

Segmentación de imágenes utilizando Redes Generativas Antagónicas

UOC

Ponce Miguel

Máster en Ciencia de Datos
Visión por Computador

Tutor/a de TF

Diego Calvo Barreno

Profesor responsable de la asignatura

Ferran Prados Carrasco

15/01/2023

Universitat Oberta
de Catalunya



Esta obra está sujeta a una licencia de Reconocimiento-No Comercial-Sin ObraDerivada [3.0 España de Creative Commons](#)

Ficha del Trabajo Final

Título del trabajo:	Segmentación de imágenes utilizando Redes Generativas Antagónicas (GAN)
Nombre del autor/a:	Miguel Ponce
Nombre del Tutor/a de TF:	Diego Calvo Barreno
Nombre del/de la PRA:	Ferran Prados Carrasco
Fecha de entrega:	15/01/2023
Titulación o programa:	Máster en Ciencia de Datos
Área del Trabajo Final:	Ciencia de Datos aplicada a la visión por computador
Idioma del trabajo:	Castellano
Palabras clave	Deep learning, Image Segmentation, Generative Adversarial Networks
Resumen del Trabajo	
Esta tesis está enfocada en el uso de redes generativas antagónicas para la segmentación de imágenes y detección de personas usando armas(cortas). Este trabajo propone modelos basados en Aprendizaje profundo (Deep learning) y establece una comparación con los métodos tradicionales de segmentación. Finalmente, se realizan aportes para la detección y alerta temprana en sistemas de vigilancia mediante el empleo de herramientas y técnicas para la segmentación de objetos en imágenes.	
Abstract	
This thesis is focused on using generative adversarial networks for image segmentation and detection of people using guns. This work proposes models based on Deep learning and establishes a comparison with traditional methods of segmentation. Finally, it makes contributions to early detection and warning in surveillance systems by using tools and techniques for the segmentation of objects in images.	

Agradecimientos

Ante todo, quiero agradecer a mi Tutor Diego Calvo Barreno, por sus sugerencias y su compromiso durante la elaboración del proyecto. El estar al tanto y mantenerme informado de las fechas límite de entrega, me ayudó a motivarme y continuar escribiendo e investigando. En el ámbito científico, la inclusión de redes de mejora en la calidad de la imagen le otorgó ese valor adicional a mi trabajo y de manera personal me permitió descubrir nuevas arquitecturas y técnicas. Finalmente, reuniones con la cámara encendida tan solo demuestran la calidad y sencillez de un gran profesional.

Agradezco el apoyo del departamento de gestión de la UOC, en especial a Raquel Orallo, como tutora y a Jordi Casas Roma, director del Máster, con quienes he tenido contacto más de una vez y siempre he recibido una respuesta rápida y clara.

A Marco Russo y Antonio Sarasa, en particular, y a todos los profesores que me han ayudado a formarme, por su interés en mis estudios y su preocupación por mi mejora constante.

A mi querido amigo Julio Cabrera, porque lo considero un referente en el mundo de la Ciencia de Datos y un ejemplo de persona a seguir. Gracias por tu ayuda, sé que siempre puedo contar contigo si surge algún problema o duda.

Para mi amada esposa Carla y mi adorada hija Mia, puedo afirmar que se avecinan tiempos nuevos, ahora tendré mayor tiempo y disponibilidad para ambas. Su guía y paciencia durante estos años de aprendizaje me han hecho crecer y creer que merece la pena seguir estudiando.

A mi querida Madre, gracias por ayudarme a seguir este camino tan generoso y desde ahora motivarme a seguir soñando con un PhD, sin su consejo y guía no soy nadie.

Finalmente, puedo concluir con las siguientes frases propias:

"I can never reach these pictures manually, but my GAN model does."

"Gracias, Señor por otro día más para estudiar, investigar, crear y amar, tu presencia me guía y protege."

Índice

1. Introducción	1
1.1. Contexto y justificación del trabajo	1
1.2. Motivación	2
1.3. Objetivos del Trabajo	3
1.4. Impacto en sostenibilidad, ético-social y de diversidad	3
1.5. Enfoque y método seguido	4
1.6. Planificación del trabajo	4
1.7. Breve sumario de productos obtenidos	7
1.8. Breve descripción de otros capítulos de la memoria	7
2. Materiales y métodos	8
3. Estado del arte	10
3.1. Introducción GAN	10
3.1.1. Elementos de las redes GAN	11
3.1.2. Proceso de entrenamiento	12
3.1.3. Motivaciones para el uso de redes GAN	13
3.2. Arquitecturas de redes generativas antagónicas	13
3.2.1. Principios GAN	14
3.2.2. cGAN	15
3.2.3. infoGAN	15
3.2.4. CycleGAN	16
3.2.5. Pix2Pix	16
3.3. Métricas	17
3.3.1. Equilibrio de Nash	17
3.3.2. Wasserstein GAN	18
3.3.3. Inception score	18
3.3.4. Métricas de rendimiento	19
3.4. Ciclo de vida de la ciencia de datos	20
3.5. Preprocesamiento de imágenes	20
3.5.1. Roboflow	21
3.6. Detección de Objetos	22
3.6.1. YOLO	22

3.6.2. YOLOv5 v7.0	24
3.7. Segmentación de objetos	25
3.7.1. Tipos de segmentación	26
3.7.2. Deep Learning aplicado en tareas de segmentación	27
3.7.3. Detectron2	29
3.7.4. Segmentación personas (BodyPix)	29
3.8. Super Resolución en Imágenes (SR)	30
3.8.1. RDN/RRDN Red densa residual para superresolución de imágenes	31
4. Resultados	33
4.1. Recursos	33
4.2. Organización de proyecto y licenciamiento	34
4.3. Conjuntos de datos	34
4.3.1. Personas y Armas	35
4.3.2. Armas cortas(pistola)	39
4.3.3. Segmentación armas cortas	40
4.4. Modelos detección objetos	42
4.4.1. Detección de armas cortas	42
4.4.2. Detección personas con armas cortas	43
4.4.3. Conversión de imágenes hacia resolución en alta definición (HR) con Super Pixeles	44
4.4.4. Detección y segmentación de armas cortas	45
4.4.5. Segmentación partes cuerpo personas con armas cortas utilizando modelos pre entrenados	50
4.4.6. Preparación de los datos	52
4.4.7. Red de segmentación GAN	57
5. Conclusiones y trabajos futuros	63
6. Glosario	68
7. Bibliografía	69
8. Anexos	73
8.1. Instalar Cuda 11.8 y cuDDN 11.x	73
8.2. Instalar Python 3.11.10	76
8.3. Configurar un entorno con Anaconda y Jupyter	76
8.4. Configurar un entorno de trabajo	78

8.5. Arquitectura Generador	80
8.6. Arquitectura Discriminador	81
8.7. Otras pruebas	82

Lista de Figuras

Figura 1: Esquema general de segmentación de imágenes usando GAN(Linear Semantics in Generative Adversarial Networks, n.d.)	1
Figura 2: Cálculo estimado de costo mensual para el desarrollo del proyecto.....	5
Figura 3. Ciclo de vida para productos de ciencia de datos(marktab, 2020).....	6
Figura 4. Fases método científico(Marcos Gómez-Puerta, 2018).....	8
Figura 5. Ejemplo de una red GAN(Typical Generative Adversarial Networks (GAN) Architecture. Download Scientific Diagram, n.d.).....	10
Figura 6. CGI en una red StyleGAN2 para 'repintar' rostros faciales humanos(Anderson, 2022)	11
Figura 7. Edición de emociones basada en StyleGAN Latent Space, las expresiones con plantilla se aplican a un rostro de entrada(Anderson, 2022).....	11
Figura 8. La arquitectura de la SegAN propuesta para segmentar el tumor cerebral(Xun et al., 2022).	11
Figura 9. Evolución cronológica de las redes GAN(GAN Variations and Timelines Hands-On Neural Networks, n.d.)	14
Figura 10. Tipos de Arquitecturas GAN(Xun et al., 2022).....	14
Figura 11. Ejemplo de proyecto esquema de ciencia de datos, basado en TDSP(marktab, 2020).	20
Figura 12. Pasos para construir, implementar y mejorar un modelo de visión artificial(Overview - Roboflow, n.d.).....	21
Figura 13. Ejemplos de clasificación en imágenes versus detección objetos y segmentación(YOLO Object Detection Explained, n.d.).	22
Figura 14. Ejemplo detección de objetos con YOLOv5 v7.0(Boesch, 2022).	23
Figura 15. Resultados de la velocidad de Yolov5x6(Jocher, 2020)	25
Figura 16. Diferencias entre segmentaciones en cuanto a clase/semántica o de instancia(Ch. 6 - Object Detection and Segmentation, n.d.).	26
Figura 17. Ejemplo de BodyPix v2, detección de múltiples personas.	30
Figura 18, A la derecha una imagen con superresolución y a la izquierda un método tradicional (bicubic upscaling)	31
Figura 19. Red densa residual para superresolución de imágenes(Leverxgroup/Esgan: Enhanced SRGAN. Champion PIRM Challenge on Perceptual Super-Resolution, n.d.)	32
Figura 20. ESRGAN: redes antagónicas generativas de superresolución mejoradas(Leverxgroup/Esgan: Enhanced SRGAN. Champion PIRM Challenge on Perceptual Super-Resolution, n.d.)	33
Figura 21. Ejemplo carga imágenes en Roboflow.	37
Figura 22. Particionamiento del conjunto de datos	37

Figura 23. Identificación de mascará única	37
Figura 24. Creación primera versión del conjunto de datos.....	38
Figura 25. Ejemplo recuadro detección arma.	39
Figura 26. División del conjunto de datos para la detección del arma.	40
Figura 27. Ejemplo de segmentación de armas cortas.	40
Figura 28 División del conjunto de datos para la segmentación de armas cortas.	42
Figura 29. Resultados de la red utilizada en la detección de armas cortas.	43
Figura 30. Ejemplos y resultados de la red empleada en la detección de armas cortas.	43
Figura 31. Ejemplos y resultados de la red utilizada en la detección de personas con armas cortas.	44
Figura 32. Resultados de la precisión para la red de detección de armas cortas con Faster R-CNN.....	46
Figura 33. Resultados de regresión de caja detección de objetos con Faster R-CNN.....	46
Figura 34. Rendimiento de detectores generales de objetos sobre COCO(What Do These Different AP Values Mean, 2020).	47
Figura 35. Resultados detección de armas con sus valores de caja(box).	48
Figura 36. Resultados precisión para la red de segmentación (máscara arma).	49
Figura 37. Resultados de las pérdidas de la red de segmentación.....	49
Figura 38. Ejemplos y resultados de la red de segmentación de armas.	50
Figura 39. Imágenes de ejemplo a la derecha segmentada y a la izquierda original.	55
Figura 40. División del conjunto de datos, imágenes segmentada y original.	56
Figura 41. Ejemplo conjunto datos, imagen real a la derecha, imagen segmentada a la izquierda.	58
Figura 42. Ejemplo rotaciones aleatorias y zoom(jitter)	58
Figura 43. Arquitectura del generador.	58
Figura 44. Ejemplo resultado generador.	59
Figura 45. Arquitectura del discriminador.	59
Figura 46. Ejemplo resultado del discriminador.....	60
Figura 47. Ejemplo imágenes partes cuerpo personas segmentadas en 3D(3DHumanGAN: Towards Photo-Realistic 3D-Aware Human Image Generation, n.d.).	66
Figura 48. Ejemplo conjunto datos armas segmentadas entorno 3D(Giddens, 2019)	66

Tablas

Tabla 1: Tiempo estimado actividades para desarrollo del TFM.....	5
Tabla 2, Contrastos entre los objetivos del discriminador y el del generador.	12
Tabla 3. Esquemas, de los procesos de entradas y salidas para redes GAN.....	12
Tabla 4. Redes incluidas en Detectron2 para la detección y segmentación de objetos.	29
Tabla 5. Conjuntos de datos personas armadas.	36
Tabla 6. Descripción y características conjunto de datos personas armadas	37
Tabla 7. Conjunto de datos resultado persona armada.	38
Tabla 8. Características conjunto de datos resultado persona armada.	39
Tabla 9. Conjunto de datos para detección de armas.	39
Tabla 10. Características del conjunto de datos para la detección del arma.	40
Tabla 11. Conjuntos de datos para le segmentación de armas cortas.	41

Tabla 12. Resultado integración conjuntos de datos armas cortas.....	41
Tabla 13. Características del conjunto de datos para la segmentación de armas cortas.....	42
Tabla 14. Resultados luego de realizar conversión con superpixeles.....	45
Tabla 15.. Resultados procesos de segmentación de partes del cuerpo de personas y armas	52
Tabla 16. Evaluación estadística de las dimensiones de imágenes transformadas con superpixeles.....	53
Tabla 17 Evaluación estadística radio de máscara segmentada del arma.	54
Tabla 18. Comparaciones segmentación de armas; primera fila valores menores al 3.er cuartil.....	55
Tabla 19. Resultados del conjunto de datos de imágenes tratado.....	55
Tabla 20. Conjunto de datos de prueba imágenes segmentadas y real.	56
Tabla 21. Características del conjunto de entramiento.....	56
Tabla 22. Ejemplo resultados visuales por interacción.	61
Tabla 23. Resultados de la evaluación de métricas de pérdida de la red generador y discriminador.	62
Tabla 24. Comparación de los resultados de los métodos tradicionales y GAN frente a COCO.....	63

1. Introducción

Las redes generativas antagónicas (GAN) pueden generar imágenes, síntesis de imágenes a partir de texto, conversión de imagen a imagen y generación de imágenes de alta resolución, pero sigue siendo difícil especificar explícitamente la segmentación de objetos en imágenes. En este trabajo, nuestro objetivo es proponer una red GAN que permita segmentar y extraer objetos en una imagen, ver Figura 1: Esquema general de segmentación de imágenes usando GAN(Linear Semantics in Generative Adversarial Networks, n.d.).

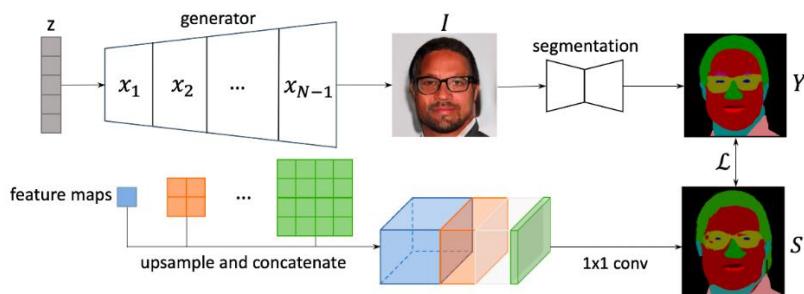


Figura 1: Esquema general de segmentación de imágenes usando GAN(Linear Semantics in Generative Adversarial Networks, n.d.)

1.1. Contexto y justificación del trabajo

El desarrollo de nuevas tecnologías y servicios en la nube ha permitido que en la actualidad existan una serie de herramientas disponibles como servicios para el procesamiento y tratamiento de imágenes. Estas tecnologías fueron pensadas para el desarrollo de aplicaciones a gran escala; en la actualidad tiene un uso ampliamente distribuido en diferentes sectores y mercados. Con el desarrollo de la inteligencia artificial(IA) y de manera particular en los campos de Informática para el modelado de sistemas complejos y de visión por computadora; ha permitido simplificar los procesos de creación de nuevos proyectos, propuestas e investigaciones basadas en técnicas de segmentación y reconocimiento de objetos en imágenes. El propósito de estas herramientas permitirá un avance en los sistemas de recomendación, detección de patrones de comportamiento, vigilancia y alerta temprana, entre otros; mediante el empleo de los objetos o entidades segmentadas dentro de las imágenes, además de establecer relaciones semánticas entre estos. El aporte científico está relacionado con la investigación; efectuando aportes basados en redes generativas para la mejora de la resolución de imágenes, y sugerir mejoras en las técnicas de segmentación en imágenes.

En los últimos años, el uso de redes generativas antagónicas ha consistido generalmente en varias tareas en las cuales se incluyen: la generación de rostros de animales, de humanos, de imágenes a partir de texto, mejorar la calidad o los píxeles dentro de una imagen, y otros. Existen trabajos recientes en los cuales se espera que exista un proceso de generación para encontrar ciertos atributos, colores o patrones dentro de una

imagen; de manera particular corresponde a la segmentación de objetos. Con el propósito de cumplir con este objetivo existen ciertas interrogantes que todavía son productos de investigaciones complejas, por ejemplo: *¿Cómo codifica una GAN bien entrenada, la semántica de la imagen, como el cabello, la nariz y los sombreros en una imagen facial, en su proceso de generación de imágenes?* Si bien podemos extraer la semántica y entender el proceso de extracción en una imagen, podemos desarrollar una idea de cómo se codifica la semántica de la imagen(Xu & Zheng, 2021).

El problema se resuelve mediante la evaluación de la **viabilidad científica**; en donde la capacidad de investigador y del tutor de tesis; son clave para desarrollar este documento porque deben ser profesionales dedicados o con experiencia en al área de Visión por Computadora y Técnicas de Aprendizaje Automático. Adicionalmente, el desarrollador principal de la investigación cuenta con experiencia previa en el campo de la inteligencia artificial y de manera particular en la creación de modelos basados en Aprendizaje Profundo. Los datos o resultados preliminares en conjunto con investigaciones previas relacionadas con la segmentación de imágenes y los conjuntos de datos usados permitirán el desarrollo de la tesis.

Planes de contingencia; debido a la experiencia del tutor en temas relacionados con inteligencia artificial, se puede proponer diversas opciones respecto de las técnicas y modelos de segmentación; es decir, se desarrollarán varios casos de investigación con técnicas basadas en redes profundas. Finalmente, debido a que el campo de tratamiento de imágenes es un campo en desarrollo, se pueden sugerir otro tipo de técnicas basadas en segmentación o librerías existentes de imágenes para la extracción de características; con el objetivo de crear recursos propios como fuentes de datos.

Los resultados de la investigación están determinados por el tiempo dedicado a efectuar investigaciones sobre nuevas propuestas en el estado del arte; en la detección de objetos obtenidos como resultado de la segmentación de imágenes. Realizar aportes en técnicas de segmentación de objetos en imágenes utilizando redes generativas antagónicas. Desarrollar laboratorios con técnicas novedosas para la segmentación y contraste contra otros métodos de segmentación.

1.2. Motivación

En la última década, me he desempeñado como desarrollador de aplicaciones a gran escala, contribuciones emblemáticas y de reconocimiento internacional en otros países. Sin embargo, hace un par de años, en mi búsqueda por integrar soluciones basadas en el empleo de estadísticas y matemática aplicada, me encontré con una serie de cursos abiertos ofertados por MIT(OpenCourseWare), donde encontré varias lecturas sobre inteligencia artificial y la motivación para optar por esta carrera como una Maestría, debido al gran campo de Investigación relacionado. Al finalizar mi formación y debido a mi compromiso con la sociedad, decidí ser parte de un proyecto para la aplicación de redes neuronales y reconocimiento de parentesco (Kinship Recognition) usando imágenes que me permitió conocer con mayor profundidad, técnicas e investigaciones avanzadas aplicadas en la extracción de características en imágenes.

Luego decidí realizar una segunda formación en Ciencia de Datos; en la Universidad Oberta de Cataluña, este es un campo que abarca la IA y que permite desde una visión más amplia conocer arquitecturas, herramientas, aplicaciones, y fundamentos teóricos de la Ciencia Aplicada, mediante el uso de datos para la resolución de problemas. Soy amante de participar en retos de IA y desarrollador de software apasionado porque considero que la combinación de ambos es la clave para dar soluciones a los problemas graves que hoy enfrenta la sociedad.

Como una persona apasionada en el desarrollo de nuevas tecnologías, cuya especialización hace uso de estadística, análisis cuantitativos y cualitativos de datos, me he propuesto como objetivo el crear soluciones basadas en inteligencia artificial, realidad virtual y procesamiento de imágenes. Finalmente, el participar, en conjunto con varios de los profesionales con conocimiento en el uso de técnicas avanzadas y laboratorios para el desarrollo de mi investigación, es clave para conseguir los objetivos de mi propuesta. Luego, me permite estar en contacto con personas que trabajan en el campo de la Ciencia de Datos y relacionados, ya que son profesionales experimentados; puedo referirme a ellos como apoyo y siempre abierto a cualquier sugerencia.

1.3. Objetivos del Trabajo

Proponer modelos innovadores para segmentar objetos en imágenes utilizando redes antagónicas generativas y sus aplicaciones.

- **Objetivo específico 1.** Realizar un estudio para implementar redes antagónicas generativas para mejorar la resolución y la segmentación de objetos en imágenes.

Este objetivo permite estudiar el estado del arte de la segmentación de imágenes con redes GAN.

- **Objetivo específico 2.** Estudiar el estado de la técnica en cuanto a patrones basados en la detección de objetos basados en segmentación de imágenes.

Este objetivo permite estudiar los patrones de segmentación de imágenes y sus aplicaciones en la detección de objetos.

- **Objetivo específico 3.** Evaluar la efectividad de los modelos (redes profundas).

Este proceso permite recopilar datos para una prueba preliminar con el fin de determinar el alcance. Además, comparar diferentes conjuntos de datos para ejecutar las segmentaciones de objetos. Finalmente, comparar los modelos tradicionales de segmentación con la propuesta.

1.4. Impacto en sostenibilidad, ético-social y de diversidad

En los últimos años, los gobiernos del Ecuador han estado involucrados en casos de corrupción, lavado de activos, narcotráfico, microtráfico, y otros; como consecuencia directa es considerado como un país de tránsito y un centro de distribución de droga en América

Latina. En noviembre del 2021, en la Cárcel de Guayaquil, se produjo la masacre de 68 presos, por las diferentes luchas entre los cárteles por obtener el poder(EL MUNDO, 2021). El resultado general del número de reos fallecidos durante ese año fueron alrededor de 329(EL PAÍS, 2021). Adicionalmente, se incrementó la tasa de muertes violentas en 11,5 por cada 100.000 habitantes; número que todavía es alto con relación a los años precedentes y para el 2022, se proyecta, será mucho mayor(Primicias, 2021). Estos hechos han sumergido; a la República del Ecuador a enfrentar una de las más graves y profundas crisis sociales conocida como sicariato; el cual se suele caracterizar por el porte y uso ilegal de armas de fuego con el objetivo de cometer crímenes contra la vida. Desde mi responsabilidad como ciudadano en la lucha en pro del buen vivir; parte de los compromisos de **RESPONSABILIDAD SOCIAL** instituidos en la Constitución de la República del Ecuador, TÍTULO VII y formación como profesional, he notado que no existen sistemas efectivos o aplicaciones de inteligencia artificial para controlar y prevenir la detección de este tipo de actividades(Ferret, 2016). Como una persona apasionada en el desarrollo de nuevas tecnologías, cuya especialización hace uso de estadística, análisis cuantitativos y cualitativos de datos, me he propuesto como objetivo el crear soluciones basadas en inteligencia artificial, realidad virtual y procesamiento de imágenes, que permitan la detección temprana y notificación a los organismos competentes, de este tipo de prácticas.

La tesis pretende demostrar que mediante técnicas de segmentación de objetos en imágenes se pueden sugerir sistemas de vigilancia y alerta temprana contra sospechosos de intento de asesinato y porte ilegal de armas. Por tanto, como ciudadano y profesional, he visto que no existen sistemas efectivos para controlar y prevenir la detección de armas de fuego e intento de asesinato, además es un amplio campo de investigación y aporte para la IA.

1.5. Enfoque y método seguido

El método consiste en seleccionar varios conjuntos de datos que permitan detectar y segmentar personas y armas. A continuación, se crearán una serie de modelos con tecnologías de última generación para detectar los objetos de interés y, posteriormente, se aplicarán algoritmos para la segmentación y extracción de características. Aplicar arquitecturas de redes ampliamente utilizadas para la detección de las partes del cuerpo de las personas y generar datos nuevos para la segmentación. Aplicar la arquitectura Pix2Pix para detectar segmentos en imágenes. Para todos los casos las arquitecturas corresponden a investigaciones ampliamente distribuidas, presentadas como soluciones de cada problema.

1.6. Planificación del trabajo

La planificación del trabajo se basa principalmente en aspectos relacionados con la **viabilidad económica y el tiempo dedicado** al cumplimiento de las diferentes actividades. La tesis cumple con las características técnicas y operativas necesarias para cumplir con sus metas y objetivos en el tiempo establecido para el desarrollo de **300 horas estimadas**

a tiempo parcial. Los resultados e investigaciones que lo conforman consideran los diferentes aspectos técnicos y tiempos de ejecución. En el cálculo de la inversión referencial se tendrá en cuenta el uso de software libre en la medida de lo posible, y solo de manera justificada se usará software con licencia que pueda permitir reducir tiempos y costos de la investigación, ya sea por su versatilidad o funcionalidad durante la investigación. La visión por ordenador, en combinación con la inteligencia artificial, hace uso de plataformas en la nube basadas en GPU, que permiten reducir los tiempos de procesamiento en el tratamiento de estas e incluso algunas arquitecturas dependen de estos dispositivos. Para este caso se utilizó la calcadora ofrecida por Amazon(*AWS Pricing Calculator*, n.d.), ya que permite realizar este cálculo de manera mensual ingresando ciertos parámetros que permitan identificar las necesidades de los Científicos de Datos.

A continuación, en la Figura 2, se presenta un resumen de los costos y el total de la inversión, estos son parámetros referenciales.

Costos Referenciales			
#Unidades GPU	Costo x Hora/Linux	Costo x Hora/Windows	
g4ad.2xlarge	0.3958	0.5304	
g4ad.4xlarge	0.2664	0.9961	
g4ad.8xlarge	0.5202	1.9922	
Show calculations			
1 data scientist(s) x 1 Studio Notebook instance(s) = 1.00 Studio Notebook instance(s)			
1.00 Studio Notebook instance(s) x 10 hours per day x 20 days per month = 200.00 SageMaker Studio Notebook hours per month			
200.00 hours per month x 5.44 USD per hour instance cost = 1,088.00 USD(monthly On-Demand cost)			
Total cost for Studio Notebooks (monthly): 1,088.00 USD			

Figura 2: Cálculo estimado de costo mensual para el desarrollo del proyecto.

En la Tabla 1, se presenta un resumen de las actividades y el tiempo estimado de dedicación en horas, estos son parámetros referenciales.

Actividad	Nro. Horas
Estudiar segmentación de imágenes.	30
Realizar investigación sobre detección de objetos en imágenes.	30
Validar diferentes conjuntos de datos para segmentaciones de objetos	20
Generar una red especializada para detectar la pose y la ubicación de los objetos en imágenes.	90
Investigación otros métodos basados en técnicas tradicionales	30
Combinación con técnicas de ML	30
Elaboración Memoria	70
TOTAL	300

Tabla 1: Tiempo estimado actividades para desarrollo del TFM.

Los equipos de científicos de datos no suelen participar de los beneficios ofrecidos por las metodologías de desarrollo de sistemas actuales(Schleier-Smith, 2015). La metodología ágil ofrece el desarrollo de productos en ciclos cortos de tiempo, pero el soporte

para flujos de trabajo en aplicaciones de aprendizaje automático no está en desarrollo(*Manifesto for Agile Software Development*, n.d.). En general, los científicos de datos deben, a partir de fuentes de datos, construir modelos predictivos; que luego serán validados mediante métricas y finalmente suelen ser publicados en ambientes productivos(Schleier-Smith, 2015). El investigador desarrollará nuevos modelos entrenados en entornos de desarrollo controlados. Al final de cada iteración se validará el cumplimiento de los objetivos establecidos en este documento, donde se realizará la validación de las características. Al inicio de cada nueva iteración, el equipo volverá a planificar y propondrá mejoras a la última iteración, efectuando la retroalimentación(Rubin, 2012).

Debido al tiempo limitado para desarrollar el proyecto, con fuentes de datos y modelos cambiantes, necesitamos una metodología que nos permita abordar la construcción de estos en períodos cortos; lo más importante son los modelos; por lo tanto, es adecuado el uso de metodologías ágiles para este proyecto. Data Science Steam Process es una metodología ágil e iterativa que permite entregar soluciones de Ciencia de Datos de forma eficiente. Al igual que Scrum, incluye varias de las mejoras prácticas de líderes de la industria que han desarrollado iniciativas de software relacionadas con la Ciencia de Datos(marktab, 2020).

El código fuente y las imágenes se guardarán en un repositorio en [git](#), de manera que será más fácil mantenerlos y compartirlos. Se genera un repositorio en la nube ([carpeta drive](#)) donde se almacenarán los pesos de los modelos de los diferentes algoritmos utilizados para el desarrollo del TFM. En el futuro se propone la creación de infraestructura y recursos de sistemas basados en la nube, en particular de Google Cloud Plataforma (GCP), y de contenedores Docker para la publicación y construcción automática de modelos. Las herramientas y utilitarios tienen como función automatizar ciertas tareas relacionadas con el preprocesamiento de datos, la exploración y creación de modelos(marktab, 2020).

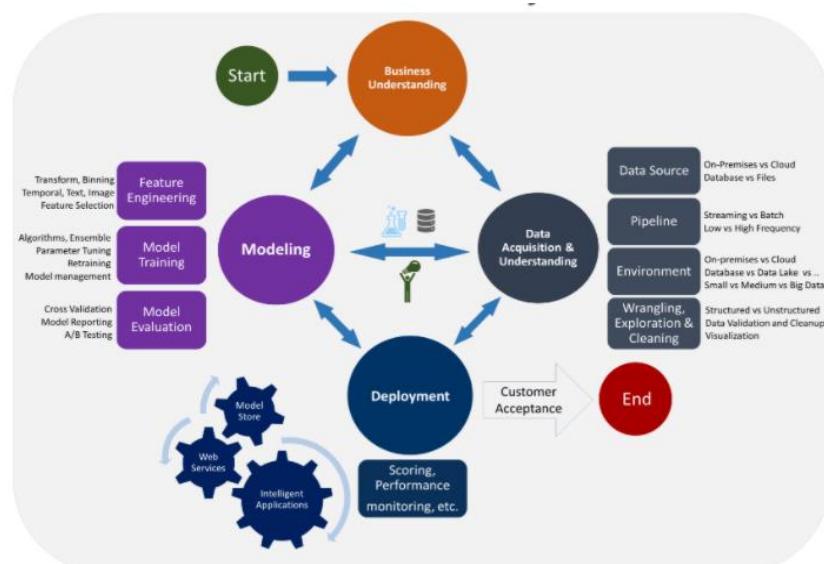


Figura 3. Ciclo de vida para productos de ciencia de datos(marktab, 2020).

