



UNIVERSIDAD AUTÓNOMA DE QUERÉTARO  
MAESTRIA EN CIENCIAS EN INTELIGENCIA ARTIFICIAL

**REDUCCIÓN DE VALORES ATÍPICOS Y PRESENCIA DE  
RUIDO EN NUBE DE PUNTOS EN 3D UTILIZANDO  
TÉCNICAS DE APRENDIZAJE PROFUNDO.**

**Tesis**

*que como parte de los requisitos para obtener el grado de maestro en ciencias  
en inteligencia artificial.*

**Presenta:**

Ing. Israel Sotelo Rodriguez

**Dirigida por:**

M.C. Luis Rogelio Román Rivera

Mayo 2022

Universidad Autónoma de Querétaro, Querétaro, Querétaro, México.



UNIVERSIDAD AUTÓNOMA DE QUERÉTARO  
MAESTRIA EN CIENCIAS EN INTELIGENCIA ARTIFICIAL

**REDUCCIÓN DE VALORES ATÍPICOS Y PRESENCIA DE  
RUIDO EN NUBE DE PUNTOS EN 3D UTILIZANDO  
TÉCNICAS DE APRENDIZAJE PROFUNDO.**

**Tesis**

*que como parte de los requisitos para obtener el grado de maestro en ciencias  
en inteligencia artificial.*

**Presenta:**

Ing. Israel Sotelo Rodriguez

**Dirigida por:**

M.C. Luis Rogelio Román Rivera

**Sinodo:**

Presidente: M.C. Luis Rogelio Román Rivera

Secretario: Dr. Jesús Carlos Pedraza Ortega

Vocal: Dr. Juan Manuel Ramos Arreguín

Suplente: Dr. Efrén Gorrostieta Hurtado

Suplente: Dr. Marco Antonio Aceves Fernández

Mayo 2022

Universidad Autónoma de Querétaro, Querétaro, Querétaro, México.

© 2022, Sotelo Rodríguez Israel

Se autoriza la reproducción total o parcial, con fines académicos, por cualquier medio o procedimiento, incluyendo la cita bibliográfica del documento.

A mis padres, el esfuerzo no fue mío, fue de ustedes.

## AGRADECIMIENTOS

Quiero comenzar esta sección expresando mi agradecimiento al director de esta tesis de maestría, M.C Luis Rogelio Román Rivera, por la dirección, el apoyo, el rigor y la dedicación que ha brindado para la elaboración de este proyecto, facilitando herramientas y ayudando en los momentos que mas lo requería, gracias por la confianza brindada desde el día que comenzamos a trabajar juntos.

Así mismo, agradezco al Dr. Carlos Pedraza Ortega quién de igual forma me apoyo de forma incondicional en el proyecto, quién fue también mi guía y tutor, quién es la voz de la experiencia en temas amenos y con quien he aprendido que un trabajo de investigación requiere esfuerzo pero también es bien recompensado. Gracias por su amabilidad para guiarme en esta tesis, por su tiempo, orientación, atención y por sus consejos.

Por su orientación y atención a mis consultas a todo mi sínodo, el Dr. Juan Manuel Ramos Arreguín, el Dr. Efrén Gorrostieta Hurtado y al Dr. Marco Antonio Aceves Fernández. Gracias a todos por el tiempo y esfuerzo que dedicaron a este trabajo de investigación y como profesores de las asignaturas que contribuyeron a mi formación académica.

Gracias a mi familia, a mis padres y hermanos, porque con ellos he compartido los momentos de mi vida que me hacen quien soy ahora. Por las palabras de aliento y por el apoyo incondicional.

Gracias a mis amigos, que siempre han sido un pilar de apoyo durante este largo camino, con quienes he compartido el cansancio, trabajo y recompensa de concluir este trabajo.

Gracias a mi compañera, por su paciencia, apoyo incondicional y solidaridad durante todo este proyecto. Es un gusto compartir este camino contigo.

A todos, muchas gracias.

## Resumen

Para poder realizar la reconstrucción de objetos en 3D, puede recurrirse a un tipo de estructura geométrica llamada “Nube de puntos”. Una nube de puntos es un conjunto de coordenadas que representan la forma o superficie de un objeto. Uno de los principales problemas al hacer el mapeo 3D de escenas u objetos es la presencia de ruido y valores atípicos en una nube de puntos, producido por distintos factores como la luz solar o artificial, la reflectividad de los objetos, la geometría o limitantes relacionadas con el sensor (Sotoodeh, 2006).

**Keywords** – Palabras clave: Nube de puntos, inferencia, matriz de confusión, caracterización, valores típicos.

## Abstract

To reconstruct 3D objects, a type of geometric structure called “point cloud” can be used. A point cloud is a set of coordinates that represent the shape or surface of an object. One of the main problems when doing 3D mapping of scenes or objects is the presence of noise and outliers in the point cloud, produced by different factors such as sunlight or artificial light, object reflectivity, geometry, or sensor-related constraints (Sotoodeh, 2006).

**Keywords** – Point cloud, inference, confusion matrix, characterization, outliers.

# Índice general

|  |           |
|--|-----------|
| <b>AGRADECIMIENTOS</b>   | <b>I</b>  |
| <b>Resumen</b>   | <b>II</b> |
| <b>Abstract</b>  | <b>II</b> |
| <b>1. Introducción</b>   | <b>1</b>  |
| 1.1. Reconstrucción de objetos en 3D . . . . .                       | 1         |
| 1.2. Planteamiento del Problema . . . . .                            | 2         |
| 1.3. Justificación . . . . .   | 3         |
| 1.4. Hipótesis . . . . .   | 5         |
| 1.5. Objetivos . . . . .   | 5         |
| 1.5.1. Objetivos generales . . . . .                                 | 5         |
| 1.5.2. Objetivos específicos . . . . .                               | 5         |
| <b>2. Antecedentes</b>   | <b>6</b>  |
| 2.1. Objetos 3D y sus modelos . . . . .                              | 16        |
| 2.2. Dispositivos para el procesamiento de la profundidad . . . . .  | 16        |
| 2.3. Dispositivos y el preprocesamiento de nubes de puntos . . . . . | 17        |
| 2.4. Nubes de puntos, sensores, escenas y objetos . . . . .          | 17        |
| <b>3. Contenido teórico</b>  | <b>18</b> |
| 3.1. Inteligencia Artificial . . . . .                               | 18        |
| 3.2. Aprendizaje profundo . . . . .                                  | 19        |
| 3.3. Modelo de una neurona . . . . .                                 | 20        |
| 3.4. Capas convolucionales . . . . .                                 | 21        |
| 3.5. Función de agrupación máxima . . . . .                          | 23        |
| 3.6. Sobre-ajuste y sub-ajuste . . . . .                             | 24        |
| 3.7. Aprendizaje por atajos . . . . .                                | 26        |
| 3.8. Aumento de datos . . . . .                                      | 27        |
| <b>4. Metodología y Materiales utilizados</b>                        | <b>30</b> |
| 4.1. Materiales utilizados . . . . .                                 | 34        |
| 4.2. Arquitectura de red . . . . .                                   | 36        |
| 4.3. Capas del modelo propuesto . . . . .                            | 39        |

---

|   |           |
|---|-----------|
| 4.4. Base de datos . . . . .                                    | 42        |
| 4.5. Entrenamiento de la red . . . . .                          | 44        |
| 4.5.1. Parámetros del entrenamiento . . . . .                   | 44        |
| 4.5.2. Hiper-parámetros del modelo . . . . .                    | 45        |
| 4.6. Criterios de evaluación . . . . .                          | 46        |
| <b>5. Resultados</b>  | <b>48</b> |
| 5.1. Resultados Cuantitativos . . . . .                         | 48        |
| 5.2. Resultados Cualitativos . . . . .                          | 50        |
| <b>6. Conclusión</b>  | <b>59</b> |
| 6.1. Trabajo futuro . . . . .                                   | 61        |
| <b>Referencias</b>  | <b>62</b> |
| <b>Apéndices</b>  | <b>66</b> |
| <b>A. Anexos</b>  | <b>66</b> |
| A.0.1. Artículo presentado y aceptado en COMIA 2021 . . . . .   | 66        |
| A.0.2. Artículo aceptado y presentado en CONIIN 2021 . . . . .  | 80        |
| A.0.3. Convalidación examen inglés comprensión textos . . . . . | 81        |
| A.0.4. Convalidación examen inglés manejo de lengua . . . . .   | 82        |
| A.0.5. Protocolo aprobado por consejo. . . . .                  | 83        |

# Índice de tablas

|   |    |
|---|----|
| 2.1. Comparación de arquitecturas del estado del arte para tareas de<br>reducción de ruido, clasificación y segmentacion. . . . . | 15 |
| 5.1. Comparación de resultados cualitativos de los resultados de<br>ejecución en GPU . . . . .                                    | 49 |

# Índice de figuras

|   |    |
|---|----|
| 1.1. Nubes de puntos con presencia de ruido, las nubes fueron tomadas de la base de datos de PointCleanNet (Rakotosaona et al., 2020).  | 3  |
| 1.2. Izquierda nube de puntos con presencia de valores atípicos, derecha Nube de puntos sin valores atípicos. . . . .   | 4  |
| 2.1. Reconstrucción de un objeto en 3D utilizando mapeo y seguimiento de superficies densas en tiempo real (Newcombe et al., 2011). . . .   | 8  |
| 2.2. Reconstrucción del conejo de Stanford realizada con un scanner Cyberware 3030 MS, nube de puntos 3D tomados de (Curless and Levoy, 1996). . . . .  | 9  |
| 2.3. Captura de pantalla del algoritmo ICP en funcionamiento con la reconstrucción de Miguel Ángel (Levoy et al., 2000). . . . .  | 10 |
| 2.4. Inferencia con modelo PCPNet, nubes de puntos 3D de la base de datos de (Guerrero et al., 2018). . . . .   | 11 |
| 2.5. Resultados cualitativos de la segmentación semántica (Qi et al., 2017). . . . .  | 12 |
| 3.1. Diagrama de los subcampos de la inteligencia artificial, basada en (Chollet, 2021). . . . .  | 19 |
| 3.2. Diagrama de una red neuronal simple con función de activación basada en (Haykin et al., 2009). . . . .   | 20 |
| 3.3. Las imágenes pueden dividirse en patrones locales que pueden representar bordes, colores texturas (Chollet, 2021). . . . .   | 23 |
| 3.4. Representación de extracción de los mapas de características con distintas capas convolutivas, basada en (Chollet, 2021). . . . .  | 24 |
| 3.5. Este es un ejemplo de sobre-ajuste: un modelo que funciona mejor en los datos de entrenamiento no es necesariamente un modelo que lo hará mejor en los datos que nunca ha visto antes. . . . . | 26 |
| 3.6. Propuesta de la representación de aprendizaje por atajos en una red neuronal. . . . .  | 28 |
| 3.7. Ejemplo de aumento de datos lineal de un conjunto de datos. Nube de puntos 3D tomados de (Rakotosaona et al., 2020). . . . .   | 29 |
| 4.1. Metodología que se llevara a cabo para la implementación, entrenamiento y validación del algoritmo. . . . .  | 31 |
| 4.2. Jerarquía de carpetas para el entrenamiento e inferencia. . . . .  | 32 |

|  |    |
|--|----|
| 4.3. Formato .xyz utilizado para las nubes de puntos. . . . .  | 32 |
| 4.4. Metodología propuesta para la caracterización del ruido en distintas magnitudes. . . . .  | 33 |
| 4.5. Etapa de reducción de valores atípicos y etapa de reducción de ruido, basada en (Rakotosaona et al., 2020). . . . .   | 36 |
| 4.6. En PCPNet la red aprende un conjunto de funciones puntuales $h$ en el espacio local de un conjunto de puntos, basada en (Guerrero et al., 2018). . . . .  | 37 |
| 4.7. La arquitectura de PointNet toma $n$ puntos como entrada, aplica transformaciones que son utilizadas mas tarde como entrada a capas que realizan maxpooling, basada en (Qi et al., 2017). . . . .   | 37 |
| 4.8. Arquitectura de red propuesta basada en PointCleanNet. . . . .  | 38 |
| 4.9. Bloque básico Conv1D PointCleanNet. . . . .   | 38 |
| 4.10. Bloque básico Conv1D propuesta. . . . .  | 39 |
| 4.11. Bloque básico Lineal PointCleanNet. . . . .  | 39 |
| 4.12. Bloque básico Lineal propuesta. . . . .  | 40 |
| 4.13. Nube de puntos del conejo de Standford. Nube de puntos 3D tomados de (Rakotosaona et al., 2020). . . . .   | 42 |
| 4.14. Caja de peluche sin ruido (izquierda), con ruido blanco (en medio) y nube después de la inferencia utilizando el modelo de PointCleanet (derecha). Nube de puntos 3D tomados de (Rakotosaona et al., 2020). . . . .  | 43 |
| 4.15. Armadillo con distintas magnitudes de ruido blanco. De izquierda a derecha, nube de puntos sin ruido, nube con ruido Gaussiano $1e^{-2}$ , nube con ruido Gaussiano $2,5e^{-2}$ , Ruido Gaussiano $5e^{-2}$ . Nube de puntos 3D tomados de (Rakotosaona et al., 2020). . . . .   | 43 |
| 5.1. Comparativa de resultados obtenidos para nubes de puntos con presencia de valores atípicos. De izquierda a derecha y de arriba abajo: Primer columna: Nubes con 126,000 puntos y valores atípicos. Segunda columna: Inferencia utilizando el modelo propuesto por PointCleanNet para las nubes de 126, 000 puntos. Tercer Columna: Nubes con 84,000 puntos y valores atípicos., Cuarta columna: Inferencia utilizando el modelo propuesto por PointCleanNet para las nubes de 84, 000 puntos. . . . . | 50 |
| 5.2. Resultados obtenidos con la matriz de confusión, estas inferencias se realizaron con el modelo propuesto por el PointCleanNet. . . . .  | 51 |
| 5.3. Diagrama de caja de las métricas aplicadas a las inferencias con el modelo de PointCleanNet. . . . .  | 52 |
| 5.4. Bloque básico para la reducción de valores atípicos. . . . .  | 52 |
| 5.5. Resultados obtenidos con la matriz de confusión de distintas nubes de puntos, estas inferencias se realizaron con nuestra propuesta de modelo. . . . .  | 53 |
| 5.6. Diagrama de caja de las métricas aplicadas a las inferencias con el modelo propuesto a 300 iteraciones. . . . .   | 54 |

---

|   |    |
|---|----|
| 5.7. Resultados obtenidos con la matriz de confusión de distintas nubes de puntos, estas inferencias se realizaron con nuestra propuesta de modelo, además se incrementó el número de épocas a 600. . . . . | 55 |
| 5.8. Diagrama de caja de las métricas aplicadas a las inferencias con el modelo propuesto y 600 iteraciones. . . . .  | 56 |
| A.1. Convalidación examen inglés comprensión textos . . . . .   | 81 |
| A.2. Convalidación examen inglés manejo de lengua . . . . .   | 82 |
| A.3. Protocolo aprobado por consejo. . . . .  | 83 |

# Capítulo 1

## Introducción

### 1.1. Reconstrucción de objetos en 3D

La reconstrucción de objetos en 3D hoy en día juega un papel crucial en muchas áreas de investigación, tal es el ejemplo de la medicina que a través de la utilización de dispositivos 3D como las tomografías computarizadas, busca una buena segmentación de los objetos para así lograr la reducción de riesgo en los procedimientos quirúrgicos. O en el área automotriz, que con la ayuda de la asistencia avanzada de manejo provee una mejor visión del entorno, haciendo un mapeo para clasificar o identificar objetos estáticos o en movimiento, y de esta forma reducir el número de incidentes producidos involuntariamente por los conductores. El mapeo de superficies irregulares y su reconstrucción en 3D en el área de topografía ha ayudado a disminuir las limitantes de mapeo para la reconstrucción de los objetos (Zollhöfer et al., 2018).

La mayoría de los algoritmos están apostando por técnicas de aprendizaje profundo, las que tienden a ser el siguiente primer paso para garantizar una alta calidad en la reconstrucción de objetos. Ayudando a un mejor procesamiento de profundidad, reducción de ruido y la eliminación de valores atípicos para las nubes de puntos en 3D (Zollhöfer et al., 2018).

El entrenamiento de redes neuronales para la clasificación o segmentación han hecho posible obtener nubes de puntos de objetos tridimensionales con una buena reconstrucción, empleando modelos para la eliminación de ruido y valores atípicos, algunos de los principales algoritmos que se han utilizado como referencia en la eliminación de ruido y valores atípicos en las nubes de puntos son PointNet (Qi et al., 2017) y PCPNet (Guerrero et al., 2018).

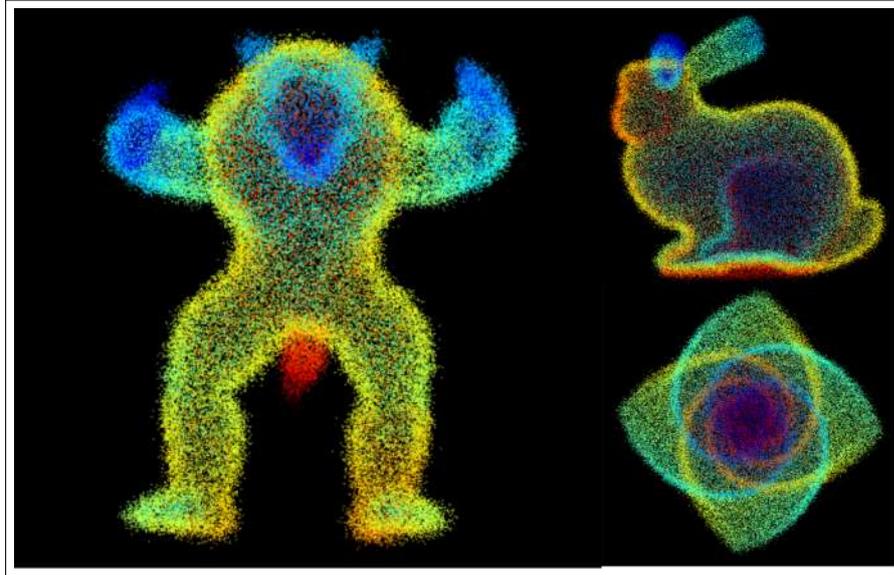
## 1.2. Planteamiento del Problema

La reconstrucción de objetos en tercera dimensión utilizando sensores RGBD se puede llevar a cabo mediante la construcción de nubes de puntos. Uno de los principales problemas a los que nos enfrentamos es la presencia de ruido ocasionado por distintos factores como fuente de luz, interferencia producida por objetos reflectivos y también errores en las mediciones de profundidad en los bordes de los objetos (Barron and Malik, 2013).

Esto puede afectar seriamente la geometría, la profundidad de la reconstrucción del objeto, así como introducir valores atípicos en la nube. Debido al gran reto que representa eliminar valores atípicos y reducir el ruido en nubes de puntos, han surgido varias metodologías para atacar el problema, como métodos de estadística robusta, métodos para mejorar el plano tangente a cada punto y ajustes de parámetros tediosos a los algoritmos (Bookstein, 1989).

La detección de valores atípicos ha sido un gran reto desde varias décadas atrás, para atacar el problema existen también distintos métodos basados en distribución, en profundidad, y en agrupaciones (Sotoodeh, 2006). En la figura 1.1 se muestra tres objetos escaneados, podemos observar cómo el ruido ocasiona pérdida de textura y definición en la nube de puntos.

El ruido en las nubes de puntos puede producir problemas en los métodos de triangulación local, además los modelos que se utilizan dependen más de las características que se presentan en la superficie escaneada y no se ataca propiamente el problema del ruido (Rakotosaona et al., 2020). Los valores atípicos

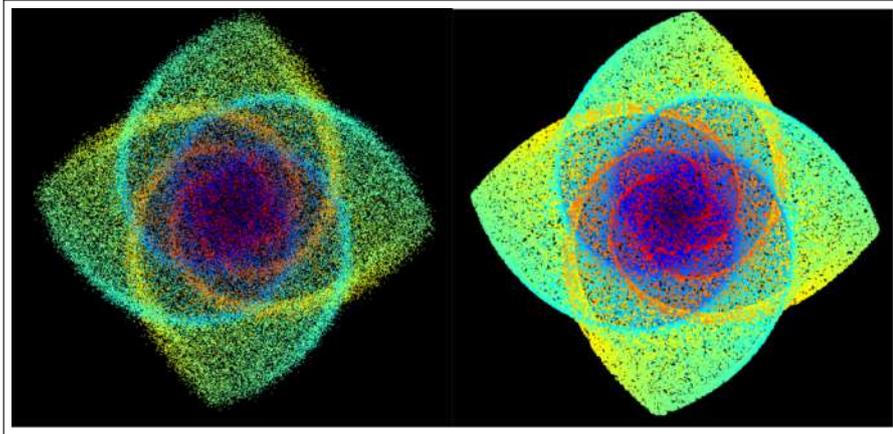


**Figura 1.1:** Nubes de puntos con presencia de ruido, las nubes fueron tomadas de la base de datos de PointCleanNet (Rakotosaona et al., 2020).

en las nubes de puntos son puntos que tienen discontinuidades geométricas, y las formas de sus superficies pueden ser arbitrarias. Los valores atípicos están presentes en las nubes por distintas razones, algunas de ellas son la reflectividad en los materiales, geometría de los objetos u escenas, luz solar, limitantes del sensor, entre otras. Es muy complicado obtener nubes de puntos sin presencia de valores atípicos. En la figura 1.2 observamos como los valores atípicos producen pérdidas de las características del objeto escaneado. Para intentar mitigar este problema nuevas tecnologías con sensores láser están siendo desarrolladas, el problema con estas tecnologías es el alto costo de los sensores (Ning et al., 2018).

### 1.3. Justificación

Una de las principales motivaciones para realizar este trabajo es profundizar en las técnicas utilizadas por dichos algoritmos que encabezan el estado del arte, ya que han demostrado tener gran éxito y popularidad al utilizar técnicas de aprendizaje que atacan el problema con un alto grado de eficacia. Esto con el objetivo de hacer uso de las bases del algoritmo para poder robustecer o mejorar el estado del arte de dichos modelos.



**Figura 1.2:** Izquierda nube de puntos con presencia de valores atípicos, derecha Nube de puntos sin valores atípicos.

Técnicas para la reconstrucción en 3D mediante cámaras RGBD son utilizadas no solo en escenas estáticas sino también en escenas en movimiento. Gracias al fuerte impacto que han tenido estas técnicas, las investigaciones en visión por computadora y la implementación de nuevos modelos de reconstrucción como escaneo en tiempo real y escaneo de integración han tenido un gran avance en los últimos años (Morell et al., 2014).

Hoy en día existen algoritmos para capturar modelos de escenas 3D estáticas y dinámicas con cámaras RGBD, lo que ha llevado a importantes avances en el estado del arte en distintos ámbitos, por ejemplo algunos de los métodos más novedosos consiguen reconstrucciones con muy alta resolución, esto reduciendo al máximo las limitantes del sensor (Dou et al., 2015).

## 1.4. Hipótesis

Es posible desarrollar un algoritmo utilizando inteligencia artificial que aplicado a nubes de puntos de objetos en tercera dimensión, reduzca los valores atípicos y la concentración de ruido en la nube, y de esta forma aumentar la calidad de la reconstrucción de los objetos.

