UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

DISEÑO, IMPLEMENTACIÓN Y EVALUACIÓN DE MODELOS MATEMÁTICOS PARA ESTIMAR DIÁMETROS Y NÚMERO DE TRONCOS EN BANCOS DE MADERA SOBRE CAMIÓN, MEDIANTE IMÁGENES

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERA CIVIL ELÉCTRICA

CECILIA NICOL IBARRA DÍAZ

PROFESOR GUÍA:
CARLOS MOLINA SÁNCHEZ

MIEMBROS DE LA COMISIÓN:
IVÁN CASTRO OJEDA
ANDRÉS CABA RUTTE

SANTIAGO DE CHILE
2019

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERA CIVIL ELÉCTRICA
POR: CECILIA NICOL IBARRA DÍAZ
FECHA: 2019
PROF. GUÍA: CARLOS MOLINA SÁNCHEZ

DISEÑO, IMPLEMENTACIÓN Y EVALUACIÓN DE MODELOS MATEMÁTICOS PARA ESTIMAR DIÁMETROS Y NÚMERO DE TRONCOS EN BANCOS DE MADERA SOBRE CAMIÓN, MEDIANTE IMÁGENES

Con el crecimiento del sector forestal y el aumento en la producción de madera aserrada, los procesos de control de calidad deben ser más rigurosos y eficientes. El control de calidad permite clasificar la madera según sus usos, generando un alto impacto en los precios de comercialización. Adicionalmente, permite generar registros de las cargas. Parte de los parámetros que requieren ser registrados en una carga de madera sobre camión, que ingresan a las plantas de aserraderos, son, el número de troncos en la carga, y sus diámetros.

Considerando los avances de los últimos años, en procesamiento de imágenes y visión artificial, y el aumento considerable en la capacidad de desarrollo de los procesadores, se propone usar modelos matemáticos para estimar diámetro y número de troncos en bancos de madera sobre camión, usando como fuente, imágenes de carga. La propuesta de desarrollo considera dos etapas. En la primera, se exploran modelos de detección de caras de troncos. Luego, en la segunda, se identifica el mejor modelo para el ajuste fino del contorno y así determinar el diámetro del tronco.

Para la primera etapa, utilizando Redes Neuronales Convolucionales se obtiene una tasa de detección del 97 % sobre el conjunto de validación. Los tiempos de ejecución del algoritmo son cercanos a medio segundos. En la segunda etapa, se comparan los resultados de diámetro obtenidos con una marca manual sobre la imagen. El mejor resultado se obtiene combinando Contornos Activos con preprocesamiento de imágenes con promedio del error de 3.2 y desviación estándar de 4.65 píxeles (9 %) de diferencia con la marca manual sobre un universo de 405 caras. Esta segunda etapa toma, el promedio, un minuto por cada carga.

Con el trabajo realizado, se observa que a pesar del bajo contraste de algunas de las caras de troncos dentro de la imagen, la exactitud en la detección de caras es de 97 % y la desviación en el cálculo de diámetro es de 4.65 píxeles. Además de las métricas de cálculo, es importante mencionar que, los tiempos de detección de caras son consistentes con un desarrollo en línea. Con el trabajo realizado se verifica la propuesta de utilizar redes neuronales y procesamiento de imágenes para conteo de troncos y estimación de diámetro.

Con la implementación de este tipo de propuestas en las plantas de aserradero se podría mantener un control de cantidad de troncos y diámetros de cada una de las cargas que ingresan utilizando como entrada una imagen sobre la cara de la madera. Se identifican lineamientos para continuar mejorando la solución propuesta, dentro de los que destaca la extensión de la base de datos para generar una red capaz de ser utilizada en todas las plantas de aserradero. Finalmente, el algoritmo desarrollado queda disponible para ser integrado con cámaras tridimensionales y además, ser integrado con un motor de manejo de software que permita obtener resultados en línea con el ingreso de un camión.

Desde antes de nacer, buscabas como darme lo mejor. Hemos trabajado juntos por cada uno de nuestros sueños. Dedicado a quienes hacen que cada momento pueda ser siempre mejor.
José Leonardo Ibarra Díaz, mi hermano.
Cecilia del Pilar Díaz Díaz, mi madre.
José Eulogio Ibarra Leyton, mi padre.
Bertina Del Carmen Díaz Varela Q.E.P.D, mi abuela.

Agradecimientos

Dentro de toda mi etapa de formación como Ingeniera Civil Eléctrica, hubieron muchas personas que contribuyeron al proceso y a la conclusión de este camino. Quiero agradecer en primer lugar a Carlos Molina, mi profesor guía, por su ayuda y disposición. Agradecer a todo el equipo de desarrollo por hacer de esta experiencia un camino de aprendizaje continuo y a la empresa Woodtech S.A. por la oportunidad de realizar mi memoria con ellos.

En segundo lugar, quiero agradecer a mi familia por apoyar cada una de mis decisiones, a mis padres por siempre entregarnos las mejores herramientas para afrontar el mundo, a mi hermano por su cariño incondicional. A mi amiga de la vida, Sandra Nogales, por su confianza y energía. A mi gran compañero de carrera, Octavio Pérez, por su compañía desde el primer día. A Ivan Castro, Roberto Aguirre y Bastian Salinas por hacer de mi vida una aventura. Finalmente agradecer a cada persona con la que he podido compartir y aprender.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Introducción

0.1. Motivación

El valor anual de la madera y los productos forestales en la economía mundial se estima en más de 400.000 millones de dólares. La producción, extracción, elaboración y comercio son fuentes importantes de ingresos y empleo especialmente en zonas rurales. Solo en Chile el 55 % de la superficie continental (41.3 mill. ha) corresponden a terrenos forestales (bosques, humedales, praderas naturales y matorrales). Las plantaciones forestales corresponden al 17,2 % del total de bosques en Chile. [20]

El sector forestal, al año 2010, participaba con el 3.1 % del PIB Nacional, con exportaciones que superan los US$5.000 millones anuales (promedio entre 2007 y 2011). El 85 % lo conforman productos de alto valor agregado, destacando la celulosa, los tableros y la madera aserrada. En el área de celulosa se separa la celulosa presente en la madera para generar distintos tipos de papel. En paneles se crean planchas de madera o aglomerados. En madera aserrada es donde se obtienen productos de madera terminados o productos intermedios. [2]

Dentro de las tres Áreas estratégicas mencionada, la que concentra los mayores requerimientos de calidad es la de Aserraderos y cuya producción al año 2017 alcanzó los 7.9 millones de m^3. Con el aumento en la producción de madera aserrada, los procesos de control de calidad deben ser más rigurosos y eficiencia. El control de calidad impacta directamente en los costos de producción de las Plantas de Aserraderos ya que pueden pagar el costo adecuado por la madera que adquieren.

Charlie Thomas VP. Operaciones en la planta de Shuqualak, Mississippi, afirma: "por cada dolar ganado el 65 % corresponde a costos de la madera, es relevante tener un control eficaz de las cargas que ingresan". Danny White, presidente y director general de TR miller Mill co., Inc comenta: "medir entre 30 a 50 % de las cargas que ingresan permite asegurarse de obtener exactamente el producto por el que se está pagando". Ambos concuerdan en la importancia de un proceso de evaluación y selección.

En Shuqualak detectar camiones con cargas por debajo de las especificaciones podría tomar entre dos a tres meses. En TR miller Mill solo es posible tomar mediciones del 20 % de las cargas que ingresan por falta de espacio. El proceso de control de calidad requiere de un lugar para descargar los bancos de madera y la utilización de grúas (ver fig. 1). Posteriormente se obtienen los datos de largo, diámetro y algunas características de los principales defectos

de la madera tronco por tronco.

Figura 1: Proceso de descarga de camión para control de calidad

En la industria forestal el control de calidad de la madera realizado con métodos no destructivos (evaluar la calidad sin afectar su integridad) permiten clasificar la madera según su uso, de manera confiable y económica. Generando un alto impacto en los precios de comercialización de la madera. En relación a los estándares de dimensiones, la gama de opciones que ofrece la madera aserrada es muy amplia y existen diferentes normas, dependiendo del país de origen, la especie y las adaptaciones resultantes para surtir el mercado.

Actualmente el control de calidad se realiza muestreando de forma aleatoria cargas de camiones que ingresan a la planta, descargando (imagen 1 y tomando las dimensiones de manera manual para cada uno de los troncos. El gran inconveniente de la medición manual es que el error de cálculo típico del volumen de una carga oscila entre un 5 % y un 10 %, y siempre en contra del comprador. En otras palabras, si una planta de celulosa gasta US$100 millones al año en madera puede estar perdiendo entre US$5 y US$10 millones anuales.

Una mejor alternativa al control de calidad para la madera fue desarrollada por la empresa Chilena, Woodtech S.A. Consiste en automatizar el control de calidad de la madera utilizando un sistema de medición láser que, en el tiempo que le tarda al camión cruzar un portal, toma mediciones y crea una imagen tridimensionales del camión y su carga. Este sistema permite el cálculo de volumen de la madera y además entrega información de cuán rectos o deformados están los troncos.

En el proceso de operación es posible aumentar la precisión, reducir costos en materia de personal y espacio, disminuir los tiempos de espera por camión y disminuir la posibilidad de accidentes. Adicionalmente es posible crear registros computacionales de las cargas, estando disponible en caso de ser necesarias futuras consultas. A nivel de clientes permite mejorar la relación con los proveedores al obtener mediciones objetivas de la calidad de la madera y motivándolos a aumentar la vigilancia a los productos para cumplir con los estándares. [25]

Considerando los avances de los últimos años, en los que se han consolidado metodologías inspiradas en los sistemas biológicos y ha aumentado considerablemente la capacidad de desarrollo de los procesadores, es posible obtener soluciones robustas y de fácil implementación. Adicionalmente, las técnicas de procesamiento de imágenes y métodos de segmentación automáticos representan nuevas formas de mejorar los resultados conocidos. Es aquí donde nade el marco para el trabajo realizado.

En conjunto con la empresa Woodtech S.A se busca determinar la factibilidad técnica

de utilizar modelos matemáticos para el procesamiento de imágenes de bancos de maderas, que permita obtener resultados similares a los de Logmeter en el conteo y determinación del diámetro de la carga. Adicionalmente, con la utilización de imágenes se busca reducir el costo de equipos e instalación, para llegar a clientes de menor tamaño, manteniendo una solución automatizada.