1.7. Breve sumario de productos obtenidos

Elaborar diversos modelos para la detección y segmentación de armas. Obtener nuevos conjuntos de datos que permiten asociar armas con personas segmentadas. Generar un repositorio público en GitHub con la estructura de un proyecto de ciencia de datos. Realizar diversas pruebas y evaluaciones de métricas para establecer los modelos más eficaces. Efectuar conversiones de imágenes a representaciones de super pixeles para la detección y predicción de armas. Los modelos pre-entrenados están disponibles en la nube y son gratuitos.

1.8. Breve descripción de otros capítulos de la memoria

La memoria está dispuesta de la siguiente forma, en el primer capítulo se hace una introducción que da un contexto a la problemática. El capítulo 2 presenta la metodología usada para el desarrollo del proyecto. A continuación, en el capítulo 3 se presenta una revisión de las redes generativas antagónicas; en este capítulo, se exponen trabajos previos relacionados con GAN. En el capítulo 4 se explica cómo se han obtenido los datos y cómo se han tratado las imágenes; en este capítulo se encuentra el mayor contenido de la memoria, ya que explica de forma detallada los modelos elaborados y las compara contra otros modelos. Asimismo, se presentan las arquitecturas empleadas para el desarrollo del problema. Por último, el capítulo 5 se refiere a los criterios de evaluación de los resultados y conclusiones.

2. Materiales y métodos

El método científico puede hacer uso de dos tipos de un razonamiento inductivo o bien deductivo, estos se definen según(*Significados*, n.d.):

“El método deductivo consiste en extraer una conclusión con base en una premisa o a una serie de proposiciones que se asumen como verdaderas.”

“El método inductivo es una estrategia de razonamiento que se basa en la inducción, para ello, procede a partir de premisas particulares para generar conclusiones generales.”

Por un lado, el método deductivo parte de una teoría y posteriormente se recogen datos para aceptarla o bien refutarla; por el otro lado, el método inductivo se caracteriza por identificar leyes o teorías partiendo de hechos particulares, para llegar a conclusiones generales. Los dos contribuyen enormemente al método científico, ya que el ser humano en general busca comprender la realidad para adquirir conocimiento a través de leyes que sean generalizables y permitan comprender el funcionamiento de procesos complejos y, en la medida de lo posible, predecir sus resultados. Las fases y componentes de dicho método son las siguientes(Marcos Gómez-Puerta, 2018):

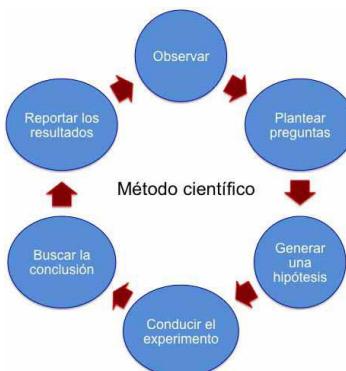


Figura 4. Fases método científico(Marcos Gómez-Puerta, 2018)

Dado que el papel del Científico de datos es una mezcla de una serie de perfiles que abarcan estadística, matemática, informática y comunicación, podría decirse que el hacerse preguntas debe fundamentarse en el hecho de que puedan responder a cuestiones que se basen en posibles patrones y números(Rouhiainen, 2019). Por ejemplo, cuál podría ser el precio ideal de un producto en función del comportamiento de compra. Además, es necesario conocer cuáles departamentos participan en este requerimiento, por ejemplo, marketing y ventas porque son ellos quienes deben responder a estas preguntas.

El papel de un Científico de datos va desde la captura, tratamiento, publicación y hasta el fin del ciclo de vida de datos; tiene por detrás inversión sea de tiempo o de investigación para una necesidad (Kim et al., 2018). Posteriormente, el uso del método científico requiere identificar qué fuentes potenciales pueden ayudar a formular hipótesis, hacer proyecciones o comparar con estudios recientes que puedan justificar la inversión de tiempo en una propuesta o proyecto. Este trabajo seguirá los pasos establecidos en el método científico, en donde se pretende definir y poner a prueba una hipótesis mediante

experimentos controlados que sigan el método científico. Los pasos a seguir de manera general son:

- **Observación e Hipótesis:** Se realiza la observación del mundo con el objetivo de recoger información de la realidad y se realizan hipótesis para tratar de obtener una afirmación que sea extensible de manera general (Horan & Lavelle, n.d.)
- **Experimentación:** Se realiza un experimento que prueba si la predicción es precisa, repetible y, por lo tanto, si su hipótesis es compatible o no. Pueden existir cambios de ciertos factores, pero estos no pueden alterar el resultado de la prueba. Se basa en repetir los experimentos varias veces para asegurarse de que los resultados iniciales son veraces y no son resultado de un accidente (*Steps of the Scientific Method*, n.d.).
- **Analizar los datos y extraer conclusiones:** Independientemente del resultado de la prueba, es necesario mostrar los resultados de manera cuantitativa, en donde se muestran los resultados del experimento. Se pueden emplear cálculos, gráficos o tablas que permitan resumir o agregar la información y facilitar la comprensión de los datos obtenidos de la experimentación(*Steps of the Scientific Method*, n.d.). Los científicos de datos y, en general, descubren que sus predicciones pueden ser imprecisas, por lo que sus hipótesis se descartan. Sin embargo, estos casos deben comunicarse y, a continuación, construir una nueva hipótesis.
- **Comunicar:** Dentro de las etapas finales el resultado de la metodología es comunicar estos resultados. En el ámbito científico, los investigadores tienen algo en común: están interesados en los hallazgos, sin importar si respaldan o no la hipótesis original, y es por eso que es clave la comunicación, porque de eso dependerá la validez o no de la propuesta(*Steps of the Scientific Method*, n.d.).

3. Estado del arte

En este capítulo se presenta la teoría requerida para comprender los objetivos del presente estudio. En concreto, la Sección 3.1 introduce las Redes Generativas Antagónicas, la Sección 3.2 expone las principales arquitecturas de redes generativas, la Sección 3.3 describe las características y los desafíos respecto a las métricas utilizadas para la evaluación de este tipo de redes.

3.1. Introducción GAN

En 2014, Ian Goodfellow creó las redes generativas antagónicas (GAN). Esta técnica permite sintetizar y generar datos realistas empleando dos redes neuronales independientes. Existen otras técnicas para generar datos; por ejemplo, el imbalanced learning (manejo de datos desbalanceados) se usa como técnica de generación de datos, pero sus resultados y versatilidad lo distinguen del resto. Las GAN han logrado resultados notables, como la capacidad de generar imágenes falsas con una calidad similar a la del mundo real, convertir imágenes en garabatos, segmentar imágenes, todo sin la necesidad de grandes cantidades de datos(Langr & Vladimir, 2019a). El generador aprende principalmente la distribución real de los datos y produce datos "falsos" similares, parecidos al original, el discriminador; se considera como un juez, discrimina entre lo que considera como original y datos falsos(generados), como se puede ver en la Figura 5. Este retroalimenta el resultado al generador, que utiliza este insumo para generar datos cada vez más realistas. A medida que la red aprende, las imágenes generadas se asemejan cada vez más al original, y la capacidad del discriminador para evaluar las imágenes generadas se vuelve cada vez más compleja. Finalmente, cuando ambas partes llegan a un punto de equilibrio, se puede decir que han entrado en un estado de equilibrio y por ende sería adecuado que terminen su etapa de entrenamiento(Xun et al., 2022).

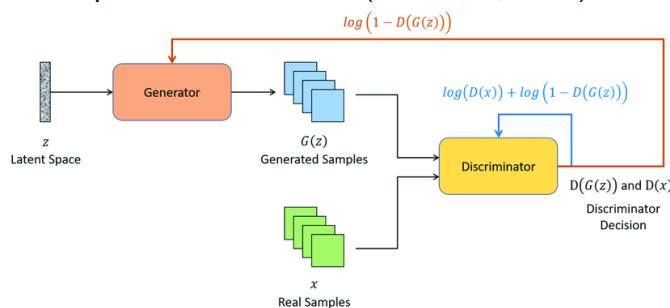


Figura 5. Ejemplo de una red GAN(Typical Generative Adversarial Networks (GAN) Architecture. | Download Scientific Diagram, n.d.)

Debido a su versatilidad podemos encontrar varios ejemplos de sus usos y aplicaciones. Por ejemplo, en la síntesis de rostros humanos, ilustrada en la Figura 6. Generación de sentimientos sobre imágenes, como se puede ver en la Figura 7. Adicionalmente, existen ejemplos en medicina relacionados con la segmentación de tumores, esto se puede ver en la Figura 8. En adelante presentaremos sus aplicaciones para el tercer caso de manera particular.

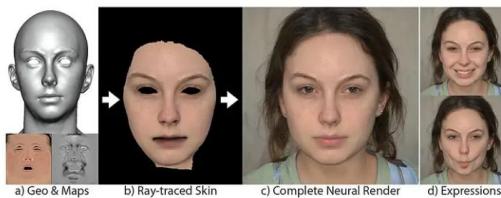


Figura 6. CGI en una red StyleGAN2 para 'repintar' rostros faciales humanos(Anderson, 2022)

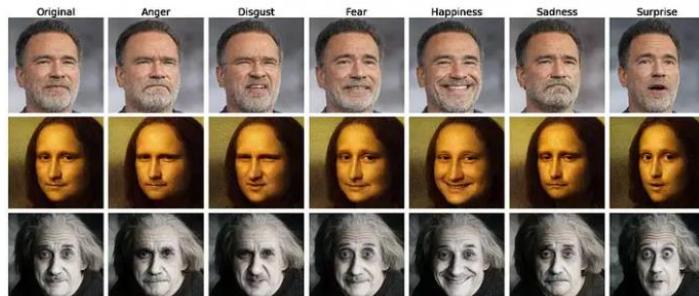


Figura 7. Edición de emociones basada en StyleGAN Latent Space, las expresiones con plantilla se aplican a un rostro de entrada(Anderson, 2022).

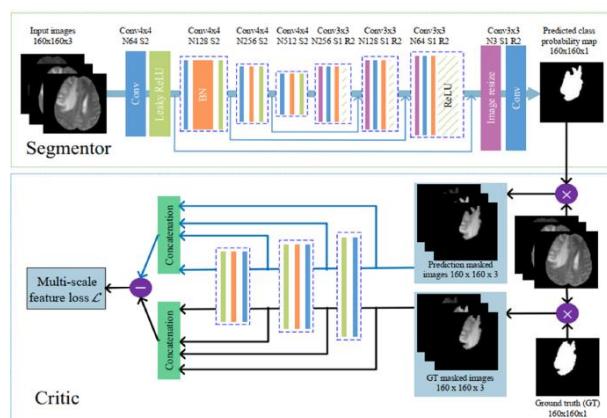


Figura 8. La arquitectura de la SegAN propuesta para segmentar el tumor cerebral(Xun et al., 2022).

3.1.1. Elementos de las redes GAN

Según (Langr & Vladimir, 2019, cap1), los elementos que comúnmente se suelen encontrar en estas redes son:

- **Datos de entrenamiento:** es un conjunto de ejemplos reales que queremos que el Generador aprenda a simular, y que con el tiempo mejore cada vez la calidad.
- **Vector de ruido aleatorio:** Esta entrada es un vector con datos aleatorios que el Generador usa como punto de partida para sintetizar ejemplos falsos.
- **Generador:** es una red, que toma un vector de datos aleatorios (z) como entrada y genera ejemplos falsos (x^*). Tiene como objetivo que los nuevos datos sean indistinguibles de los ejemplos reales.
- **Discriminador (Juez):** es una red que toma como entrada un ejemplo real (x) o un ejemplo falso (x^*) producido por el Generador; en donde trata de determinar y genera la probabilidad de que el ejemplo sea real o falso.

- **Entrenamiento/ajuste:** es un proceso iterativo que, para cada juicio del Discriminador, utiliza los resultados para ajustar iterativamente las redes.

Objetivo del Discriminador	Objetivo del Generador
Los pesos y sesgos de Discriminador se actualizan para maximizar su precisión.	Los pesos y sesgos del Generador se actualizan para maximizar la probabilidad de que el Discriminador se equivoque al etiquetar muestra.

Tabla 2, Contrastes entre los objetivos del discriminador y el del generador.

3.1.2. Proceso de entrenamiento

Las subredes del Generador y Discriminador tienen ciertas entradas y salidas; posteriormente veremos su implementación más general mediante pseudocódigo, por el momento nos interesa conocer sus diferencias; según (Langr & Vladimir, 2019, cap1),

	Generador	Discriminador
Entradas	Un vector de números aleatorios	Dos fuentes: Ejemplos reales provenientes del conjunto de datos de entrenamiento Ejemplos falsos provenientes del Generador
Salidas	Ejemplos falsos que se esfuerzan por ser lo más convincentes posible	Probabilidad prevista de que el ejemplo de entrada sea real

Tabla 3. Esquemas, de los procesos de entradas y salidas para redes GAN.

El objetivo, al generar datos falsos que no se puedan distinguir de los reales, hace que el juez haga su “mejor esfuerzo” al distinguir entre los ejemplos provenientes del generador. El algoritmo de entrenamiento para redes GAN; según (Langr & Vladimir, 2019, cap1), es:

Algoritmo de entrenamiento GAN

Para cada iteración de entrenamiento hacer

Entrenar al discriminador:

- ✓ Tomar un ejemplo real aleatorio \mathbf{x} del conjunto de datos de entrenamiento.
- ✓ Obtener un nuevo vector con ruido aleatorio \mathbf{z} y, utilizando la red del generador, generar un ejemplo falso \mathbf{x}^* .
- ✓ Emplear la red del discriminador para clasificar \mathbf{x} y \mathbf{x}^* .
- ✓ Calcular los errores de clasificación y retro propaga el error total para actualizar los parámetros del discriminador, buscando minimizar los errores de clasificación.

Entrenar al Generador:

- ✓ Obtener un nuevo vector de ruido aleatorio z y, empleando la red del generador; luego, sintetice un ejemplo falso x^* .
- ✓ Utilizar la red del discriminador para clasificar x^* .
- ✓ Calcular el error de clasificación y retro propagar el error para actualizar los parámetros del generador, buscando maximizar el error del Discriminador.

3.1.3. Motivaciones para el uso de redes GAN

En general, se han propuesto muchos enfoques y usos interesantes basados en GAN para la síntesis de imágenes. Según Wikipedia("Segmentación (procesamiento de imágenes)," 2022) la segmentación se concibe como:

"La segmentación es uno de los problemas generales del campo de la visión artificial y consiste en dividir una imagen digital en varias regiones (grupos de píxeles) denominadas segmentos. Más concretamente, la segmentación es un proceso de clasificación por píxel que asigna una categoría a cada píxel de la imagen analizada."

En general, se han sugerido muchos enfoques y usos interesantes basados en GAN para la síntesis de imágenes. Se puede decir que el mayor aporte científico es el aplicado en la medicina; y un caso particular es la segmentación de objetos y órganos en imágenes médicas, se considera esencial para muchas aplicaciones, como detección, clasificación y análisis de formas. La naturaleza tediosa y lenta de la segmentación manual hizo que los métodos basados en aprendizaje profundo, sea uno de los campos más activos de investigación. Sin embargo, en muchos escenarios, el procedimiento de evaluación y optimización basado en píxeles con el empleo de redes profundas no es suficiente para capturar nociones de estructuras anatómicas(Xun et al., 2022). La transferencia del concepto GAN al área de la segmentación de imágenes ha resultado ser altamente útil. Por ejemplo, forzar a la red para que genere segmentaciones que no pueden diferenciarse de las segmentaciones manuales a través del entrenamiento adversario es una alternativa eficaz y eficiente a la construcción de una canalización de post procesamiento manual. Un enfoque diferente que utiliza arquitecturas avanzadas de redes GAN ha demostrado éxito en la segmentación de tumores hepáticos y cerebrales en resonancia magnética, así como en la segmentación celular en datos microscópicos. Estas redes también se pueden usar para superar problemas de desequilibrio de clases(Kazeminia et al., 2020).

3.2. Arquitecturas de redes generativas antagónicas

En la actualidad existen desarrollos significativos relacionados con la investigación de GAN en los últimos tiempos. La siguiente línea de tiempo, muestra algunos de los avances más notables y cuáles son las arquitecturas más destacadas:

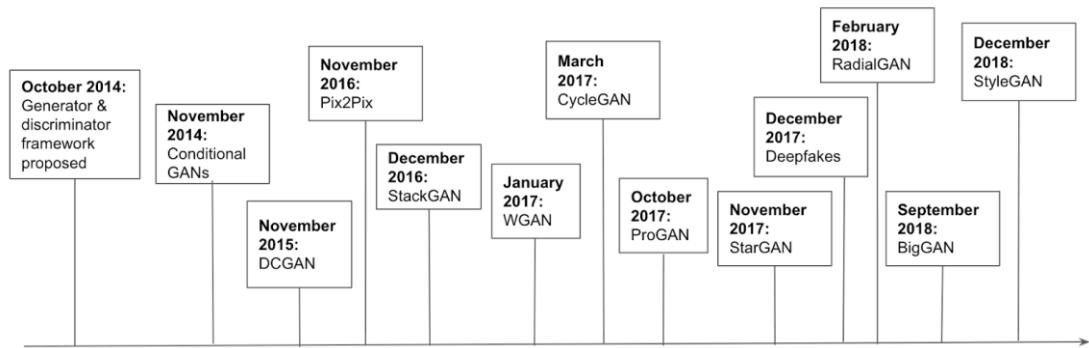
Timeline: key developments in GAN research

Figura 9. Evolución cronológica de las redes GAN(GAN Variations and Timelines | Hands-On Neural Networks, n.d.)

Adicionalmente, existen diversas arquitecturas de redes generativas, en particular para la segmentación de imágenes se pueden destacar las redes cGAN, DCGAN, infoGAN, CycleGAN, etc, como se puede ver en la Figura 10.

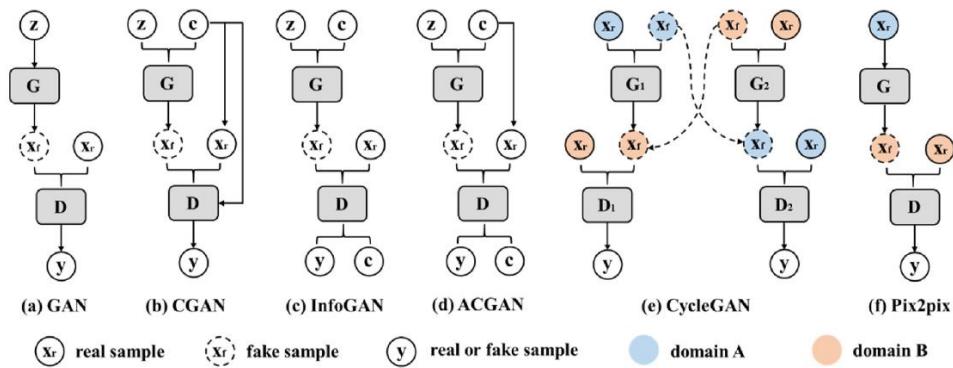


Figura 10. Tipos de Arquitecturas GAN(Xun et al., 2022).

3.2.1. Principios GAN

El marco GAN consiste en un conjunto de datos de entrenamiento \mathbf{X} , cuya distribución subyacente denotamos P_{real} , y un par de redes en competencia: un generador (**G**) con parámetros θ_G y un discriminador (**D**) con pesos θ_D (ver Fórmula 1). **G** tiene como objetivo encontrar un mapeo $\hat{x} = G(z; \theta_G)$ que mapee las variables aleatorias latentes $z \sim p_z(z)$, extraídas de una distribución anterior p_z , a los datos generados $\hat{x} \in \widehat{\mathbf{X}}$, que se supone que siguen la distribución $p_\theta(\hat{x}|z)$. El objetivo principal es optimizar este mapeo de modo que la distribución de los datos generados $\widehat{\mathbf{X}}$ se asemeje a la distribución de los datos de entrenamiento \mathbf{X} . En otras palabras, se supone que G genera datos falsos que no deben distinguirse de los datos reales. Esto se logra con la ayuda de la red discriminadora D, cuya tarea es clasificar entre muestras falsas y reales. Esencialmente, D es un clasificador binario que produce $D(x)=1$ para muestras reales y $D(\hat{x})=0$ para datos falsos. Ambas redes son adversarias porque G intenta sintetizar gradualmente muestras cada vez más realistas que

D clasificaría erróneamente como reales, mientras que D aprende constantemente a diferenciar entre muestras reales y sintetizadas. Hablando matemáticamente, D y G juegan un juego minimax de dos jugadores con la siguiente función de valor V(G, D), esto según (Kazeminia et al., 2020), a continuación se presenta la fórmula matemática:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log(D(x))] + \mathbb{E}_{z \sim p_z(z)} [1 - \log(D(G(z)))]$$

Fórmula 1. Representación minmax GAN(Kazeminia et al., 2020)

Nota: Las demostraciones o particularidades de estas fórmulas; se encuentran fuera del alcance de este trabajo.

En adelante, las definiciones particulares de las fórmulas para las diferentes arquitecturas hacen referencia hacia este esquema base, en donde en la práctica, estas redes generalmente se implementan como perceptrones multicapa (MLP) o redes neuronales convolucionales y se entrena con gradientes estocásticos de mini lotes que descienden de manera alterna. Una vez entrenado, es suficiente muestrear un z aleatorio y alimentarlo a través del generador para sintetizar datos(Kazeminia et al., 2020).

3.2.2. cGAN

cGAN es un modelo extendido de GAN propuesto por Mirza y Osindero en 2014. Dado que en el GAN original no se proporciona un control explícito sobre la generación de datos reales, se propuso el GAN condicional (cGAN) para incorporar información adicional como etiquetas de clase en el proceso de síntesis (ver Fig. 9). En la cGAN, al generador se le presenta un ruido aleatorio z junto con alguna información previa c . Además, el conocimiento previo c se alimenta al discriminador junto con los datos reales o falsos correspondientes. Matemáticamente hablando, el marco cGAN se da de la siguiente manera(Xun et al., 2022):

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log(D(x|c))] + \mathbb{E}_{z \sim p_z(z)} [1 - \log(D(G(z|c)))]$$

Fórmula 2. Representación minmax cGAN(Kazeminia et al., 2020)

cGAN es realizado mediante la transmisión de la información adicional c a la generación modelo y modelo discriminante como parte de la capa de entrada. Del mismo modo, la función objetivo de cGAN es un juego minimax para dos jugadores con condiciones probabilidad. Como modelo de confrontación generacional con restricciones condicionales, cGAN puede considerarse como una mejora de la GAN no supervisada en el modelo supervisado(Xun et al., 2022).

3.2.3. infoGAN

En 2016, Chen et al. Propuso InfoGAN, una GAN que es capaz de aprender la representación desenredada sin supervisión (ver Fig. 9). Basado en la teoría de la

información, InfoGAN tiene el potencial de resolver el problema explicativo de las variables ocultas de GAN. La “Info” representa la información mutua entre la distribución generada $\mathbf{G}(\mathbf{z}, \mathbf{c})$ y la codificación implícita \mathbf{c} . Para hacer la correlación entre \mathbf{x} y \mathbf{c} cerca, $I(\mathbf{c}; \mathbf{G}(\mathbf{z}, \mathbf{c}))$ debe maximizarse. Por lo tanto, se agregan restricciones de términos regulares a la función objetivo original de GAN. Mientras tanto, se define una distribución auxiliar $\mathbf{Q}(\mathbf{c}|\mathbf{x})$ para aproximarse a posterior $\mathbf{P}(\mathbf{c}|\mathbf{x})$, de modo que el límite inferior de la variación de $\mathbf{P}(\mathbf{c}|\mathbf{x})$ puede ser obtenido. Finalmente, la función objetivo se puede expresar como(Xun et al., 2022):

$$\min_{\mathbf{G}, \mathbf{Q}} \max_{\mathbf{D}} V_I(D, G, Q) = V(D, G) - \lambda L_I(G, Q)$$

Fórmula 3. Representación minmax cGAN(Kazeminia et al., 2020)

3.2.4. CycleGAN

Para la transformación de imágenes entre dos dominios, el modelo debería poder extraer características de ambos dominios y descubrir la relación subyacente entre ellos(ver Fig. 9). Jun-Yan Zhu et al, en 2017, propuso CycleGAN para proporcionar estos principios. Esta arquitectura se dedica a resolver el problema de la transformación de imágenes entre dos dominios; utiliza dos GAN con espejo simétrico para formar una red en anillo para encontrar el mapeo entre el dominio \mathbf{X} y el dominio \mathbf{Y} (Kazeminia et al., 2020). El generador \mathbf{G} toma una imagen de \mathbf{X} e intenta asignarla a \mathbf{Y} . El discriminador \mathbf{D}_Y determina si la imagen es generada por \mathbf{G} o \mathbf{Y} . De manera similar, el generador \mathbf{F} toma una imagen de \mathbf{Y} e intenta mapearla a \mathbf{X} , y el discriminador \mathbf{D}_X determina si la imagen es generada por \mathbf{F} o si existe en \mathbf{X} (Xun et al., 2022). Más precisamente, esta función de pérdida cíclica minimiza la discrepancia entre la imagen original y la reconstrucción obtenida de los generadores encadenados(Kazeminia et al., 2020). La función de pérdida se define como:

$$L(G, F, D_X, D_Y) = L_{\text{GAN}}(G, D_Y, X, Y) + L_{\text{GAN}}(F, D_X, Y, X) + \lambda L_{\text{cyc}}(G, F)$$

with

$$L_{\text{cyc}}(G, F) = \mathbb{E}_{x \sim P_{\text{data}}(x)} [\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim P_{\text{data}}(y)} [\|G(F(y)) - y\|_1]$$

Fórmula 4. Función de perdida CycleGAN(Kazeminia et al., 2020)

3.2.5. Pix2Pix

Esta es una variante muy exitosa del cGAN para traducción de imagen a imagen de alta resolución. Isol et al, en 2017 propuso pix2pix (ver Fig. 9) como un caso especial cGAN para conversión entre diferentes tipos de imágenes. Esta red usa U-net (Ronneberger et al. (2015) para mejorar el detalle y Markov discriminador (PatchGAN) para procesar las partes de alta frecuencia de la imagen. La entrada de \mathbf{G} es una imagen de tipo \mathbf{Y} y (condición y), y la salida es la imagen generada $\mathbf{G}(\mathbf{y})$. La entrada de \mathbf{D} es una imagen de tipo \mathbf{X} x y una Imagen tipo \mathbf{Y} y. \mathbf{D} necesita determinar si la imagen \mathbf{x} es la imagen real de \mathbf{y} , y genera una

probabilidad. La pérdida del modelo incluye la pérdida L1 y Pérdida de CGAN. Se encuentra a través de experimentos que agregando pérdida L1/L2 en la imagen generada y la imagen real puede acelerar la convergencia y mejorar la precisión.(Xun et al., 2022). Los autores demostraron que las conexiones de salto dentro del generador U-Net son beneficiosas para la coherencia global de las imágenes sintetizadas. En contraste con el marco GAN original, Pix2Pix requiere pares de imágenes de entrada y salida correspondientes. Esto permite el uso de la pérdida ℓ_1 entre la salida de los generadores y la imagen real para estabilizar el entrenamiento(Kazeminia et al., 2020).

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z)))] ,$$

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1]. \quad \mathcal{L}_{L2}(G) = \mathbb{E}_{x,y,z} (y - G(x, z))^2$$

Fórmula 5. Función de pérdida Pix2Pix y funciones de distancia L1 y L2(Mady), 2019, p. 2)

3.3. Métricas

Uno de los mayores logros de estas redes está relacionado con el campo de la medicina y sus avances; para medir la solidez y el grado de realismo, los trabajos revisados han tomado diferentes caminos, algunos de los cuales son discutibles. Las métricas de síntesis comunes como PSNR, SSIM o MSE pueden compararse utilizando estas medidas; sin embargo, estas pueden no captar los beneficios introducidos por las GAN(Sara, 2019). De hecho, la mayoría de las obras se fundamentan en el juicio de expertos humanos (pruebas visuales de Turing) para evaluar el realismo. Posteriormente, la validez del método de evaluación o el diseño del estudio es cuestionable. En algunos modelos se puede evaluar el nivel de realismo aplicando medidas específicas del dominio, por ejemplo, buscando puntos de referencia anatómicos y su posición relativa contra los datos sintéticos. Los modelos de aprendizaje profundo también pueden emplearse como proxy para medir la fidelidad y el contenido de la información entrenando un modelo con datos sintéticos, un segundo modelo con datos reales y comparando su rendimiento en un conjunto de pruebas de imágenes reales(Kazeminia et al., 2020). Si no se dispone de medios específicos se puede usar, medidas enigmáticas como Inception-Score, Frechet-Inception-Distance o Sliced-Wasserstein-Distance, que cuantifican la superposición distribucional de los valores reales contra los datos sintéticos en el espacio de funciones(Langr & Vladimir, 2019b). En general, la comunidad carece aún de una medida cuantitativa universal y significativa para juzgar el realismo de las imágenes sintéticas. Independientemente de las propuestas de los trabajos antes mencionados, se ha demostrado que las GAN se pueden emplear con éxito en tareas de clasificación y segmentación(Kazeminia et al., 2020).

3.3.1. Equilibrio de Nash

El equilibrio de Nash según Wikipedia("Equilibrio de Nash," 2022) se define como:

"*El equilibrio de Nash es una situación en la cual todos los jugadores han puesto en práctica, y saben que lo han hecho, una estrategia que maximiza sus ganancias dadas las*

estrategias de los otros. Consecuentemente, ningún jugador tiene ningún incentivo para modificar individualmente su estrategia.”

Se podría decir que las redes GAN representan un juego de suma cero entre dos jugadores, un generador y un discriminador, en el cual cada uno aprende del otro. Aunque las GAN han alcanzado un rendimiento y un interés científico elevados, la optimización minimax utilizada en estas supone grandes desafíos teóricos y empíricos. Las GAN entrenadas con métodos de optimización de primer orden a menudo no logran converger en una solución estable; esto quiere decir que los jugadores no pueden mejorar su objetivo, y por ende el equilibrio de Nash del juego subyacente. Estos problemas plantean la cuestión sobre la existencia de equilibrios de Nash en los juegos de suma cero aplicados en las redes GAN. Según el trabajo desarrollado por Farzan Farnia et al, logran demostrar a través de resultados teóricos y numéricos que, de hecho, los juegos de suma cero de GAN pueden no tener equilibrios de Nash(Farnia & Ozdaglar, 2020). Según Langr et al, un rede GAN alcanza el equilibrio de Nash cuando se cumplen las siguientes condiciones(Langr & Vladimir, 2019a):

- El generador produce ejemplos falsos que son indistinguibles de los datos reales.
- El discriminador puede, en el mejor de los casos, adivinar aleatoriamente (50/50) si un ejemplo es real o falso.

