Detección de objetos 3D con PointNet para la conducción autónoma

Juan Pablo González Mendoza
Depto. Ingeniería Electrónica
Universidad de Guanajuato
Salamanca, Gto. México
Email: jp.gonzalezmendoza@ugto.mx

Felipe Trujillo-Romero
Depto. Ingeniería Electrónica
Universidad de Guanajuato
Salamanca, Gto. México
Email: fdj.trujillo@ugto.mx

Juan José Cárdenas Cornejo
Depto. Ingeniería Electrónica
Universidad de Guanajuato
Salamanca, Gto. México
Email: jj.cardenascornejo@ugto.mx

***Resumen*—La creciente popularidad de los vehículos autónomos subraya la necesidad de una detección precisa de obstáculos en su entorno. Este artículo presenta un análisis de detección de objetos 3D utilizando nubes de puntos con diversas densidades: 128, 256, 512 y de 1024 puntos. El modelo fue entrenado con el conjunto de datos KITTI, enfocándose en la extracción de ciclistas, autos y peatones, y se evaluaron los últimos dos en dos bases de datos adicionales. Se implementaron técnicas de regularización y se optimizaron los algoritmos de entrenamiento utilizando la arquitectura PointNet. Los resultados indican que el modelo alcanzó un rendimiento óptimo con más de 1024 puntos en modelos CAD, logrando un 84 % de rendimiento y un 91 % en el conjunto de datos de Sydney con menor densidad de puntos.**

Palabras clave: PointNet, Kitti, Sydney, Modelos CAD, Nubes de puntos, Procesamiento de datos 3D, Velodyne

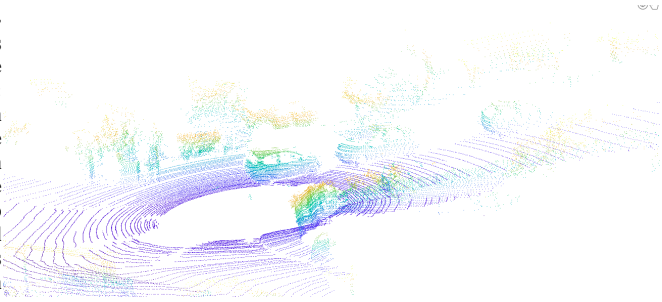

Figura 1: Nube de puntos generada por el sensor HDL64-E, dataset KITTI

I. Introducción

Dentro de los próximos diez años se prevé que los vehículos autónomos tengan el potencial de transformar significativamente el transporte y movilidad urbana [1]. Para lograr una conducción autónoma segura y eficiente, es primordial que dichos vehículos tengan la capacidad de comprender su entorno de manera precisa. Por ello la importancia de utilizar sensores para capturar información tridimensional (3D) del entorno y procesarla a alta velocidad tendiendo a tiempo real. Entre las tecnologías más importantes utilizadas en este campo son las nubes de puntos usualmente proporcionada por sensores LiDAR (Light Detection and Ranging) y cámaras de visión.

Las nubes de puntos como se observa en la figura 1, son una representación discreta de los objetos y superficies en el espacio tridimensional, donde cada punto contiene información sobre coordenadas espaciales y en algunos casos datos adicionales como la intensidad de la señal reflejada. Estas nubes de puntos permiten a los vehículos autónomos a detectar obstáculos, identificar objetos y mapear el entorno con precisión. En algunos casos la comparación con otro tipo de sensores tales como cámaras de visión y radares, la tecnología LiDAR ofrecen ventajas únicas, como la independencia de condición de iluminación y capacidad de medir distancias. Sin embargo este tipo de datos presenta desafíos significativos. El volumen masivo de datos generado por esta clase de sensores requiere de métodos de procesamiento eficientes para respuestas en tiempo real. Además, la clasificación y segmentación de objetos en entornos complejos, como zonas urbanas densas, sigue siendo un área activa de investigación.

