tiempo transcurrido.</p>	<ul style="list-style-type: none"> -Forma no alterada del aislador. -Color madera corpulento del polvo. -Textura rugosa.
	<p>Imagen tomada en ambiente no controlado (aprox. 12 pm – 1 pm)</p>	<p>Aislador de forma anillada cubierto por polvo de color madera corpulenta con mayor intensidad en la parte izquierda extrema, cuenta también con pequeños grumos acumulados en la superficie.</p>	<ul style="list-style-type: none"> -Forma no alterada del aislador. -Color madera corpulento del polvo. -Textura rugosa.

		<p>La forma del aislador se encuentra intacta a pesar del tiempo transcurrido.</p>	
	<p>Imagen tomada en ambiente controlado bajo un fondo blanco</p>	<p>Aislador de textura lisa, de color café oscuro, libre casi de impurezas solidas en su totalidad, con forma intacta.</p>	<ul style="list-style-type: none"> -Color café oscuro -Textura lisa -Forma intacta
	<p>Imagen tomada en ambiente no controlado (aprox. 12 pm – 1 pm</p>	<p>Aislador cubierto por polvo de color madera corpulento con pequeños grumos acumulados en la superficie. Su textura es rugosa debido al polvo. La forma del aislador se encuentra intacta a pesar del tiempo transcurrido.</p>	<ul style="list-style-type: none"> -Color madera corpulenta del polvo. -Textura rugosa -Forma intacta

b) Seleccionar algoritmos para el reconocimiento de impurezas en el aislador.

Tabla 14: Selección de algoritmos para esta investigación. Fuente: Propia.

ANTECEDENTES	PRE – PROCESAMIENTO	SEGMENTACION	EXTRACCION DE CARACTERISTICAS	CLASIFICACION	EFICIENCIA	Tiempo
Saban, Bayram (2015). “comparación de algoritmos de detección de borde para el análisis de textura en la producción de vidrio”	FILTRO DE MEDIANA	LAPLACIANO GAUSSIANO(LOG) CANNY ROBERTS PREWITT SOBEL	FILTROS DE GABOR TRANSFORMADA WAVELET		76,52%(Log) 75,79%(Canny) 69,4%(Roberts) 68,26%(Prewitt) 64,92%(Soble)	
YongHua, Jin-Cong (2014) “identificación de defectos de la superficie de madera basados en textura”			GLCM (gray-level co-occurrence matrix). Escala de grises matriz de co-ocurrencia Tamura Texture.	Red Neuronal BP(backpropagation)	90,67% de tasa de reconocimiento con Tamura Texture. 91,33% de tasa de reconocimiento utilizando GLCM.	Tiempo medio de funcionamiento de la detección de 1,56 s con tamura texture.

					92,67% de tasa de reconocimiento. mezclando ambos métodos GLM y Tamura textura	Tiempo medio de ejecución es de 1,83 s en GLCM.
Kim, Ho, Man (2014). "reconocimiento de las monedas mediante la rotación invariante binaria región basada en imágenes patrones basados en magnitudes gradiente"			LBP(Local Pattern Binary) FILTROS DE GABOR magnitudes RFR-gradiente(RFR-GM),			81,4 % de exactitud, con 1062 ms de tiempo. 62,7 % de exactitud, con 43 ms de tiempo. 87,4 % de exactitud, con 10 ms de tiempo.

<p>Shen, H., Chen, P., & Chang, L. (2013). “Método automático de reconocimiento por defecto de recubrimiento de óxido en puentes de acero basado en función del color y la textura”</p>	<p>Transformada de Fourier</p>	<p>Espacios de color RGB Espacios de color CMY</p>	<p>RUDERM</p>	<p>K – means RUDERM & K - means</p>	<p>36.83% de óxido (K – means) 8.91% de óxido (RUDERM & K - means)</p>	<p>1.72 s 1.17 s</p>
<p>Merida, M(2012) “RECONOCIMIENTO BIOMÉTRICO BASADO EN IMÁGENES DE HUELLAS PALMARES”</p>	<p>Binarización. Filtrado paso bajo: Filtro gaussiano</p>		<p>Filtros de Gabor transformada de Karhunen-Loeve Transformada de wavelet. Operador de Moravec. Operador Plessey</p>	<p>Distancia de Hamming Distancia de Mahalanobis. Distancia de Manhattan. Distancia euclídea</p>	<p>Filtro de gabor fue el más efectivo que los algoritmos en cuanto a textura.</p>	

			<p>cálculo de GTE (Global Texture Energy) MFRAT (Modified Finite Radon Transform)</p>	<p>Red Neuronal probabilística(PNN) Distancia de Hausdorff BPPN (backpropagation neural network).</p>		
<p>ÁLVAREZ G., DAMIÁN A.; GUEVARA G., MARTA L.; HOLGUÍN L., GERMÁN A. (2006) PREPROCESAMIENTO DE IMÁGENES APLICADAS A MAMOGRAFÍAS DIGITALES</p>	<p>Filtrado de Mediana. Filtrado de Wiener. Ecuación de histograma. Técnica de realce filtro unsharp</p>				<p>La combinación de estos algoritmos arroja mejores resultados que actuando aisladamente.</p>	

c) Implementar algoritmos seleccionados que permitan realizar un análisis efectivo de la imagen.

A. IMÁGENES DE ENTRADA.

Las imágenes evaluadas contaron con el siguiente estándar:

- a. La orientación de los aisladores debe estar de manera vertical.
- b. Las imágenes digitales fueron de fondo homogéneo en algunas fotografías fondo blanco y otras con fondo cielo.

B. PRE – PROCESAMIENTO DE IMÁGENES.

Para el pre-procesamiento de las imágenes de los aisladores eléctricos se aplicó los siguientes algoritmos con el objetivo de tener una mejor segmentación del aislador.

a) Realce de la imagen

En la etapa de preprocesamiento se pasó a aplicar filtros como Blur normalizado y mediana, para eliminar todo el ruido que pueda existir en ellas, además de suavizarlas y mantener un equilibrio uniforme en la imagen, ya que la textura en los aisladores en uniforme ya sean en su estado sucio como en su estado limpio.

Ambos son algoritmos paso bajo, los cuales estos tipos de filtros son utilizados comúnmente porque ayudan a resaltar características basadas en la textura.

Se pasó a aplicar estos dos tipos de filtros con la finalidad de comparar cuál de ellas muestra un mejor resultado en esta investigación.



1) Filtro Blur Normalizado

El filtro blur normalizado, se utilizó por la característica que tiene de promediar y homogeneizar a la imagen, permitiendo que haya una diferencia entre el objeto de interés y el fondo. Por ello se pasó a implementar:

El filtro Blur normalizado o promediado se sustituye cada pixel en $I(x, y)$ de la Imagen, por el promedio (o media) U_{mean} de sus vecinos (ventana o vecindad W con centro en el pixel en cuestión), en su mayoría también por una máscara 3X3. La forma y tamaño de W puede variar de acuerdo al filtro utilizado.

$$U_{in} = I(x, y) ; U_{mean}(x, y) = \frac{\sum I(x', y')}{card(W(x, y))}; I(x', y') \in W(x, y)$$

$W =$ *vecindad alrededor de cada (x, y) , del filtro*

$card(W) =$ *tamaño total del filtro (total de pixeles)*

Sea una matriz $I(x, y)$

100	5	9
8	9	4
1	7	3

$$= U_{mean}(x, y) = \frac{\sum I(100+5+9+8+9+4+1+7+3)}{Size(I(x,y))} = \frac{146}{9} = 16.2$$

El nuevo valor $U_{mean}(x, y)$ se sustituye en la matriz resultante.



Todo lo mostrado anteriormente se ha implementado en el lenguaje de Programación C++ y la librería Opencv de la siguiente forma:

Se empieza leyendo la imagen

```
String imagen = "D:\\imagen_aislador.bmp";
Mat img = imread(imagen,0);
```

//Declaramos variables donde se almacenará los valores obtenidos.

```
float valor, pixel; float total;
Mat aux = Mat::zeros(img.rows, img.cols, img.type());
```

Se recorre la imagen original

```
for (int i = 1; i < img.rows - 1; ++i) {
    for (int j = 1; j < img.cols - 1; ++j) {
```

//Agrupamos solo los 9 primeros valores ya que es un filtro 3X3 para luego sumarlos y dividirlo entre la cantidad total de la matriz

```
valor = img.at<float>(i-1,j-1)+img.at<float>(i-1,j)+
img.at<float>(i-1,j+1) + img.at<float>(i,j-1)+
img.at<float>(i,j) + img.at<float>(i,j+1)+
img.at<float>(i+1,j-1)+img.at<float>(i+1,j)+
img.at<float>(i+1,j+1);
```

```
pixel = valor / 9;
```

//Asignamos el nuevo valor a nueva matriz de salida

```
aux.at<float>(i,j) = valor;
```



a. Resultado



Figura 22: Imagen de Entrada. Fuente: Propia.



Figura 23: Imagen de Salida de Filtro Blur Normalizado. Fuente: Propia.

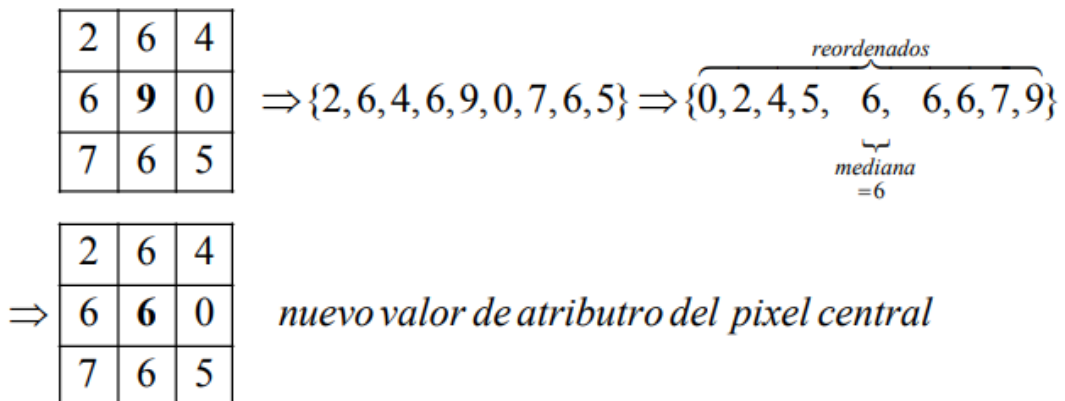
Luego de haber implementado el filtro blur normalizado, la matriz resultante es una imagen con valores promediados con un filtro 3X3, todo esto con el fin de obtener un mejor resultado en las etapas posteriores.

2) Filtro Mediana

El filtro de mediana se aplicó también por la característica que tiene de reducir los puntos de intensidad muy distintas y hacerlas muy parecidas a sus vecinos, esto nos sirve porque elimina pixeles de intensidad muy distintas, con esto se lograría hacer de la textura más uniforme.