3.3.2. Wasserstein GAN

Wasserstein GAN es un nuevo desarrollo en el entrenamiento que rápidamente alcanzó cierta popularidad académica, es importante por tres razones(Langr & Vladimir, 2019b):

- ✓ Mejora significativamente las funciones de pérdida, que ahora son interpretables y proporcionan criterios de parada más claros.
- ✓ Empíricamente, la WGAN tiende a tener mejores resultados.
- ✓ Propone una mejor función de pérdida que mitiga, los problemas actuales para establecer la pérdida y la divergencia KL que estamos tratando de aproximar.

Debido a que el generador y discriminador, son competidores, no tenemos un punto claro para establecer cuando dejar de entrenar. La WGAN utiliza la métrica de distancia de Wasserstein como una función de pérdida que se correlaciona claramente con la calidad visual de las muestras generadas(Langr & Vladimir, 2019b).

3.3.3. Inception score

La ventaja de esta métrica es que ha sido ampliamente validadas con aplicaciones relacionadas con el realismo de la imagen. Se ha demostrado que se correlaciona con la intuición humana sobre lo que constituye una imagen real. Luego este al ser un método de evaluación estadística, establecemos una lista de requisitos de alto nivel de lo que garantizaría nuestro método de evaluación ideal(Langr & Vladimir, 2019b), por ejemplo:

- ✓ Las muestras generadas parecen algo real y distinguible.
- ✓ Las muestras parecen reales y podemos generar muestras de elementos en nuestro conjunto de datos.
- ✓ El clasificador confía en que lo que ve es un elemento que reconoce. De hecho, existen clasificadores de visión por computadora que son capaces de clasificar una imagen hacia una clase en particular, con un cierto grado de confianza, en donde Inception, es uno de esos clasificadores.
- ✓ Las muestras generadas son variadas y contienen, idealmente, todas las clases que estaban representadas en el conjunto de datos original.

Pueden existir requisitos adicionales en nuestro modelo generativo, sin embargo, lo mencionado anteriormente es un punto de partida. La puntuación inicial (IS) se introdujo por primera vez en un artículo de 2016 que validó ampliamente esta métrica y confirmó que, de hecho, se correlaciona con las percepciones humanas de lo que constituye una muestra de alta calidad. Luego, esta métrica se ha vuelto popular en la comunidad de investigación de GAN(Langr & Vladimir, 2019b).

3.3.4. Métricas de rendimiento

Existen varias métricas de rendimiento en los modelos de segmentación de GAN. Las medidas de rendimiento múltiples se utilizan para analizar el rendimiento de los modelos, a veces, el modelo tiene un rendimiento superior para cierta medida o en otros casos uno deficiente con otra métrica. Es necesario usar métricas de rendimiento que reflejen si el modelo propuesto se entrena correctamente o mal en un conjunto de datos determinado. Para este caso, las métricas están destinadas a comparar los resultados de la segmentación con las imágenes reales, que pueden ser una etiqueta o una máscara. La mayoría de estas métricas se calculan empleando una matriz de confusión mediante el uso de cuatro elementos básicos, como la tasa de verdaderos positivos (TP), la tasa de verdaderos negativos (TN), la tasa de falsos positivos (FP) y la tasa de falsos negativos (FN)(Iqbal et al., 2022). Ciertas investigaciones que usaron imágenes generadas para aumentar el entrenamiento de los modelos de segmentación informaron las métricas de segmentación del conjunto de prueba final del coeficiente de similitud de dados (Dice) y, a veces, informaron la precisión, precisión, sensibilidad y especificidad de los píxeles de segmentación(Jeong et al., 2022). Este coeficiente es un índice de superposición espacial y una métrica de validación de reproducibilidad. El valor de Dice varía de 0 hasta 1, en donde valores cercanos o igual a cero indica que no hay superposición espacial entre dos conjuntos de resultados de segmentación binaria; por el otro lado, valores cercanos a uno indican una superposición completa. Existen otras métricas de validación consideradas para la validación estadística incluyeron el coeficiente de similitud de Jaccard u otras estadísticas basadas en la distancia(Zou et al., 2004).

3.4. Ciclo de vida de la ciencia de datos

En el proyecto actual se realizarán diferentes tareas relacionadas con el preprocesamiento de datos, la creación de modelos de prueba y el uso de modelos preentrenados. Como parte de un proyecto de Ciencia de Datos, al tener varias tareas previas como insumo para la creación de modelos basados en Deep Learning, existen varias etapas que se ejecutan de forma iterativa y se pueden observar en la Figura 3. Ciclo de vida para productos de ciencia de datos(marktab, 2020).

TDSP: En este proyecto se empleará un sistema de control de versiones denominado Git, y se creará una estructura de repositorios separados con criterios basados en la ubicación de los datos en forma cruda, datos preprocesados y procesados, código fuente, y exportación de modelos pre-entrenados. La figura siguiente muestra un ejemplo de esta estructura:

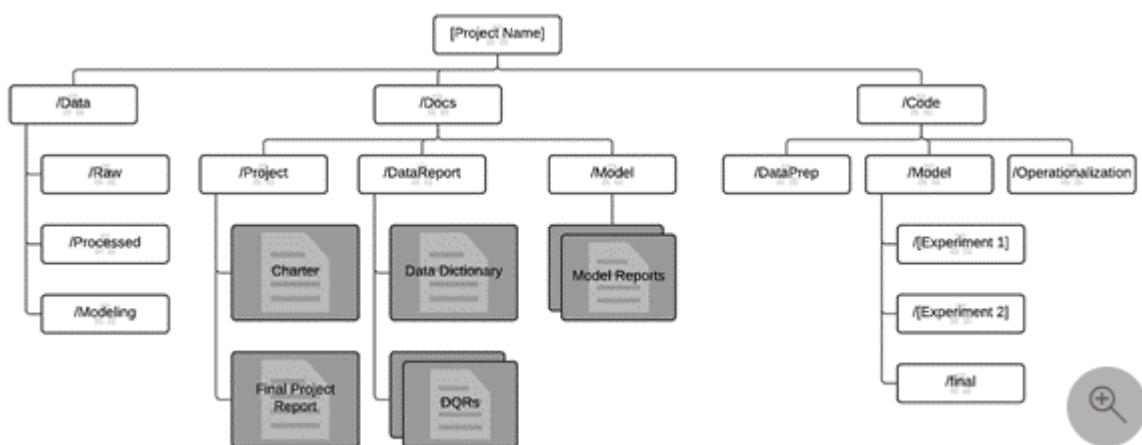


Figura 11. Ejemplo de proyecto esquema de ciencia de datos, basado en TDSP(marktab, 2020).

3.5. Preprocesamiento de imágenes

El procesamiento de imágenes según Wikipedia("Procesamiento digital de imágenes," 2022) se define como:

"El procesamiento de imágenes digitales es el conjunto de técnicas que se aplican a las imágenes digitales con el objetivo de mejorar la calidad o facilitar la búsqueda de información."

Debido a que las redes neuronales hacen uso de un conjunto de entradas(inputs) basados en el tamaño específico; las imágenes deben coincidir con el tamaño de entrada de la red. Luego es necesario ajustar el tamaño de entrada de estas, se puede cambiar la escala o recortar al tamaño requerido(*Preprocess Images for Deep Learning - MATLAB & Simulink - MathWorks España*, n.d.). Otra técnica relacionada con el objetivo de reducir el sobre aprendizaje(overfitting) implica aumentar la cantidad de datos de entrenamiento aplicando un aumento aleatorio a sus datos. Se pueden aplicar distorsiones, zoom, rotaciones o traslaciones aleatorias a las imágenes de entrada para que una red sea invariable a la presencia de este tipo de transformaciones. Existen operaciones más

avanzadas para procesar imágenes relacionadas con imágenes en 3D, donde se hace uso de modelos previos. Asimismo, puede procesar las imágenes previamente de acuerdo con su propia configuración mediante las funciones de transformación y combinación.(*Preprocess Images for Deep Learning - MATLAB & Simulink - MathWorks España*, n.d.).

3.5.1. Roboflow

Se trata de una herramienta en la nube que simplifica la creación de los conjuntos de datos necesarios para los problemas de visión artificial, específicamente en los modelos de aprendizaje profundo. Esta herramienta permite a los usuarios crear, copiar, modificar y etiquetar imágenes que serán utilizadas por aplicaciones de visión por computadora. Es útil para detectar y clasificar objetos. Adicionalmente, tiene herramientas y utilidades que permiten anotar objetos de manera automática con modelos pre-entrenados. Dentro de las funciones de la herramienta tenemos(*Overview - Roboflow*, n.d.):

- Etiquetado de imágenes.
- Preprocesamiento de conjuntos de datos.
- Fusionar proyectos/conjuntos de datos.
- Realizar separaciones automáticas en conjuntos de datos de entrenamiento, prueba y test.
- Exportación de conjunto de datos en múltiples formatos.
- Entrenamientos de modelos.

Esta herramienta facilita la importación y exportación de imágenes durante el etiquetado y el preprocesamiento. Estos formatos se corresponden con diferentes tipos de anotaciones muy utilizadas en problemas de visión por computadora para la detección y segmentación de imágenes. A continuación, se enumeran los tipos de formatos de entrada y salida.(*Overview - Roboflow*, n.d.):

- Formatos de imagen que incluyen JPG, PNG y BMP.
- Formatos de video que incluyen MOV y MP4.
- Formatos de anotación de detección de objetos, incluidos JSON, XML, CSV y TXT.
- Puede exportar el conjunto de datos anotado en el formato que desee. Incluidos YOLO V5, YOLO V7, COCO y otros.

Build Better Computer Vision Models Faster

Product Overview

Roboflow enables teams to create and improve computer vision models quickly and accurately without larger machine learning infrastructure teams — saving thousands of engineering hours.

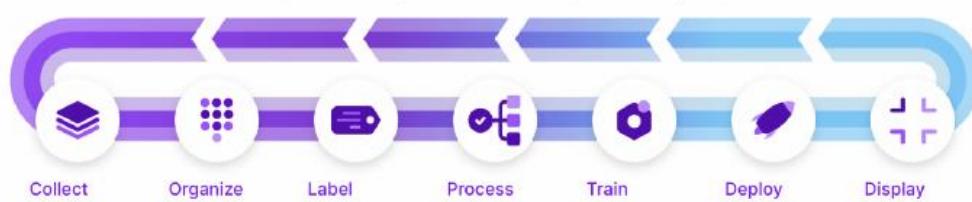


Figura 12. Pasos para construir, implementar y mejorar un modelo de visión artificial(*Overview - Roboflow*, n.d.).

3.6. Detección de Objetos

La detección de objetos es una técnica empleada en la visión por ordenador para la identificación y ubicación de objetos en una imagen o un vídeo. Este proceso consiste en identificar la posición correcta de uno o varios objetos mediante cuadros delimitadores alrededor de los objetos. Este proceso se distingue de otros métodos, tales como la clasificación de clase para una imagen o si existe un objeto dentro de estas que pueda corresponder a una clase. Los modelos de clasificación de imágenes clasifican objetos dentro de esta en una o varias categorías(*YOLO Object Detection Explained*, n.d.). La segmentación es la detección del objeto y la extracción de los bordes que corresponden a este objeto. En la figura se puede observar cómo se diferencian estas técnicas.

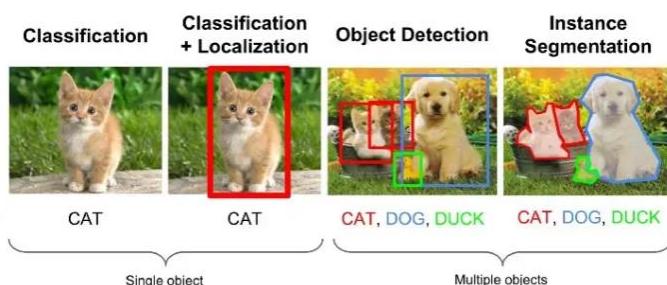


Figura 13. Ejemplos de clasificación en imágenes versus detección objetos y segmentación (*YOLO Object Detection Explained*, n.d.).

La detección de objetos en tiempo real es un componente clave de la visión por computadora, ya que es necesario en sistemas diseñados para aprender a través de imágenes o videos. Algunas de las aplicaciones son: seguimiento de múltiples objetos, conducción autónoma, robótica, análisis de imágenes médicas, etc. Existen miles de dispositivos que ejecutan la detección de objetos en tiempo real, entre los que se incluyen los procesadores de CPU y GPU, así como los modelos y procesos basados en redes neuronales desarrollados por los principales fabricantes de dispositivos de hardware y software para estos procesamientos, como son Apple, Intel, Nvidia, Google y otros(Boesch, 2022). Varios de estos dispositivos hacen uso de arquitecturas de borde que se enfocan en la aceleración de diferentes operaciones en redes neuronales como son, la convolución vainilla, en profundidad u operaciones MLP. En los últimos años, la detección de objetos en tiempo real ha permitido el desarrollo y evolución de diferentes dispositivos de borde. Existen métodos que se centran en mejorar la velocidad de inferencia de varias GPU y que centran su diseño en una arquitectura eficiente; para la CPU, su diseño se basa principalmente en MobileNet, ShuffleNet o GhostNet. Otra corriente son los detectores de objetos están desarrollados para GPU, estos utilizan principalmente ResNet, DarkNet o DLA, y CSPNet para optimizar la arquitectura(Boesch, 2022).

3.6.1. YOLO

YOLO, fue presentada por Joseph Redmon, al et, en la conferencia CVPR en el 2015. Este trabajo presenta un nuevo enfoque para la detección de objetos; hasta el momento se usaban clasificadores para llevar a cabo la detección; sin embargo, la novedad

surge debido a que la detección de objetos se presenta como un problema de regresión basado en cuadros delimitadores separados y el uso de probabilidades para clases asociadas. Una única red neuronal predice los cuadros delimitadores y las probabilidades de la clase a partir de imágenes. Esta fue presentada como una arquitectura unificada que es extremadamente rápida para su época; por tanto, se acuñó el término YOLO (you only look once); es decir “solo miras una vez”, este es un algoritmo de detección de objetos en tiempo real de última generación que hoy se presenta de manera general como un estándar para la detección de objetos(Redmon et al., 2016). En la siguiente figura se puede ver un ejemplo de detección de objetos utilizando YOLOv5 v7.0.



Figura 14. Ejemplo detección de objetos con YOLOv5 v7.0(Boesch, 2022).

Los autores enmarcan el problema de detección de objetos como un problema de regresión en lugar de una tarea de clasificación, separando espacialmente los cuadros delimitadores y asociando probabilidades a cada una de las imágenes detectadas usando una única red neuronal convolucional (CNN). Algunas de las razones por las que YOLO lidera la competencia incluyen(*YOLO Object Detection Explained*, n.d.):

- Velocidad.
- Precisión de detección.
- Buena generalización.
- Fuente abierta (Open source).

El algoritmo funciona según(Redmon et al., 2016):

- **Bloques residuales:** Se divide la imagen original (A) en celdas de cuadrícula NxN de igual forma. Cada celda de la cuadrícula es responsable de localizar y predecir la clase del objeto que cubre, junto con el valor de probabilidad/confianza.
- **Regresión de cuadro delimitador:** El siguiente paso es determinar los cuadros delimitadores que corresponden a rectángulos que resaltan todos los objetos de la imagen. Se puede tener tantos cuadros delimitadores como objetos haya dentro de una imagen dada. YOLO define los atributos de estos cuadros delimitadores utilizando un único módulo de regresión en el siguiente formato, donde se obtiene una representación vectorial final para cada cuadro delimitador.
- **Intersección sobre uniones(IOU):** En general, un objeto en una imagen puede tener varios candidatos de cuadro de cuadrícula para la predicción. El objetivo del IOU (un valor entre 0 y 1) es descartar dichos aquellos cuadros de cuadrícula que no son relevantes y conservar únicamente los de interés.

- **Supresión no máxima:** En este paso se trata de establecer un umbral para el IOU que no siempre es suficiente porque un objeto puede tener varios cuadros, y dejar todos estos puede incluir ruido. Aquí es donde podemos usar NMS para mantener solo las casillas con la puntuación de probabilidad más alta de detección. El propósito de la supresión no máxima es seleccionar el mejor cuadro delimitador para un objeto y rechazar o “suprimir” todos los demás cuadros delimitadores. Esta puntuación indica qué tan seguro está el modelo de que el objeto deseado está presente en este cuadro delimitador. NMS tiene en cuenta dos aspectos:
 - La puntuación de objetividad viene dada por el modelo.
 - La superposición o IOU de los cuadros delimitadores.

3.6.2. YOLOv5 v7.0

Esta es una versión desarrollada por la empresa Ultralytics cuya fecha de liberación fue en noviembre de 2022, el repositorio del proyecto está en el siguiente [enlace](#). En el pasado, en el entrenamiento de la red profunda, la asignación de etiquetas se refiere a “generar etiqueta de acuerdo con las reglas dadas”; sin embargo, en los últimos años, en la detección de objetos, los investigadores suelen utilizar la calidad y la distribución de la predicción en la salida de la red. Luego consideran varios métodos de cálculo y optimización para generar una etiqueta fiable. Por ejemplo, YOLO usa IOU de predicción como esquema fundamental para el etiquetado del objeto. La Arquitectura de este modelo se basa en las siguientes partes, según (Jocher, 2020):

- **Columna vertebral:** utilizada principalmente para extraer características importantes de la imagen de entrada dada. Las redes parciales CSP—Cross Stage se usan como columna vertebral para extraer características informativas de una imagen de entrada. CSPNet ha mostrado una mejora significativa en el tiempo de procesamiento con redes más profundas(Gutta, 2021).
- **Cuello:** se emplea principalmente para generar pirámides de características. Las pirámides de características ayudan a los modelos a generalizarse bien en la escala de objetos. Ayuda a identificar un mismo objeto con diferentes tamaños y escalas. Las pirámides de características son muy útiles y ayudan a los modelos a funcionar bien en datos no vistos. PANet se usa como modelo cuello para obtener pirámides de funciones(Gutta, 2021).
- **Cabeza:** sirve principalmente para realizar la parte de detección final. Se aplican cuadros de anclaje en características y generan vectores de salida final con probabilidades de clase, puntajes de objetividad y cuadros delimitadores. La cabeza es la misma que las versiones anteriores de YOLO V3 y V4(Gutta, 2021).
- **Función de activación:** La elección de las funciones de activación es crucial en cualquier red neuronal profunda. Para este modelo se utiliza la función de activación Leaky ReLU y Sigmoid. La función de activación de Leaky ReLU

se emplea en las capas intermedias/ocultas y la función de activación sigmoidea se usa en la capa de detección final(Gutta, 2021).

- **Función de optimización:** La función de optimización predeterminada para el entrenamiento es SGD. Sin embargo, puede cambiarlo a Adam empleando el argumento de línea de comando "—— adam"(Gutta, 2021).

YOLOv5 fue diseñado para que sea fácil de usar y aprender. Prioriza los resultados del mundo real y en esta versión supera a todas las versiones anteriores y ofrece un FPS más alto, según la siguiente Figura 15.

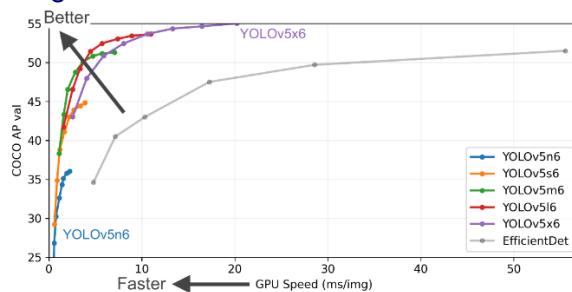


Figura 15. Resultados de la velocidad de Yolov5x6(Jocher, 2020)

3.7. Segmentación de objetos

La segmentación de objetos es uno de los métodos más nombrados durante este último año en sistemas de IA, es decir, cada vez se desarrollan más artículos y algoritmos relacionados. Según la Wikipedia("Segmentación (procesamiento de imágenes)," 2022) de manera particular define a la segmentación de imágenes, como:

"La segmentación de imágenes es el proceso de dividir una imagen digital en varios segmentos de imagen, también conocidos como regiones de imagen u objetos de imagen (conjuntos de píxeles). Más precisamente, la segmentación de imágenes es el proceso de asignar una etiqueta a cada píxel de una imagen de modo que los píxeles con la misma etiqueta comparten ciertas características."

La diferencia entre la segmentación y la detección radica en que el objetivo de la segunda es construir un recuadro delimitador correspondiente a cada clase dentro de una imagen, pero sin tener en cuenta la forma o los bordes del objeto. La segmentación de objetos crea máscaras de píxeles con la forma de cada objeto, por tanto, busca obtener detalles granulares sobre este(Vyas, 2020).

Los casos de uso de este tipo de redes son muy variados, según (Vyas, 2020) destacan:

- **Segmentación facial:** ayuda a los sistemas de visión por computadora a lograr tareas como el reconocimiento de expresiones, de edad, predicción del género o de la etnia de las personas. La segmentación semántica permite estas tareas al separar regiones de la cara en atributos importantes como la boca, el mentón, la nariz, los ojos y el cabello.
- **Imágenes médicas:** se utiliza en diagnósticos médicos, especialmente en la evaluación de análisis de rayos X y resonancias magnéticas. Permite

destacar y clasificar regiones relevantes de una imagen, haciendo que las pruebas de diagnóstico sean más fáciles y simples.

- **Autoconducción:** la conducción autónoma es una tarea extremadamente compleja que requiere percepción, análisis y cambios en tiempo real. La segmentación semántica se utiliza para identificar objetos como otros automóviles y señales de tráfico y regiones como carriles de carretera y aceras. La segmentación de instancias se emplea en la conducción autónoma, para la detección individual de coches, peatones, señales, etc.
- **Imágenes satelitales:** para analizar el uso de la tierra, las áreas sufridas por la deforestación, el análisis de las tierras agrícolas, etc.

3.7.1. Tipos de segmentación

La clasificación es la base de la visión artificial; pero en la actualidad muchos estudios promueven que no basta con clasificar las imágenes. Existen diferentes técnicas y métodos para segmentar, como la segmentación semántica, la segmentación de instancias y la detección de puntos clave. Estos algoritmos son tareas más difíciles y significativas. Comparado con la clasificación, la segmentación de objetos da una mayor comprensión del primer plano y el fondo de la imagen; estos algoritmos necesitan separar el objetivo de interés del fondo para determinar su forma, su categoría y ubicación(Wu et al., 2019).

- **Segmentación semántica:** es el proceso de asignar una etiqueta a cada píxel de la imagen, esta trata múltiples objetos de la misma clase como una sola entidad.
- **Segmentación de instancias:** la segmentación de instancias trata múltiples objetos de la misma clase como objetos individuales distintos (o instancias); esta es más difícil que la anterior.

En la siguiente Figura 16. Diferencias entre segmentaciones en cuanto a clase/semántica o de instancia(Ch. 6 - Object Detection and Segmentation, n.d.).se puede ver un ejemplo de las diferencias entre estos dos tipos de segmentación:

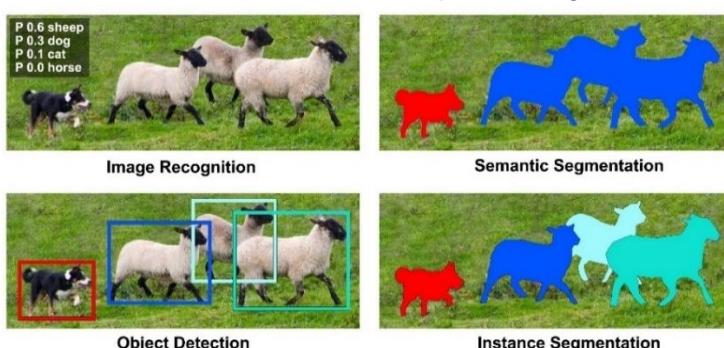


Figura 16. Diferencias entre segmentaciones en cuanto a clase/semántica o de instancia(Ch. 6 - Object Detection and Segmentation, n.d.).

3.7.2. Deep Learning aplicado en tareas de segmentación

Las redes neuronales convolucionales (CNN) están revolucionando muchos campos de la visión artificial. Existen diferentes arquitecturas aplicadas para la segmentación de la imagen, que se basan en el uso de CNN y redes profundas pre entrenadas. Uno de los principales inconvenientes de enfoques de redes profundas es su “hambre” de datos de entrenamiento(Caelles et al., 2017). Una de las propiedades más sorprendentes de las redes profundas es su aplicación para la segmentación entre instancia y su capacidad para transferirse a nuevas tareas (“aprendizaje de transferencia”). Una red que se entrenó previamente en un gran conjunto de datos como ImageNet o COCO, se puede ajustar con una cantidad relativamente menor de datos etiquetados para un nuevo conjunto de clases que son relevantes para nuestra aplicación particular. De hecho, a menudo se dice que las arquitecturas tienen una **“columna vertebral”** y una **“cabeza”**: para entrenar un nuevo conjunto de clases, a menudo es posible quitar la cabeza existente y reemplazarla con una nueva para las nuevas etiquetas(*Ch. 6 - Object Detection and Segmentation*, n.d.). La mayoría de los métodos de segmentación clásicos son dependientes de encontrar características particulares o granulares, difíciles de encontrar con métodos sencillos. Luego, el uso de las CNN permite el desarrollo e investigación en el campo de la detección de objetos. A continuación, se introducen varios algoritmos típicos de detección de objetos con la ayuda del aprendizaje profundo. En la actualidad, hay dos ideas principales en el campo de la detección de objetos basada en el aprendizaje profundo. Uno es el método de series RCNN basado en la propuesta de la región, y el otro es YOLO algoritmo de serie basado en la regresión(Wu et al., 2019), a continuación se presentan algunas redes basadas en arquitecturas CNN:

RCNN Regiones con características de CNN

El algoritmo clásico de detección de objetos utiliza un método de ventana deslizante para determinar áreas posibles, a su vez extrae una serie de candidatos más probables de regiones de antemano con el método de búsqueda selectiva, y luego las CNN basadas en las características de extracción se usan solo en estas regiones candidatas(Girshick et al., 2014).

El algoritmo se puede dividir en cuatro pasos:

1. **Generación de región candidata:** una imagen genera 1K ~ 2K regiones candidatas (método de búsqueda selectiva).
2. **Extracción de características:** uso de una red de convolución profunda para extraer características (CNN) para cada región candidata.
3. **Juicio de categoría:** la característica se envía a cada clase de clasificador SVM para distinguir si pertenece a la clase.
4. **Ubicación:** Refinamiento empleando la posición del cuadro candidato de corrección fina del regresor.

El inconveniente de este método es que RCNN tiene el problema del cálculo repetido. Es decir, puede haber miles de regiones en la propuesta, la mayoría de las cuales se superponen entre sí. Para resolver el problema, Kaiming Él et al. en el 2014, proponen la SPP-Net para mejorar el RCNN.

Fast-RCNN

Trabajo desarrollado por Girshick et al. en el 2015, combinado con SPP-net para mejorar el rendimiento de las redes RCNN. Este se emplea incluso para el reconocimiento de objetos en video. Específicamente, en Fast-RCNN, el autor puso regresión de cuadro delimitador en la red neuronal interna y lo clasificó en la región y se convirtió en una multitarea(Girshick, 2015).

Faster R-CNN

Es un modelo de detección de objetos basado en CNN de extremo a extremo. Los autores sugieren que las características de nivel convolucional en la red se pueden emplear para predecir la región dependiente de la categoría propuesta, y no es necesario realizar algoritmos como la búsqueda selectiva por adelantado. Este integra la extracción de propuesta de región y la parte Fast-RCNN en un modelo de red. Aunque la etapa de entrenamiento sigue siendo de varios pasos, la fase de detección es muy conveniente y rápida, y la tasa de precisión no es muy diferente del Fast-RCNN original. Faster R-CNN utiliza un detector de objetos de aprendizaje profundo de dos etapas: primero, identifica regiones de interés y luego pasa estas regiones a una red neuronal convolucional. Los mapas de características generados se pasan a una máquina de vectores de soporte (SVM) para su clasificación. Se calcula la regresión entre los cuadros delimitadores predichos y los cuadros delimitadores reales(Ren et al., 2016).

Máscara R-CNN

Máscara-RCNN un trabajo de Kaiming He, et al. en el 2017, es una red que permite la detección y segmentación paralela, lo que significa que los dos resultados se pueden obtener al mismo tiempo, pero a diferencia de otros métodos en donde la segmentación se realiza después de la clasificación(He et al., 2018). El marco general de Máscara-RCNN sigue siendo el marco de trabajo para Faster-RCNN. La red pasa de las dos originales tareas (clasificación+regresión) hacia tres tareas (clasificación+regresión+segmentación).

El algoritmo se puede dividir en dos etapas:

Etapa 1: Generar el área de caja de candidatos. Este proceso es el mismo que Faster R-CNN, que utiliza RPN (Red de propuesta regional).

Etapa 2: RoIPool se usa en el área del cuadro de candidatos para extraer características, clasificar y efectuar la regresión de cuadro de borde, y se genera una máscara de dos elementos para cada RoI.

Durante la detección, existen algunos errores relacionados con el cuadro delimitador dado y en el gráfico original, lo cual no afecta los resultados de la detección de clasificación. Pero en la imagen de nivel de píxel segmentado, tal error de ubicación espacial afectará seriamente los resultados de la segmentación. Por lo tanto, esta red propone el uso de la interpolación bilineal para resolver este problema, es decir, RoIAgn. La imagen se pasa a través de RoIAgn en lugar de RoIPool, lo que hace que el área del mapa de características seleccionado por RoIPool corresponda con mayor precisión al área de la imagen original(He et al., 2018).

Mask R-CNN, amplía Faster R-CNN al agregar una rama para predecir una máscara de objeto en paralelo con la rama existente para el reconocimiento del cuadro delimitador. Mask R-CNN es fácil de entrenar y agrega solo una pequeña sobrecarga a Faster R-CNN,

que se ejecuta a 5 fps. Además, Mask R-CNN es fácil de generalizar a otras tareas, por ejemplo, permitiéndonos estimar poses humanas en el mismo marco(He et al., 2018).