A pesar de estos retos, los avances recientes en algoritmos de aprendizaje automático y visión por computadora han permitido importantes mejoras en la detección y clasificación basada en nubes de puntos, incrementando la robustez y fiabilidad de los sistemas de percepción de vehículos autónomos. En este trabajo, se exploran técnicas para la detección y clasificación de objetos utilizando nubes de puntos con diferentes densidades en el contexto de la conducción autónoma. Se presentan metodologías basadas en redes neuronales profundas que abordan los desafíos de procesar datos tridimensionales, destacando tanto los beneficios como las limitaciones de las soluciones actuales. Finalmente, se proponen mejoras para la optimización del procesamiento de este tipo de datos y la integración de nubes de puntos con otros sensores, con el objetivo de mejorar la percepción y la toma de decisiones en vehículos autónomos.

El sensor HDL-64E utiliza 64 haces láser infrarrojos distribuidos uniformemente para escanear el entorno en 360 grados. Cada haz láser emite pulsos de luz que impactan en los objetos del entorno, y al medir el tiempo que tarda el pulso en viajar hasta el objeto y regresar al sensor (tiempo

de vuelo), se calcula con gran precisión la distancia a dicho objeto. Este proceso se repite de manera simultánea para todos los haces, lo que permite capturar múltiples puntos en cada ciclo de escaneo. Al combinar las distancias medidas con los ángulos de cada haz, el sensor construye un mapa tridimensional detallado del entorno. Estos datos se representan en forma de una nube de puntos tridimensional, que es fundamental para la percepción y navegación [15].

Un estudio por parte de B. Padmaja, *et al.* [14], menciona problemáticas clave en el uso de este tipo de sensores. En primer lugar, uno de los principales desafíos es su alto costo, que afecta la escalabilidad y accesibilidad de esta tecnología en el mercado de vehículos autónomos. Además, los sensores LiDAR requieren procesamiento intensivo de datos debido a la gran cantidad de información que generan, lo que aumenta la necesidad de hardware y algoritmos eficientes para manejar esta carga de datos.

En la última década, los vehículos autónomos han pasado de ser una idea futurista a una realidad concreta, con tecnologías avanzadas que prometen transformar el transporte y afectar a la sociedad. Sin embargo, su adopción enfrenta desafíos técnicos, éticos, legales y sociales que requieren un enfoque integral.

- **Impacto Potencial:**
  Los vehículos autónomos pueden transformar la movilidad, mejorar la seguridad vial, reducir la congestión y las emisiones. Investigar estos avances aportará información clave para tomadores de decisiones en transporte y gobierno.
- **Desafío técnico:**
  La integración de vehículos autónomos enfrenta retos técnicos, incluyendo algoritmos de percepción, comunicación V2V y V2I, y ciberseguridad.
- **Impacto Social:**
  La adopción de vehículos autónomos impactará áreas como el empleo, la accesibilidad a diferentes usuarios con capacidades reducidas y la organización urbana. Investigar estos efectos ayudará a anticipar desafíos y optimizar sus beneficios para la sociedad.

Estudiar y abordar los retos de los vehículos autónomos es esencial para maximizar su capacidad transformadora y asegurar una implementación ética y justa. Al abordar estos temas desde una perspectiva interdisciplinaria, este artículo contribuirá al progreso del conocimiento en este ámbito emergente y proporcionará orientación para el desarrollo futuro de los vehículos autónomos.

II. Antecedentes

La organización SAE clasifica los vehículos autónomos en seis niveles, lo que permite entender mejor las capacidades del vehículo. Varias empresas tecnológicas involucradas en la conducción autónoma como: Waymo, Cruise de GM, Aurora de Toyota y Tesla, etc. Han apostado al uso exclusivo de cámaras de visión u otras por su parte a utilizar tecnología láser y radares. La tecnología exclusiva por alguno de estos sensores presenta fallas en su detección. En 2018, un Tesla Model X, presentó un error en el Autopilot (asistente de conducción) al estrellarse contra una barrera en Mountain View, California, terminando en el deceso del piloto [2]. La investigación determinó que el sistema de conducción tuvo un error al detectar la barrera y el conductor no tomó el control a tiempo. Por otra parte un vehículo autónomo de la empresa Uber atropelló y mató a una peatona en Arizona [3]. Donde se determinó en la investigación que el sistema no logró detectar correctamente el obstáculo, fallando en realizar maniobras evasivas. Este accidente fue un caso icónico que resaltó las deficiencias en la percepción y toma de decisiones en sistemas autónomos creando nuevas áreas de investigación. Un incidente reciente involucró un Ford Mustang Mach-E equipado con el sistema semi-autónomo BlueCruise. El sistema falló en detectar un vehículo detenido en la carretera, lo que resultó en una colisión fatal. Aunque el sistema permite la conducción automatizada, se espera que el conductor mantenga atención, lo que plantea preocupaciones sobre la capacidad de estos sistemas para reaccionar ante imprevistos [4]. Tesla ha sido objeto de múltiples investigaciones debido a accidentes fatales que involucran su sistema Autopilot, diseñado para manejar parcialmente el vehículo en ciertas condiciones. En varios incidentes, los sensores del sistema fallaron en detectar obstáculos en la carretera, como vehículos detenidos o peatones. Desde 2019, al menos 17 accidentes fatales han sido relacionados con fallos en la detección de su sistema [5].