El filtro mediana en una imagen consiste en reemplazar el valor central $u=l(x,y)$ por U_{med} , para cada pixel de la imagen. Típicamente W es una vecindad de 3x3 pixeles, con $N=8$ más el centro.





La primera matriz 3 x 3 de la imagen, es originalmente obtenida de la imagen, recorreremos la matriz y transformamos dicha matriz en una lista con los 9 valores obtenidos, para reordenarla de menor a mayor, e identificar el valor central de la imagen del arreglo y almacenarla en una nueva imagen.

Para eso utilizando la herramienta de opencv, lo hacemos de la siguiente forma:

Se empieza leyendo la imagen

```
String imagen = "D:\\imagen_aislador.bmp";
Mat img = imread(imagen,0);
```

Recorremos la imagen original

```
for (int i = 1; i < img.rows - 1; ++i) {
    for (int j = 1; j < img.cols - 1; ++j) {
        //Creamos una lista donde se almacenará la lista ordenada.
        list<float> lista;
        //Agregamos cada valor de la matriz a la lista, solo los 9 primeros valores ya que es un filtro 3X3
        lista.push_back(img.at<float>(i-1,j-1));
        lista.push_back(img.at<float>(i-1, j));
        lista.push_back(img.at<float>(i-1, j+1));
        lista.push_back(img.at<float>(i, j-1));
        lista.push_back(img.at<float>(i, j));
        lista.push_back(img.at<float>(i, j+1));
        lista.push_back(img.at<float>(i+1, j-1));
        lista.push_back(img.at<float>(i+1, j));
```



```

        lista.push_back(img.at<float>(i+1,j+1));
//Obtenida la lista pasamos a ordenar la lista de menor a
mayor
        lista.sort();
//Identificamos el valor central de la lista ya ordenada.
        list<float>::iterator pos = lista.begin();
        pos++; pos++; pos++; pos++;
        valor = *pos;
//Asignamos el valor central en la nueva matriz de salida
        aux.at<float>(i,j) = valor;
    }
}

```

a. Resultado



Figura 24: Imagen de Entrada.

Fuente: Propia.



Figura 25: Imagen de Salida de Filtro

de la Mediana. Fuente: Propia

Al finalizar la etapa de preprocesamiento con el filtro de mediana, se notó un cambio en la imagen, suavizándola y atenuando la imagen para la siguiente etapa.



C. SEGMENTACIÓN

Una vez filtrada las imágenes de los aisladores eléctricos, se pasó a segmentar mediante umbralización, por el motivo de que en todas las imágenes casi siempre tendrán un fondo claro azulino debido a que son tomadas en postes de media tensión y estas están en altura y el fondo es claro y a simple vista se puede ver una gran diferencia entre el objeto de interés y el fondo.

Entre los algoritmos de segmentación por a umbralización encontramos a el algoritmo de Otsu y algoritmo Balu (De la herramienta Balu, Matlab).

1) Otsu

Otsu es un método de segmentación por Umbralización, consiste en calcular el valor del mejor umbral aumentando la varianza entre clases mediante una búsqueda integral, para discriminar entre el objeto de interés y el fondo.

Este algoritmo se adecua a lo que se quiere lograr ya que se quiere analizar solo parte del aislador y no el aislador en su conjunto.

Utilizamos un método simple de segmentación de la imagen

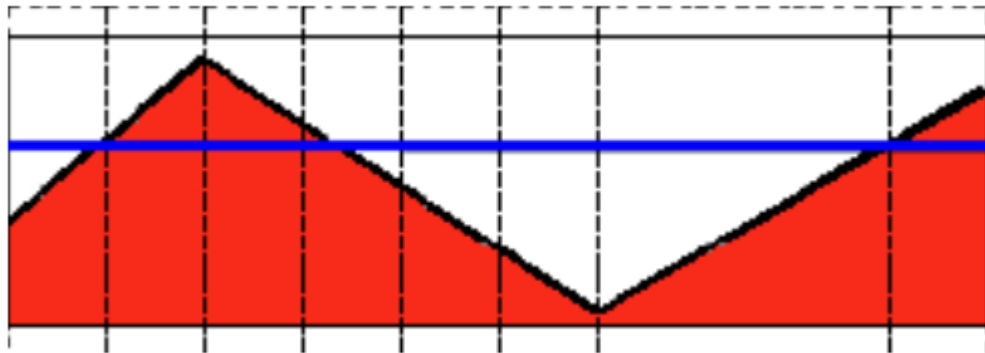
resultante para esto utilizamos umbralización binaria, esta segmentación

se basa en la variación de intensidad entre los pixeles que conforman la imagen del aislador, tanto el aislador propio como el fondo, es decir le asignamos un valor de 0 (negro - fondo) o 255 (blanco - aislador),



La asignación de un pixel a uno de los dos segmentos (0 y 255) se consigue comparando su nivel de gris $g(x, y)$ con un cierto valor umbral preestablecido t o $thresh$ (en inglés *threshold*). La imagen final es muy sencilla de calcular ya que para cada pixel sólo hay que realizar una comparación numérica.

En la ilustración siguiente, la línea azul horizontal representa el umbral (fijo).



Por lo tanto, si la intensidad del píxel es mayor que $thresh$, a continuación, la nueva intensidad de los píxeles se establece en un $MaxVal$ que en este caso es **255**. De lo contrario, los píxeles se establecen en **0**.

$$dst(x, y) = \begin{cases} MaxVal & \text{si } g(x, y) > thresh \\ 0 & \text{otherwise} \end{cases}$$

Opencv, tiene implementado ya este algoritmo Otsu, es por eso que utilizaremos la facilidad que nos ofrece esta valiosa herramienta en el proceso de imágenes.

Los parámetros que se envían son la matriz de entrada, de la cual se quiere umbralizar, la matriz resultante, el valor de sensibilidad, el máximo número de valores y el tipo de umbralización en este caso Binaria.



```
cv::threshold (aux, Otsu, 0, 255, cv::THRESH_BINARY | cv::THRESH_OTSU).
```

a. Resultado



Figura 26: Imagen de entrada. Fuente: Propia.



Figura 27: Imagen segmentada con Otsu. Fuente: Propia.

Una vez obtenida la imagen segmentada, se pasó a extraer sola una muestra de la imagen 180X180, ya que la textura es uniforme en todo el aislador, basta solo con una parte del aislador para determinar si es aislador eléctrico está sucio o limpio, en opencv se hizo de la siguiente manera:

```
Mat sacarMuestra (Mat img_input) {
    Rect rec (180,180,80,80);
    Rectangle (img_input, rec, Scalar (255),1,8,0);
    Mat Roi = img_input(rec);
    return Roi;
}
```



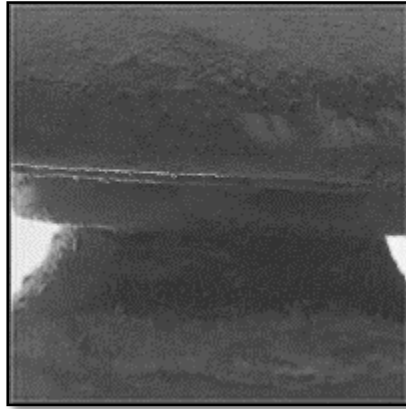


Figura 28: Muestra extraída de 180X180. Fuente propia.

La muestra que se extrajo fue de 180X180, debido a que si es más pequeña pierde detalles, en este tamaño se puede determinar la textura del aislador.

D. Extracción de características

Entre los algoritmos, extractores de características existen algoritmos implementados ya, que se encargan de extraer características basados en textura, entre ellos tenemos a los filtros de Gabor y Patrones locales binarios, (LBP), estos según las investigaciones documentadas en la Tabla 13, “Selección de algoritmos de extracción basados en textura”, muestran resultados muy aceptables.

Cada uno de los algoritmos de extracción, lo que hacen es manipular los valores de los píxeles de las imágenes, para así poder detectar mejor la característica de textura. Luego buscamos la manera de poder representar cada imagen con uno o más valores que diferencien una imagen de un aislador sucio y limpio. Por ejemplo, con el algoritmo Gabor, usamos el conteo de contornos de cada imagen y con el algoritmo LBP, usamos la cantidad de valores de los píxeles comprendidos entre ciertos rangos.

1. Filtros de Gabor

El algoritmo de Gabor, o filtros de Gabor son filtros pasa banda que al convolucionar con la imagen a tratar preserva direcciones de la imagen original, entre las cuales se puede buscar mediante la orientación y la frecuencia del kernel.

El filtrado de una imagen con funciones de Gabor, se relaciona con los procesos en la corteza visual.

1. Para ello la imagen de entrada se convoluciona con un kernel o filtro con simetría par esta dado por la siguiente ecuación:

$$f(x, y) = \exp \left[-\frac{1}{2} \left[\frac{x_r^2 + y_r^2}{\sigma^2} \right] \right] \cos \left(\frac{2\pi}{\lambda} x_r + \Psi \right); \text{ donde:}$$

x = Ancho máximo de la mascara de Gabor.

y = Altura máxima de la mascara de Gabor.

$$x_r = x * \cos \theta + y * \sin \theta$$

$$y_r = -x * \sin \theta + y * \cos \theta$$

σ = Sigma o varianza de la funcion gaussiana

λ = Longitud de Onda o frecuencia

Ψ = El desfase

Con la herramienta opencv y el lenguaje c++ lo implementamos de la siguiente manera:



Establecemos el ancho y largo de la máscara de Gabor, la orientación(θ), la frecuencia λ (Lambda), función gaussiana σ (sigma), y desfase Ψ (psi).

