



UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MORELOS

INSTITUTO DE INVESTIGACIÓN EN CIENCIAS BÁSICAS Y
APLICADAS

CENTRO DE INVESTIGACIÓN EN CIENCIAS

**"Detección automática de objetos con errores de manufactura
usando imágenes RGB-D"**

TESIS

QUE PARA OBTENER EL GRADO DE:
LICENCIADO EN CIENCIAS ÁREA TERMINAL CIENCIAS COMPUTACIONALES
Y COMPUTACIÓN CIENTÍFICA

PRESENTA: LUIS ALFREDO REYNOSO GOMEZ

DIRECTOR DE TESIS: DR. JORGE ALBERTO FUENTES PACHECO

ÍNDICE GENERAL

1	INTRODUCCIÓN	1
1.1	Planteamiento del problema	1
1.2	Hipótesis	2
1.3	Justificación	2
1.4	Objetivo general	2
1.5	Objetivos específicos	2
1.6	Alcances y limitaciones	3
1.6.1	Alcances	3
1.6.2	Limitaciones	3
2	ESTADO DEL ARTE	4
2.1	Técnicas clásicas	4
2.1.1	Umbral automático para la detección de defectos	4
2.1.2	SIFT (Scale Invariant Feature Transform), modelo <i>Bag of Visual Words</i> y algoritmos de clasificación.	7
2.1.3	Máquinas de soporte vectorial para la detección de defectos en lami- nado de acero.	9
2.1.4	Máquinas de soporte vectorial para la detección de defectos en la producción de tortillas.	12
2.2	Técnicas de aprendizaje profundo	16
2.2.1	PaDiM: <i>Framework</i> basado en el modelado de distribución de parches para la detección y localización de anomalías.	16
2.2.2	Detección de defectos de superficies metálicas mediante el uso de un <i>Auto-Encoder</i>	19
2.2.3	<i>ANOMALIB</i> : Una librería para la detección de anomalías basada en aprendizaje profundo.	21
3	METODOLOGÍA DE SOLUCIÓN	23

3.1	El conjunto de datos	23
3.1.1	Conjunto de datos 1	25
3.1.2	Conjunto de datos 2	25
3.1.3	Boosting Monocular Depth	27
3.2	DESCRIPCIÓN DE LA CNN	29
3.3	Extracción de características	30
3.4	Aprendiendo de la normalidad: la matriz de parámetros gaussianos	32
3.5	Mapa de anomalías y la distancia de Mahalanobis	33
3.6	Calculo del Umbral para un análisis cualitativo	33
4	PRUEBAS Y RESULTADOS	34
4.1	Métricas de desempeño	35
4.1.1	Matriz de confusión	35
4.1.2	Precisión - recuerdo - especificidad	36
4.1.3	Curvas ROC - curvas PR	36
4.1.4	<i>AUC score - PR score</i>	37
4.1.5	<i>Matthews correlation coefficient (MCC)</i>	38
4.1.6	Análisis cualitativo	39
4.1.7	Análisis cuantitativo	42
5	CONCLUSIONES	45
6	REFERENCIAS BIBLIOGRÁFICAS	47

ÍNDICE DE FIGURAS

Figura 1	Selección de umbral óptimo en histograma de nivel de gris: (a) bimodal; (b) unimodal. Imagen extraída de [Ng, 2006].	4
Figura 2	Resultados de los métodos, énfasis en el valle y Otsu para detectar rayaduras en hojas de metal, (a) imagen de la hoja de metal; (b) resultado del umbral del método énfasis en el valle; (c) resultado del umbral del método Otsu; (d) histograma y valores de umbral. Imagen extraída de [Ng, 2006]. . .	5
Figura 3	Métricas del modelo para la detección de defectos utilizando el modelo <i>Bag of Visual Words</i> y el descriptor SIFT. Imagen extraída de [Jia y col., 2004].	8
Figura 4	Esquema del sistema de visión por computadora para la detección de grietas o fracturas en laminado de acero. Imagen extraída de [Jia y col., 2004].	9
Figura 5	Idea visual de la expansión de la zona dañada Imagen extraída de [Jia y col., 2004].	10
Figura 6	Comparación del desempeño del modelo de SVM y una red neuronal artificial. Imagen extraída de [Jia y col., 2004].	10
Figura 7	Comparación del desempeño del modelo de máquinas de soporte vectorial y una red neuronal artificial después de la reducción de ruido. Imagen extraída de [Jia y col., 2004].	11
Figura 8	Datos generados por la encuesta, a) color b) diferencia observable del tamaño de los granos de maíz c) grosor d) fracturas o grietas e) tamaño f) áreas quemadas g) arrugas h) circularidad i) masa residual j) grietas en la capa fina de la tortilla k) rayas en la superficie de la tortilla l) homogeneidad en los bordes. Imagen extraída de [Mery y col., 2010].	12
Figura 9	Galería de tortillas representando cada clase y subclase. Imagen extraída de [Mery y col., 2010].	13
Figura 10	Representación del funcionamiento del sistema detector de errores en tortillas. Imagen extraída de [Jia y col., 2004].	14
Figura 11	Metodología para la obtención de la matriz de parámetros Gaussianos. Imagen extraída de [Defard y col., 2020].	17

Figura 12	Arquitectura de la red <i>auto-encoder</i> (AE). Imagen extraída de [Tao y col., 2018].	19
Figura 13	Arquitectura del modelo propuesto. Imagen extraída de [Tao y col., 2018].	20
Figura 14	Número de conjuntos de datos y publicaciones científicas recientemente publicados en la literatura. Imagen extraída de [Akçay y col., 2022]. . .	21
Figura 15	Arquitectura de <i>Anomalib</i> . Imagen extraída de [Akçay y col., 2022]. . .	21
Figura 16	Columna izquierda: imágenes normales. Columna central: imágenes de las mismas clases con anomalías resaltadas en amarillo. Columna derecha: mapas de calor de anomalías obtenidos por el modelo PaDiM. Imagen extraída de [Defard y col., 2020].	24
Figura 17	Ejemplo de imágenes obtenidas con la cámara <i>Intel RealSense D435</i> .	25
Figura 18	Ejemplo de imágenes obtenidas con la cámara <i>Intel RealSense D435</i> .	25
Figura 19	Ejemplo de la obtención de un mapa de profundidad a partir de una imagen RGB obtenida con el modelo <i>Boosting Monocular Depth</i>	27
Figura 20	Ejemplo de imagen RGB e imagen de profundidad obtenido mediante <i>Boosting Monocular Depth</i>	28
Figura 21	Ejemplo de imagen RGB e imagen de profundidad obtenido mediante <i>Boosting Monocular Depth</i>	28
Figura 22	Arquitectura de la CNN ResNet50V2.	29
Figura 23	Generación de <i>patches</i> a partir de una imagen.	30
Figura 24	Extracción de vectores característicos para generar una matriz de parámetros gaussianos. Imagen extraída de [Defard y col., 2020].	31
Figura 25	Comparación de predicción con el GT a nivel píxel y parche, en donde el color azul representa los píxeles dañados indicados por el GT, el color verde representa los parches dañados indicados por el modelo pero que no son ciertas y el color rojo la intersección de la predicción con el GT.	34
Figura 26	Matriz de confusión para la tarea detección de anomalías.	35
Figura 27	Ejemplo de curva ROC.	36
Figura 28	Ejemplo de curva PR.	37

ÍNDICE DE TABLAS

1	Comparación de la predicción del modelo utilizando el umbral descrito en la sección 3.6 con GT a nivel píxel en las base de datos 1 y 2, en donde el color azul representa los píxeles dañados indicados por el GT, el color verde representa los parches dañados indicados por el modelo pero que no son ciertas y el color rojo la intersección de la predicción con el GT.	39
2	Comparación de la predicción del modelo utilizando el umbral descrito en la sección 3.6 con GT a nivel píxel en la base de datos 1, en donde el color azul representa los píxeles dañados indicados por el GT, el color verde representa los parches dañados indicados por el modelo pero que no son ciertas y el color rojo la intersección de la predicción con el GT.	40
3	Comparación de la predicción del modelo utilizando el umbral descrito en la sección 3.6 con GT a nivel píxel en la base de datos 1, en donde el color azul representa los píxeles dañados indicados por el GT, el color verde representa los parches dañados indicados por el modelo pero que no son ciertas y el color rojo la intersección de la predicción con el GT.	41
4	Desempeño de los modelos RGB y RGB+D en su versión de 6×6 en las bases de datos 1 y 2 utilizando las métricas <i>PR score</i> y <i>AUC score</i>	42
5	Desempeño de los modelos RGB y RGB+D en su versión de 8×8 en las bases de datos 1 y 2 utilizando las métricas <i>PR score</i> y <i>AUC score</i>	43
6	Desempeño de los modelos RGB y RGB+D en su versión de 10×10 en las bases de datos 1 y 2 utilizando las métricas <i>PR score</i> y <i>AUC score</i>	44

1 INTRODUCCIÓN

Desde que la revolución industrial trajo consigo la mecanización de la producción, se tuvieron grandes mejoras, como la reducción de costos y tiempos de manufactura. De igual manera surgieron nuevos retos que han sido resueltos poco a poco a través de técnicas de Inteligencia Artificial. No obstante, la detección de errores de fabricación en piezas industriales a través del análisis de imágenes continúa siendo un reto, [Ren y col., 2021].

1.1 PLANTEAMIENTO DEL PROBLEMA

En la actualidad, es muy común que una empresa dedicada a la manufactura, realice cientos de productos al día. Sin embargo, el área de control de calidad normalmente no es capaz de verificar que cada producto esté garantizado a la perfección debido a su gran volumen de producción lo que se convierte en una tarea muy complicada y costosa para el ser humano.

Es por eso que el uso de tecnologías para la detección automática de errores de manufactura se vuelve cada vez más solicitado por estas empresas.

En esta tesis se pretende implementar un sistema automático para la detección errores de manufactura en objetos, mediante el uso de imágenes RGB+D. Para lo cual, se utilizará una cámara Intel Realsense 2.5 D, técnicas de Aprendizaje Profundo y visión por computadora.

1.2 HIPÓTESIS

La combinación de la información extraída de una imagen a color y de su respectivo mapa de profundidad, obtenidos por medio de una cámara de bajo costo, puede mejorar el proceso de detección automática de errores de fabricación en objetos.

1.3 JUSTIFICACIÓN

Al desarrollar e implementar un modelo automático de detección de errores en objetos, se podría ahorrar tanto dinero, como tiempo, pues identificar un artículo dañado reduce considerablemente las fallas del producto final. Por ejemplo, identificar un chip dañado el cual se utilizará para la fabricación de un celular, permite cambiarlo por uno en buenas condiciones y evitar todos los procesos de garantía cuando el equipo no funcione de manera adecuada, reduciendo así la cantidad de devoluciones del producto.

Es posible generar un modelo con buen desempeño en el área, incluso los hay en la literatura para casos más particulares, de igual manera existen modelos de Aprendizaje Profundo previamente entrenados en grandes conjuntos de datos que pueden ser reutilizables y ajustables para un problema particular como el nuestro.

1.4 OBJETIVO GENERAL

Desarrollar un sistema automático para detectar errores de manufactura en objetos usando imágenes RGB+D y Aprendizaje Profundo.

1.5 OBJETIVOS ESPECÍFICOS

- Realizar un estudio del estado del arte de las técnicas usadas para la detección de errores de manufactura en imágenes.
- Implementar y modificar una de las técnicas analizadas incorporando los mapas de profundidad obtenidos por una cámara Intel Realsense D435.
- Evaluar el desempeño del modelo generado.

1.6 ALCANCES Y LIMITACIONES

1.6.1 Alcances

- El modelo será funcional para distintos tipos de objetos, considerando que no se conocen los errores de fabricación que podrían presentar.
- El modelo será probado en un ambiente con condiciones de iluminación variable para probar su desempeño.
- Las imágenes RGB y los mapas de profundidad serán obtenidos por una cámara Intel realsense D435 y su librería pyrealsense2.

1.6.2 Limitaciones

- Para el propósito de este proyecto el rango de errores no puede ser muy grande.
- Las fotografías de los objetos deben ser capturadas en los mismos ángulos, esto es que, los objetos deben estar en la misma posición relativa a las diferentes poses de la cámara.

2 ESTADO DEL ARTE

En este capítulo se presenta un breve resumen sobre las técnicas clásicas de visión por computadora y técnicas de aprendizaje profundo para la detección de errores de manufactura junto a sus ventajas y limitaciones.

2.1 TÉCNICAS CLÁSICAS DE VISIÓN POR COMPUTADORA

En esta sección discutiremos los métodos clásicos para la detección de errores de manufactura sin incluir los métodos de aprendizaje profundo.

2.1.1 Umbral automático para la detección de defectos

Un problema clásico en la rama de visión por computadora, es la distinción de píxeles que conforman un objeto de interés del fondo de la imagen. Esto seleccionando un umbral que es capaz de separar el objeto del fondo de la imagen en función de su distribución de nivel de grises. En la figura 1 se muestra la representación de un umbral óptimo en diferentes histogramas de escala de grises.

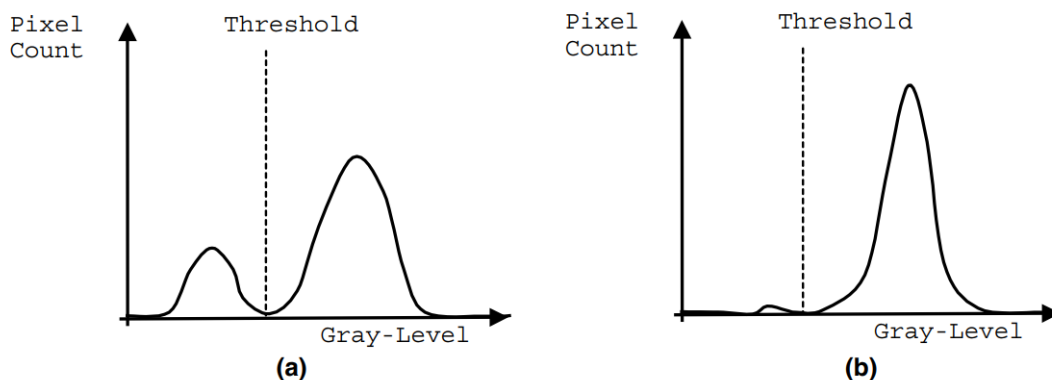


Figura 1: Selección de umbral óptimo en histograma de nivel de gris: (a) bimodal; (b) unimodal. Imagen extraída de [Ng, 2006].