0.2. Objetivos

El desarrollo de este trabajo tiene como objetivo principal diseñar y desarrollar un sistema basado en imágenes que permita estimar diámetros y número de troncos en bancos de madera sobre camiones. Tomando como referencia los resultados actuales de Logmeter, y considerando que es posible obtener menor certeza en compensación a un menor costo de implementación. Para el cumplimiento de este objetivo se plantean los siguientes objetivos específicos:

- Comprender el funcionamiento de los diferentes modelos matemáticos para preprocesamiento y segmentación de imágenes a través de la revisión bibliográfica.
- Determinar el mejor modelo de clasificación y localización para identificar las regiones de interés (caras de troncos) dentro de la imagen.
- Establecer el modelo de segmentación y el ajuste de parámetros que mejor delimite el contorno de las caras de troncos localizada.
- En base a la clasificación y segmentación de la imagen, determinar el diseño del algoritmo para estimar número y diámetro de las caras contenidas en la imagen.
- Determinar métricas de evaluación del rendimiento de los algoritmos de conteo de troncos y estimación de diámetros.

0.3. Alcances

Para el desarrollo y la determinación del mejor modelo se utilizó la base de datos de una planta de aserraderos, que cuenta con el producto Logmeter5000. proporcionada por la empresa Woodtech S.A. La base de datos filtrada contiene, por cada paso de un camión por el pórtico de Logmeter, dos vistas laterales y una superior de alrededor de 20 instantes del paso del camión. Se diseño e implemento un modelo que aborda el conteo de troncos y la medición de diámetro (en píxeles) a través del procesamiento de imágenes.

El desarrollo se realizó en Python y está enfocado en la implementación base que permita cumplir los objetivos planteados, y obtener las métricas de evaluación de los modelos aplicados. Se requiere etapas posteriores de desarrollo para llegar a una versión comercial. Para la determinación de diámetros se consideraron solo las caras de tronco en las que la superficie visible era mayor a un 90 % y se sugiere una etapa futura que permita estimar sobre las caras de troncos parciales.

El trabajo realizado se enmarca en el contexto de desarrollo de productos de bajo costo de Woddtech. El producto completo considera la integración del algoritmos desarrollado con: cámaras tridimensionales para estimación de distancia y con un motor de manejo de software que permita obtener resultados en línea con el ingreso de un camión. La implementación de las cámaras tridimensionales se encuentra actualmente en curso. El algoritmo desarrollado queda disponible para ser integrado.

0.4. Contexto general del trabajo

0.4.1. La empresa Woodtech S.A.

Woodtech S.A es una empresa de desarrollo de tecnología para la industria que tiene como principal objetivo agregar valor a los procesos industriales mediante la incorporación de tecnología para aumentar la eficiencia, controlar y optimizar los recursos. Integrando tecnología de hardware y desarrollando sistemas de software inteligentes, utilizando técnicas de reconocimiento de patrones, análisis de imágenes, aprendizaje automático, entre otros. Destacando en el desarrollo de productos par la industria de la madera.

0.4.2. El producto Logmeter

Logmeter esta diseñado para automatizar una serie de procesos de medición de calidad en las plantas de aserraderos. Consiste en un sistema de medición de volumen de la la madera cargada sobre camiones que ingresa a las plantas utilizando tecnología láser. Las mediciones se realizan cuando el camión pasa a través de un portal equipados con láser (ver figura 2). Logmeter es capaz de reconstruir el modelo tridimensional de los camiones y su carga, además de obtener medidas de calidad de manera rápida, confiable y automatizada.

El portal de Logmeter adicionalmente obtiene capturas de imágenes mediante cámaras estáticas, capturando una serie de fotografías del paso del camión y la carga a través del portal. Este producto, incluye una herramienta para desplegar las imágenes capturadas en una interfaz de usuario para realizar inspección visual de la carga. Las imágenes es posible revisarlas manualmente, pero la mayor parte de la información útil para el proceso es obtenida desde los láser.

0.5. Estructura del documento

En esta primera parte del trabajo se introdujo al lector en el contexto en que se enmarca el trabajo realizado en Woodtech SA. identificando el problema que se busca resolver, mostrando los objetivos y dando cuenta de las soluciones existentes. En el capitulo 1 se entrega un contexto de los temas a tratar abordando desde los métodos más generales de reconocimiento

Figura 2: Sistema de medición de volumen de madera, Logmeter

de imágenes hasta algunos métodos más complejos como redes neuronales, dando al lector un buen punto de inicio para los capítulos siguientes.

A continuación, en el capítulo 2, se presenta la metodología empleada para alcanzar los objetivos propuestos. Las etapas abordaran los dos objetivos principales propuestos; Conteo de caras de troncos y Diámetro. En este capitulo, además, se caracteriza el hardware y software utilizado así como una descripción detallada de la base de datos a utilizar. Posteriormente, en el capítulo 3 se entregan los resultados por cada etapa de realización y los tiempos de ejecución.

Finalizando el trabajo se presentan las conclusiones obtenidas, planteando trabajos futuros que permitan mejorar o continuar el trabajo realizado y enmarcando los resultados en los futuros desarrollos de Woodtech SA. en relación al procesamiento de imágenes. Se agrega adicionalmente, en Anexos, la planificación del trabajo realizado (A), ejemplos de las imágenes utilizadas (B), instrucciones para la creación de bases de datos (C), guías de instalación (D), guía de ejecución (E) y ejemplos de datos de entrada y salida (F).

Capítulo 1

Estado del Arte

El reconocimiento de objetos, colores y movimientos, la interacción con ellos y la estimación de distancias, nos permite desenvolvernos en el mundo que nos rodea. Contar con la capacidad de distinguir formas y separarlas del resto de la imagen, enfocándonos en la información relevante, es un proceso fundamental para determinar la forma en que reaccionamos a los estímulos externos. En el caso de los seres humanos es posible interpretar el entorno gracias a los rayos de luz que alcanzan al órgano receptor, los ojos.

En el ojo, la intensidad de luz es ajustada por la pupila y el iris y se enfoca el objetivo mediante la contracción de músculos unidos al cristalino en un proceso conocido como acomodación. En la retina, se tiene un proceso diferente para escala de grises (bastones) y la visión en colores (conos), para finalmente transformar la información lumínica en señales eléctricas y transportarla a través del nervio óptico. El estimulo emerge en la parte frontal inferior del cerebro, donde se lleva a cabo el proceso de percepción visual.

En el cerebro las señales son interpretadas a través de un complejo mecanismo en el que intervienen diferentes clases de neuronas. Los diferentes aspectos presentes en la imagen como luminancia, diferencias espectrales, orientación y movimiento y las características semánticas significativas (segmentos de línea, límites y forma) son codificados. El lóbulo temporal participa en el reconocimiento de objetos, el lóbulo parietal en el movimiento. La visión depende de la integración de la información en todas las áreas. [1]

Las estructuras presentes en la visión humana han servido como inspiración para algunos de los procesos y estructuras empleadas en visión por computadora. Sin embargo, a pesar de que el modelo humano ha servido por años de inspiración en la visión artificial (VA), existen estructuras que no pueden ser detectadas de manera robusta por la visión por computadora. Las características semánticas significativas no son detectadas de manera robusta por las técnicas de procesamiento de información.

La VA tiene como fin producir información numérica o simbólica para que puedan ser tratados por un ordenador, utilizando características que puedan ser sólidamente detectadas como los colores, las texturas, pero son menos significativas. Incluye métodos para adquirir, procesar, analizar y comprender las imágenes combinando módulos de cámara con la potencia

de procesamiento de un computador, incluyendo herramientas de software. El bajo consumo de energía de estos sistema posibilitan una amplia variedad de aplicaciones. [27]

En la industria, la VA tiene como objetivo comprobar ciertos requisitos, tales como dimensiones, cantidad, entre otros. Por ejemplo, en una carga de camión, un sistema de visión artificial podría inspeccionar en segundos las características de la madera presente, mejorando la calidad del producto y reduciendo el consumo de recursos (predicción, análisis y automatización). Además de los incalculables beneficios en seguridad al disminuir la participación humana en faenas peligrosas.

Iniciaremos el capitulo mostrando los diferentes niveles de procesamiento de datos en la sección 1.1. La siguiente parte del capítulo (1.2) describe la aplicación de redes neuronales en reconocimiento, clasificación y localización en imágenes y sus posibilidades de uso en todos los niveles de procesamiento de imágenes. Adicionalmente se revisaran algoritmos de segmentación de imágenes, y profundizaremos en los dos niveles que tienen mayor énfasis en el desarrollo de este trabajo, preprocesamiento (1.3) y segmentación (1.4).

1.1. Taxonomía del Procesamiento de Imágenes

Las técnicas que se pueden emplear para analizar una imagen comprenden un área denominada Procesamiento Digital de Imágenes. Para analizar la información contenida en una imagen se requieren diferentes niveles de procesamiento que comprenden el preprocesamiento, la segmentación la detección y clasificación para finalizar con el análisis de la imagen. Estos niveles se organizan en cuatro pasos (como se ve en la figura 1.1), cada uno con objetivos específicos.

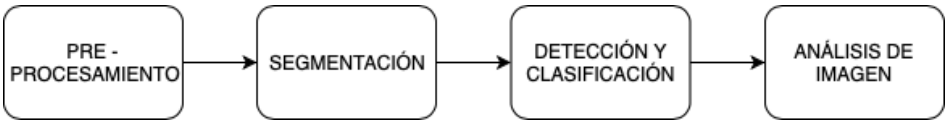

Figura 1.1: Niveles de Procesamiento Digital de Imágenes

El preprocesamiento incluye las operaciones que permiten adaptar la información presente en la imagen para obtener un mejor análisis posterior. La segmentación hace referencia a la partición de la imagen en regiones con información útil. La detección y clasificación permite entregar la etiqueta del objeto detectados dentro de la imagen. El último paso es obtener información de alto nivel a través del análisis de imagen. En el análisis pueden estar presentes todos estos pasos u omitir algunos dependiendo del objetivo. [15]

Existen diversas herramientas, desde simples a más sofisticadas, para resolver problemas en cada uno de los diferentes niveles de procesamiento de imágenes. Dentro de los algoritmos simples existen operaciones como estiramiento de contraste, eliminación de ruido, ecualización de histograma, operaciones de filtros y variadas técnicas de detección de contornos. En oposición, las redes neuronales, que por su versatilidad pueden actuar en cualquiera de los niveles.