3.7.3. Detectron2

Detectron2 es un utilitario desarrollado por Facebook AI Research que proporciona algoritmos de detección y segmentación de última generación. Es el sucesor de Detectron y maskrcnn-benchmark y es compatible con una serie de proyectos de investigación de visión artificial y aplicaciones. Se puede decir que tiene varios modelos pre-entrenados para la detección de objetos, segmentación de instancias, segmentación panóptica, segmentación semántica y detección de puntos clave(Facebookresearch/Detectron2, 2019/2022). Este consta de:

- Más de 80 modelos pre-entrenados.
- Soporte de conjuntos de datos para conjuntos de datos de visión populares como COCO, Cityscapes, LVIS, PASCAL VOC, ADE20k.

La siguiente Tabla 4, muestra los algoritmos de detección de objetos, que son parte de esta herramienta:

Modelo	Entrenado	Métrica Box AP	~Flops	Publicación	Año
TensorMask	COCO	41.4	504 Billion	ICCV 2019 · Xinlei Chen, Ross Girshick, Kaiming He, Piotr Dollár	2019
TridentNet	COCO	43.6	888 Billion	ICCV 2019 · Yanghao Li, Yuntao Chen, Naiyan Wang, Zhao-Xiang Zhang .	2019
RetinaNet	COCO	40.4	273 Billion	ICCV 2017 · Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, Piotr Dollár	2017
Mask R-CNN	COCO	44.3		ICCV 2017 · Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick	2017
Faster R-CNN	COCO	43	406 Billion	NeurIPS 2015 · Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun	2015
Fast R-CNN	COCO	37.8		ICCV 2015 · Ross Girshick	2015

Tabla 4. Redes incluidas en Detectron2 para la detección y segmentación de objetos.

3.7.4. Segmentación personas (BodyPix)

Como se ha mencionado, la segmentación de imágenes se refiere a la técnica de agrupar píxeles en una imagen en áreas semánticas, típicamente con el objetivo de ubicar objetos y límites. El modelo BodyPix es capaz de aplicar este concepto para una persona y veinticuatro partes del cuerpo, por ejemplo, mano izquierda, parte delantera inferior de la pierna derecha o parte posterior del torso, y etc, ver Figura 17. Ejemplo de BodyPix v2, detección de múltiples personas. En otras palabras, BodyPix puede clasificar los píxeles de una imagen en dos categorías(Dan Oved, 2019):

- a) Píxeles que representan a una persona

b) Píxeles que representan el fondo.

Esta tecnología puede ser una herramienta para una amplia gama de aplicaciones, desde realidad aumentada hasta edición de fotografías y efectos artísticos en imágenes o videos. BodyPix es una herramienta de fácil acceso para artistas, codificadores creativos y aquellos que son nuevos en la programación(Dan Oved, 2019).

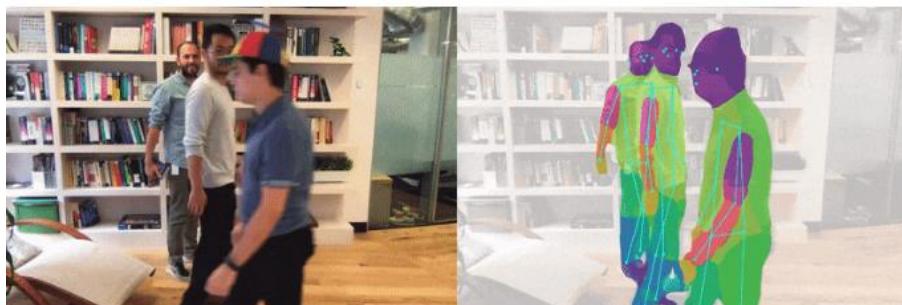


Figura 17. Ejemplo de BodyPix v2, detección de múltiples personas.

3.8. Super Resolución en Imágenes (SR)

Existen varias aplicaciones que ofrecen sus productos en línea, pero como usuarios preferiríamos tener una experiencia lo más cercana posible a una interacción real entre cliente y vendedor. En general, las imágenes grandes y nítidas pueden guiar mejor a un usuario durante el proceso de compra en línea, ya que esto le permite realizar una inspección detallada del producto y mejora la sensación general de una compra. Sin embargo, muchas de las imágenes alojadas en diferentes repositorios de contenido (CDN), suelen ser imágenes de baja calidad, principalmente debido a que por temas de rendimiento y latencia de red, su distribución con miles o millones de usuarios en línea permite una interacción más rápida entre el cliente y lo que está comprando(Cardinale, 2019). En [idealode](#); sitio web de comparación de precios líder en Europa; ofrece una plataforma atractiva y fácil de usar que muestra una serie de productos catalogados e identificados de manera simple mediante imágenes de alta resolución (HR). Por otro lado, no todas las tiendas, o servicios en la nube pueden proporcionar plataformas con imágenes buenas y de alta calidad para sus productos: a menudo son demasiado pequeñas, con baja resolución (LR) o de baja calidad(Cardinale, 2019).

Muchos dispositivos modernos, como teléfonos inteligentes, drones, vehículos y otros dispositivos de Internet de las cosas (IoT), están equipados con cámaras de alta calidad que pueden capturar imágenes y videos de alta resolución. Las resoluciones pueden aumentarse a miles de millones de píxeles (a menudo llamados giga píxeles), como se muestra en la Figura 18, A la derecha una imagen con superresolución y a la izquierda un método tradicional (bicubic upscaling). Las tareas de aprendizaje profundo dependen de los datos capturados y su entrenamiento se comporta de manera diferente en comparación con el ojo humano. En el aprendizaje automático, las redes se entrena en función de los detalles específicos; cuanta más información se recopile, mejor. Por lo tanto, el éxito en el uso de investigaciones de alta resolución depende de esta tarea en concreto. El uso de imágenes y videos de alta resolución (HR) como entradas para modelos de aprendizaje profundo crea desafíos en las fases de entrenamiento e inferencia. A excepción de las redes totalmente

convolucionales (FCN), el número de parámetros en los modelos de aprendizaje profundo suele aumentar a medida que los tamaños de entrada se incrementan.(Alzubaidi et al., 2021).

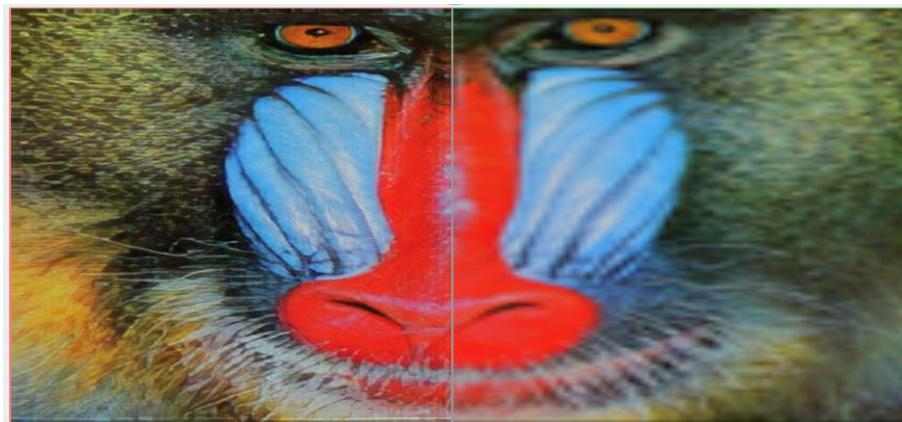


Figura 18, A la derecha una imagen con superresolución y a la izquierda un método tradicional (bicubic upscaling)

El empleo de estas imágenes influye en el tiempo de inferencia/entrenamiento, así como el consumo de GPU, consumo de memoria y disco. Es probable que los problemas mencionados persistan incluso con los avances de hardware en tecnología(Pandey et al., 2022). Si se procesa una imagen con alta resolución, conduce a mejoras en función del problema a resolver. No es probable que aumentar la resolución de las imágenes de objetos o animales para giga píxeles revele más detalles beneficiosos y mejore la exactitud. De igual modo, se ha demostrado que el uso de resoluciones más altas en histopatología puede conducir a resultados mejores. Un campo particular de aplicación de estos métodos son las imágenes médicas. Por ejemplo, el procesamiento de tomografías computarizadas de alta resolución con aprendizaje profundo es cada vez más frecuente. Estudios en COVID-19 en tomografías computarizadas de alta resolución del pulmón, con el empleo de aprendizaje profundo para mejorar la calidad de los datos capturados en ultraalta resolución, ha demostrado obtener resultados similares a otros métodos convencionales con velocidad clínicamente factible(Bakhtiarnia et al., 2022).

3.8.1. RDN/RRDN Red densa residual para superresolución de imágenes

Para abordar este problema, implementamos y entrenamos una red neuronal convolucional (CNN) de última generación basada en el artículo de investigación de 2018 Residual Dense Network (RDN) for Image Super-Resolution (Zhang et al.). El objetivo es conceptualmente simple, tomar imágenes pequeñas y, al igual que con una lupa real, obtener una versión más grande de ellas. Una red neuronal convolucional muy profunda (CNN) ha logrado un gran éxito para la superresolución de imágenes y también ofrece características jerárquicas. Sin embargo, la mayoría de los modelos SR basados en CNN profundos no hacen uso de las características jerárquicas de las imágenes originales de baja resolución (LR). Yulun Zhang, et al, proponen una red la cual aprovecha al máximo las

características jerárquicas de todas las capas convolucionales. Específicamente, de bloques densos residuales (RDB) para extraer abundantes características locales a través de capas convolucionales densas conectadas. RDB además permite conexiones directas desde el estado de RDB anterior a todas las capas de RDB actual, lo que lleva a un mecanismo de memoria contigua (CM). La fusión de características locales en RDB se utiliza luego para aprender adaptativamente características más efectivas de las características locales anteriores, actuales y estabiliza el entrenamiento de una red más amplia. Después de obtener completamente las características locales densas, hacen empleo de la fusión de características globales para aprender de manera conjunta y adaptativa las características jerárquicas globales de una manera holística(Zhang et al., 2018).

Arquitectura de red RDN (Figura 19)

Según Zhang et al., los principales parámetros de la estructura de la arquitectura son:

- D - número de Bloques Densos Residuales (RDB)
- C - número de capas convolucionales apiladas dentro de una RDB
- G - número de mapas de características de cada capa convolucional dentro de los RDB
- G0: número de mapas de características para convoluciones fuera de RDB y de cada salida de RBD

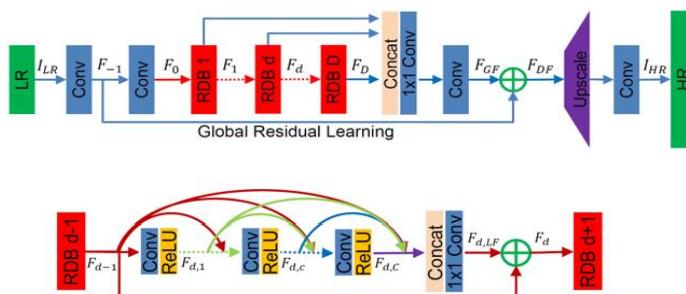


Figura 19. Red densa residual para superresolución de imágenes(Leverxgroup/Esgan: Enhanced SRGAN. Champion PIRM Challenge on Perceptual Super-Resolution, n.d.)

La Super-Resolution Generative Adversarial Network (SRGAN) es un trabajo que es capaz de generar texturas realistas durante la superresolución de una imagen, utilizando redes antagónicas generativas. Para mejorar aún más la calidad visual, se realizó una investigación de tres componentes clave de SRGAN: arquitectura de red, pérdida de adversario y pérdida de percepción, y los autores Xintao Wang, et al, ofrecen mejor a cada uno de estos que derivo en un SRGAN mejorado (ESRGAN). En particular, Residual-in-Residual Dense Block (RRDB) sin normalización por lotes como la unidad básica de creación de redes, usa una red GAN relativista para permitir que el discriminador prediga una realidad relativa en lugar del valor absoluto. Finalmente, ofrecieron una mejora a la función de pérdida de percepción al emplear las funciones antes de la activación, beneficiándose de estas mejoras, ESRGAN logra una mejor calidad visual con texturas más realistas y naturales que SRGAN y ganó el primer sitio en el Desafío PIRM2018-SR(Wang et al., 2018).

Arquitectura de red RRDN (Figura 20)

Los principales parámetros de la estructura de la arquitectura son:

- T - número de Residual en Bloques Residuales Densos (RRDB)
- D - número de bloques densos residuales (RDB) dentro de cada RRDB
- C - número de capas convolucionales apiladas dentro de una RDB
- G - número de mapas de características de cada capa convolucional dentro de los RDB
- G0: número de mapas de características para circunvoluciones fuera de RDB y de cada RBD

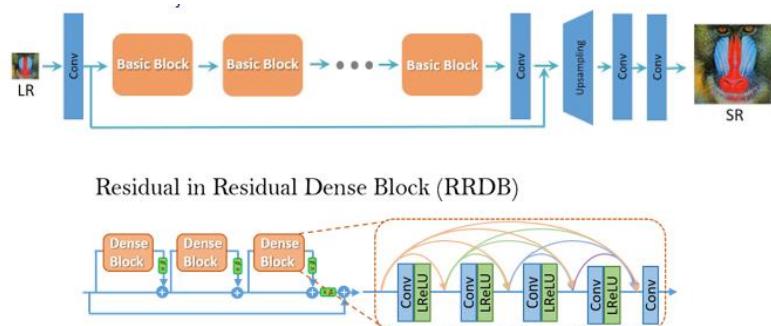


Figura 20. ESRGAN: redes antagónicas generativas de superresolución mejoradas (Leverxgroup/Esgan: Enhanced SRGAN. Champion PIRM Challenge on Perceptual Super-Resolution, n.d.)

4. Resultados

4.1. Recursos

Una de las principales ventajas del uso del aprendizaje profundo es el poder contar con dispositivos especializados para la ejecución de diversas tareas de procesamiento de datos y entrenamiento de modelos. A continuación, se presentan los principales componentes del hardware utilizado en el desarrollo del proyecto.

GPU(40GB):						
+-----+ NVIDIA-SMI 460.32.03 Driver Version: 460.32.03 CUDA Version: 11.2 +-----+						
GPU Name	Persistence-M	Bus-Id	Disp.A	Volatile Uncorr. ECC		
Fan	Temp	Perf	Pwr:Usage/Cap	Memory-Usage	GPU-Util	Compute M.
						MIG M.
0 A100-SXM4-40GB	Off	00000000:00:04.0	Off		0	
N/A 33C	P0	52W / 400W	0MiB / 40536MiB	0%	Default	Disabled

		RAM(89Gigas)	
--	--	--------------	--

CPU(12 cores)		Memory Usage					
Architecture:	x86_64						
CPU op-mode(s):	32-bit, 64-bit						
Byte Order:	Little Endian						
CPU(s):	12						
On-line CPU(s) list:	0-11						
Thread(s) per core:	2						
Core(s) per socket:	6						
Socket(s):	1						
NUMA node(s):	1						
Vendor ID:	GenuineIntel						
CPU family:	6						
Model:	85						
Model name:	Intel(R) Xeon(R)						
Stepping:	7						
CPU MHz:	2200.218						
BogoMIPS:	4400.43						
Hypervisor vendor:	KVM						
Virtualization type:	full						
L1d cache:	32K						
L1i cache:	32K						
L2 cache:	1024K						
L3 cache:	39424K						
NUMA node0 CPU(s):	0-11						
DISCO (164GB)							
Filesystem	Size	Used	Avail	Use%	Mounted on		
overlay	167G	23G	144G	14%	/		
tmpfs	64M	0	64M	0%	/dev		
shm	41G	0	41G	0%	/dev/shm		
/dev/root	2.0G	1.1G	910M	54%	/sbin/docker-init		
/dev/sda1	174G	25G	149G	15%	/opt/bin/.nvidia		
tmpfs	42G	48K	42G	1%	/var/colab		
tmpfs	42G	0	42G	0%	/proc/acpi		
tmpfs	42G	0	42G	0%	/proc/scsi		
tmpfs	42G	0	42G	0%	/sys/firmware		

4.2. Organización de proyecto y licenciamiento

El proyecto tiene una licencia MIT License Open Source, que permite el uso, modificación y redistribución del código tal como se encuentra en el repositorio git, sin ningún tipo de responsabilidad por daños y perjuicios. Esta es una licencia de código abierto, que otorga a cualquiera la libertad de usar el código y mejorarlo.

El repositorio de código fuente se encuentra en la siguiente URL:

☞ [gan-segmentation-person-gan](#)

data: Ruta donde se encuentran las diferentes imágenes procesadas utilizadas como recursos para el entrenamiento de los modelos basados en redes generativas.

model: Ruta donde se encuentran los diferentes notebooks, empleados para la generación de modelos tanto para la detección de objetos y la segmentación de imágenes.

Los resultados de los modelos o pesos pre-entrenados se encuentran en una carpeta en drive, llamada **results**; en esta carpeta se encuentran los diferentes resultados de los modelos de entrenamiento realizados en las diferentes etapas del proyecto.

 [drive-modelos-pesos/results](#)

4.3. Conjuntos de datos

Uno de los objetivos de este trabajo es la creación e implementación de conjuntos de datos de entrenamiento; generalmente, estos serán imágenes, que son un insumo necesario para la implementación de la red neuronal basada en redes GAN. Hay que señalar que en el procesamiento de imágenes se deben seguir una serie de etapas de pre y post procesamiento que permitan ajustar nuestros datos al objetivo propuesto. En este proyecto se usan herramientas en la nube para simplificar los procesos de etiquetado de datos y

permitir que se opere sobre conjuntos de datos ya entrenados que se hayan publicado o que sean parte de trabajos previos.

4.3.1. Personas y Armas

Conjuntos de datos de imágenes:

Tipo Conjunto	Fuente
Persona armada	<pre>@misc{ gunman2_dataset, title = { gunman2 Dataset }, type = { Open Source Dataset }, author = { personal work }, howpublished = { \url{ https://universe.roboflow.com/personal-work-atipa/gunman2 } }, url = { https://universe.roboflow.com/personal-work-atipa/gunman2 }, journal = { Roboflow Universe }, publisher = { Roboflow }, year = { 2022 }, month = { may }, note = { visited on 2022-12-13 }, }</pre>
Persona armada	<pre>@misc{ gun-detection-cctv-br07f_dataset, title = { Gun Detection CCTV Dataset }, type = { Open Source Dataset }, author = { sc19sjjd@leeds.ac.uk }, howpublished = { \url{ https://universe.roboflow.com/sc19sjjd-leeds-ac-uk/gun-detection-cctv-br07f } }, url = { https://universe.roboflow.com/sc19sjjd-leeds-ac-uk/gun-detection-cctv-br07f }, journal = { Roboflow Universe }, publisher = { Roboflow }, year = { 2022 }, month = { apr }, note = { visited on 2022-12-13 }, }</pre>
Persona armada	<pre>@misc{ pistol-y5ku1_dataset, title = { pistol Dataset }, type = { Open Source Dataset }, author = { AbhishekParth }, howpublished = { \url{ https://universe.roboflow.com/abhishekparth/pistol-y5ku1 } }, url = { https://universe.roboflow.com/abhishekparth/pistol-y5ku1 }, journal = { Roboflow Universe }, publisher = { Roboflow }, year = { 2022 }, }</pre>

```

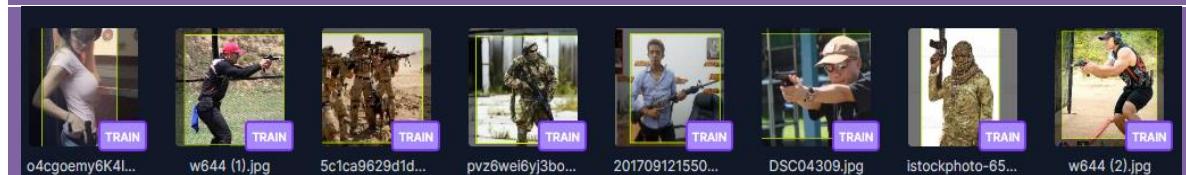
month = { jan },
note = { visited on 2022-12-13 },
}

```

Tabla 5. Conjuntos de datos personas armadas.

Se presentan algunos ejemplos y características de estos conjuntos de datos.

Gunman2



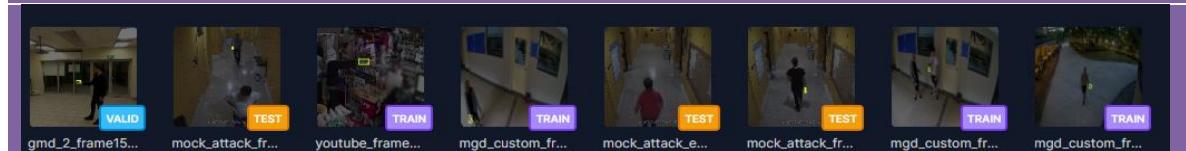
Nro. Imágenes 500

Formato YOLO v5/COCO JS

Versión 01

Autor NA(Atipa)

Gun Detection CCTV Dataset



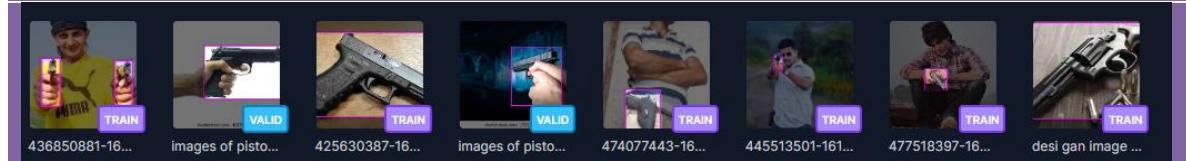
Nro. Imágenes 3367

Formato YOLO v5/COCO JS

Versión 01

Autor Stuart Dingwell

Pistol Dataset



Nro. Imágenes	455
Formato	YOLO v5/COCO JS
Versión	01
Autor	AbhishekParth

Tabla 6. Descripción y características conjunto de datos personas armadas

Para las tareas de procesamiento de imágenes se realizó el notebook que se encuentra en el repositorio:

☞ [1TFMMergeDataSetImagenes.ipynb](#)

Luego, se realiza la carga de los resultados procesados en Roboflow, en donde se crea un nuevo conjunto de datos.

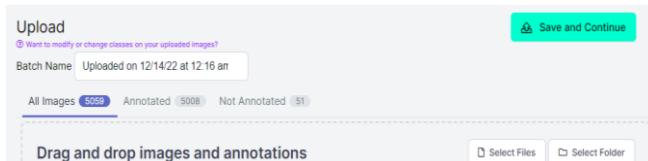


Figura 21. Ejemplo carga imágenes en Roboflow.

Se efectúa el particionamiento del conjunto de datos en entrenamiento, test y validación.

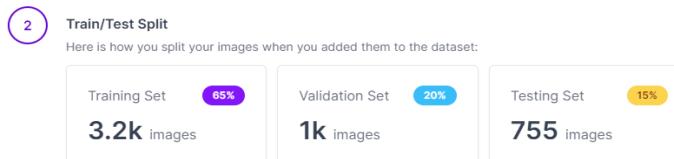


Figura 22. Particionamiento del conjunto de datos

Se produce una etiqueta que identifique de manera única a la clase.



Figura 23. Identificación de máscara única

Una vez finalizados se procede a crear una primera versión del conjunto de datos.

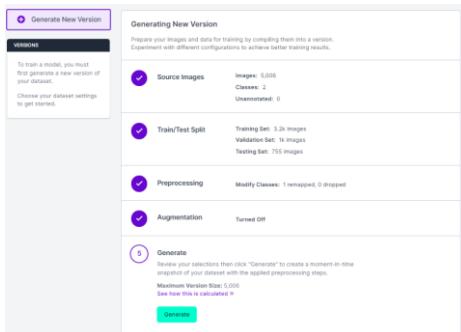


Figura 24. Creación primera versión del conjunto de datos.

Como resultado se obtiene un nuevo conjunto de datos de prueba, el cual será preparado y validado luego con otras herramientas para la detección de personas y armas.

Tipo Conjunto	Fuente
Persona armada	<pre>@misc{ datasetgunori_dataset, title = { DataSetGunOri Dataset }, type = { Open Source Dataset }, author = { Miguel Alejandro Ponce Proaño }, howpublished = { \url{ https://universe.roboflow.com/miguel-alejandro-ponce-proano/datasetgunori } }, url = { https://universe.roboflow.com/miguel-alejandro-ponce-proano/datasetgunori }, journal = { Roboflow Universe }, publisher = { Roboflow }, year = { 2022 }, month = { dec }, note = { visited on 2022-12-14 }, }</pre>

Tabla 7. Conjunto de datos resultado persona armada.

DataSetGunOri	
	VALID mgd_custom_fr...
	TRAIN mgd_custom_fr...
	TRAIN mock_attack_fr...
	VALID mock_attack_fr...
	TRAIN mock_attack_fr...
	TRAIN mgd_custom_fr...
	VALID mgd_varying_fr...
	TRAIN gmd_1_frame85...
Nro. Imágenes	5006
Formato	YOLO v5/COCO JS
Versión	01
Autor	Miguel P.

Tabla 8. Características conjunto de datos resultado persona armada.

4.3.2. Armas cortas(pistola)

Este conjunto de datos será la fuente para la detección de objetos de tipo caja (bounding box); en donde se utiliza un recuadro delimitador para describir la ubicación espacial de un objeto, sobre los conjuntos de imágenes. En la figura se puede ver un ejemplo de las anotaciones de este conjunto de datos, donde se han incluido etiquetas para identificar un recuadro alrededor del arma (objeto a identificar).

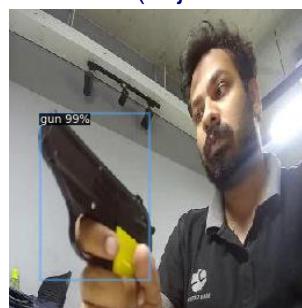


Figura 25. Ejemplo recuadro detección arma.

Conjuntos de datos de imágenes:

Tipo Conjunto	Fuente
Armas cortas/ personas	<pre>@misc{ gun-dataset_dataset, title = { Gun dataset Dataset }, type = { Open Source Dataset }, author = { workspace }, howpublished = { \url{ https://universe.roboflow.com/workspace-zqssx/gun-dataset } }, url = { https://universe.roboflow.com/workspace-zqssx/gun-dataset }, journal = { Roboflow Universe }, publisher = { Roboflow }, year = { 2022 }, month = { apr }, note = { visited on 2022-12-18 }, }</pre>

Tabla 9. Conjunto de datos para detección de armas.

Se presentan algunos ejemplos y características de este conjunto de datos.

Gun dataset							
Nro. Imágenes							5261

Reajuste Tamaño	416x416
Formato	COCO JS
Versión	12
Autor	NA

Tabla 10. Características del conjunto de datos para la detección del arma.

El conjunto de datos se encuentra dividido de la siguiente manera:

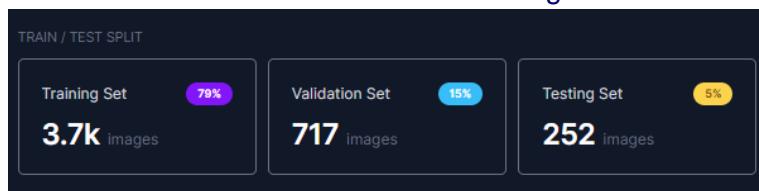


Figura 26. División del conjunto de datos para la detección del arma.

4.3.3. Segmentación armas cortas

Este conjunto de datos será la fuente para la segmentación de objetos tipo pistola; se emplearán delimitadores alrededor de la forma del arma y mano para describir el objeto segmentado y su ubicación espacial. Para aumentar la cantidad de imágenes segmentadas, se propone integrar dos fuentes en un único conjunto de datos. El resultado final se usará más adelante para crear los modelos de segmentación del arma. En la siguiente figura se muestra un ejemplo de las anotaciones de este conjunto de datos, en el que se ha realizado la segmentación del objeto arma resaltado en color verde (objeto a identificar).



Figura 27. Ejemplo de segmentación de armas cortas.

Conjuntos de datos de imágenes segmentadas:

Tipo Conjunto	Fuente
Segmentación modelos 3D armas	@misc{ convert-pafmj_dataset, title = { convert Dataset }, type = { Open Source Dataset }, author = { 3d }, howpublished = { \url{ https://universe.roboflow.com/3d/convert-pafmj } }, url = { \url{https://universe.roboflow.com/3d/convert-pafmj} }, journal = { Roboflow Universe }, publisher = { Roboflow },

	<pre>year = { 2022 }, month = { dec }, note = { visited on 2022-12-18 }, }</pre>
Segmentación mano-arma	<pre>@misc{ firearms-detection-8b36u_dataset, title = { firearms detection Dataset }, type = { Open Source Dataset }, author = { Thesis }, howpublished = { \url{ https://universe.roboflow.com/thesis-iohre/firearms-detection-8b36u } }, url = { https://universe.roboflow.com/thesis-iohre/firearms-detection-8b36u }, journal = { Roboflow Universe }, publisher = { Roboflow }, year = { 2022 }, month = { jul }, note = { visited on 2022-12-18 }, }</pre>

Tabla 11. Conjuntos de datos para la segmentación de armas cortas.

Debido a que ambos conjuntos están en los repositorios públicos de la herramienta Roboflow; las utilidades de la aplicación permitieron de forma transparente y simple realizar la integración en un solo repositorio.

Conjuntos de datos de imágenes segmentadas:

Tipo Conjunto	Fuente
Segmentación mano- arma/arma	<pre>@misc{ guns-segmentation_dataset, title = { Guns Segmentation Dataset }, type = { Open Source Dataset }, author = { Miguel Alejandro Ponce Proaño }, howpublished = { \url{ https://universe.roboflow.com/miguel-alejandro-ponce-proano/guns-segmentation } }, url = { https://universe.roboflow.com/miguel-alejandro-ponce-proano/guns-segmentation }, journal = { Roboflow Universe }, publisher = { Roboflow }, year = { 2022 }, month = { dec }, note = { visited on 2022-12-18 }, }</pre>

Tabla 12. Resultado integración conjuntos de datos armas cortas.

Se presentan algunos ejemplos y características de estos conjuntos de datos.

Guns Segmentation



Nro. Imágenes	1435
Reajuste Tamaño	240x240
Formato	YOLOv7/COCO JS
Versión	5
Autor	Miguel P.

Tabla 13. Características del conjunto de datos para la segmentación de armas cortas.