## 1.5. Objetivos

### 1.5.1. Objetivos generales

Diseñar, implementar y evaluar un algoritmo para la reducción de valores atípicos y ruido en una nube de puntos en tercera dimensión, haciendo uso de sus características mediante técnicas de aprendizaje profundo.

### 1.5.2. Objetivos específicos

- Definir, estructurar e implementar las funciones requeridas para el procesamiento y adaptación de la nube de puntos, utilizando un entorno de desarrollo integrado.
- Definir las funciones para que el algoritmo sea invariante a transformaciones que sufra la nube de puntos para mantener consistencia en los resultados.
- Realizar la implementación una red covolucional que haga uso de las propiedades y características de las nubes de puntos.
- Comprobar que la fidelidad de la nube de puntos sea mínimamente afectada cuando se remuevan valores atípicos para evitar distorsiones.
- Definir las pruebas para el algoritmo para medir la capacidad de remover valores atípicos y ruido en la nube de puntos.
- Ejecutar las pruebas y analizar que los resultados obtenidos sean consistentes, utilizando métricas que sean comunes para todos los resultados.
- Realizar comparativa de valores esperados y obtenidos en el algoritmo, así como con algoritmos del estado del arte, mediante un análisis profundo de los resultados.

## Capítulo 2

# Antecedentes

Las cámaras de luz estructurada tuvieron su comienzo a finales de los años 90, una de sus principales aplicaciones fue el control de velocidad, aplicada a vehículos que excedían los límites de velocidad o que no cumplían con los señalamientos en los cruces de las calles (Retting et al., 1999).

Es aquí donde surge una de las principales necesidades de identificar y reconstruir escenarios de alta calidad para la asociación de objetos. Fue en el año 2007 cuando surge una nueva tecnología llamada “Cámaras de luz estructurada”. Dichas cámaras tenían un mejor desempeño midiendo la profundidad en las escenas, así como la intensidad de los píxeles en la imagen (Hagebeuker et al., 2007).

Más tarde comenzaron a desarrollarse sensores que utilizaban técnicas de estimación de distancias mediante el tiempo transcurrido entre emisión y recepción de la onda, esta metodología es llamada “Estimación en tiempo de vuelo”. Sin embargo, todas aquellas que podían hacer una reconstrucción en 3D eran bastante costosas, sin mencionar el gasto computacional que se necesitaba para reconstruir un mapa de profundidad tridimensional (Hagebeuker et al., 2007).

En el año 2010 la empresa Microsoft lanza “Kinect” para su plataforma XBOX 360. “Kinect” es un sensor de tipo RGBD, que ofrece información en color y además de hacer una estimación de la profundidad por píxel, este sensor fue uno de los pioneros en su tipo, y fue rápidamente adoptado debido al bajo costo que tenía (Litomisky, 2012).

Kinect se utiliza para la reconstrucción de objetos y modelado del entorno. Es aquí donde distintas técnicas de reconstrucción como la localización simultanea y mapeo de escenas comienza a tomar fuerza. Esto abrió la puerta a la reconstrucción del entorno para sensores o escenas en movimiento (Morell et al., 2014).

Existen dos enfoques principales en la detección objetos por alcance, la triangulación y el tiempo de vuelo (ToF). La triangulación puede realizarse de distintas formas como enfoque pasivo, es decir, visión estereoscópica, o como sistemas activos, como la luz estructurada.

La visión estereoscópica calcula la diferencia entre dos imágenes tomadas en posiciones diferentes, mientras que las cámaras de luz estructurada proyectan un patrón de luz infrarroja sobre el objeto y estiman la diferencia dada por la distorsión de la perspectiva del patrón debido de la profundidad del objeto. Los escáneres de detección y alcance de luz (LIDAR), miden el tiempo en que la luz emitida por un dispositivo infrarrojo tarda en viajar a un objeto y volver a un detector, en la figura 2.1 donde se realiza una reconstrucción en 3D en tiempo real (Lindner et al., 2007).

Los sensores LIDAR utilizan componentes mecánicos para realizar el barrido, a diferencia de las cámaras ToF las cuales realizan cálculos de tiempo de vuelo en circuitos integrados (Keller et al., 2013).



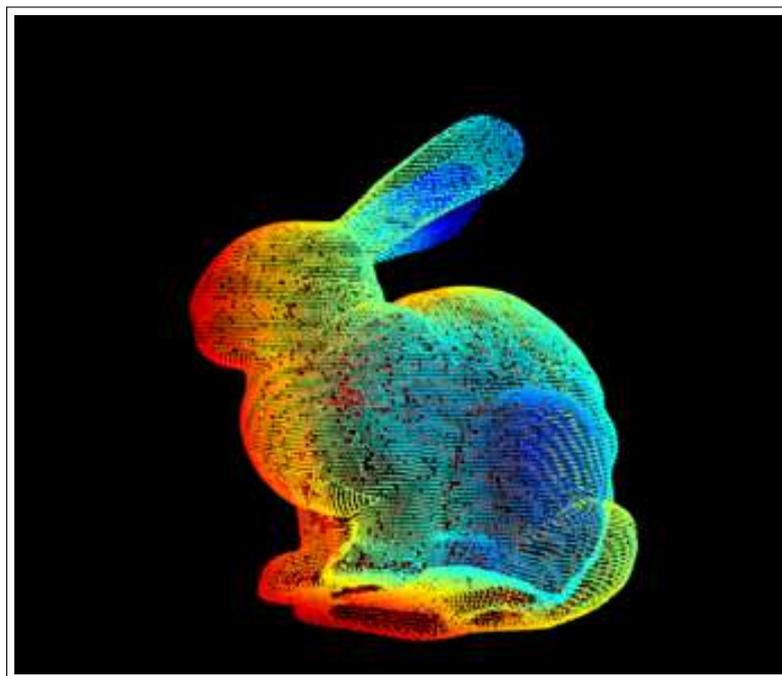
**Figura 2.1:** Reconstrucción de un objeto en 3D utilizando mapeo y seguimiento de superficies densas en tiempo real (Newcombe et al., 2011).

Para poder realizar la reconstrucción de objetos en 3D, puede recurrirse a un tipo de estructura geométrica llamado “Nube de puntos”. Una nube de puntos puede entenderse como un conjunto de características particulares que representan la forma o figura de un objeto. (Morell et al., 2014). Un ejemplo de una nube de puntos se puede observar en la figura 2.2.

Una nube de puntos puede obtenerse mediante un sensor que tenga una Cámara RGB, un proyector o emisor y una cámara infrarroja como receptor. Una excelente opción respecto al costo cómo ya se mencionó anteriormente es el sensor Kinect, ya que en su versión básica es capaz de producir trescientos mil puntos por segundo, con una resolución de 640 x 480 píxeles a 30 FPS (Weber et al., 2015).

Numerosos artículos han utilizado distintas configuraciones para obtener una imagen en 3D mediante un sensor Kinect (Henry et al., 2014). Estas configuraciones son utilizadas para mitigar algunas limitantes, como el hecho de que un sensor Kinect solo permite captar una escena 3D desde una sola perspectiva cuando se utiliza un solo sensor, algunas de las configuraciones más utilizadas son:

- Colocar un objeto sobre una plataforma que permita dar giros de trescientos sesenta grados, y mantener el sensor Kinect estático, con la finalidad de



**Figura 2.2:** Reconstrucción del conejo de Stanford realizada con un scanner Cyberware 3030 MS, nube de puntos 3D tomados de (Curless and Levoy, 1996).

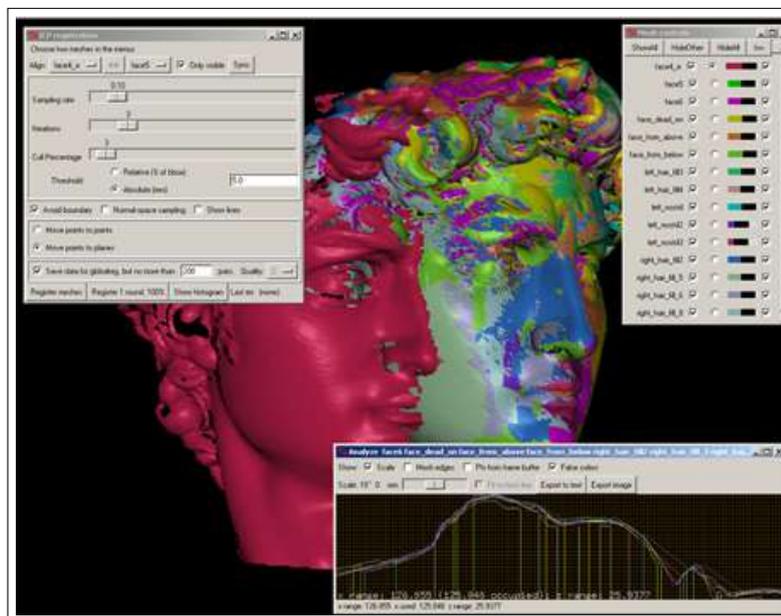
rotar el objeto y tener una reconstrucción desde distintas perspectivas.

- Mantener el objeto estático y montar el sensor Kinect sobre una plataforma que rote entorno al objeto.
- Utilizar distintos sensores posicionados entorno al objeto para realizar la reconstrucción del entorno, los sensores o el objeto pueden estar en rotación con el fin de tener distintas perspectivas.

Algunos sensores utilizados para el mapeo 3D como cámaras RGBD, Estéreo, cámaras monoculares, escáner de largo alcance y láser, requieren de una alineación espacial de las distintas tramas de las nubes de puntos capturados por el sensor.

Existen diversas técnicas para resolver el problema de la alineación espacial, una de las más populares es la Iteración de Puntos Cercanos (ICP, por sus siglas en inglés). El algoritmo ICP itera entre cada punto de la nube en un marco de tiempo, con el objetivo de asociar cada punto captado por el sensor, con el punto más cercano en el modelo de referencia, y así dar lugar al cálculo de transformaciones rígidas para minimizar la distancia entre los mismos tras el transcurso de las iteraciones (Henry et al., 2014).

A grandes rasgos el algoritmo ICP nos ayuda a alinear y rotar una nube de puntos obtenidos por un sensor, con el objetivo de colocarlos en la posición mas cercana a los puntos en un modelo de referencia. Para lograr este objetivo el algoritmo ICP asume que los conjuntos de puntos de la nube obtenidos, y del modelo se encuentran lo suficiente cerca para mantener una correspondencia. En la figura 2.3. podemos observar un ejemplo de la ejecución del algoritmo ICP (Henry et al., 2014).



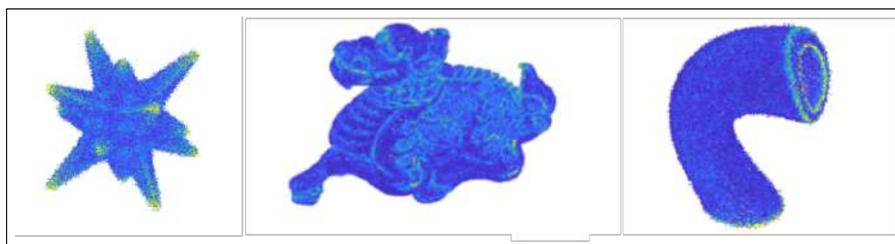
**Figura 2.3:** Captura de pantalla del algoritmo ICP en funcionamiento con la reconstrucción de Miguel Ángel (Levoy et al., 2000).

El algoritmo ICP es iterativo hasta que el error es reducido, el resultado de estas iteraciones es la alineación, la rotación y traslación de las nubes de puntos. Existen variantes del algoritmo ICP que incluyen fórmulas de optimización, métodos para la correspondencia entre la nube de puntos medida y el modelo, así como distintos métodos para remover o descartar valores atípicos o ruido, que comúnmente son provocados por distintos factores durante el escaneo del objeto (Guerrero et al., 2018).

Uno de los principales malestares a los que nos enfrentamos al hacer el mapeo 3D de escenas u objetos, es la presencia de ruido y valores atípicos en las nubes de puntos, es muy común encontrar ruido en las reconstrucciones producido por luz

solar o artificial, la reflexividad de los objetos, geometría o limitantes relacionadas con el sensor (Guerrero et al., 2018).

Para atacar este problema han surgido distintos algoritmos como es el caso del algoritmo PCPNet, el cual mediante la estimación de propiedades locales logra reducir el nivel de ruido en una nube de puntos. En la figura 2.4 podemos observar cómo el algoritmo PCPNet, contiene u error mínimo en la estimación de normales. Para el aprendizaje sugieren un método basado en parches, el enfoque principal es estimar propiedades de forma local para normales (orientadas y desorientadas) y la curvatura de nubes de puntos sin procesar con presencia alta de ruido (Guerrero et al., 2018).



**Figura 2.4:** Inferencia con modelo PCPNet, nubes de puntos 3D de la base de datos de (Guerrero et al., 2018).

PointNet, es otro algoritmo que es importante mencionar, este algoritmo se enfoca principalmente en la clasificación de objetos, segmentación de partes y análisis semántico de la escena. Una de las principales ventajas que tiene PointNet sobre otros algoritmos de clasificación y segmentación es que, al procesar una nube de puntos, dichos algoritmos requieren que las nubes de puntos sean regulares, por lo que estas tienen que ser sometidas a transformaciones. PointNet no necesita realizar transformaciones de este tipo, ya que cada punto se trata de forma única e independiente debido a que utiliza un método llamado “agrupación máxima” (en inglés, Max Pooling) como método de discretización (Qi et al., 2017).

Las nubes de puntos son representadas como un conjunto de puntos en 3D, donde cada vector está representado con las coordenadas  $(x, y, z)$ . Para la tarea de clasificación, las entradas al algoritmo son nubes de puntos muestreadas o pre-segmentadas. En la segmentación semántica, la entrada es un objeto o un

sub-volumen de una escena en 3D (Qi et al., 2017). Podemos observar en la figura 2.5 la segmentación semántica realizada por el algoritmo PointNet.



**Figura 2.5:** Resultados cualitativos de la segmentación semántica (Qi et al., 2017).

La arquitectura propuesta por PointNet, cuenta dos redes neuronales principalmente, una de ellas es utilizada para la clasificación de la nube de puntos y la otra sirve para la clasificación de las propiedades. La red neuronal para la clasificación de la nube utiliza  $n$  puntos de entrada, aplicando transformaciones a las características, finalmente agrega puntos característicos utilizando el método de agrupación máxima. La salida de la red en son los valores de las evaluaciones para cada clasificación (Qi et al., 2017).

---

La red neuronal para la segmentación de la nube de puntos es el complemento de la red neuronal utilizada para la clasificación, esta neurona utiliza como base una red neuronal tipo perceptrón con múltiples capas, la función de activación utilizada es del tipo ReLU. La salida de la red neuronal son nubes de puntos con características globales y locales para cada una de las evaluaciones hecha por la red neuronal encargada de la clasificación (Qi et al., 2017).

Cuando un objeto en tercera dimensión es escaneado para obtener una nube de puntos normalmente es contaminada por distintos tipos de ruido o magnitudes, esto provoca que las nubes de puntos tengan que someterse a un pre-procesamiento para poder realizar tareas de segmentación o clasificación de forma eficiente. Existen diversas metodologías para remover los valores atípicos de una nube de puntos, algunas están basadas en extraer características de puntos vecinos para poder clasificar los valores atípicos y reducirlos. La ecuación 4.1 describe una nube de puntos con presencia de valores atípicos. Dónde  $P_i$ , describe el conjunto de puntos de la superficie del objeto y  $O$  describe el conjunto de valores atípicos presentes (Rakotosaona et al., 2020).

De manera general se establecen las siguientes metodologías utilizadas para la reducción de ruido y valores atípicos en nubes de puntos (Javaheri et al., 2017). La reducción de valores atípicos por radio establece que los puntos que corresponden a los valores atípicos tienen menor densidad de puntos vecinos que los puntos que conforman al objeto (Schoenenberger et al., 2015).

La reducción de valores atípicos por esparcimiento utiliza la distancia entre puntos y el número de vecinos, asumiendo que los valores atípicos tienen una distribución normal y que los puntos que tienen en promedio dos sigmas como desviación estándar deben de ser clasificados como valores atípicos (Rusu et al., 2008).

PointCleanNet es un algoritmo que estima los puntos vecinos, para cada punto de la nube de puntos, esto permite que nubes de puntos densas puedan ser procesadas sin perder características esenciales. Los puntos vecinos se calculan utilizando un radio  $r$ , el radio es calculado en base a la diagonal descrita por los valores máximos y mínimos de los puntos frontera.

Este tipo de algoritmos están pensados para ser ejecutados en GPU. Las nubes de puntos son cada vez más densas debido a la resolución de los sensores utilizados para el escaneo o reconstrucción de superficies u objetos, produciendo que cada vez más puntos tengan que ser cargados en memoria (Javaheri et al., 2017).

Existen distintas propuestas para evitar el consumo excesivo de memoria para el procesamiento de los puntos vecinos de una nube, por ejemplo, el algoritmo “k nearest neighbors” (Sankaranarayanan et al., 2007), que utiliza la localidad de puntos para intentar reducir el tiempo de ejecución. El tiempo de ejecución para el procesamiento de una nube de puntos está en función de la densidad de la nube de puntos y el algoritmo para la estimación de los puntos vecinos (Javaheri et al., 2017).

A través de técnicas de aprendizaje profundo aplicadas a nubes de puntos, basadas en la extracción de características estimadas, es posible reducir valores atípicos y concentración de ruido de varias magnitudes.

| Estado del arte              |                                 |  |              |
|------------------------------|---------------------------------|--|--------------|
| Autor                        | Nombre del modelo               | Categoría  | Arquitectura |
| (Hang Su et al., 2013)       | Multi-View CNN                  | Reconocimiento de objetos en 3D partiendo de distintas perspectivas del objeto utilizando redes neuronales convolucionales.  | CNN          |
| (Katja Wolff et al., 2016)   | Point Denosing                  | Eliminación de ruido de nube de puntos y valores atípicos para la reconstrucción 3D basada en imágenes.  | DPL          |
| (Charles R. Qi et al., 2017) | PointNet                        | Segmentación y clasificación de nubes de puntos en 3D mediante técnicas de aprendizaje profundo.   | RNN          |
| (Vahid Balali et al., 2017)  | 3D recognition and localization | Localización y reconocimiento de señales de tránsito utilizando modelos de nubes de puntos basados en imágenes. Estimación de propiedades en superficies de objetos en 3D. | SfM          |
| (Bo Wu et al., 2018)         | PCPNet                          | Estimación de las propiedades de formas 3D locales en nubes de puntos  | RNN          |

**Tabla 2.1:** Comparación de arquitecturas del estado del arte para tareas de reducción de ruido, clasificación y segmentacion.

## 2.1. Objetos 3D y sus modelos

En visión por computadora un objeto en 3D puede definirse como la representación gráfica de las características que componen a un objeto a través de mapas de rango que pueden representar color, forma y profundidad (Giancola et al., 2018). Existen distintas técnicas y sistemas para la adquisición de los objetos en 3D, por ejemplo, técnicas “de contacto”, de las que derivan destructivas y constructivas, o las técnicas de “no contacto”, de las que derivan las “ópticas activas”, como son el caso de tiempo de vuelo y luz estructurada, siendo estas últimas las más utilizadas (Giancola et al., 2018). Los dispositivos ópticos activos son llamados así debido a que utilizan una fuente de luz activa externa al sensor, la cual ayuda a proporcionar más información de las formas de los objetos.

Un ejemplo de un sensor que utiliza la metodología “Tiempo de vuelo” es el sensor de detección de luces y rango (Lidar, por sus siglas en inglés), quien ha tenido un alto impacto en áreas como la agricultura, en la que se utiliza para proporcionar información del crecimiento y salud de las plantas, o el sector automotriz en el cual es utilizado para la reconstrucción de caminos, objetos estáticos y dinámicos, así como para la clasificación de objetos (Giancola et al., 2018).

## 2.2. Dispositivos para el procesamiento de la profundidad

Las cámaras 3D son muy similares a las cámaras convencionales en el sentido de que captan el entorno que nos rodea, podríamos decir que una cámara es un sensor de tipo óptico pasivo, ya que la fuente de luz es principalmente la luz del entorno que la rodea (Giancola et al., 2018).

Para poder estimar la forma de un objeto en 3D existen varios métodos que nos permiten captar el entorno. Los métodos de triangulación utilizan dos cámaras o dos cámaras y una fuente de luz para poder captar la profundidad de un objeto, algunos de estos modelos están basados en la capacidad de nuestro cerebro para

procesar imágenes captadas de dos puntos distintos.

## 2.3. Dispositivos y el preprocesamiento de nubes de puntos

Los fabricantes de sensores RGBD suelen aplicar filtros a las nubes de puntos, estos filtros están integrados en el software de los sensores, y muchas veces es una caja negra para los usuarios, ya que en la mayoría de los sensores no es posible modificar o ajustar dichos filtros. Es importante tener bajo consideración las limitantes que tienen los distintos sensores del mercado antes de comenzar a trabajar con ellos.

## 2.4. Nubes de puntos, sensores, escenas y objetos

Las principales características del sensor que influyen en la calidad de la reconstrucción de las nubes de puntos pueden verse afectadas por las técnicas de modulación (onda continua o pulsos), las técnicas de medición (tiempo de vuelo, basados en fase o amplitud), detección (coherente o directa) y también de las configuraciones de los transmisores y receptores (mono estática o bi-estática) (Weinmann et al., 2016).

Algunas condiciones atmosféricas como la presencia de agua en el ambiente, la presencia de luz proveniente del sol o alguna fuente artificial, así como el tipo de escena, ya sea dentro de una casa o al aire libre, pueden afectar seriamente la reconstrucción de la escena (Giancola et al., 2018).

La reflectividad de la superficie y la rugosidad del material puede afectar en la luz reflectada al sensor y ocasionar distorsiones en la reconstrucción. La forma del objeto, la orientación y la distancia a la que se encuentra respecto del sensor, pueden influenciar en la reconstrucción de la escena (Giancola et al., 2018).

# Capítulo 3

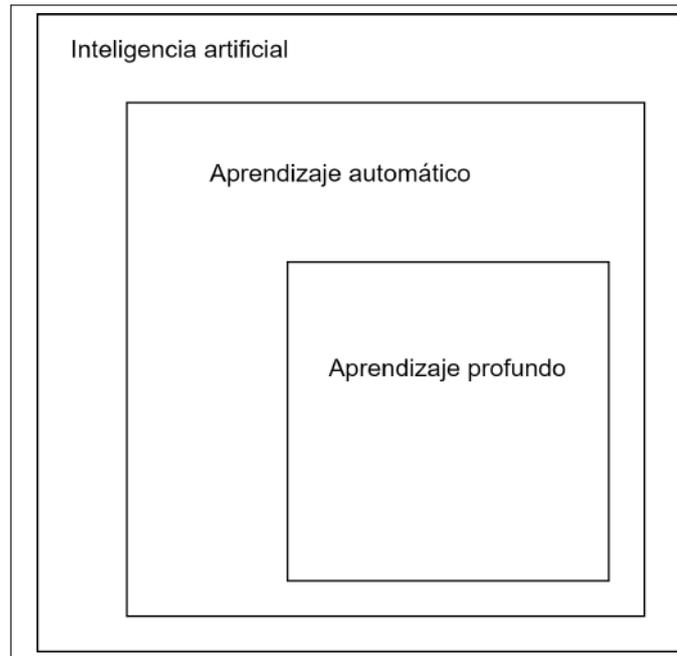
## Contenido teórico

### 3.1. Inteligencia Artificial

La inteligencia artificial tuvo su comienzo cuando algunos pioneros en ciencia computacional comenzaron a cuestionar si una computadora algún día alcanzaría el nivel cognitivo de una persona y si llegase a realizar tareas que una persona realiza sin ningún problema (An et al., 2020). Durante mucho tiempo se creía que una computadora podría alcanzar un nivel cognitivo significativo si se estipulaban ciertas normas o reglas bien definidas y lo suficientemente robustas. Esta metodología es conocida como inteligencia artificial simbólica (en inglés, Symbolic AI) la cual fue popular desde los años 50 hasta los años 80 y de la que se resalta su eficiencia al resolver problemas lógicos. El problema con esta metodología es que es ineficiente cuando se trata de problemas de lógica difusa, clasificación en imágenes o reconocimiento de voz. En la figura 3.1 se muestran algunos de los subcampos que componen la inteligencia artificial. Esto llevó a los investigadores a la búsqueda de una nueva metodología en la que fuera posible atacar estos problemas de forma eficiente. Es aquí cuando nace el aprendizaje automático (en inglés, Machine Learning) (Chollet, 2021).