1.2. Redes Neuronales

Según la descripción de la Real Academia Española, Aprendizaje es la adquisición por la práctica de una conducta duradera. En computación, la inteligencia es una metodología que provee a los sistemas de habilidades de aprender y/o tratar con situaciones nuevas. Éstos modelos tienen la capacidad de adquirir conocimiento a través de un proceso de aprendizaje y almacenarlo en las conexiones sinápticas. Existen dos formas de aprendizaje: supervisado y no supervisado, y se muestra en la figura 1.2. [7]

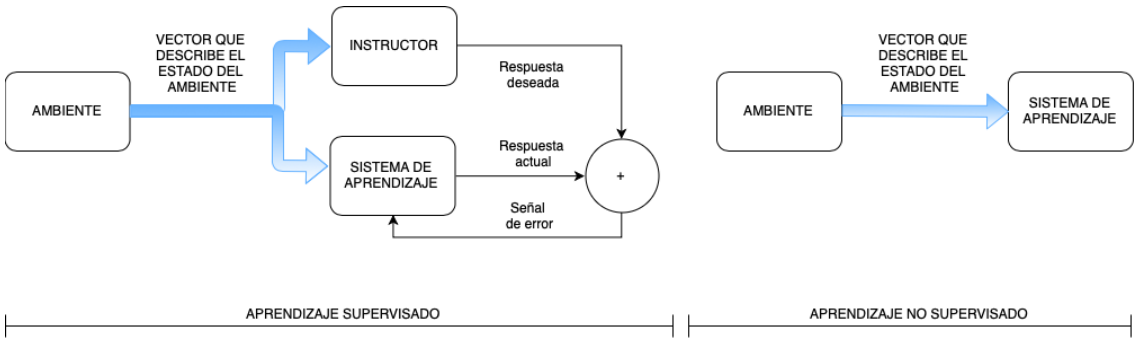

Figura 1.2: Diagrama de bloques de aprendizaje supervisado (izq) y no supervisado (der)

Aprendizaje Supervisado: En estos sistemas, se premia por las conductas correctas y contribuyen a aumentar la probabilidad de ocurrencia mientras que las conductas castigadas tienden a desaparecer. Por cada salida se tiene una retroalimentación que corresponde a la diferencia entre la salida deseada y la real. Es necesario contar con un conjunto de datos etiquetados (clases) previamente. Una vez que el algoritmo aprende de los ejemplos entregados es capaz de generalizar y clasificar de forma autónoma ejemplos que no ha visto antes.

Aprendizaje no Supervisado: En estos sistemas no se dispone de muestras previamente etiquetadas ni de recompensas, por lo que no es posible saber de antemano cuales son las conductas correctas. Es posiblemente el próximo giro del aprendizaje de máquinas ya que posibilita a los algoritmos a aprender sin intervención humana previa. Ellos mismos son capaces de sacar conclusiones de la semántica embebida en los datos.

1.2.1. Aprendizaje Profundo

La posibilidad de que modelos puedan aprender representaciones sobre datos con varios niveles de abstracción se desarrolla por el concepto de aprendizaje profundo o Deep Learning. Es posible, bajo este concepto, descubrir representaciones precisas de forma autónoma sobre grandes volúmenes de datos utilizando arquitecturas convolucionales. En Deep Learning se tienen capas de unidades de proceso llamadas neuronas artificiales, cada una especializada en detectar características específicas ocultas en los datos.

1.2.2. Redes Neuronales Convolucionales

En visión artificial es posible obtener mejoras significativas a través del uso de Deep Learning en comparación con algoritmos más tradicionales. Un caso particular de Deep Learning son las redes convolucionales o Convolutional Neural Network (CNN), que a través de la extracción de características permite detectar y clasificar los objetos dentro de la imagen. Están formadas por una secuencia de capas compuesta por neuronas con pesos y sesgos que se pueden entrenar.

Las neuronas dentro de la red reciben una entrada y la transforman a través de una serie de capas ocultas. Cada neurona de las capas intermedias esta completamente conectada con las neuronas de la capa anterior pero no comparte ninguna conexión con las neuronas de la misma capa. Toda la red expresa una única función de puntuación diferenciable. Posee una función de pérdida , como por ejemplo, SVM/ Softmax en la última capa. La última capa fully-connected o totalmente conectada representa las puntuaciones de clase.

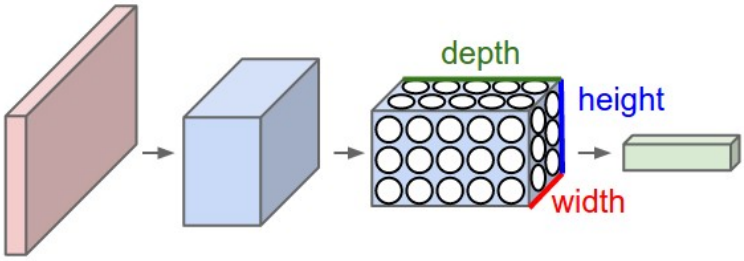

Figura 1.3: Estructura básica de una Red Neuronal Convolucional

En una CNN se supone, de antemano, que las entradas de la red son imágenes, permitiendo codificar ciertas propiedades en la arquitectura. Esto hace que la función de avance sea más eficiente de implementar y reduce en gran medida la cantidad de parámetros en la red. En 1.4 se observa una CNN organizada en sus tres dimensiones (alto, ancho y profundidad). La capa de entrada roja contiene la imagen, por lo que su ancho y alto corresponde a las dimensiones de la imagen y la profundidad a los canales de color (R, G y B).

Las capas de una red neuronal convolucional incluyen tres capas principales, la capa convolucional, la de agrupación y la capa totalmente conectada. La capa convolucional o convolutional layer permite a la red aprender representaciones de características dentro de la imagen. La capa de agrupación o pooling layer se centra en la extracción de características secundarias. La capa totalmente conectada o Fully-Connected layer es donde se calculan los puntajes de clase.

Convolutional Layer

El objetivo de las capas convolucionales es aprender representaciones de características de las imágenes de entrada. Esta capa es la parte central en una CNN. Esta formada por varios mapas de características o feature map, cada neurona del mismo feature map se usa para extraer características locales de diferentes posiciones en la capa anterior.

Para obtener una nueva función, los feature map de entrada se combinan con un kernel aprendido y posteriormente pasan a una función de activación no lineal. Obtendremos diferentes mapas de características mediante la aplicación de diferentes núcleos. Las funciones de activación típicas son sigmoides, tanh y Relu. [23]

Pooling Layer

Esta capa normalmente es utilizada entre dos capas convolucionales. Tiene efectos de extracción de características secundarias y puede recibir las dimensiones de los mapas de características, aumentando la robustez de la extracción. El proceso de muestreo es equivalente al filtrado difuso.

El tamaño de los mapas de características en la capa de agrupación se determina de acuerdo con el paso en movimiento de los núcleos. Las operaciones de agrupación típicas son la agrupación media (overage pooling) y la agrupación máxima (max pooling). Podemos extraer las características de alto nivel de las entradas al apilar varias capas convolucionales y de pooling.

Fully-Connected Layer

Las capas totalmente conectadas toman todas las neuronas de la capa anterior y las conectan a cada una de las neuronas de la capa actual calculando los puntajes de clase. Estas capas no conservan la información espacial y generalmente están antes de la capa de salida. Para las tareas de clasificación, la regresión de softmax se usa comúnmente debido a que genera una distribución de probabilidad de las salidas bien realizada. [8]

1.2.3. Transferencia de Aprendizaje

El entrenamiento de una red neuronal desde cero puede significar grandes costos computacionales. La transferencia de aprendizaje permite obtener un sistema de clasificación de manera rápida y es una opción razonable si consideramos que muchas imágenes comparten algunas similitudes y características. Se transfiere lo que ya ha aprendido a distinguir la red y se perfecciona para las nuevas funciones. En consecuencia se logran disminuir los tiempos de entrenamiento y la cantidad de recursos computacionales necesarios.

Como ya sabemos, durante el entrenamiento la red calcula los pesos óptimos para la clasificación, asociados a conexiones neuronales. Los pesos que se fijan en esta etapa determinan el funcionamiento que tendrá la red. Tanto los pesos como la arquitectura de la red se pueden guardar para ser usados en el futuro sin tener que volver a entrenar. Eso permite solo volver a entrenar ciertas capas finales tomando el resto de la red neuronal como un extractor de características ya construido.

You Only Look Once: YOLO

Gran parte de las tareas de visión artificial consideran tanto clasificar el objeto como determinar en que punto exacto de la imagen se encuentra. En YOLO se aborda la detección y la localización, por lo tanto se obtiene una clasificación del objeto con la probabilidad de que el objeto sea o no el correcto y la posición dentro de la imagen donde se encuentra. La localización de la red se hace a través de un cuadro delimitador alrededor del objeto detectado.

YOLO (You Only Look Once, «sólo se mira una vez») es un algoritmo de visión artificial que detecta y clasifica objetos en tiempo real. Es capaz de procesar a fotogramas por segundo (fps) mayores a los que solemos ver películas (45 fps). La gran ventaja de utilizar YOLO radica en que solo necesita ver una vez la imagen para realizar clasificación (como su nombre lo dice), aumentando la rapidez (solo necesita 22 milisegundos por imagen) en contraste a una disminución de exactitud.

En esta red se utiliza deep learning y CNN para la detección que se lleva a cabo dividiendo la imagen en recuadros de tamaño estimado para la clase buscada. Se calcula la probabilidad de que el recuadro corresponda a la clase indicada para N diferentes recuadros. Los recuadros con probabilidad menor a un límite son eliminados. Posteriormente se aplica Supresión no máxima para eliminar objetos duplicados. [14] [12]

Se enmarca la detección de objetos como un problema de regresión y probabilidades de clases asociadas. Si bien comete errores de localización, es menos probable que prediga falsos positivos en el fondo. El sistema de detección cambia el tamaño de las imágenes de entrada a 448x448, ejecuta una sola red convolucional y limita las detecciones resultantes por la confianza del modelo. Logrando más del doble de precisión promedio que otros sistemas en tiempo real. [17]

En la red se utiliza la información contextual de las clases al momento de entrenamiento y prueba. Es altamente generalizada pero aún presenta desventajas en la precisión de la localización, principalmente los objetos más pequeños. El sistema divide la imagen de entrada en cuadriculas (1.4 izq). Si el centro de un objeto presente en la imagen cae en una celda de cuadricula, esa celda es responsable de su detección (centro). Cada celda predice un numero fijo de cajas delimitadoras y un nivel de confianza para esa predicción (der).

Cada cuadro delimitador entrega de 5 predicciones: x, y, w, h y confianza. Las coordenadas (x, y) representan el centro del cuadro en relación con los límites de la celda de la cuadrícula. El ancho y la altura se predicen en relación con toda la imagen. Finalmente, la predicción de confianza representa el IOU entre el cuadro predicho y cualquier cuadro de verdad fundamental.