Algunas empresas se inclinan por el uso exclusivo de cámaras de visión, otras por su parte consideran que el uso de láser o radares pueden aumentar el nivel de detección, aunque conlleve un procesamiento más exhaustivo. En la tabla I se describen algunas ventajas contra desventajas en el uso de este tipo de sensores.

Tabla I: Comparativa cámaras de visión vs sensor LiDAR

Cámaras de visión		**Sensor LiDAR**	
Ventajas	**Desventajas**	**Ventajas**	**Desventajas**
Baratas	Dependencia de la iluminación	Precisión en la detección de distancias	Coste elevado
Alta resolución	Perspectiva limitada	Independencia de la iluminación	Complejidad y tamaño
Compatibilidad con IA	Procesamiento intensivo de datos	Detección de objetos pequeños	Procesamiento intensivo de datos
Reconocimiento de color y texto	—	—	—

Varios estudios en segmentación y clasificación de nubes de puntos han demostrado que los avances en los sensores LiDAR proporcionan datos tridimensionales de alta calidad, lo que mejora significativamente la comprensión de las escenas en 3D. Sin embargo, factores como la oclusión, las variaciones en la densidad de los puntos y la pérdida de señal hacen que

las nubes de puntos generadas por LiDAR sean, en la práctica, parciales, ya que solo capturan fragmentos de los objetos. Esto representa un desafío considerable para la percepción tridimensional. Para abordar esta problemática, se ha desarrollado un modelo de detección de objetos tridimensional basado en LiDAR, conocido como Behind the Curtain Detector (BtcDet), que aprende las formas típicas de los objetos y estima las formas completas de aquellos que están parcialmente ocluidos en las nubes de puntos [7].

Por otro lado, el modelo "Frustum PointNets for 3D Object Detection from RGB-D Data"propone un enfoque innovador para la detección de objetos en 3D, combinando información de imágenes RGB con datos de profundidad. Este método segmenta las imágenes en frustum (volúmenes tridimensionales) que corresponden a los objetos detectados en la imagen 2D [10]. Además, el artículo "PointAugment: An Auto-Augmentation Framework for Point Cloud Classification"presenta un marco de auto-aumento que mejora la clasificación de nubes de puntos. Este enfoque automatiza la generación de variaciones en los datos, aplicando transformaciones como rotaciones, escalados y perturbaciones a las nubes de puntos originales. Al hacerlo, amplía el conjunto de datos disponible para el entrenamiento de modelos de clasificación, lo que contribuye a mejorar su rendimiento y robustez. PointAugment se centra en identificar las mejores transformaciones a aplicar, optimizando el proceso de aumento y logrando resultados más precisos en la clasificación de objetos en 3D [11].

Finalmente, el trabajo que presenta PAttFormer introduce una arquitectura eficiente de aprendizaje multitarea que mejora la segmentación semántica y la detección de objetos en nubes de puntos, sin necesidad de codificadores de funciones separados. Esto resulta en una red tres veces más pequeña y 1.4 veces más rápida que las arquitecturas actuales, alcanzando un rendimiento competitivo en los benchmarks nuScenes y KITTI. Las evaluaciones muestran mejoras del 1.7 por ciento en mIoU para la segmentación semántica y en la detección de objetos 3D, evidenciando así la efectividad del enfoque multitarea [12].