#### Aprendizaje automático

Este nuevo paradigma de la programación nace de las preguntas: ¿Puede una computadora superar lo que conocemos sobre ordenamiento?, ¿Puede una computadora aprender cómo realizar tareas específicas por su cuenta?, ¿Puede aprenderlo sin necesidad de que un programador especifique cada tarea que



**Figura 3.1:** Diagrama de los subcampos de la inteligencia artificial, basada en (Chollet, 2021).

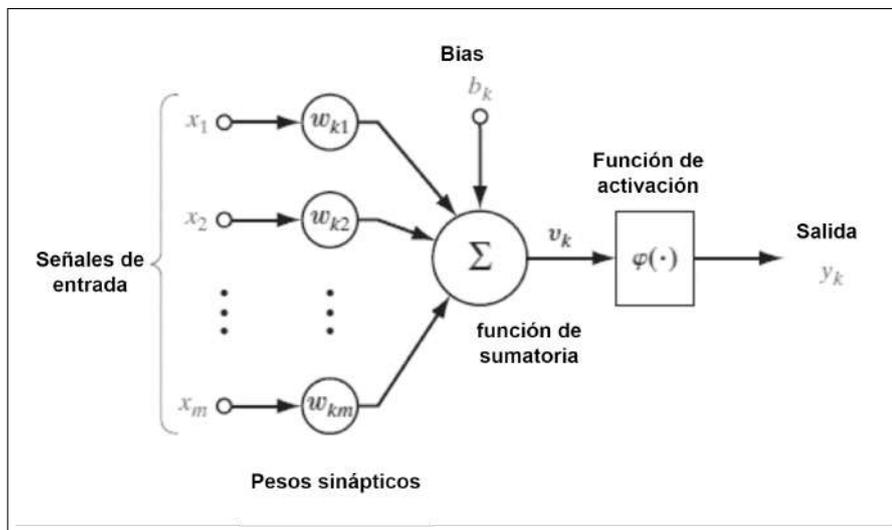
debe de realizar? Podemos tratar de contestar estas preguntas diciendo que un sistema basado en aprendizaje automático debe de ser entrenado en lugar de ser explícitamente programado, es decir, se le dan tareas específicas al sistema y debe de encontrar patrones por su cuenta para así resolverlas. Un ejemplo es la clasificación que hacen los teléfonos celulares para diferenciar los rostros de personas o animales y de esta forma podamos buscar todas las fotografías en las que aparecen de forma eficiente (Chollet, 2021).

## 3.2. Aprendizaje profundo

El aprendizaje profundo (en inglés, Deep Learning) es un subcampo del aprendizaje automático. Es una metodología en la que el enfoque principal es el aprendizaje en capas sucesivas de cada vez más representaciones significativas. Este término es comúnmente confundido ya que se cree que hace referencia a un entendimiento más profundo, pero la realidad es que más bien hace referencia a llevar al aprendizaje a distintas capas (Chollet, 2021).

### 3.3. Modelo de una neurona

Para tener una mejor noción del funcionamiento de una red neuronal, es necesario entender como está compuesta, una neurona es básicamente una unidad utilizada para el procesamiento de información. El modelo de una neurona tiene tres componentes clave:



**Figura 3.2:** Diagrama de una red neuronal simple con función de activación basada en (Haykin et al., 2009).

1. Un conjunto de conexiones o “sinapsis”, cada conexión debe de tener su propio peso característico, en la figura 3.2 estas conexiones están definidas por los pesos sinápticos (en inglés, synaptic weights),  $w_{k1}, w_{k2} \dots w_{km}$ .
2. Un sumador que combine las entradas a la neurona con los respectivos pesos que fueron asignados a cada conexión. En la figura 3.2 este sumador está definido por la suma ponderada definida por la unión sumativa (en inglés, summing junction).
3. Una función de activación (en inglés, activation function), sirve para limitar la amplitud de la salida de la neurona (en inglés, output), también se utiliza como referencia para ajustar el peso que se le da a cada conexión. En la figura 3.2 esta función de activación está representada por  $\varphi^*$ .

De forma matemática podemos representar la neurona de la figura 3.2 con las ecuaciones 3.1 y 3.2. En la ecuación 3.1  $x_1, x_2 \dots x_j$ , son las señales de entrada a

las neuronas (en inglés, input signals),  $w_{k1}, w_{k2}, \dots, w_{km}$  son los respectivos pesos sinápticos para cada una de las entradas,  $u_k$  es la salida del combinador lineal, para las señales de entrada. En la ecuación 3.2  $b_k$  es el bias, su propósito es aplicar una transformación a la salida  $u_k$ , estos dos parámetros son utilizados por la función de activación  $\phi$ , lo cual nos ayuda a obtener la salida  $y_k$  (Haykin et al., 2009).

$$U_k = \sum_{j=1}^m w_{kj} x_j \quad (3.1)$$

$$Y_k = \phi(U_k + b_k) \quad (3.2)$$

### 3.4. Capas convolucionales

Una de las diferencias fundamentales entre una capa que está densamente conectada y una capa convolutiva, es que la capa densa aprende por lo general patrones globales en el espacio de las características dadas en la entrada de la capa, por ejemplo en una imagen dichas características serían todos los píxeles. Siguiendo este mismo ejemplo una capa convolutiva aprendería patrones locales, estos patrones podrían referirse a pequeñas ventanas conformados por una cantidad delimitada de píxeles, dichas ventanas representan patrones locales (Albawi et al., 2017).

Dicho de otra forma, las imágenes pueden dividirse en ventanas o en patrones locales, los patrones locales pueden representar o contener información de los bordes texturas o color de las imágenes.

Algunas de las ventajas principales que podemos encontrar en redes convolutivas en comparación con las redes densamente conectadas son las siguientes:

Los patrones aprendidos en una capa convolutiva son invariantes a la translación, esto quiere decir que después de aprender un patrón específico que puede representar una textura, color o borde, la red adquiere la capacidad de reconocer

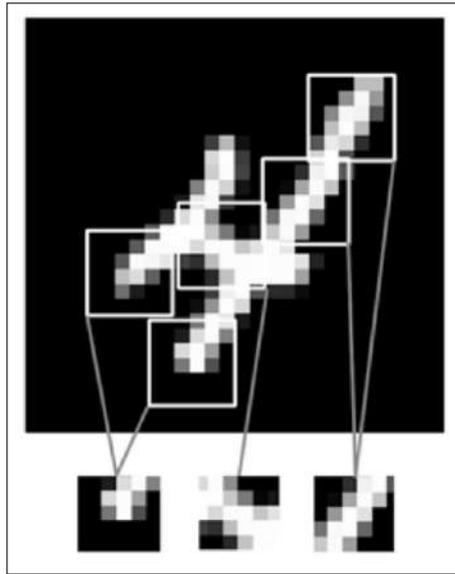
este patrón en cualquier lugar sin importar si la imagen sufre algún tipo de rotación o traslación. En el caso de las redes densamente conectadas, la red tendría que aprender un nuevo patrón por cada nueva transformación de la imagen. Además debido a que estamos aprendiendo patrones locales se necesitan menos muestras para el entrenamiento de la red para aprender presentaciones que pueden llegar a contribuir a la generalización de la tarea a realizar (Kim, 2017).

Las redes convolutivas tienen además la capacidad de aprender jerarquías espaciales de patrones, esto debido que cada una de las capas convolutivas realiza una tarea en específico, por ejemplo, una primera capa de convolución puede aprender pequeños patrones locales, como color, bordes, o texturas. Una segunda capa convolutiva puede aprender patrones más grandes que están conformados por las características de la salida de la primera capa convolutiva y así sucesivamente durante todas las capas que conforman la red. Esto les da la capacidad a las redes convolucionales de aprender patrones complejos y abstractos.

Las capas convolutivas están descritas por tensores en 3D, llamados mapas de características, estos tensores están compuestos por dos ejes espaciales (alto y ancho), así como un eje de profundidad (también llamado eje de canales) (Qu et al., 2018). Para una nube en 3D, la dimensión del eje de profundidad es 3, porque la nube en 3D tiene tres canales de información:  $x$ ,  $y$ ,  $z$ .

La operación de convolución extrae parches o secciones de un mapa de características que es dado a la entrada de la capa, enseguida se aplica una transformación a todos estos parches, produciendo un nuevo mapa de características a la salida, este mapa de características sigue siendo un tensor con la misma dimensionalidad que el que se utilizó como entrada en la capa convolutiva, es decir si hablamos de una reconstrucción de un objeto en tercera dimensión ( $x$ ,  $y$ ,  $z$ ), la salida va a ser un tensor con dimensiones ( $x$ ,  $y$ ,  $z$ ). Un ejemplo de las características que puede tener un patrón local puede observarse en la figura 3.3.

Los valores de los ejes  $x$ ,  $y$ ,  $z$  ya no representan formas específicas de la nube como el objeto en la entrada; más bien, representan filtros. Los filtros codifican aspectos

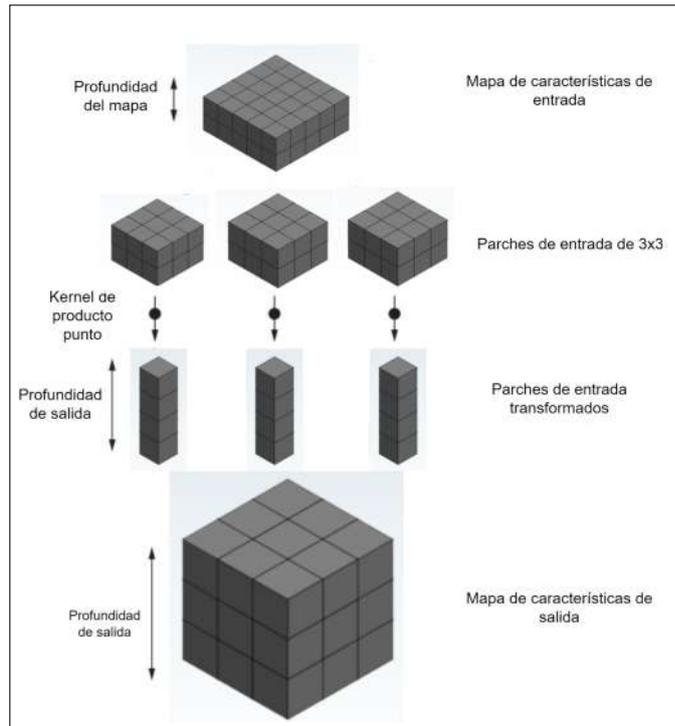


**Figura 3.3:** Las imágenes pueden dividirse en patrones locales que pueden representar bordes, colores texturas (Chollet, 2021).

específicos de los datos de entrada: a un nivel alto, un solo filtro podría codificar el concepto "presencia de un conjunto de puntos con valores atípicos presentes", por ejemplo. Esto es a lo que se refiere el término mapa de características es decir, cada dimensión en el eje de profundidad es una característica (o filtro), y el tensor de salida, es el mapa espacial de la respuesta de este filtro sobre la entrada.

### 3.5. Función de agrupación máxima

Una de las funciones que es indispensable para el procesamiento de los mapas de características generados por las capas convolutivas, es la función de agrupación máxima o MaxPoling por sus siglas en inglés. La principal tarea de esta función es reducir de forma drástica la dimensionalidad de los mapas de características. Esto se logra extrayendo ventanas de los mapas de características de entrada y obteniendo el valor máximo de cada uno de los canales. La principal característica de la agrupación máxima es que transforma los parches locales de los mapas de características a través de una operación de tensor máximo codificada, utilizando ventanas normalmente de  $2n$  con el fin de reducir los mapas de características en un factor de 2 (Chollet, 2021).



**Figura 3.4:** Representación de extracción de los mapas de características con distintas capas convolutivas, basada en (Chollet, 2021).

Una de las principales razones por las cuales se utiliza la función de agrupación máxima, es reducir la dimensionalidad, de esta forma se logra reducir el número de mapas de características que se tienen que procesar, para así introducir jerarquías de filtros espaciales haciendo que las capas consecutivas de convolución generen ventanas cada vez más amplias. Existe otra técnica llamada función de agrupación promedio en este caso los parches locales de los mapas de características se transforman utilizando la operación de tensor promedio codificada. Una de las principales desventajas de utilizar función de agrupación promedio es que puede ocurrir que características significativas se pierdan, ya que se realiza un suavizado dentro de los mapas de características. Una representación de los mapas de características se puede observar en la figura 3.4.

### 3.6. Sobre-ajuste y sub-ajuste

Uno de los principales problemas con los que nos enfrentamos cuando estamos realizando aprendizaje automático es el sobre entrenamiento, éste aparece cuando

el rendimiento del modelo que intentamos entrenar alcanza un máximo después de un número determinado de épocas, para después comenzar a degradarse. Es por ello que es importante entender que el sobre-ajuste ocurre en todos o casi todos los problemas que intentamos resolver con aprendizaje automático.

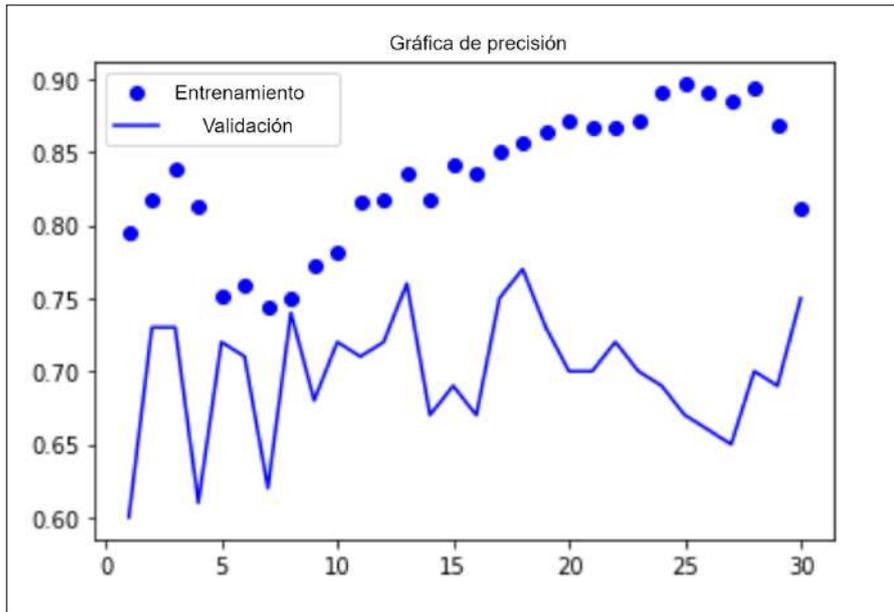
La principal pregunta que nos debemos de hacer cuando estamos utilizando técnicas de aprendizaje automático es la relación que existe entre la optimización y la generalización. La optimización se refiere al proceso de ajuste de un modelo para obtener el mejor rendimiento posible en los datos de entrenamiento, mientras que la generalización se refiere a lo bien que funciona el modelo entrenado en datos que nunca ha visto antes (Chollet, 2021).

El objetivo principal es conseguir una buena generalización, por supuesto, pero es muy complicado controlar la generalización; debido a que sólo se puede ajustar el modelo basándose en los datos de entrenamiento.

Podemos decir que durante el entrenamiento la optimización y la generalización están correlacionadas de forma directa es decir, cuanto menor sea la pérdida en los datos de entrenamiento, menor será la pérdida en los datos de prueba (Rozsa et al., 2016). Mientras esto ocurre, se dice que su modelo está sub-ajustado, por lo que todavía hay que entrenar el modelo, es decir que la red aún no ha modelado todos los patrones relevantes en los datos de entrenamiento. Un ejemplo visual del un sobre-ajuste o sobre entrenamiento puede observarse en la figura 3.5.

Si observamos que después de un cierto número de iteraciones en los datos de entrenamiento, la generalización deja de mejorar, y las métricas de validación se estancan y empiezan a degradarse: es aquí cuando decimos que el modelo empieza a sobre-ajustarse. Es decir, empieza a aprender patrones que son específicos de los datos de entrenamiento pero que son engañosos o irrelevantes cuando se trata de nuevos datos. (Chollet, 2021).

Una forma de prevenir que un modelo aprenda patrones irrelevantes de los datos de entrenamiento es generalizando aún más los datos de entrenamiento, es decir aumentando la base de datos, de esta forma nuestro modelo generalizará mejor.



**Figura 3.5:** Este es un ejemplo de sobre-ajuste: un modelo que funciona mejor en los datos de entrenamiento no es necesariamente un modelo que lo hará mejor en los datos que nunca ha visto antes.

en ocasiones aumentar la base de datos no es factible, las siguientes solución es modular o limitar la información que el modelo puede utilizar como entrenamiento, por ejemplo, se pueden limitar los patrones que la red puede identificar, estas restricciones obligan a la red a centrarse en los patrones más significativos, que tienen mayor probabilidad de generalizar el modelo (Chollet, 2021).

### 3.7. Aprendizaje por atajos

El desempeño de una red convolutiva también esta limitado por distintos factores, como son la base de datos y su distribución, la arquitectura del modelo, el número de épocas y también la función de costo. Todos estos factores contribuyen a un solo objetivo, la generalización para resolver de forma efectiva el problema que estamos intentando atacar.

Es común encontrar redes con problemas de generalización, por ejemplo, imaginemos que intentamos clasificar un gato, pero cuando lo movemos de posición, es decir, lo rotamos la red pierde la capacidad de clasificarlo apropiadamente, es

aquí cuando la red puede que este aprendiendo patrones muy generales como la posición del gato en los datos de entrenamiento y lo esté asociando a la clasificación, este es un ejemplo del aprendizaje por atajos. Ahora bien, si la red fuera capaz de utilizar las características de la silueta podría generalizar aún más este atajo, la red podría ser capaz de clasificar de forma apropiada al gato.

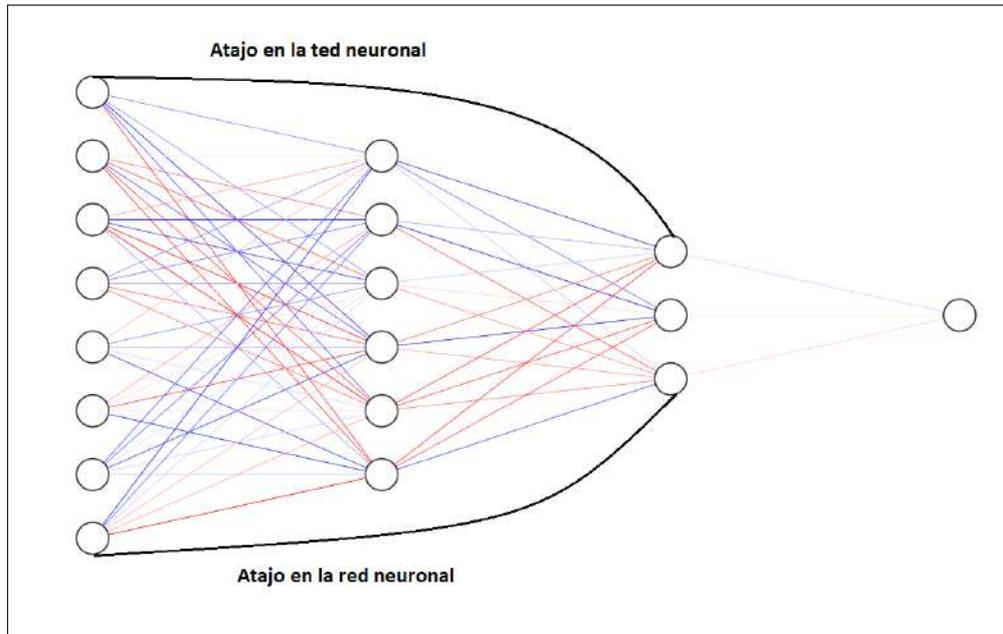
Sin duda alguna el aprendizaje por atajos es una característica que está presente en las redes convolutivas, las redes explotan los atajos del conjunto de datos, y seleccionan características predictivas en lugar de considerar cuidadosamente los mapas de características mas densos, en consecuencia, sufren fallos de generalización (Geirhos et al., 2020).

Así pues el aprendizaje por atajos se puede describir como la acción de la red convolutiva para aprender reglas de decisión que no son robustas a cambios en la distribución de los datos, pero también pueden utilizarse de forma efectiva para lograr generalizar características entre capas, también es importante mencionar que el aumento de datos y la distribución de las bases de datos, puede contribuir a que este tipo de aprendizaje se realice de forma efectiva (Amenábar Montenegro, 2021). Un ejemplo de aprendizaje por refuerzo en capas convolutivas puede observarse en la figura 3.6.

### 3.8. Aumento de datos

Cuando entrenamos nuestros modelos lo que buscamos es una buena generalización, para que consecuentemente la pérdida de nuestro modelo sea mínima, y de esta forma logre realizar la tarea que se intenta resolver como objetivo de forma correcta. Las redes neuronales del estado del arte tienden a ser redes muy densas, con parámetros en el orden de los millones. Esto implica un reto debido a que todos o casi todos estos parámetros van a necesitar entrenamiento, y están ligados a la complejidad de la tarea que se desea resolver.

Una de las principales tareas que debemos de resolver antes de entrenar un modelo es encontrar una base de datos lo suficiente robusto que nos permita realizar un



**Figura 3.6:** Propuesta de la representación de aprendizaje por atajos en una red neuronal.

entrenamiento efectivo. De esta forma si tenemos miles de parámetros va a ser necesario utilizar miles de ejemplos para el entrenamiento. Por lo general solemos estar preocupados por si el tamaño de la base de datos para el entrenamiento es lo suficiente robusto, y cuando no lo logramos buscamos técnicas que nos permitan robustecer la base de datos, a esto se le conoce como aumento de datos.

Para robustecer la base de datos basta con realizar pequeñas alteraciones en la base de datos ya existente, estas alteraciones pueden estar descritas por giros, translaciones, rotaciones, cambio de color, etc. Las redes neuronales convolucionales que tienen la capacidad de clasificar objetos o figuras de forma eficiente aún cuando están colocadas en distintas orientaciones adquieren una propiedad de invarianza, este tipo de redes son llamadas redes invariantes ante translaciones o rotaciones (Shorten and Khoshgoftaar, 2019).