La segunda versión, YOLOv2 detecta objetos en tiempo real para más de 9000 categorías de objetos diferentes, mejorando la localización y manteniendo la rapidez. Considerando que etiquetar imágenes para detección es mucho mas costoso que etiquetar imágenes para clasificación, en esta versión se amplia el vocabulario y robustez de la red aprovechando los millones de imágenes clasificadas disponibles y se mejora la precisión en la localización de

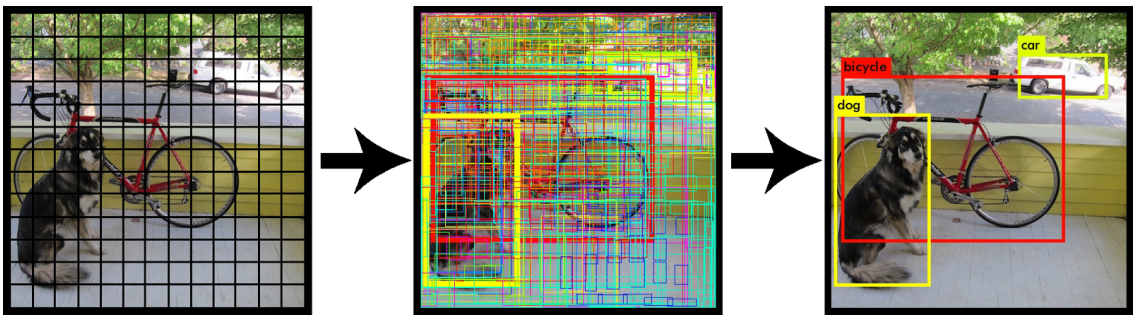

Figura 1.4: Estructura de la detección realizada por YOLO

objetos con las miles de imágenes de detección etiquetadas. [18] [16]

Finalmente se elige esta arquitectura de software por ser un algoritmo escrita en C y CUDA, de código abierto. Adicionalmente por robustez frente a variaciones de fondo y luminosidad. A continuación se muestran diferentes topologías de YOLO, con sus respectivos conjuntos de entrenamiento y prueba y el valor de mAP, FLOPS y FPS obtenido.

Modelo	Train	Test	mAP	FLOPS	FPS
SSD300	COCO trainval	test-dev	41.2	-	46
SSD500	COCO trainval	test-dev	46.5	-	19
YOLOv2 608x608	COCO trainval	test-dev	48.1	62.94 Bn	40
Tiny YOLO	COCO trainval	test-dev	23.7	5.41 Bn	244
SSD321	COCO trainval	test-dev	45.4	-	16
DSSD321	COCO trainval	test-dev	46.1	-	12
R-FCN	COCO trainval	test-dev	51.9	-	12
SSD513	COCO trainval	test-dev	50.4	-	8
DSSD513	COCO trainval	test-dev	53.3	-	6
FPN FRCN	COCO trainval	test-dev	59.1	-	6
Retinanet-50-500	COCO trainval	test-dev	50.9	-	14
Retinanet-101-500	COCO trainval	test-dev	53.1	-	11
Retinanet-101-800	COCO trainval	test-dev	57.5	-	5
YOLOv3-320	COCO trainval	test-dev	51.5	38.97 Bn	45
YOLOv3-416	COCO trainval	test-dev	55.3	65.86 Bn	35
YOLOv3-608	COCO trainval	test-dev	57.9	140.69 Bn	20
YOLOv3-tiny	COCO trainval	test-dev	33.1	5.56 Bn	220
YOLOv3-spp	COCO trainval	test-dev	60.6	141.45 Bn	20

Tabla 1.1: Comparación de diferentes topologías de YOLO

1.3. Preprocesamiento de Imágenes

El preprocesamiento de imágenes puede reducir el ruido en la información, simplificar los datos y mejorar la confiabilidad, extracción de características, segmentación, reconocimiento y eficiencia en el procesamiento de imágenes. Para introducirnos en los conceptos de proce-

samiento de imágenes describiremos un punto dentro de la imagen $[(x, y) \in R_2]$ determinado por un funcional de intensidad de luz $u(x, y)$ que puede tomar valores entre cero (negro) y 256 (blanco), pasando por todos los valores enteros intermedios (escala de grises).

La caracterización básica de una imagen la entregan los bordes, que contienen información relevante de las características de textura y la base del análisis de la calidad de la forma. Los algoritmos clásicos de detección de bordes son operadores de gradiente de ventanas locales, como Sobel y Laplaciano, pero presentan sensibilidad al ruido. El operador Canny es reconocido como un estándar en reconocimiento de bordes. Además de alternativas que pueden ser ajustada a características morfológicas como Contornos Activos. [28]

Para definir la escala de color a utilizar debemos tener en cuenta que el procesamiento de imágenes en escala de grises presenta dos características importantes. La primera es el valor intrínseco del color, con potencial para ayudar en varios aspectos de la inspección, principalmente en aquellos donde el color juega un rol importante. En algunos casos (1.5) el color ayuda en la segmentación, beneficiándose de los diferentes tonos de un paisaje natural (izq) o detectando las señales de transito y marcas de calzada, para el caso de la ciudad (der).

Figura 1.5: Valor del color en la segmentación y el reconocimiento

La segunda es la penalización de almacenamiento y procesamiento que podrían generar imágenes de alta resolución. Para realizar la conversión a escala de grises se obtiene una media ponderada de las distintas componentes de color de cada píxel (canales rojo(R), verde(G) y azul(B)), obteniendo como resultado el equivalente en blanco y negro. La imagen de salida, en consecuencia, tendrá un solo canal y, con una combinación adecuada de ellos, un buen realce de contornos. El color solo tomara un papel importante cuando aporte información.

Umbral (Threshold)

Método más simple de segmentación, permite obtener la binarización de una imagen que actualmente se encuentra en escala de grises. Por cada píxel dentro de la imagen, se modifica el valor de la función de intensidad $(u(x, y))$, tras compararlo respecto al valor del umbral fijado $(thresh)$. Si los objetos aparecen sobre un fondo con alto contraste, la aplicación de umbral permitirá facilitar la visualización. Existen diferentes tipos de umbrales. Consideremos una imagen inicial en escala de grises (fig 1.6, izq).

El Umbral Binario (T_{Bin}) establece la función de intensidad en el valor máximo (V_{max})

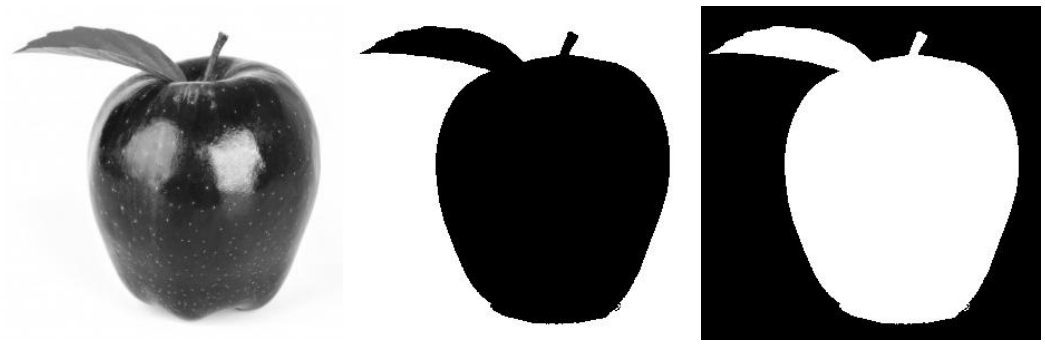

Figura 1.6: Umbral Binario, Ejemplo 1: Imagen Original (izq), Binaria (centro) y Binaria Inversa (der)

para todos los valores de intensidad mayor al umbral y cero para el resto, como se expresa en la ecuación 1.1. El Umbral Binario Inverso ($T_{Bin-Inv}$) asigna los valores opuestos al Umbral Binario como se muestra en la ecuación 1.2. En la figura 1.6 se muestra a la izquierda la imagen inicial en escala de grises, al centro la imagen binarizada y a la derecha la binarización inversa.

Figura 1.7: Umbral Binario, Ejemplo 2: Imagen Truncada (izq), Umbral a Cero (centro) y Umbral a Cero Inverso (der)

La función Truncar (T_{Trunc}), Umbral a Cero (T_{Toz}) y Umbral a Cero Invertido ($T_{Toz-\text{inv}}$) quedan descritas en las ecuaciones 1.3, 1.4 y 1.5 respectivamente. En estas funciones, para una condición determinada se modifica el valor de intensidad a un valor fijo o humbral, y para los valores fuera de este umbral, se mantiene el valor de intensidad actual. En la figura 1.7 se muestra a la izquierda la imagen truncada, en el centro el Umbral a Cero y a la derecha el Umbral a Cero Inverso. [6]

$$T(Bin) = \begin{cases} \text{si } u(x,y) > thresh & V(max) \\ \text{si } u(x,y) \leq thresh & 0 \end{cases} \tag{1.1}$$

$$T(Bin - Inv) = \begin{cases} \text{si } u(x,y) > thresh & 0 \\ \text{si } u(x,y) \leq thresh & V(max) \end{cases} \tag{1.2}$$

$$T(Trunc) = \begin{cases} \text{si } u(x,y) > thresh & thresh \\ \text{si } u(x,y) \leq thresh & u(x,y) \end{cases} \tag{1.3}$$

$$T(Toz) = \begin{cases} \text{si } u(x,y) > thresh & u(x,y) \\ \text{si } u(x,y) \leq thresh & 0 \end{cases} \tag{1.4}$$

$$T(Toz - \text{inv}) = \begin{cases} \text{si } u(x,y) > thresh & 0 \\ \text{si } u(x,y) \leq thresh & u(x,y) \end{cases} \tag{1.5}$$

1.3.1. Operaciones de Filtros

La aplicación de filtros puede cumplir diferentes objetivos tanto en el suavizado, la eliminación de ruido, el realce de las zonas importantes dentro de la imagen o la detección de bordes. El proceso consiste en la aplicación de una matriz de filtrado de NxN compuesta por números enteros, sobre cada uno de los píxeles de la imagen. El resultado se divide por un escalar, generalmente la suma de los coeficientes de ponderación, para mantener constante la intensidad de la imagen. La aproximaciones varían con el tamaño del vecindario elegido.

Por cada movimiento del kernel $w(x, y)$ sobre la imagen $f(x, y)$, el valor ubicado en el centro del kernel es reemplazado, generando una nueva imagen resultante $g(x, y)$, representado en la ecuación 1.6. Dentro del proceso de filtrado, se reduce la imagen en $N - 2$ filas y $N - 2$ columnas, correspondiente a las zonas donde la matriz de filtrado queda fuera de la imagen. Es importante resaltar que el tipo de filtro aplicado estará determinado por el contenido de la mascara. [5]

$$g(x,y) = \sum_{\text{i}=0}^{n} \sum_{j=0}^{m} w(\text{i}, j) f(x + \text{i}, y + j) \tag{1.6}$$

Filtros Pasa Bajos: Suavizado

El objetivo de estos filtros es suavizar la imagen, cuando se tiene gran cantidad de ruido que se desea eliminar. Por ejemplo, si tenemos ruido de tipo impulsivo (Sal y Pimienta), que se presenta como un punto de luz en un fondo oscuro o un punto oscuro en un fondo claro, podemos obtener buenos resultados con un filtro de vecinos cercanos como el filtro de mediana, manteniendo los detalles de contorno. El cálculo de mediana se realiza ordenando los valores de los píxeles vecinos en orden y seleccionado el que queda en medio.