La técnica antes mencionada puede ser utilizada para la detección de defectos, al ser capaz de separar o distinguir el fondo de la imagen de un objeto se pueden detectar agujeros que no deberían existir en diversos materiales, un ejemplo clásico para este método es la detección de contaminantes en muestras líquidas. En una imagen con una muestra de agua contaminada con aceite y un umbral adecuado, es posible identificar el aceite como el objeto de interés y el agua como el fondo de la imagen, así al analizar una muestra de agua, si se encuentra un objeto, será considerada una muestra contaminada, por otro lado si no hay ningún objeto que separar del fondo, el resultado será que únicamente encontraremos el fondo de la imagen lo cual indicara que la muestra no esta contaminada.

La idea anterior puede ser extrapolada a problemas similares.

Para obtener un buen umbral que separe objetos del fondo de la imagen, es necesario como se observa en la figura 2, encontrar el valle en el histograma de nivel de grises, para esta tarea en [Ng, 2006] se exploran 2 técnicas distintas las cuales son, el método Otsu y el método del énfasis en el valle.

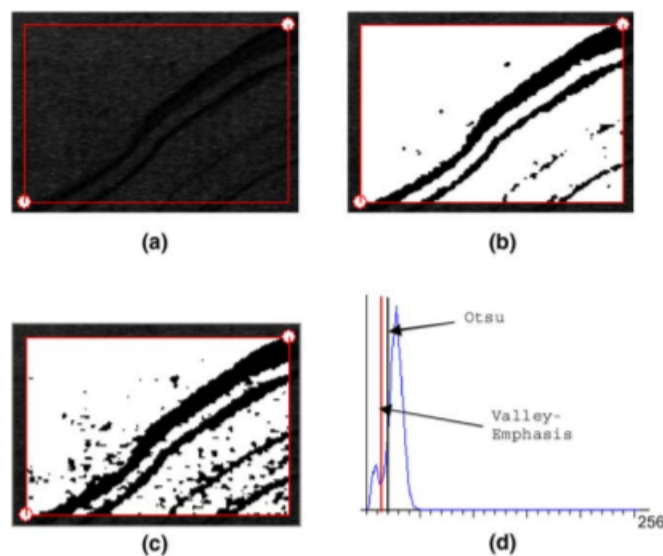


Figura 2: Resultados de los métodos, énfasis en el valle y Otsu para detectar rayaduras en hojas de metal, (a) imagen de la hoja de metal; (b) resultado del umbral del método énfasis en el valle; (c) resultado del umbral del método Otsu; (d) histograma y valores de umbral. Imagen extraída de [Ng, 2006].

Las ventajas del método son:

- Es relativamente sencillo de implementar.
- Puede ser implementado para la inspección de algunos líquidos, verificando que no haya otros líquidos u objetos extraños en la muestra.
- Al ser capaz de distinguir entre un objeto y su fondo, se puede usar para el análisis de objetos que no deban tener perforaciones.
- Existen distintos métodos para la selección de un umbral, de esta manera, como se discute en, [Ng, 2006], se puede trabajar con histogramas unimodales y bimodales.

Respecto a las limitaciones, se pueden enumerar las siguientes:

- Al no utilizar imágenes RGB, parte de la información de la imagen se pierde.
- Cuando el histograma no presenta valles o picos claros, es difícil seleccionar un umbral óptimo.
- La cantidad de errores que es capaz de detectar es muy limitado.
- Cuando los defectos son pequeños, es muy complicado identificarlos.

2.1.2 SIFT (Scale Invariant Feature Transform), modelo *Bag of Visual Words* y algoritmos de clasificación.

A lo largo de la historia de la visión por computadora, muchas técnicas han surgido para poder describir una imagen, de tal manera que una computadora pueda interpretar dichas descripciones y eventualmente sea capaz de clasificar imágenes como pertenecientes a la misma clase o no.

En [Dunderdale y col., 2019] se propone el uso combinado del modelo *Bag of Visual Words*, el método de descriptores SIFT de imágenes, los modelos de *random forest* y máquinas de soporte vectorial para la detección de defectos en sistemas fotovoltaicos.

Como se menciona anteriormente, una forma de obtener un buen algoritmo encargado de clasificar un objeto como defectuosos o no, es obtener primero, una buena descripción de la imagen que contiene el objeto en cuestión, esto es, implementar un algoritmo que sea capaz de detectar características locales en una imagen y a su vez describirlas, así es necesaria la implementación de un detector de características y un descriptor de características. Afortunadamente el algoritmo SIFT puede realizar ambas tareas y fungir como detector y descriptor de características.

En [Dunderdale y col., 2019] se menciona que el proceso para calcular los descriptores SIFT consta de los siguientes pasos:

- Detección de máximos y mínimos de espacio de escala.
- Umbralización.
- Asignación de orientación.
- Descripción de puntos clave.

A su vez se describe el desarrollo del modelo *Bag of Visual Words* en tres puntos:

- Extracción de características locales.
- Generación de un *Codebook*.
- Representación de cada imagen usando el *Codebook*.

En donde el método SIFT es utilizado para realizar el primer punto. Es así como el modelo *Bag of Visual Words* toma la salida del método SIFT para que finalmente la información producida por el modelo *Bag of Visual Words* sea utilizada en algún modelo de *Machine Learning* como lo son RF o SVM. En la figura 3 se pueden apreciar los resultados del modelo evaluado con diferentes métricas.

Model	Average Accuracy	Standard Deviation	Maximum Accuracy	Precision	Recall
Random Forest	91.2%	3.6%	93.7%	91.9%	92.0%
SVM (Poly)	90.7%	4.3%	94.9%	90.4%	91.5%
SVM (Radial)	89.6%	3.5%	92.4%	90.1%	89.5%

Figura 3: Métricas del modelo para la detección de defectos utilizando el modelo *Bag of Visual Words* y el descriptor SIFT. Imagen extraída de [Jia y col., 2004].

Como se puede observar en la figura 3, los modelos implementados en [Dunderdale y col., 2019], obtuvieron una precisión media cercana entre el 90 %, por lo que se puede decir que no se obtuvieron malos resultados y los modelos tuvieron un buen comportamiento.

Las ventajas del método son:

- Sencillo de entender.
- El método de extracción y descripción de características SIFT puede implementarse en lenguajes como: C, C++, Java and Python (OpenCV, 2017),
- Su costo computacional lo hace ideal para trabajar con grandes conjuntos de datos.
- Mostró un gran desempeño tanto para la detección de defectos en sistemas fotovoltaicos y un avance para la clasificación automática de ellos.

Respecto a las limitaciones, se pueden enumerar las siguientes:

- El algoritmo SIFT por un periodo de tiempo, dejó de ser gratuito para uso comercial, ya que su creador, David Lowe, registro una patente para el algoritmo SIFT.
- Funciona solo para sistemas fotovoltaicos.
- Al no tener un gran conjunto de datos, no se realizaron pruebas que puedan garantizar un mejor desempeño en ellos.

2.1.3 Máquinas de soporte vectorial para la detección de defectos en laminado de acero.

En [Jia y col., 2004] se aborda un problema de manufactura particular, la detección de grietas o fracturas en acero laminado donde también hacen uso de una SVM para atacar el problema.

La figura 4 muestra el funcionamiento general del sistema de visión por computadora empleado.

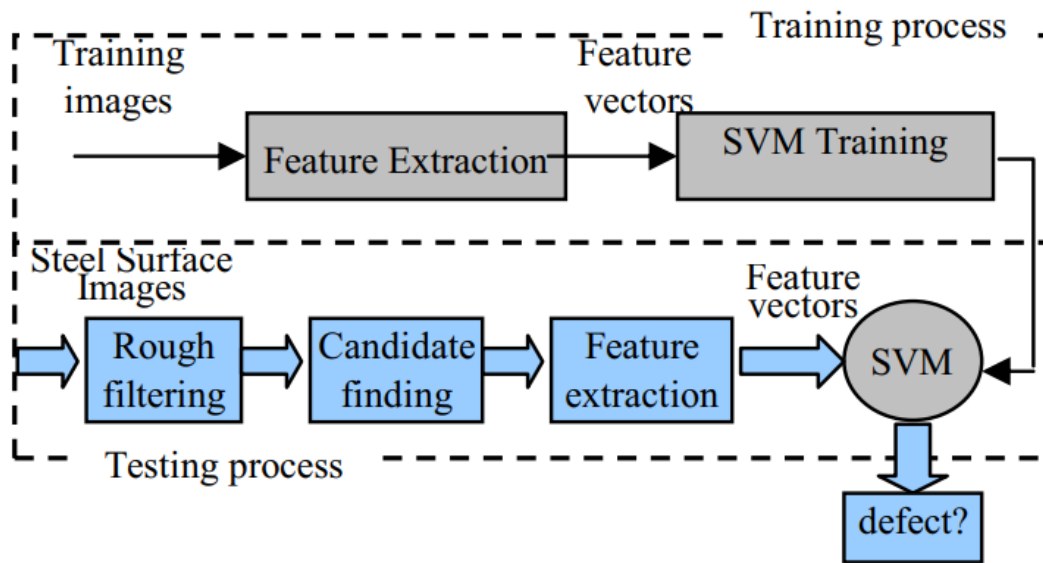


Figura 4: Esquema del sistema de visión por computadora para la detección de grietas o fracturas en laminado de acero. Imagen extraída de [Jia y col., 2004].

En donde *Rough filtering* es un algoritmo que aplica un operador de gradiente horizontal para la detección de grietas o fracturas en el acero laminado, al aplicar el gradiente horizontal, si este no contiene valores relevantes, véase más en [Jia y col., 2004] se puede concluir que una imagen pertenece a la clase normal, es decir que el laminado de acero no tiene grietas o fracturas. Esto ayuda a descartar de manera rápida estas imágenes para continuar con los siguientes en la línea de producción.

Si el gradiente horizontal sí contiene valores relevantes entonces, se aplica la fase de *candidate finding* la cual consiste en expandir las zonas iniciales de fracturas en 6 direcciones de píxeles vecinos para encontrar la zona entera dañada. La figura 5 ilustra la manera en que la zona se expande a los píxeles vecinos, una vez la zona dañada es completamente localizada, se procede a la extracción de características, para finalmente utilizar SVM y realizar

la clasificación como una imagen perteneciente a la clase normal o la clase dañada.

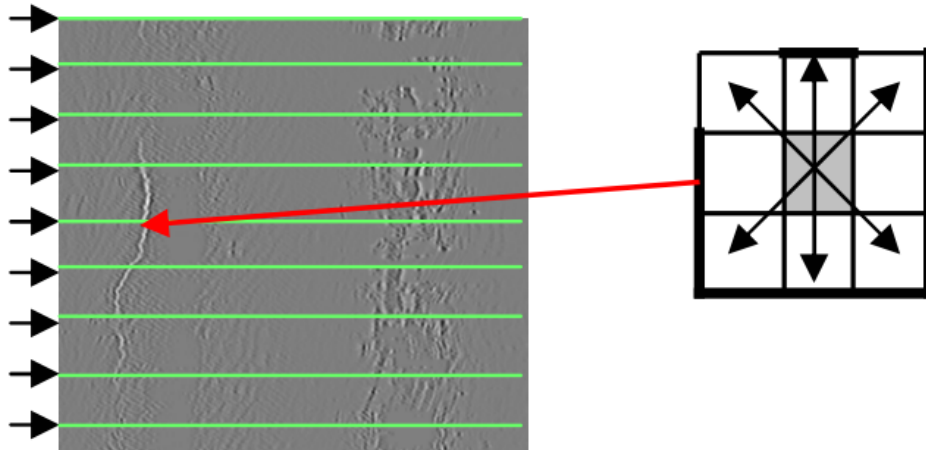


Figura 5: Idea visual de la expansión de la zona dañada Imagen extraída de [Jia y col., 2004].

Como se puede observar en la figura 6, la maquina de soporte vectorial utilizada tiene un mejor desempeño que una red neuronal artificial, para obtener un mejor desempeño se implemento un pre-procesamiento para la eliminación de ruido, el desempeño de ambos modelos después de la eliminación de ruido se muestra en la siguiente figura:

Predict \ True	Total	SVM classifier			ANN classifier		
		Seam	Non-seam	Accuracy	Seam	Non-seam	Accuracy
Seam	49	40	9	85.1%	27	22	55.1%
Non-seam	257	20	237	91.5%	9	248	96.5%
Total	306	60	246	90.5%	34	272	89.9%

Figura 6: Comparación del desempeño del modelo de SVM y una red neuronal artificial. Imagen extraída de [Jia y col., 2004].

La figura 7 muestra los resultados de las métricas después de aplicar un pre-procesamiento las imágenes. Si bien el desempeño de la red neuronal artificial aumentó, el desempeño de la máquina de soporte vectorial se mantuvo con un porcentaje superior.

Predict \ True	From SVM classifier				From ANN classifier		
	Total	Seam	Non-seam	Accuracy	Seam	Non-seam	Accuracy
Seam	47	41	6	87.2%	34	13	72.3%
Non-seam	259	11	248	95.8%	15	244	94.2%
Total	306	52	254	94.4%	49	257	90.8%

Figura 7: Comparación del desempeño del modelo de máquinas de soporte vectorial y una red neuronal artificial después de la reducción de ruido. Imagen extraída de [Jia y col., 2004].

Las ventajas del método son:

- Mejor desempeño en comparación con una red neuronal artificial.
- La previa clasificación de imágenes normales reduce significativamente el tiempo de inspección al descartar imágenes para su inspección.
- La previa clasificación de imágenes normales siempre encuentra las grietas o fracturas si las hay.
- La reducción de ruido aumenta la exactitud en la clasificación.
- Mostró una velocidad de extracción de características y clasificación total de 7.121 segundos en todo el conjunto de prueba.

Respecto a las limitaciones, se pueden enumerar las siguientes:

- Más del 50 % de grietas o fracturas son falsos positivos, no son grietas o fracturas.
- En general más del 50 % de falsos positivos hace que el algoritmo no sea eficiente.
- Funciona solo para laminado de acero.

2.1.4 Máquinas de soporte vectorial para la detección de defectos en la producción de tortillas.

Uno de los alimentos más comunes, en México y gran parte de Latinoamérica son las tortillas, es por esto que desde hace algunos años la producción en masa de tortillas aumentó, dando lugar incluso a las tortillas empacadas en bolsas de plástico las cuales están disponibles en los grandes supermercados, principalmente de México.

En México existen organismos dedicados a establecer y verificar normas junto a estándares de salubridad para garantizar que se produzcan alimentos aptos para el consumo humano, no existen organismos que se encarguen de verificar la calidad o el sabor de un producto, esto genera que exista muy poca información en la literatura sobre la calidad o el sabor que debería tener una tortilla, sin estos datos es muy complicado que la producción de tortillas en masa tenga una línea bien definida de calidad.