Se dice que una red neuronal convolucional que puede clasificar objetos de forma robusta aunque se coloquen en diferentes orientaciones tiene la propiedad de invarianza. Más concretamente, una CNN puede ser invariante a la traslación, el punto de vista, el tamaño o la iluminación (o una combinación de los anteriores) (Kauderer-Abrams, 2017).

Existen dos técnicas muy conocidas que nos permiten realizar aumento de datos, la primera técnica es conocida como aumento de datos tradicional, se utiliza generalmente en base de datos pequeñas, debido a que el tamaño final de la base de datos estaría en función de las transformaciones que se realicen. Es decir, si realizo una rotación a cada uno de los elementos de la base de datos, la base de datos final aumentara en un factor de dos. (Mikołajczyk and Grochowski, 2018).

La segunda técnica es llamado aumento en línea o aumento sobre la marcha, es comúnmente utilizado en conjunto de datos robustos, donde se realizan transformaciones a subsecciones de la base de datos mientras se entrena el modelo, este tipo de técnicas ayudan a reducir el tiempo de ejecución en GPU (Nguyen et al., 2020).



**Figura 3.7:** Ejemplo de aumento de datos lineal de un conjunto de datos. Nube de puntos 3D tomados de (Rakotosaona et al., 2020).

Se utiliza en base de datos robustas debido a que si se utilizaran técnicas como el aumento de datos sin conexión la base de datos aumentaría aún mas su tamaño y su gasto computacional sería aún mayor. En la figura 3.7 puede observarse un ejemplo del uso de esta metodología.

## Capítulo 4

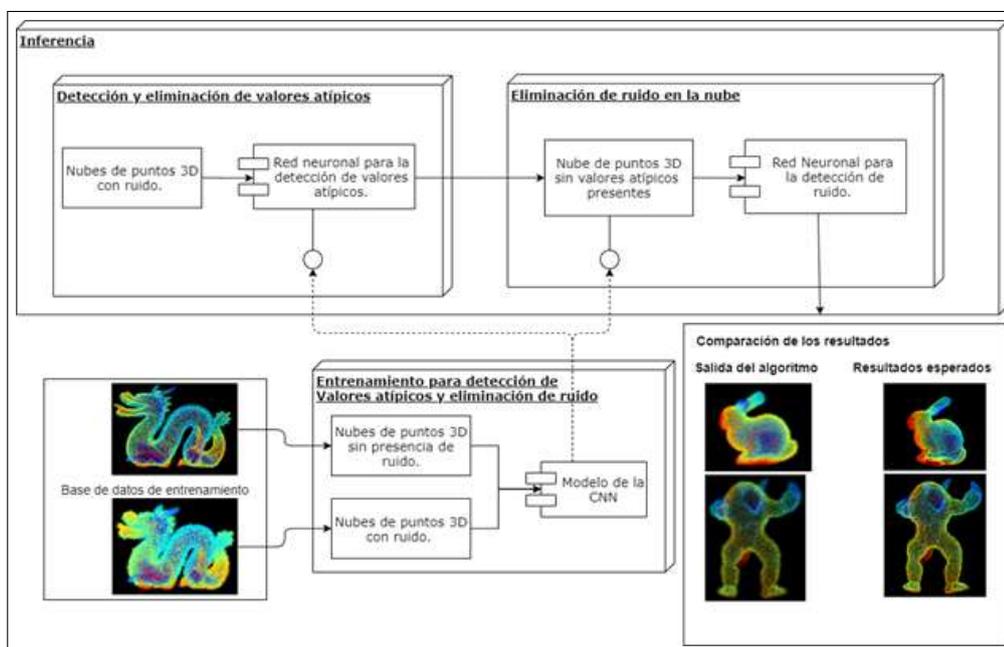
# Metodología y Materiales utilizados

El desarrollar un algoritmo para remover los valores atípicos y el ruido en una nube de puntos conlleva la realización de diversas actividades, algunas de ellas son:

1. Seleccionar una base de datos pública que contenga nubes de puntos comúnmente utilizadas en el estado del arte, para entrenamiento, prueba y validación del algoritmo. Así como la creación de una base de datos con nubes de puntos para robustecer el entrenamiento.
2. Realizar una exhaustiva investigación sobre las arquitecturas de CNN que actualmente se utilizan en el estado del arte, así como nuevas metodologías y modelos para la eliminación de ruido y valores atípicos en nubes de puntos.
3. Definir la división de la base de datos para entrenamiento y prueba de la CNN, debe de considerarse una base de datos robusta, y se debe definir qué porcentaje corresponde al entrenamiento y a la prueba.
4. Identificar dentro de la base de datos nubes de puntos con diferentes características, distintos niveles de ruido y de porcentaje de valores atípicos presentes.
5. Realizar el entrenamiento de la CNN con los elementos de la base de datos que previamente fueron definidos.
6. Llevar a cabo la inferencia para remover los valores atípicos y el ruido dentro de la nube, con las nubes de puntos que se habían definido previamente para

la prueba del modelo.

7. Comprobar e interpretar las nubes de puntos obtenidas después de la inferencia, se debe de llevar un control de los resultados de éxito y falló que se hayan obtenido para ajustar la arquitectura de la CNN.
8. Identificar los resultados obtenidos por algoritmos del estado del arte relacionados con métodos para remover el ruido y valores atípicos en nubes de puntos y compararlos con los resultados obtenidos con nuestro algoritmo propuesto.



**Figura 4.1:** Metodología que se llevara a cabo para la implementación, entrenamiento y validación del algoritmo.

Es importante definir los requisitos de las circunstancias en las que el algoritmo debe de tener un óptimo desempeño. Por ejemplo en la figura 4.1 podemos observar algunas de las etapas a alto nivel como: seleccionar la base de datos, implementar la arquitectura de una CNN, entrenar CNN para que sea capaz de reducir la cantidad de valores atípicos y de ruido, etapa de validación donde se evalúa la implementación de la red comparándola con resultados esperados. El diagrama de la figura 4.1 muestra la metodología a seguir para la implementación, entrenamiento y validación del algoritmo.

En la figura 4.2 se muestra la distribución de las carpetas, donde se podrán encontrar las nubes de puntos utilizadas para el entrenamiento, los modelos entrenados, así como los scripts para remover los valores atípicos y el ruido.

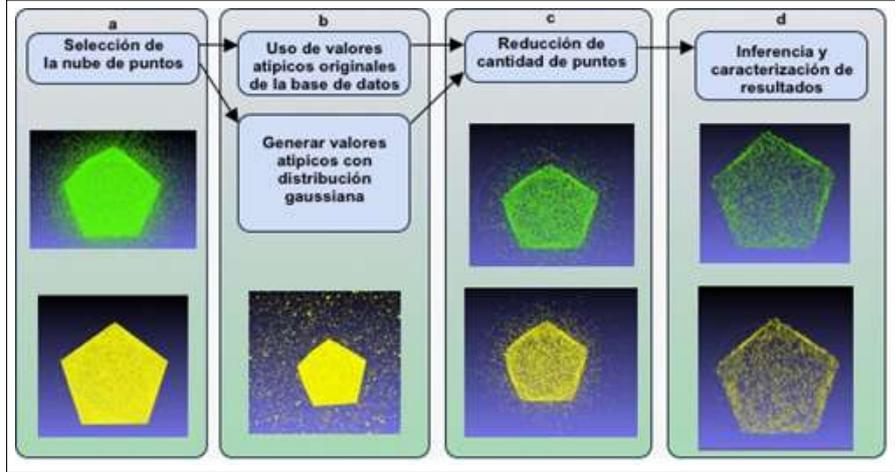
| Nombre           | Fecha                  | Tamaño | Extensión           |
|------------------|------------------------|--------|---------------------|
| data             | 03/05/2020 01:01 p. m. |        | Carpeta de archivos |
| images           | 26/04/2020 09:51 p. m. |        | Carpeta de archivos |
| models           | 01/05/2020 07:41 p. m. |        | Carpeta de archivos |
| noise_removal    | 03/05/2020 01:03 p. m. |        | Carpeta de archivos |
| outliers_removal | 28/04/2020 07:20 a. m. |        | Carpeta de archivos |
|                  | 26/04/2020 09:51 p. m. | 1 KB   | Documento de tex... |
|                  | 26/04/2020 09:51 p. m. | 2 KB   | Archivo             |
|                  | 26/04/2020 09:51 p. m. | 4 KB   | Archivo MD          |

**Figura 4.2:** Jerarquía de carpetas para el entrenamiento e inferencia.

En la figura 4.3 se muestran algunas nubes de puntos con y sin ruido utilizadas para el entrenamiento y validación de la red neuronal.

| Nombre                                       | Fecha                  | Tamaño | Extensión         |
|--|------------------------|--------|-------------------|
| armadillo100k.clean_xyz                      | 25/09/2018 07:06 a. m. |        | Archivo CLEAN_XYZ |
| armadillo100k.xyz                            | 25/09/2018 07:06 a. m. |        | Archivo XYZ       |
| armadillo100k_noise_white_1.00e-02.clean_xyz | 25/09/2018 07:06 a. m. |        | Archivo CLEAN_XYZ |
| armadillo100k_noise_white_1.00e-02.xyz       | 25/09/2018 07:06 a. m. |        | Archivo XYZ       |
| armadillo100k_noise_white_1.50e-02.clean_xyz | 25/09/2018 07:06 a. m. |        | Archivo CLEAN_XYZ |
| armadillo100k_noise_white_1.50e-02.xyz       | 25/09/2018 07:06 a. m. |        | Archivo XYZ       |
| armadillo100k_noise_white_2.50e-02.clean_xyz | 25/09/2018 07:06 a. m. |        | Archivo CLEAN_XYZ |
| armadillo100k_noise_white_2.50e-02.xyz       | 25/09/2018 07:06 a. m. |        | Archivo XYZ       |
| armadillo100k_noise_white_2.50e-03.clean_xyz | 25/09/2018 07:06 a. m. |        | Archivo CLEAN_XYZ |
| armadillo100k_noise_white_2.50e-03.xyz       | 25/09/2018 07:06 a. m. |        | Archivo XYZ       |
| armadillo100k_noise_white_5.00e-03.clean_xyz | 25/09/2018 07:06 a. m. |        | Archivo CLEAN_XYZ |
| armadillo100k_noise_white_5.00e-03.xyz       | 25/09/2018 07:06 a. m. |        | Archivo XYZ       |
| boxunion2100k_noise_white_2.50e-02.clean_xyz | 25/09/2018 07:06 a. m. |        | Archivo CLEAN_XYZ |
| bunny100k.clean_xyz                          | 25/09/2018 07:06 a. m. |        | Archivo CLEAN_XYZ |

**Figura 4.3:** Formato .xyz utilizado para las nubes de puntos.



**Figura 4.4:** Metodología propuesta para la caracterización del ruido en distintas magnitudes.

La caracterización del ruido se realizó utilizando la metodología propuesta en la figura 4.4. Se utilizaron nubes de puntos de la base de datos de PointCleanNet para reducción de valores atípicos (a). De la base de datos se generaron 288 nubes, que corresponden a 4 figuras, de las cuales se obtuvo una nube de puntos con ruido gaussiano con una desviación estándar de 1%, 2.5% y 5% (b). El modelo que describe la nube de puntos y el ruido está descrito por la ecuación 4.1.  $P'$  representa la nube de puntos con ruido,  $P$  son el conjunto de puntos que representan la superficie del objeto,  $n_i$  es el ruido aditivo,  $O$  es el conjunto de valores atípicos presentes en la nube mostrados en la figura 4.4.

$$P' = (P'_i) = (P' + n_i)_{p_i \in P} \cup (O_j)_{O_j \in O} \quad (4.1)$$

Las nubes de puntos están compuestas por los puntos que describen la superficie del objeto, presencia de ruido y valores atípicos. Para el entrenamiento se utiliza una función no lineal que es utilizada para remover los valores atípicos, esta función se muestra en la ecuación 4.2.

$$\tilde{O}_i = g(P'_i) > 0,5 \quad (4.2)$$

La ecuación 4.2 representa la probabilidad de valor atípico  $\tilde{O}_i$ , establece que un

punto es añadido al conjunto de valores atípicos si la probabilidad de valor atípico es mayor al 0.5.

Se generó un archivo en el que se clasifican cada uno de los puntos. Además, la densidad de las nubes varía con intervalos de 10 % en un rango de 10 % a 90 % (c).

Posterior a esto, se realizaron 144 inferencias con el modelo propuesto por PointCleanNet, utilizando como entrada las nubes con distintas magnitudes de ruido y distinta densidad como se muestra en la figura 4.4. El objetivo de la inferencia es obtener el rendimiento de la red neuronal con distintos niveles de ruido y densidad para determinar el punto de inflexión con el que la remoción de puntos no afecte el rendimiento y a la vez contribuya a reducir el tiempo de inferencia.

## 4.1. Materiales utilizados

Debido al alto gasto computacional que implica procesar nubes densas de puntos, fue necesario utilizar una workstation de alto rendimiento para las inferencias, además de una laptop donde se realizaban tareas que no implicaban un alto rendimiento como el aumento de datos, evaluación de resultados, configuración de los parámetros de entrenamiento, etc.

La computadora de bajo rendimiento cuenta con las siguientes características: Laptop Hp Pavilion 4 Gb RAM, 500 Gb HDD, GTX1650, Rizen 7. En esta computadora se utilizó el siguiente SW para el procesamiento y visualización de las nubes.

- MeshLab versión 2020.12
- Pycharm versión 2020.1
- Anaconda Python 3.7 64-Bit
- Jupyter notebook

Los algoritmos para el preprocesamiento de las nubes fueron desarrollados en Python 3.7 utilizando Jupiter Notebook. Las nubes eran cargadas en la memoria de la GPU para el preprocesamiento, MeshLab fue utilizado como herramienta de visualización, una nube de puntos con una densidad de 140, 000 puntos al ser visualizada ocupaban 532 MB de memoria RAM, por lo que las especificaciones de la Laptop HP Pavilion eran suficientes.

La computadora de alto rendimiento fue utilizada para las inferencias hechas con el modelo de PointCleanNet y nuestro modelo propuesto, la computadora cuenta con las siguientes características: Workstation NVIDIA GeForce RTX 3060 TI. 8GB GDDR6. 4864 CUDA Cores, AMD Ryzen 5600x, 6 Cores, 12 hilos, 3.7GHz 32MB L3 Cache 3MB L2 Cache, 16 GB RAM.

En esta computadora se tenía instalado el siguiente SW para poder realizar la configuración del ambiente y la inferencia de las nubes:

- Python 3.7 64-Bits
- Docker container Linux Engine Versión 19.03.8
- Ubuntu 20.04.2 LTS

Se opto por utilizar una workstation debido a que la alta densidad de las nubes de puntos requiere un alto procesamiento en GPU. Se utilizó una tarjeta RTX 3060, con sistema operativo Ubuntu, y contenedores Docker para la manipulación del ambiente de programación, aquí logramos disminuir el tiempo de entrenamiento e inferencias de la red neuronal, esta workstation es accedida de forma remota.

Adicionalmente decidimos utilizar Google colab para el entrenamiento e inferencias de la red, las características que tiene nuestra cuenta de Google colab son las siguientes:

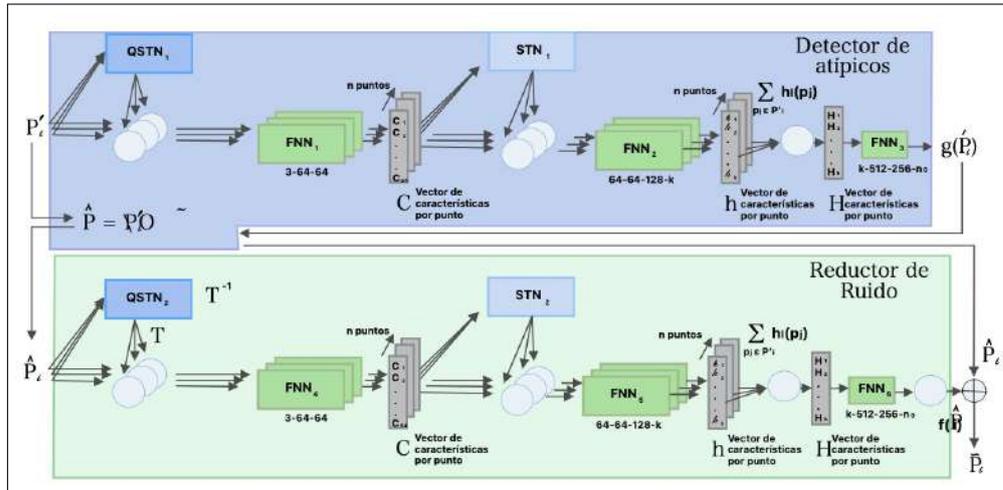
- K80, P100, T4 24 GB GPU
- 2 x vCPU
- Tiempo continuo de ejecución de 24 hrs
- Acceso a CMD

- Almacenamiento en la nube

La ventaja que tiene Google colab es que provee de servicios en la nube, es barato y ofrece una mayor disposición de recursos que otras plataformas como AWS. Es relativamente fácil programar entrenamientos para la red, además de que es menos complicado instalar dependencias para el ambiente de programación, debido a que ya cuenta con ambientes de programación predefinidos.

## 4.2. Arquitectura de red

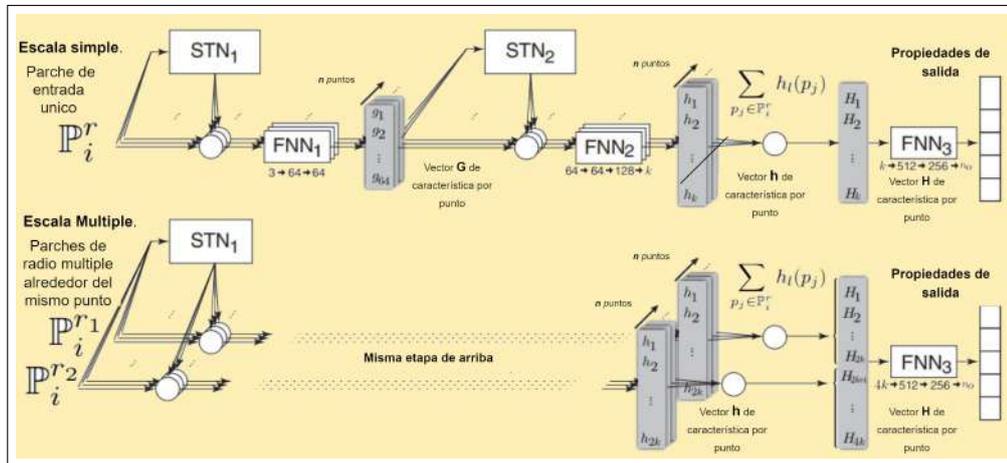
La arquitectura de la red neuronal convolucional esta basada en la red propuesta por PointCleanNet descrito en la figura 4.5, tiene dos etapas principales, una etapa dedicada a reducir el número de valores atípicos y otra a reducción de ruido.



**Figura 4.5:** Etapa de reducción de valores atípicos y etapa de reducción de ruido, basada en (Rakotsoana et al., 2020).

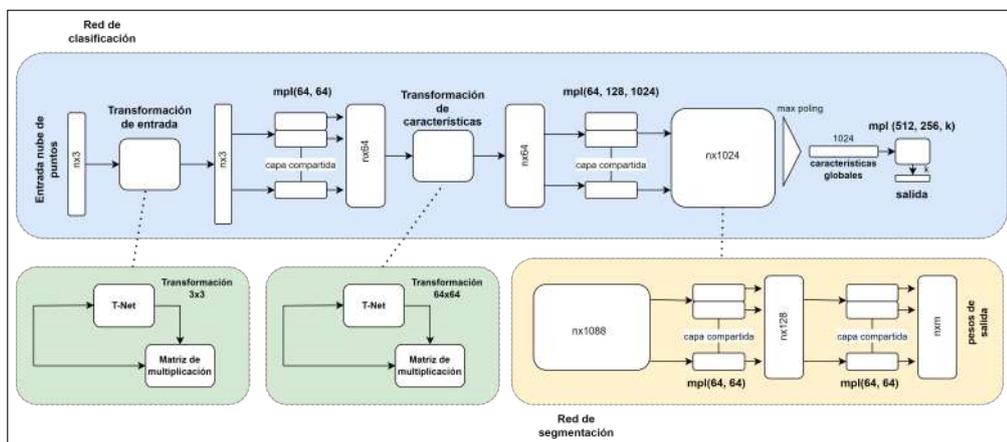
Para la reducción de valores atípicos la red neuronal está basada en las arquitecturas de PCPNet, figura 4.6, donde se utilizan STN, son básicamente redes de transformadores espaciales que aplican una rotación de cuaterniones (STN1) o una transformación lineal completa (STN2) (Guerrero et al., 2018) y PointNet (Qi et al., 2017), esta red aprende un conjunto de criterios de optimización que seleccionan puntos interesantes o informativos de la nube de puntos y utiliza una función de selección, la arquitectura de PointNet se observa

en la figura 4.6.

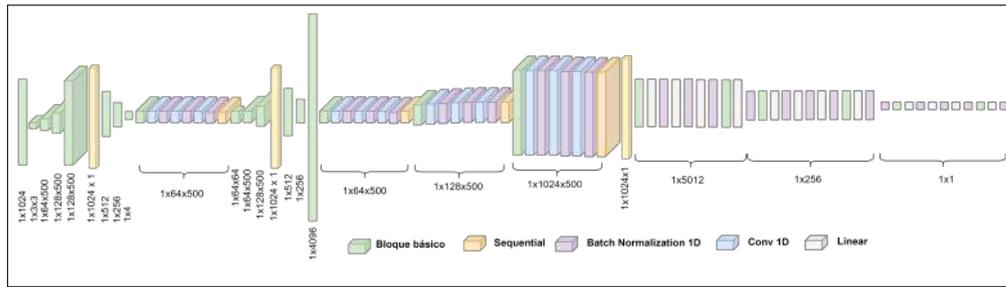


**Figura 4.6:** En PCPNet la red aprende un conjunto de funciones puntuales  $h$  en el espacio local de un conjunto de puntos, basada en (Guerrero et al., 2018).

Las características de PCPNet y PointNet son utilizadas por PointCleanNet haciendo uso de perceptrones de múltiples capas y transformadores espaciales para la extracción de características significativas y utilizando una metodología para saltar conexiones, logra promover la propagación del gradiente y mejorar el entrenamiento (Rakotosaona et al., 2020). Esta arquitectura de red, además utiliza reducción de valores atípicos por radio, esta metodología es utilizada para estimar que tan probable es el punto central de ser un valor atípico.

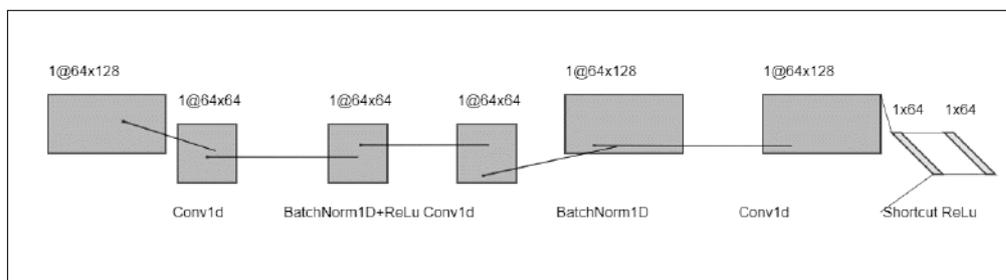


**Figura 4.7:** La arquitectura de PointNet toma  $n$  puntos como entrada, aplica transformaciones que son utilizadas mas tarde como entrada a capas que realizan maxpooling, basada en (Qi et al., 2017).