La ventaja de este filtro es que el valor central es un valor real, presente en la imagen original, y no un promedio, evitando efectos borrosos como en el caso del filtro de promedio que se vera más adelante. También es menos sensible a la presencia de valores extremos. La desventaja es la dificultad para calcular, siendo necesario ordenar los valores para luego seleccionar el píxel central. En la figura 1.8 se observa la imagen inicial en escala de grises (derecha) y la imagen tras aplicar un filtro de mediana de tamaño 3x3 (centro).

Si tomamos ahora otro ejemplo, supongamos que lo que queremos eliminar ya no es un ruido de tipo impulsivo sino más bien queremos reducir las variaciones de intensidad, con un

Figura 1.8: Filtro de Mediana y Promedio: Imagen en Escala de Grises (izq), Filtro de Mediana (centro) y Filtro de Promedio (der)

método de calculo rápido y simple. El filtro utilizado en este caso seria un filtro de promedio, donde el valor central es reemplazado por el promedio de los píxeles dentro del vecindario. Un ejemplo de máscara, de 3x3, utilizada en este caso será la mostrada en la ecuación 1.7 y su aplicación en la figura 1.8 (derecha).

$$w(x,y) = \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \tag{1.7}$$

$$w(x,y) = \frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix} \tag{1.8}$$

Ambas máscaras operan en un vecindario de 3x3, por lo que tienen una carga computacional relativamente pequeña. La ecuación 1.8 presenta una máscara que se aproxima mejor a un perfil Gaussiano y se conoce como Filtro Gaussiano. Altamente efectivo en imágenes cuyo ruido presenta una distribución normal. El filtro Gaussiano simula una distribución bivariante, para calcularla se utiliza la ecuación 1.9 donde el valor máximo aparecerá en la posición central, disminuyendo hacia los extremos tan rápido como pequeña sea la varianza σ.

$$G(x,y) = \frac{1}{2\pi\sigma^2} \cdot \mathrm{e}^{\frac{(x^2+y^2)}{2\sigma^2}} \tag{1.9}$$

$$\frac{\delta G}{\delta x} = ky \cdot exp(\frac{-x^2}{2\sigma^2}) \cdot exp(\frac{-y^2}{2\sigma^2}) \tag{1.10}$$

$$\frac{\delta G}{\delta y} = ky \cdot exp(\frac{-x^2}{2\sigma^2}) \cdot exp(\frac{-y^2}{2\sigma^2}) \tag{1.11}$$

El filtro gaussiano permite separar una convolución bidimensional en dos convoluciones unidimensionales. Una en sentido horizontal (ecuación 1.10) y otra en sentido vertical (ecuación 1.11). Presenta ventajas en la uniformidad del suavizado en comparación a otros filtros

[10]. En la figura 1.9 se observa el resultado de aplicar filtro gaussiano con variaciones en el tamaño del kernel. De izquierda a derecha se muestra la imagen con kernel de $3x3$, $7x7$ y $11x11$ sobre la imagen original.

Figura 1.9: Filtro Gaussiano: Con Kernel de $3x3$ (izq), $7x7$ (centro) y $11x11$ (der)

El filtro de promedio y el filtro gaussiano suprimen el ruido afectando, de paso, la señal inicial. Es posible reducir el desenfoque utilizando un filtro más estrecho, pero al mismo tiempo, disminuye la capacidad para eliminar ruido. Es sensible a cambios locales, si la imagen presente picks de intensidad producidos por distorsión, la aplicación de estos filtros ponderaran ese valor, esparciéndolo en los vecinos cercanos. También puede crear nuevas intensidades de grises que no estaban presentes en la imagen original [22]

1.3.2. Ecualización de Histograma

Un histograma muestra la medida de frecuencia, relativa o absoluta, en que se repite un cierto nivel de intensidad de gris dentro de la imagen. Útiles para obtener una primera visión general de la distribución, para nuestro caso particular permite evaluar el contraste presente en la imagen. En el caso de un histograma estrecho se identifica un bajo contraste, ya que se utilizan pocos niveles y con valores cercanos. El caso contrario indica una amplia distribución de los niveles de intensidad de grises, por lo tanto un mayor contraste.

La ecualización lineal de histograma corresponde a una operación global sobre la imagen, que permite acentuar el contraste visual y la iluminación ajustando los rangos de intensidad de la imagen. Se considera que el histograma esta ecualizado cuando todos los niveles de intensidad tienen una frecuencia similar, en otras palabras, cuando el histograma acumulativo de la imagen se aproxima a una recta. El valor linealizado (1.12) es una función del histograma acumulativo, $ha(p)$, y las dimensiones h y w.

$$P_{Linealizado} = ha(p) \cdot \frac{255}{h \cdot w} \tag{1.12}$$

Se muestran a continuación diferentes ejemplos de histogramas ecualizados. En ñlas siguientes figuras se tiene, de izquierda a derecha, la imagen en escala de grises junto a su histograma (en negro la frecuencia por rango de intensidad y en celeste la frecuencia acumulada). Seguida de la imagen ecualizada y el histograma de la imagen ecualizada. En la

figura 1.10, el histograma inicial esta desplazado hacia la izquierda ya que en la imagen inicial predominan los tonos oscuros.

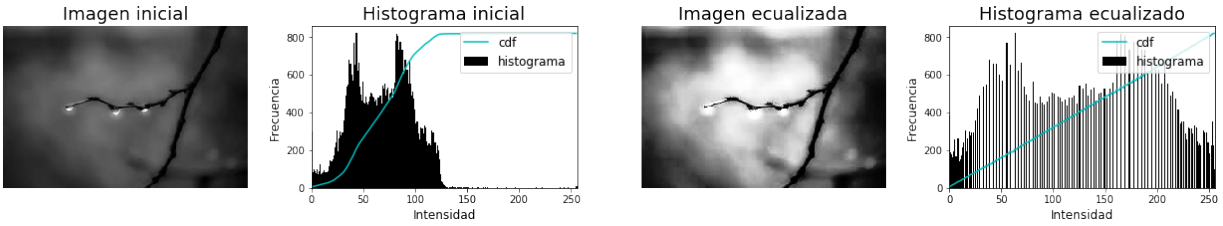

Figura 1.10: Ecualización de histograma para imagen con predominio de tonos oscuros

Para imágenes con predominio de tonos claros, como en la figura 1.11, el histograma inicial se desplaza hacia la derecha. Si el histograma esta centrado, pero no hay objetos completamente claros u oscuros se tendrá un histograma estrecho. Para los tres ejemplos anteriores, la imagen resultante presenta mejor iluminación y un mayor contraste. En el histograma ecualizado es posible notar que todos los niveles de intensidad tienen una frecuencia más o menos similar.

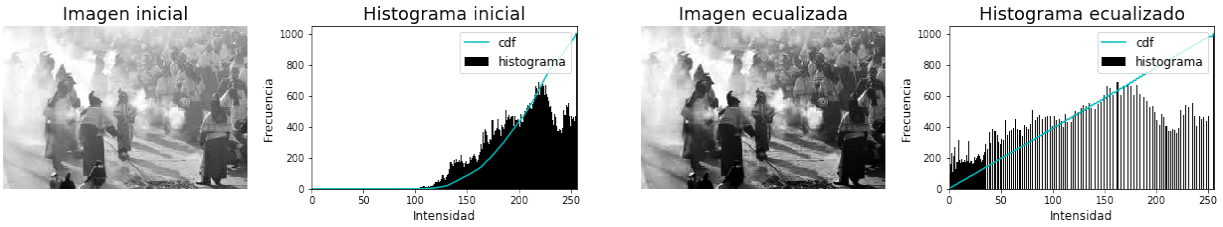

Figura 1.11: Ecualización de histograma para imagen con predominio de tonos claros

Como aporte adicional, la ecualización de histograma es utilizada en el área de visión por computadora, en aplicaciones de inspección automática. En algunos casos es posible obtener histogramas bimodales, una de las agrupaciones de niveles corresponde al objeto y la otra al fondo de la imagen. El nivel de gris equidistante entre los dos máximos resulta ser adecuado para separar el fondo de la imagen. En la figura1.12 se observa una agrupación de niveles desplazada a la izquierda, tonos oscuros, que corresponden al fondo de la imagen.

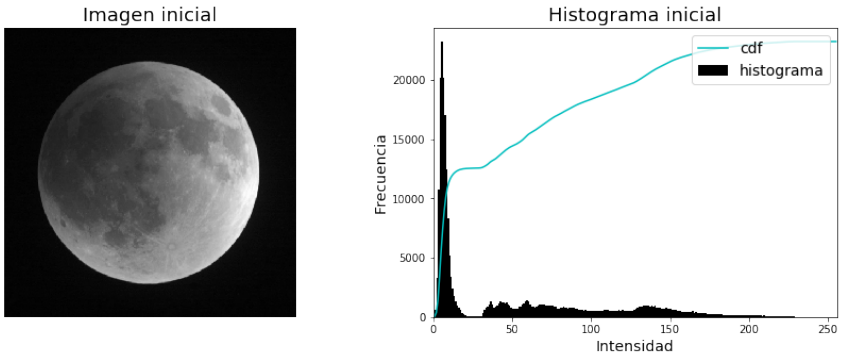

Figura 1.12: Ecualización de histograma para detección de objetos

1.4. Segmentación de Imágenes

La segmentación de imágenes se propone como una forma de encontrar regiones uniformes dentro de la imagen. Se considera que las áreas encontradas a través de este método tendrá una alta probabilidad de coincidir con los objetos en la imagen. Como lo plantean algunos autores la mayoría de las técnicas de detección de imágenes tienen su origen en la teoría de detección de contornos, manteniendo como idea central que la información de una imagen esta contenida en los contornos aparentes que dejan los objetos físicos en la imagen. [3]

Un contorno es una curva cerrada definida por los puntos donde la intensidad de luz cambia bruscamente, equivalente a tener gradientes de intensidad altos ($|\nabla u(x,y)|$). La dirección v queda determinada por el gradiente normalizado (ecuación 1.13) e indica la dirección perpendicular a la silueta. La detección de contornos reduce la redundancia de información, y por consiguiente, el espacio necesario. Puede resultar inadecuado para imágenes ruidosas, donde los gradientes altos no se organizan siempre delimitando contornos reales.