III. Metodología

Este trabajo se enfoca en el análisis del conjunto de datos KITTI, empleando técnicas de filtrado y clasificación basadas en etiquetas. Se ha llevado a cabo un proceso de transformación para convertir una matriz de dimensión $N \times 7$ a $N \times 3$. Este filtrado se realizó ajustando los parámetros para permitir la inclusión de puntos con diversas densidades, realizando pruebas con un número mínimo de objetos hasta alcanzar un máximo de 2048 en algunas clases. El conjunto de datos estudiado incluye etiquetas, calibraciones de cámaras en relación al sensor Velodyne, así como los archivos de nube de puntos generados por dicho sensor. Tras revisar la documentación, se concluye que el conjunto de datos es sólido, aunque presenta variaciones en la cantidad de información disponible para cada clase lo cuál es un reto poder generar datos homogéneos para cada clase. En la figura 2 se muestra el diagrama general implementado.

III-A. Base de datos Kitti

Para iniciar el proceso de extracción de características a partir de los archivos Velodyne, se comienza con el procesamiento de los datos de la nube de puntos generada por el sensor Velodyne HDL64-E. Esta información se obtiene de archivos binarios que contienen coordenadas espaciales (x, y, z) y valores de intensidad. Las etiquetas asociadas a los objetos detectados se extraen de archivos de texto, lo que facilita la identificación de la ubicación y la clase de cada objeto a través de cuadros delimitadores. La asignación de etiquetas a los puntos se lleva a cabo en función de su posición relativa a estos límites, utilizando matrices de proyección, un paso crucial para asegurar la correcta alineación de los datos y, en consecuencia, mejorar la precisión del análisis. A pesar de que el conjunto de datos KITTI ofrece una amplia variedad de escenarios de conducción, es fundamental considerar las variaciones en la cantidad de datos disponibles por clase, ya que esto puede afectar la capacidad de generalización de los modelos. Por lo tanto, se realizaron pruebas con diversas configuraciones para evaluar el rendimiento y la eficacia del enfoque implementado. En la etapa de extracción de objetos por clase en la tabla II se muestran los archivos extraídos para cada clase y la cantidad de puntos.

En este caso de estudio se realiza la extracción de tres clases en específico:

- Car
- Pedestrian
- Cyclist

Tabla II: Datos para entrenamiento

Clase	**Extracción de archivos KITTI**		
	Np>=128	Np>= 256	Np>= 512
Car	4717	2793	1302
Pedestrian	912	563	106
Cyclist	266	212	49

III-B. Sydney Urban Dataset

Este conjunto de datos abarca una amplia gama de objetos viales urbanos comunes, escaneados con un LIDAR Velodyne HDL64-E en el centro de Sídney, Australia. Incluye un total de 631 escaneos individuales que representan diversas categorías de vehículos, peatones, señales de tráfico y árboles. La recopilación de estos datos tiene como propósito evaluar algoritmos de coincidencia y clasificación, proporcionando condiciones de detección subóptimas que reflejan los desafíos presentes en sistemas de detección urbana reales, caracterizados por una notable variabilidad en los ángulos de visión y oclusiones. Además, junto con este conjunto de datos, se modelan automóviles y peatones con distintas

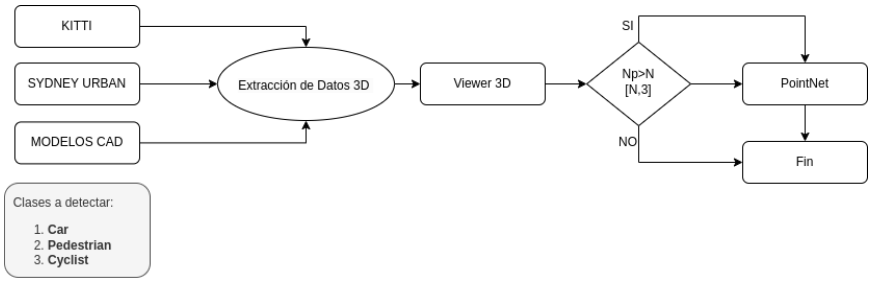

Figura 2: Metodología implementada

Tabla III: Validación Dataset Sydney

Clase	Datos para validación		
	Np>=128	Np>= 256	Np>= 512
Car	88	N/A	N/A
Pedestrian	152	N/A	N/A
Cyclist	2	N/A	N/A

densidades de nubes de puntos [8]. Este dataset se utiliza para hacer un test al modelo entrenado con Kitti, comparando con sus diferentes niveles de puntos para cada clase.