Los valores que se utilizaron para generar el filtro de Gabor y de los umbrales fueron encontrados a base de prueba y error.

```
int kernel_size=21;
int pos_sigma= 3;
int pos_lm = 4;
int pos_th = 45;
int pos_psi = 74;
cv::Mat src_f;
cv::Mat dest;
```

Se crea el kernel con la formula anterior

```
Mat crearKernel (int ks, double sig, double th, double lm,
double ps) {
    int hks = (ks-1) /2;
    double theta = th * CV_PI/180;//
    double psi = ps * CV_PI/180; //el desfase
    double del = 2.0/(ks-1);
    double lmbd = lm;
    double sigma = sig/ks;
    double x_theta;
    double y_theta;
    Mat kernel (ks, ks, CV_32F);
    for (int y =-hks; y<=hks; y++) {
        for (int x=-hks; x<=hks; x++) {
            x_theta = x*del*cos(theta)+y*del*sin(theta);
            y_theta = -x*del*sin(theta)+y*del*cos(theta);
            kernel.at<float>(hks+y, hks+x) = (float)exp(-
            0.5*(pow(x_theta,2)+pow(y_theta,2))/pow(sigma,2)) *
            cos(2*CV_PI*x_theta/lmbd + psi);
        }
    }
    return kernel;
}
```

a. Resultado



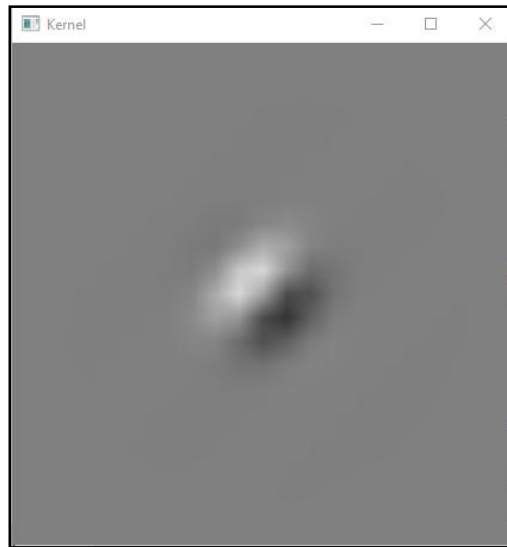


Figura 29: Máscara de Gabor de tamaño 21x21, varianza de la función gaussiana = 3, frecuencia = 4, orientación = 45°, desfase = 74

La representación de la transformada Gabor de una imagen, es la convolución de la imagen con el banco de filtros Gabor de la formula anterior. Aplicando el teorema de convolución, la salida del filtro Gabor de la imagen por medio de la transformada de Fourier se define de la siguiente manera

$$\mathfrak{F}\{\theta_{f,\theta}(x,y)\} = \mathfrak{F}^{-1}\{\mathfrak{F}\{I(x,y)\}\mathfrak{F}\{\Psi_{f,\theta}(x,y)\}\}$$

donde: $\theta_{f,\theta}(x,y)$ = es la imagen filtrada, $I(x,y)$ = imagen original ;

$\Psi_{f,\theta}(x,y)$ = es la onduleta de Gabor o el kernel;

$\mathfrak{F}^{-1}, \mathfrak{F}$ = transformada inversa de fourier y la transformada de Fourier

Vemos que al haber creado el kernel se debe convolucionar con la imagen original para poder así filtrar características de textura, en opencv lo realizamos de la siguiente manera:

```
cv::filter2D(src_f, dest, CV_32F, kernel);
return dest;
```



b. Resultado

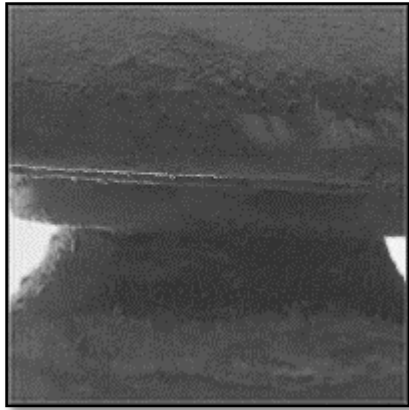


Figura 30: Imagen de Entrada 180 x 180. Fuente: Propia.

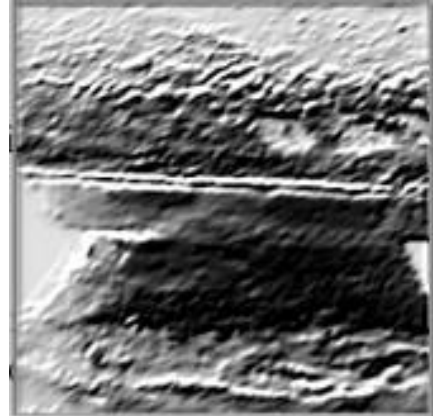


Figura 31: Imagen resultante con algoritmo Gabor. Fuente: Propia.

Luego de aplicar el filtro de Gabor y convolucionar la fig. 30 y el kernel, como resultado nos mostró una imagen en blanco y negro Fig. 31, resaltando parte de los bordes de la textura del aislador. Luego de aplicar el filtro de Gabor, para poder diferenciar un aislador sucio de un aislador limpio.

Una vez filtrada la imagen se pasó a contar los contornos que tiene cada imagen para luego clasificarlos según la cantidad de cada uno, para ello se hizo un análisis mediante observación y se pudo notar lo siguiente:

- 1) Los aisladores sucios tenían una cantidad de valores menor o igual a 31.
- 2) Si el valor era mayor a 31, las imágenes eran de un aislador limpio (con un poco de margen de error) como veremos en la tabla:



Tabla 15: Análisis de las características de las imágenes de aisladores limpios y sucios en el algoritmo Gabor. Fuente: Propia.

	Sucios	Limpios
	14	34
	21	29
	10	37
	28	29
	27	46
	14	9
	24	16
	21	36
	23	43
	20	18
	28	43
	18	56
	20	41
	14	33
	35	39
	21	17
	62	20
	28	9
	26	21
	35	50



	20	
	22	
	21	
	23	
	83	
	61	
	44	
	32	
	40	
	42	
Promedio	29.2333333	31.3
Max	83	56
Min	10	9
Menores o iguales a	21	11
31		

En la tabla podemos apreciar que, bajo este análisis de 30 aisladores sucios, 21 tienen valores de conteo de contornos menores o iguales a 31; y de los 20 aisladores limpios, 11 tienen valores de conteo de contornos mayores a 31.

En la etapa de entrenamiento estos valores del conteo de contornos son los que se guardan en la matriz, que luego serán cargados en la etapa de clasificación.



Con ayuda de la función `findContours` de `Opencv`, se pudo implementar lo anteriormente explicado:

```

findContours(dest1, contornos, hierarchy, CV_RETR_CCOMP,
CV_CHAIN_APPROX_SIMPLE );

    if(contornos.size()<=31) {
        aisladores1.at<float>(i,0) =10;

        cout<<"Sucio"<<endl;
        msg_cm=msg_cm + "<br> El aislador esta
sucio";

    } else {
        aisladores1.at<float>(i,0) =100;

        cout<<"Limpio"<<endl;
        msg_cm=msg_cm + "<br> El aislador esta
limpio";

    }

```

Luego de hacer el conteo de contornos de la etapa anterior, se guardó el resultado en un archivo `txt` (Fig. 32.), para su utilización en la etapa de entrenamiento. En la primera fila del `txt`, columna 1, se muestra la cantidad de imágenes que se extrajeron características en este caso 150, a partir de la fila 2, columna 1, se muestra la primera imagen de la cual se extrajo 16 contornos, y así sucesivamente de cada una de las imágenes.



FPA_Gabor.txt: Bloc de notas

Archivo	Edición	Formato	Ver	Ayuda
150	1			
16	0			
24	0			
20	0			
29	0			
45	0			
50	0			
20	0			
16	0			
16	0			
35	0			
27	0			
21	0			
14	0			
20	0			
18	0			
20	0			
16	0			
6	0			
27	0			
26	0			
27	0			
23	0			
10	0			
6	0			

Figura 32: Las imágenes con cantidad de contornos (Gabor). Fuente: Propia.



2. Algoritmo Local Patern Binary

El algoritmo de LBP, consiste en la comparación de pixel central con los vecinos, en el cual el pixel central es tomado como el umbral con respecto a sus vecinos, al realizar la comparación del pixel central con el vecino se le asigna un valor de uno "1" si el vecino el mayor o igual, en caso contrario se le asigna un valor de cero "0", a cada resultado del umbral se le asigna un peso de 2^n , en donde "n" depende de la posición del vecino con respecto al pixel central, finalmente se realiza una suma de los diferentes pesos obteniendo la representación LBP de pixel, cabe mencionar que LBP trabaja con imágenes en escala de grises, por ello debemos transformarlas, ya que trabaja con valores entre 0 a 255.

$$LBP_{P,R}(x_c, y_c) = \sum_{P=0}^{P-1} s(g_p - g_c) 2^P$$

(x_c, y_c) = posición del pixel de interes de la imagen

g_p = Es el valor del pixel central

g_c = Es el valor de los vecinos circulares del pixel central

2^P

= El peso asignado a cada operacion entre el pixel central y el vecino

$g_p - g_c$ = Se asignan valores de 1 a 0 segun la siguiente ecuacion:

$$s(g_p - g_c) = \begin{cases} 1; & g_p - g_c \geq 0 \\ 0; & g_p - g_c < 0 \end{cases}$$



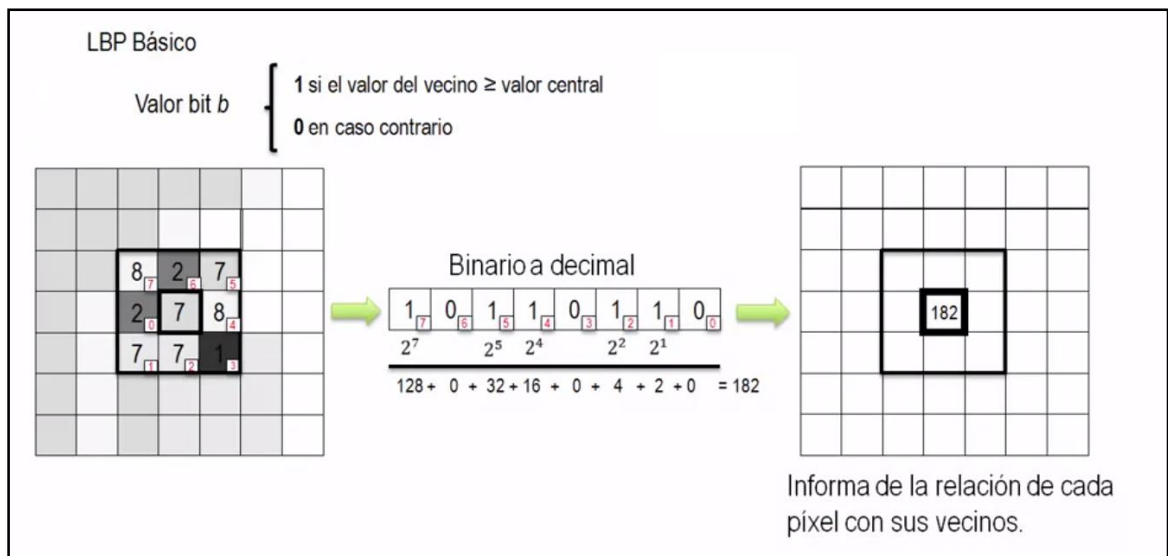


Figura 33: a) Aplicación de LBP. Fuente Universidad Autónoma de Barcelona

Para encontrar los valores de los bordes de la imagen, se coloca la matriz a recorrer en la esquina y se les asignan valores similares a sus vecinos, en otros casos se obvian los valores de los bordes de la imagen y se empieza calculando a partir de la segunda fila.