$$v = \frac{\nabla u(x,y)}{|\nabla u(x,y)|} \tag{1.13}$$

El rendimiento de los operadores en la detección de bordes depende de diversos factores como la calidad de la imagen, iluminación, la presencia de objetos de intensidad similares, la densidad del borde y el ruido. Una opción para mejorar la detección de bordes es realizar el cálculo de gradiente después de un filtro de suavizado, que representaremos como $u(t,x,y)$ donde $t > 0$ mide cuanto hemos suavizado. En el caso de un filtro Gaussiano el parámetro t corresponde a la vecindad del filtro. [21]

Operador de Bordes de Sobel

El operador de bordes de Sobel es un operador diferencial discreto ya que utiliza el cálculo de derivadas para determinar los bordes de la imagen. Este filtro calcula el gradiente de la intensidad de la imagen por cada píxel, entregando como salida la magnitud del mayor cambio posible. Para cada píxel de la imagen, el vector gradiente apunta en la dirección del incremento máximo posible de intensidad, y la magnitud del vector gradiente corresponde a la cantidad de cambio de la intensidad en esa dirección. [28]

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \cdot I \tag{1.14}$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \cdot I \tag{1.15}$$

Los cambios de magnitud se calculan mediante la convolución de I con un kernel G con tamaño impar. Un ejemplo de kernel de $3x3$ para obtener los cambios horizontales es G_x (ecuación 1.14). Un ejemplo de kernel para obtener los cambios verticales es G_y (ecuación 1.15). Por cada punto de la imagen se obtendrá una aproximación del gradiente en ese punto combinando los cambios verticales y horizontales, obteniendo el valor de G como se muestra en la ecuación 1.16. Aveces la ecuación utilizada es la ecuación 1.17 por su simplicidad.

$$G = \sqrt{G_x^2 + G_y^2} \tag{1.16}$$

$$G = \|G_x\| + \|G_y\| \tag{1.17}$$

Sobel permiten determinar con precisión la ubicación del borde de la imagen, y posee cierto grado de manejo de ruido. Sin embargo, el borde determinado es ancho llevando a una baja precisión de la ubicación del borde. En la figura 1.13 (centro) se muestra el resultado de aplicar filtro Sobel sobre la imagen inicial (izquierda).

Detector de Bordes Canny

El operador Canny se basa en un algoritmo de optimización ampliamente utilizado ya que tiene una buena precisión de detección, fuerte capacidad anti-interferencia y mejora la continuidad del borde ya que aplica doble contorno para seleccionar entre los contornos potenciales. En la figura 1.13 se muestra la imagen inicial en escala de grises (izq) y el resultado de aplicar el operador de bordes Canny (der). Una de las características de los métodos basados en gradiente es que no forman necesariamente curvas cerradas.

Figura 1.13: Detector de bordes Canny: Imagen en escala de grises (izq), operador de bordes de Sobel (centro) y el resultado de aplicar el operador de bordes Canny (der)

En el detector de bordes de Canny se buscan los puntos de contorno (x, y) dentro de una imagen suavizada $u(t, x, y)$, donde el valor de t mide cuanto se ha suavizado la imagen, que cumplan para todo $t > 0$ que la función 1.18 tenga su máximo en $s = 0$ y su valor supere un cierto umbral. Para su correcto funcionamiento se debe especificar cuidadosamente el ancho de banda sobre el cual se espera que funcione. La utilización de este método requiere de varias etapas de procesamiento. [9]

$$s \rightarrow \|\Delta u(t,(x,y)) + sv\| \tag{1.18}$$

La primera etapa de procesamiento (ver diagrama 1.14) consiste en la aplicación de filtros convolucionales Gaussianos, para los cuales se conoce la desviación estándar σ. Se utiliza el operador de bordes de Sobel como máscara diferencial de primer orden. Con el operador de bordes de Sobel ya se incluyen filtros pasa bajos, por lo que es posible reducir los de la primera etapa. Con la supresión no máxima solo se conservan los puntos de borde que se han definido como máximos locales a lo largo de la normal redimensionando el borde a la unidad.

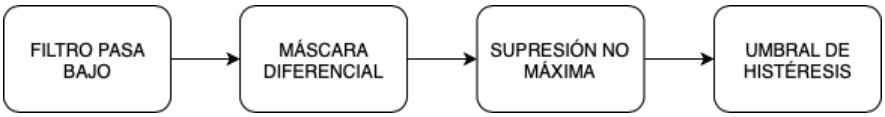

Figura 1.14: Etapas de procesamiento del Operador Canny

La aplicación de Canny termina con la aplicación de dos umbrales de Histéresis, un umbral superior que garantiza que los bordes seleccionados sean confiables, el segundo umbral permite seleccionar puntos en los que sea altamente probable que pertenezcan al borde por la cercanía a los puntos de bordes confiables. En la práctica los umbrales deben ser examinados dependiendo de los datos de cada imagen, sin embargo una forma simple par elegir el umbral inferior es que debe ser aproximadamente la mitad del umbral superior. [24]

1.4.1. Contornos Activos

Las técnicas de detección de contornos son útiles cuando las imágenes son sencillas, con objetos o fondo uniforme y poco ruido. Permiten buenos resultados cuando tenemos la posibilidad de adaptar el algoritmo a cada una de las imágenes que se desee y fue el inicio de varios desarrollos orientados al objeto. Cuando las situaciones son más complejas será necesario un procesamiento adicional para reconocer los objetos segmentados. Técnicas más robustas fueron introducidas por *M. Kass, A. Witkin y D. Terzopoulos* en 1988 acuñando por primera vez el termino de Contornos Activos o "Snakes".

El modelo de Contornos Activos (AC por su nombre en Ingles, Active Contourn) es un modelo variacional que parte de un modelo genérico del objeto, C_0, que puede ser una plantilla geométrica, parametrizada de forma explicita o implícita. Posteriormente la curva C_0 es ajustada a la frontera de un objeto a través de la minimización de energía [13]. El funcional de energía utilizado (ecuación 1.19) consta de dos términos, uno que permite controlar la suavidad de la curva y otro que atrae la curva hacia las fronteras de los objetos en las imágenes:

- **Energía interna:** Determina la regularidad y elasticidad de las fronteras, útil para modificar el comportamiento físico de AC y la continuidad local. Corresponde al primer y segundo término de la ecuación 1.19.

- **Energía externa:** Corresponde a la energía potencial externa que atrae a la curva hacia los contornos del objeto. Se diseñan para que sus mínimos locales coincidan con extremos de intensidad, bordes, o cualquier otra característica de interés en la imagen. Corresponde al tercer termino de la ecuación 1.19

$$E(C) = \alpha \int_0^1 C'(q)^2 \cdot \mathrm{d}q + \beta \int_0^1 C''(q)^2 \cdot \mathrm{d}q - \lambda \int_0^1 \Delta I(C(q)) \cdot \mathrm{d}q \tag{1.19}$$

Los términos α, β y λ son constantes reales y positivas, α determinan la elasticidad y β la rigidez de la curva. El funcional de la ecuación consta de dos términos. Para obtener los contornos de un objeto utilizando AC se debe minimizar el funcional para un conjunto constante de α, β y λ dadas. Partiendo de una curva C_0 que se desplaza en dirección del máximo decrecimiento de energía utilizando el método del descenso. Utilizando contornos activos es posible localizar eficientemente curvas cercanas a la curva inicial C_0.

Es necesario tener en consideración que, en contornos activos, la regularidad de la parametrización no implica la regularidad geométrica de la curva. En consecuencia, para cualquier valor de $\alpha, \beta >= 0$ con $\alpha + \beta > 0$ y $\lambda > 0$ es posible obtener curvas con ángulos. Otra desventaja es que debido a que los términos de regularización no permiten a la curva romperse, no es posible detectar varios objetos simultáneamente, aunque la curva los rodee a todos ellos.

En CA es necesario gestionar directamente los cambios de topología de la curva que evoluciona cambiando su parametrización en el momento que se prevea una singularidad. Para evitar estos inconvenientes fueron creados los Modelos Geométricos de AC, donde la curva es deformada según una velocidad que depende de parámetros geométricos intrínsecos y de la adaptación de los contornos a la imagen. Esta basado en la formulación por conjuntos de nivel que permiten cambios automáticos de la topología y la detección simultanea.

Contornos Activos Geodésicos

Los Modelos de Contornos Activos Geodésicos abreviados GAC (por su nombre en ingles Geodesic Avtive Contours) minimizan la longitud de una curva en R^2 dotada de una métrica de Reimann derivada de la imagen que asigna menos longitud a las curvas situadas en puntos de la imagen donde el modulo del gradiente de intensidad es más grande. [4]

Contornos Activos Morfológicos

Los Contornos Activos Morfológicos o MorphAC (por su nombre en Inglés, Morphological Active Contours) presentan un comportamiento similar al de AC, pero, a diferencia de ellos también utilizan operadores morfológicos como la dilatación y la eroción. Existen dos implementaciones conocidas de contornos activos morfológicos, Contornos activos morfológicos geodésicos (MorphGAC, por su nombre en ingles) y contornos activos morfológicos con épocas (MorphACWE, por su nombre en ingles)

- MorphGAC: es adecuado para imágenes con contornos visibles, incluso cuando los contornos presentan ruido, están cortados o parcialmente ocluidos. Sin embargo, requiere que la imagen sea preprocesada para resaltar los contornos. Las características de MorphGAP dependen fuertemente de los pasos que se realicen en el preprocesamiento.
- MorphACME: funciona bien cuando los valores de píxeles de las regiones interna y externa del objeto a segmentar tienen promedios diferentes. No requiere que los contornos del objeto estén bien definidos, y funciona sobre la imagen original sin ningún procesamiento previo.

Capítulo 2

Metodología

Con lo revisado anteriormente, en el capítulo 1, conocemos las diferencias de algunas técnicas de procesamiento y segmentación de imágenes. Sin embargo, los resultados siempre dependerán de las condiciones de la imagen a analizar. Algunas de las condiciones que pueden influir son: luz, contraste, superposición de objetos, forma de los objetos, textura, entre otros. Adicionalmente, alguno de los procedimientos descritos anteriormente requieren ajustes manuales para cada nueva imagen.