El conjunto de datos se encuentra dividido de la siguiente manera:

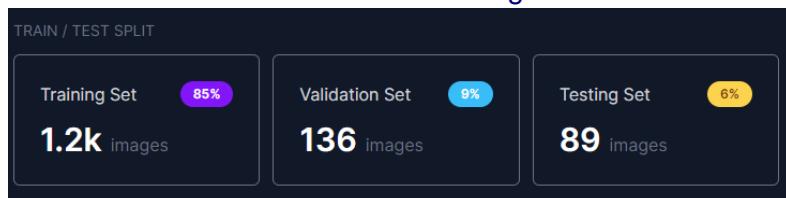


Figura 28 División del conjunto de datos para la segmentación de armas cortas.

4.4. Modelos detección objetos

A continuación, se presentan los diferentes modelos de Deep Learning, utilizados en la detección de objetos; ya que como objetivo de este trabajo se pretende detectar una persona armada y posteriormente segmentarla. Esto corresponde a la identificación de los objetos de interés en una imagen, en particular la detección del arma.

4.4.1. Detección de armas cortas

En esta sección se utiliza la fuente de datos de **Armas Cortas**, descrita en la sección anterior. Para la detección de armas de fuego, se empleará un modelo pre_entrenado desarrollado por Ultralytics, que obtiene valores altos en relación con la métrica mAP para la detección de objetos. Para mejorar la precisión de la red, se usa el modelo pre entrenado y, mediante el conjunto de datos, se realiza un nuevo entrenamiento para la red. Una vez completado, se guarda una copia del modelo en la nube, que se usará más adelante en el siguiente proceso de detección de armas y personas. Después de efectuar varias pruebas se ha determinado que un entrenamiento con 200 épocas(epochs) es adecuado porque al evaluar sus métricas se obtienen valores idóneos para esta red. Esta herramienta debido a su versatilidad permite el entrenamiento de un modelo para la detección de armas con una sola línea de código.

```
!python train.py --img 240 --batch 16 --epochs 200 --data
{dataset.location}/data.yaml --weights yolov5s.pt --cache
```

Para esta tarea se realizó el notebook que se encuentra en el repositorio Git:

○ [2TFMModeloDeteccionArmasCortasYOLOv5.ipynb](#)

Resultados luego de entrenar durante 200 épocas:

```
200 epochs completed in 1.212 hours.
Optimizer stripped from runs/train/exp/weights/last.pt, 14.3MB
Optimizer stripped from runs/train/exp/weights/best.pt, 14.3MB

Validating runs/train/exp/weights/best.pt...
Fusing layers...
Model summary: 157 layers, 7012822 parameters, 0 gradients, 15.8 GFLOPS
      Class   Images  Instances       P       R    mAP50  mAP50.95: 100% 23/23 [00:04<00:00,  4.81it/s]
        all     717     765  0.956  0.908  0.957  0.716
Results saved to runs/train/exp
```

A continuación, se muestra las gráficas de la métrica mAP, para la detección de armas:



Figura 29. Resultados de la red utilizada en la detección de armas cortas.

La métrica IOU (Intersección sobre Unión), 0.957, es el promedio de los AP calculados con un umbral 0.5; lo cual significa que los recuadros de detección de objetos para este modelo se superponen entre sí. Además, según se muestra en la gráfica de precisión y ajuste, el modelo se reajusta hasta alcanzar una precisión del 0.955, punto en el que ya no existe una mejora significativa del modelo.

Algunos resultados son:



Figura 30. Ejemplos y resultados de la red empleada en la detección de armas cortas.

4.4.2. Detección personas con armas cortas

En esta sección se realizarán dos procesos:

1. Detectar a la persona.

2. Determinar si la persona está armada.

Para ambos casos debido a que existen modelos previamente pre entrenados:

- Para detectar las personas se hará uso de los pesos ofrecidos por YOLOv5.
- Para el arma los pesos desarrollados en el apartado anterior.

Para esta tarea se realizó el notebook que se encuentra en el repositorio:

- [3TFMProcesoDeteccionPersonasConArmasCortasYOLOv5.ipynb](#)

Basado en el conjunto de datos COCO 128, elaborado en el 2017. YOLOv5 puede detectar hasta 80 clases de objetos, por ejemplo: persona, pájaro, gato, vaca, perro, caballo, oveja, avión, bicicleta, barco, autobús, y etcétera. Para la detección de personas se hará uso de un umbral de confidencia (--conf) mayor a 0.80, un valor alto debido a que más adelante se realizarán tareas de segmentación; donde, una mayor exactitud facilita obtener la silueta de cada persona. Esta herramienta debido a su versatilidad permite la detección de una persona con una sola línea de código.

```
!python detect.py --weights yolov5s.pt --img 240 --conf 0.80 --  
project detect-humans-train --classes 0 --save-crop --source  
/content/yolov5/DataSetGunOri-1/train/images
```

Con el resultado del proceso anterior que se encuentra bajo el proyecto detect-humans/.../crops , se procede a detectar a aquellas personas que tienen armas cortas, para esto se ejecuta la siguiente línea de código:

```
!python detect.py --weights yolov5sArmaCortas.pt --img 240 --conf  
0.80 --project detect-humans-guns --classes 0 --save-crop --source  
. /personas-detectadas
```

En la siguiente figura se muestran los resultados de este proceso:



Figura 31. Ejemplos y resultados de la red utilizada en la detección de personas con armas cortas.

Finalmente, se obtiene un nuevo conjunto de datos el cual corresponde a todas las personas con armas cortas, que fueron detectadas por los modelos pre-entrenados.

4.4.3. Conversión de imágenes hacia resolución en alta definición (HR) con Super Pixel

Para el desarrollo de esta actividad se hará uso del trabajo del Francesco Cardinale, y Dat Tran, que se encuentra en el proyecto [github](#), que contiene una serie de clases relacionadas con la implementación de Image Super-Resolution (ISR). Los pesos de la red RDN entrenada en el conjunto de datos están disponibles en la nube en formato hdf5.

El conjunto de datos utilizados será el que se elaboró en el paso anterior [personas-armadas-detectadas.tar.gz.](#), la ejecución de este proceso debido a las características y complejidad puede tomar varias horas e incluso días. Una vez finalizado este proceso será

usado como insumo para las redes de segmentación de personas y armas cortas, que veremos más adelante.

- ✓ Para tratar las imágenes con la red RDN, los parámetros empleados para este modelo son C=3, D=10, G=64, G0=64 como parámetros ([ver arquitectura para más detalles](#)).
- ✓ Para tratar las imágenes con la red ESRGAN, los parámetros utilizados para este modelo son C=4, D=3, G=32, G0=32 como parámetros ([ver arquitectura para más detalles](#)).

Para esto se desarrolló el siguiente notebook, en este se implementan dos clases tanto para la generación de super imágenes con RDN(SuperResolutionRDN) y con ESRGAN(SuperResolutionRRDN):

☞ [4TFMProcesoConversionImagenesAltaDefinicion.ipynb](#)

En la siguiente tabla se puede evidenciar los resultados de esta tarea.

Imagen real: 389x412	Imagen HR RDN: 778x824	Imagen HR ESRGAN: 1156 x 1648
		

Tabla 14. Resultados luego de realizar conversión con superpixeles.

4.4.4. Detección y segmentación de armas cortas

Para el desarrollo de esta actividad se hará uso de la herramienta Detectronv2, en donde se realiza primero el modelo para la detección de los objetos tipo arma corta y luego la segmentación del arma.

Paso 1: Se utiliza la red Faster R-CNN para la detección del arma, la cual se considera un marco general para la detección de objetos de última generación. A diferencia de YOLOv5 v7.0, es un poco más lento para la inferencia, pero a cambio ofrece una mayor precisión. Se usa el API de detección de objetos de TensorFlow proporcionada de forma predeterminada, que incluye pesos pre-entrenados, y la implementación en este cuaderno estará vinculada con Tensorboard.

Para esta tarea se realizó el notebook que se encuentra en el repositorio:

☞ [5_1TFMModeloDeteccionArmasCortasDetectronv2.ipynb](#)

Los parámetros empleados para este modelo son NUM_WORKERS=2, IMS_PER_BATCH=2, BASE_LR= 0.00025, MAX_ITER(épocas)=800, BATCH_SIZE_PER_IMAGE=128 como parámetros (ver arquitectura para más detalles). Se realizó diferentes pruebas donde la red que entreno durante 800 épocas ofreció los siguientes resultados:

```

iter: 599 total_loss: 0.5259 loss_cls: 0.1507 loss_box_reg: 0.5856 loss_rpn_cls: 0.000482 loss_rpn_loc: 0.0008 time: 1.4555 valid_time: 0.0074
iter: 419 total_loss: 0.515 loss_cls: 0.1238 loss_box_reg: 0.3678 loss_rpn_cls: 0.000156 loss_rpn_loc: 0.006228 time: 1.4599 data_time: 0.0067 1
iter: 439 total_loss: 0.458 loss_cls: 0.09933 loss_box_reg: 0.2933 loss_rpn_cls: 0.001445 loss_rpn_loc: 0.006993 time: 1.4614 data_time: 0.0065 1
iter: 459 total_loss: 0.4489 loss_cls: 0.1086 loss_box_reg: 0.3066 loss_rpn_cls: 0.001613 loss_rpn_loc: 0.008878 time: 1.4617 data_time: 0.0118 1
iter: 479 total_loss: 0.3797 loss_cls: 0.0899 loss_box_reg: 0.2821 loss_rpn_cls: 0.001063 loss_rpn_loc: 0.005488 time: 1.4648 data_time: 0.0055 1
iter: 499 total_loss: 0.3173 loss_cls: 0.07185 loss_box_reg: 0.2324 loss_rpn_cls: 0.000969 loss_rpn_loc: 0.005676 time: 1.4643 data_time: 0.0108
iter: 519 total_loss: 0.3838 loss_cls: 0.08245 loss_box_reg: 0.2723 loss_rpn_cls: 0.0009434 loss_rpn_loc: 0.004797 time: 1.4648 data_time: 0.0083
iter: 539 total_loss: 0.3499 loss_cls: 0.07257 loss_box_reg: 0.2687 loss_rpn_cls: 0.0001045 loss_rpn_loc: 0.00528 time: 1.4656 data_time: 0.0074
iter: 559 total_loss: 0.3813 loss_cls: 0.09174 loss_box_reg: 0.2877 loss_rpn_cls: 0.0009764 loss_rpn_loc: 0.005236 time: 1.4661 data_time: 0.0068
iter: 579 total_loss: 0.3404 loss_cls: 0.08094 loss_box_reg: 0.2287 loss_rpn_cls: 0.00225 loss_rpn_loc: 0.006253 time: 1.4662 data_time: 0.0064 1
iter: 599 total_loss: 0.2718 loss_cls: 0.06116 loss_box_reg: 0.1843 loss_rpn_cls: 0.0009419 loss_rpn_loc: 0.00410 time: 1.4627 data_time: 0.0053
iter: 619 total_loss: 0.3808 loss_cls: 0.0952 loss_box_reg: 0.2953 loss_rpn_cls: 0.002437 loss_rpn_loc: 0.005221 time: 1.4630 data_time: 0.0064
iter: 639 total_loss: 0.2795 loss_cls: 0.06661 loss_box_reg: 0.1678 loss_rpn_cls: 0.0003968 loss_rpn_loc: 0.003968 time: 1.4623 data_time: 0.0096
iter: 659 total_loss: 0.3849 loss_cls: 0.08415 loss_box_reg: 0.2753 loss_rpn_cls: 0.001416 loss_rpn_loc: 0.004207 time: 1.4626 data_time: 0.0082
iter: 679 total_loss: 0.3997 loss_cls: 0.08936 loss_box_reg: 0.2923 loss_rpn_cls: 0.001617 loss_rpn_loc: 0.004633 time: 1.4643 data_time: 0.0080
iter: 699 total_loss: 0.3116 loss_cls: 0.05984 loss_box_reg: 0.2088 loss_rpn_cls: 0.0006302 loss_rpn_loc: 0.003978 time: 1.4649 data_time: 0.0088
iter: 719 total_loss: 0.3272 loss_cls: 0.07049 loss_box_reg: 0.251 loss_rpn_cls: 0.001993 loss_rpn_loc: 0.004183 time: 1.4630 data_time: 0.0073 1
iter: 739 total_loss: 0.3332 loss_cls: 0.07983 loss_box_reg: 0.2645 loss_rpn_cls: 0.0004828 loss_rpn_loc: 0.005705 time: 1.4622 data_time: 0.0068
iter: 759 total_loss: 0.3091 loss_cls: 0.06419 loss_box_reg: 0.2295 loss_rpn_cls: 0.0007336 loss_rpn_loc: 0.003687 time: 1.4623 data_time: 0.0066
iter: 779 total_loss: 0.2508 loss_cls: 0.04971 loss_box_reg: 0.1808 loss_rpn_cls: 0.0009957 loss_rpn_loc: 0.0034 time: 1.4616 data_time: 0.0083 1
iter: 799 total_loss: 0.2525 loss_cls: 0.06516 loss_box_reg: 0.1833 loss_rpn_cls: 0.0001471 loss_rpn_loc: 0.00467 time: 1.4617 data_time: 0.0066
ng speed: 798 iterations in 0:19:26 (1.4617 s / it)
time: 0:19:38 (0:00:03 on hooks)

```

Precisión de la red:

fast_rcnn

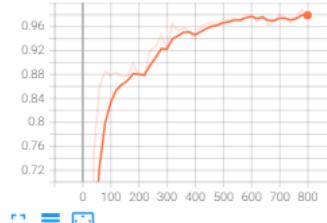
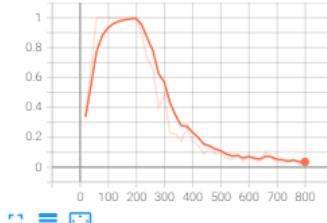
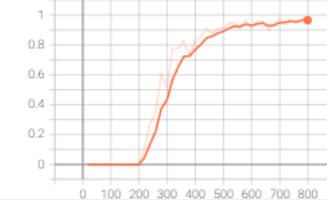
fast_rcnn/cls_accuracy
tag: fast_rcnn/cls_accuracyfast_rcnn/false_negative
tag: fast_rcnn/false_negativefast_rcnn_fg/cls_accuracy
tag: fast_rcnn_fg/cls_accuracy

Figura 32. Resultados de la precisión para la red de detección de armas cortas con Faster R-CNN.

Regresión de caja y pérdida de la red:

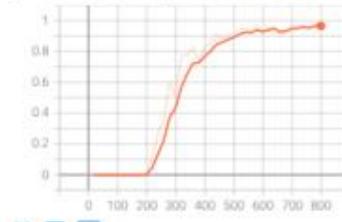
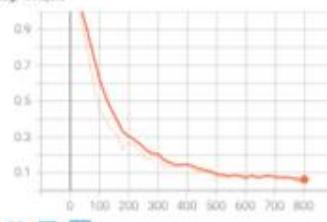
fast_rcnn_fg/cls_accuracy
tag: fast_rcnn_fg/cls_accuracyloss_cls
tag: loss_cls

Figura 33. Resultados de regresión de caja detección de objetos con Faster R-CNN.

Resultados presición AP:

```

Average Precision (AP) @[ IoU=0.50:0.95 | area= all | maxDets=100 ] = 0.575
Average Precision (AP) @[ IoU=0.50 | area= all | maxDets=100 ] = 0.864
Average Precision (AP) @[ IoU=0.75 | area= all | maxDets=100 ] = 0.659
Average Precision (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.200
Average Precision (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.335
Average Precision (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.667
Average Recall (AR) @[ IoU=0.50:0.95 | area= all | maxDets= 1 ] = 0.604
Average Recall (AR) @[ IoU=0.50:0.95 | area= all | maxDets= 10 ] = 0.641
Average Recall (AR) @[ IoU=0.50:0.95 | area= all | maxDets=100 ] = 0.641
Average Recall (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.267
Average Recall (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.422
Average Recall (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.733
[01/03 18:44:33 d2.evaluation.coco_evaluation]: Evaluation results for bbox:
| AP | AP50 | AP75 | APs | APm | APl |
|-----|-----|-----|-----|-----|-----|
| 57.501 | 86.374 | 65.925 | 19.984 | 33.541 | 66.684 |
[01/03 18:44:33 d2.evaluation.coco_evaluation]: Per-category bbox AP:
| category | AP | category | AP |
|-----|-----|-----|-----|
| 0 | nan | gun | 57.501 |
OrderedDict([('bbox', {'AP': 57.501, 'AP50': 86.37382805446936, 'AP75': 65.92500484699299, 'APs': 19.984258474995304, 'APm': 33.540779895223274, 'APl': 66.6841831819537, 'AP-0': nan, 'AP-gun': 57.50127630362272}))])

```

Para la métrica IoU se establecen los umbrales del 50% y 75%, con los resultados de 86.37382805446936 y 65.92500484699299 respectivamente. Al comparar con las métricas de evaluación generales para el conjunto de datos COCO (ver Figura 34), el grado de superposición para la detección de objetos es más alto.

	backbone	AP	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
<i>Two-stage methods</i>							
Faster R-CNN+++ [3]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [6]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [4]	Inception-ResNet-v2 [19]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [18]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	52.1

Figura 34. Rendimiento de detectores generales de objetos sobre COCO(What Do These Different AP Values Mean, 2020).

A continuacion, se muestran algunos ejemplos de imágenes detectadas con una presición mayor al 80%.

```
cfg.MODEL.ROI_HEADS.SCORE_THRESH_TEST = 0.80
```



Figura 35. Resultados detección de armas con sus valores de caja(box).

El resultado final de los pesos del [modelo de detección](#) pre-entrenado es de 796 MB.

Paso 2: Se hará uso de la red Mask R-CNN para la segmentación del arma, la cual se considera como un marco general para la segmentación de objetos para cada instancia de un objeto en la imagen. A diferencia de YOLOv5 v7.0, proporciona la segmentación de instancias al reconocer objetos en imágenes junto con su forma asociada; se considera más pequeño y generalmente más sencillo de utilizar en producción debido a su implementación en Pytorch. Está implementada en TensorFlow y se vincula con TensorBoard, que se utiliza como herramienta de depuración y visualización. Registra las pérdidas y pesos al final de cada época.

Para esta tarea se realizó el notebook que se encuentra en el repositorio:

☞ [5_2TFMModeloSegmentacionArmasCortasDetectronv2.ipynb](#)

Los parámetros empleados para este modelo son NUM_WORKERS=2, IMS_PER_BATCH=2, BASE_LR= 0.00055, MAX_ITER(épocas)=1200, BATCH_SIZE_PER_IMAGE=512 como parámetros (ver arquitectura para más detalles). Se realizaron diferentes pruebas donde la red entrenó durante 1200 épocas, ofreció los siguientes resultados:

```
iter: 599  total_loss: 0.3024  loss_cls: 0.0924  loss_box_reg: 0.1507  loss_rpn_cls: 0.000482  loss_rpn_loc: 0.000686  time: 1.4535  data_time: 0.0007  l
iter: 419  total_loss: 0.515  loss_cls: 0.1238  loss_box_reg: 0.3678  loss_rpn_cls: 0.0009156  loss_rpn_loc: 0.006228  time: 1.4599  data_time: 0.0067  l
iter: 439  total_loss: 0.458  loss_cls: 0.09933  loss_box_reg: 0.2933  loss_rpn_cls: 0.001445  loss_rpn_loc: 0.006993  time: 1.4614  data_time: 0.0065  l
iter: 459  total_loss: 0.4488  loss_cls: 0.1066  loss_box_reg: 0.3066  loss_rpn_cls: 0.001613  loss_rpn_loc: 0.008878  time: 1.4617  data_time: 0.0110  l
iter: 479  total_loss: 0.3797  loss_cls: 0.0899  loss_box_reg: 0.2821  loss_rpn_cls: 0.001063  loss_rpn_loc: 0.005488  time: 1.4648  data_time: 0.0055  l
iter: 499  total_loss: 0.3173  loss_cls: 0.07185  loss_box_reg: 0.2324  loss_rpn_cls: 0.000969  loss_rpn_loc: 0.005676  time: 1.4643  data_time: 0.0108  l
iter: 519  total_loss: 0.3839  loss_cls: 0.08245  loss_box_reg: 0.2723  loss_rpn_cls: 0.0009434  loss_rpn_loc: 0.004797  time: 1.4648  data_time: 0.0083  l
iter: 539  total_loss: 0.3499  loss_cls: 0.07257  loss_box_reg: 0.2687  loss_rpn_cls: 0.0001045  loss_rpn_loc: 0.005232  time: 1.4656  data_time: 0.0074  l
iter: 559  total_loss: 0.3813  loss_cls: 0.09174  loss_box_reg: 0.2877  loss_rpn_cls: 0.0009764  loss_rpn_loc: 0.005236  time: 1.4661  data_time: 0.0088  l
iter: 579  total_loss: 0.3404  loss_cls: 0.08094  loss_box_reg: 0.2287  loss_rpn_cls: 0.002226  loss_rpn_loc: 0.006253  time: 1.4652  data_time: 0.0084  l
iter: 599  total_loss: 0.2718  loss_cls: 0.06118  loss_box_reg: 0.1843  loss_rpn_cls: 0.0009345  loss_rpn_loc: 0.004118  time: 1.4627  data_time: 0.0053  l
iter: 619  total_loss: 0.3808  loss_cls: 0.09521  loss_box_reg: 0.2953  loss_rpn_cls: 0.002437  loss_rpn_loc: 0.005222  time: 1.4630  data_time: 0.0064  l
iter: 639  total_loss: 0.2795  loss_cls: 0.06661  loss_box_reg: 0.1678  loss_rpn_cls: 0.0003646  loss_rpn_loc: 0.003968  time: 1.4623  data_time: 0.0096  l
iter: 659  total_loss: 0.3849  loss_cls: 0.08445  loss_box_reg: 0.2753  loss_rpn_cls: 0.001416  loss_rpn_loc: 0.004207  time: 1.4626  data_time: 0.0082  l
iter: 679  total_loss: 0.3997  loss_cls: 0.080936  loss_box_reg: 0.2923  loss_rpn_cls: 0.001617  loss_rpn_loc: 0.004633  time: 1.4642  data_time: 0.0080  l
iter: 699  total_loss: 0.3116  loss_cls: 0.06984  loss_box_reg: 0.2088  loss_rpn_cls: 0.0006802  loss_rpn_loc: 0.003978  time: 1.4649  data_time: 0.0088  l
iter: 719  total_loss: 0.3272  loss_cls: 0.07045  loss_box_reg: 0.251  loss_rpn_cls: 0.001993  loss_rpn_loc: 0.004103  time: 1.4650  data_time: 0.0073  l
iter: 739  total_loss: 0.3332  loss_cls: 0.07983  loss_box_reg: 0.2645  loss_rpn_cls: 0.0004828  loss_rpn_loc: 0.005705  time: 1.4622  data_time: 0.0068  l
iter: 759  total_loss: 0.3091  loss_cls: 0.06419  loss_box_reg: 0.2295  loss_rpn_cls: 0.0007336  loss_rpn_loc: 0.003687  time: 1.4623  data_time: 0.0066  l
iter: 779  total_loss: 0.25808  loss_cls: 0.04971  loss_box_reg: 0.1808  loss_rpn_cls: 0.0009957  loss_rpn_loc: 0.0034  time: 1.4616  data_time: 0.0083  l
iter: 799  total_loss: 0.2525  loss_cls: 0.06516  loss_box_reg: 0.1833  loss_rpn_cls: 0.0001471  loss_rpn_loc: 0.00467  time: 1.4617  data_time: 0.0066  l
ng speed: 798 iterations in 0:19:26 (0:00:03 on hooks)
time: 0:19:38 (0:00:03 on hooks)
```

Precisión de la máscara:

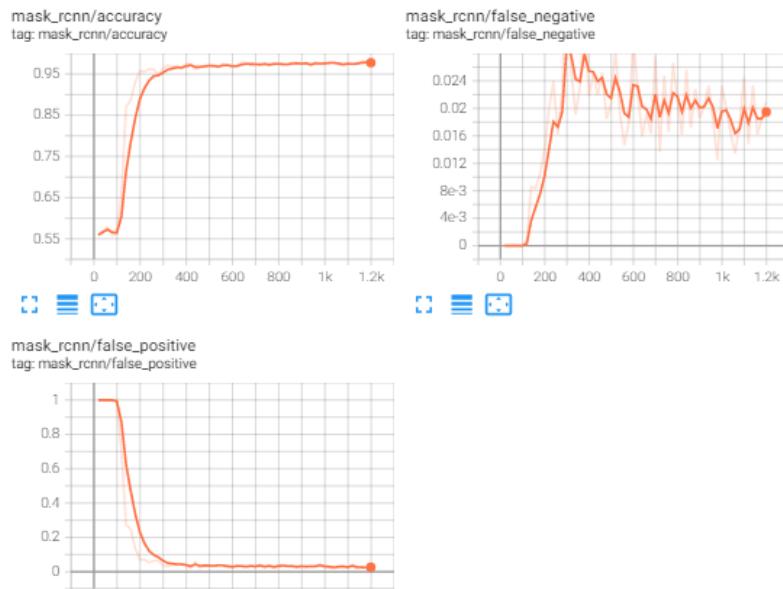


Figura 36. Resultados precisión para la red de segmentación (máscara arma).

Pérdidas de red de segmentación:

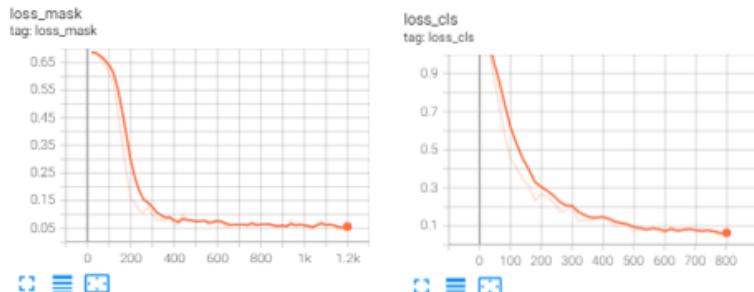


Figura 37. Resultados de las pérdidas de la red de segmentación.

Resultado precisión AP:

```
DONE (t=0.01s).
Average Precision (AP) @[ IoU=0.50:0.95 | area= all | maxDets=100 ] = 0.837
Average Precision (AP) @[ IoU=0.50 | area= all | maxDets=100 ] = 0.898
Average Precision (AP) @[ IoU=0.75 | area= all | maxDets=100 ] = 0.869
Average Precision (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
Average Precision (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
Average Precision (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.837
Average Precision (AP) @[ IoU=0.50:0.95 | area= all | maxDets= 1 ] = 0.860
Average Recall (AR) @[ IoU=0.50:0.95 | area= all | maxDets= 10 ] = 0.866
Average Recall (AR) @[ IoU=0.50:0.95 | area= all | maxDets=100 ] = 0.866
Average Recall (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
Average Recall (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
Average Recall (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.866
[01/03 20:07:56 d2.evaluation.coco_evaluation]: Evaluation results for bbox:
| AP      | AP50   | AP75   | APs    | APm   | AP1    |
|:-----:|:-----:|:-----:|:-----:|:-----:|:-----:|
| 83.674 | 89.758 | 86.925 | nan    | nan    | 83.674 |

OrderedDict([('bbox', {'AP': 83.67361734576924, 'AP50': 89.75836063301301, 'AP75': 86.92451330153258, 'APs': nan, 'APm': nan, 'AP1': 83.67361734576924}), ('segm', {'AP': 79.29605605494781, 'AP50': 90.57844264121383, 'AP75': 87.01067736469533, 'APs': nan, 'APm': nan, 'AP1': 83.67361734576924})])
```

```
'APl': 79.29605605494781, 'AP-guns': nan, 'AP-gun':  
79.29605605494781}))])
```

Para la métrica AP, el resultado es 83.67361734576924; en comparación con la métrica de evaluación AP para la red Mask R-CNN aplicada al conjunto de datos COCO ([enlace](#)), se observa una mayor superposición entre los objetos.

A continuación se muestran algunos ejemplos de imágenes detectadas con una presición mayor al 70%.

```
cfg.MODEL.ROI_HEADS.SCORE_THRESH_TEST = 0.70
```

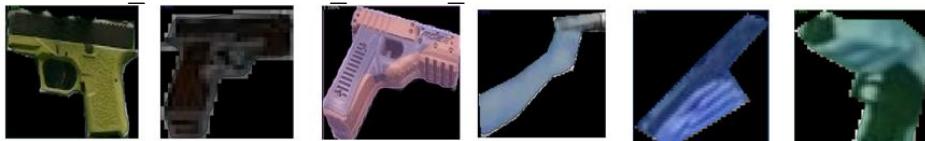


Figura 38. Ejemplos y resultados de la red de segmentación de armas.

El resultado final de los pesos del [modelo de segmentación](#) pre-entrenado es de 334.8MB.

4.4.5. Segmentación partes cuerpo personas con armas cortas utilizando modelos pre entrenados

Esta tarea es una de las más complejas, puesto que requiere la integración de los pesos y resultados de los modelos entrenados y desarrollados en los apartados anteriores, así como la combinación de actividades de detección y segmentación de personas con BodyPix. Se utilizará la librería PIL, que es una de las más conocidas para el tratamiento de imágenes.

Se trata de una actividad de integración, que no produce resultados de evaluación o métricas, ya que no genera ningún modelo. Se crean dos archivos con formato notebook, uno para cada arquitectura de superpixeles RDN y ESRGAN. Estos documentos contienen la implementación específica mediante clases, constantes, parámetros y métodos relacionados con la implementación del siguiente fragmento de pseudocódigo; el cual propone una línea general para el procesamiento, detección y segmentación de imágenes para personas con armas cortas:

```
# Estructura general para la segmentación de armas  
Descargar conjunto de datos persona armada  
Configurar modelos pre entrenados  
Para cada imagen:  
    # Detección Objetos  
    Realizar la detección objeto de interés (armas cortas)  
    Recortar y guardar imagen(jpg)  
    Guardar recuadro detección(txt)  
    # Segmentación Objetos  
    Realizar la segmentación objeto de interés (armas cortas)
```

Guardar imagen formato mapa de bits(png)

Segmentación Objetos

Realizar la segmentación de las partes del cuerpo (persona)

Recortar y guardar imagen(jpg)

Cambiar color máscara

Cambiar color máscara fondo blanco por rojo

Guardar imagen formato mapa de bits(png)

Agregar máscara color en persona segmentada

Identificar en imagen segmentada recuadro detección(txt)

Agregar en esta imagen segmentada la máscara con color(png)

Guardar imagen segmentada partes del cuerpo con arma(jpg)

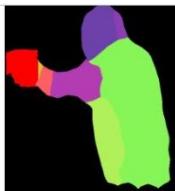
continuar

Comprimir imágenes

Para esta tarea se realizó los siguientes notebooks que se encuentran en los repositorios:

- [6_1TFMSegmentacionCuerpoPersonasConArmasCortasRDN.ipynb](#)
- [6_2TFMSegmentacionCuerpoPersonasConArmasCortasESRGAN.ipynb](#)

En la siguiente tabla se muestran algunos ejemplos de los resultados(raw/data/segmented-body-parts-gun), divididos en filas. En la parte superior se muestran las imágenes originales, seguidas de las imágenes segmentadas, y al final una máscara de segmentación de bits resultado de las actividades desarrolladas en los puntos anteriores.