Con ayuda de opencv implementamos lo anterior de la siguiente manera:

```
//Declaramos las variables donde se almacenar el valor después de aplicar lbp.
```

```
int vecULBP[8], val=0;
cv::cvtColor(otsu, grisLBP, CV_BGR2GRAY);
Mat mat_extra(grisLBP.rows, grisLBP.cols, CV_8UC1);
```



//Recorremos la imagen evaluando si los valores vecinos;
1 si el vecino es mayor al pixel central y 0 en caso
contrario.

```

for(int x=0;x<grisLBP.rows;x++){
    for(int y=0;y<grisLBP.cols;y++){
        try{
            if((int)grisLBP.at<uchar>(x+1,
y)>=(int)grisLBP.at<uchar>(x,y)){
                vecULBP[0]={1};
            }else{
                vecULBP[0]={0};
            }
            if((int)grisLBP.at<uchar>(x+1, y-
1)>=(int)grisLBP.at<uchar>(x,y)){
                vecULBP[1]={1};
            } else {
                vecULBP[1]={0};
            }
            if((int)grisLBP.at<uchar>(x, y-
1)>=(int)grisLBP.at<uchar>(x,y)){
                vecULBP[2]={1};
            } else {
                vecULBP[2]={0};
            }
            if((int)grisLBP.at<uchar>(x-1, y-
1)>=(int)grisLBP.at<uchar>(x,y)){
                vecULBP[3]={1};
            } else {
                vecULBP[3]={0};
            }
            if((int)grisLBP.at<uchar>(x-1,
y)>=(int)grisLBP.at<uchar>(x,y)){
                vecULBP[4]={1};
            } else {
                vecULBP[4]={0};
            }
            if((int)grisLBP.at<uchar>(x-1,
y+1)>=(int)grisLBP.at<uchar>(x,y)){
                vecULBP[5]={1};
            } else {
                vecULBP[5]={0};
            }
            if((int)grisLBP.at<uchar>(x, y+1)>=(int)grisLBP.at<uchar>(x,y)){
                vecULBP[6]={1};
            } else {
                vecULBP[6]={0};
            }
            if((int)grisLBP.at<uchar>(x+1,
y+1)>=(int)grisLBP.at<uchar>(x,y)){

```



```

        vecULBP[7]={1};
    } else {
        vecULBP[7]={0};
    }
} catch(int e){

    cv::imshow("Error",e);
}
//Una vez etiquetado el vector con 0 y 1 con respecto a
la matriz, se eleva el 2 según el peso establecido, del
vector y se suman los valores para obtener el nuevo
valor

for(int i=0;i<8;i++){
    val=val+(vecULBP[i]*(pow(2,i)));
}

mat_extra.at<uchar>(x,y) = val;

//Se inicializan los valores del vector a 0 para la
siguiente iteración.
vecULBP[0]={0};
vecULBP[1]={0};
vecULBP[2]={0};
vecULBP[3]={0};
vecULBP[4]={0};
vecULBP[5]={0};
vecULBP[6]={0};
vecULBP[7]={0};
val=0;
}
}

```

a. Resultado



Figura 34: Imagen de Entrada 180 x 180 - Muestra Paso Bajo. Fuente: Propia.



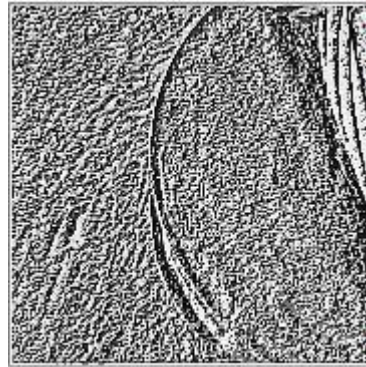


Figura 35: Matriz resultante con algoritmo LBP. Fuente: Propia.

Una vez de haber aplicado LBP, para diferenciar imágenes de aisladores sucios de limpios con el algoritmo LBP, mediante un análisis pudimos apreciar que los aisladores sucios tienen mayor cantidad de valores de pixeles comprendidas entre 240 a 254 y los limpios tienen mayor cantidad de valores entre el rango de 112 a 127; así como lo muestra la tabla del anexo 4.

Esto se implemento opencv de la siguiente manera

```
int m=0, n=0;

for (int k=0; k<sal_extra.rows; k++) {
    for (int j=0; j<sal_extra.cols; j++) {
        if(112<=(int)sal_extra.at<uchar>(k,
            j)&&(int)sal_extra.at<uchar>(k,j)<=127) {
            m++;
        }
        if(240<=(int)sal_extra.at<uchar>(k,
            j)&&(int)sal_extra.at<uchar>(k,j)<=254) {
            n++;
        }
    }
}
```

Luego de haber evaluado cada imagen se pasó a guardar en un txt para su posterior utilización en la etapa de entrenamiento, la matriz tiene 2 valores, el primer valor es la cantidad de valores comprendidos entre 112 a 127 y el



segundo valor es la cantidad de valores comprendidos entre 240 a 254. Así mismo, si el primer valor es mayor que el segundo, el aislador sería limpio, de lo contrario sería un aislador sucio.

Archivo	Edición	Formato	Ver	Ayuda
150	2			
2297	4393			
3046	3003			
2421	2833			
3614	3175			
2341	3015			
3839	3083			
2347	2836			
2377	4395			
2377	4395			
2536	5328			
2365	4830			
2103	3827			
2376	4687			
2344	2581			
2304	2800			
2348	2836			
2378	4395			
2689	4166			
2366	4830			
2355	4286			
2366	4830			
2487	4221			
3807	3858			
2689	4166			
3863	3265			

Figura 36: Matriz de entrada (LBP). Fuente: Propia.



D) CLASIFICACIÓN

Cada algoritmo clasificador recibe para la etapa de clasificación, los archivos .txt de donde se originarán una matriz con valores de características que originaron los algoritmos de extracción en la etapa de entrenamiento, una matriz de etiquetas cuando las imágenes son de aisladores sucios y limpios; y una matriz de las características de las imágenes a clasificar. Todas estas matrices son las que reciben los algoritmos de clasificación para la etapa de clasificación, obteniendo como resultado las etiquetas que podemos interpretar como aisladores limpios y sucios.

Las imágenes que sirvieron de entrenamiento y prueba para cada uno de los clasificadores, en esta investigación son los de la Fig. 14 (Imágenes de Entrenamiento), y las imágenes de prueba son los de la Fig. 15 (Imágenes de aisladores limpios) y Fig. 16 (Imágenes de aisladores sucios).

1. Algoritmo de KNN

Este algoritmo clasifica cada dato nuevo en el grupo que corresponda, según tenga k vecinos más cerca de un grupo o de otro. Es decir, calcula la distancia del elemento nuevo a cada uno de los existentes, y ordena dichas distancias de menor a mayor para ir seleccionando el grupo al que pertenecer. Este grupo será, por tanto, el de mayor frecuencia con menores distancias.



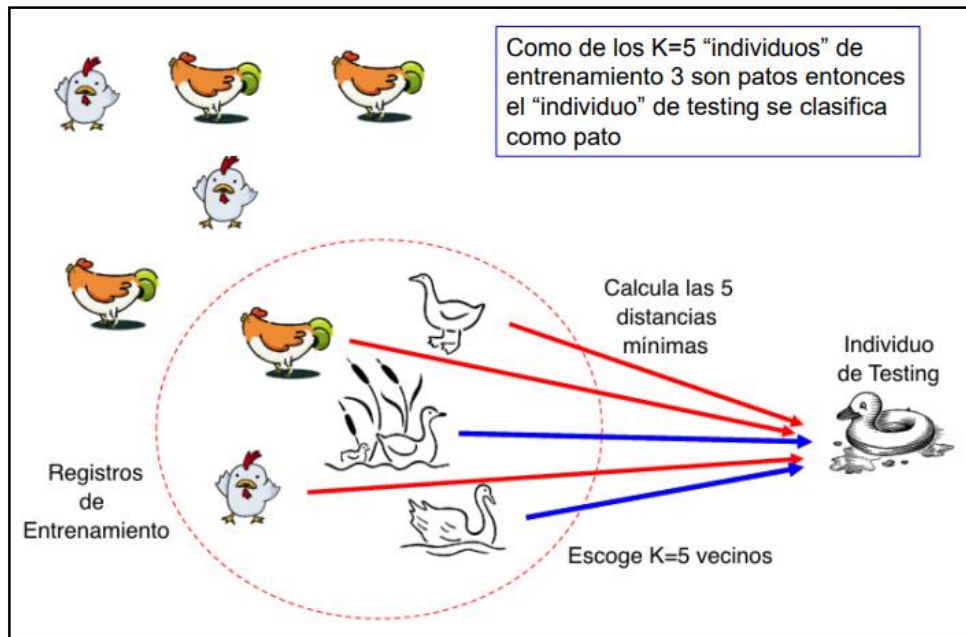


Figura 37: Knn, evalúa sus vecinos más cercanos

Para ello debe realizar lo siguiente:

- 1) Calcular la distancia mínima entre el ítem a clasificar y el resto de ítems del dataset de entrenamiento.

$$d(A, B) = \sqrt{\sum_{j=1}^n (A_j - B_j)^2}$$

- 2) Seleccionar los "k" elementos más cercanos (con menor distancia, según la función que se use)
- 3) Realizar una "votación de mayoría" entre los k puntos: los de una clase/etiqueta que <<dominen>> decidirán su clasificación final.



En opencv el algoritmo de clasificación ya esta implementado.

```
//Inicializamos la variable knn;

    Ptr<KNearest> knn = KNearest::create();
    Ptr<TrainData> trainingData;

//Establecemos los parámetros
    knn->setIsClassifier(true);
    knn-
>setAlgorithmType(KNearest::Types::BRUTE_FORCE);
    knn->setDefaultK(1);

//Entrenamos con la data almacenada en una matriz de los
txt anteriores

    trainingData = TrainData::create(trainData1,
    SampleTypes::ROW_SAMPLE, etiquetas);

    knn->train(trainingData);
    int limite = lista.size();
    Mat matResults(0,0, CV_32F);
    for (int i = 0; i < limite; i++) {
        float r=0;

//Encontramos el vecino más cercano con k=1
        r = knn->findNearest(aisladores1.row(i), knn-
>getDefaultK(), matResults);

//Clasificamos según las etiquetas dadas
        if(r == 1){
            cout<<"Resultado:"<<r<<" El Aislador esta
sucio"<<endl;
            sucios++;
        }
        if(r == 0){
            cout<<"Resultado:"<<r<<" El Aislador
esta limpio"<<endl;
            limpios++;
        }
    }
}
```



a. Resultado

Los resultados de cada clasificación Knn se encuentran en el apartado Capítulo IV: Análisis e interpretación de los resultados

2. Máquinas de Vectores Soporte (SVM)

Consiste en construir un hiperplano en un espacio de dimensionalidad muy alta (o incluso infinita) que separe las clases que tenemos. Una buena separación entre las clases permitirá una clasificación correcta de la nueva muestra, es decir; necesitamos encontrar la máxima separación a los puntos más cercanos a este hiperplano.