[Mery y col., 2010] menciona, que realizada una encuesta a 100 mexicanos principalmente residentes de la ciudad y estado de México, se obtuvo que el color de una tortilla es lo más importante al calificar a una tortilla como de buena calidad, el resto de datos obtenidos por la encuesta se observan en la figura 8:

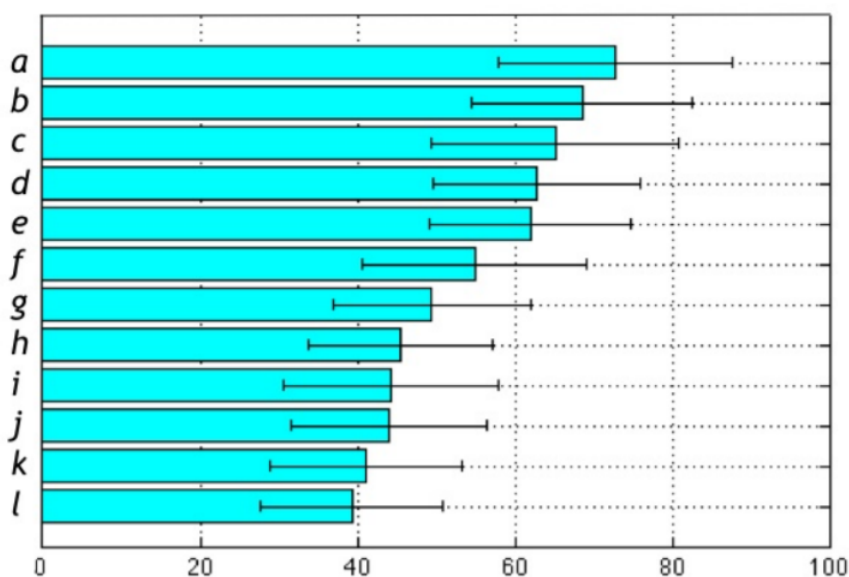


Figura 8: Datos generados por la encuesta, a) color b) diferencia observable del tamaño de los granos de maíz c) grosor d) fracturas o grietas e) tamaño f) áreas quemadas g) arrugas h) circularidad i) masa residual j) grietas en la capa fina de la tortilla k) rayas en la superficie de la tortilla l) homogeneidad en los bordes. Imagen extraída de [Mery y col., 2010].

Utilizando la información anterior se logró generar 5 subclases: "extremadamente bien", "bien", "neutro", "mal" y "muy mal", sumado a 3 previas clases: "chica", "mediana" y "grande". Las subclases hacen referencia al gusto de la población por las tortillas, donde "extremadamente bien" significa que la tortilla es altamente aceptada como una buena tortilla de esta manera, con las clases y subclases mencionadas anteriormente, se puede realizar una clasificación de tortillas y definir cuando una tortilla tiene algún problema en la manufactura.

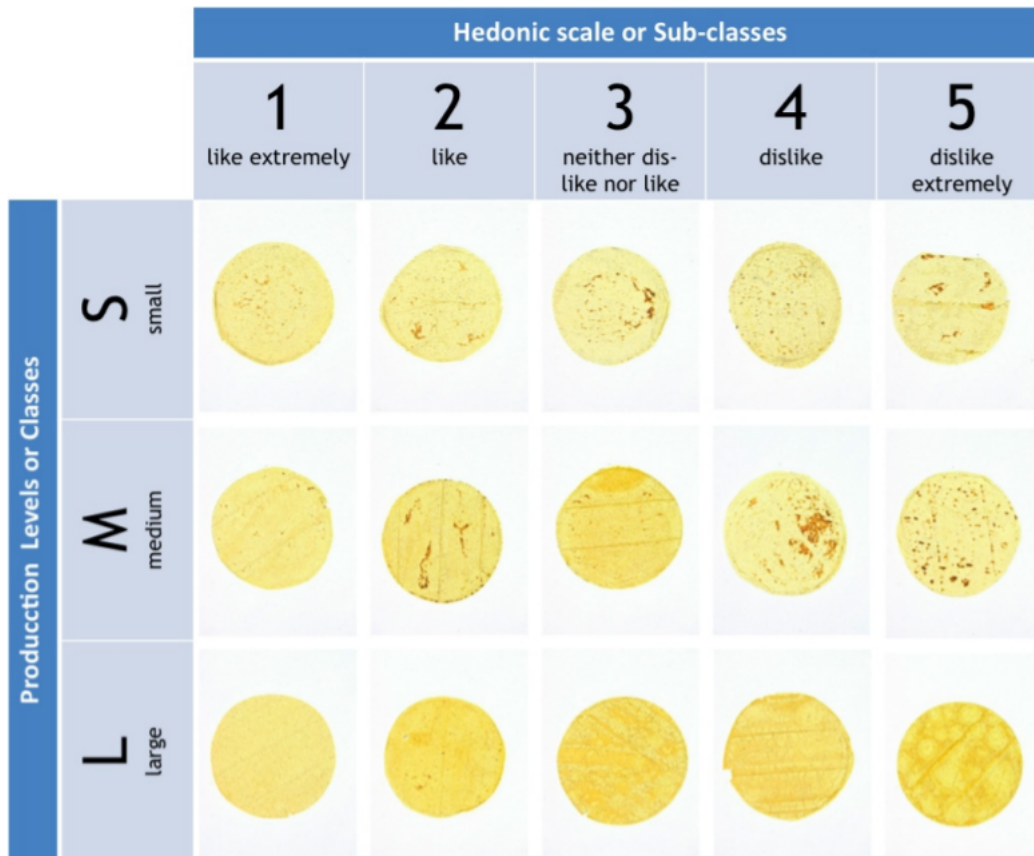


Figura 9: Galería de tortillas representando cada clase y subclase. Imagen extraída de [Mery y col., 2010].

Para realizar la clasificación de tortillas y etiquetar la clase y subclase a la que pertenece, se generaron vectores característicos de cada imagen, para después, utilizar una clasificación en dos fases, primero se utilizó un modelo de SVM para realizar la clasificación de la subclase, esto utilizando la estrategia uno contra todos, de esta manera se pueden obtener todas las posibles subclases a la que pertenece una imagen, posteriormente utilizando una técnica de promedio, se obtiene la clase perteneciente, teniendo como resultado en las métricas del problema de hasta un 99% de efectividad.

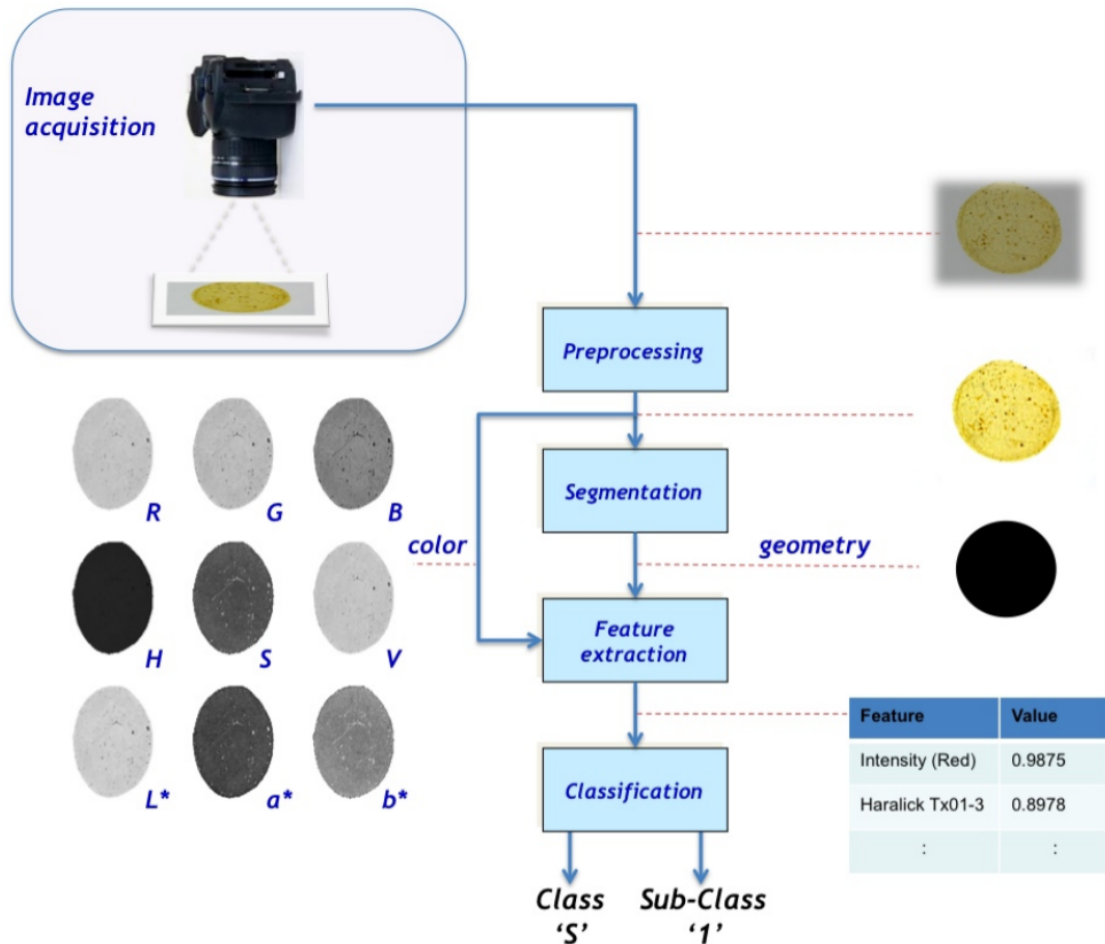


Figura 10: Representación del funcionamiento del sistema detector de errores en tortillas. Imagen extraída de [Jia y col., 2004].

De manera similar, para la clasificación de clases la información obtenida por los vectores característicos es muy amplia, utilizando un clasificador que emplea la distancia de Mahalanobis se obtuvo hasta un 99% de efectividad en sus métricas.

Las ventajas del método son:

- Se realizó un estudio de mercado, el cual indica lo que es una tortilla de “calidad” para algunos mexicanos.
- Se obtuvo un buen desempeño al momento de clasificar una imagen como perteneciente a una tortilla de buena calidad o de mala calidad, donde una tortilla de mala calidad es considerada una tortilla con problemas.
- Los vectores característicos contienen una gran cantidad de información de distintos espacios de color, forma, tamaño, etc. Para ver más consulte, [Jia y col., 2004].

La principal desventaja es que:

- Únicamente funciona en tortillas.

2.2 TÉCNICAS DE APRENDIZAJE PROFUNDO

2.2.1 PaDiM: *Framework* basado en el modelado de distribución de parches para la detección y localización de anomalías.

El desarrollo de nuevas tecnologías facilita de diversas maneras la resolución de problemas para la humanidad, pero muchas veces, con nuevas tecnologías surgen nuevos problemas.

Si bien el aprendizaje profundo es una técnica robusta, útil y bastante usada en el área de la inteligencia artificial, un problema relativamente común es la falta de datos de entrenamiento.

[Defard y col., 2020] propone un modelo denominado PaDiM, el cual hace uso de la técnica de transferencia de aprendizaje. La transferencia de aprendizaje, hablando de manera superficial, consta de, reutilizar los conocimientos adquiridos de un modelo que resuelve un determinado problema, para resolver otro problema relativamente parecido. Es así como en [Defard y col., 2020] se resuelve el problema de la escasez de datos de entrenamiento.

Los pasos que PaDiM realiza, son descritos de manera burda a continuación:

- Se adquieren N imágenes de entrenamiento, las cuales son N fotografías tomadas en el mismo ángulo y posición tanto de la cámara como del objeto en buen estado que se desea examinar.
- La imagen es dividida en parches.
- Se utiliza una red neuronal convolucional previamente entrenada en la misma tarea, para generar n vectores característicos, por cada parche de la imagen.
- Se calcula la media y covarianza de los n vectores característicos por cada parche, generando así una matriz de parámetros Gaussianos.
- Dada una imagen de prueba, se repiten los primeros 3 puntos generando así un vector característico por cada parche de la imagen.
- Se utiliza la distancia de Mahalanobis para calcular la distancia entre el vector característico de la imagen de prueba y la matriz de parámetros Gaussianos.

- Finalmente la matriz de distancias es usada para formar un mapa de anomalías, en donde valores elevados indican áreas con alguna problemática.

Es así como PaDiM, además de detectar errores de manufactura, gracias a la división de las imágenes en parches, identifica en que parte de la imagen se encuentra el error, la figura 11 ilustra de manera visual algunos de los puntos antes mencionados.

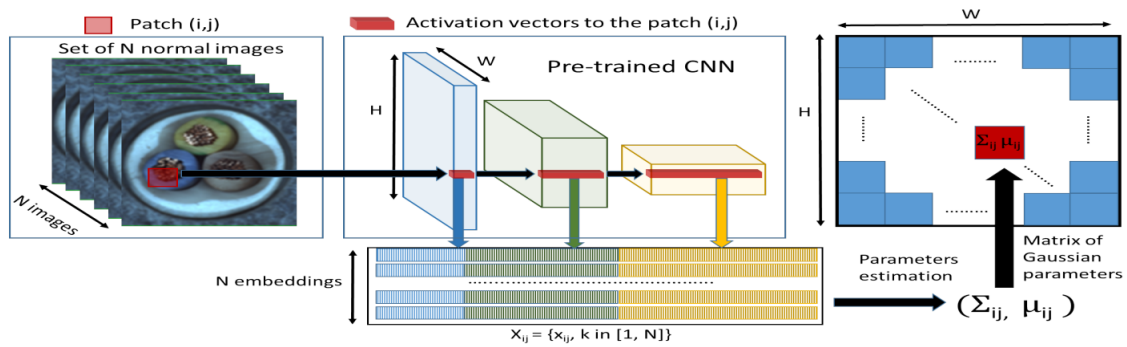


Figura 11: Metodología para la obtención de la matriz de parámetros Gaussianos. Imagen extraída de [Defard y col., 2020].

Las ventajas del método son:

- PaDiM toma en cuenta la correlación entre los diferentes niveles semánticos de la red convolucional previamente entrenada.
- Se experimentó con diferentes métodos de reducción de dimensiones, para finalmente seleccionar la que generará mejores resultados.
- Escalabilidad para diferentes tipos de objetos y texturas.
- PaDiM no sufre problemas ocasionados por variaciones de la iluminación cuando las fotografías son capturadas.

La principal desventaja es que:

- Los objetos no pueden tener cambios drásticos en su posición respecto a la cámara cuando las fotografías son obtenidas.

2.2.2 Detección de defectos de superficies metálicas mediante el uso de un *Auto-Encoder*.

Con la llegada de las redes neuronales convolucionales se desarrollaron nuevos modelos inteligentes que sugieren tener un mejor desempeño en el área de visión por computadora, en comparación con el desempeño de una red neuronal convencional.

Los *auto-encoders* (AE) pueden ser utilizados para la codificación y reconstrucción de datos, esto usando dos redes, la primera se encarga de una transformación de datos mediante la cual una imagen de entrada se convierte en una imagen o matriz de características multidimensionales y la segunda red puede ser diseñada para que, a partir de la imagen de características se genere una nueva imagen de las mismas dimensiones que la imagen de entrada original.