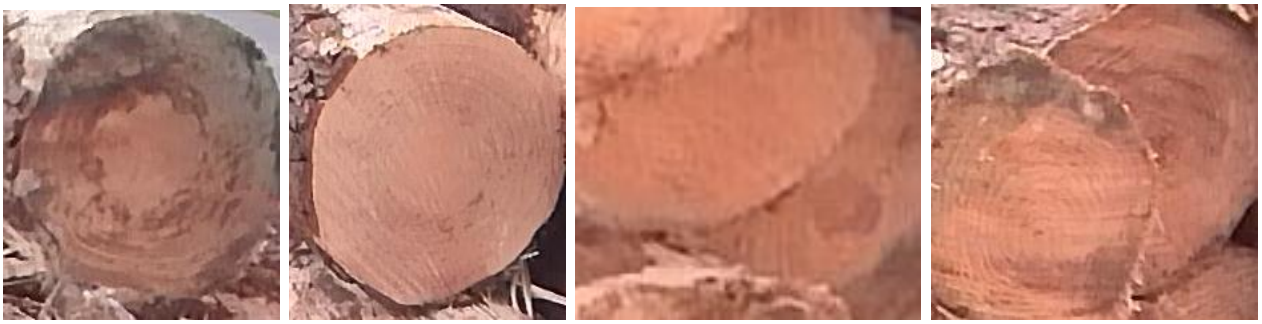

Figura 2.1: Ejemplos de caras detectadas por la red

De las caras de troncos que es posible observar dentro de la imagen se logran distinguir dos clases importantes. La primera comprende a aquellas caras en las que se puede ver sobre el 90 % de ellas y el segundo corresponde a aquellas caras parcialmente ocluidas en las que se ve menos del 90 %. Llamaremos al primer conjunto 'Caras Full' y al segundo 'Caras Partiales'. En la imagen 2.1 se observan ejemplos para ambas clases, de izquierda a derecha, dos ejemplos de Caras Full y dos de Caras Partiales.

Considerando que no estamos en el caso ideal, con imágenes con bastante textura, superpuestas, difíciles de segmentar se seleccionó la Red Neuronal Convolucional como punto de partida para la detección de las regiones de interés (ROI). La red entrega recuadros de las zonas detectadas como caras, con la probabilidad asociada. Para las caras detectadas como Full se realizara un procesamiento adicional que permitirá obtener el borde exacto del contorno y así calcular el diámetro de cada una de esas caras.

Para completar el proceso descrito se contará con tres etapas: 'Detección de caras', 'Ajuste de contorno fino' y 'Análisis'. La primera etapa de desarrollo permite identificar las caras de troncos presentes en la imagen. La segunda etapa toma como entrada las detecciones realizadas por la etapa anterior y busca ajustar el contorno fino de las detecciones. La última etapa concluye el proceso al calcular el número de caras y los diámetros. Las etapas mencionadas se describen a continuación.

2.1. Estructura de desarrollo

Para abordar los objetivos propuestos, obtener el número total de caras y el diámetro de las caras de un banco de madera, consideraremos una estructura de desarrollo con las tres etapas mencionadas anteriormente. La primera etapa, 'Detección de caras', considera como entrada una imagen de la carga de un camión. La detección de caras considera la localización y segmentación de las caras de troncos dentro de la imagen. Como salida de esta etapa se tiene la posición y categoría de cada uno de las segmentos de caras de troncos detectadas.

La etapa 'Ajuste de contornos', tomará como entrada la detección de segmentos de cara. En ésta etapa se ajusta el contorno para determinar de manera más exacta el diámetro. Como salida se entrega una lista con el diámetro de cada una de las caras detectadas. En ésta etapa se recurre a técnicas de filtrado para sortear problemas en la segmentación del contorno. Por ejemplo, problemas de contraste, de calidad de la imagen, distorsión producida por movimiento durante la captura o la aparición de objetos adicionales en la imagen.

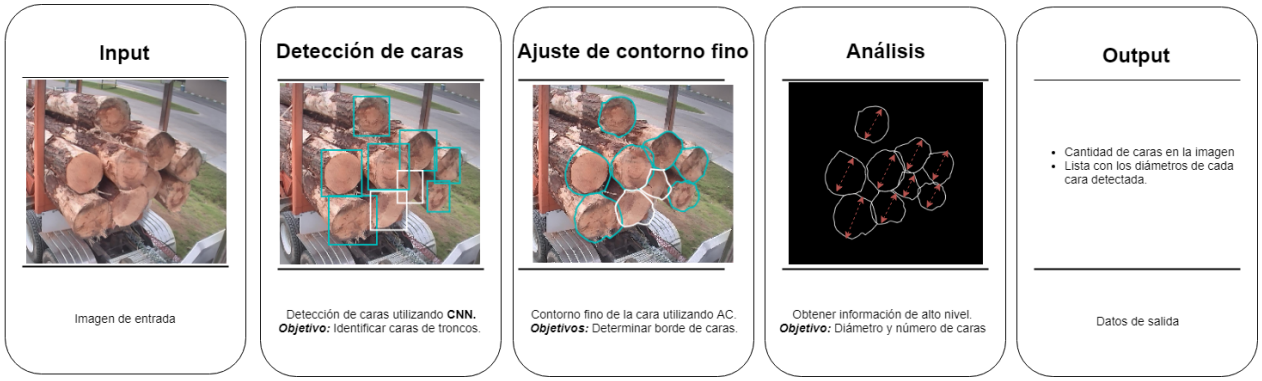

Figura 2.2: Etapas de desarrollo para detección de caras, conteo y obtención de diámetro

En la tercera etapa, 'Análisis', revisaremos los problemas que afectan la detección y/o segmentación de imágenes. Se considera el análisis de los resultados obtenidos, así como la verificación de los algoritmos utilizados. En el diagrama 2.2 se muestran las etapas de desarrollo descritas: la etapa de Detección de Caras para identificar las caras de troncos, la de Ajuste de Contorno fino para obtener el borde de la cara y se finaliza con la etapa de Análisis.

Para la etapa de 'Detección de Caras' y 'Ajuste de Contorno fino' será necesario agregar una etapa paralela de validación de los algoritmos seleccionados. Para la etapa de 'Detección de caras' se incluirá la selección del mejor umbral en la detección y la selección de la mejor red

entrenada para cada objetivo; contar o medir diámetro. Para la etapa de 'Ajuste de contornos' se incluirá la selección de los mejores parámetros para la etapa de preprocesamiento de la imagen y la elección del algoritmo de segmentación.

2.1.1. Detección de caras

La estructura simplificada de ésta etapa se muestra en el diagrama 2.3. Se recibe como entrada una imagen, para la que no se tiene información previa ni datos marcados. La imagen es procesada por la red ya entrenada para detectar caras de troncos. El primer resultado de la red corresponden a la detección de segmentos de cara, caracterizados a través de la posición central, ancho y alto del cuadrado que contiene la cara y la categoría a la que corresponde la detección.

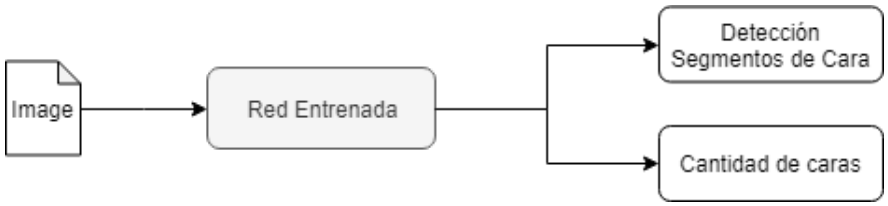

Figura 2.3: Detección de caras, diagrama de la primera etapa de desarrollo

Con la detección de los segmentos de cara se obtiene el segundo resultado y corresponde a la cantidad de caras detectadas. El número de caras detectadas es el resultado principal de ésta etapa y completa el primer objetivo de este trabajo. El modo de ejecución descrito tomará el nombre de 'Modo On-line'. Previo a la ejecución y caracterización del modo on-line para detección de caras será necesaria una etapa de entrenamiento y validación del entrenamiento de la red a utilizar.

Entrenamiento de la red

Las imágenes utilizadas para el entrenamiento se marcan previamente, especificando la posición de cada una de las caras y su categoría. El proceso de marcaje de las imágenes utilizadas en el entrenamiento se describe en el anexo **??**. Del total de la base de datos utilizada, dispondremos del 80 % para el entrenamiento, el 20 % restante se utilizará en el ajuste de los modelos seleccionados (10 % como imágenes de prueba) y para la validación del modelo completo (10 % como imágenes de validación).

En el diagrama 2.4 se muestran las etapas del entrenamiento de la red. Para la detección y clasificación se utilizó una red neuronal convolucional ya entrenada. La red utilizada es YOLO y se profundizó en sus características en el capítulo 1.2.3. Para el entrenamiento de la red se debe contar con un mínimo de archivos que contengan las topología de la red a utilizar como la estructura de la red, las clases, y los pesos de inicialización de la red. La estructura de la red puede modificarse para obtener mejores resultados.

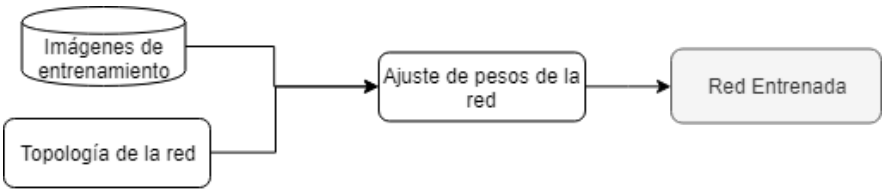

Figura 2.4: Diagrama del entrenamiento de la red convolucional

Dentro de las modificaciones de la red es posible variar la estructura, los pesos de inicialización del entrenamiento y el número de categorías. Las características de las redes a utilizar se muestran en la tabla 2.1. Se consideró fijo el número de bach en 64, las subdivisiones en 8 y se fija el número máximo de 25.000 bach. Éstos parámetros corresponden a los parámetros sugeridos, pero pueden ser modificados sin afectar el resultado de las métricas.

Nombre	Red	Pesos	Categorías
Red_1	yolov2-tinny	darknet53.conv.74.	1
Red_2	yolov2-tinny	darknet53.conv.74.	2

Tabla 2.1: Topología de las redes a utilizar

Para definir los parámetros de la red, como el threshold o umbral de detección de la red, la mejor época de entrenamiento y el número de clases se ejecutó YOLO Tiny, que corresponde a una implementación de red liviana, pero con buena performance. El entrenamiento de YOLO tiny se inicializa con los pesos sugeridos, darknet53.conv.74.

Para el cálculo del rendimiento de las diferentes configuraciones de red se utilizan las imágenes de prueba. Se implementó la descripción de métricas de Recall, Accuracy e IoU aplicadas sobre cada detección. Se entrega, para cada imagen, el valor de las métricas de cada detección y el promedio de las métricas sobre la imagen. Cuando se aplica sobre múltiples imágenes, el promedio de las métricas se hace sobre todas las detecciones por separado.