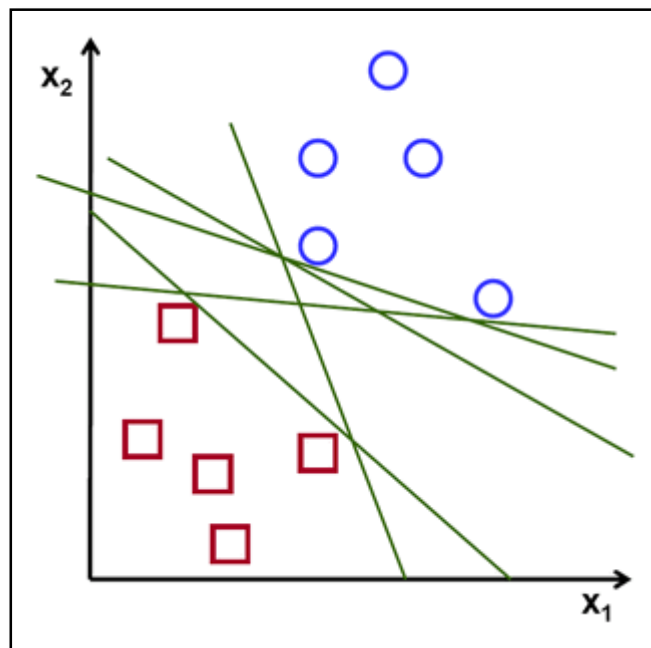


Figura 38: Caso linealmente separable. Fuente Opencv

En la imagen anterior se ven varias líneas que ofrecen una solución al problema. ¿Alguno de ellos es mejor que los demás? Se define que una



línea es mala si pasa demasiado cerca de los puntos porque será sensible al ruido y no se generalizará correctamente. Por lo tanto, nuestro objetivo debe ser encontrar la línea que pasa lo más lejos posible de todos los puntos.

La operación del algoritmo SVM se basa en encontrar el hiperplano que proporciona la mayor distancia mínima a los ejemplos de entrenamiento. Dos veces, esta distancia recibe el nombre importante de margen dentro de la teoría de SVM. Por lo tanto, el hiperplano de separación óptimo maximiza el margen de los datos de entrenamiento.

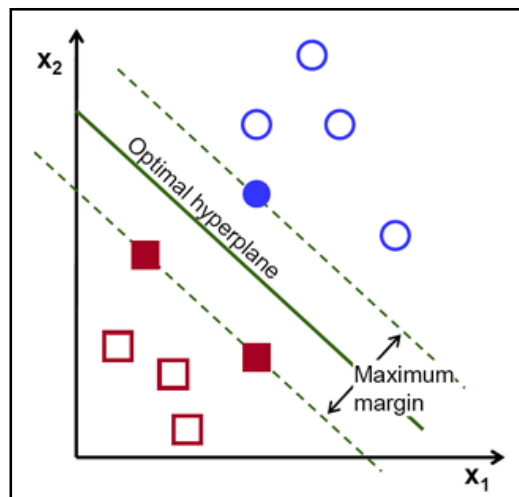


Figura 39: Búsqueda del hiperplano óptimo para maximizar el margen. Fuente Opencv

La búsqueda del hiperplano óptimo para maximizar el margen esta dada por la siguiente

$$L_B = \frac{1}{2} \|B\|^2 ; \text{suje}to \ a \ y_i(B^T x_i + B_0) \geq 1 \forall i$$

Donde: y_i = etiquetas de entrenamiento

B_0 = Como el sesgo

B = vector de peso



$$y_i = \begin{cases} 1 & \text{si } X \in \text{Clase 1} \\ -1 & \text{si } X \in \text{Clase 2} \end{cases}$$

En la presente investigación haremos uso de clasificación binaria de svm por la razón de que tendremos 2 clases una limpia y otra sucia.

Opencv nos facilita la implementación de svm, en este caso haremos, con un tipo de kernel lineal:

```
//Se inicializa la variable de svm
Ptr<SVM> svm = SVM::create();
//Establecemos los parámetros de tipo kernel lineal,
ya que solo contamos con 2 clases
svm->setType(SVM::C_SVC);
svm->setKernel(SVM::LINEAR);
svm-
>setTermCriteria(TermCriteria(TermCriteria:MAX_ITER,10
0,1e-6));

//Entrenamos el clasificador svm con la data
almacenada en una matriz de los txt anteriores.

svm->train(trainData1, ROW_SAMPLE,etiquetas);
int limite = lista.size();
for (int i = 0; i < limite; i++) {
    float r=0;

//Se predice el nuevo valor una vez entrenada
    r = svm->predict(aisladores1.row(i));
    if(r == -1){
        cout<<"Resultado: "<<r<<" El
Aislador esta sucio"<<endl;
        sucios++;
    }
    if(r == 1){
        cout<<"Resultado: "<<r<<" El
Aislador esta limpio"<<endl;
    }
}
}
```



a. Resultado

Los resultados de la clasificación SVM se encuentran en el apartado

Capítulo IV: Análisis e interpretación de los resultados

3. Red neuronal probabilística(RNP)

Una red neuronal consta de la capa de entrada, la capa de salida y una o más capas ocultas. Cada capa incluye una o más neuronas vinculadas direccionalmente con las neuronas de la capa anterior y la siguiente.

El siguiente ejemplo representa un perceptrón de 3 capas con tres entradas, dos salidas y la capa oculta que incluye cinco neuronas:

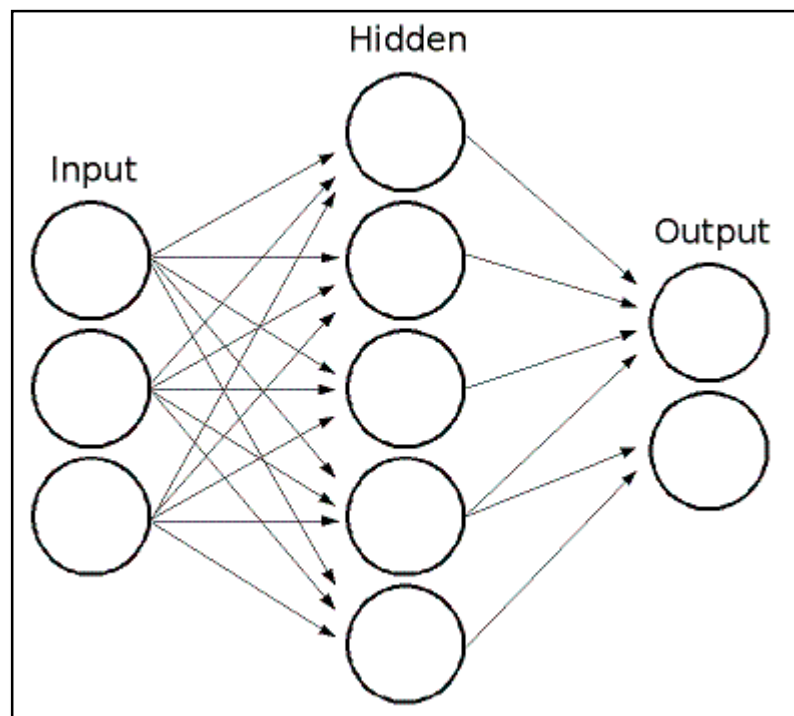


Figura 40: Grafico que representa una neurona, la cual tiene datos de entrada, capas ocultas y el dato de salida. Fuente Opencv



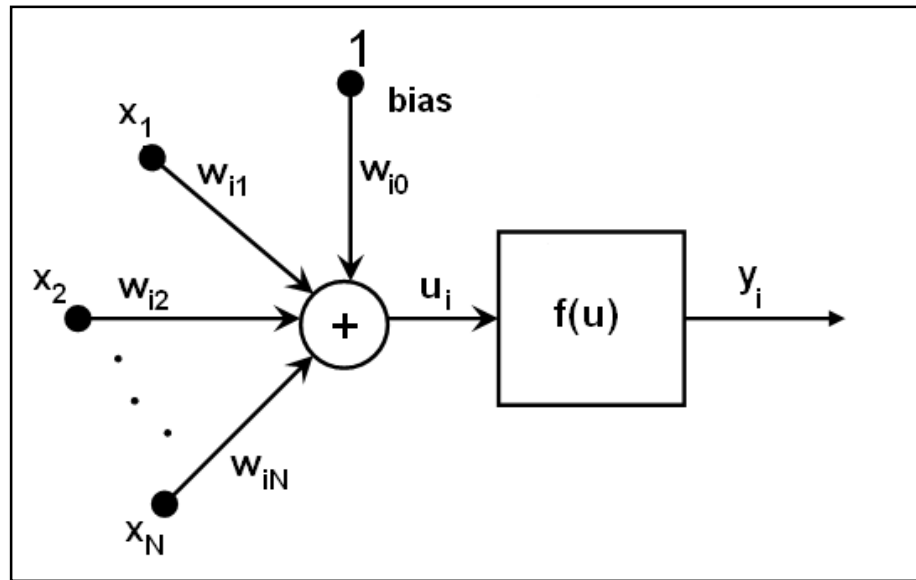


Figura 41: Red neuronal

Dados los resultados x_j de la capa n , los resultados y_i de la capa $n + 1$ se calculan como:

$$U_i = \sum (w_{ij}^{n+1} * x_j) + w_{i,bias}^{n+1}$$

$$y_i = f(U_i)$$

Entonces, toda la red entrenada funciona de la siguiente manera:

- 1) Tome el vector de características como entrada. El tamaño del vector es igual al tamaño de la capa de entrada.
- 2) Pasa valores como entrada a la primera capa oculta.
- 3) Calcule las salidas de la capa oculta utilizando los pesos y las funciones de activación.
- 4) Pase las salidas más hacia abajo hasta que calcule la capa de salida.

Para calcular la red, necesita conocer todos los pesos $w_{i,j}^{n+1}$. Los pesos son calculados por el algoritmo de entrenamiento. El algoritmo toma un conjunto de entrenamiento, múltiples vectores de entrada con los



vectores de salida correspondientes, y ajusta de manera iterativa los pesos para permitir que la red brinde la respuesta deseada a los vectores de entrada proporcionados.