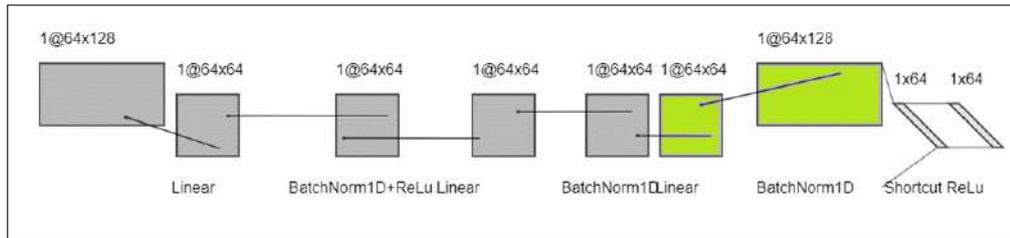
**Figura 4.8:** Arquitectura de red propuesta basada en PointCleanNet.

En la figura 4.9 se puede observar un bloque básico convolucional de una dimensión, este bloque es propuesto por PointCleanNet, también se encuentra en la arquitectura propuesta en la figura 4.8. Podemos observar que comienza con una capa convolutiva la cual está encargada de extraer mapa de características, enseguida se aplica una transformación a todos estos parches, produciendo un nuevo mapa de características, después la salida de capa es reducida utilizando batch normalization, una de las tareas principales de esta capa es reducir la dimensionalidad de los mapas de características, extrayendo mapas aún más pequeños y obteniendo el valor máximo de cada uno de los mapas, para transformar los parches, después de esto pasa por una capa shortcut, los cuales ayudan a aprender características de alto nivel que pueden perderse durante las tareas de batch normalization, de esta forma dichos cortos se encargan de obtener características significativas de capas superiores.



**Figura 4.9:** Bloque básico Conv1D PointCleanNet.

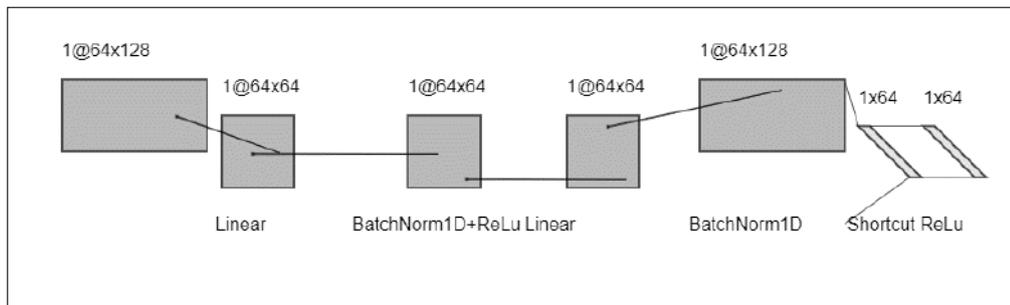
Las capas convolutivas (Conv1d) aplica una convolución 1D sobre una señal de entrada compuesta por varios planos de entrada. Estas capas son utilizadas para aprender patrones locales de la nube de puntos, este tipo de convoluciones son llamadas “mapas de características” (en inglés, Feature maps). Las capas convolutivas toman mapas de características de dimensiones  $n \times n$  y regresa un



**Figura 4.10:** Bloque básico Conv1D propuesto.

mapa de respuesta  $n \times n$  Esto quiere decir que por cada capa convolutiva se aplica un filtro y que cada filtro contiene un mapa de respuesta de  $[in\_maps * maps]$  valores.

En nuestra propuesta de red añadimos más capas convolutivas para obtener una mayor extracción de cara de características significativas, a las cuales nos van a ayudar a identificar valores atípicos dentro de la nube y presencia de ruido.



**Figura 4.11:** Bloque básico Lineal PointCleanNet.

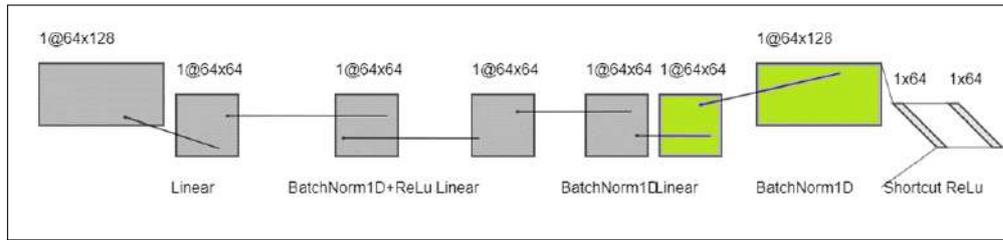
La figura 4.11 muestra una simple red feed-forward. Toma la entrada, la hace pasar por varias capas, una tras otra, y finalmente da la salida.

Aplica una transformación lineal a los datos entrantes:  $y = xA^T + b$  o  $y = xAT + b$

Es un módulo que crea una red de alimentación directa de una sola capa con  $n$  entradas y  $m$  salidas. Matemáticamente, este módulo está diseñado para calcular la ecuación lineal  $Ax = b$  donde  $x$  es entrada,  $b$  es salida,  $A$  es peso. De aquí proviene el nombre 'Lineal'.

### 4.3. Capas del modelo propuesto

Para el modelo de la red neuronal decidió utilizar capas convolucionales de  $1 \times 1$  ya que nos ayudan a identificar patrones locales en las nubes de puntos,



**Figura 4.12:** Bloque básico Lineal propuesta.

la arquitectura propuesta para las capas lineales puede observarse en la figura 4.12. Una de las principales ventajas de utilizar capas convolucionales es que son invariantes a la translación de los patrones que aprenden, esto quiere decir que, si un patrón aprendido se encontraba originalmente en la esquina superior derecha, y después ese patrón es trasladado a cualquier otra esquina de la imagen, la red convolucional será capaz de reconocerlo.

Otra gran ventaja de las redes convolucionales es que pueden aprender jerarquías espaciales de los patrones, es decir que, si en la primera capa la red neural aprendió como ejemplo los bordes de la oreja de un gato, en las capas posteriores podrá aprender un patrón que involucra los bordes de la oreja del gato, para poder aprender las características de la oreja.

Las capas convolutivas (Conv1d) aplica una convolución 1D sobre una señal de entrada compuesta por varios planos de entrada, la ecuación que representa esta convolución está dada por la ecuación 4.3.

$$out(N_i, C_{outj}) = bias(C_{outj}) + \sum_{k=0}^{C_{in}-1} weight(C_{outj}, k_i) * input(N_i, k) \quad (4.3)$$

donde  $*$  es el operador de correlación cruzada válido,  $N_i$  es un tamaño de lote, y  $C$  denota un número de canales.

También se utilizaron capas para normalización por lotes (también conocida como norma por lotes), ecuación 4.4. Es un método utilizado para hacer que las redes neuronales artificiales sean más rápidas y estables mediante la normalización

de las entradas de las capas por medio del re-centrado y el re-escalado, la arquitectura propuesta puede observarse en la figura 4.10.

$$y = \frac{x - E[x]}{\sqrt{Var[X] + \epsilon}} * \gamma + \beta \quad (4.4)$$

La media y la desviación estándar se calculan por dimensión sobre los minilotes.  $\gamma$  y  $\beta$  son vectores de parámetros aprendibles de tamaño  $C$  (donde  $C$  es el tamaño de la entrada). Por defecto, los elementos de  $\gamma$  se fijan en 1 y los elementos de  $\beta$  en 0. La desviación estándar se calcula mediante el estimador sesgado.

Estas capas son utilizadas para reducir “el cambio covariable interno” (internal covariance shift), que es producido por los parámetros de inicialización y cambios en la distribución de las entradas de cada una de las capas afectando la tasa de aprendizaje. Las capas para la normalización por lotes (batchnorm1d) normalizan la entrada a la capa haciendo un escalado y centrado de la entrada.

Se cree que puede mitigar el problema del desplazamiento interno de las covariables, en el que la inicialización de los parámetros y los cambios en la distribución de las entradas de cada capa afectan a la tasa de aprendizaje de la red.

Otro tipo de capa que se utilizan son de agrupación máxima (MaxPool1D) 4.5, estas capas son utilizadas para reducir la dimensionalidad de los mapas de características de cada capa. Esto reduce la posibilidad de tener un sobre entreno de la red (over-fitting), y también reduce el costo computacional de procesar los coeficientes de los mapas de características.

$$out(N_i, C_j, k) = \max_{m=0, \dots, kernel\ size - 1} input(N_i, C_j, strides * k + m) \quad (4.5)$$

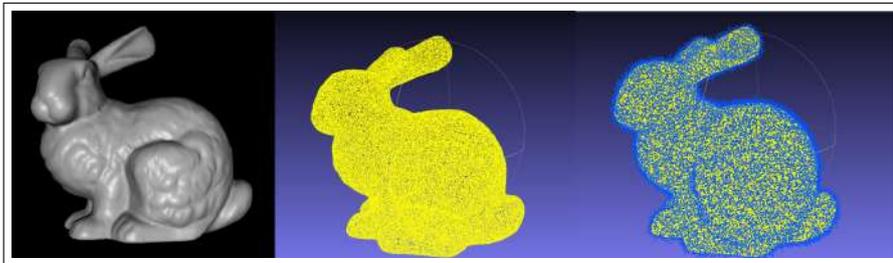
Donde “kernel size” tamaño de la ventana deslizante, debe ser  $>0$ . “Strides” es el paso de la ventana deslizante, debe ser  $>0$ .

La operación de “*max-pooling*” encuentra el valor máximo entre una ventana de muestra y pasa este valor como resumen de características sobre esa área. Como resultado, el tamaño de los datos se reduce por un factor igual al tamaño de la ventana de muestra sobre la cual se opera.

## 4.4. Base de datos

Se considero la base de datos de PointCleanNet para el entrenamiento y prueba de la red neuronal entrenada.

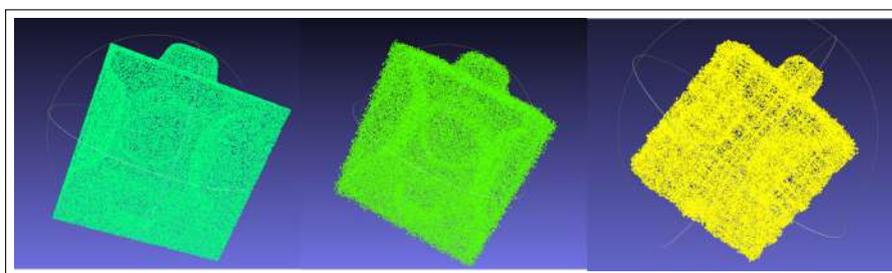
La base de datos está dividida en dos secciones, una dedicada para entrenamiento con 354 nubes y otra para prueba con 333. La base de datos contiene nubes de puntos sin ruido (referencia o groundtruth), y nubes con distintas magnitudes de ruido blanco y valores atípicos. Un ejemplo de un objeto reconstruido, puede observarse en la figura 4.13 donde se muestra de izquierda a derecha la reconstrucción en 3D del conejo de Standford, la nube de referencia y también una nube de puntos con ruido gaussiano.



**Figura 4.13:** Nube de puntos del conejo de Standford. Nube de puntos 3D tomados de (Rakotosaona et al., 2020).

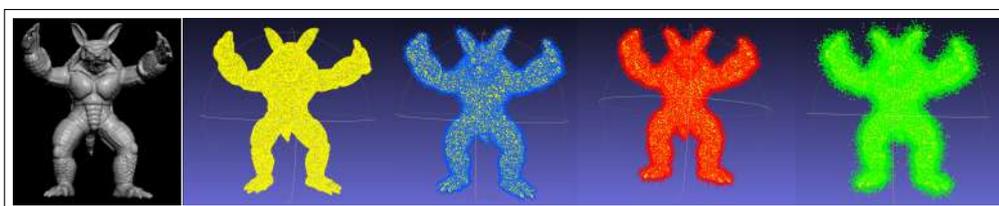
Esta base de datos fue publicada en 2019, puede encontrarse en el repositorio de PointCleanNet o también en PointCleanNet Dataset. La base de datos esta comprimida en el formato .7zip, y pesa 1.4 GB comprimida. Contiene 28 formas diferentes, de las cuales 18 figuras se encuentran divididas para el entrenamiento y 10 figuras para prueba. A partir de las mallas triangulares originales de cada forma se muestrearon 100.000 puntos, uniformemente al azar en la superficie, para generar una nube de puntos limpia o como valor de referencia (ground truth).

De acuerdo con (Rakotosaona et al., 2020) en total, el conjunto de nubes para el entrenamiento para la reducción de ruido contiene 108 variaciones de figuras, que surgen de 6 niveles de ruido (incluidos los puntos limpios) para cada uno de los 18 objetos, en la figura 4.15 se puede observar el objeto reconstruido de un armadillo en el primer recuadro, seguido de una nube de puntos sin ruido (armadillo amarillo), después un armadillo con ruido gaussiano con una desviación estándar de  $1e^{-2}$  (armadillo azul),  $2,5e^{-2}$  (armadillo rojo), y  $5e^{-2}$  (armadillo verde).



**Figura 4.14:** Caja de peluche sin ruido (izquierda), con ruido blanco (en medio) y nube después de la inferencia utilizando el modelo de PointCleanet (derecha). Nube de puntos 3D tomados de (Rakotosaona et al., 2020).

Para la tarea de eliminación de valores atípicos, se generaron figuras nuevas para entrenamiento y prueba, utilizando las nubes de puntos limpias y con un total de 140.000 puntos por forma, un ejemplo de una nube de puntos sin ruido y con ruido gaussiano con desviación estándar se puede observar en la figura 4.14.



**Figura 4.15:** Armadillo con distintas magnitudes de ruido blanco. De izquierda a derecha, nube de puntos sin ruido, nube con ruido Gaussiano  $1e^{-2}$ , nube con ruido Gaussiano  $2,5e^{-2}$ , Ruido Gaussiano  $5e^{-2}$ . Nube de puntos 3D tomados de (Rakotosaona et al., 2020).

Para generar los valores atípicos, añadimos ruido gaussiano con una desviación estándar del 20% de la diagonal del cuadro delimitador a un subconjunto aleatorio

de puntos.

## 4.5. Entrenamiento de la red

1. Para el entrenamiento debe de utilizar ambas nubes de puntos con ruido y nubes de puntos de referencia o ground truth.
2. Las imágenes deben de tener el formato “.xyz”, en caso de realizar inferencias con nubes de puntos externas.
3. La base de datos debe de dividirse entre prueba y validación.
4. Un archivo con extensión “.txt” debe de crearse para poder diferenciar y procesar las nubes de puntos de que corresponden a la sección de entrenamiento y también para la sección de validación.

Una vez definida la base de datos para entrenamiento y validación, debemos de entrenar y probar la red neuronal para que sea capaz de identificar los valores atípicos y el ruido dentro de la nube.

Primero debemos asegurarnos que la base de datos esté dividida en base de datos de prueba y entrenamiento, además debemos de conocer cuales son los puntos que no corresponden al objeto, de está forma tendremos nubes de referencia que nos permitan calcular las métricas de evaluación.

La red neuronal también se puede entrenar con nubes de puntos de las cuales se desconocen los valores atípicos, esto en caso de que se quiera utilizar alguna de las técnicas de aumento de datos, las cuales permiten lograr una mayor generalización.

### 4.5.1. Parámetros del entrenamiento

Los siguientes parámetros se utilizaron para el entrenamiento, la división para el entrenamiento de la base de datos fue segmentado en una relación 80-20. El número de épocas del entrenamiento es fijo, y no se implantaron condiciones de paro tempranas.

- Numero de épocas de entrenamiento: 600

- Tamaño de datos de entrenamiento: 64
- Radio de las nubes para cálculo de diagonal: 0.05
- Desviación estándar del número de puntos en un parche: 0
- Número de parches por nube de puntos: 100
- Número de trabajadores para carga de dato: 1
- Número de nubes cargadas en caché: 600
- Variable aleatoria para reproducción de datos: 3627473
- Orden de entrenamiento de las nubes: Aleatorio
- Tasa de aprendizaje: 0.0001
- Momento del gradiente descendiente: 0.9

### 4.5.2. Hiper-parámetros del modelo

La red neuronal se entrenó con los siguientes parámetros de entrada:

Algunos de los hiper-parámetros que tienen mayor importancia en el entrenamiento del modelo son:

- El factor de impulso o momentum, este coeficiente se aplica a un término adicional en la actualización de los pesos. Es una técnica que a menudo mejora tanto la velocidad como la precisión del entrenamiento.
- La tasa de aprendizaje es un parámetro de ajuste en un algoritmo de optimización que determina el tamaño del paso en cada iteración mientras se mueve hacia un mínimo de una función de pérdida.
- Número de épocas: Es uno de los parámetros mas importantes en el entrenamiento, ya que afecta de forma proporcional el sub entrenamiento o sobre entrenamiento de la red y de esta forma impactar directamente en los resultados obtenidos.

La red neuronal utiliza como entrada un tensor de la forma  $(x, y, z)$ . Para el entrenamiento de esta red neuronal se utilizó  $(100\ 000, 100\ 000, 100\ 000)$ , es decir nubes densas, es importante tomar en cuenta la densidad de las nubes,

ya que a mayor densidad de las nubes mayor gasto computacional va a ser requerido.

Otros factores que afectan el tiempo del entrenamiento son: El número de los parches o características locales extraídas de las capas convolutivas, debido al gasto computacional que pueden producir utilizar varias capas convolutivas. Las dimensiones de los mapas de características, que corresponde al número de filtros que se aplican durante la convolución. La función de activación de cada una de las capas, esta puede impactar directamente en que tan bien se generaliza, y está en función del problema que se intenta resolver.

## 4.6. Criterios de evaluación

Cuando intentamos reducir el nivel de valores atípicos en una nube de puntos debemos de considerar que los puntos que se están removiendo con el modelo propuesto correspondan a los valores atípicos (True Positive), que la cantidad de puntos que se remuevan erróneamente sean cuantificados (False Positive). Estas dos métricas contribuyen a la Precisión, ecuación 4.6, Recall, ecuación 4.7, F1-Score, ecuación 4.8 y el Error Cuadrático Medio, ecuación 4.9 del modelo 4.4.

$$Precision = \frac{\sum truepositive}{\sum truenegative} \quad (4.6)$$

$$TruePositive, Recall = \frac{\sum truepositive}{\sum truenegative} \quad (4.7)$$

$$F1 - Score = 2 * \frac{\sum truepositive}{\sum Positivecondition} \quad (4.8)$$

$$mse = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (4.9)$$

La ecuación de recall nos permite determinar de todas las clases positivas, cuantas de ellas nosotros predijimos correctamente. Nos conviene que intentemos que el

recalce a lo más alto posible.

La ecuación de precisión indica de todas las clases positivas y negativas, cuantas de ellas hemos predicho correctamente, en un modelo es conveniente tener una precisión alta.

Puede ocurrir que al evaluar nuestro modelo obtengamos un recall alto y una precisión baja o viceversa, para este caso existe una métrica que nos permite comparar la precisión y el recall de forma sencilla y correlacionada, esta métrica se llama F1-Score.

La matriz de confusión es comúnmente utilizada para la evaluación de algoritmos de inteligencia artificial, ya que nos permite visualizar el comportamiento de nuestro algoritmo en base a métricas que permiten identificar sesgos dentro de la base de datos utilizada, es importante mencionar que para poder utilizar la matriz de confusión en nubes de puntos, es indispensable tener una imagen de referencia, además es necesario identificar los puntos que corresponden a valores atípicos o ruido dentro de la nube.

# Capítulo 5

## Resultados

En esta sección se describen los resultados cuantitativos y cualitativos obtenidos después de realizar las inferencias para la reducción de valores atípicos utilizando el modelo propuesto por PointCleanNet.

### 5.1. Resultados Cuantitativos

La tabla 5.1 muestra los resultados de las inferencias de las nubes de puntos con presencia de valores atípicos realizadas con el modelo propuesto por PointCleanNet, cada columna de la tabla representa las métricas de evaluación así como el tiempo de ejecución en GPU.

Se realizaron inferencias para 144 nubes de puntos con distintas magnitudes de ruido con una desviación estándar de 1 %, 2.5 % y 5 % y distinta densidad como se muestra en la tabla 5.1. La densidad de la nube está en función del porcentaje de puntos que se removieron, siendo de menor densidad aquellas nubes a las que se removió mayor porcentaje de puntos. Las inferencias se realizaron utilizando la estación de trabajo definida en la sección de material utilizado.

Se puede observar en la tabla 5.1. que para una densidad de 112, 000 puntos se obtiene una Precisión del 87.32 % , Recall del 84.08 %, F1-Score del 91.48 %, MSE del 05.09 %, y tiempo de ejecución en GPU de 68 minutos. Para esta densidad de puntos el error cuadrático indica que el rendimiento del modelo no se degrada con

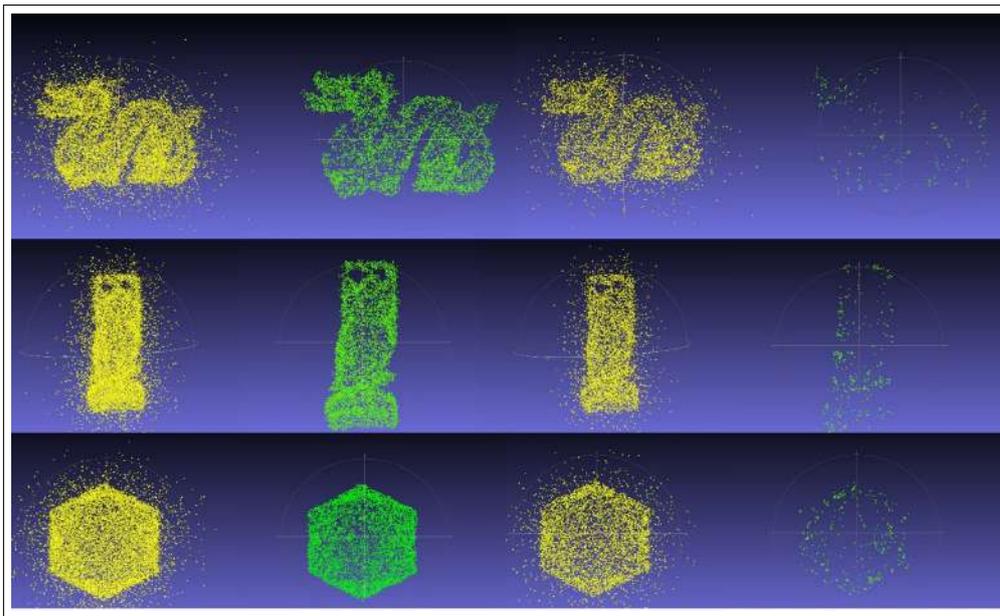
| Comparación de resultados cualitativos. |             |          |            |       |                                   |
|---|-------------|----------|------------|-------|-----------------------------------|
| Densidad de puntos                      | Precisión % | Recall % | F1-score % | MSE % | GPU Tiempo de ejecución (minutos) |
| 14, 000                                 | 29.78       | 100      | 45.89      | 72.71 | 8.58                              |
| 28, 000                                 | 29.23       | 99.75    | 45.23      | 68.50 | 17.14                             |
| 42, 000                                 | 32.84       | 99.26    | 49.39      | 58.14 | 25.71                             |
| 56, 000                                 | 39.64       | 96.72    | 56.57      | 42.41 | 34.24                             |
| 70, 000                                 | 50.19       | 94.68    | 66.40      | 27.92 | 42.78                             |
| 84, 000                                 | 63.70       | 92.13    | 77.05      | 16.31 | 51.31                             |
| 98, 000                                 | 77.93       | 87.43    | 86.33      | 08.59 | 59.85                             |
| 112, 000                                | 87.32       | 84.08    | 91.48      | 05.09 | 68.57                             |
| 126, 000                                | 95.38       | 80.75    | 95.03      | 02.84 | 77.14                             |
| 140,000                                 | 99.61       | 90.78    | 94.99      | 02.75 | 85.71                             |

**Tabla 5.1:** Comparación de resultados cualitativos de los resultados de ejecución en GPU

la reducción de la densidad de puntos. Por otra parte, los resultados obtenidos en nubes de puntos con una densidad de 84, 000 puntos indican una degradación en la clasificación de los valores atípicos, en comparación con los resultados obtenidos con una densidad de 112,000 puntos.