III-C. ModelNet 40

El uso de modelos CAD para entrenar PointNet y otras redes neuronales basadas en nubes de puntos es una estrategia eficaz para generar grandes volúmenes de datos anotados y precisos. No obstante, es fundamental tener en cuenta las diferencias entre los datos sintéticos y los datos reales. Para garantizar que el modelo sea robusto y capaz de generalizar adecuadamente en aplicaciones prácticas, es esencial complementar el entrenamiento con datos del mundo real. Esta combinación permite que el modelo aprenda a manejar las variabilidades y complejidades presentes en entornos del mundo real, mejorando su rendimiento y eficacia en situaciones de uso cotidiano. [9].

Tabla IV: Validación Dataset ModelNet40

Clase	Datos para validación		
	Np>=128	Np>= 256	Np>= 512
Car	299	N/A	N/A
Pedestrian	110	N/A	N/A
Cyclist	N/A	N/A	N/A

Al asignar puntos aleatorios dentro del volumen del sólido, podemos determinar una cantidad específica de puntos que se utilizarán para el análisis. En la Tabla IV, se establece un rango de puntos que va desde el mínimo requerido de 128 hasta un máximo de 2048. Esto significa que podemos realizar experimentos variando la cantidad de puntos, asegurando así que el modelo se entrene y evalúe de con diferentes densidades de datos.

III-D. CloudCompare

Inicialmente, la herramienta fue diseñada para permitir comparaciones directas entre nubes de puntos 3D densas. Para verificar que los datos extraídos del conjunto de datos KITTI y el procesamiento de filtrado se realizan correctamente, es fundamental contar con una visualización del objeto. Esta visualización facilita la validación de los experimentos, asegurando que se pueden evaluar de manera adecuada las nubes de puntos con diferentes densidades.

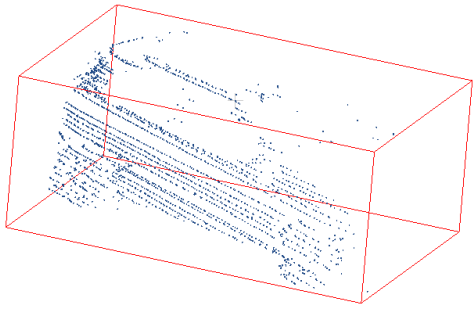

Figura 3: Visualización de datos

En la figura 3 se muestra visualmente le extracción correcta del objeto perteneciente a clase Auto.

III-E. PointNet

En esta investigación se emplea PointNet, una arquitectura seleccionada por sus ventajas particulares para el procesamiento de nubes de puntos. Debido a la variabilidad en las densidades de puntos en las bases de datos empleadas, es común que surjan desafíos con la normalización de los datos de entrada. PointNet permite trabajar directamente con nubes de puntos y muestra una notable robustez ante posibles perturbaciones, lo cual es beneficioso para las bases de datos utilizadas en este estudio. Además, esta arquitectura admite datos de entrada en formato estándar (x, y, z), lo que facilita su manipulación mediante arreglos en diversos lenguajes de programación. PointNet aprovecha propiedades

fundamentales de los puntos en el espacio tridimensional como: el desorden, la interacción entre puntos y la invarianza ante transformaciones geométricas. Otra ventaja significativa es su eficiencia al manejar densidades bajas de puntos, lo cual es especialmente útil para clases de menor densidad, como peatones y ciclistas, en contraste con las clases de vehículos ya que se reportan experimentos desde densidades de 64 puntos [13].