Real				
Segmentada				
Máscara				
Real				

Segmentada				
Máscara				

Tabla 15.. Resultados procesos de segmentación de partes del cuerpo de personas y armas.

4.4.6. Preparación de los datos

La limpieza y preparación de los datos es un aspecto crucial a tener en cuenta en los problemas de Aprendizaje automático e inteligencia artificial. Para el análisis de imágenes en relación con la detección de objetos y segmentación de imágenes, no existe una forma “simple” de realizar exploraciones de datos de manera sistemática con el objetivo de tener datos de calidad previo a las etapas de entrenamiento y validación.

Según lo detallado por Jakub Cieślik, para aplicaciones en aprendizaje profundo basadas en imágenes, se presentan varios problemas comunes respecto de la calidad de los datos. De manera general, al enfrentar cualquier problema de aprendizaje automático (segmentación de imágenes, detección de objetos), es vital comprender y evaluar la calidad de sus datos. Los problemas de datos comunes al entrenar modelos de detección de objetos y segmentación de imágenes incluyen(Cieślik, 2022):

- Dimensiones de la imagen y relaciones de aspecto (especialmente cuando se trata de valores extremos)
- Composición de etiquetas: desequilibrios, tamaños de cuadros delimitadores, relaciones de aspecto (por ejemplo, muchos objetos pequeños).
- La preparación de datos no es adecuada para su conjunto de datos.
- Enfoque de modelado no alineado con los datos.

Para este trabajo se pretende abordar los siguientes aspectos como críticos, principalmente porque se han utilizado varios conjuntos de datos previos sin tratamiento.

Calidad de los datos generales

Se puede considerar un aspecto básico y bastante obvio; pero es necesario realizarlo para todos los problemas relacionados con imágenes. En este paso realizamos:

- Efectuar una inspección visual de algunas imágenes, con el objetivo de obtener una idea general del conjunto de datos e inspeccionarlo visualmente.
- Validar que ningún archivo esté corrupto y no contenga ningún problema “obvio”. Para el caso de las máscaras de bit tener imágenes solo en negro o solo en blanco.

Tamaños de imagen y relación de aspecto

En el mundo real, es poco probable tener conjuntos de datos que contengan imágenes del mismo tamaño y relación de aspecto. Para ciertos conjuntos de imágenes, incluso se ha efectuado una tarea previa de conversión de una imagen de tamaño pequeño hacia una imagen grande y de alta resolución. La inspección de estadísticas básicas de conjuntos de datos, como relaciones de aspecto, anchos y alturas de imágenes, permite tomar decisiones importantes:

- Por ejemplo, el cambio de tamaño puede implicar una mejora o un problema al entrenar las imágenes.
- Si se desea modificar el tamaño de una imagen, es necesario reflexionar y considerar como un cambio de tamaño, puede afectar una imagen. Por ejemplo, la conversión fue adecuada o destructiva. **¿Cuál debería ser la resolución de salida deseada?**
- Los modelos de aprendizaje profundo pueden tener hiperparámetros que se pueden ajustar según lo anterior (por ejemplo, el tamaño y las proporciones del arma) o incluso pueden tener requisitos estrictos respecto al tamaño mínimo de la imagen de entrada.

Para esta tarea se realizó el siguiente notebook que se encuentran en el repositorio:

☞ [7_1TFMCrearDataSetImagenes.ipynb](#)

Los resultados luego de la valoración estadística son:

- 1) Se aprecia que después del proceso de conversión hacia imágenes de gran y alta resolución existen imágenes, cuyo tamaño corresponde a medidas anómalas (outliers).

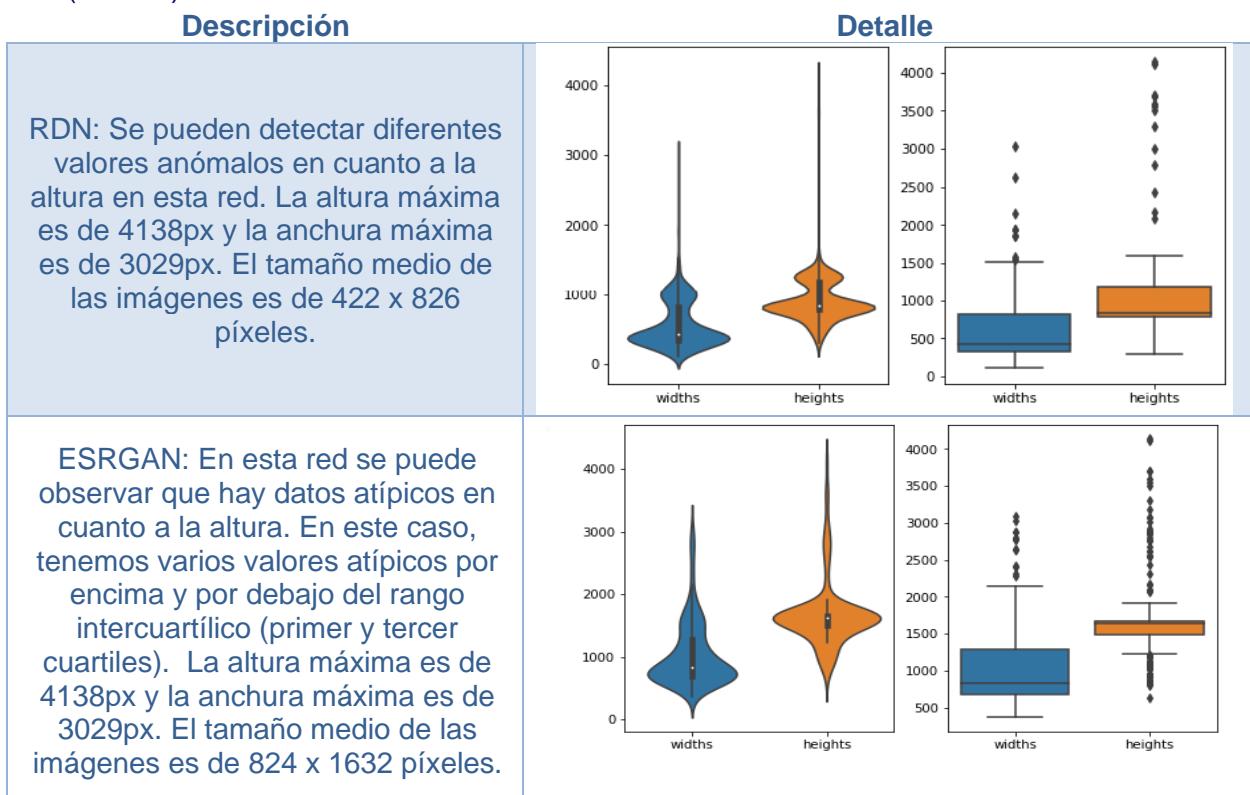


Tabla 16. Evaluación estadística de las dimensiones de imágenes transformadas con superpixeles.

2) Existen varias imágenes cuyo resultado de segmentación presenta máscaras de bits con valores en su mayoría iguales a uno. Por consiguiente, se puede afirmar que la segmentación no fue eficaz en esas armas; ya sea por el color de fondo o por las modificaciones (suavizado y eliminación de bordes) en la imagen producto de la conversión hacia superpixeles.

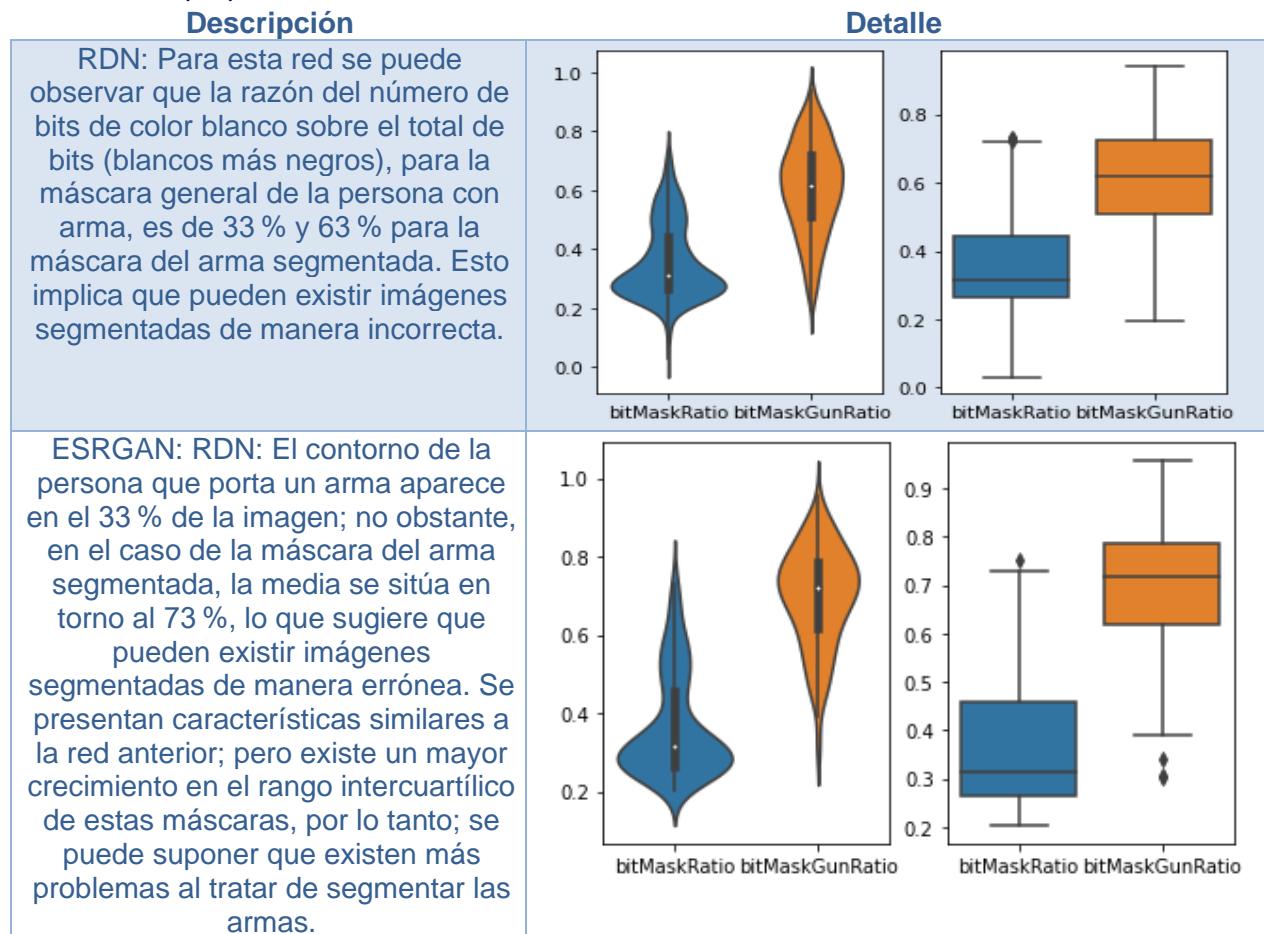


Tabla 17 Evaluación estadística radio de máscara segmentada del arma.

3) Para los dos conjuntos de imágenes, se puede hacer una comparación visual para observar los resultados del número de bits de las máscaras de armas. Se toma un conjunto de 20 imágenes aleatorias, en las que se puede apreciar que para las imágenes que se encuentran por debajo del tercer cuartil, las redes de aprendizaje profundo generativo recurrente (RDN) tienen un mejor resultado al segmentar en comparación con la red ESRGAN, según se muestra en la segunda fila.





Tabla 18. Comparaciones segmentación de armas; primera fila valores menores al 3.er cuartil.

Una vez finalizados los procesos de limpieza y exploración se obtienen los siguientes resultados estadísticos del conjunto de datos que será utilizado por la red GAN.

```
dataFrameCleanRDN.describe()
```

	widths	heights	bitMaskRatio	bitMaskGunRatio
count	683.000000	683.000000	683.000000	683.000000
mean	514.852123	871.181552	0.347419	0.545777
std	280.963174	229.391799	0.122948	0.123844
min	110.000000	298.000000	0.096862	0.195707
25 %	332.000000	788.000000	0.261501	0.465053
50 %	410.000000	824.000000	0.303101	0.571719
75 %	650.000000	850.000000	0.408101	0.649210
max	1440.000000	1600.000000	0.735198	0.724247

Tabla 19. Resultados del conjunto de datos de imágenes tratado.

CONCLUSIÓN:

Después de analizar las estadísticas para cada grupo de datos, decidimos usar el conjunto RDN (imágenes grandes) en el cual el tamaño medio de los píxeles es de 410 por 824. Se seleccionaron solo las imágenes en las que la segmentación del arma se encontraba en el tercer cuartil. Se crea un conjunto de datos en el que se tiene, por un lado, la imagen segmentada y a su costado la imagen real como se puede ver en la figura; en el que se aplican convoluciones sobre las imágenes y se propone el uso de una red U-Net; estas deben tener el mismo largo y ancho cuyo valor será 256 x 256 píxeles, por cada imagen (ver Figura 39).

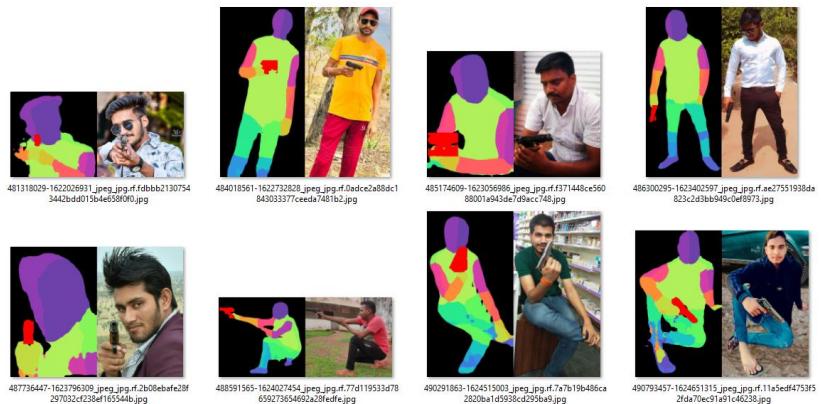


Figura 39. Imágenes de ejemplo a la derecha segmentada y a la izquierda original.

Las imágenes se encuentran subidas con la herramienta Roboflow, en un repositorio público:

Tipo Conjunto	Fuente
Personas armas cortas segmentadas	<pre>@misc{ dataset-gun-segmented_dataset, title = { dataset-gun-segmented Dataset }, type = { Open Source Dataset }, author = { Miguel Alejandro Ponce Proa\~no }, howpublished = { \url{ https://universe.roboflow.com/miguel-alejandro-ponce-proano/dataset-gun-segmented } }, url = { https://universe.roboflow.com/miguel-alejandro-ponce-proano/dataset-gun-segmented }, journal = { Roboflow Universe }, publisher = { Roboflow }, year = { 2023 }, month = { jan }, note = { visited on 2023-01-10 }, }</pre>

Tabla 20. Conjunto de datos de prueba imágenes segmentadas y real.

Se presentan algunos ejemplos y las características de este conjunto de datos.

data-set-person-armed-segmented

	gmd_6_frame5... gmd_3_frame61... gmd_2_frame51... gmd_4_frame51... armas-42.jpg mgd_custom_fr... mgd_custom_fr... gmd_3_frame8...
Nro. Imágenes	915
Reajuste Tamaño	512x1024 / 256x512
Formato	COCO JS
Versión	3
Autor	Miguel Ponce

Tabla 21. Características del conjunto de entrenamiento

El conjunto de datos se encuentra dividido de la siguiente manera:

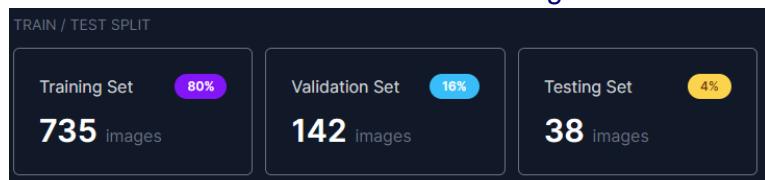


Figura 40. División del conjunto de datos, imágenes segmentada y original.

4.4.7. Red de segmentación GAN

A continuación, se realiza una descripción del modelo de segmentación mediante el empleo de redes antagónicas, en donde se hace uso de las imágenes generadas en los apartados anteriores. El entorno de trabajo es Google Colaborador en su versión Pro+, adicionalmente se utilizó

Para esta tarea se realizó el siguiente notebook que se encuentran en el repositorio:

∞ [8TFMMModeloGANPix2Pix_Unet.ipynb](#)

Red: Modelo de detección y segmentación de personas armadas

Contexto: Se desea crear un modelo que permita segmentar personas con armas cortas.

Problema: El objetivo del proyecto es sugerir nuevos sistemas de vigilancia y alerta temprana contra sospechosos de intento de asesinato y porte ilegal de armas mediante la mejora de los métodos de segmentación de objetos en imágenes. Es necesario generar un modelo de segmentación de personas con armas cortas, basado en un conjunto de datos pre-entrenados utilizando redes GAN y arquitecturas pensadas para la segmentación de objetos.

Solución: Elaborar una red antagónica generativa condicional (cGAN) inspirada en el trabajo de pix2pix, la cual es una red que efectúa un proceso de mapeo de imágenes de entrada hacia imágenes de salida. Se conoce a esta arquitectura como traducción de imagen a imagen con GAN, trabajo desarrollado por Isola et al. en 2017. Aplicar esta arquitectura de red multipropósito porque este trabajo puede ser empleado en varias tareas.

Entorno: con GPU.

Licencia: MIT, La licencia del Massachusetts Institute of Technology (MIT) es un software de licencia gratuito.

Características de la red pix2pix

Se desarrollará una red que genere imágenes segmentadas de las partes del cuerpo de una persona armada, permitiendo así una mejor identificación de estas. Emplea una base de datos que contiene a las personas y las armas segmentadas (color rojo), trabajo desarrollado por Miguel Ponce, estudiante del Máster de Ciencia de Datos de la Universidad de UOC. La arquitectura de esta red está formada por:

- **Generador:** una arquitectura basada en U-Net.
- **Discriminador:** representado por un clasificador PatchGAN convolucional propuesto en el artículo de pix2pix.
- **Épocas:** Se debe tomar en cuenta que cada época puede demorar alrededor varias horas dependiendo de la cantidad de GPU aprovisionada. Se realizará el entrenamiento de la red cGAN desde 40000 hasta, 160000 épocas en el conjunto de datos.

Conjunto de datos

Cada imagen tiene un tamaño de 256x512 píxeles, y se divide en dos imágenes de tamaño 256x256 píxeles; a la izquierda se encuentra la imagen segmentada y a la derecha la imagen original. A la derecha se muestra la imagen original y a la izquierda la misma imagen, pero segmentada, ver Figura 41.



Figura 41. Ejemplo conjunto datos, imagen real a la derecha, imagen segmentada a la izquierda.

El preprocesamiento de los datos se hace aplicando rotaciones y reflejos aleatorios (random_jitter) según la recomendación de la publicación de la red pix2pix; en donde tenemos las funciones:

- Se cambia el tamaño de cada imagen 256 x 256 a una mayor altura y anchura: 266 x 266.
- Se recorta aleatoriamente a 256 x 256.
- Se voltear aleatoriamente la imagen horizontalmente, es decir, de izquierda a derecha (espejo aleatorio).
- Se Normaliza las imágenes al rango [-1, 1].



Figura 42. Ejemplo rotaciones aleatorias y zoom(jitter).

GENERADOR

El generador de la red pix2pix cGAN es una red U-Net modificada. Una U-Net consta de un codificador (disminución de muestreo) y un decodificador (disminución de muestreo). En donde cada bloque del codificador está formado por: **Convolution -> Batch normalization -> Leaky ReLU**. Cada bloque en el decodificador es: **Transposed convolution -> Batch normalization -> Dropout (aplicado a los 3 primeros bloques) -> ReLU**. Existen conexiones de salto entre el codificador y el decodificador. Se puede ver la definición de esta arquitectura detallada en, el anexo **8.5 Arquitectura Generador**; sin embargo, el proceso que sigue en generador es el que se muestra en la siguiente figura:

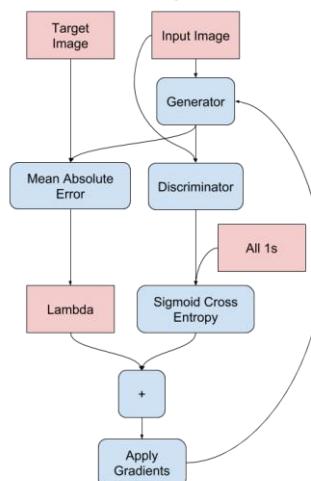


Figura 43. Arquitectura del generador.

Podemos visualizar un ejemplo del generador para un tensor con la forma:
TensorShape([1, 256, 256, 3])

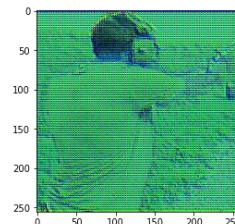


Figura 44. Ejemplo resultado generador.

Pérdida del generador

Las cGAN aprenden de una pérdida que penaliza una posible estructura que difiere de la salida de la red y la imagen de destino, como se describe en el artículo [Image-to-Image Translation with Conditional Adversarial Networks](#).

- **gan_loss:** es la pérdida del generador es una pérdida de entropía cruzada sigmoidea de las imágenes generadas y una matriz de unos.
- **L1:** es una pérdida que mide el MAE (error absoluto medio) entre la imagen generada y la imagen de destino. Esto permite que la imagen generada se vuelva estructuralmente similar a la imagen de destino.

La fórmula para calcular la pérdida total del generador es la suma de la pérdida de la red GAN (gan_loss) + LAMBDA * l1_loss, donde LAMBDA = 100.

DISCRIMINADOR

PatchGAN es un tipo de discriminador para redes antagónicas generativas que solo penaliza la estructura a escala local de parches de imagen. El discriminador PatchGAN trata de determinar si cada parche en una imagen es real o no. Este discriminador se ejecuta de forma convolucional a través de la imagen, obteniendo la media de todas las respuestas para proporcionar el resultado final. Este discriminador modeliza efectivamente la imagen como un campo aleatorio de Markov, suponiendo independencia entre píxeles separados por más de un diámetro de parche. Puede entenderse como un tipo de pérdida de textura/estilo(*Papers with Code - PatchGAN Explained*, n.d.), se puede ver la arquitectura detallada en el anexo 8.6 Arquitectura Discriminador; sin embargo, el proceso que sigue en discriminador es el que se muestra en la siguiente figura:

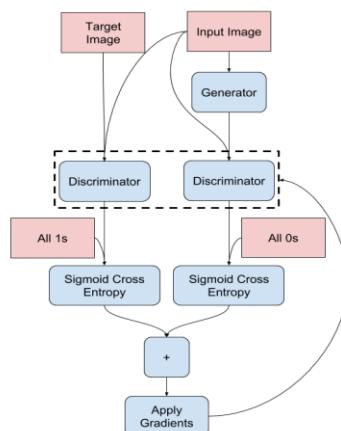


Figura 45. Arquitectura del discriminador.

El discriminador es un clasificador PatchGAN convolucional que intenta clasificar si cada imagen parche es real o no, en donde:

- Cada bloque en el discriminador es: **Convolution -> Batch normalization -> Leaky ReLU.**
- La forma de la salida después de la última capa es (batch_size, 30, 30, 1).
- Cada parche de imagen 30 x 30 de la salida clasifica una porción 70 x 70 de la imagen de entrada.

El discriminador recibe 2 entradas:

- La imagen de entrada y la imagen de destino, que debe clasificar como real.
- La imagen de entrada y la imagen generada (la salida del generador), que debe clasificar como falsa.
- Utiliza `tf.concatenate([inp, tar], axis=-1)` para concatenar estas 2 entradas juntas.

Podemos visualizar un ejemplo del generador para un tensor con la forma:

`TensorShape([1, 256, 256, 3])`

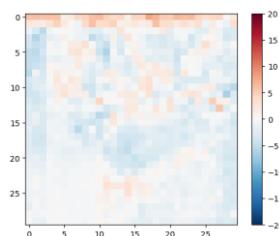


Figura 46. Ejemplo resultado del discriminador.

Pérdida del discriminador

La función `discriminator_loss` tiene 2 entradas: las imágenes reales y las imágenes generadas.

- **real_loss:** es una pérdida de entropía cruzada sigmoidea de las imágenes reales y una matriz de unos (ya que estas son las imágenes reales).
- **generated_loss:** es una pérdida de entropía cruzada sigmoidea de las imágenes generadas y una matriz de ceros (puesto que estas son las imágenes falsas).

La pérdida total es la suma de la pérdida real y la pérdida generada.

Optimizadores

Utiliza Adam, porque es un algoritmo popular en el campo del aprendizaje profundo porque logra buenos resultados rápidamente. Resultados empíricos demuestran que Adam funciona bien en la práctica y se compara favorablemente con otros métodos de optimización estocástica. Jason Brownlee define los parámetros de Adam según(Brownlee, 2017):

- **alfa.** También se conoce como tasa de aprendizaje o tamaño de paso. La proporción en que se actualizan los pesos (por ejemplo, 0,001). Los valores más grandes (p. ej., 0,3) dan como resultado un aprendizaje inicial más rápido antes de que se actualice la tasa. Los valores más pequeños (por ejemplo, 1.0E-5) ralentizan el aprendizaje durante el entrenamiento
- **beta1.** La tasa de caída exponencial para las estimaciones del primer momento.

- **beta2.** La tasa de caída exponencial para las estimaciones del segundo momento. Este valor debe establecerse cerca de 1,0 en problemas con un gradiente escaso, por ejemplo, para problemas de visión por computadora.
- **epsilon.** Es un número muy pequeño para evitar cualquier división por cero en la implementación.

```
generator_optimizer = tf.keras.optimizers.Adam(2e-4, beta_1=0.5)
discriminator_optimizer = tf.keras.optimizers.Adam(2e-4, beta_1=0.5)
```

Proceso de entrenamiento:

El proceso es el siguiente:

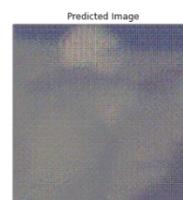
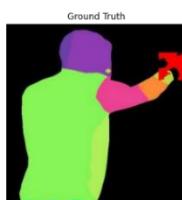
El generador para cada imagen de entrada genera una imagen de salida.

El discriminador recibe la imagen de entrada y la imagen generada como primera entrada. La segunda entrada es input_image y target_image.

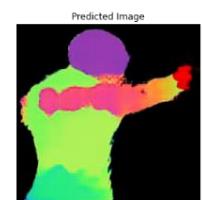
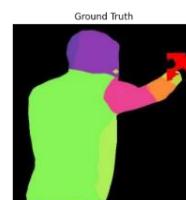
Calcular el generador y la pérdida del discriminador.

Calcular las pérdidas con respecto tanto al generador y al discriminador(entradas) y se deben aplicar al optimizador.

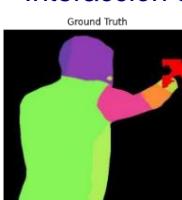
Interacción 0:



Iteración 25000



Iteración 35000



Iteración 60000

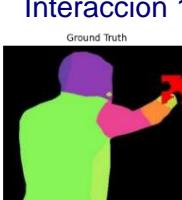


Tabla 22. Ejemplo resultados visuales por interacción.

Los resultados de las métricas para 160.000 iteraciones son:

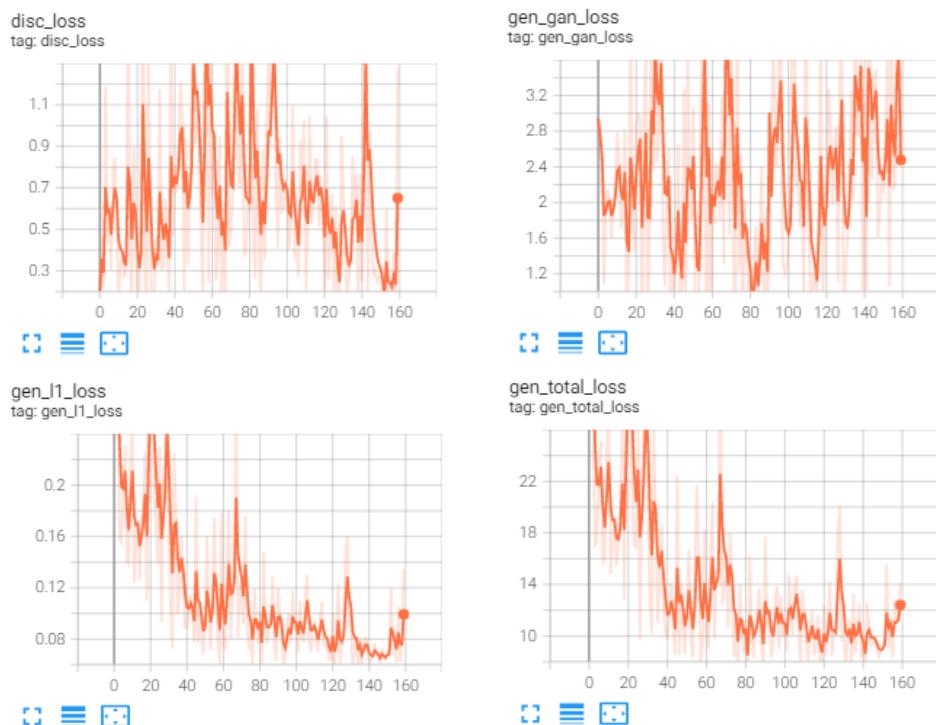


Tabla 23. Resultados de la evaluación de métricas de pérdida de la red generador y discriminador.