Para hacer uso de este algoritmo, usamos la función ANN_MLP de la librería OpenCV. Los parámetros que tenemos son hiddenLayerSize, que nos indica el número de capas; setActivationFunction, que nos indica la función de activación para cada neurona, así como también acepta los 2 parámetros para la ecuación; setTrainMethod, nos indica el modo de entrenamiento BACKPROP o RPROP; y por último TermCriteria, que nos indica el número de interacciones y la tolerancia de error que tendrá el algoritmo. Al igual que los otros algoritmos, también necesitaremos las matrices de entrenamiento y clasificación de las imágenes

```
//Definimos el número de neuronas en cada capa,
incluidas las capas de entrada y salida.
```

```
const int hiddenLayerSize = 100;

Ptr<ANN_MLP> rn = ANN_MLP::create();
Mat layersSize = Mat(3, 1, CV_16U);

layersSize.row(0) = Scalar(trainData1.cols);
layersSize.row(1) = Scalar(hiddenLayerSize);
layersSize.row(2) = Scalar(etiquetas1.cols);

rn->setLayerSizes(layersSize);

//Establecemos la función de activación tipo sigmoideal.
 $f(x) = \beta * (1 - e^{-\alpha x}) / (1 + e^{-\alpha x})$ , El sigmoide estándar
con  $\beta = 1, \alpha = 1$ .
rn-
>setActivationFunction(ANN_MLP::ActivationFunctions::SI
GMOID_SYM, 1.0, 1.0);

rn->setTrainMethod(ANN_MLP::TrainingMethods::BACKPROP,
0.0001);
```



```
//Terminar el entrenamiento después de 100
iteraciones.

    TermCriteria termCrit = TermCriteria(TermCriteria
:: MAX_ITER + TermCriteria :: EPS , 100, 0.000001);

    rn->setTermCriteria(termCrit);

//Creamos la set de datos de entrenamiento.

Ptr<TrainData> trainingData =
TrainData::create(trainData1, SampleTypes::ROW_SAMPLE,
etiquetas1);

//Entrenamos la red neuronal con el set de datos.

rn->train(trainingData
, ANN_MLP::TrainFlags:NO_INPUT_SCALE +
ANN_MLP::TrainFlags:NO_OUTPUT_SCALE
);

//Predecimos un nuevo dato.

for (int i = 0; i < aisladores1.rows;i++) {
    float r=0;
    Mat result;
    rn->predict(aisladores1.row(i), result);
    r=result.at<float>(0,0);
    int pro = r*100000000;
    if(pro < 0){ //Representa que el aislador esta sucio
        cout<<pro<<" Resultado: El Aislador esta
sucio"<<endl;
        sucios++;
    }
    if(pro > 0){//Representa que el aislador esta limpio
        cout<<pro<<" Resultado: El Aislador esta
limpio"<<endl;
        limpios++;
    }
}
}
```

a. Resultado

Los resultados de la clasificación RNP se encuentran en el apartado Capítulo IV: Análisis e interpretación de los resultados



d) Evaluar la efectividad de los algoritmos.

El desarrollo de este objetivo lo apreciamos en el Capítulo IV Análisis e Interpretación de los resultados.

e) Costear desarrollo del proyecto.

Costo del proyecto

Tabla 16: Costo del Proyecto. Fuente: Propia.

PROYECTO				
RUBRO	CANTIDAD	PRECIO	COSTO TOTAL	
			FIJO	VARIABLE
ACTIVO FIJO				
Maquinarias y equipos				
Laptop	2	S/. 3000.00	S/. 6000.00	
Cámara fotográfica	1	S/. 2200.00	S/. 2200.00	
Total, Maquinaria y Equipo			S/. 8200.00	
Herramientas				
Licencia de software	1	S/. 1,170.00	S/. 1,170.00	
Total, Herramientas			S/. 1,170.00	
TOTAL, ACTIVO FIJO				
			S/. 9,370.00	



Tabla 17: Costo de desarrollo del proyecto. Fuente: Propia.

DESARROLLO DE PROYECTO				
RUBRO	CANTIDAD	PRECIO	COSTO TOTAL	
			FIJO	VARIABLE
CAPITAL DE TRABAJO				
Materia prima e insumos				
Papel Bond A-4 2 millares	1	S/.14.00	S/.14.00	
Lapiceros	3	S/.2.00	S/.6.00	
Memorias USB Kinston 16 Gb	2	S/.30.00	S/.60.00	
TOTAL MATERIA PRIMA			S/. 80.00	
Mano de obra				
Investigador especialista del tema	2	S/. 1700.00	S/. 3,400.00	
Total mano de obra			S/. 3,400.00	
TOTAL CAPITAL TRABAJO			S/. 3,483.00	
COSTOS INDIRECTOS				
Gastos Administrativos				
Varios (tinta, papel, lapiceros, otros)	1	S/. 250.00	S/. 250.00	
Luz, teléfono transporte, internet.	2	S/. 800.00	S/. 1,600.00	
Total, gastos Administrativos			S/. 1,850.00	
TOTAL, COSTOS INDIRECTOS			S/. 1,850.00	
TOTAL			S/. 5,330.00	



Costos de inversión o de desarrollo

Tabla 18: Costo de Recursos Humanos. Fuente: Propia.

RECURSOS HUMANOS			
Cargos	PAGO POR DIA (S/.)	DÍAS DE TRABAJO	TOTAL(S/.)
Investigador 1	S/. 40.00	80	S/. 3,200.00
Investigador 2	S/. 40.00	80	S/. 3,200.00
Camarógrafa	S/. 50.00	1	S/. 50.00
TOTAL			S/. 6,450.00

Tabla 19: Costo de Útiles de Escritorio. Fuente: Propia.

ÚTILES DE ESCRITORIO			
Descripción	Cantidad	Precio Unitario (S/.)	TOTAL (S/.)
Papel Bond A-4 2 millares	1	S/. 14.00	S/. 14.00
Lapiceros	3	S/. 2.00	S/. 6.00
Memorias USB Kinston 16 Gb	2	S/. 30.00	S/. 60.00
TOTAL			S/. 80.00



Tabla 20: Costos de Hardware. Fuente: Propia.

HARDWARE		
Nombre	Descripción	Precio (S/.)
Computadora Asus Core i5	Procesador: Intel, 8GB DDR3 L Memory Sistema de 64 bits Disco duro de 1 TB Adquisición: compra	S/. 3,500.00
Laptop Asus Core i7	Procesador: Intel, 12GB DDR3 L Memory Sistema de 64 bits Disco duro de 1TB Adquisición: compra	S/. 4,000.00
TOTAL		S/. 7,500.00



Tabla 21: Depreciación de Hardware. Fuente: Propia.

DEPRECIACION			
	Costo (S/.)	Depreciación (%)	Total (S/.)
Hardware	S/. 7,500.00	25%	S/. 1,875.00
Total			S/. 1,875.00

Tabla 22: Costos de Licencias de Software. Fuente: Propia.

LICENCIAS DE SOFTWARE		
Nombre	Descripción	Precio
Matlab R2014a	Herramienta de Software Matemático	S/. 1,170.00
Microsoft Office	Paquete de ofimática	S/. 489.00
Windows 10	Sistema Operativo	S/. 1,010.00
TOTAL		S/. 2,669.00



Tabla 23: Costos de Materiales de Escritorio. Fuente: Propia.

MATERIALES DE ESCRITORIO	
	TOTAL(S/.)
Varios (tinta, papel, lapiceros, otros)	S/. 250.00
TOTAL	S/. 250.00

INVERSION TOTAL

Tabla 24: Inversión Total. Fuente: Propia.

TOTAL DE INVERSIÓN	
RECURSOS HUMANOS	S/. 6,450.00
HARDWARE	S/. 7,500.00
ÚTILES DE ESCRITORIO	S/. 80.00
LICENCIAS DE SOFTWARE	S/. 2,669.00
MATERIALES DE ESCRITORIO	S/. 250.00
TOTAL DE INVERSIÓN	S/. 16,949.00

Beneficios

Estos costos son la limpieza que se hacen por aislador. Así también, se investigó que, un poste cuenta con 3 de estos aisladores, que cada técnico



limpia 12 aisladores, que existen 24 técnicos por día, y que el trabajo de limpieza dura 3 días. Fuente: Electronorte.

Tabla 25: Beneficios. Fuente: Electronorte.

BENEFICIOS			
Mano de Obra			
	Cantidad	Precio Neto	Costo Unitario Comercial
Limpiar Aislador Pin o Suspensión	1	S/. 5.70	S/. 7.40
Total de Mano de Obra		S/. 5.70	S/. 7.40
Materiales			
	Cantidad	Valor Material	Valor Comercial
Waype Paño Fino	100g	S/. 1	S/. 1.50
Silicona Industrial	1	S/. 3.23	S/. 3.38
Total, de Materiales		S/. 4.23	S/. 4.88
TOTAL, DE BENEFICIOS		S/. 9.93	S/. 12.28

Es así que el resultado S/. 609.10 es por un aislador, se sacará el costo total por el trabajo.

$$S/. 12.28 \times 3 \text{ aisladores en un poste} = S/. 36.84$$

$$S/. 36.84 \times 12 \text{ postes por técnico} = S/. 442.08$$



S/. 442.08 x 24 técnicos por jornada = S/. 10 609.92

S/. 10 609.92 x 3 días de trabajo = **S/. 31 829.76**

El resultado final de beneficios es de S/. 31 829.76

CÁLCULO DEL VALOR PRESENTE NETO (VPN)

$$VPN = \sum_{i=0}^{n=1} \frac{Bi - Cj}{\left(1 + \frac{I}{100}\right)^n}$$

Donde:

n = Vida útil en años (5 años).

I = Tasa de interés anual en el tiempo (19%, Tasas de interés establecidos por los bancos. Fuente: SBS).

Ci = Costos (de Inversión S/. 16,949.00)

Costos de operación = S/. 8,575.00 (Sumatoria de recursos humanos, materiales y depreciación de hardware).

Bi = Beneficios (S/. 31 829.76).

$$VPN = -16949 + \frac{31829.76 - 8575}{(1+0.19)^1} + \frac{31829.76 - 8575}{(1+0.19)^2} + \frac{31829.76 - 8575}{(1+0.19)^3} + \frac{31829.76 - 8575}{(1+0.19)^4} + \frac{31829.76 - 8575}{(1+0.19)^5}$$

$$VPN = -16949 + \frac{23254.76}{1.19} + \frac{23254.76}{1.42} + \frac{23254.76}{1.69} + \frac{23254.76}{2} + \frac{23254.76}{2.39}$$

$$VPN = -16949 + 19541.82 + 16376.59 + 13760.21 + 11627.38 + 9730.03$$

$$VPN = 54087.03$$

Análisis:

Como el VPN > 0, entonces es factible el desarrollo del proyecto.