El funcionamiento de PointNet se puede entender como una serie de pasos que permiten procesar nubes de puntos 3D de manera eficiente, preservando la invariancia al orden de los puntos. De manera sencilla se puede seguir parte del proceso de los datos en diferentes etapas:

- Entrada de la nube de puntos
- Transformación de las coordenadas
- Extracción de características locales (MLP)
- Transformación de características (Feature T-Net)
- Agregación global de características (Max-pooling)
- Clasificación o Segmentación
- Salida

La entrada a PointNet es una nube de puntos de tamaño $N \times 3$, donde N es el número de puntos en la nube, una secuencia promedio de la base de datos Kitti reportan densidades aproximadas a 120,000 puntos por escena, para poder asignar una correcta extracción de características se regularizan los datos a coordenadas (x, y, z) de cada punto en el espacio tridimensional.

PointNet comienza con una capa inicial llamada T-Net, que es una red pequeña diseñada para alinear los puntos en el espacio 3D mediante una transformación afín. Esta transformación es aprendida por la red y permite que los puntos de la nube estén alineados independientemente de su rotación o escala. T-Net aprende una matriz de transformación que se aplica a todos los puntos de la nube para llevarlos a un espacio canónico, lo que ayuda a la red a ser invariante a rotaciones y traslaciones. Después de la transformación inicial, los puntos pasan por una red neuronal de perceptrones multicapa (MLP), la cual consiste en varias capas completamente conectadas (fully connected layers). Cada punto de la nube es procesado independientemente por la MLP, que transforma sus coordenadas (x, y, z) en un vector de características de mayor dimensión, representando más que solo la posición espacial del punto. Esta operación ocurre en paralelo para todos los puntos de la nube, lo que da como resultado una representación local enriquecida de cada punto. Por ejemplo, la red puede mapear cada punto a un vector de 64 o 1024 dimensiones, dependiendo de la capa de la red. Una segunda T-Net, conocida como Feature T-Net, se puede utilizar para aprender una transformación en el espacio de características (no solo en las coordenadas espaciales), ayudando a la red a capturar relaciones complejas entre puntos. Esta segunda transformación ayuda a garantizar que las características aprendidas sean invariantes a transformaciones no lineales o complejas en el espacio de características. Una vez que se han extraído características locales para todos los puntos, PointNet aplica una operación de max-pooling para obtener una característica global que represente a toda la nube de puntos. La operación de max-pooling selecciona el valor máximo de cada dimensión a través de todos los puntos, creando un único vector de características globales. Esta operación es clave para garantizar la invariancia al orden de los puntos, ya que no importa en qué orden se ingresen los puntos a la red, el resultado final no cambiará. Dependiendo de la tarea, la red utilizará las características aprendidas de manera diferente, en este caso se utiliza para clasificación.

Beneficios clave de PointNet:

- Puede manejar nubes de puntos de tamaño variable.
- Es eficiente y directa para trabajar con datos de nubes de puntos sin necesidad de convertirlas a otras representaciones (como mallas o volúmenes).
- Este enfoque ha demostrado ser efectivo para tareas de clasificación, segmentación, y detección en escenarios tridimensionales.
- Es capaz de capturar tanto características locales como globales.
- Invariancia al orden de los puntos gracias al max-pooling global.

En la figura 4 se observa la extracción de características dado el etiquetado 2D, pasando por el sensor Velodyne y finalizando con el objeto que ingresa a la red.

III-F. Ajuste de parámetros para arquitectura PointNet

En el análisis de nubes de puntos, se observa que la representación de un peatón suele ser inferior en densidad en comparación con la de un automóvil. Para abordar esta diferencia, se introduce una clase denominada “Cyclist”, que se basa en las características geométricas de los peatones, dado que tanto un ciclista como un peatón comparten similitudes en su forma física. Sin embargo, para preparar los datos para la red neuronal, es necesario ajustar los parámetros y equilibrar las clases, buscando una mayor homogeneidad entre ellas. Se llevaron a cabo experimentos con diversas densidades de puntos utilizando el optimizador Adam con una tasa de aprendizaje de 0.001. La estrategia incluyó aplicar trimesh.sample, un método para agregar puntos aleatorios a mallas con diferentes densidades, para este caso se maneja un rango de 128 a 2048 puntos, dependiendo de las clases, así como ajustar el número de épocas y reducir las capas de entrada de la red neuronal, utilizando la función de activación ReLU recomendada en la documentación.