Evaluación general del modelo:

La interpretación de los registros es más compleja y abstracta cuando se entrena una GAN (o una cGAN como pix2pix), en comparación con otros modelos como la clasificación o la regresión. Se puede comprobar que, dado que se busca el equilibrio en la red según la teoría de Nash, **ni el generador ni el discriminador han “obtenido un beneficio”**.

- Como **gen_gan_loss** o **disc_loss** son altos, es un indicativo de que el modelo combinado se está entrenando correctamente.
- El valor $\log(2.4) = 0.38$ es un buen punto de referencia para las pérdidas, ya que indica una perplejidad de 2.4. Donde el discriminador no tiene ninguna preferencia por ninguna de las dos opciones.
- Para **gen_gan_loss**, un valor por debajo de 0.38 significa que está tratando de discriminar de manera aleatoria entre el conjunto de imágenes reales y generadas. Por tanto, el generador está tratando de engañar al discriminador.
- A medida que progresa el entrenamiento, la pérdida de **gen_l1** sigue disminuyendo. A continuación, la pérdida **L1**, también denominada pérdida de error absoluto, que es la diferencia absoluta entre una predicción y el valor real. La pérdida **L1** es un cálculo de error que se realiza para cada ejemplo en el que se desea comprender qué tan bien se predijo para esa observación; luego, un L1 que disminuye y tiende a cero es un buen indicador.

Otras métricas:

Inception Score

Esta puntuación es una medida de cuán realista es la salida de una GAN. En palabras de sus autores, “encontramos que [el IS] se correlaciona bien con la evaluación humana [de la calidad de la imagen]”. Es una alternativa automática a que los humanos califiquen la calidad de las imágenes. Inception Score mide dos cosas simultáneamente:

- Las imágenes tienen variedad (por ejemplo, cada imagen es una raza de perro diferente)
- Cada imagen claramente se parece a algo diferente.

Si ambas cosas son ciertas, la puntuación será alta. Si uno o ambos son falsos, la puntuación será baja. Una puntuación más alta es mejor. Significa que su GAN puede generar muchas imágenes distintas diferentes. El IS ha demostrado ser útil y popular, aunque tiene limitaciones. El resultado de esta métrica es 1.0096176, ya que el modelo en realidad muestra una sola persona.

Intersección sobre Unión

Esta métrica es mucho más interesante, también conocido como Índice Jaccard, es una de las métricas más utilizadas en la segmentación semántica. El IOU es el área de superposición entre la segmentación predicha y la verdadera. Esta métrica varía de 0 a 1 (0 a 100 %), donde 0 significa que no hay superposición y 1 significa una segmentación perfectamente superpuesta. Como se puede ver, la métrica tiene un valor de 0.920772745950344, lo que indica que el modelo está generando máscaras de segmentos muy parecido a la imagen original.

Dice

El índice de Dice es muy similar al índice de Jaccard, ya que mide la similitud y diversidad de conjuntos de muestras. Aunque se calculan de manera similar, el índice de Dice es un poco más intuitivo porque se puede ver como el porcentaje de superposición entre dos conjuntos, que es un valor entre 0 y 1. Para este trabajo el índice es de 0.94959, por tanto, se están solapando las imágenes real y predicha.

Finalmente, se puede ver los resultados de otras pruebas en 8.7 Otras pruebas, de los Anexos, y en el repositorio de GitHub en el siguiente [enlace](#).

El resultado final de los pesos del [modelo de segmentación de peronas con armas pre-entrenado](#) es de 654MB.

Resultados

La siguiente tabla presenta una comparación de las métricas de este apartado y el anterior en relación con el conjunto de datos COCO.

Algoritmo	Red	Métrica	Nuestro	COCO
Detección arma	Faster R-CNN	IoU 50%	86,37	55,7
Detección arma	Faster R-CNN	IoU 75%	65,92	34,7
Segmentación arma	Mask R-CNN	AP	83,67	43,3
Segmentación persona armada	GAN	IoU	92	NA
Segmentación persona armada	GAN	Dice	94	NA

Tabla 24. Comparación de los resultados de los métodos tradicionales y GAN frente a COCO.

5. Conclusiones y trabajos futuros

En este trabajo se han presentado varios modelos para segmentar imágenes de personas armadas y establecer comparaciones entre los métodos tradicionales y aquellos basados en redes antagónicas generativas. El enfoque se centra en el uso de herramientas e investigaciones de vanguardia para la detección y segmentación de personas con armas mediante modelos considerados como estándares para abordar este tipo de problemas. Se desarrollaron diversos recursos que permitieron sintetizar y simplificar los procesos para la creación de un conjunto de datos de imágenes reales y segmentadas utilizados como insumo para una red GAN conocida como pix2pix con un generador basado en Unet. Se presentó la evaluación de las métricas, estas demuestran la idoneidad de la red. Debido a sus características, la red neuronal puede seguir expandiéndose en función de nuevos tipos o modelos de armas.

Las GPU desempeñan un papel muy relevante en el desarrollo de redes antagónicas generativas; dado que, por su propia naturaleza, estas buscan aprender a través de la competencia entre una y otra, hasta alcanzar un “**equilibrio**”. Por consiguiente, su uso requiere una gran cantidad de tiempo de entrenamiento y se convierte en un factor decisivo para productos o servicios de predicción en la nube. En este proyecto se adquirió varias unidades de GPU de gama alta (A100-SXM4-40GB) y se trabajó con un entorno local con una GPU (RTX 3070 Ti); se evidenció que la velocidad de entrenamiento de la red depende de la calidad de la GPU utilizada; en donde superó ampliamente los tiempos de respuesta de una GPU tradicional. En consecuencia, las redes GAN requieren una gran cantidad de tiempo para completar los entrenamientos y siempre estarán consumiendo recursos hasta alcanzar su objetivo.

El empleo indiscriminado de técnicas para mejorar la calidad de la imagen puede afectar adversamente la calidad de los modelos de detección y, en particular, de segmentación de objetos. El intentar mejorar la calidad de imagen puede producir resultados contrarios. Por ejemplo, al convertir una imagen pequeña (200 píxeles) en una imagen de superpíxeles (4200 píxeles), se puede distorsionar la imagen original y hacer que pierda los bordes o formas que le confieren su característica propia. Las técnicas de preparación de datos y visualización nos pueden sugerir o ayudar a identificar los beneficios o pérdidas que se derivan de estos procesos. En consecuencia, llevar a cabo procesos de validación, preparación y visualización de datos en combinación con la estadística, puede ayudar a decidir sobre qué datos nos serán beneficiosos para los procesos posteriores.

Los datos no equilibrados nos obligan a tener sesgos, que durante el entrenamiento pueden deteriorar la calidad del modelo. Los sesgos pueden surgir debido a que no existe variedad en los datos o que las muestras no tienen en cuenta a las minorías. La calidad de la red se ve afectada por los datos, de modo que si existen datos no balanceados, esto puede tener un impacto en el entrenamiento de la misma. El disponer de procesos claros para el procesamiento de imágenes puede facilitar la reducción de los tiempos. Se necesitó agregar nuevas imágenes (personas con armas y sin armas) al conjunto de datos original para entrenar los modelos; en donde, gracias a los procesos creados inicialmente y en colaboración con herramientas y utilitarios de imágenes, fue posible recopilar nuevos datos

de manera rápida y transparente para la nueva red. En conclusión, se determina que para una red con una variedad de datos ayuda a resolver problemas de sobreentrenamiento.

En las redes antagónicas generativas, un generador y un discriminador compiten como si fuera un juego; pero ¿**Cómo podemos evaluar si alcanzamos un “equilibrio”?** Las métricas inadecuadas pueden proporcionar una visión subjetiva o parcializada de un modelo. Existen diversas métricas de calidad para comprender si una red logra alcanzar este “**equilibrio**” para las redes GAN. En el trabajo se utilizan técnicas y modelos de segmentación; por lo tanto, lo mejor será buscar métricas que permitan justificar si los procesos de segmentación se cumplen. En consecuencia, el uso de otras métricas como Dice o IOU puede ayudar a entender si existe una segmentación adecuada de las imágenes.

Al evaluar los resultados de las diversas redes: detección de objetos, segmentación y GAN, se puede confirmar su eficacia. La detección y segmentación de tareas mediante métodos tradicionales proporcionan una solución eficaz y robusta, ya que existe una amplia variedad de trabajos y librerías optimizadas para este propósito. Por ejemplo, podrían generarse nuevos modelos con otras arquitecturas para la detección y segmentación; puesto que el problema original se podría extender hacia otro tipo de armas. Por el contrario, la red GAN ofrece resultados notables; si se examinan visualmente, se puede apreciar que al segmentar el arma se pretende darle una forma más estilizada. Aunque las redes son comparables en sus métricas y resultados, si pensamos en el costo-beneficio, el mantenimiento y disponibilidad para un servicio en la nube, la balanza se puede inclinar hacia el uso de redes tradicionales.

El empleo del método científico como metodología de trabajo ha permitido atenernos al cronograma del proyecto. Esta es una opción adecuada para alcanzar los objetivos del proyecto, ya que la definición de hipótesis nos permite formular preguntas y determinar si existen respuestas. En consecuencia, podemos hallar respuestas mediante investigaciones, arquitecturas o herramientas previas que puedan auxiliarnos a responder a un problema. Se utilizaron redes preentrenadas para la detección de objetos, en la medida de lo posible, con el fin de mejorar la precisión del modelo. Por el contrario, la carencia de recursos o modelos previamente entrenados para la segmentación de armas requiere reducir el alcance y enfocar la investigación en un tipo específico de arma. El éxito y resultado final del trabajo se debió a que, al realizar la experimentación, se pudieron probar algoritmos y herramientas de última generación, que permitieron crear productos reproducibles y útiles.

El trabajo tiene un impacto social positivo porque busca mitigar un problema social como es la delincuencia y el sicariato con una solución real que involucra a la sociedad en general. La estructura del proyecto permitirá agregar más modelos o tipos de armas de manera transparente. Los conjuntos de datos y los resultados de los modelos son públicos y pueden ser descargados o modificados por nuevos autores. Las diversas piezas de código fuente se pueden ejecutar en la nube de manera inmediata sin necesidad de instalar software complementario. Al desarrollar métodos y herramientas que permitan hacer frente a la delincuencia y el sicariato mediante la detección de personas con armas de fuego, se puede decir que se ha logrado cumplir con este objetivo social.

LÍNEAS DE TRABAJO FUTURO

Nuevas redes de segmentación en 3D

En el proyecto [3DHumanGAN](#), de Zhuoqian Yang, et al, publicado en diciembre de 2022, en el cual se propone una red GAN en 3D que sintetiza imágenes de personas de cuerpo completo, con apariencias consistentes bajo diferentes ángulos de visión y poses corporales. Este trabajo todavía no tiene cargados los recursos en un repositorio público; sin embargo, desde una visión crítica, esta red ofrece una mayor calidad en la segmentación de una persona, detectando incluso detalles particulares como formas de los dedos y otros. Una aplicación interesante sería incluir la detección del arma segmentada dentro de la red de detección de las personas(*3DHumanGAN: Towards Photo-Realistic 3D-Aware Human Image Generation*, n.d.).



Figura 47. Ejemplo imágenes partes cuerpo personas segmentadas en 3D(*3DHumanGAN: Towards Photo-Realistic 3D-Aware Human Image Generation*, n.d.).

Conjuntos de datos en 3D

La empresa [EdgcaseAI](#), que brinda soluciones de inteligencia artificial a gran escala. Adicionalmente, dentro de sus principios buscan establecer una relación respecto de los principios éticos y social porque como mencionan en su blog oficial: “En Edgcase.ai siempre nos esforzamos por brindar valor a la comunidad.”. Luego, en un intento de enfrentar, cito: “las tragedias que nos rodean, es imperativo que tanto las empresas como los aficionados tengan los datos y las herramientas necesarias para detectar armas en la naturaleza.”, han desarrollado un conjunto de datos de detección de armas sintéticas, para ayudar a los investigadores y a las partes interesadas en desarrollar algoritmos de detección de armas(Giddens, 2019).



Figura 48. Ejemplo conjunto datos armas segmentadas entorno 3D(Giddens, 2019)

Este conjunto de datos puede favorecer la detección debido a que al ser 3D, se puede:

- Cambiar el ángulo, posición y campo de visión de la cámara.
- Establecer colores y texturas.
- Modificar condiciones de iluminación interior/exterior.
- Anotación de instancia de píxeles.
- Posibilidad de generar mapa en profundidad.
- Nube de puntos con simulación de todos los parámetros de detección 3D.
- Control sobre la distribución de la cantidad de los objetos.
- Crear entornos y distribución de objetos.
- Control sobre obstáculos y visibilidad de piezas.

Otros modelos prometedores de segmentación

Los modelos de difusión están inspirados en la termodinámica de no equilibrio. Definen una cadena de Markov de pasos de difusión para agregar lentamente ruido aleatorio a los datos y luego aprenden a invertir el proceso de difusión para construir las muestras de datos deseadas a partir del ruido. A diferencia de otros modelos, los modelos de difusión se aprenden con un procedimiento fijo y la variable latente tiene una alta dimensionalidad(Weng, 2021). El rendimiento superior de los modelos de difusión los ha convertido en una herramienta atractiva; los modelos de difusión también pueden servir como un instrumento para la segmentación semántica, especialmente cuando los datos etiquetados son escasos. Estos modelos son una opción excelente para realizar la segmentación de objetos, debido a que sus activaciones capturan efectivamente la información semántica de una imagen de entrada(Baranchuk et al., 2022).

6. Glosario

Algoritmo: Un algoritmo es un conjunto de instrucciones paso a paso para completar una tarea.

Datos: Los datos son información que las computadoras almacenan y procesan.

Datos entrenamiento: Un conjunto de datos de entrenamiento es un conjunto de datos que las máquinas procesan para aprender.

Modelo: Un modelo hace referencia a un algoritmo que se ha entrenado en función de un conjunto de datos para reconocer ciertos tipos de patrones y hacer predicciones.

Modelo: Un modelo está diseñado para que una computadora se comporte como un sistema del mundo real.

Aprendizaje automático (machine learning): El aprendizaje automático es un tipo de IA que aprende, mediante ejemplos, a reconocer patrones y resolver problemas.

Ciencia de datos: La ciencia de datos es un campo amplio que usa enormes cantidades de información disponible para proporcionar conocimientos significativos. Aunque puede emplear herramientas de IA para el análisis, no siempre lo hace.

IA: La IA débil, o inteligencia artificial estrecha, incluye aplicaciones de algoritmos de IA para realizar tareas específicas. La IA débil es la que tenemos ahora. La IA avanzada, o inteligencia artificial general (IAG), es cuando una máquina puede razonar en su totalidad como un ser humano. La IA avanzada aún debe lograrse.

Aprendizaje supervisado: El aprendizaje supervisado es un método de entrenamiento que emplea conjuntos de datos etiquetados en el que se dan ejemplos de pares de entrada-salida a un algoritmo. Un conjunto de imágenes que se identifica como “gato” o “no gato” es un ejemplo de un conjunto de datos etiquetado.

Aprendizaje no supervisado: El aprendizaje no supervisado es un método de entrenamiento que no usa etiquetas ni salidas correctas. El algoritmo descubre la estructura de los datos por sí mismo.

La adaptabilidad es la habilidad de mejorar el desempeño aprendiendo a través de la experiencia. La autonomía es la habilidad de realizar tareas en entornos complejos sin la guía constante de un usuario.

Red Neuronal: Una red neuronal es un sistema inspirado por la biología humana que emplea una red de algoritmos para comprender datos de entrada y traducirlos en valores de salida. Las redes neuronales cuentan con conjuntos de datos de entrenamiento para aprender y mejorar con el tiempo.

Deep learning: El aprendizaje profundo es un subconjunto de aprendizaje automático que usa redes neuronales con un mínimo de tres capas. Una capa es un paso del procesamiento de datos. Un deepfake es una imagen, un video o un clip de audio alterado o completamente creado mediante el uso de herramientas de inteligencia artificial.

Visión por computadora: La visión computarizada es un campo de la IA que posibilita a las computadoras emplear datos para reconocer imágenes, videos y otros estímulos visuales. Un programa de reconocimiento de imágenes puede identificar si personas y ciertos objetos aparecen en imágenes o videos a través de algoritmos entrenados y un sistema de cámaras.

7. Bibliografía

- 3DHumanGAN: Towards Photo-Realistic 3D-Aware Human Image Generation.* (n.d.). Retrieved January 12, 2023, from <https://3dhumangan.github.io/>
- Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, 8(1), 53. <https://doi.org/10.1186/s40537-021-00444-8>
- Anderson, M. (2022, June 7). GAN as a Face Renderer for 'Traditional' CGI. *Unite.AI*. <https://www.unite.ai/gan-as-a-face-renderer-for-traditional-cgi/>
- AWS Pricing Calculator.* (n.d.). Retrieved October 8, 2022, from <https://calculator.aws/#/>
- Bakhtiarnia, A., Zhang, Q., & Iosifidis, A. (2022). *Efficient High-Resolution Deep Learning: A Survey* (arXiv:2207.13050). arXiv. <http://arxiv.org/abs/2207.13050>
- Baranchuk, D., Rubachev, I., Voynov, A., Khrulkov, V., & Babenko, A. (2022). *Label-Efficient Semantic Segmentation with Diffusion Models* (arXiv:2112.03126). arXiv. <https://doi.org/10.48550/arXiv.2112.03126>
- Boesch, G. (2022, August 11). *YOLOv7: The Most Powerful Object Detection Algorithm (2022 Guide)*. Viso.Ai. <https://viso.ai/deep-learning/yolov7-guide/>
- Brownlee, J. (2017, July 2). Gentle Introduction to the Adam Optimization Algorithm for Deep Learning. *MachineLearningMastery.Com*. <https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/>
- Caelles, S., Maninis, K.-K., Pont-Tuset, J., Leal-Taixe, L., Cremers, D., & Van Gool, L. (2017). One-Shot Video Object Segmentation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 5320–5329. <https://doi.org/10.1109/CVPR.2017.565>
- Cardinale, F. (2019, June 27). A Deep Learning based magnifying glass. *Idealo Tech Blog*. <https://medium.com/idealo-tech-blog/a-deep-learning-based-magnifying-glass-dae1f565c359>
- Ch. 6—Object Detection and Segmentation.* (n.d.). Retrieved December 29, 2022, from <https://manipulation.csail.mit.edu/segmentation.html>
- Cieślik, J. (2022, July 21). *How to Do Data Exploration for Image Segmentation and Object Detection (Things I Had to Learn the Hard Way)*. Neptune.Ai. <https://neptune.ai/blog/data-exploration-for-image-segmentation-and-object-detection>
- Dan Oved. (2019, November). [Updated] BodyPix: Real-time Person Segmentation in the Browser with TensorFlow.js. *BodyPix: Real-Time Person Segmentation in the Browser with TensorFlow.Js*. <https://blog.tensorflow.org/2019/11/updated-bodypix-2.html>
- EL MUNDO. (2021, November 13). *Una nueva masacre en la cárcel de Guayaquil deja 68 reos fallecidos y 25 heridos*. ELMUNDO. <https://www.elmundo.es/internacional/2021/11/13/619001bcfc6c83f66b8b458b.html>
- EL PAÍS. (2021, December 30). *Masacres carcelarias y 2.330 asesinatos en el año más sangriento de la década en Ecuador | Internacional | EL PAÍS*. <https://elpais.com/internacional/2021-12-30/masacres-carcelarias-y-2330-asesinatos-en-el-ano-mas-sangriento-de-la-decada-en-ecuador.html>
- Equilibrio de Nash. (2022). In *Wikipedia, la enciclopedia libre*. https://es.wikipedia.org/w/index.php?title=Equilibrio_de_Nash&oldid=146829794

- Facebookresearch/detectron2.* (2022). [Python]. Meta Research.
https://github.com/facebookresearch/detectron2/blob/857d5de21a7789d1bba46694cf608b1cb2ea128a/MODEL_ZOO.md (Original work published 2019)
- Farnia, F., & Ozdaglar, A. (2020). Do GANs always have Nash equilibria? *Proceedings of the 37th International Conference on Machine Learning*, 3029–3039.
<https://proceedings.mlr.press/v119/farnia20a.html>
- Ferret. (2016, June 1). Video camaras de vigilancia conceptos y debate etico. *Camaras de seguridad, camaras IP, KIT de camaras, camaras espia, CCTV, camaras de vigilancia.*
<https://topsecurityperu.com/video-camaras-de-vigilancia-conceptos-y-debate-etico/>
- GAN variations and timelines | Hands-On Neural Networks.* (n.d.). Retrieved November 6, 2022, from
<https://subscription.packtpub.com/book/data/9781788992596/10/ch10lvl1sec44/gan-variations-and-timelines>
- Giddens, K. (2019, November 30). World's First Synthetic Gun Detection Dataset from Edgecase.ai. *Medium.* https://medium.com/@kyle_68514/worlds-first-synthetic-gun-detection-dataset-from-edgecase-ai-dbe3ea8eeb7e
- Girshick, R. (2015). *Fast R-CNN* (arXiv:1504.08083). arXiv.
<https://doi.org/10.48550/arXiv.1504.08083>
- Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). *Rich feature hierarchies for accurate object detection and semantic segmentation* (arXiv:1311.2524). arXiv.
<https://doi.org/10.48550/arXiv.1311.2524>
- Gutta, S. (2021, August 4). Object Detection Algorithm—YOLO v5 Architecture. *Analytics Vidhya.* <https://medium.com/analytics-vidhya/object-detection-algorithm-yolo-v5-architecture-89e0a35472ef>
- He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2018). *Mask R-CNN* (arXiv:1703.06870). arXiv.
<https://doi.org/10.48550/arXiv.1703.06870>
- Horan, R., & Lavelle, M. (n.d.). *Proof by Induction.* 34.
- Iqbal, A., Sharif, M., Yasmin, M., Raza, M., & Aftab, S. (2022). Generative adversarial networks and its applications in the biomedical image segmentation: A comprehensive survey. *International Journal of Multimedia Information Retrieval*, 11(3), 333–368.
<https://doi.org/10.1007/s13735-022-00240-x>
- Jeong, J. J., Tariq, A., Adejumo, T., Trivedi, H., Gichoya, J. W., & Banerjee, I. (2022). Systematic Review of Generative Adversarial Networks (GANs) for Medical Image Classification and Segmentation. *Journal of Digital Imaging*, 35(2), 137–152.
<https://doi.org/10.1007/s10278-021-00556-w>
- Jocher, G. (2020). *YOLOv5 by Ultralytics* (7.0) [Python].
<https://doi.org/10.5281/zenodo.3908559>
- Kazeminia, S., Baur, C., Kuijper, A., van Ginneken, B., Navab, N., Albarqouni, S., & Mukhopadhyay, A. (2020). GANs for medical image analysis. *Artificial Intelligence in Medicine*, 109, 101938. <https://doi.org/10.1016/j.artmed.2020.101938>
- Kim, M., Zimmermann, T., DeLine, R., & Begel, A. (2018). Data Scientists in Software Teams: State of the Art and Challenges. *IEEE Transactions on Software Engineering*, 44(11), 1024–1038. <https://doi.org/10.1109/TSE.2017.2754374>
- Langr, J., & Vladimir, B. (2019a). *Chapter 1. Introduction to GANs* (First). O'Reilly.
https://learning.oreilly.com/library/view/gans-in-action/9781617295560/Text/kindle_split_010_split_000.html

- Langr, J., & Vladimir, B. (2019b). *Chapter 5. Training and common challenges: GANing for success*. O'Reilly. https://learning.oreilly.com/library/view/gans-in-action/9781617295560/Text/kindle_split_015_split_000.html
- leverxgroup/esrgan: Enhanced SRGAN. Champion PIRM Challenge on Perceptual Super-Resolution. (n.d.). Retrieved January 11, 2023, from <https://github.com/leverxgroup/esrgan>
- Linear Semantics in Generative Adversarial Networks. (n.d.). Retrieved October 8, 2022, from <https://atlantixjj.github.io/LinearSemanticsGAN/>
- Mady), M. S. (. (2019, March 15). Ch 14.2 Pix2Pix Gan and Cycle Gan. *Deep Math Machine Learning.Ai*. <https://medium.com/deep-math-machine-learning-ai/ch-14-2-pix2pix-gan-and-cycle-gan-55cd84318fb8>
- Manifesto for Agile Software Development. (n.d.). Retrieved November 4, 2020, from <https://agilemanifesto.org/>
- Marcos Gómez-Puerta. (2018). *Cómo realizar el apartado de método en un TFG/TFM*.
- marktab. (2020). *What is the Team Data Science Process?* <https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/overview>
- Overview—Roboflow. (n.d.). Retrieved December 14, 2022, from <https://docs.roboflow.com/>
- Pandey, M., Fernandez, M., Gentile, F., Isayev, O., Tropsha, A., Stern, A. C., & Cherkasov, A. (2022). The transformational role of GPU computing and deep learning in drug discovery. *Nature Machine Intelligence*, 4(3), Article 3. <https://doi.org/10.1038/s42256-022-00463-x>
- Papers with Code—PatchGAN Explained. (n.d.). Retrieved January 10, 2023, from <https://paperswithcode.com/method/patchgan>
- Preprocess Images for Deep Learning—MATLAB & Simulink—MathWorks España. (n.d.). Retrieved December 15, 2022, from <https://es.mathworks.com/help/images/preprocess-images-for-deep-learning.html>
- Primicias. (2021, October). Ecuador camina hacia la tasa de muertes violentas más alta desde 2012. *Primicias*. <https://www.primicias.ec/noticias/sociedad/ecuador-tasa-muertes-violentas-alta/>
- Procesamiento digital de imágenes. (2022). In *Wikipedia, la enciclopedia libre*. https://es.wikipedia.org/w/index.php?title=Procesamiento_digital_de_im%C3%A1genes&oldid=147937050
- Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 779–788. <https://doi.org/10.1109/CVPR.2016.91>
- Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks (arXiv:1506.01497). arXiv. <https://doi.org/10.48550/arXiv.1506.01497>
- Rouhiainen, L. (2019). *Inteligencia artificial: 101 cosas que debes saber hoy sobre nuestro futuro*. Alienta Editorial.
- Rubin, K. S. (2012). *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. Addison-Wesley.
- Sara, U. (2019). Comparative Study Of Different Quality Assessment Techniques On Color Images. *IREE Journals*, 2(11), 127–133.
- Schleier-Smith, J. (2015). An Architecture for Agile Machine Learning in Real-Time Applications. *KDD '15*. <https://doi.org/10.1145/2783258.2788628>

- Segmentación (procesamiento de imágenes). (2022). In *Wikipedia, la enciclopedia libre*. [https://es.wikipedia.org/w/index.php?title=Segmentaci%C3%B3n_\(procesamiento_de_im%C3%A1genes\)&oldid=145476584](https://es.wikipedia.org/w/index.php?title=Segmentaci%C3%B3n_(procesamiento_de_im%C3%A1genes)&oldid=145476584)
- Significados. (n.d.). Retrieved January 9, 2023, from <https://www.significados.com/>
- Steps of the Scientific Method. (n.d.). Retrieved March 19, 2022, from <https://www.sciencebuddies.org/science-fair-projects/science-fair/steps-of-the-scientific-method>
- Typical Generative Adversarial Networks (GAN) architecture. | Download Scientific Diagram.* (n.d.). Retrieved November 5, 2022, from https://www.researchgate.net/figure/Typical-Generative-Adversarial-Networks-GAN-architecture_fig2_349182009
- Vyas, K. (2020, April 26). Object Segmentation. *VisionWizard*. <https://medium.com/visionwizard/object-segmentation-4fc67077a678>
- Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., Loy, C. C., Qiao, Y., & Tang, X. (2018). *ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks* (arXiv:1809.00219). arXiv. <https://doi.org/10.48550/arXiv.1809.00219>
- Weng, L. (2021, July 11). *What are Diffusion Models?* <https://lilianweng.github.io/posts/2021-07-11-diffusion-models/>
- What Do These Different AP Values Mean.* (2020, January 15). Julius' Data Science Blog. <https://jss367.github.io/what-do-these-different-ap-values-mean.html>
- Wu, H., Liu, Q., & Liu, X. (2019). A Review on Deep Learning Approaches to Image Classification and Object Segmentation. *Computers, Materials & Continua*, 60(2), 575–597. <https://doi.org/10.32604/cmc.2019.03595>
- Xu, J., & Zheng, C. (2021). *Linear Semantics in Generative Adversarial Networks* (arXiv:2104.00487). arXiv. <https://doi.org/10.48550/arXiv.2104.00487>
- Xun, S., Li, D., Zhu, H., Chen, M., Wang, J., Li, J., Chen, M., Wu, B., Zhang, H., Chai, X., Jiang, Z., Zhang, Y., & Huang, P. (2022). Generative adversarial networks in medical image segmentation: A review. *Computers in Biology and Medicine*, 140, 105063. <https://doi.org/10.1016/j.compbiomed.2021.105063>
- YOLO Object Detection Explained: A Beginner's Guide.* (n.d.). Retrieved December 18, 2022, from <https://www.datacamp.com/blog/yolo-object-detection-explained>
- Zhang, Y., Tian, Y., Kong, Y., Zhong, B., & Fu, Y. (2018). *Residual Dense Network for Image Super-Resolution* (arXiv:1802.08797). arXiv. <https://doi.org/10.48550/arXiv.1802.08797>
- Zou, K. H., Warfield, S. K., Bharatha, A., Tempany, C. M. C., Kaus, M. R., Haker, S. J., Wells, W. M., Jolesz, F. A., & Kikinis, R. (2004). Statistical Validation of Image Segmentation Quality Based on a Spatial Overlap Index. *Academic Radiology*, 11(2), 178–189. [https://doi.org/10.1016/S1076-6332\(03\)00671-8](https://doi.org/10.1016/S1076-6332(03)00671-8)

8. Anexos

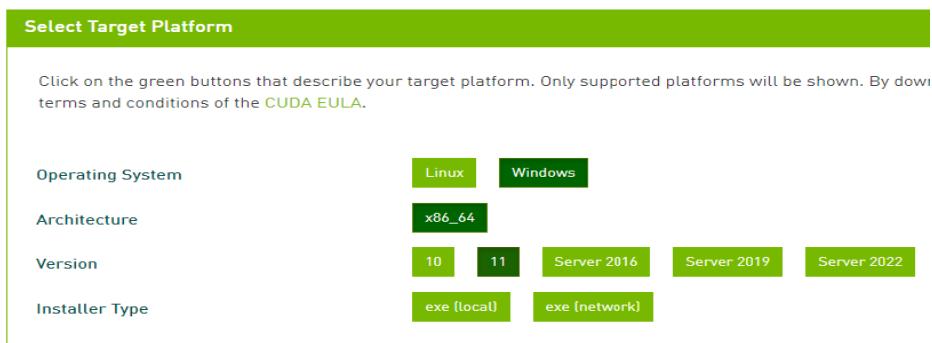
8.1. Instalar Cuda 11.8 y cuDDN 11.x

La configuración desarrollada a continuación se realiza tomando como base, las siguientes referencias:

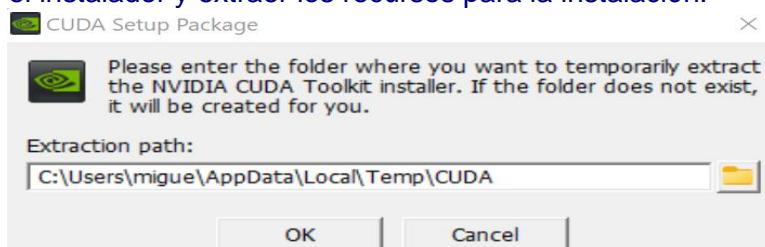
- ✓ https://docs.nvidia.com/cuda/pdf/CUDA_Installation_Guide_Windows.pdf
- ✓ <https://medium.com/geekculture/install-cuda-and-cudnn-on-windows-linux-52d1501a8805>

Pasos:

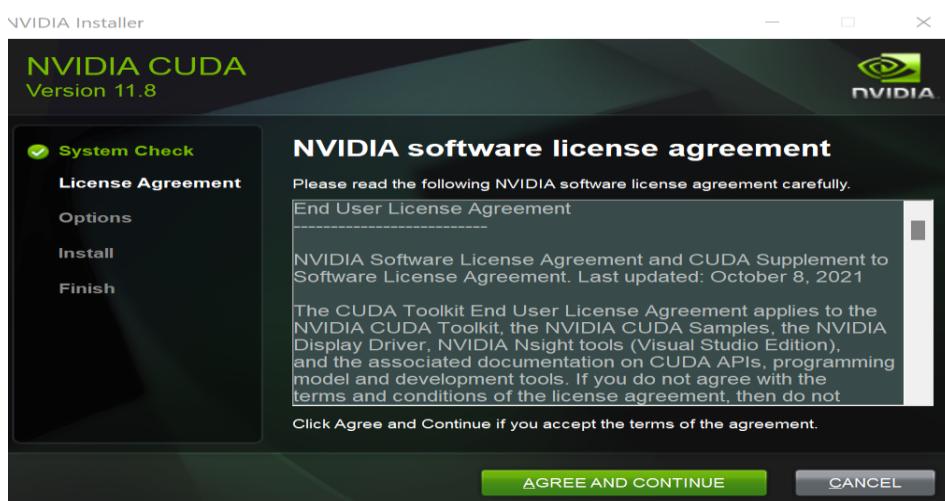
1. Descargar el instalador de la url y configurar el tipo de descarga.
 - ✓ <https://developer.nvidia.com/cuda-downloads>



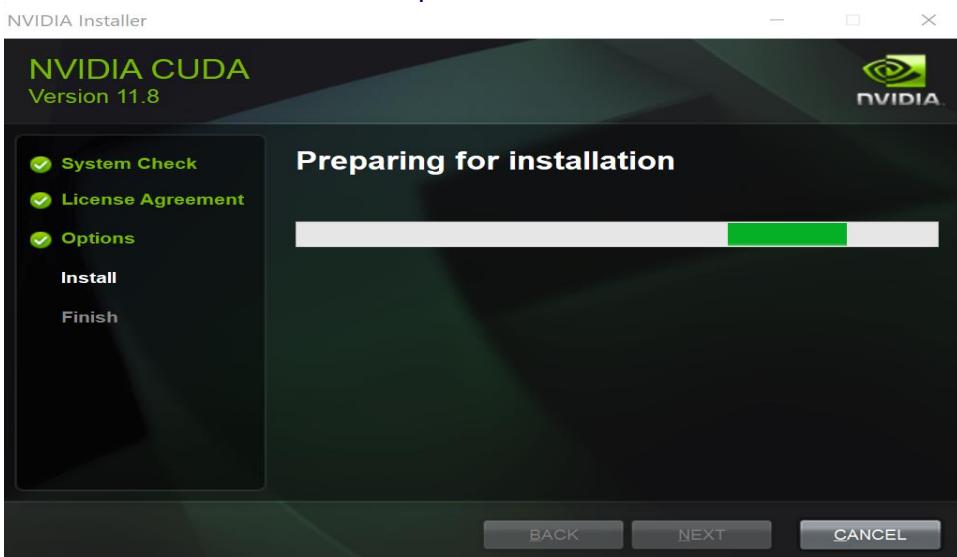
2. Ejecutar el instalador y extraer los recursos para la instalación.



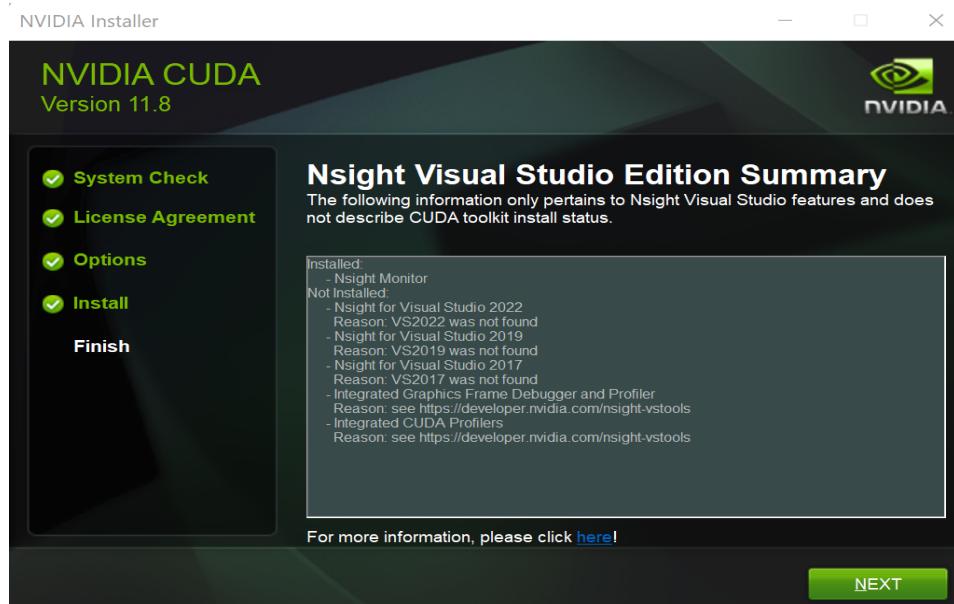
3. Aceptamos los términos de licencia.



4. Realizamos la instalación por defecto.



5. Finalizamos la instalación



6. Verificar que se encuentren correctamente creadas las rutas de CUDA

System variables	
Variable	Value
ACSetupSvcPort	23210
ACSvcPort	17532
ComSpec	C:\Windows\system32\cmd.exe
CUDA_PATH	C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.8
CUDA_PATH_V11_8	C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.8
DriverData	C:\Windows\System32\Drivers\DriverData
JAVA_HOME	C:\Program Files\Java\graalvm
M2_HOME	D:\MAVEN\apache-maven-3.8.6

7. Descargar el paquete de cuDNN(GPU-accelerated library of primitives for deep neural networks) y descomprimir.

✓ <https://developer.nvidia.com/rdp/cudnn-download>

cuDNN Download

NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks.

I Agree To the Terms of the cuDNN Software License Agreement

Note: Please refer to the [Installation Guide](#) for release prerequisites, including supported GPU architectures and compute capabil

For more information, refer to the cuDNN Developer Guide, Installation Guide and Release Notes on the [Deep Learning SDK Docu](#)

[Download cuDNN v8.6.0 \[October 3rd, 2022\], for CUDA 11.x](#)

Local Installers for Windows and Linux, Ubuntu(x86_64, armsbsa)

[Local Installer for Windows \[Zip\]](#)

[Local Installer for Linux x86_64 \[Tar\]](#)

[Local Installer for Linux PPC \[Tar\]](#)

8. Copiar el contenido de la carpeta en la carpeta donde está instalado CUDA.

Carpeta Origen:

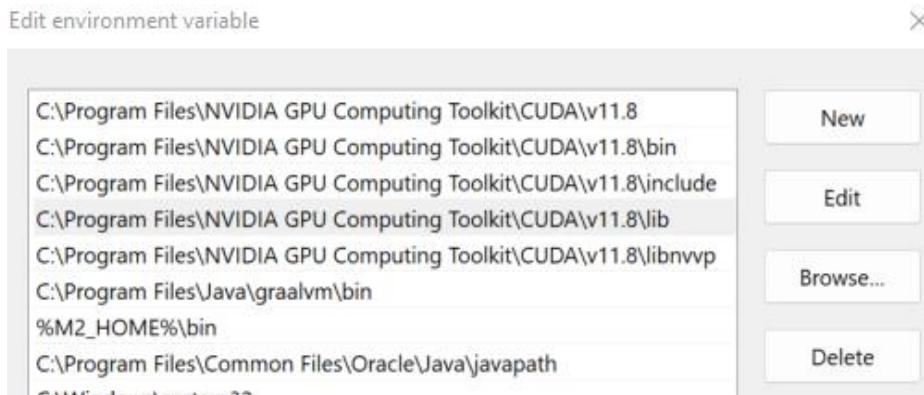
cudnn-windows-x86_64-8.6.0.163_cuda11-archive		
	New	...
← → ↑	📁 This PC > Data (D:) > cudnn > cudnn-windows-x86_64-8.6.0.163_cuda11-archive	Sort View ...
▼ This PC	Name	Date modified
> Desktop	bin	9/19/2022 5:31 PM
> Documents	include	9/19/2022 5:31 PM
> Downloads	lib	9/19/2022 5:31 PM
...	LICENSE	8/8/2022 2:38 PM

Carpeta destino:

v11.8		
	New	...
← → ↑	📁 OS (C:) > Program Files > NVIDIA GPU Computing Toolkit > CUDA > v11.8	Sort View ...
▼ This PC	Name	Date modified
> OneDrive - Perso	bin	11/15/2022 9:28 PM
> Desktop	compute-sanitizer	11/15/2022 9:28 PM
> Documents	extras	11/15/2022 9:28 PM
...	include	11/15/2022 9:28 PM
	lib	11/15/2022 9:28 PM

9. Crear las variables de entorno:

- ✓ C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.8
- ✓ C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.8\bin
- ✓ C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.8\include
- ✓ C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.8\lib



8.2. Instalar Python 3.11.10

La configuración desarrolla a continuación se realiza tomando como base, las siguientes referencias:

- ✓ <https://www.tensorflow.org/install/gpu>

Pasos:

1. Descargar e instalar visual Studio C++ 2015-2022 Redistributable
 - ✓ <https://learn.microsoft.com/es-ES/cpp/windows/latest-supported-vc-redist?view=msvc-170>
2. Descargar e instalar Python (3.11.10 – lastest realease)
 - ✓ <https://www.python.org/downloads/windows/>

8.3. Configurar un entorno con Anaconda y Jupyter

La configuración desarrolla a continuación se realiza tomando como base, las siguientes referencias:

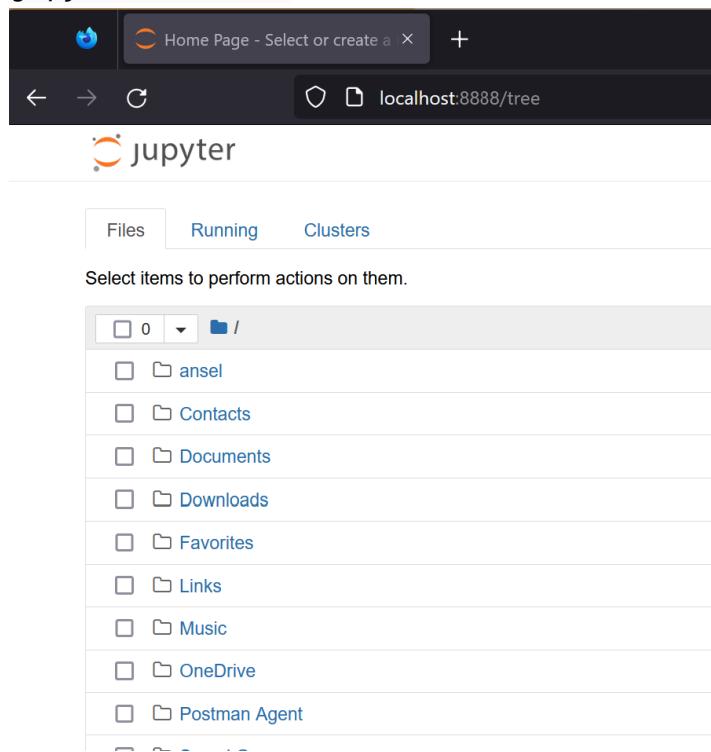
- ✓ <https://www.e2enetworks.com/blog/how-to-setup-nvidia-gpu-enabled-deep-learning-development-environment-with-cuda-gpu-anaconda-jupyter-keras-and-tensorflow-for-windows>

Pasos:

1. Descargar e instalar Anaconda
 - ✓ <https://www.anaconda.com/products/distribution>



2. Para configurar la instalación de notebooks, ejecutar en una terminal
`pip install notebook`
3. Correr la instancia, luego cerrar la ventana con la combinación CTRL + C
`jupyter notebook`



4. Crear la variable de entorno para que Anaconda se ejecute desde cualquier terminal:
✓ `C:\ProgramData\Anaconda3\Scripts`

Edit environment variable X

C:\ProgramData\Anaconda3\Scripts
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.8

New

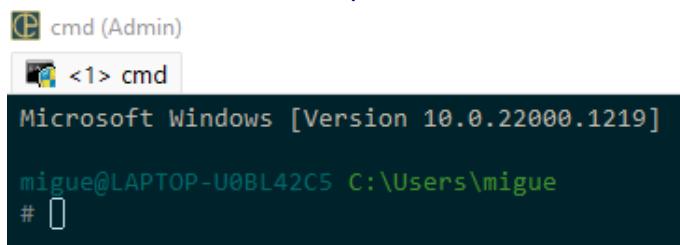
8.4. Configurar un entorno de trabajo

La configuración desarrollada a continuación se realiza tomando como base, las siguientes referencias:

- ✓ <https://lifewithdata.com/2022/01/16/how-to-install-tensorflow-and-keras-with-gpu-support-on-windows/>

Pasos:

1. Se debe tener una terminal con permisos de administrador



```
cmd (Admin)
<1> cmd
Microsoft Windows [Version 10.0.22000.1219]
migue@LAPTOP-U0BL42C5 C:\Users\migue
#
```

2. Para configurar un entorno de trabajo, ejecutar en la terminal
`conda create --name tf_tfm_gans python=3.9`
3. Activar el entorno, se debe ejecutar:
`conda activate tf_tfm_gans`
4. Instalar tensorflow, se debe ejecutar:
`pip install tensorflow`
5. Instalar jupyter-notebook, se debe ejecutar:
`pip install notebook`
6. Para que Jupyter encuentre el entorno se debe ejecutar
`python -m ipykernel install --user --name=tf_tfm_gans`
7. Instalar requerimientos
`pip install -r requirements.txt`
8. Correr la instancia
`jupyter notebook --notebook-dir="D:\proyectotfm\tfm-laboratory-test-results"`
9. Correr la instancia local con colab
`jupyter notebook --notebook-dir="D:\proyectotfm\tfm-laboratory-test-results" --NotebookApp.allow_origin='https://colab.research.google.com' --port=8888 --NotebookApp.port_retries=0`

The screenshot shows a Jupyter Notebook interface running on a local host. At the top, there's a browser window titled "Home Page - Select or create a notebook" with the URL "localhost:8888/tree". Below it, the Jupyter logo is visible. A navigation bar includes "Files", "Running", and "Clusters". A message says "Select items to perform actions on them." Below this is a file list with "0" files and one item named "TestGPU.ipynb".

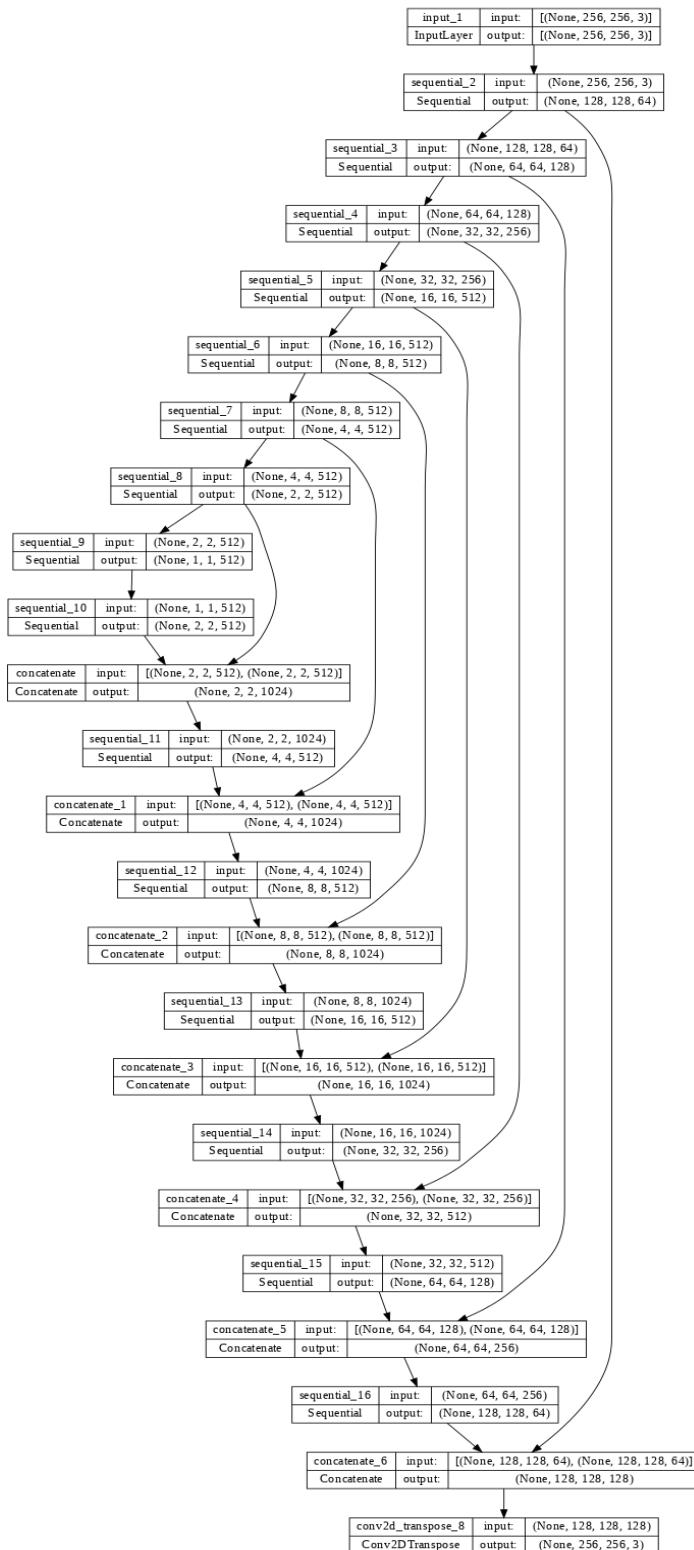
The main area of the interface is a code editor with two cells:

```
In [1]: import tensorflow as tf
In [2]: print(f"Tensorflow version: {tf.__version__}")
       print(f"Keras Version: {tf.keras.__version__}")
       print("GPU is", "available" if tf.config.list_physical_devices('GPU') else "NOT AVAILABLE")
```

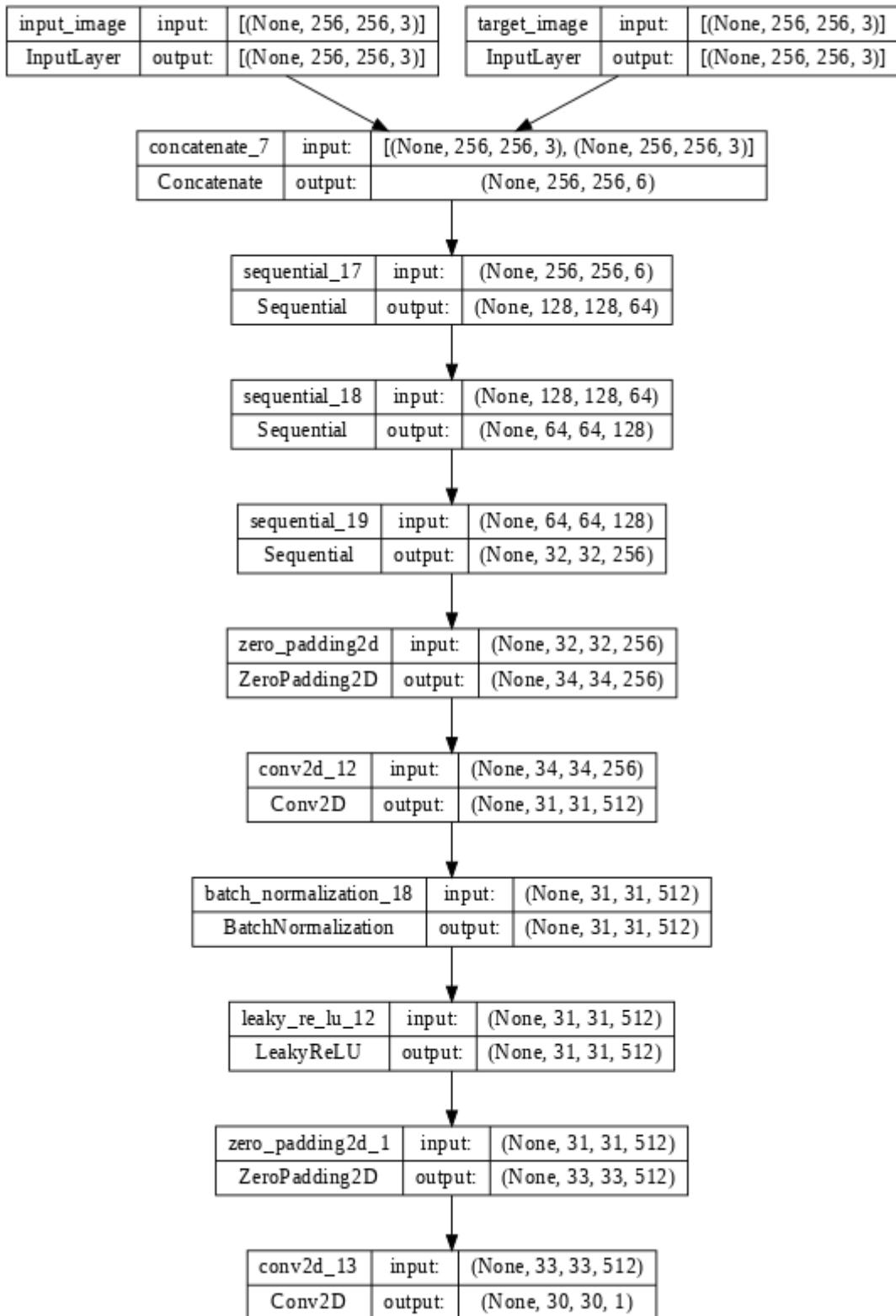
The output of the second cell is:

```
Tensorflow version: 2.10.0
Keras Version: 2.10.0
GPU is available
```

8.5. Arquitectura Generador



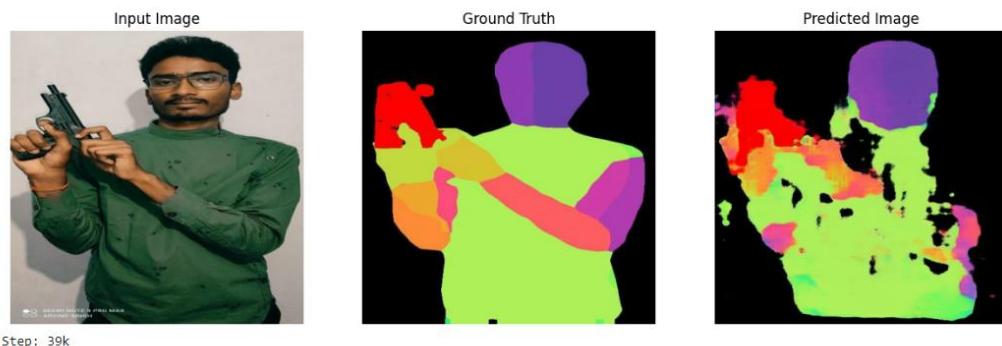
8.6. Arquitectura Discriminador



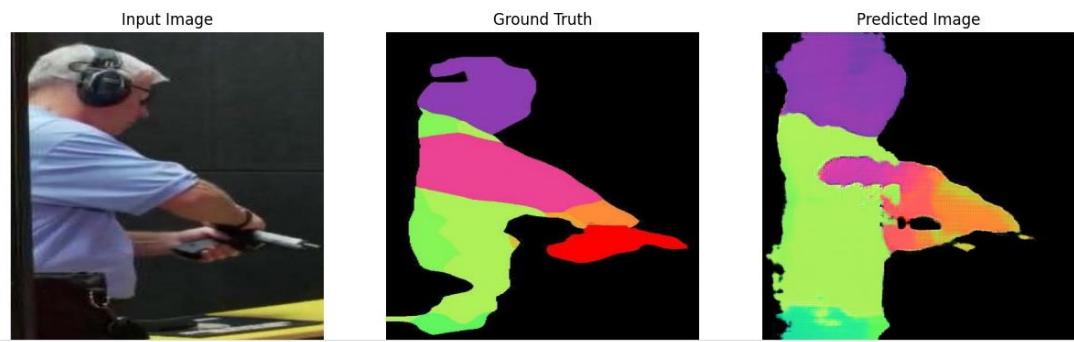
8.7. Otras pruebas

Sin jitter y pocas imágenes con 512x512

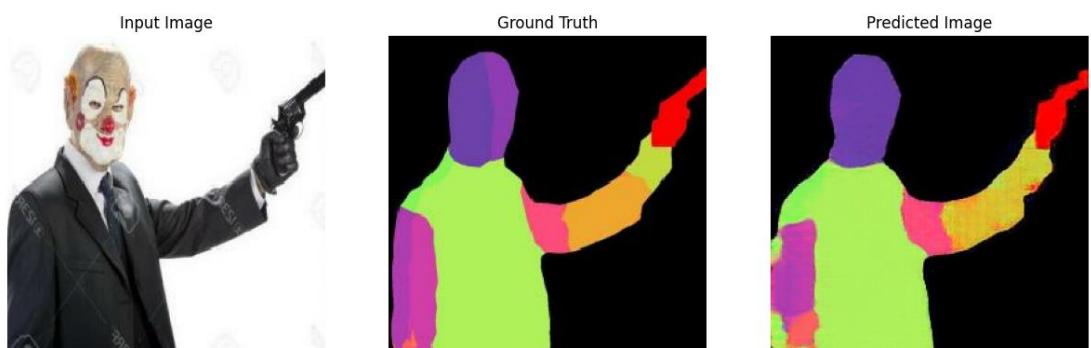
Time taken for 1000 steps: 156.52 sec



Con Jitter, imágenes con arma y sin arma 512x512 con datos aumentados.



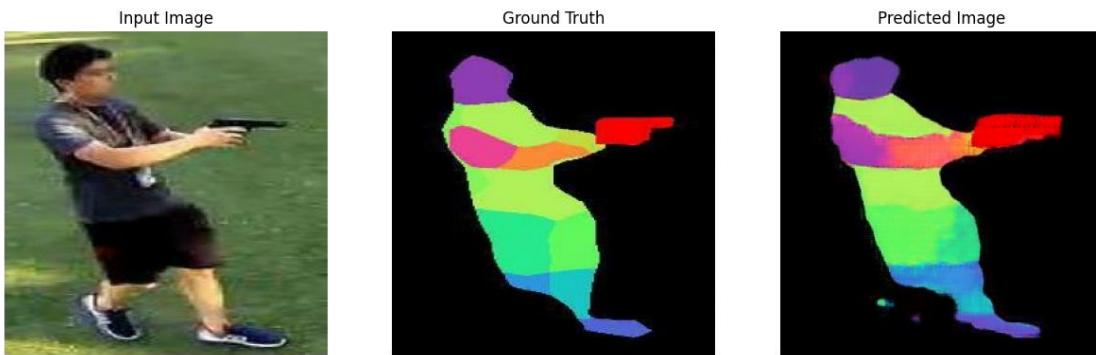
Con Jitter, imágenes con arma y sin arma 256x256, presenta los mejores resultados en la red.



Para todos los casos anteriores se identificó que los problemas de entrenamiento estaban relacionados con la herramienta Roboflow que incluye opciones de auto orientación automática y al incluir conversiones o procesos adicionales se puede presentar estos errores. Adicionalmente en un conjunto de datos existía una imagen que se repetía varias veces, por tanto, tenía un efecto negativo durante el entrenamiento, pues, su selección era la más presente al momento de entrenar. Se optó por realizar varios notebooks adicionales

que permita el procesamiento de un conjunto de datos adicional dedicado a la detección de armas, se cambia la función Jitter para que no realice un zoom demasiado pronunciado sobre la imagen debido a que tiende a perder el arma segmentada. Ya que de manera general esta se encuentra en un costado de la imagen.

Con la red mejorada y eliminados las imágenes repetidas de un solo tipo.



Para agregar más imágenes se utiliza las fuentes:

- [YouTube-GDD](#), el cual es un conjunto de datos de detección de armas de YouTube (YouTube-GDD). Este incluye 343 videos de YouTube de alta definición y contiene 5000 imágenes bien elegidas, en las que se anotan 16064 instancias de armas y 9046 instancias de personas.
- [Human Pose](#): el cual es un conjunto de datos de referencia de última generación para la evaluación de la estimación de la pose humana articulada. El conjunto de datos incluye alrededor de 25 000 imágenes que contienen más de 40 000 personas con articulaciones corporales anotadas.

Para esta tarea se realizó los siguientes notebooks que se encuentran en el repositorio:

- ☞ [6_3TFMSegmentacionCuerpoPersonasConArmasCortasRDNDatasetYoutube.ipynb](#)
 - ☞ [7_2TFMCrearDataSetSegmentacionCuerpoPersonasSinArmas.ipynb](#)
 - ☞ [7_3TFMCrearDataSetImagenesYouTube_GDD.ipynb](#)