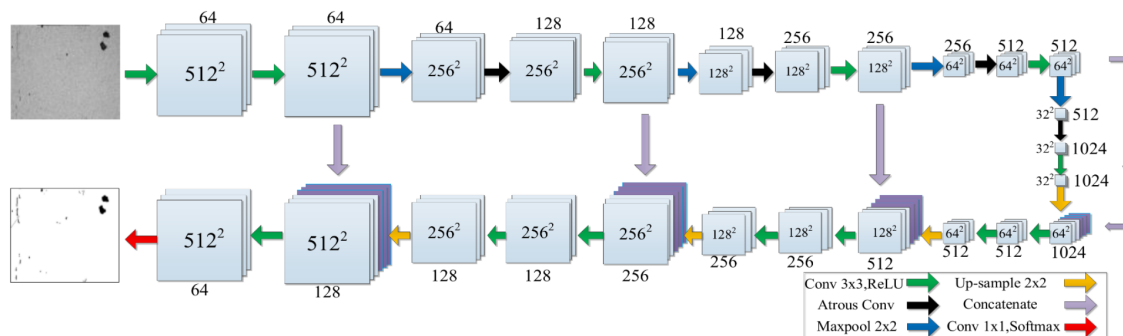


Figura 12: Arquitectura de la red *auto-encoder* (AE). Imagen extraída de [Tao y col., 2018].

En [Tao y col., 2018] se hace uso de dos *auto-encoders* con una arquitectura de cascada, en donde la salida del primer AE se convierte en la entrada del segundo AE, ambos AE comparten la misma arquitectura, añadiendo en la capa final una convolución 1×1 con una capa *softmax* para transformar las imágenes de salida de cada AE en mapas de probabilidad.

Los mapas de probabilidad pasan a un módulo de umbralización que identifica los píxeles que son mayores y menores al umbral para asignarles el número 1 o 0 respectivamente, de este modo, el número 0 representa un pixel con anomalías en la superficie del metal y 1 representa un pixel normal.

Al tener un buen desempeño para la identificación de anomalías, posteriormente se ataca el problema de clasificación de defectos, para el cual se pueden consultar los detalles en [Tao y col., 2018] la siguiente figura ilustra la arquitectura del modelo utilizado por [Tao y col.,

2018].

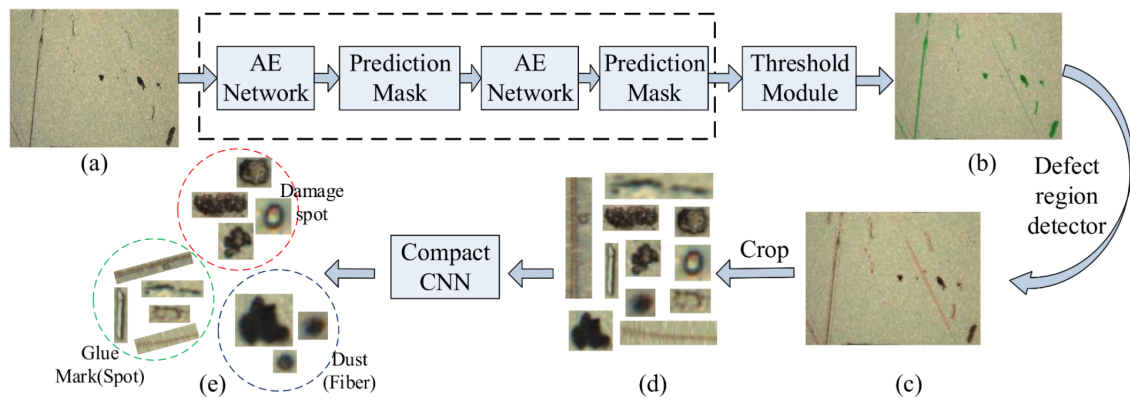


Figura 13: Arquitectura del modelo propuesto. Imagen extraída de [Tao y col., 2018].

Entre las ventajas, se tienen las siguientes:

- El puntaje máximo para la métrica utilizada fue de 89.60 %
- Los resultados experimentales sugieren que el algoritmo de detección es suficiente para cumplir con los requisitos del complejo entorno industrial.

La principal problemática que tiene es que:

- El entrenamiento de la red profunda requiere datos etiquetados manualmente, lo que requiere mucho tiempo y dinero.

2.2.3 ANOMALIB: Una librería para la detección de anomalías basada en aprendizaje profundo.

En los últimos años el campo de detección de anomalías a crecido considerablemente, tanto en la industria como en la investigación científica, teniendo como resultado un aumento considerable de trabajos y conjuntos de datos que atacan el problema utilizando diferentes técnicas y herramientas. La siguiente figura ilustra el número de conjuntos de datos y publicaciones científicas recientemente publicados en la literatura.

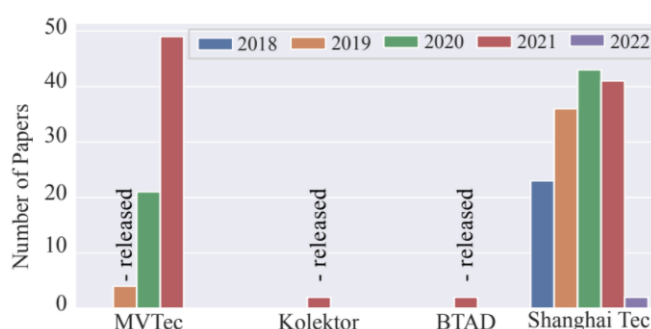


Figura 14: Número de conjuntos de datos y publicaciones científicas recientemente publicados en la literatura. Imagen extraída de [Akçay y col., 2022].

Si bien, existen diversas implementaciones o librerías que atacan el problema en mención, estas en su mayoría se enfocan en un algoritmo únicamente. La librería *Anomalib* propuesta en [Akçay y col., 2022] surge de la necesidad de implementar diversos algoritmos que sugieren tener un gran rendimiento en comparación con el estado del arte y a su vez tener la capacidad de incorporar nuevos trabajos para tener una librería dotada de algunas de las mejores técnicas de la actualidad para enfrentar el problema de detección de anomalías.

En la siguiente ilustración podemos observar la arquitectura de *Anomalib*:

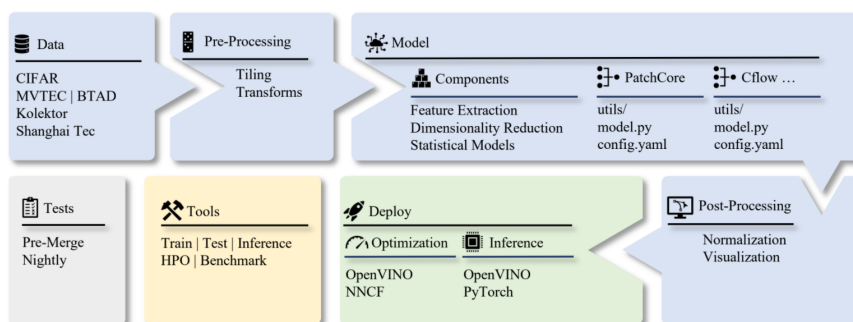


Figura 15: Arquitectura de *Anomalib*. Imagen extraída de [Akçay y col., 2022].

Las ventajas de la librería *Anomalib* son:

- Todos los componentes de los modelos se implementan en *PyTorch*, lo que permite ejecutar todas las operaciones en la GPU y exportar los modelos a *ONNX* y *OpenVINO*.
- *Anomalib* sigue cuatro principios de diseño: reproducibilidad, extensibilidad, modularidad y rendimiento en tiempo real.
- *Anomalib* provee conjuntos de datos tanto de imágenes como video.
- *Anomalib* contiene diversas métricas para medir los desempeños de los modelos.

Su principal inconveniente es que:

- La librería *Anomalib* y el trabajo [Akçay y col., 2022] fueron lanzados en el año 2022, por lo cual, al día de hoy sigue en desarrollo, mejoras y reparaciones deben realizarse, pues, aunque *Anomalib* soporte el uso de videos en sus conjuntos de datos, estos únicamente funcionan a nivel de *frames*. Actualmente se está trabajando para el soporte de detección de anomalías utilizando videos.

3 METODOLOGÍA DE SOLUCIÓN

Como se pudo observar previamente en el capítulo 2, muchos modelos existentes en la literatura tienen buenos resultados al enfrentarse al problema de detección automática de objetos con errores de manufactura, es por eso que, para el desarrollo de este proyecto, se plantea la modificación del modelo PaDiM propuesto en [Defard y col., 2020], reemplazando las imágenes RGB por imágenes RGB+D.

3.1 EL CONJUNTO DE DATOS

[Defard y col., 2020] señala que algunos modelos para la detección de anomalías, contienen en su conjunto de datos de entrenamiento únicamente imágenes provenientes de la clase normal, es decir imágenes correspondientes a objetos sin anomalías, de igual manera dado este enfoque no es necesaria una gran cantidad de datos de entrenamiento ni conocer la totalidad de anomalías que podría tener el objeto. Así imágenes que difieren del conjunto de entrenamiento, son consideradas imágenes con anomalías, PaDiM conserva el enfoque anterior.

En adición [Defard y col., 2020] concluye que, PaDiM puede ser robusto en conjuntos de datos reales, por lo que, para este proyecto el conjunto de datos pueden ser fotografías que contengan, desde piezas mecánicas muy complejas, hasta una hoja de papel.

Se utilizará una cámara *Intel RealSense D435* para capturar n fotografías y generar los valores de profundidad correspondientes para el conjunto de entrenamiento.

Cada fotografía debe tomarse conservando lo más posible la orientación del objeto con respecto a la cámara, con una resolución configurada de 1280×720 pixeles, tanto para la imagen RGB como para el canal de profundidad. El sensor de profundidad se configuró en *High Accuracy*.

La siguiente figura contiene muestras de fotografías en las que PaDiM fue puesto a prueba.

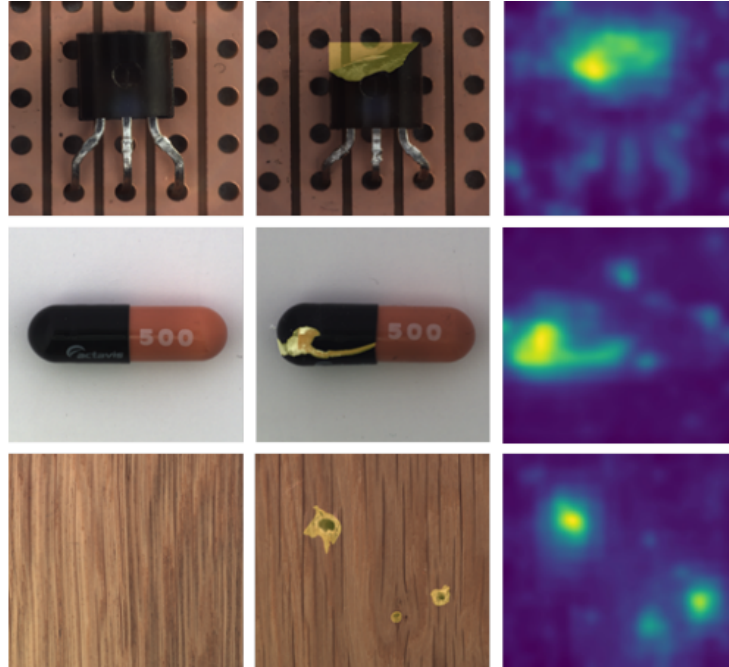


Figura 16: Columna izquierda: imágenes normales. Columna central: imágenes de las mismas clases con anomalías resaltadas en amarillo. Columna derecha: mapas de calor de anomalías obtenidos por el modelo PaDiM. Imagen extraída de [Defard y col., 2020].

Para los propósitos de este trabajo se generaron dos bases de datos, mediante los cuales el modelo resultante será implementado. Ambas bases de datos son imágenes capturadas por la cámara *Intel RealSense D435*, para cada imagen capturada su respectivo mapa de profundidad se encuentra en la base de datos.

Cada base de datos contiene 120 imágenes normales RGB, 50 imágenes seleccionadas aleatoriamente se reservan para el entrenamiento del modelo, 50 imágenes seleccionadas aleatoriamente se reservan para el cálculo del umbral del modelo y 20 imágenes seleccionadas aleatoriamente se reservan para medir el desempeño del modelo en la etapa de pruebas. Adicionalmente cada conjunto de datos contiene 7 imágenes anormales para la etapa de pruebas, recordando que por cada imagen se tiene su respectivo mapa de profundidad, se tienen 127 imágenes RGB más 127 imágenes/mapas de profundidad dando un total de 254 imágenes por conjunto de datos, Finalmente para cada imagen anormal, se tiene su *ground truth* (GT).

3.1.1 Conjunto de datos 1

El primer conjunto de datos, contiene fotografías de una palanca de velocidades para un automóvil. Para generar las imágenes de prueba, de la clase anormal, se hicieron modificaciones a la pieza original, añadiendo objetos que no pertenecen a la pieza o removiendo diferentes partes. La siguiente figura muestra un ejemplo de las fotografías que se pueden obtener mediante la cámara *Intel RealSense D435*.

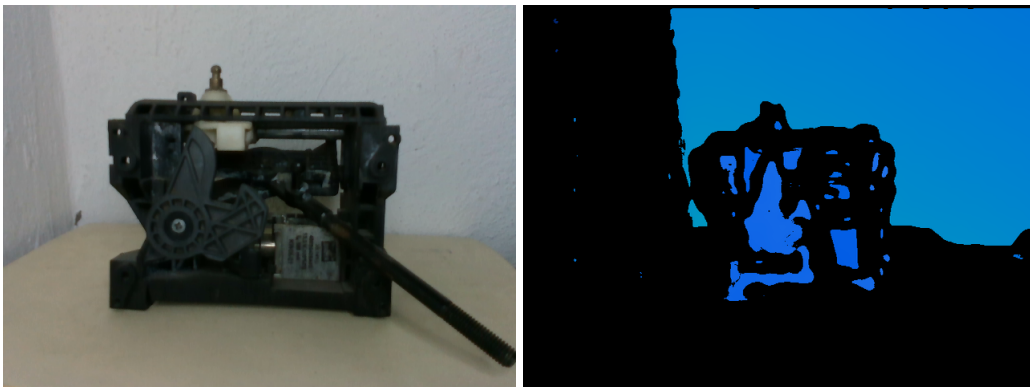


Figura 17: Ejemplo de imágenes obtenidas con la cámara *Intel RealSense D435*.

3.1.2 Conjunto de datos 2

El segundo conjunto de datos, contiene fotografías de una bomba hidráulica de clutch para un automóvil, al igual que en el conjunto de datos 1, para generar las imágenes de prueba, de la clase anormal, se hicieron modificaciones a la pieza original, añadiendo objetos que no pertenecen a la pieza o removiendo partes del objeto original, la siguiente figura muestra un ejemplo de las fotografías que se pueden obtener mediante la cámara *Intel RealSense D435*.