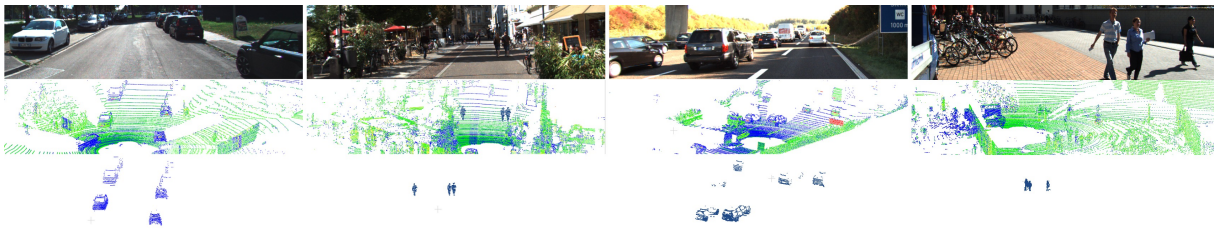

Figura 4: Recorte de datos

RESULTADOS

En este experimento, se observó una tendencia hacia el sobre-ajuste de los datos, lo que llevó a la implementación de técnicas de regularización. Para optimizar el rendimiento del modelo, se evaluaron diferentes optimizadores, incluyendo Adam con tasas de aprendizaje de 0.001 y 0.00125, además de emplear métodos adaptativos para ajustar estas tasas. Se utilizó un conjunto de validación para monitorizar el rendimiento, seleccionando métricas adecuadas como precisión y F1-score, que fueron reportadas en las tablas. Las pruebas iterativas facilitaron la identificación de la mejor combinación de parámetros y arquitectura; sin embargo, aún hay margen para mejorar el ajuste de parámetros.

En el caso de Model40, se obtuvo un mejor rendimiento al utilizar una mayor cantidad de puntos; sin embargo, la detección no fue efectiva con menos de 1024 puntos. Dado que no había ciclistas en "ModelNet", se realizó una prueba utilizando el conjunto de datos KITTI, donde se observó una confusión entre la clase Ciclista y la clase Peatón del modelo CAD.

Tabla V: Resultado General Sydney Dataset

Clase	**No. Puntos**	**Reporte de clasificación**			
		Presition	***Recall***	***f-score***	***Accuracy***
Car	128	1.0	1.0	1.0	0.91
Pedestrian		0.86	0.90	0.88	
Cyclist		0.89	0.88	0.87	
Car	256	0.75	1.0	0.86	0.52
Pedestrian		0.40	0.41	0.88	
Cyclist		0.37	0.35	0.36	
Car	512	1.0	1.0	1.0	0.77
Pedestrian		0.83	0.52	0.62	
Cyclist		0.64	0.90	0.75	

Tabla VI: Resultado General ModelNet Dataset

Clase	**No. Puntos**	**Reporte de clasificación**			
		Presition	***Recall***	***f-score***	***Accuracy***
Car	128	0.57	1.0	0.73	0.74
Cyclist		1.0	0.98	0.99	
Pedestrian		0.80	0.09	0.16	
Car	256	0.60	1.0	0.75	0.78
Cyclist		1.0	1.0	1.0	
Pedestrian		1.0	0.18	0.31	
Car	1024	0.67	1.0	0.80	0.84
Cyclist		1.0	1.0	1.0	
Pedestrian		1.0	0.41	0.58	

IV. CONCLUSIÓN

Los resultados del experimento indican una tendencia hacia el sobre-ajuste de los datos, lo que destaca la necesidad de implementar técnicas de regularización. A pesar de esto, se observó que aún hay espacio para mejorar el ajuste de los parámetros, lo que sugiere que futuras investigaciones deben adoptar un enfoque más exhaustivo y sistemático. En el caso del conjunto de datos Model40, el rendimiento del modelo mejoró al aumentar la cantidad de puntos, aunque se demostró que la detección no fue efectiva con menos de 1024 puntos. Además, la confusión observada entre las clases Ciclista y Peatón en el conjunto de datos KITTI, es crucial contar con más datos específicos de estas clases para aumentar la precisión del modelo.