## 5.2. Resultados Cualitativos

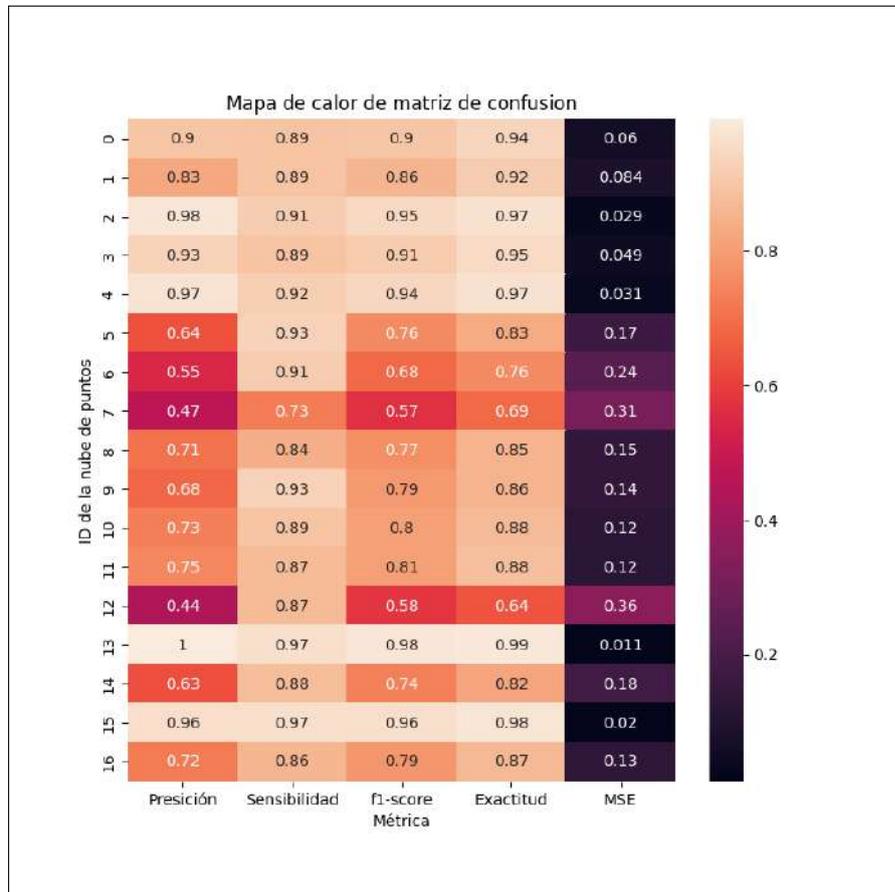
La siguiente imagen muestra tres figuras distintas, las figuras cuentan con distinto nivel de valores atípicos y distinta densidad. Cabe recalcar que las inferencias corresponden al análisis hecho para 126,000 y 84,000 puntos respectivamente, el propósito principal es mostrar el punto en el que el rendimiento del modelo conlleva a una degradación en la clasificación de valores atípicos. Nubes de puntos con una densidad de 84,000 puntos o menor, tienen a degradar el rendimiento del modelo. Por otro lado el tiempo de inferencia disminuye cuando la densidad es mayor a 112,000 puntos, donde el rendimiento de la red no se ve perjudicado obteniendo F1-Score de 91.48 %.



**Figura 5.1:** Comparativa de resultados obtenidos para nubes de puntos con presencia de valores atípicos. De izquierda a derecha y de arriba abajo: Primer columna: Nubes con 126,000 puntos y valores atípicos. Segunda columna: Inferencia utilizando el modelo propuesto por PointCleanNet para las nubes de 126, 000 puntos. Tercer Columna: Nubes con 84,000 puntos y valores atípicos., Cuarta columna: Inferencia utilizando el modelo propuesto por PointCleanNet para las nubes de 84, 000 puntos.

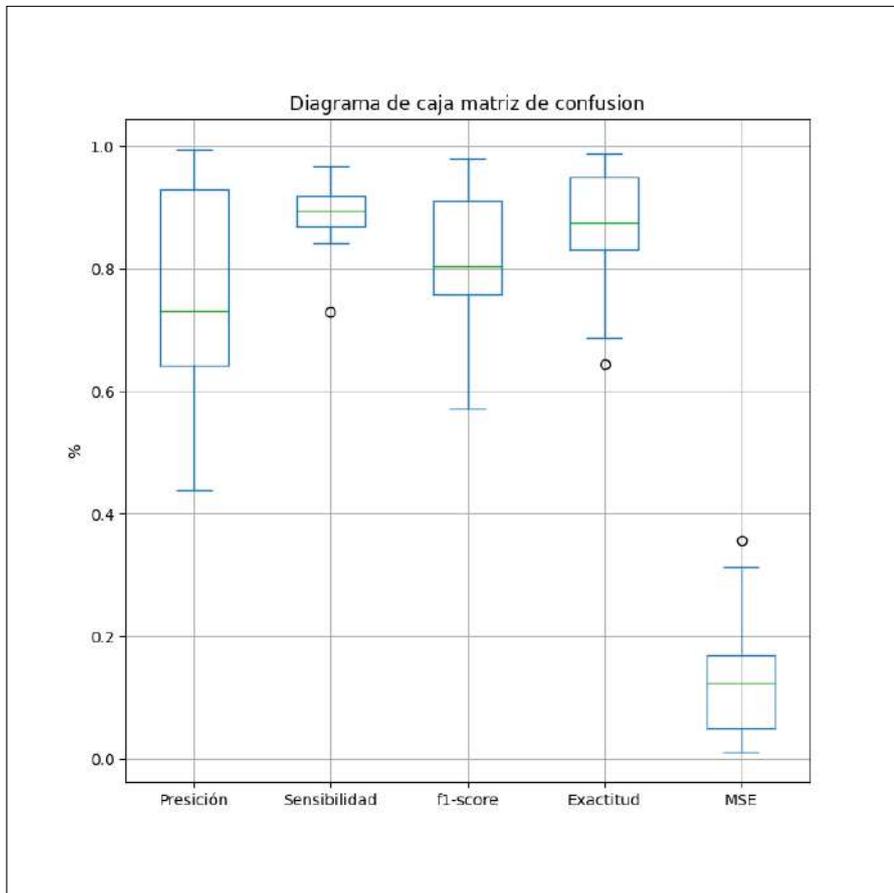
En el modelo de PointCleanNet podemos observar en la siguiente comparación obtenida a través de la matriz de confusión que si bien la precisión está en un rango de entre el 60 % y el 90 %, métricas como recall y F1-score se encuentran debajo del 80 % como se muestra en la figura 5.2. Representado de otra forma

podemos observar en el diagrama de caja figura 5.3 que la mediana de la precisión se encuentra por debajo del 80 %



**Figura 5.2:** Resultados obtenidos con la matriz de confusión, estas inferencias se realizaron con el modelo propuesto por el PointCleanNet.

Si bien los resultados obtenidos con la primera inferencia (figura 5.5) del modelo propuesto realizada únicamente con 300 iteraciones no son tan favorables, podemos observar que el balance entre las métricas de precisión, recall, F1-score no se encuentran dispersas como se muestra en la figura 5.7. También haciendo referencia en el diagrama de caja del primer modelo propuesto figura 5.6, podemos observar que la mediana de la precisión aumento con respecto a la inferencia hecha con el modelo de PointCleanNet.



**Figura 5.3:** Diagrama de caja de las métricas aplicadas a las inferencias con el modelo de PointCleanNet.

```

class ResPointNetfeat(dim, num_points, sym_op, num_scale)

self.stn1 = ResSTN(num_scales=self.num_scales, num_points=num_points*self.point_tuple, dim=3, sym_op=self.sym_op, quaternion=True)
self.stn2 = ResSTN(num_scales=self.num_scales, num_points=num_points, dim=64, sym_op=self.sym_op)
self.b0a = BasicBlock(3*self.point_tuple, 64, conv = True)
self.b0b = BasicBlock(64, 64, conv=True)

self.b1 = BasicBlock(64, 64, conv = True)
self.b2 = BasicBlock(64, 128, conv = True)
self.b3 = BasicBlock(128, 1024, conv = True)

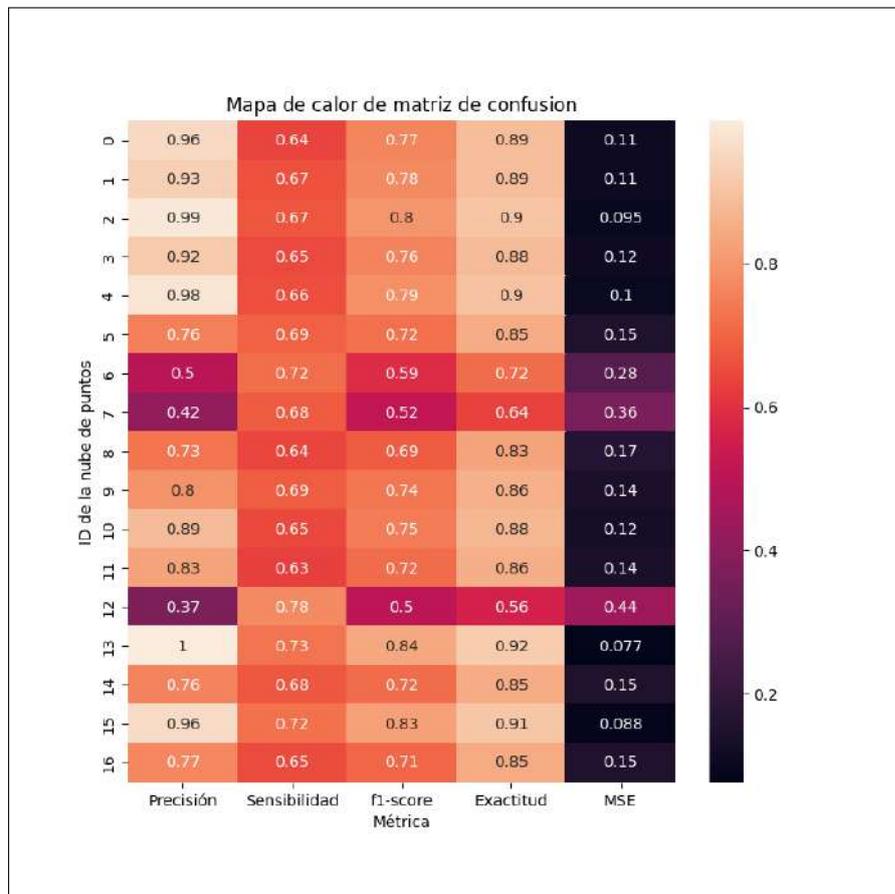
self.b4 = BasicBlock(1024, 1024*self.num_scs, conv = True)
self.mp1 = torch.nn.MaxPool1d(num_points)

torch.FloatTensor()

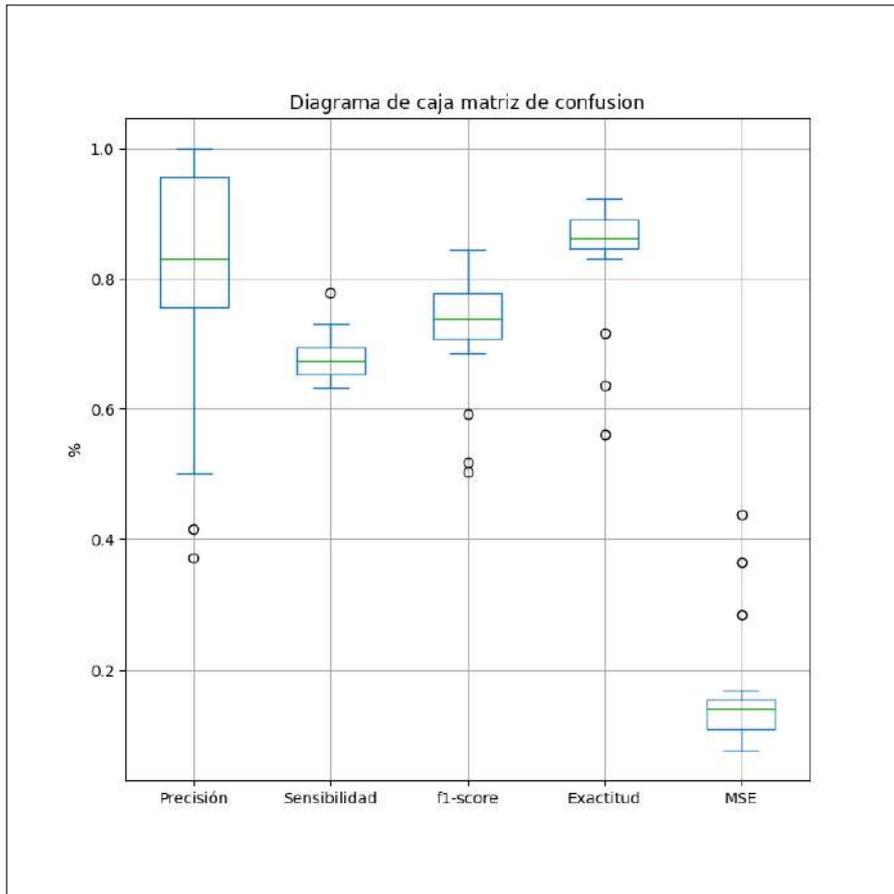
```

**Figura 5.4:** Bloque básico para la reducción de valores atípicos.

En la segunda inferencia descrita por la figura 5.7 podemos observar que los valores de MSE, en la que aumentó el número épocas en el entrenamiento a 600, son aceptables debido a que se encuentran por debajo del umbral de referencia en comparación con los resultados de PointCleanNet. Se puede observar en la figura 5.5, que existe un alto porcentaje en la precisión del modelo para la detección de valores atípicos, esto indica una apropiada clasificación, más sin embargo vale la pena prestar atención a métricas f1-score y recall ya que pueden indicar un desvalance entre las clases. El aumento en la precisión se puede observar con el diagrama de cajas de la figura 5.8 donde podemos observar que la mediana se encuentra cerca del tercer cuartil y superior al 80 % en comparación con los resultados obtenidos con el modelo de PointCleanNet.

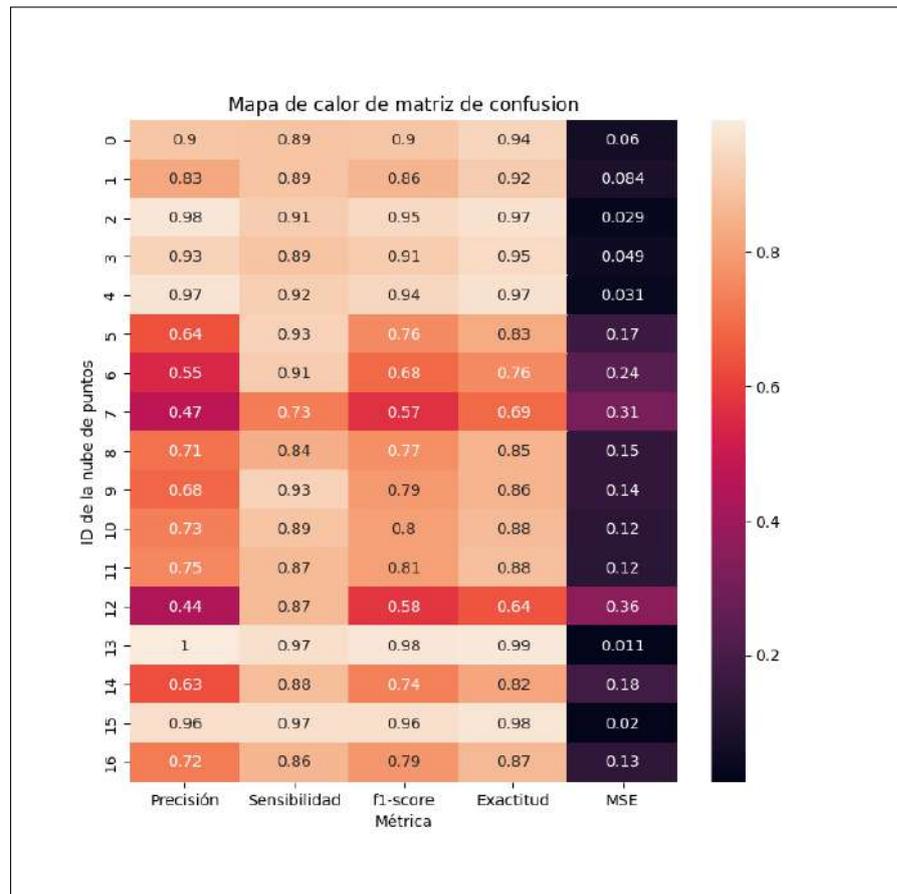


**Figura 5.5:** Resultados obtenidos con la matriz de confusión de distintas nubes de puntos, estas inferencias se realizaron con nuestra propuesta de modelo.



**Figura 5.6:** Diagrama de caja de las métricas aplicadas a las inferencias con el modelo propuesto a 300 iteraciones.

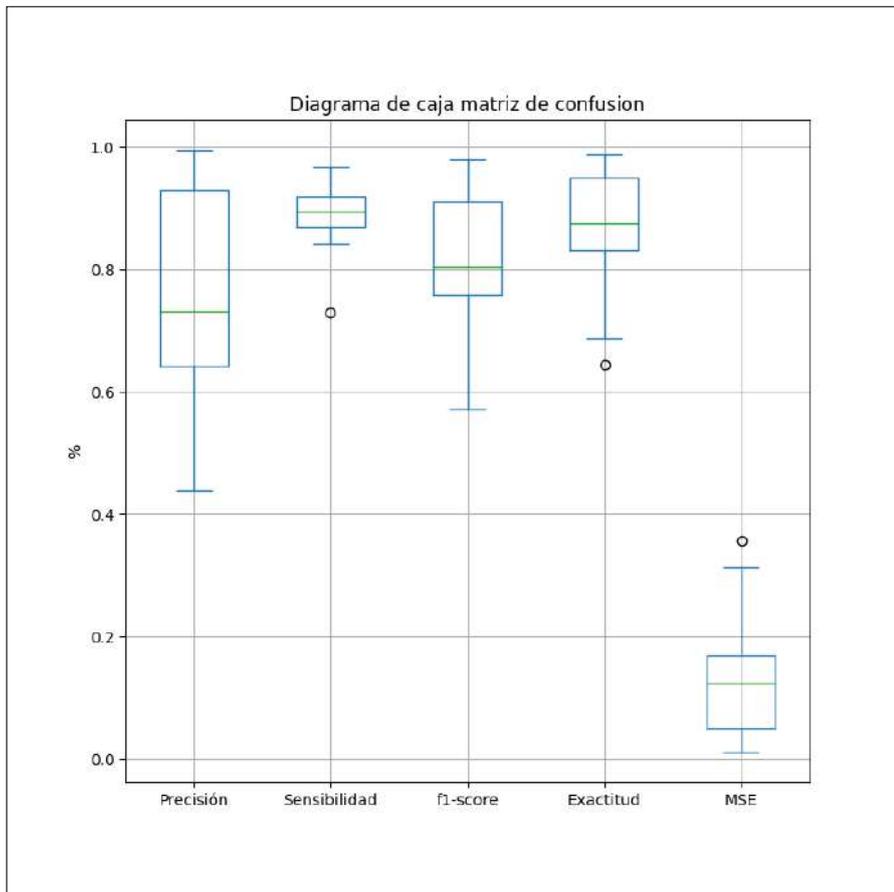
Una de las razones principales por las que se está trabajando con bloques básicos, es que son utilizados en distintos niveles de la red neuronal, debido a que en algunas secciones de la red es más conveniente extraer mapas de características con capas convolutivas y en otros niveles de la red es más importante la generalización de las características extraídas, y para este tipo de generalización tiende a recurrirse a a capas con operaciones lineales y la clase para el bloque básico permite poder intercambiar entre los distintos tipos de capas de forma eficiente. En la figura 5.4 se puede observar la implementación de los bloques básicos.



**Figura 5.7:** Resultados obtenidos con la matriz de confusión de distintas nubes de puntos, estas inferencias se realizaron con nuestra propuesta de modelo, además se incrementó el número de épocas a 600.

Se decidió utilizar la métrica de precisión porque podemos medir el desempeño del nuestro modelo en tareas de clasificación para outliers. En este caso la precisión nos permite contestar es la respuesta a la pregunta ¿qué porcentaje de los outlier's que el modelo identifique serán outliers reales?.

La métrica de exhaustividad (Recall) nos indica del total de outliers en la nube, que porcentaje se clasificaron correctamente. La sensibilidad (recall) es la respuesta a la pregunta ¿qué porcentaje de outliers reales el modelo es capaz de identificar?, por último y no menos importante el valor F1, este valor se utiliza para combinar las medidas de precisión y recall como una correlación entre ambas mediciones. Esto es práctico porque hace más fácil el poder comparar el rendimiento combinado de la precisión y la exhaustividad entre varias soluciones, esto se puede observar en la figura 5.8.



**Figura 5.8:** Diagrama de caja de las métricas aplicadas a las inferencias con el modelo propuesto y 600 iteraciones.

La inferencia para el procesamiento de una nube con 140,000 puntos toma en promedio 85 minutos 42 segundos. Para una nube con 112,000 puntos toma en promedio 68 minutos 34 segundos. Reducir el tamaño de una nube de puntos en un 20 % contribuye a la reducción del tiempo de inferencia en un 24.99 %. El f1-score muestra que al remover el 20 % de los puntos se obtiene 91.48 %, es

aquí cuando la reducción del tiempo de ejecución alcanza su óptimo, ya que se preserva la clasificación de los valores atípicos. Por el contrario, analizando los resultados de la nube con una densidad de 84,000 puntos, esta toma en promedio 51 minutos 30 segundos, pero la reducción del rendimiento para la clasificación de nubes de puntos es evidente con un 77.05 % de F1-score.

Al hacer las inferencias en las nubes de puntos podemos observar que uno de los factores que afectan a la reducción de ruido y valores atípicos es la forma o la figura de las nubes de puntos. Nubes de puntos con formas complejas como las presentadas en el análisis, se ven afectadas dando como resultado a pérdida de características en la nube. El tipo de ruido y la distribución de este puede perjudicar la calidad de la nube, se realizó el experimento con distintos niveles de ruido para probar la eficacia de la red en la remoción de ruido y valores atípicos de las nubes de puntos, pudimos concluir que entre mayor sea la dispersión del ruido más complicado será para la red neuronal eliminar los valores atípicos.