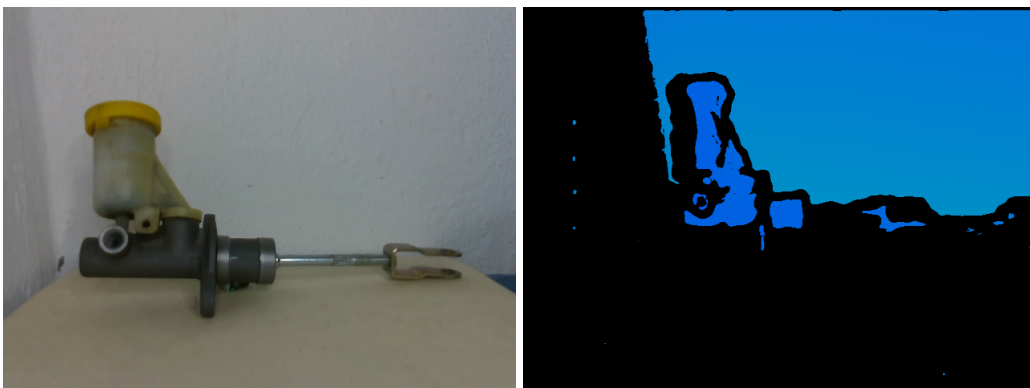


Figura 18: Ejemplo de imágenes obtenidas con la cámara *Intel RealSense D435*.

Como se puede apreciar en las dos subsecciones anteriores, los mapas de profundidad contienen ruido, haciendo que no sean precisos. En una tarea como la detección de anomalías no se puede tener ruido en las imágenes, pues este ruido entorpecería el desempeño del modelo.

Debido a lo anterior, en la siguiente sección se plantea el uso del modelo *Boosting Monocular Depth* propuesto en [Miangoleh y col., 2021] para la generación de los mapas de profundidad correspondientes, ya que es un referente en el estado del arte debido a su precisión en los detalles finos.

3.1.3 Boosting Monocular Depth

Dado que los mapas de profundidad obtenidos por la cámara *Intel RealSense D435* no son aptas para el desarrollo de este proyecto, el modelo *Boosting Monocular Depth* propuesto en [Miangoleh y col., 2021] fue utilizado para la generación de los mapas de profundidad a partir de las imágenes RGB. En la imagen 19 se puede observar un mapa creado por este método.

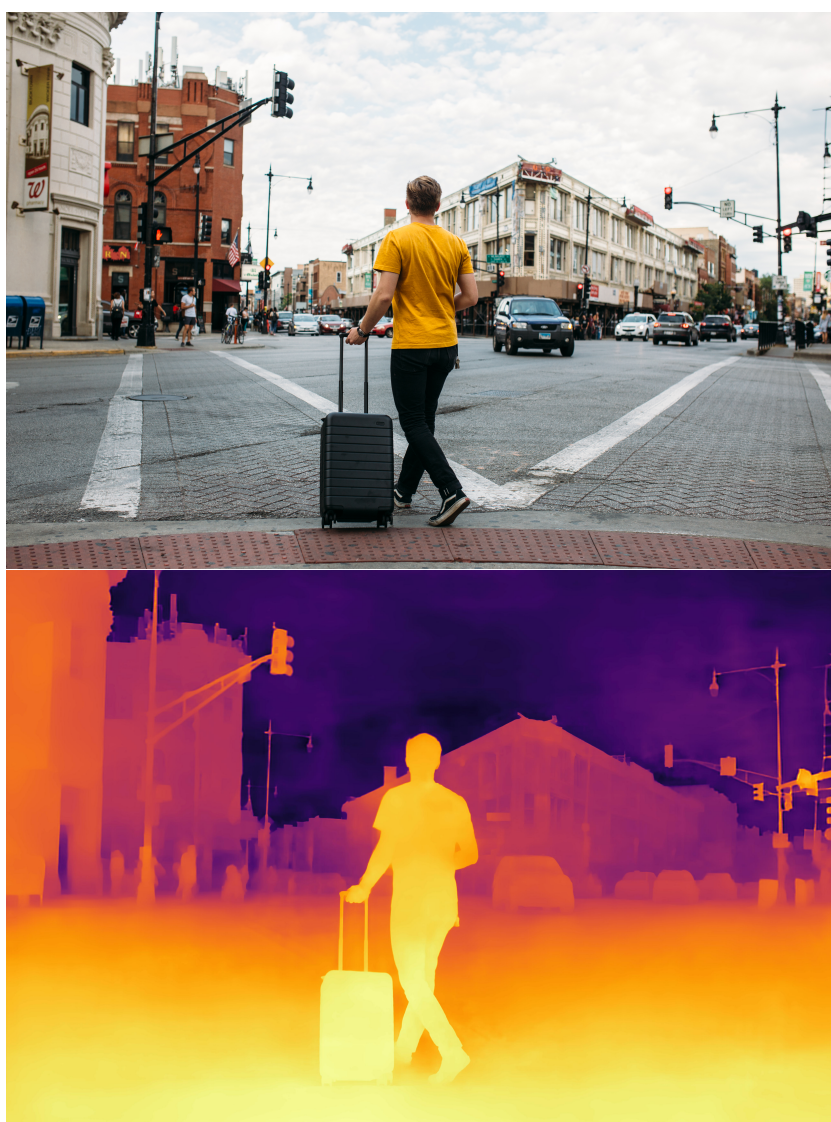


Figura 19: Ejemplo de la obtención de un mapa de profundidad a partir de una imagen RGB obtenida con el modelo *Boosting Monocular Depth*.

Boosting Monocular Depth es un modelo de aprendizaje automático previamente entrenado para la generación de mapas de profundidad a partir de imágenes RGB, está diseñado para trabajar bajo tres modelos distintos para la estimación de los mapas de profundidad, en

particular utilizaremos el modelo *LeReS* [Yin y col., 2021]. En la siguiente figura se muestran los resultados obtenidos por el modelo *Boosting Monocular Depth* en los conjuntos de datos uno y dos.

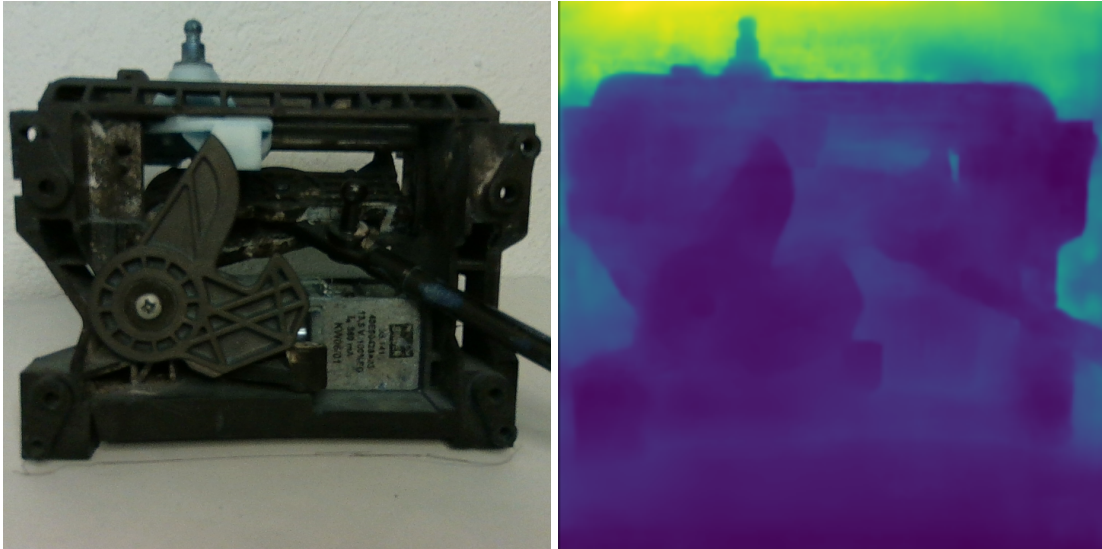


Figura 20: Ejemplo de imagen RGB e imagen de profundidad obtenido mediante *Boosting Monocular Depth*.

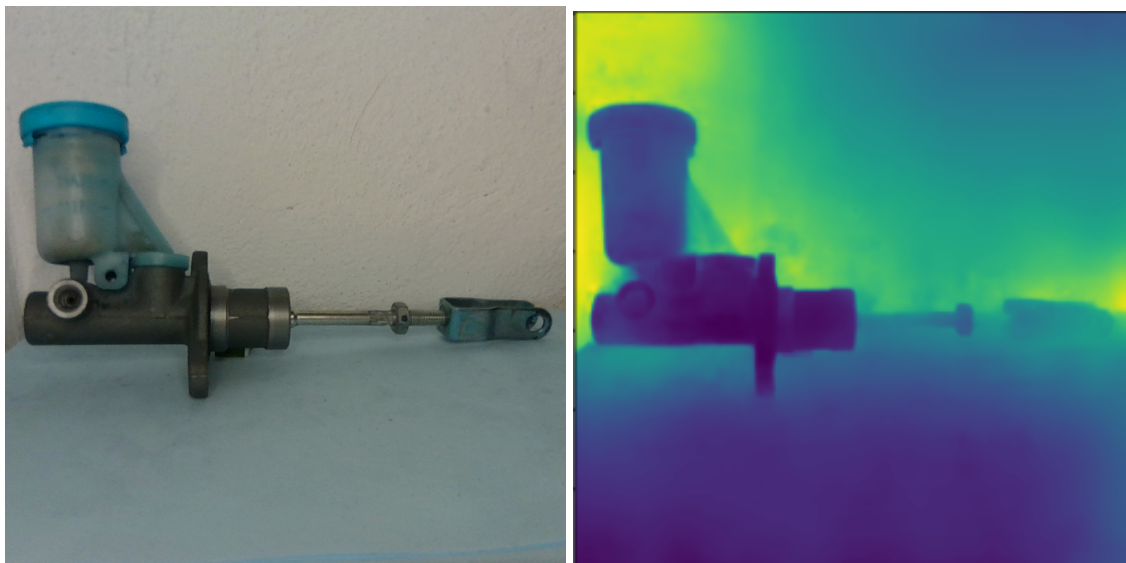


Figura 21: Ejemplo de imagen RGB e imagen de profundidad obtenido mediante *Boosting Monocular Depth*.

Dada una simple inspección visual, podemos apreciar que los mapas de profundidad obtenidos por el modelo *Boosting Monocular Depth* contienen una cantidad significativa menor de ruido en comparación con los mapas de profundidad obtenidos con la cámara *Intel RealSense* por lo que, se tomó la decisión de que los mapas serían generados por el modelo *Boosting Monocular Depth*.

3.2 DESCRIPCIÓN DE LA CNN

El modelo PaDiM hace uso de un modelo de distribución de *patches* en conjunto con una red neuronal convolucional (CNN) pre-entrenada para generar vectores característicos de cada imagen de entrenamiento, en adición, [Defard y col., 2020] afirma lo siguiente:

- Cada *patch* es descrito por una distribución Gaussiana multivariable.
- PaDiM toma en cuenta la correlación entre diferentes niveles semánticos de la CNN.

Como se mencionó anteriormente, PaDiM hace uso de una CNN previamente entrenada, para el desarrollo de este proyecto se utilizará la CNN ResNet50V2 [He y col., 2016], la cual fue entrenada en el conjunto de datos ImageNet, la siguiente figura ilustra la arquitectura de la CNN ResNet50V2.

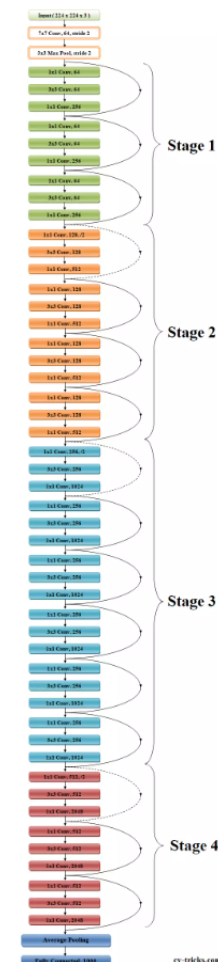


Figura 22: Arquitectura de la CNN ResNet50V2.

3.3 EXTRACCIÓN DE CARACTERÍSTICAS

Al tener imágenes con resoluciones de 1280×720 píxeles, se procede a recortar las imágenes a una resolución de 720×720 píxeles conservando el centro de la imagen original, teniendo así una imagen cuadrada. La información de profundidad cuenta con un solo canal, por lo que se triplicó dicho canal, para generar una imagen de 3 canales, generando como resultado, por cada imagen RGB, su correspondiente imagen de profundidad.

Posteriormente las imágenes son reescaladas a otra resolución la cual es calculada como sigue: $resolución = (224 * patches, 224 * patches, 3)$ en donde, $patches$ corresponde al número de $patches$ que se utilizarán por renglón, en secciones posteriores se analizará el uso de distintos números de $patches$.

De acuerdo a la sección anterior, la arquitectura de la red permite el uso de imágenes con una resolución de $224 \times 224 \times 3$, por lo que, para cada imagen RGB cada $patch\ patch_{rgb}(i, j)^k$ conservara la resolución antes mencionada.

En la siguiente figura se muestra la manera en que cada $patch$ será generado a partir de una imagen dada, para el caso $dim = 4$.

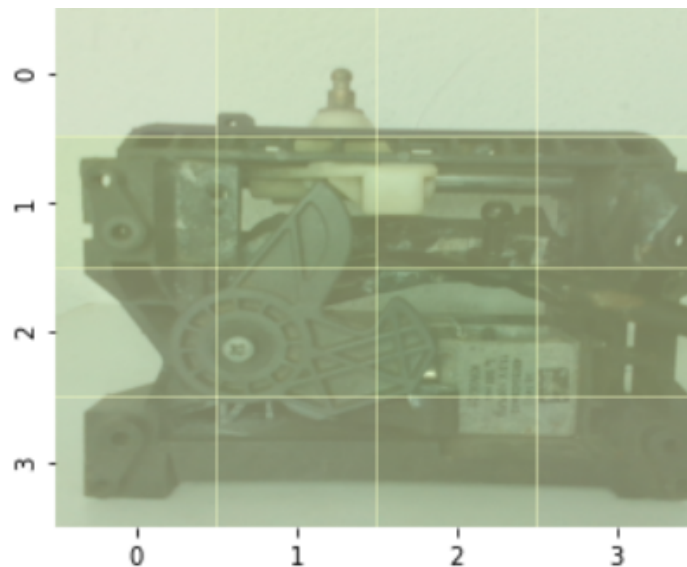


Figura 23: Generación de $patches$ a partir de una imagen.

Cada $patch$ corresponde a un cuadrante de la imagen el cual no contiene píxeles de otro cuadrante. Para las imágenes de profundidad los $patches\ patch_d(i, j)^k$ serán generados del mismo modo.