Para los estudios futuros, se recomienda integrar conjuntos de datos más diversos y equilibrados que aseguren una representación adecuada de todas las clases. Asimismo, se debe realizar un ajuste más minucioso de los hiperparámetros para optimizar el rendimiento. Además, explorar técnicas de aumento de datos y mejorar la calidad de las nubes de puntos podría resultar fundamental para incrementar la robustez y precisión del modelo en la detección de objetos en escenarios del mundo real.

REFERENCIAS

[1] A. Valero Matas, A. De la Barrera, "The Autonomous Car: A better future?", Universidad de Valladolid, Madrid, Palencia, Tech. Rep, 1989-8487, 2019

[2] Anonymous. (2018-03-23) Incident Number 321. in Lam, K. (ed.) Artificial Intelligence Incident Database. Responsible AI Collaborative. Retrieved on September 21, 2024 from incidentdatabase.ai/cite/321.

[3] T. Griggs y D. Wakabayashi. "How a Self-Driving Uber Killed a Pedestrian in Arizona (Published 2018)". The New York Times. Accedido el 22 de septiembre de 2024. [En línea]. Disponible: https://www.nytimes.com/interactive/2018/03/20/us/self-driving-uber-pedestrian-killed.html

[4] C. Gleeson. "Ford's 'BlueCruise' Hands-Free Driving System Under Investigation After Fatal Crashes". Forbes. Accedido el 22 de septiembre de 2024. [En línea]. Disponible: https://www.forbes.com/sites/caileygleeson/2024/04/29/fords-bluecruise-hands-free-driving-system-under-investigation-after-fatal-crashes/

[5] Guardian staff reporter. "Tesla Autopilot feature was involved in 13 fatal crashes, US regulator says". The Guardian. Accedido el 22 de septiembre de 2024. [En línea]. Disponible: https://www.theguardian.com/technology/2024/apr/26/tesla-autopilot-fatal-crash

[6] "The KITTI Vision Benchmark Suite." Andreas Geiger. Accedido el 22 de septiembre de 2024. [En línea]. Disponible: https://www.cvlibs.net/datasets/kitti/

[7] ”Behind the Curtain: Learning Occluded Shapes for 3D Object Detection”, xu2021behind, Xu, Qiangeng and Zhong, Yiqi and Neumann, Ulrich, arXiv preprint arXiv:2112.02205, 2021.
[8] “Sydney Urban Objects Dataset - ACFR - The University of Sydney”. Australian Centre for Robotics - Faculty of Engineering. Accedido el 22 de septiembre de 2024. [En línea]. Disponible: https://www.acfr.usyd.edu.au/papers/SydneyUrbanObjectsDataset.shtml
[9] “Princeton ModelNet”. Princeton ModelNet Dataset. Accedido el 22 de septiembre de 2024. [En línea]. Disponible: https://modelnet.cs.princeton.edu/
[10] ”Frustum PointNets for 3D Object Detection from RGB-D Data”, Charles R. Qi and Wei Liu and Chenxia Wu and Hao Su and Leonidas J. Guibas, arXiv:1711.08488, 2018.
[11] R. Li, X. Li, P. -A. Heng and C. -W. Fu, ”PointAugment: An Auto-Augmentation Framework for Point Cloud Classification”, 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2020, pp. 6377-6386, doi: 10.1109/CVPR42600.2020.00641. keywords: Three-dimensional displays;Training;Shape;Feature extraction;Two dimensional displays;Solid modeling;Training data.
[12] Christopher Lang and Alexander Braun and Lars Schillingmann and Abhinav Valada, .ª Point-Based Approach to Efficient LiDAR Multi-Task Perception”, arxiv:2404.12798, 2024.
[13] Charles R. Qi and Hao Su and Kaichun Mo and Leonidas J. Guibas, ”PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation”, arXiv:1612.00593, 2017, url=https://arxiv.org/abs/1612.00593.
[14] Moorthy C.V.K.N.S.N. Venkateswarulu N. et al Padmaja, B. Exploration of issues, challenges and latest developments in autonomous cars. Springer Open, 2023.
[15] ”Velodyne LiDAR, HDL-64E Datasheet”, 2017.