CALCULO DE LA TASA DE INTERES (TIR)

Igualamos el VPN a cero, y hallamos la tasa de interés anual, que finalmente será el TIR.

$$0 = -16\,949 + \frac{31\,829.76}{(1+x)^1} + \frac{31\,829.76}{(1+x)^2} + \frac{31\,829.76}{(1+x)^3} + \frac{31\,829.76}{(1+x)^4} + \frac{31\,829.76}{(1+x)^5}$$

$$16\,949 = 31\,829.76 \left(\frac{1}{(1+x)^1} + \frac{1}{(1+x)^2} + \frac{1}{(1+x)^3} + \frac{1}{(1+x)^4} + \frac{1}{(1+x)^5} \right)$$

$$\frac{16\,949}{31\,829.76} = (x+1)^{-1} + (x+1)^{-2} + (x+1)^{-3} + (x+1)^{-4} + (x+1)^{-5}$$

$$\log\left(\frac{16\,949}{31\,829.76}\right)$$

$$= \log(x+1)^{-1} + \log(x+1)^{-2} + \log(x+1)^{-3} + \log(x+1)^{-4} + \log(x+1)^{-5}$$

$$\log\left(\frac{16\,949}{31\,829.76}\right)$$

$$= (-1)\log(x+1) + (-2)\log(x+1) + (-3)\log(x+1) + (-4)\log(x+1) + (-5)\log(x+1)$$

$$\log\left(\frac{16\,949}{31\,829.76}\right) = \log(x+1)[-1-2-3-4-5]$$

$$\log\left(\frac{16\,949}{31\,829.76}\right) = \log(x+1)[-15]$$

$$0.018 = \log(x+1)$$

$$10^{0.018} = x+1$$

$$1.042 = x+1$$

$$X = 0.042 = 4.2\%$$

$$\text{TIR} = 4.2\%$$



CÁLCULO DEL PERIODO DE RETORNO (PR)

$$PR = \frac{\textit{Inversion Total}}{\textit{Prom. Beneficios} - \textit{Costos de operacion}}$$

$$PR = \frac{16\,949}{31\,829.76 - 8\,575} = \frac{16\,949}{23\,254.76} = 0.73 \text{ años}$$

$$= 0.73 \times 12 = 8.76 \text{ meses}$$

$$= 0.76 \times 30 = 22.8 \text{ dias}$$

El periodo de retorno es de 8 meses y 23 días.



CAPITULO VI: CONCLUSIONES Y RECOMENDACIONES

6.1. Conclusiones

- a) En la etapa de pre - procesamiento es muy importante en este trabajo ya que a partir de ella los resultados difieren en la clasificación, los filtros empleados en este trabajo de investigación sirvieron para hallar resultados más óptimos, pero, así como estos algoritmos de pre - procesamiento, existen muchos más.
- b) En la etapa de segmentación solo se utilizó el algoritmo de Otsu, por ser el mejor de acuerdo a los antecedentes documentados en esta investigación. También se hicieron pruebas con el algoritmo Balu de la herramienta del mismo nombre, pero no se encontraron diferencias destacadas entre ambos.
- c) Se pudo determinar que para la evaluación de textura solo se necesita una muestra de la imagen del aislador para determinar su estado, en este estudio se utilizó una muestra de 180 X 180 pixeles.
- d) En la etapa de Extracción de características se usaron los algoritmos de Gabor y LBP, debido a que son algoritmos basados en textura y en las imágenes de aisladores se quería extraer características basado en la textura, esta etapa es muy decisiva también para la clasificación, a partir de ella se puede encontrar una diferencia de una imagen de aislador sucio de otro limpio. Al ejecutar ambos algoritmos, las muestras extraídas se realizaron con la tonalidad de color en escala de grises.



- e) En la última etapa de Clasificación se compararon 4 algoritmos siendo estos KNN, SVM y Redes Neuronales junto con una clasificación semiautomática, en cuanto a resultados el clasificador SVM, obtuvo una tasa mayor de aciertos con respecto a los otros.

- f) En esta investigación, se obtuvo mejores resultados con respecto a los demás con los algoritmos filtro Normalizado en la etapa de pre - procesamiento LBP en la etapa de extracción de características y SVM al momento de clasificar.



6.2. Recomendaciones

- a) Utilizar otros algoritmos en la fase de pre – procesamiento aparte de los utilizados en esta investigación, que realcen los detalles de las imágenes para obtener un mejor resultado.
- b) Se recomienda realizar la extracción de texturas con otros algoritmos u otras técnicas como, por ejemplo, utilizando la matriz de co – ocurrencia. En la cual después de generarse la matriz de co – ocurrencia de las imágenes, se pueden obtener diferentes características propias de esta técnica como lo son: homogeneidad, entropía, rugosidad, etc.
- c) En la etapa de segmentación se recomienda utilizar segmentación por crecimiento de regiones ya que solo se necesita una muestra para el estudio de la imagen.
- d) Se recomienda buscar otras características de diferenciación entre una imagen de aislador sucio y un aislador limpio más exactas, para lograr mejorar resultados.
- e) Se puede hacer uso de Componentes de Análisis Principal (PCA) después de la etapa de extracción de características, para reducir la dimensionalidad de los datos y para que en la clasificación se den datos más significativos, con ello se logrará reducir el coste computacional y el tiempo de procesamiento.



Bibliografía

- Agency, I. E. (2013). *International Energy Agency*. Obtenido de Indicadores de energía:
<http://energyatlas.iea.org/?subject=-297203538>
- Aldalur, B. S. (2002). Realce de imágenes: filtrado espacial. *Teledetección*, 31-42.
- Aristondo, J. (2010). Algoritmo de reconocimiento de forma y color para una plataforma robótica.
España: Universidad del País Vasco.
- Benitez, R., Escudero, G., Kanaan, S., & Masip, D. (2014). *Inteligencia Artificial Avanzada*.
Barcelona: Editorial UOC.
- Breiman, L. (2001). Random Forest. California, Estados Unidos: Universidad de California.
- Cano, A. (2005). *GAMMA*. Obtenido de Consideraciones en la selección de aisladores bajo
condiciones de contaminación atmosférica:
<http://www.gamma.co/pdf/boletines/tecnicos/boletin05.pdf>
- Codensa. (24 de Abril de 2004). *Likinormas*. Obtenido de
http://likinormas.micodensa.com/Especificacion/aisladores/et257_aisladores_porcelana_tipo_estacion
- COES. (2015). *COES*. Obtenido de Participacion del numero de fallas por tipo de causa de falla
CIER: <http://www.coes1.org.pe/PortalInformacion/WebPages/home.aspx#/EventosSEIN>
- Corona, M., & Ancona, M. (2011). *Diseño de algoritmos y su codificación en lenguaje C*. Mexico:
McGraw – Hill / Interamericana Editores S.A.
- Esqueda, J., & Palafox, L. (2005). *Fundamentos de procesamiento de imágenes*. Mexico:
Departamento de Editorial Universitaria.
- García, E. (2008). Detección y clasificación de objetos dentro de un salón de clases empleando
técnicas de procesamiento digital de imágenes. Mexico.



- Gavidia, C. (2014). Segmentación de imágenes médicas mediante algoritmos de colonia de hormigas. Lima, Lima, Peru: Universidad Catolica del Peru.
- Kim, S., Ho, S., & Man, Y. (2014). *CONCYTEC*. Obtenido de Estudio sobre la identificación de los defectos de la superficie de madera en base a características de textura:
<http://ezproxy.concytec.gob.pe:2053/science/article/pii/S1047320315001546>
- Mery, D. (2016). *Dmery*. Obtenido de
<http://dmery.ing.puc.cl/index.php/teaching/patrones/programa/>
- Montaño, D., & Huamán, C. (2013). Desarrollo de un software de monitoreo para el proceso de Germinación de plantones usando patrones de reconocimiento de imágenes. Pimentel, Chiclayo, Peru: Universidad de Señor de Sipan.
- Naciones Unidas*. (2014). Obtenido de La situación demográfica en el mundo en 2014:
http://www.un.org/en/development/desa/population/events/pdf/other/4/World%20Population%20Situation_2014_10%20key%20findings_es.pdf
- Ozturk, S., & Akdemir, B. (2015). *CONCYTEC*. Obtenido de Comparación de los algoritmos de detección de borde para el Análisis de textura en producción de vidrio:
<http://ezproxy.concytec.gob.pe:2053/science/article/pii/S1877042815039567>
- Pajares, G., & De la Cruz, J. (2002). *Visión por computador imágenes digitales y aplicaciones*. Mexico: RA-MA Editorial.
- Portilla, B. M. (2016). Implementación del Algoritmo de Otsu sobre FPGA. *Revista Cubana de Ciencias Informáticas*, 16-26.
- Posadas, A. (2013). *Determinación de Errores y Tratamiento de Datos. Ingeniería de Materiales*. Almeria: Universidad de Almeria.
- Pu, X., Fan, K., Cheng, X., Ji, L., & Zhou, Z. (2014). *CONCYTEC*. Obtenido de Reconocimiento de expresiones faciales de secuencias de imágenes usando doble clasificador random forest:
<http://ezproxy.concytec.gob.pe:2053/science/article/pii/S0925231215006220>



Quilmes, U. N. (2005). *UNQ*. Obtenido de

<http://iaci.unq.edu.ar/materias/vision/archivos/apuntes/Segmentaci%C3%B3n%20por%20Umbralizaci%C3%B3n%20-%20M%C3%A9todo%20de%20Otsu.pdf>

Santana, W., Osorio, G., Mari, I., & Sanín, I. (2012). *www.jornadastecnicasisa.co*. Obtenido de Metodología TcT para el cambio de aisladores poliméricos con condición crítica:

http://www.jornadastecnicasisa.co/Historico/2012/Lineas_Transmision/1901090.pdf

Semin, K. S. (2015). *Reconocimiento de monedas basado en imágenes utilizando binario de región invariante a la rotación, Patrones basados en magnitudes de gradiente*. Obtenido de

<http://www.sciencedirect.com/science/article/pii/S1047320315001546>

Shen, H., Chen, P., & Chang, L. (2013). *CONCYTEC*. Obtenido de Método de reconocimiento de defectos óxido recubrimiento puente de acero automatizada basada en el color y la

textura: <http://ezproxy.concytec.gob.pe:2053/science/article/pii/S0926580512001987>

Tribuj, M. a. (2016). *Segmentación de imágenes texturadas*. Buenos Aires: Universidad de Buenos Aires.

Vasallo. (2005). *Procesamiento de imágenes digitales utilizando descriptores de forma para la identificación de deficiencias nutricionales a nivel foliar del cafeto*. Chiclayo: Universidad Señor de Sipán.

Venero, M. (2011). *Técnicas de mantenimiento en equipos de Alta Tensión. Segundo Foro Regional de Electricidad en Arequipa "Modernización del Servicio Público de Electricidad"*.

Arequipa.