Durante la fase de entrenamiento, cada *patch* es introducido como entrada a la CNN, primeramente y con base en los resultados obtenidos por [Defard y col., 2020], son seleccionadas las 3 primeras capas en donde la información residual regresa al flujo de información. Dado el mapa de activaciones de la CNN, únicamente las salidas de las capas anteriormente seleccionadas son extraídas para ajustar su tamaño individual a un vector, teniendo así 3 vectores característicos los cuales son concatenados dando lugar a un único vector por cada *patch* de una imagen.

El procedimiento antes descrito se realiza tanto para las n imágenes RGB, como para las n imágenes de profundidad, posteriormente se aplica una reducción de dimensiones a una dimensión final 1500 a los vectores característicos $x_{(i,j)}^k$, los cuales son el resultado de concatenar los vectores generados por el $patch_{rgb}(i, j)^k$ de la imagen k RGB y el $patch_d(i, j)^k$ de la imagen de profundidad correspondiente, en el orden, RGB+D. Finalmente los parámetros gaussianos son calculados dando lugar a una matriz de parámetros gaussianos, en secciones posteriores se estudiará el comportamiento del modelo con diferentes valores para 1500.

[Defard y col., 2020] notó, que una reducción aleatoria de dimensiones a los vectores $x_{(i,j)}^k$ genera mejores clasificaciones que un análisis de componentes principales (PCA), por lo que, se optó por aplicar de igual manera, una reducción de dimensiones aleatoria.

La siguiente figura ilustra la manera en que [Defard y col., 2020] genera los n vectores característicos correspondientes a las imágenes RGB, los cuales son utilizados para calcular los parámetros gaussianos, para propósitos de este proyecto y como se mencionó anteriormente, un paso intermedio es generar los vectores característicos de cada $patch_d(i, j)^k$ de las imágenes de profundidad, para entonces concatenarlos al vector correspondiente al $patch_{rgb}(i, j)^k$ antes de la reducción aleatoria y el calculo de los parámetros gaussianos.

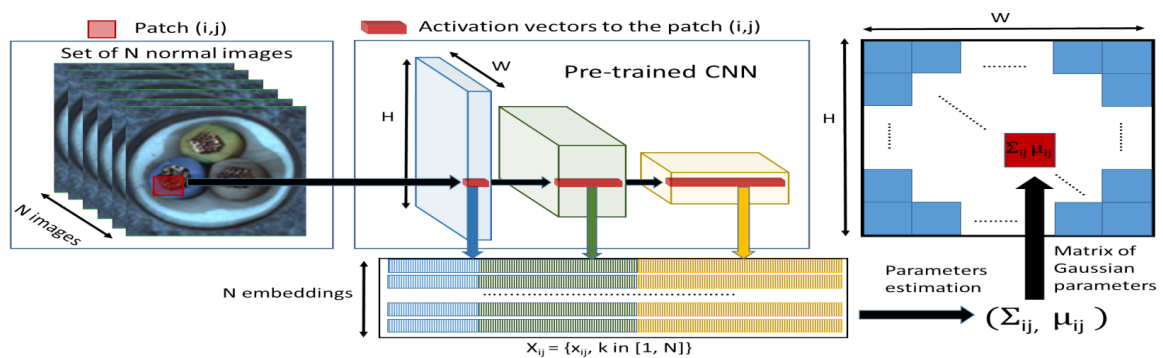


Figura 24: Extracción de vectores característicos para generar una matriz de parámetros gaussianos. Imagen extraída de [Defard y col., 2020].

3.4 APRENDIENDO DE LA NORMALIDAD: LA MATRIZ DE PARÁMETROS GAUSSIANOS

De acuerdo con [Defard y col., 2020] para aprender las características correspondientes al cuadrante (i, j) , es necesario generar el conjunto $X_{(i,j)} = \{x_{(i,j)}^k\}$ con $k \in 1, 2, \dots, s, \dots, n$, esto es generar la cantidad de *patches*² conjuntos, cada conjunto contiene n vectores característicos generados como se describe en la sección anterior, correspondientes al cuadrante (i, j) .

Una vez obtenidos estos conjuntos y para calcular la matriz de parámetros gaussianos $N(\mu_{(i,j)}, \Sigma_{(i,j)})$, se calcula el vector medio $\mu_{(i,j)}$, el cual contiene, en cada posición $i \in 1, \dots, 1500$ el valor medio de todos los vectores en el conjunto en la posición i , por su parte, $\Sigma_{(i,j)}$ es calculado con la formula (1) dada en [Defard y col., 2020]:

$$\Sigma_{(i,j)} = 1/(n - 1) \sum_{k=1}^n (x_{(i,j)}^k - \mu_{(i,j)})(x_{(i,j)}^k - \mu_{(i,j)})^T + \epsilon I \quad (1)$$

en donde $\epsilon = 0.01$ e I es una matriz identidad de dimensiones $(1500, 1500)$.

Con lo anterior, la etapa de entrenamiento queda completada junto a la matriz $N = N(\mu_{(i,j)}, \Sigma_{(i,j)})$ en donde N tiene las dimensiones $(patches, patches)$.

3.5 MAPA DE ANOMALÍAS Y LA DISTANCIA DE MAHALANOBIS

Como lo hace notar [Defard y col., 2020] la distancia de Mahalanobis $M(x_{(i,j)})$ puede ser interpretada como la distancia entre el vector $x_{(i,j)}$ correspondiente a las imágenes RGB y de profundidad de prueba y la distribución aprendida $N(\mu_{(i,j)}, \Sigma_{(i,j)})$, en donde $M(x_{(i,j)})$ se define en (2).

$$M(x_{(i,j)}) = \sqrt{(x_{(i,j)} - \mu_{(i,j)})^T \Sigma_{(i,j)}^{-1} (x_{(i,j)} - \mu_{(i,j)})} \quad (2)$$

Con lo cual, la matriz de distancias de Mahalanobis $M = M(x_{(i,j)})$ que forma un mapa de anomalías es calculada. Puntajes altos en este mapa indican las áreas con anomalías.

3.6 CALCULO DEL UMBRAL PARA UN ANÁLISIS CUALITATIVO

Para efectos de este trabajo, se busca realizar una evaluación cualitativa y cuantitativa, en esta sección se propondrá el cálculo de un umbral, para realizar la evaluación cualitativa del modelo.

Se propone, para el calculo del Umbral, la generación de las distancias de Mahalanobis para 50 imágenes de la clase normal junto a sus mapas de profundidad, de esta manera se tendrá una matriz $umbral = M(x_{(i,j,l)})$ con $l \in 1, 2, \dots, 50$, esto es una matriz con dimensiones $(i, j, 50)$. Posteriormente por cada parche (i, j) se calculara el valor máximo en la matriz $umbral$ manteniendo fija la posición (i, j) y variando la posición l , lo cual es el valor máximo de la distancia de Mahalanobis generado para las 50 imágenes de profundidad por cada parche (i, j) , teniendo como resultante una matriz $t = M_{max}(x_{(i,j)})$ con dimensiones (i, j) en donde, cada posición (i, j) representa el valor máximo que una imagen normal puede alcanzar en la distancia de Mahalanobis hasta el momento para el parche (i, j) .

4 PRUEBAS Y RESULTADOS

En esta sección se discutirá el desempeño del modelo final de manera cualitativa y cuantitativa en las bases de datos 1 y 2, de igual manera se analizará el desempeño del modelo en comparación con el modelo empleando únicamente imágenes RGB. Así como también se discutirán las limitantes del modelo y sus ventajas.

Como se mencionó en la sección de la metodología, cada conjunto de datos cuenta con un total de 254 imágenes, las cuales están divididas en 100 imágenes de entrenamiento, 100 imágenes para generar el umbral y 54 imágenes para realizar las pruebas y medir el desempeño del modelo, recordando que el número de imágenes anterior contempla los mapas de profundidad de cada imagen RGB.

En adición se realizaron pruebas contemplando diferentes números de parches, esto para tener una detección más precisa, pues el modelo realiza una detección a nivel de parches mientras que las máscaras GT se encuentran a nivel píxel, por esto se realizaron dos medidas de desempeño por cada métrica a usar, la primera considerando el GT a nivel píxel y la segunda a nivel parche.

La figura 25 muestra un ejemplo de la comparación entre la predicción del modelo, los GT a nivel píxel y parche utilizando el Umbral establecido en el capítulo anterior.

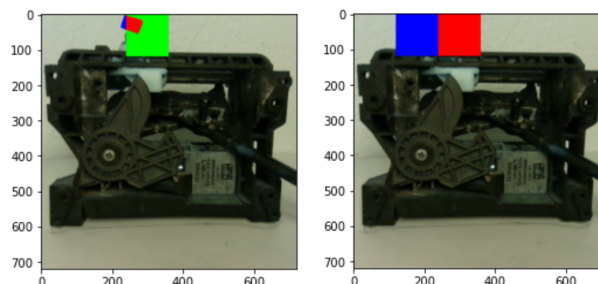


Figura 25: Comparación de predicción con el GT a nivel píxel y parche, en donde el color azul representa los píxeles dañados indicados por el GT, el color verde representa los parches dañados indicados por el modelo pero que no son ciertas y el color rojo la intersección de la predicción con el GT.

4.1 MÉTRICAS DE DESEMPEÑO

4.1.1 Matriz de confusión

La matriz de confusión es una herramienta que otorga información relevante acerca de las clasificaciones que un modelo obtiene al ser probado, la siguiente figura ilustra el diseño de una matriz de confusión para nuestro caso particular.

		Clasificado como		Realmente es	
		Normal	Anormal	Normal	Anormal
	TP		FN		
	FP		TN		

Figura 26: Matriz de confusión para la tarea detección de anomalías.

Observando la figura anterior podemos apreciar que la matriz de confusión cuenta con 4 elementos clave, los cuales son TP, FN, FP, TN, en donde:

- TP indica el número de píxeles que el modelo clasificó como normales y efectivamente son normales.
- FN indica el número de píxeles que el modelo clasificó como anormales, pero en realidad son normales.
- FP indica el número de píxeles que el modelo clasificó como normales, pero en realidad son anormales.
- TN indica el número de píxeles que el modelo clasificó como anormales y efectivamente son normales.

4.1.2 Precisión - recuerdo - especificidad

Si bien la matriz de confusión otorga información relevante sobre las clasificaciones, puede ser utilizada como base para generar métricas más particulares. Tres métricas que ayudan a entender el desempeño de nuestro modelo son:

$$precision = \frac{TP}{TP + FP} \quad (3)$$

$$recuerdo = \frac{TP}{TP + FN} \quad (4)$$

$$especificidad = \frac{FP}{FP + TN} \quad (5)$$

Note que a menor número de FP es mayor la precisión y a menor número de FN el recuerdo es mayor.

4.1.3 Curvas ROC - curvas PR

En una clasificación binaria, los modelos de predicción suelen trabajar mediante probabilidades, esto para indicar la probabilidad de pertenecer a la clase positiva. Las curvas ROC representan la relación entre el recuerdo (eje y) y $1-especificidad$ (eje x) en diferentes umbrales que indiquen cuándo la probabilidad es suficiente para pertenecer a la clase positiva, ver figura 27.

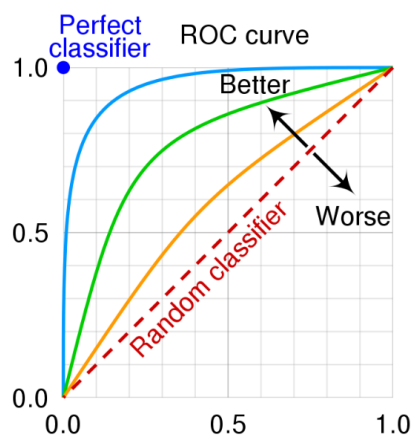


Figura 27: Ejemplo de curva ROC.

Por otro lado, la curva PR representa la relación entre la precisión (eje y) y el recuerdo (eje x) en diferentes umbrales que indiquen cuándo la probabilidad es suficiente para pertenecer a la clase positiva, ver figura 28.

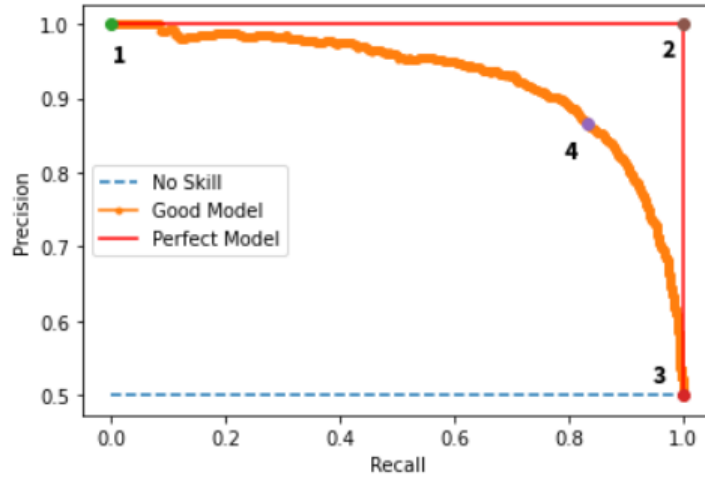


Figura 28: Ejemplo de curva PR.

4.1.4 *AUC score - PR score*

En la práctica, las curvas ROC y PR son utilizadas para generar dos métricas de desempeño, las cuales son *AUC score* y *PR score*, respectivamente, ambas métricas parten de la misma base, calcular el área bajo su respectiva curva, con la diferencia en que, un *AUC score* = 0.5 representa un modelo incapaz de distinguir entre clase positiva y clase negativa, mientras que en *PR score* se representa con un valor de 0.0.

[Branco y col., 2015] señala que, *PR score* es altamente recomendado para atacar problemas de clases no balanceadas, pues en esas situaciones *AUC score* puede ser poco fiable tendiendo a obtener resultados optimistas.

Para realizar el análisis cuantitativo del modelo, se emplearán las métricas ROC score y *PR score* descritas en la sección anterior, note que en nuestro caso particular, al tratarse de una clasificación de píxeles las clases no se encuentran balanceadas.

4.1.5 *Matthews correlation coefficient (MCC)*

Como se menciona en [Abhishek y Hamarneh, 2020], la métrica MCC indica la correlación entre las predicciones y las etiquetas del GT, además de no presentar complicaciones cuando se enfrenta a problemas de clases no balanceadas.

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (5)$$

Finalmente las métricas discutidas en esta sección serán utilizadas para realizar el análisis cuantitativo de los modelos resultantes.