Adicional al análisis con la base de datos creada, se utilizaron nubes de puntos para el entrenamiento, validación y la inferencia de la red neuronal, las figuras utilizadas en el análisis fueron tomadas de la base de datos ya existente propuesta por PointCleanNet, dichas nubes de puntos proporcionaban diversas figuras con distintas propiedades que permitieron realizar aumento de datos a la base de datos y de esta forma generalizar aún más el entrenamiento de la red.

Comenzando con las ventajas del modelo propuesto en comparación con el modelo propuesto por PointCleanNet, podemos observar que la precisión aumenta considerablemente para algunas de las figuras, esto debido a los cambios realizados en la arquitectura de la red, es importante mencionar que además, después de cada iteración se está determinando la precisión del modelo, esto anteriormente no se realizaba con el modelo propuesto por PointCleanNet, lo que permite tener un modelo que permite clasificar los valores atípicos de forma más eficiente.

Continuando con las limitaciones, existen algunas nubes de puntos en las cuales la precisión es considerablemente baja, esto es debido a la forma de la figura, es decir, figuras más complejas tienden a hacer que la probabilidad de que un punto

de la nube sea un valor atípico sea muy baja, por debajo del 50 %. Una de las posibles soluciones para este problema consiste en modificar la sensibilidad del modelo para determinar si un punto es atípico o no, descrito en la ecuación 4.2. Otra posible solución sería incrementar las nubes de puntos complejas en la base de datos de entrenamiento.

## Capítulo 6

### Conclusión

Las nubes de puntos procesadas permitieron analizar los resultados de las inferencias para verificar bajo qué circunstancias la red neuronal tiene resultados aceptables, es decir cuando la degradación de las características de la figura en si se ven perjudicadas, esto para determinar hasta qué punto es recomendable reducir la densidad de la nube, además permite determinar cuando la densidad de la nube afecta de manera significativa el rendimiento de la red.

Al hacer la inferencia en la nube de puntos podemos observar que uno de los factores que afectan a la reducción de ruido y valores atípicos es la forma o la figura de las nubes de puntos, así como el nivel de ruido presente, esto normalmente es generado por limitantes del sensor o factores externos como la luz ambiental o la forma del objeto reconstruido. Nubes de puntos con formas complejas se ven afectadas dando como resultado a la pérdida de características en la nube

Una consideración que se debe de tener al momento de utilizar bases públicas, es la integridad de la base de datos, es decir antes de aplicar cualquier tipo de modelo ya sea de clasificación o predicción es indispensable conocer las nubes de puntos que se vayan a procesar, en el sentido de saber que tipo de distribución tiene, si hay datos faltantes, plantear estrategias para crear nuevas nubes de puntos con valores atípicos y ruido, así como determinar algún tipo de correlación en los datos. Para este caso en específico se optó por utilizar las nubes de puntos

limpias para generar nuevas figuras con ruido y valores atípicos con distinta distribución.

Una de las grandes ventajas de utilizar la distribución gaussiana para generar ruido y valores atípicos en las nubes, es que debido a su naturaleza estadística no fue necesario de disponer de grandes recursos computacionales, pudimos observar que dependiendo de la distribución y cantidad de ruido en las nubes influye en la capacidad de la red en la clasificación final, lo que hace que los resultados sean interpretados con facilidad.

Es importante conocer distintos métodos de validación del modelo, K fold cross es una metodología estocástica que nos permite evaluar el modelo con distintas permutaciones de la base de datos. A su vez la matriz de confusión es un método determinístico que nos permite conocer el rendimiento del algoritmo en base a distintas métricas como precisión, recall, f1-score, y MSE.

La matriz de confusión nos permite medir el desempeño con distintas métricas, las cuales se utilizan directamente en la nube de referencia. Métricas como son el accuracy, la precisión, la sensibilidad y la especificidad. permiten determinar la capacidad de la red para la propia clasificación y eliminación de valores atípicos. También podemos obtener mediciones como el F1 score que nos permite analizar la precisión y la sensibilidad de forma simultanea.

Cabe mencionar que la precisión es especialmente útil cuando la base de datos es simétrica, es decir cuando la cantidad de falsos negativos y falsos positivos es similar. F1 score tiende a ser un buen indicador cuando las clases tienen una distribución desigual. De igual forma debemos de prestar atención al accuracy si en lo que estamos interesados es en los True Positive, por el contrario, si la métrica de interés es son los True Negative entonces debemos de prestar atención a la especificidad.

## 6.1. Trabajo futuro

Las siguientes son mejoras que pueden ser exploradas para aumentar el rendimiento en la red propuesta:

1. Aumento de la base de datos utilizando las nubes de puntos con ruido con una distribución que no sea gaussiana, por ejemplo nubes de puntos de escenas reales captadas con una cámara RGBD.
2. Proponer una nueva arquitectura, utilizando aprendizaje por transferencia y modificar únicamente las capas de convolución.
3. Hacer un análisis de los hiper-parámetros y determinar que métodos de ajuste fino se pueden utilizar para obtener un mejor rendimiento.
4. Reducción de muestreo de las nubes de puntos de la base de datos utilizando un filtro de cuadrícula de vóxeles.
5. Reducir la dimensionalidad de la red neuronal utilizando métodos de poda neuronal.

## Bibliografía

- Albawi, S., Mohammed, T. A., and Al-Zawi, S. (2017). Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)*, pages 1–6. Ieee.
- Amenábar Montenegro, S. (2021). Data augmentation helps to prevent shortcuts and learn representations for continual learning in neural networks.
- An, S., Lee, M., Park, S., Yang, H., and So, J. (2020). An ensemble of simple convolutional neural network models for mnist digit recognition. *arXiv preprint arXiv:2008.10400*.
- Barron, J. T. and Malik, J. (2013). Intrinsic scene properties from a single rgb-d image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 17–24.
- Bookstein, F. L. (1989). Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on pattern analysis and machine intelligence*, 11(6):567–585.
- Chollet, F. (2021). *Deep learning with Python*. Simon and Schuster.
- Curless, B. and Levoy, M. (1996). A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312.
- Dou, M., Taylor, J., Fuchs, H., Fitzgibbon, A., and Izadi, S. (2015). 3d scanning deformable objects with a single rgb-d sensor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 493–501.
- Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. (2020). Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.
- Giancola, S., Valenti, M., and Sala, R. (2018). *A survey on 3D cameras: Metrological comparison of time-of-flight, structured-light and active stereoscopy technologies*. Springer.
- Guerrero, P., Kleiman, Y., Ovsjanikov, M., and Mitra, N. J. (2018). Pcpnet learning local shape properties from raw point clouds. In *Computer Graphics Forum*, volume 37, pages 75–85. Wiley Online Library.

- Hagebeuker, D.-I. B., Marketing, P., et al. (2007). A 3d time of flight camera for object detection. *PMD Technologies GmbH, Siegen*, 2.
- Haykin, S. S. et al. (2009). Neural networks and learning machines/simon haykin.
- Henry, P., Krainin, M., Herbst, E., Ren, X., and Fox, D. (2014). Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments. In *Experimental robotics*, pages 477–491. Springer.
- Javaheri, A., Brites, C., Pereira, F., and Ascenso, J. (2017). Subjective and objective quality evaluation of 3d point cloud denoising algorithms. In *2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pages 1–6. IEEE.
- Kauderer-Abrams, E. (2017). Quantifying translation-invariance in convolutional neural networks. *arXiv preprint arXiv:1801.01450*.
- Keller, M., Lefloch, D., Lambers, M., Izadi, S., Weyrich, T., and Kolb, A. (2013). Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *2013 International Conference on 3D Vision-3DV 2013*, pages 1–8. IEEE.
- Kim, P. (2017). Convolutional neural network. In *MATLAB deep learning*, pages 121–147. Springer.
- Levoy, M., Pulli, K., Curless, B., Rusinkiewicz, S., Koller, D., Pereira, L., Ginzton, M., Anderson, S., Davis, J., Ginsberg, J., et al. (2000). The digital michelangelo project: 3d scanning of large statues. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 131–144.
- Lindner, M., Kolb, A., and Hartmann, K. (2007). Data-fusion of pmd-based distance-information and high-resolution rgb-images. In *2007 International Symposium on Signals, Circuits and Systems*, volume 1, pages 1–4. IEEE.
- Litomisky, K. (2012). Consumer rgb-d cameras and their applications. *Rapport technique, University of California*, 20:28.
- Mikołajczyk, A. and Grochowski, M. (2018). Data augmentation for improving deep learning in image classification problem. In *2018 international interdisciplinary PhD workshop (IIPhDW)*, pages 117–122. IEEE.
- Morell, V., Saval-Calvo, M., Azorin-Lopez, J., Garcia-Rodriguez, J., Cazorla, M., Orts-Escolano, S., Fuster-Guilló, A., et al. (2014). A comparative study of registration methods for rgb-d video of static scenes.
- Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohi, P., Shotton, J., Hodges, S., and Fitzgibbon, A. (2011). Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pages 127–136. IEEE.
- Nguyen, T.-S., Stueker, S., Niehues, J., and Waibel, A. (2020). Improving sequence-to-sequence speech recognition training with on-the-fly data augmentation. In

- ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7689–7693. IEEE.
- Ning, X., Li, F., Tian, G., and Wang, Y. (2018). An efficient outlier removal method for scattered point cloud data. *PloS one*, 13(8):e0201280.
- Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660.
- Qu, X., Wang, W., Lu, K., and Zhou, J. (2018). Data augmentation and directional feature maps extraction for in-air handwritten chinese character recognition based on convolutional neural network. *Pattern Recognition Letters*, 111:9–15.
- Rakotosaona, M.-J., La Barbera, V., Guerrero, P., Mitra, N. J., and Ovsjanikov, M. (2020). Pointcleannet: Learning to denoise and remove outliers from dense point clouds. In *Computer Graphics Forum*, volume 39, pages 185–203. Wiley Online Library.
- Retting, R. A., Williams, A. F., Farmer, C. M., and Feldman, A. F. (1999). Evaluation of red light camera enforcement in oxnard, california. *Accident Analysis & Prevention*, 31(3):169–174.
- Rozsa, A., Rudd, E. M., and Boulton, T. E. (2016). Adversarial diversity and hard positive generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 25–32.
- Rusu, R. B., Marton, Z. C., Blodow, N., Dolha, M., and Beetz, M. (2008). Towards 3d point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927–941.
- Sankaranarayanan, J., Samet, H., and Varshney, A. (2007). A fast all nearest neighbor algorithm for applications involving large point-clouds. *Computers & Graphics*, 31(2):157–174.
- Schoenenberger, Y., Paratte, J., and Vanderghenst, P. (2015). Graph-based denoising for time-varying point clouds. In *2015 3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON)*, pages 1–4. IEEE.
- Shorten, C. and Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48.
- Sotoodeh, S. (2006). Outlier detection in laser scanner point clouds. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(5):297–302.
- Weber, T., Hänsch, R., and Hellwich, O. (2015). Automatic registration of unordered point clouds acquired by kinect sensors using an overlap heuristic. *ISPRS Journal of Photogrammetry and Remote Sensing*, 102:96–109.
- Weinmann, M. et al. (2016). *Reconstruction and analysis of 3D scenes*. Springer.

---

Zollhöfer, M., Stotko, P., Görlitz, A., Theobalt, C., Nießner, M., Klein, R., and Kolb, A. (2018). State of the art on 3d reconstruction with rgb-d cameras. In *Computer graphics forum*, volume 37, pages 625–652. Wiley Online Library.

# Apéndice A

## Anexos

### A.0.1. Artículo presentado y aceptado en COMIA 2021

---

Artículo aceptado y presentado en COMIA 2021

## **Caracterización de valores atípicos en nube de puntos en 3D para la reducción del tiempo de ejecución en memoria**

Israel Sotelo Rodríguez<sup>1</sup>, Jesús Carlos Pedraza Ortega<sup>1</sup>, Luis Rogelio Román Rivera<sup>2</sup>, Juan Manuel Ramos Arreguín<sup>1</sup>, Efrén Gorrostieta Hurtado<sup>1</sup>

<sup>1</sup> Universidad Autónoma de Querétaro Campus Aeropuerto, Maestría en Ciencias en Inteligencia Artificial, Querétaro, Querétaro, México

<sup>2</sup> Universidad Autónoma de Querétaro Cerro de las campanas, Doctorado en Ingeniería, Querétaro, Querétaro, México

isotelo17@alumnos.uaq.mx, lronman26@alumnos.uaq.mx,  
caryoko@yahoo.com, jsistdig@yahoo.com.mx,  
efren.gorrostieta@gmail.com

**Resumen.** Para poder realizar la reconstrucción de objetos en 3D, puede recurrirse a un tipo de estructura geométrica llamada "Nube de puntos". Una nube de puntos es un conjunto de coordenadas que representan la forma o superficie de un objeto. Uno de los principales problemas al hacer el mapeo 3D de escenas u objetos es la presencia de ruido y valores atípicos en una nube de puntos, producido por distintos factores como la luz solar o artificial, la reflexividad de los objetos, la geometría o limitantes relacionadas con el sensor [1]. La caracterización de los valores atípicos se realizó utilizando nubes de puntos con ruido blanco en distintas magnitudes, y nubes de puntos sin ruido, tomadas de la base de datos de Rakotosaona [2]. Se redujo el porcentaje de coordenadas de puntos, con intervalos de 10% en un rango de 10% a 90%. Se realizaron 144 inferencias utilizando el modelo de PointCleanNet [3] y el resultado se comparó con la etiqueta de los valores atípicos de cada una de las nubes, obteniendo una matriz de confusión para medir la precisión, valor F, y el error cuadrático medio. La matriz de confusión para cada nube mostró que al remover 20% o menos de los puntos pueden alcanzarse inferencias hasta con un 91% de precisión contribuyendo a la disminución del tiempo de inferencia.

**Palabras clave:** Nube de puntos, inferencia, matriz de confusión, caracterización, valores típicos.

### **3D point cloud outlier characterization for memory runtime reduction.**

**Abstract.** To reconstruct 3D objects, a type of geometric structure called "point cloud" can be used. A point cloud is a set of coordinates that represent the shape or surface of an object. One of the main problems when doing 3D mapping of scenes or objects is the presence of noise and outliers in the point cloud, produced

by different factors such as sunlight or artificial light, object reflectivity, geometry, or sensor-related constraints [1]. Outlier characterization was performed using point clouds with noise at different magnitudes, and point clouds without noise, taken from the Rakotosaona database [2]. The percentage of point coordinates was reduced, within 10% intervals in a range from 10% to 90%. One hundred forty-four inferences were performed using the PointCleanNet model [3]. Each inference was compared with the outlier label obtaining a confusion matrix to measure the precision, F1-score, and mean square error. The confusion matrix for each cloud showed that by removing 20% or less of the points, inferences with up to 91% accuracy can be achieved, contributing to a decrease in inference time.

**Keywords:** Point cloud, inference, confusion matrix.

### 1.- Introducción

Cuando un objeto en tercera dimensión es escaneado para obtener una nube de puntos normalmente es contaminada por distintos tipos de ruido o magnitudes, esto provoca que las nubes de puntos tengan que someterse a un preprocesamiento para poder realizar tareas de segmentación o clasificación de forma eficiente. Existen diversas metodologías para remover los valores atípicos de una nube de puntos, algunas están basadas en extraer características de puntos vecinos para poder clasificar los valores atípicos y reducirlos. La ecuación (1) describe una nube de puntos con presencia de valores atípicos. Dónde  $P_i$ , describe el conjunto de puntos de la superficie del objeto y  $O$  describe el conjunto de valores atípicos presentes [3].

$$P = \{p_1, \dots, p_n\} \cup \{O_j\}_{O_j \in O} \quad (1)$$

De manera general se establecen las siguientes metodologías utilizadas para la reducción de ruido y valores atípicos en nubes de puntos [4]. La reducción de valores atípicos por radio establece que los puntos que corresponden a los valores atípicos tienen menor densidad de puntos vecinos que los puntos que conforman al objeto [10]. La reducción de valores atípicos por esparcimiento utiliza la distancia entre puntos y el número de vecinos, asumiendo que los valores atípicos tienen una distribución normal y que los puntos que tienen en promedio dos sigmas como desviación estándar deben de ser clasificados como valores atípicos [11]. PointCleanNet es un algoritmo que estima los puntos vecinos, para cada punto de la nube de puntos, esto permite que nubes de puntos densas puedan ser procesadas sin perder características esenciales. Los puntos vecinos se calculan utilizando un radio  $r$ , el radio es calculado en base a la diagonal descrita por los valores máximos y mínimos de los puntos frontera.

Artículo aceptado y presentado en COMIA 2021

Este tipo de algoritmos están pensados para ser ejecutados en GPU. Las nubes de puntos son cada vez más densas debido a la resolución de los sensores utilizados para el escaneo o reconstrucción de superficies u objetos, produciendo que cada vez más puntos tengan que ser cargados en memoria [8].

Existen distintas propuestas para evitar el consumo excesivo de memoria para el procesamiento de los puntos vecinos de una nube, por ejemplo, el algoritmo “*k nearest neighbors*” [5] que utiliza la localidad de puntos para intentar reducir el tiempo de ejecución. El tiempo de ejecución para el procesamiento de una nube de puntos está en función de la densidad de la nube de puntos y el algoritmo para la estimación de los puntos vecinos [8].

### 1.1- Trabajos relacionados

El aprendizaje profundo (en inglés, Deep Learning) es un subcampo del aprendizaje automático. Es una metodología en la que el enfoque principal es el aprendizaje en capas sucesivas de representaciones significativas [9]. Técnicas de aprendizaje profundo utilizadas en redes neuronales han demostrado tener buenos resultados en la reducción de ruido y la eliminación de valores atípicos para las nubes de puntos en 3D. Para atacar este problema han surgido distintos algoritmos como el algoritmo PCPNet [6], el cual mediante la estimación de propiedades locales logra reducir el nivel de ruido en una nube de puntos. En la fig. 1 podemos observar el resultado cualitativo de PCPNet, contiene un error mínimo en la estimación de normales. Para el aprendizaje sugieren un método basado en parches, el enfoque principal es estimar propiedades de forma local para normales (orientadas y desorientadas) y la curvatura de nubes de puntos sin procesar con presencia alta de ruido [3].

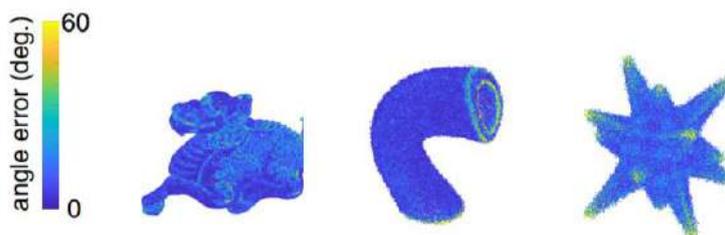


Fig. 1. Estimación de normales de PCPNet [6].

PointNet [7] es un algoritmo enfocado en la clasificación de objetos, segmentación de partes y análisis semántico de la escena. Una de las principales ventajas que tiene PointNet sobre otros algoritmos de clasificación y segmentación es que, al procesar una nube de puntos, dichos algoritmos requieren

Artículo aceptado y presentado en COMIA 2021

que las nubes de puntos sean regulares, por lo que estas tienen que ser sometidas a transformaciones. PointNet no necesita realizar transformaciones de este tipo, ya que cada punto se trata de forma única e independiente debido a que utiliza un método llamado “*Max Pooling*” como método de discretización [7].

PointCleanNet es un algoritmo dedicado a la reducción de ruido y valores atípicos. Este algoritmo está basado en la reducción de valores atípicos por radio. El radio es utilizado para la estimación de vecinos el cual ayuda a manejar nubes de puntos con alta densidad. La base de datos que utiliza para el entrenamiento del modelo contiene 28 figuras con distintas magnitudes de ruido y su correspondiente nube de puntos sin ruido, de esta forma es posible conocer qué puntos de la nube son valores atípicos. Para generar los valores atípicos de la nube se introdujo ruido gaussiano con una desviación estándar del 20% de acuerdo con la diagonal definida por los valores máximos y mínimos de cada nube [3].



Fig. 2. Nube de puntos de armadillo con distintas magnitudes de ruido [3].

## 2.- Materiales y métodos

En esta sección se describe, el material utilizado, la caracterización de ruido, la red neuronal y métricas de validación.

### 2.1- Material utilizado

Para la creación y preprocesamiento de las nubes de puntos se utilizaron las siguientes herramientas.

- Laptop Hp Pavilion 4 Gb RAM, 500 Gb HDD, GTX1650, Rizen 7.
- MeshLab versión 2020.12
- Pycharm versión 2020.1
- Anaconda Python 3.7 64-Bit
- Jupiter Notebook

---

Artículo aceptado y presentado en COMIA 2021

Los algoritmos para el preprocesamiento de la nube fueron desarrollados en Python 3.7 utilizando Jupiter Notebook, las nubes eran cargadas en la memoria de la GPU para el preprocesamiento, MeshLab fue utilizado como herramienta de visualización, una nube de puntos con una densidad de 140, 000 puntos al ser visualizada ocupaban 532 MB, por lo que las especificaciones de la Laptop HP Pavilion eran suficientes.

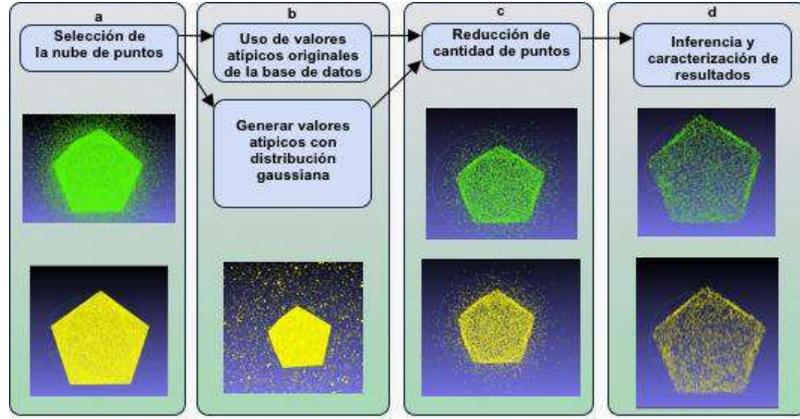
Para las inferencias el hardware y software utilizado fue el siguiente:

- Workstation NVIDIA GeForce RTX 3060 TI. 8GB GDDR6. 4864 CUDA Cores, AMD Ryzen 5600x, 6 Cores, 12 hilos, 3.7GHz 32MB L3 Cache 3MB L2 Cache, 16 GB RAM.
- Python 3.7 64-Bits
- Docker container Linux Engine Versión 19.03.8
- Ubuntu 20.04.2 LTS

Las inferencias fueron realizadas en una estación de trabajo con las especificaciones antes mencionadas, esto debido a que la alta densidad de las nubes de puntos requiere un alto procesamiento en GPU. Se utilizó una tarjeta RTX 3060, con sistema operativo Ubuntu, y contenedores Docker para la manipulación del ambiente de programación.

## **2.2 Caracterización de ruido**

La caracterización de ruido consta de 4 etapas principales como se muestra en la fig. 3., la selección de nubes de puntos a), generación de valores atípicos b), reducción de la densidad de la nube e inferencia c) y caracterización de resultados d).