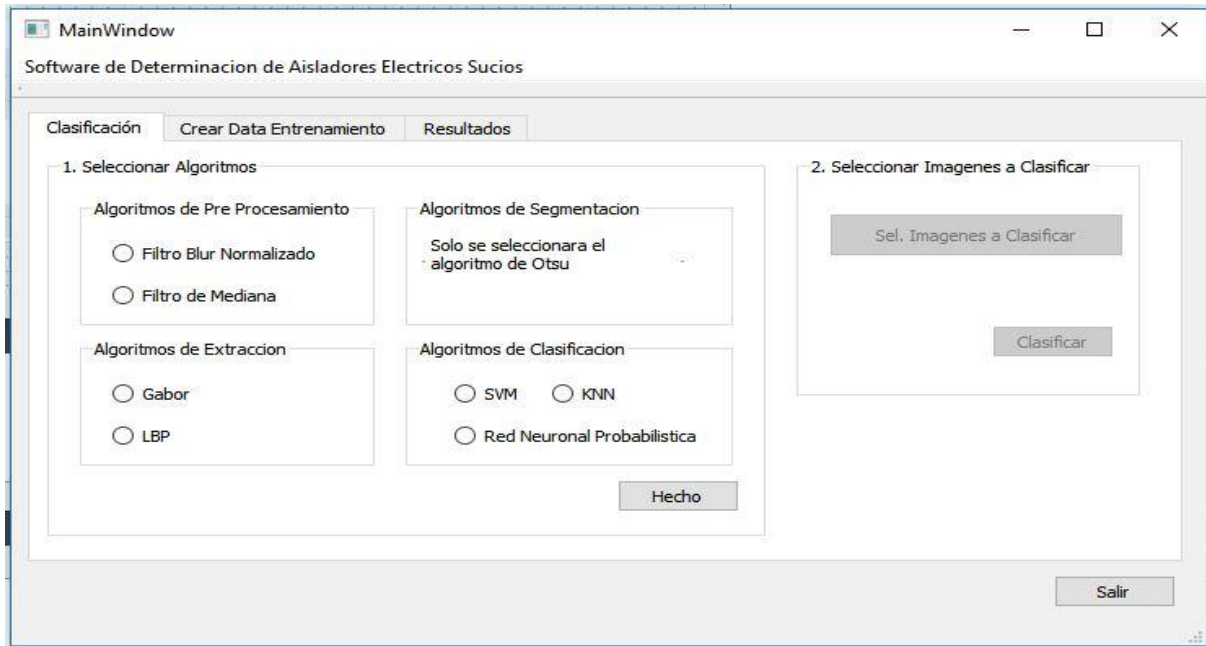
YongHua, X., & W, J.-C. (2014). *CONCYTEC*. Obtenido de Estudio sobre la identificación de defectos de la superficie de madera basados en textura:

<http://ezproxy.concytec.gob.pe:2053/science/article/pii/S0030402615004143>

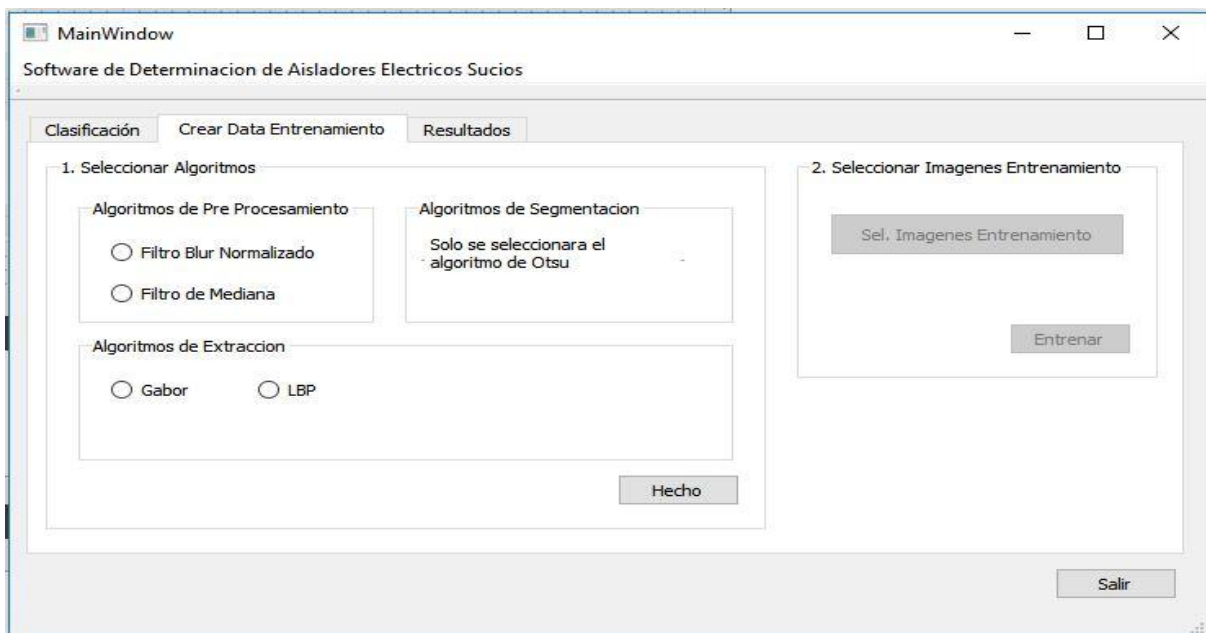


ANEXOS

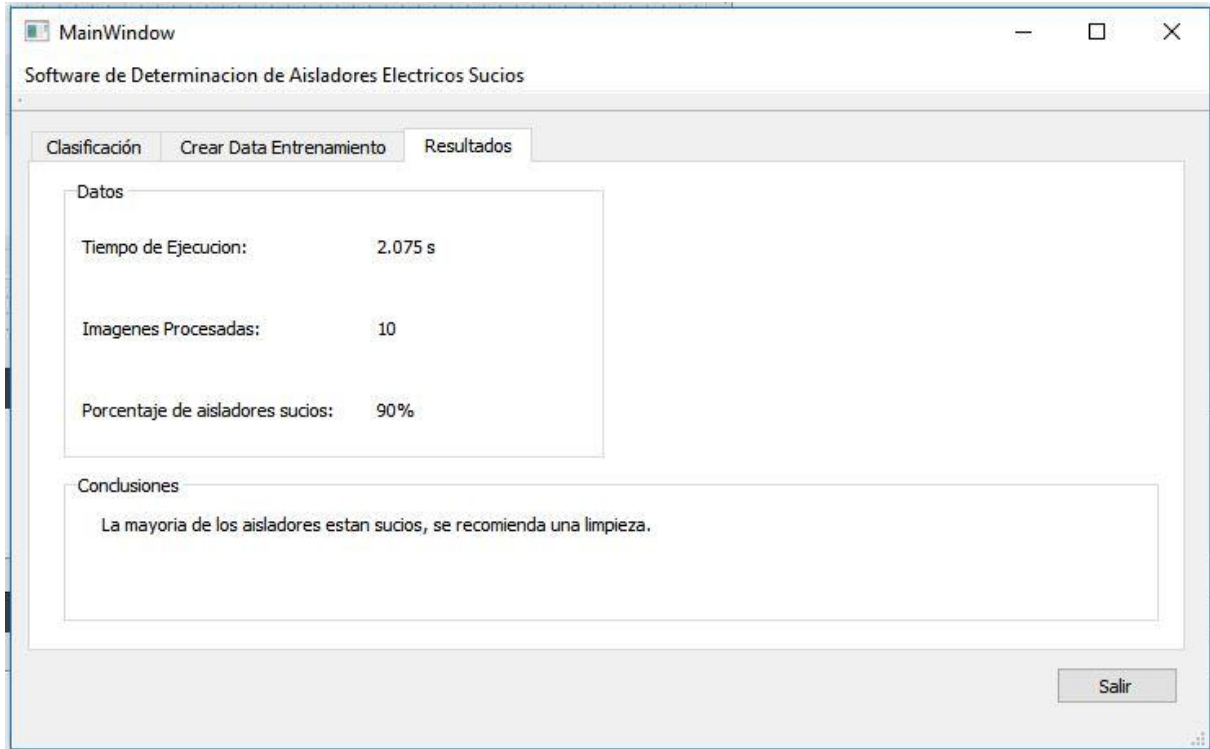
Anexo 1 – Programa Pestaña Clasificación



Anexo 2 – Programa Pestaña Entrenamiento



Anexo 3 – Programa Pestaña Resultados



Anexo 4

Cuadro de análisis después de aplicar el algoritmo LBP

Imagen	Mayores Limpios															Mayores Sucios
	De 0 a 15	De 16 a 31	De 32 a 47	De 48 a 63	De 64 a 79	De 80 a 95	De 96 a 111	De 112 a 127	De 128 a 143	De 144 a 159	De 160 a 175	De 176 a 191	De 192 a 207	De 208 a 223	De 224 a 239	
Aislador1	3180	2597	967	1681	897	624	892	2452	1547	960	533	1106	1643	1291	2198	4394
Aislador2	2423	1524	800	1862	1230	523	691	2219	2165	785	475	1264	2364	1067	1468	2797
Aislador4	2321	1364	613	2138	1367	440	571	2272	2026	675	342	1426	2576	907	1232	2583
Aislador5	3472	2799	956	1910	885	660	805	2278	1425	934	583	1050	1584	1261	2388	4290
Aislador51	3357	2715	642	2096	950	474	727	2289	1532	882	314	1118	1978	995	2563	4834
Aislador54	3007	1919	894	1790	1151	693	895	2612	1878	962	542	1404	1913	1306	1904	4170
Aislador49	3472	2799	956	1910	885	660	805	2294	1425	934	583	1051	1550	1276	2402	4303
Aislador59	3403	1859	1125	1587	1542	704	884	2027	1647	880	732	1538	1683	1200	1835	3831
Aislador64	2246	1756	1016	2158	752	827	715	3568	1536	687	515	1372	1274	1129	1214	3257
Aislador65	1993	1577	937	2092	780	705	686	3539	1853	641	464	1210	1476	1048	1136	3143
Aislador66	2259	1451	1029	1816	1020	764	762	2651	1806	776	597	1251	1584	1202	1234	2743
Aislador163	3057	1992	1192	1253	1227	936	917	1863	1430	922	884	1446	1394	1470	2088	3773
Aislador166	1769	1408	839	2662	710	683	503	3661	2394	547	445	1071	2324	986	941	2306
Aislador172	2530	1993	834	2149	873	668	599	2615	1753	775	502	1139	1780	1276	1634	3302



Anexo 5

Cuadro de análisis después de aplicar el algoritmo Filtro de Gabor

	Sucios	Limpios
	14	34
	21	29
	10	37
	28	29
	27	46
	14	9
	24	16
	21	36
	23	43
	20	18
	28	43
	18	56
	20	41
	14	33
	35	39
	21	17
	62	20
	28	9
	26	21
	35	50
	20	
	22	
	21	
	23	
	83	
	61	
	44	
	32	
	40	
	42	
Promedio	29.2333333	31.3
Max	83	56
Min	10	9
Mayores o iguales a 31	21	11