4.1.6 Análisis cualitativo

4.1.6.1 Modelo 10×10 parches

La cantidad de 10×10 parches consta de 72×72 píxeles cada uno, las siguientes tablas muestran imágenes anormales del conjunto de datos 1, 2 y los resultados de los modelos RGB y RGB+D. Para tener una visualización más completa, se realizó una superposición de los resultados del modelo y el GT a nivel píxel, sobre la imagen RGB original,

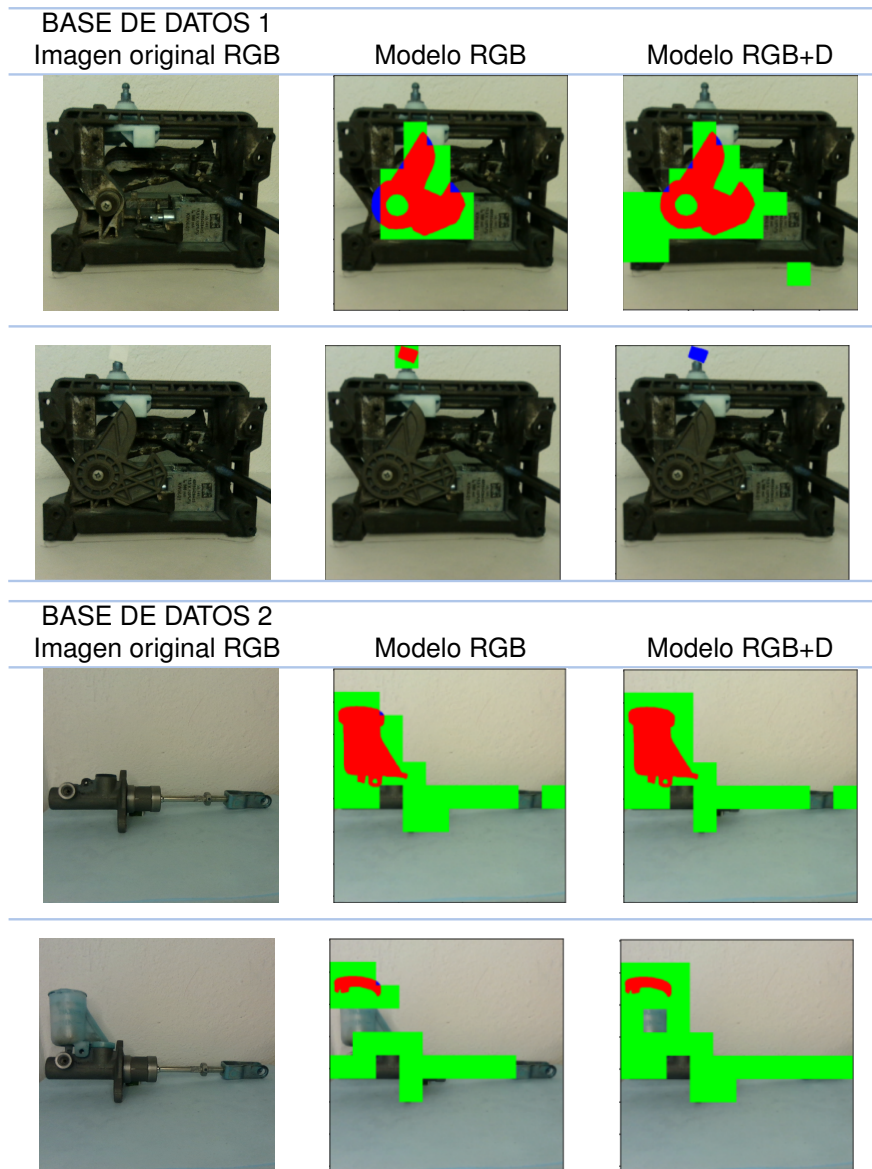


Tabla 1: Comparación de la predicción del modelo utilizando el umbral descrito en la sección 3.6 con GT a nivel píxel en las base de datos 1 y 2, en donde el color azul representa los píxeles dañados indicados por el GT, el color verde representa los parches dañados indicados por el modelo pero que no son ciertos y el color rojo la intersección de la predicción con el GT.

4.1.6.2 Modelo 8×8 parches

La cantidad de 8×8 parches consta de 90×90 píxeles cada uno, las siguientes tablas muestran imágenes anormales del conjunto de datos 1, 2 y los resultados de los modelos RGB y RGB+D. Para tener una visualización más completa, se llevó a cabo una superposición de los resultados del modelo y el GT a nivel píxel, sobre la imagen RGB original.

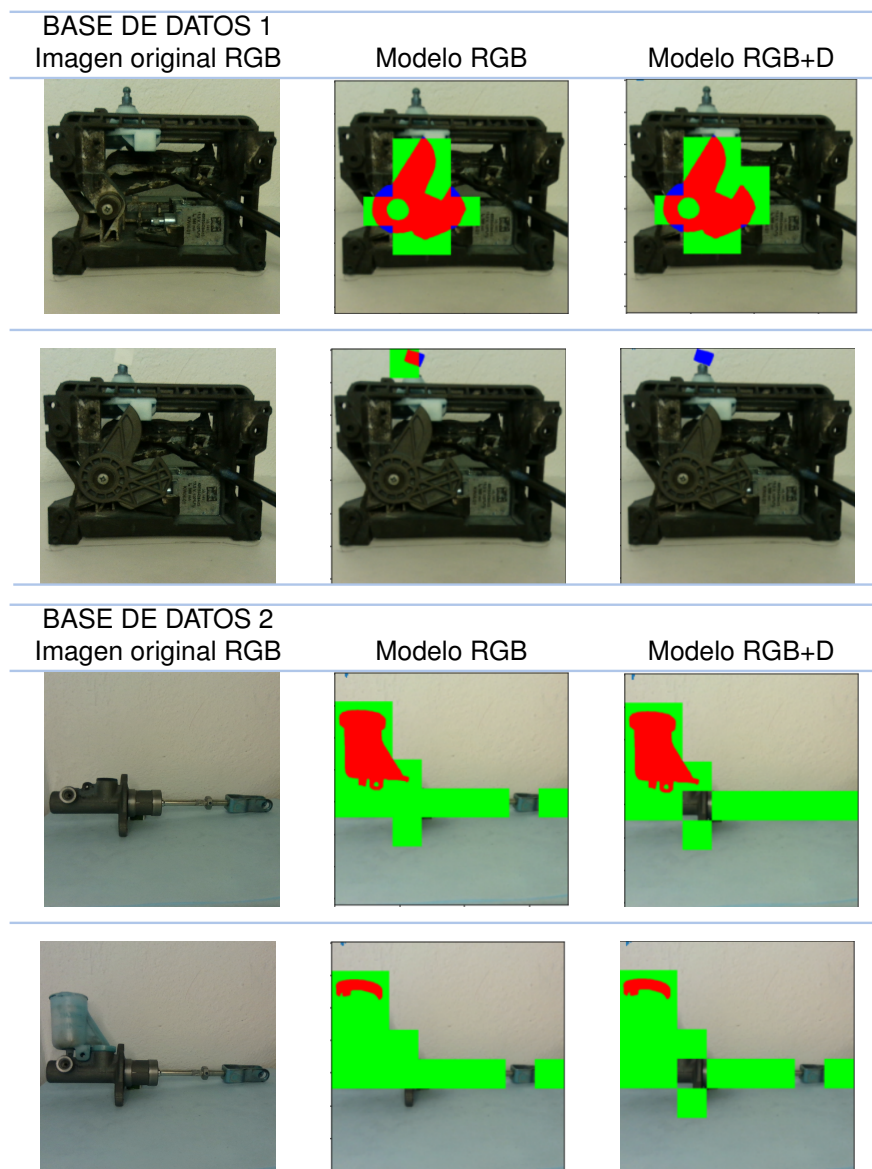


Tabla 2: Comparación de la predicción del modelo utilizando el umbral descrito en la sección 3.6 con GT a nivel píxel en la base de datos 1, en donde el color azul representa los píxeles dañados indicados por el GT, el color verde representa los parches dañados indicados por el modelo pero que no son ciertas y el color rojo la intersección de la predicción con el GT.

4.1.6.3 Modelo 6×6 parches

La cantidad de 6×6 parches consta de 122×122 píxeles cada uno, las siguientes tablas muestran imágenes anormales del conjunto de datos 1, 2 y los resultados de los modelos RGB y RGB+D. En la tabla 3 se muestra gráficamente una superposición de los resultados del modelo y el GT a nivel píxel, sobre la imagen RGB original.

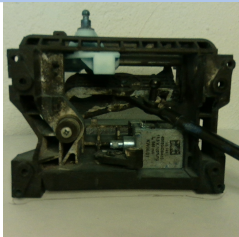
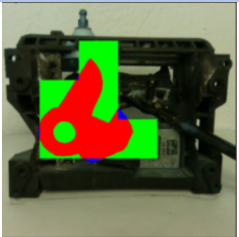
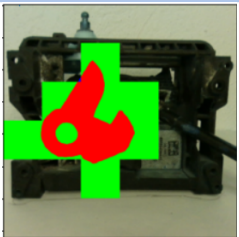
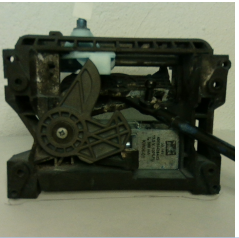
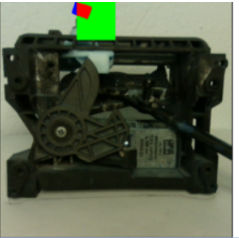
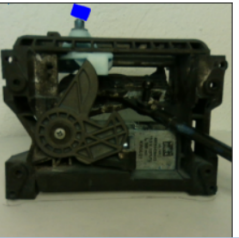

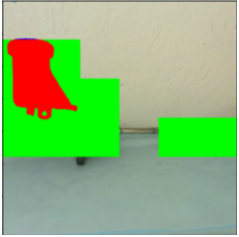
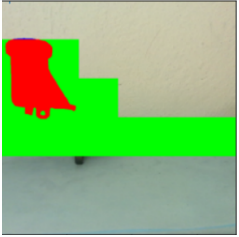

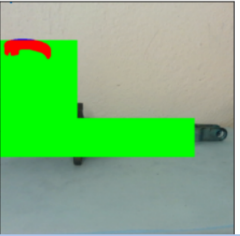
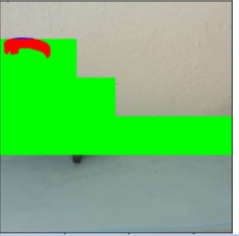
BASE DE DATOS 1		
Imagen original RGB	Modelo RGB	Modelo RGB+D
		
		
BASE DE DATOS 2		
Imagen original RGB	Modelo RGB	Modelo RGB+D
		
		

Tabla 3: Comparación de la predicción del modelo utilizando el umbral descrito en la sección 3.6 con GT a nivel píxel en la base de datos 1, en donde el color azul representa los píxeles dañados indicados por el GT, el color verde representa los parches dañados indicados por el modelo pero que no son ciertos y el color rojo la intersección de la predicción con el GT.

4.1.7 Análisis cuantitativo

En esta sección se expondrán los resultados cuantitativos de los modelos generados a lo largo de este trabajo utilizando las métricas descritas en secciones anteriores. El nombre de cada modelo hace referencia al uso de información RGB o RGB+D con el número de parches correspondientes.

Base de datos 1

	Métrica	Modelo RGB-6x6	RGB+D-6x6
<i>ground truth</i> a nivel píxel	AUC ROC SCORE	0.9590	0.9434
	AUC PR SCORE	0.4917	0.4844
	MCC	0.4093	0.3678
<i>ground truth</i> a nivel parche	AUC ROC SCORE	0.9448	0.9198
	AUC PR SCORE	0.7372	0.7326
	MCC	0.6457	0.59931

Base de datos 2

	Métrica	Modelo RGB-6x6	RGB+D-6x6
<i>ground truth</i> a nivel píxel	AUC ROC SCORE	0.9448	0.9430
	AUC PR SCORE	0.3632	0.3824
	MCC	0.2222	0.2051
<i>ground truth</i> a nivel parche	AUC ROC SCORE	0.8607	0.9174
	AUC PR SCORE	0.6319	0.5735
	MCC	0.3943	0.3551

Tabla 4: Desempeño de los modelos RGB y RGB+D en su versión de 6×6 en las bases de datos 1 y 2 utilizando las métricas *PR score* y *AUC score*.

Los resultados para el modelo en su versión de 6×6 muestran que la adición de información de profundidad no generan mejoras significativas a nivel píxel, teniendo un *score* de 95 % y 94 % de efectividad aproximadamente ante la métrica *AUC SCORE* en la base de datos 1, a su vez, en la base de datos 2 los resultados fueron iguales obteniendo 94 %, mientras que el *score* para la métrica *PR SCORE* fue de 49 % y 48 % en la base de datos 1, en la base de datos 2 se obtuvo un 36 % y 38 %, en ambas métricas el primer resultado de cada una es asociado al modelo RGB mientras que el segundo al modelo RGB+D de cada base de datos correspondiente. A un nivel de parche, la métrica *AUC SCORE* tuvo una variación mayor, obteniendo 94 % y 91 % para la base de datos 1, 86 % y 91 % para la base de datos 2 en el modelo RGB Y RGB+D respectivamente, por otro lado *PR SCORE* obtuvo los mismos resultados en la base de datos 1, teniendo un 73 % de efectividad, en la base de datos 2 se obtuvieron 63 % y 57 % de efectividad.

Base de datos 1

	Métrica	Modelo RGB-8x8	RGB+D-8x8
<i>ground truth</i> a nivel píxel	AUC ROC SCORE	0.9410	0.9369
	AUC PR SCORE	0.3988	0.4052
	MCC	0.3906	0.3424
<i>ground truth</i> a nivel parche	AUC ROC SCORE	0.9200	0.9132
	AUC PR SCORE	0.7741	0.7122
	MCC	0.6827	0.5463

Base de datos 2

	Métrica	Modelo RGB-8x8	RGB+D-8x8
<i>ground truth</i> a nivel píxel	AUC ROC SCORE	0.9563	0.9483
	AUC PR SCORE	0.4368	0.3363
	MCC	0.2700	0.2556
<i>ground truth</i> a nivel parche	AUC ROC SCORE	0.9657	0.9581
	AUC PR SCORE	0.7253	0.6189
	MCC	0.4335	0.4234

Tabla 5: Desempeño de los modelos RGB y RGB+D en su versión de 8×8 en las bases de datos 1 y 2 utilizando las métricas *PR score* y *AUC score*.

Para el modelo en su versión de 8×8 a nivel píxel se obtuvo un 94 % y 93 % de efectividad en la base de datos 1 y un 95 %, 94 % para la base de datos 2 utilizando la métrica *AUC SCORE*, ante la métrica *PR SCORE* se obtuvo un 39 %, 40 % en la base de datos 1 y un 43 %, 33 % de efectividad en la base de datos 2. A nivel parche en la métrica *AUC SCORE* se obtuvo un 92 %, 91 % en la base de datos 1 y un 96 %, 95 % para la base de datos 2, utilizando la métrica *PR SCORE* se obtuvo un 77 %, 71 % de efectividad en la base de datos 1 y un 72 %, 61 % en la base de datos 2.