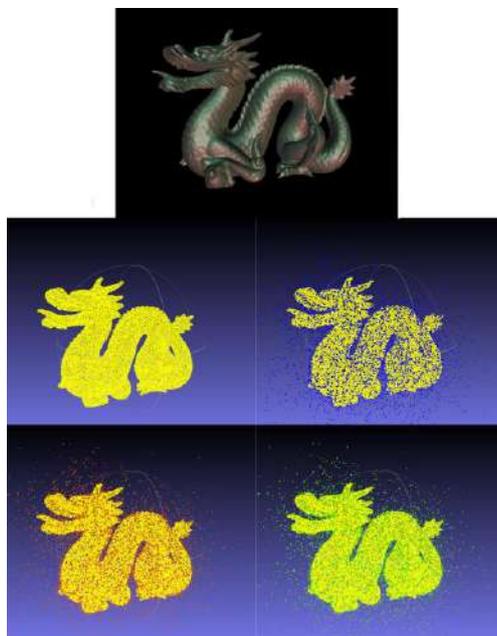
**Fig. 3.** Metodología propuesta para la caracterización del ruido en distintas magnitudes.

La caracterización del ruido se realizó utilizando la metodología propuesta en la fig. 3. Se utilizaron nubes de puntos de la base de datos de PointCleanNet para reducción de valores atípicos (a). De la base de datos se generaron 288 nubes, que corresponden a 4 figuras, de las cuales se obtuvo una nube de puntos con ruido gaussiano con una desviación estándar de 1%, 2.5% y 5% b). El modelo que describe la nube de puntos y el ruido está descrito por la ecuación (2).  $\mathbf{P}'$  representa la nube de puntos con ruido,  $\mathbf{P}$  son el conjunto de puntos que representan la superficie del objeto,  $n_i$  es el ruido aditivo,  $\mathbf{O}$  es el conjunto de valores atípicos presentes en la nube [3].

$$\mathbf{P}' = \{\mathbf{p}'_i\} = \{\mathbf{p}_i + \mathbf{n}_i\}_{\mathbf{p}_i \in \mathbf{P}} \cup \{\mathbf{o}_j\}_{\mathbf{o}_j \in \mathbf{O}} \quad (2)$$

Se generó un archivo en el que se clasifican cada uno de los puntos. Además, la densidad de las nubes varía con intervalos de 10% en un rango de 10% a 90% c).

Artículo aceptado y presentado en COMIA 2021



**Fig. 4.** Nube de puntos de dragón con distintas magnitudes de ruido con una desviación estándar de 1%, 2.5% y 5% y distinta densidad.

Posterior a esto se realizaron 144 inferencias d), con el modelo propuesto por PointCleanNet [3], utilizando como entrada las nubes con distintas magnitudes de ruido y distinta densidad. El objetivo de la inferencia es obtener el rendimiento de la red neuronal con distintos niveles de ruido y densidad para determinar el punto de inflección con el que la remoción de puntos no afecte el rendimiento y a la vez contribuya a reducir el tiempo de inferencia.

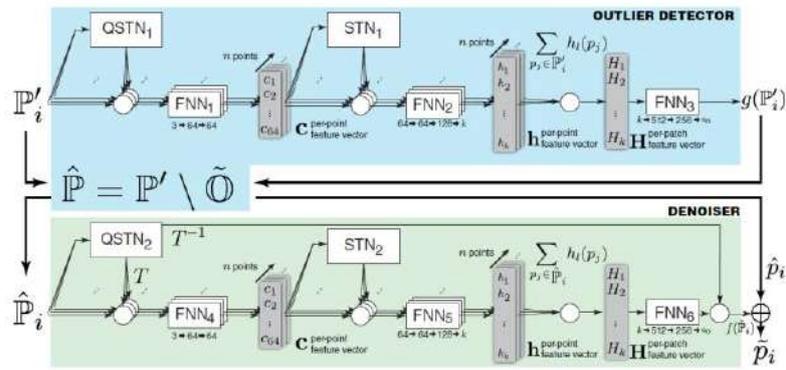
### 2.3 Red neuronal

La arquitectura de la red neuronal convolucional utilizada es la propuesta por PointCleanNet [3], tiene dos etapas principales, una etapa dedicada a reducir el número de valores atípicos y otra a reducción de ruido. Para la reducción de valores atípicos la red neuronal está basada en las arquitecturas de PCPNet [6] y PointNet [7], debido a que utilizan Perceptrones de múltiples capas para la extracción de características significativas, además utilizan una metodología para saltar conexiones con el objetivo de promover la propagación del gradiente y mejorar el entrenamiento [3]. Esta arquitectura de red, además utiliza reducción de valores atípicos por radio, esta metodología es utilizada para estimar que tan probable es el punto central de ser un valor atípico.

El modelo de la nube de puntos es descrito por la ecuación (1), la nube de puntos esta compuesta por los puntos que describen la superficie del objeto, y que además cuenta con presencia de ruido y valores atípicos. Para el entrenamiento se utiliza una función no lineal que es utilizada para remover los valores atípicos [3].

$$\tilde{O}_i = g(P'_i) > 0.5 \quad (3)$$

La ecuación (3) representa la probabilidad de valor atípico  $\tilde{O}_i$ , establece que un punto es añadido al conjunto de valores atípicos si la probabilidad de valor atípico es mayor al 0.5 [3].



**Fig. 5.** Diagrama de red convolucional de PontCleanNet[3], la red recibe como entrada una nube de puntos con presencia de ruido  $P_i$ , ésta es procesada por una red neuronal utilizada para la clasificación de valores atípicos y obtener una nube de puntos con reducción de outliers y el conjunto de outliers removidos  $\widehat{P}_i/\tilde{O}$  [3].

## 2.4 Criterios de evaluación

Cuando intentamos reducir el nivel de valores atípicos en una nube de puntos debemos de considerar que los puntos que se están removiendo con el modelo propuesto correspondan a los valores atípicos (True Positive), que la cantidad de puntos que se remuevan erróneamente sean cuantificados (False Positive). Estas dos métricas contribuyen a la Precisión (ecuación 4), Recall (ecuación 5), F1-Score (ecuación 6) y el Error Cuadrático Medio (ecuación 7) del modelo [3].

$$Precisión = \frac{\sum true\ positive}{\sum Predicted\ Positive\ Condition} \quad (4)$$

Artículo aceptado y presentado en COMIA 2021

$$True\ Positive\ Rate, Recall = \frac{\sum true\ positive}{\sum Positive\ Condition} \quad (5)$$

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (6)$$

$$MSE = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2 \quad (7)$$

### 3. Resultados

En esta sección se describen los resultados cuantitativos y cualitativos obtenidos después de realizar las inferencias para la etapa de reducción de valores atípicos utilizando el modelo propuesto por PointCleanNet.

#### 3.1 Resultados cuantitativos

La tabla 1 muestra los resultados de las inferencias de las nubes de puntos con presencia de valores atípicos realizadas con el modelo propuesto por PointCleanNet [3], cada columna de la tabla representa las métricas de evaluación así como el tiempo de ejecución en GPU.

**Tabla 1.** Comparación de resultados cualitativos.

| Densidad de nube pts. | Precisión % | Recall % | F1-Score % | MSE % | GPU min |
|-----------------------|-------------|----------|------------|-------|---------|
| 14, 000               | 29.78       | 100      | 45.89      | 72.71 | 8.58    |
| 28, 000               | 29.23       | 99.75    | 45.23      | 68.50 | 17.14   |
| 42, 000               | 32.84       | 99.26    | 49.39      | 58.14 | 25.71   |
| 56, 000               | 39.64       | 96.72    | 56.57      | 42.41 | 34.24   |
| 70, 000               | 50.19       | 94.68    | 66.40      | 27.92 | 42.78   |
| 84, 000               | 63.70       | 92.13    | 77.05      | 16.31 | 51.31   |
| 98, 000               | 77.93       | 87.43    | 86.33      | 08.59 | 59.85   |
| 112, 000              | 87.32       | 84.08    | 91.48      | 05.09 | 68.57   |
| 126, 000              | 95.38       | 80.75    | 95.03      | 02.84 | 77.14   |
| 140,000               | 99.61       | 90.78    | 94.99      | 02.75 | 85.71   |

## Artículo aceptado y presentado en COMIA 2021

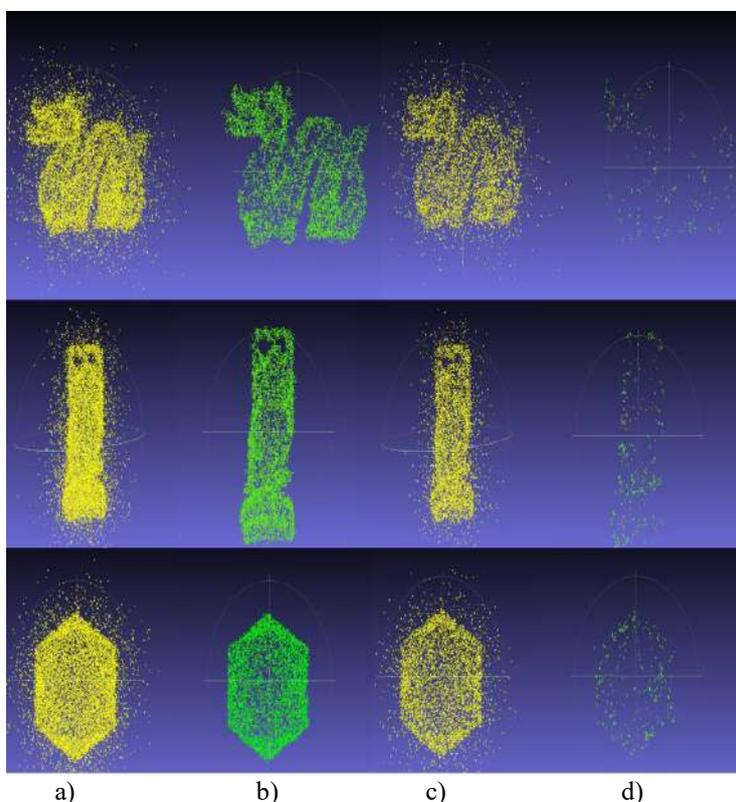
Se realizaron inferencias para 144 nubes de puntos con distintas magnitudes de ruido con una desviación estándar de 1%, 2.5% y 5% y distinta densidad como se muestra en la tabla 1. La densidad de la nube está en función del porcentaje de puntos que se removieron, siendo de menor densidad aquellas nubes a las que se removió mayor porcentaje de puntos. Las inferencias se realizaron utilizando la estación de trabajo definida en la sección de material utilizado.

Se puede observar en la tabla 1. que para una densidad de 112, 000 puntos se obtiene una Precisión del 87.32% (ecuación 4), Recall del 84.08 % (ecuación 5), F1-Score del 91.48% (ecuación 6), MSE del 05.09% (ecuación 7), y tiempo de ejecución en GPU de 68 minutos. Para esta densidad de puntos el error cuadrático indica que el rendimiento del modelo no se degrada con la reducción de la densidad de puntos. Por otra parte, los resultados obtenidos en nubes de puntos con una densidad de 84, 000 puntos indican una degradación en la clasificación de los valores atípicos, en comparación con los resultados obtenidos con una densidad de 112,000 puntos.

### 3.2 Resultados cualitativos

La siguiente imagen muestra tres figuras distintas, las figuras cuentan con distinto nivel de valores atípicos y distinta densidad. Cabe recalcar que las inferencias corresponden al análisis hecho para 126,000 y 84,000 puntos respectivamente, el propósito principal es mostrar el punto en el que el rendimiento del modelo conlleva a una degradación en la clasificación de valores atípicos. Nubes de puntos con una densidad de 84,000 puntos o menor, tienen a degradar el rendimiento del modelo. Por otro lado el tiempo de inferencia disminuye cuando la densidad es mayor a 112,000 puntos, dónde el rendimiento de la red no se ve perjudicado obteniendo F1-Score de 91.48%.

Artículo aceptado y presentado en COMIA 2021



**Fig. 6.** Comparativa de resultados obtenidos para nubes de puntos con presencia de valores atípicos. De izquierda a derecha y de arriba abajo: a) Nubes con 126,000 puntos y valores atípicos. b) Inferencia utilizando el modelo propuesto por PointCleanNet para las nubes de 126, 000 puntos. c) Nubes con 84,000 puntos y valores atípicos., d) Inferencia utilizando el modelo propuesto por PointCleanNet para las nubes de 84, 000 puntos.

#### 4. Conclusiones y trabajo futuro

La inferencia para el procesamiento de una nube con 140,000 puntos toma en promedio 85.71 minutos. Para una nube con 112,000 puntos toma en promedio 68.57 minutos. Reducir el tamaño de una nube de puntos en un 20% contribuye a la reducción del tiempo de inferencia en un 24.99%. El f1-score muestra que al remover el 20% de los puntos se obtiene 91.48%, es aquí cuando la reducción del tiempo de ejecución alcanza su óptimo, ya que se preserva la clasificación de los valores atípicos. Por el contrario, analizando los resultados de la nube con una densidad de 84,000 puntos, esta toma en promedio 51.31 minutos, pero la reducción del rendimiento para la clasificación de nubes de puntos es evidente con un 77.05% de f1-score.

## Artículo aceptado y presentado en COMIA 2021

Las nubes de puntos procesadas permitieron analizar los resultados de las inferencias para verificar bajo qué circunstancias la red neuronal tiene resultados aceptables, es decir cuando la degradación de las características de la figura en si se ven perjudicadas, esto para determinar hasta qué punto es recomendable reducir la densidad de la nube, además permite determinar cuando la densidad de la nube afecta de manera significativa el rendimiento de la red. Al hacer la inferencia en la nube de puntos podemos observar que uno de los factores que afectan a la reducción de ruido y valores atípicos es la forma o la figura de las nubes de puntos, así como el nivel de ruido presente. Nubes de puntos con formas complejas se ven afectadas dando como resultado a la pérdida de características en la nube.

Como trabajo futuro, se tiene considerado continuar con el desarrollo e implementación de una red neuronal convolucional que contribuya a mejorar el rendimiento de redes neuronales para la reducción de ruido y valores atípicos, así como mejorar el tiempo de ejecución de nubes de puntos densas. Continuar con la implementación de pruebas para medir el desempeño de la red y determinar áreas de mejora.

## 5. Referencias

- 1.- Sotoodeh, S. (2006). Outlier detection in laser scanner point clouds. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(5), 297-302.
- 2.- Rakotosaona, M. J., La Barbera, V., Guerrero, P., Mitra, N. J., & Ovsjanikov, M. (s. f.). Index of /projects/2019/pointcleannet/data. Recuperado 6 de enero de 2021, de <http://geometry.cs.ucl.ac.uk/projects/2019/pointcleannet/data/>
- 3.- Rakotosaona, M. J., La Barbera, V., Guerrero, P., Mitra, N. J., & Ovsjanikov, M. (2020, February). PointCleanNet: Learning to denoise and remove outliers from dense point clouds. In *Computer Graphics Forum* (Vol. 39, No. 1, pp. 185-203).
- 4.- Javaheri, A., Brites, C., Pereira, F., & Ascenso, J. (2017, July). Subjective and objective quality evaluation of 3D point cloud denoising algorithms. In *2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)* (pp. 1-6). IEEE.

---

Artículo aceptado y presentado en COMIA 2021

5.- Sankaranarayanan, J., Samet, H., & Varshney, A. (2007). A fast all nearest neighbor algorithm for applications involving large point-clouds. *Computers & Graphics*, 31(2), 157-174.

6.- Guerrero, P., Kleiman, Y., Ovsjanikov, M., & Mitra, N. J. (2018, May). PCPNet learning local shape properties from raw point clouds. In *Computer Graphics Forum* (Vol. 37, No. 2, pp. 75-85).

7.- Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 652-660).

8.- Javaheri, A., Brites, C., Pereira, F., & Ascenso, J. (2017, July). Subjective and objective quality evaluation of 3D point cloud denoising algorithms. In *2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)* (pp. 1-6). IEEE.

9.- Chollet, F. (2017). *Deep Learning with Python: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek*. MITP-Verlags GmbH & Co. KG.

10.- Y. Schoenenberger et al., "Graph-based Denoising for Time-Varying Point Clouds," *3DTV Conference: The True Vision-Capture, Transmission and Display of 3D Video*, Lisbon, Portugal, Jul. 2015.

11.- R. B. Rusu et al., "Towards 3D Point Cloud Based Object Maps for Household Environments," *Robotics and Autonomous Systems*, vol. S6, no. II, pp. 927-941, Nov. 2008.

A.0.2. Artículo aceptado y presentado en CONIIN 2021



**CONIIN**  
XVII INTERNATIONAL ENGINEERING  
CONGRESS

THE QUERÉTARO STATE UNIVERSITY THROUGH THE ENGINEERING FACULTY GRANT THE PRESENT ACKNOWLEDGMENT TO:

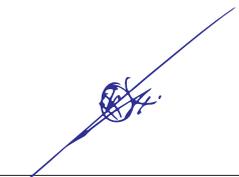
**Luis Rogelio Román Rivera, Israel Sotelo Rodríguez, Gerardo Treviño Valdés, Mayra Azucena Cíntora and Jesús Carlos Pedraza Ortega**

**General Conference:**

Detection of a sphere with a known size in a 3D cloud point using RANSAC

QUERÉTARO, MEX.  
JUNE 2021

  
Dr. Manuel Toledano Ayala  
PRINCIPAL  
ENGINEERING FACULTY

  
Dr. Gonzalo Macías Bobadilla  
GENERAL COORDINATOR CONIIN  
ENGINEERING FACULTY



### A.0.3. Convalidación examen inglés comprensión textos




**UNIVERSIDAD AUTÓNOMA DE QUERÉTARO**  
**FACULTAD DE LENGUAS Y LETRAS**



**A QUIEN CORRESPONDA:**

La que suscribe, Directora de la Facultad de Letras y Letras, hace **C O N S T A R** que

**SOTELO RODRIGUEZ ISRAEL**

Presentó y acreditó el **Examen de Comprensión de Textos en Inglés** efectuado el día dieciocho de octubre de dos mil veintiuno.

Se extiende la presente a petición de la parte interesada, para los fines escolares y legales que le convengan, en el Campus Aeropuerto de la Universidad Autónoma de Querétaro, el día veintinueve de noviembre de dos mil veintiuno.

Atentamente,  
"Enlazar Culturas por la Palabra"





**DRA. ADELINA VELÁZQUEZ HERRERA**

**AVH/japa\*CL\*FLL-C.-2334**

---

**SOMOS UAQ**  
EDUCAR CRECER CONSOLIDAR

Campus Aeropuerto, Anillo Vial Fray Junipero Serra S/N, Querétaro, Qro, C.P. 76140  
Tel. 442 192 12 00 Dirección Ext. 61010, Secretaría Administrativa Ext.61300, Posgrado Ext. 61140,  
Licenciatura Ext.61070, Centro de Idiomas Ext.61050, Secretaría Académica Ext.61100 y Planeación Ext.61110

**Figura A.1:** Convalidación examen inglés comprensión textos

#### A.0.4. Convalidación examen inglés manejo de lengua




UNIVERSIDAD AUTÓNOMA DE QUERÉTARO  
FACULTAD DE LENGUAS Y LETRAS



**A QUIEN CORRESPONDA:**

La que suscribe, Directora de la Facultad de Letras y Letras, hace **C O N S T A R** que

**SOTELO RODRIGUEZ ISRAEL**

Presentó el **Examen de Manejo de la Lengua** efectuado el día veintinueve de junio de dos mil veinte, en el cual obtuvo la siguiente calificación:

**8-**

Se extiende la presente a petición de la parte interesada, para los fines escolares y legales que le convengan, en el Campus Aeropuerto de la Universidad Autónoma de Querétaro, el día once de agosto de dos mil veintiuno.



Atentamente,  
"Enlazar Culturas por la Palabra"



**DRA. ADELINA VELÁZQUEZ HERRERA**

**AVH/japa\*CL\*FLL-C.-1599**

---

**SOMOS UAQ**  
EDUCAR CRECER CONSOLIDAR

Campus Aeropuerto, Anillo Vial Fray Junipero Serra S/N, Querétaro, Qro, C.P. 76140  
Tel. 442 192 12 00 Dirección Ext. 61010, Secretaría Administrativa Ext.61300, Posgrado Ext. 61140,  
Licenciatura Ext.61070, Centro de Idiomas Ext.61050, Secretaría Académica Ext.61100 y Planeación Ext.61110

Figura A.2: Convalidación examen inglés manejo de lengua

## A.0.5. Protocolo aprobado por consejo.

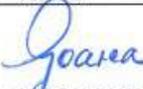
|  |  |  |  |
|--|--|--|--|
|   |  | UNIVERSIDAD AUTÓNOMA DE QUERÉTARO<br><b>FACULTAD DE INGENIERÍA</b><br>DIVISIÓN DE INVESTIGACIÓN Y POSGRADO   |  |
| <b>REGISTRO DEL PROTOCOLO DE INVESTIGACIÓN DEL ESTUDIANTE DE POSGRADO</b>  |  |  |  |
| Los 2 Espacios oscuros exclusivos para la Dirección  | No. Registro de Proyecto*:   | 11780  |  |
|  | Fecha de Registro*:  | 13/ENERO/2020  |  |
|  | Fecha de inicio de proyecto:   | 01/2020  |  |
|  | Fecha de término de proyecto:  | 07/2022  |  |
| <b>1. DATOS DEL SOLICITANTE</b>  |  |  |  |
| No. de expediente:   | 293055   |  |  |
| Apellido Paterno   | Apellido Materno   | Nombre(s)  |  |
| SOTELO   | RODRIGUEZ  | ISRAEL UNIVERSIDAD AUTÓNOMA DE QUERÉTARO   |  |
| Dirección:   |  |  |  |
| Calle y número   | Colonia  | C.P.   |  |
| Fray Junipero serra #76269 Int-165   | Paseo San Junipero   | 78146  |  |
| Estado   | Teléfono (Incluir lada)  | Correo Electrónico   |  |
| 22   | 3317405006   | kei22@uaq.mx   |  |
| <b>2. DATOS DEL PROYECTO</b>   |  |  |  |
| Facultad:  | INGENIERÍA   |  |  |
| Programa:  | MAESTRIA EN CIENCIAS EN INTELIGENCIA ARTIFICIAL  |  |  |
| Tema específico del proyecto:  | Reducción de valores atípicos y presencia de ruido en nube de puntos en 3D utilizando técnicas de aprendizaje profundo.              |  |  |
| <br>ROMAN RIVERA LUIS ROGELIO<br>Director de tesis  | <br>TOVAR ARRIAGA SAUL<br>Coordinador de programa | <br>SOTELO RODRIGUEZ ISRAEL<br>Alumno   |  |
| <br><b>Dr. Juan Carlos Jáuregui Correa</b><br>Jefe de División de Investigación y Posgrado de la Fac. de Ing. | <b>Dr. Manuel Toledano Ayala</b><br>Director de Fac. Ing.  | <br><b>Dra. Ma. Guadalupe Flavia Loarca Piña</b><br>Directora de Investigación y Posgrado UAQ |  |
| Anexo 3 Manual de procedimientos Administrativos de Posgrado de la Facultad de Ingeniería  |  |  |  |

Figura A.3: Protocolo aprobado por consejo.