Base de datos 1

	Métrica	Modelo RGB-10x10	RGB+D-10x10
<i>ground truth</i> a nivel píxel	AUC ROC SCORE	0.9784	0.9833
	AUC PR SCORE	0.5794	0.6111
	MCC	0.5040	0.3875
<i>ground truth</i> a nivel parche	AUC ROC SCORE	0.9676	0.9753
	AUC PR SCORE	0.7731	0.8032
	MCC	0.7307	0.5697

Base de datos 2

	Métrica	Modelo RGB-10x10	RGB+D-10x10
<i>ground truth</i> a nivel píxel	AUC ROC SCORE	0.9515	0.9545
	AUC PR SCORE	0.3462	0.3435
	MCC	0.3073	0.2767
<i>ground truth</i> a nivel parche	AUC ROC SCORE	0.9489	0.9457
	AUC PR SCORE	0.6160	0.6314
	MCC	0.5167	0.4828

Tabla 6: Desempeño de los modelos RGB y RGB+D en su versión de 10×10 en las bases de datos 1 y 2 utilizando las métricas *PR score* y *AUC score*.

Finalmente el modelo en su versión de 10×10 obtuvo a nivel píxel 97 %, 98 % de efectividad en la base de datos 1 y un 95 %, 95 % en la base de datos 2 usando la métrica *AUC score* y un 57 %, 61 % utilizando la métrica *PR score* en la base de datos 1, para la base de datos 2 se obtuvo en ambos modelos un 34 % de efectividad. A un nivel de parche se tiene para la métrica *AUC score* un 96 %, 97 % de efectividad en la base de datos 1 y un 94 %, 94 % para la base de datos 2, finalmente utilizando la métrica *PR score* se obtuvo una efectividad de 77 %, 80 % en la base de datos 1 y un 61 %, 63 % de efectividad en la base de datos 2.

Analizando la métrica MCC se nota que en ningún modelo hubo una mejoría al incorporar la información de profundidad, incluso en muchos de los casos la métrica arrojó peores resultados, de igual manera se puede notar que tanto la métrica MCC como la métrica *PR score* tuvieron mejores resultados en todos los modelos al comparar el GT a nivel parche.

5 CONCLUSIONES

Como se pudo apreciar en la sección 3.1 los mapas de profundidad obtenidos por la cámara *Intel Realsense D435* contenían ruido e información poco confiable haciendo inviable su uso para el desarrollo de nuevos modelos de aprendizaje automático, dado que en un problema como lo es la detección automática de errores de manufactura contener ruido en una imagen puede fácilmente señalar un error en donde no lo hay.

Considerando el párrafo anterior, podemos concluir que hasta la fecha, la combinación de la información extraída de una imagen a color y de su respectivo mapa de profundidad, obtenidos por medio de una cámara de bajo costo, no puede mejorar el proceso de detección automática de errores de manufactura en objetos.

Sin embargo, para el alcance de los objetivos de este trabajo, se introdujo el modelo *Boosting Monocular Depth* para la generación de los mapas de profundidad a partir de imágenes RGB obtenidos por una cámara de bajo costo. Estos mapas de profundidad mostraban tener mayor fiabilidad en comparación con los obtenidos por la cámara de bajo costo, por lo que se optó por utilizarlos para el desarrollo del sistema automático para detectar errores de manufactura.

De igual manera se realizó un estudio del estado del arte para seleccionar un modelo con un buen desempeño ante la misma tarea, modificando e incorporando de manera exitosa los mapas de profundidad. Finalmente se utilizaron dos métricas de desempeño para realizar un análisis cuantitativo de los resultados obtenidos, arrojando que la incorporación de los mapas de profundidad al modelo no mejoraron el desempeño de éste e incluso en algunos casos, el desempeño bajó.

Por lo anterior se concluye que, hasta el momento el uso de mapas de profundidad obtenidos por una cámara de bajo costo o la generación de estos utilizando modelos de aprendizaje automático a partir de imágenes RGB obtenidas por la misma cámara, no mejora el desempeño de los modelos originales que emplean únicamente imágenes RGB.

No obstante, hay más caminos para la incorporación de mapas de profundidad a los modelos de aprendizaje automático que se encuentran en el estado del arte, como el uso de imágenes RGB de una calidad superior para la generación de los mapas de profundidad a partir de modelos de aprendizaje automático previamente entrenados o el uso de cámaras especializadas en la obtención de estos, mejorando la calidad de tanto las imágenes RGB como de los mapas de profundidad.

6 REFERENCIAS BIBLIOGRÁFICAS

- Jia, H., Murphey, Y., Shi, J. & Chang, T.-S. (2004). An intelligent real-time vision system for surface defect detection. *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, 3, 239-242 Vol.3. <https://doi.org/10.1109/ICPR.2004.1334512>
- Ng, H.-F. (2006). Automatic thresholding for defect detection. *Pattern Recognition Letters*, 27(14), 1644-1649. <https://doi.org/https://doi.org/10.1016/j.patrec.2006.03.009>
- Mery, D., Chanona-Pérez, J. J., Soto, A., Aguilera, J. M., Cipriano, A., Veléz-Rivera, N., Arzate-Vázquez, I. & Gutiérrez-López, G. F. (2010). Quality classification of corn tortillas using computer vision. *Journal of Food Engineering*, 101(4), 357-364. <https://doi.org/https://doi.org/10.1016/j.jfoodeng.2010.07.018>
- Branco, P., Torgo, L. & Ribeiro, R. P. (2015). A Survey of Predictive Modelling under Imbalanced Distributions. *CoRR*, *abs/1505.01658*. <http://arxiv.org/abs/1505.01658>
- He, K., Zhang, X., Ren, S. & Sun, J. (2016). Identity Mappings in Deep Residual Networks. <https://doi.org/10.48550/ARXIV.1603.05027>
- Tao, X., Zhang, D., Ma, W., Liu, X. & Xu, D. (2018). Automatic Metallic Surface Defect Detection and Recognition with Convolutional Neural Networks. *Applied Sciences*, 8(9). <https://doi.org/10.3390/app8091575>
- Dunderdale, C., Brettigny, W., Clohessy, C. & Van Dyk, E. (2019). Photovoltaic defect classification through thermal infrared imaging using a machine learning approach. *Progress in Photovoltaics: Research and Applications*, 28. <https://doi.org/10.1002/pip.3191>
- Abhishek, K. & Hamarneh, G. (2020). Matthews Correlation Coefficient Loss for Deep Convolutional Networks: Application to Skin Lesion Segmentation. <https://doi.org/10.48550/ARXIV.2010.13454>
- Defard, T., Setkov, A., Loesch, A. & Audigier, R. (2020). PaDiM: a Patch Distribution Modeling Framework for Anomaly Detection and Localization.

- Miangoleh, S. M. H., Dille, S., Mai, L., Paris, S. & Aksoy, Y. (2021). Boosting Monocular Depth Estimation Models to High-Resolution via Content-Adaptive Multi-Resolution Merging. *Proc. CVPR*.
- Ren, Z., Fang, F., Yan, N. & Wu, Y. (2021). State of the Art in Defect Detection Based on Machine Vision. *International Journal of Precision Engineering and Manufacturing-Green Technology*. <https://doi.org/10.1007/s40684-021-00343-6>
- Yin, W., Zhang, J., Wang, O., Niklaus, S., Mai, L., Chen, S. & Shen, C. (2021). Learning to Recover 3D Scene Shape from a Single Image. *Proc. IEEE Conf. Comp. Vis. Patt. Recogn. (CVPR)*.
- Akcay, S., Ameln, D., Vaidya, A., Lakshmanan, B., Ahuja, N. & Genc, U. (2022). Anomalib: A Deep Learning Library for Anomaly Detection. *arXiv preprint arXiv:2202.08341*.



UNIVERSIDAD AUTÓNOMA DEL
ESTADO DE MORELOS



INSTITUTO DE INVESTIGACIÓN EN CIENCIAS BÁSICAS Y APLICADAS

Control Escolar de Licenciatura



VOTOS DE APROBATORIOS

Secretaria ejecutiva del Instituto de Investigación en Ciencias Básicas Aplicadas de la Universidad Autónoma del Estado de Morelos.
P r e s e n t e .

Por medio de la presente le informamos que después de revisar la versión escrita de la tesis que realizó la C. **REYNOSO GOMEZ LUIS ALFREDO** con número de matrícula **10018748** cuyo título es:

”Detección automática de objetos con errores de manufactura usando imágenes RGB-D”.

Consideramos que **SI** reúne los méritos que son necesarios para continuar los trámites para obtener el título de **LICENCIADO EN CIENCIAS CON AREA TERMINAL CIENCIAS COMPUTACIONALES Y COMPUTACION CIENTIFICA.**

Cuernavaca, Mor a 24 de abril del 2023

Atentamente
Por una universidad culta

Se adiciona página con la e-firma UAEM de los siguientes:

DR. JUAN MANUEL RENDÓN MANCHA	(Presidente)
DRA. LORENA DÍAZ GONZÁLEZ	(Secretario)
DR. JORGE ALBERTO FUENTES PACHECO	(Vocal)
DR. MAURICIO ROSALES RIVERA	(Suplente)
DRA. MARÍA ELISA CHINOS OLIVAN	(Suplente)



UNIVERSIDAD AUTÓNOMA DEL
ESTADO DE MORELOS

Se expide el presente documento firmado electrónicamente de conformidad con el ACUERDO GENERAL PARA LA CONTINUIDAD DEL FUNCIONAMIENTO DE LA UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MORELOS DURANTE LA EMERGENCIA SANITARIA PROVOCADA POR EL VIRUS SARS-COV2 (COVID-19) emitido el 27 de abril del 2020.

El presente documento cuenta con la firma electrónica UAEM del funcionario universitario competente, amparada por un certificado vigente a la fecha de su elaboración y es válido de conformidad con los LINEAMIENTOS EN MATERIA DE FIRMA ELECTRÓNICA PARA LA UNIVERSIDAD AUTÓNOMA DE ESTADO DE MORELOS emitidos el 13 de noviembre del 2019 mediante circular No. 32.

Sello electrónico

MAURICIO ROSALES RIVERA | Fecha:2023-04-24 16:54:44 | Firmante

EP4lUPsYteAfMgnakYfu9QrzzClk+ElZkykaPotX/5EYAUd9Uf/UZxdv8trixlb6w4Gp+/6885WKewnkKpxHZsaa72nNznGy8pXfzOqoBUIDIVxzhDDVkw3YChkd0Hk0fKt21Hkoc40mX0K+hhl05O3BJ9dF1YCYjtU7MfkvEkvag3Kp3Y/se05/pH1eJ0jyb1gJnta9buzav+YFSJpZVnKR6IZ5B7tzcN0t6DR6pPkpxFsv0jGikSYtQuc3X4pXU3/1Q8lPfoFZtkH/Mlxmc/tM+7E5pvT4MpgPMuhin/QCYVs4wi1Mq6C2H/te9xvVPLtUIHEiPR+bj/PkVA==

JORGE ALBERTO FUENTES PACHECO | Fecha:2023-04-24 17:24:36 | Firmante

UnFXykYt/OQqiWQ2EF+8LrPwYzgQ7mFmMNZf68oFSDzhW8/T4LXx5KTA2xnLoPbunY9eNCzoc7pZ1ivz+yocTh4ozGXqXY4tnyGD043pzVAPcqMdmJIQSEXeHQPmF+99Srij9T6keJgDXYkBeDZFayUSSBy7ccekpCQK7DVUN1nwQrrHQAKrM4couZOVrPpA3vi0YT8uq8jndqJwNpQqBJ3zNGy2pJDE1GqPIZDC1muPbOUGqfSICgrhIZy73nXBeFDegVAmeL5ubw8clQdk4yz/y3GbTgJmPqGK4X0u5oCN1bL8LQNXRNmg1jn0fWBQmYfgl7kWePbzqi2de+QBksw==

JUAN MANUEL RENDON MANCHA | Fecha:2023-04-24 18:40:19 | Firmante

xw86xJJdEFVcKzScXs6wSOlwT3Cu6xNR3kHqxlX6vuoZzRZsrgkXthl7XWmh7M4orqOsTcV1IZLO/zpFSsoE+VlaUPDx2aVK92ZjVWu8gOfx04bM/S3Fw7pMspLgJSqLxqRvm2uQKp9u0fhY0kOBqs0Zf2W3l3azc0ipxdufQNWPAIA69cfxguL6+rFvqRNuFOn/iFtQtuf+LvSwsTdDoPjYqeJngxAWYMX5uxil6W2u/GO3DNI/8WBNrOBLNSC5pMt+VKCsvX6IST0QOUt3LJk61HFyC81cqEvYvbVDbi+8/y/vg0JQS8Xw2ZghKRXgeHMixJinopsZb/E9DolZA==

LORENA DIAZ GONZALEZ | Fecha:2023-04-25 08:52:10 | Firmante

CK8qnSDft66i7hCPqlzm2HTCHNCUXWs4iKteC2Jv9+CnUQ+ql8w++UJCRJDh+otxpPDJblRtYVoAJLd35h8tp9/Xo5v6ANmRNgh1rliVclf+lafRcNluTZmXxq1KQBM/Xisvx+sGY7VpDWZvr/TWWVOLTpL0sSUVcx5MN5IBhCogXOllp61WHH4O1S1Qln5sFgRXLTr3hMEHDG2N/+ePu0PEwIRn/i2lchqdNa7/wWi44FRnQWHqeSdBuiPJAT4VFBB6ZhcCmdN3JypnHnHjr/CTYX9WGGJEsr4Vsg1oSRLaxh42FzfxWVfhhqrYd6kNhxFoqynd9N/TXcQpGw5A==

MARIA ELISA CHINOS OLIVAN | Fecha:2023-04-25 21:11:09 | Firmante

W1TajngDrYleLd5c6jZn6ppqKD7kfveDdlBOKBZraoU00OKMUGzt3aPDG0p1SSjEqfWl4ZXtl3s6HoZtDIUQFwu1JyPyI0ILSKFL3vTgKIOSopjm4RrSiUshRIUp2yV9zoGvPT6OVTtRbbB0/xy4G+HFN1mAwk4AloC0YdCjbZW7Un/axCugKtSbc9AcTZ81BlzZ2lGaTVLStp/ChfcBLerVi5le1BX5GysU2jW5LmoJca3wwJesqZWXz+nk/zz8Qvs4iTXyX7dF33zGKs+3CXKcablrrQEQL7ocku3TB01qy+JS9nH8m8h8vt9xM3p/MfQ+ZpxJX9bH5Okz9dw==

Puede verificar la autenticidad del documento en la siguiente dirección electrónica o escaneando el código QR ingresando la siguiente clave:



JzfdYnmGA

<https://efirma.uaem.mx/noRepudio/2X7DDsd1HCYPZsFDogWYQypluZg5DTf>