Se agregó un análisis adicional de sensibilidad que permite obtener la cantidad de detecciones correctas sobre la imagen, sin considerar el porcentaje de la detección que coincide con el original. Esta métrica será útil para determinar la confiabilidad del conteo de caras.

2.1.2. Ajuste de contorno fino

En esta etapa se recibe como entrada el resultado de la etapa anterior, segmentos de cara. Para cada segmento de cara se evalúa si la porción visible de la cara detectada es mayor al 90 %. En caso positivo, se calcula el diámetro de cada cara y el diámetro promedio de la carga. Previo a la obtención del diámetro promedio se contempla una etapa de preprocesamiento y de segmentación de la imagen. El 'Modo On-line' del ajuste de contornos se presenta en el diagrama 2.5.

Adicional al modo de ejecución online se contemplan pruebas de las diferentes técnicas de preprocesamiento y algoritmos de segmentación sobre las imágenes. Se verificaran los parámetros a utilizar de cada filtro aplicado y la mejor elección del algoritmo de segmentación.

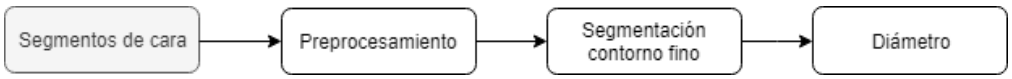

Figura 2.5: Ajuste de contornos, diagrama de la segunda etapa de desarrollo

La métricas utilizadas contemplan comparar el diámetro promedio con el diámetro obtenido desde una base de datos de contornos marcados de manera manual. La creación de la base de datos se presenta en el anexo **??**.

De ésta parte del proceso se determina el mejor algoritmo de preprocesamiento y segmentación, y se ajustan los parámetros para el conjunto de datos disponible.

Preprocesamiento de imágenes

Inicialmente se realizó una primera etapa de Preprocesamiento donde se observó el efecto de diferentes filtros por separado. Se busca comprender el efecto de las modificaciones de los parámetros sobre la imagen. Los filtros utilizados consideran Ecualización de histograma, manejo de ruido a través de escala de grises, filtro promedio, gaussiana y bilateral. Adicionalmente, se realizó una segunda prueba con combinaciones de filtros que permitan ajustar aún más el resultados.

De la literatura revisada es posible observar que diferentes configuraciones de filtros se pueden utilizar para mejorar el desempeño de los detectores de bordes. La elección del mejor dependerá de cada caso específico de estudio. Se consideraron cuatro configuraciones de filtros diferentes que se muestran en la tabla 2.2, se incluyen transformación en escala de grises, filtro de mediana, filtro gaussiano, aplicación de umbral y realce de contornos utilizando Canny.

Filtro 1	Filtro 2	Filtro 3	Filtro 4	Filtro 5
Gray	Gray	Gray	Gray	Gray
Mediana	Mediana	Mediana	Gaussiana	Gaussiano
	Gaussiana	Gaussiana	Umbral	Umbral
		Canny	Mediana	Mediana
				Canny

Tabla 2.2: Ejemplo de filtros aplicados en detección de imágenes

Las mejoras obtenidas en el desempeño, como consecuencia de las diferentes configuraciones de filtros se evalúan en la etapa de segmentación. Para realizar la evaluación se fija el algoritmo de segmentación y se evalúa el error sobre la detección para cada una de las configuraciones de filtros utilizada.

Segmentación contorno fino

Posterior al preprocesamiento de las imágenes y las pruebas de parámetros se realiza la segmentación de contornos finos. Esta etapa es crítica dentro del proceso, debido a que un resultado inadecuado ocasiona que la región segmentada no corresponda a una cara de tronco, o que sólo considere partes de ella, con lo que se obtendrán valores erróneos para el diámetro. La segmentación de la cara de troncos se basa en determinar el límite interior de la cara de cada tronco, eliminando la corteza y despreciando las marcas internas de la cara.

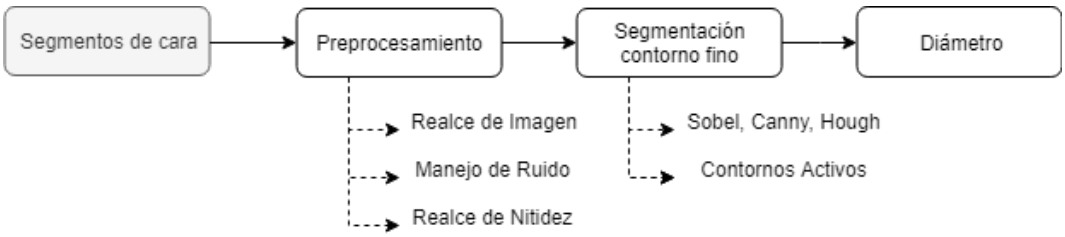

Figura 2.6: Diagrama detallado de la Etapa de Ajuste de Contorno

El diagrama que se muestra en la figura 2.6 muestra las herramientas utilizadas en cada etapa. Los algoritmos de segmentación utilizados consideran Sobel, Canny y Contornos activos. Cada uno de los algoritmos de segmentación se probaron antes y después de las configuraciones de filtros detallados. Finalmente, se realiza la evaluación del impacto del preprocesamiento sobre los resultados de la segmentación de caras de troncos para determinar el algoritmo que presente mejores resultados.

2.1.3. Descripción de Software

Detección de caras

Para la obtención de las métricas de la red entrenada se sigue el diagrama de la figura 2.7. Se consideran como datos de entrada Original_Image (OI) que corresponde a la imagen que se desea evaluar y Labelled_Image (LI) es un archivo txt que contiene los datos marcados para esa imagen. La salida storage.txt con las métricas de cada ejecución. En el capítulo **??** se muestran ejemplos de cada uno de los archivos de entrada y salida. Las funciones utilizadas son:

- execute_CNN: Ejecuta la evaluación de la imagen para una estructura de red especificada (categorías, estructura y pesos). Entrega los resultados en un archivo .txt por cada imagen evaluada. Esta función debe ser ejecutada en el servidor que cuente con todos los programas para ejecutar YOLO.
- resize_labelled_image: Obtiene los datos desde los archivos .txt de marcaje y los entrega como un arreglo, con el correspondiente reajuste de dimensiones. El reajuste de dimensiones se realiza multiplicando las variables por el largo y ancho de la imagen original.

- load_data: Obtiene los datos desde los archivos .txt entregados por la red y los entrega como un arreglo. Los archivos .txt deben ser obtenidos previamente con la ejecución de Execute_CNN.
- execute_metric: Se aplica a imágenes con datos de marcaje y permite validar las métricas obtenidas desde la red. Se realiza un emparejamiento de los datos de marcaje con los detectados por la red y para cada emparejamiento se entregan las métricas obtenidas.

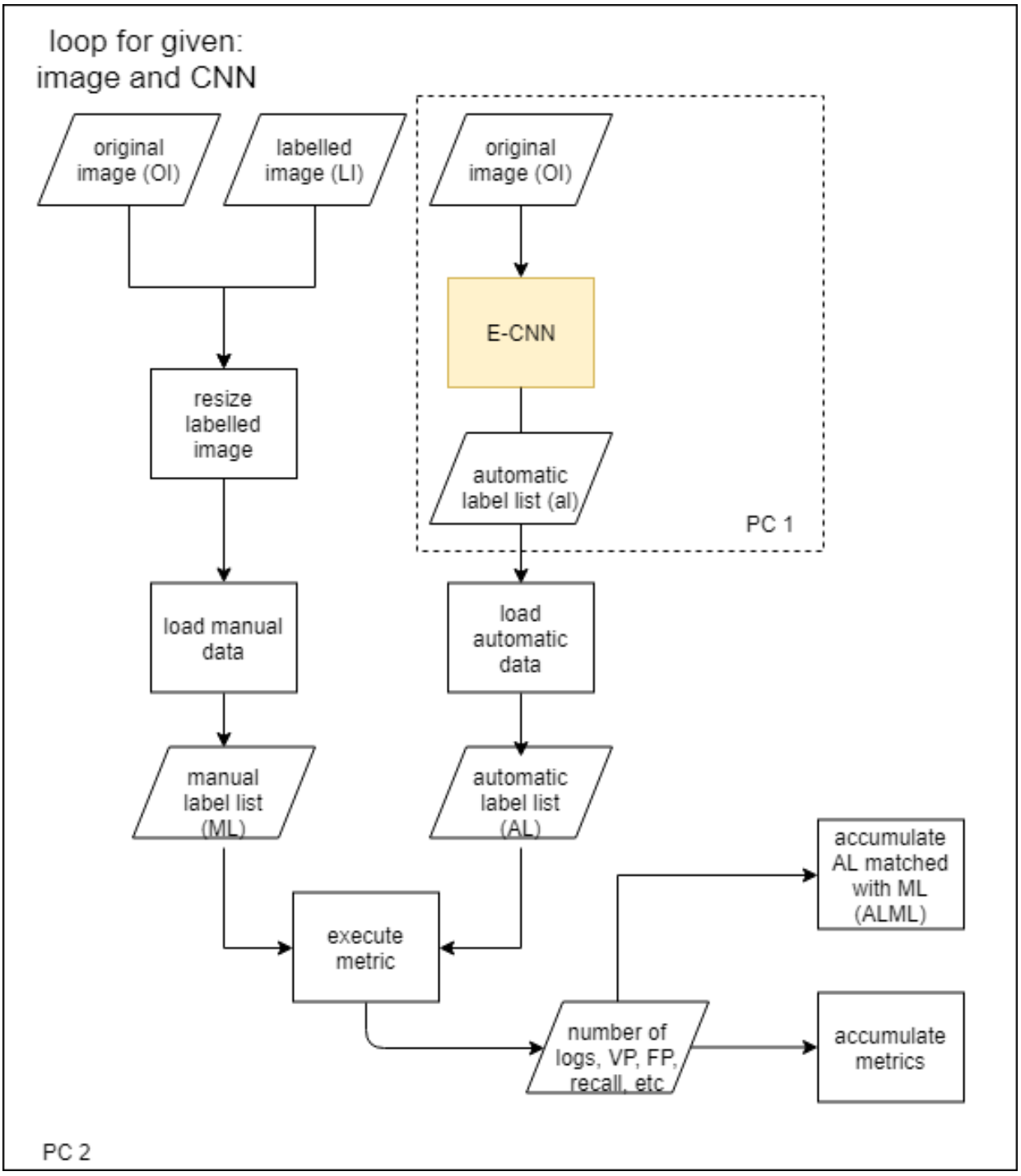

Figura 2.7: Diagrama detallado de la Etapa de obtención de métricas de la red

Ajuste de contorno fino

Para la obtención de las métricas de contornos activos se sigue el diagrama de la figura 2.8. Se consideran como datos de entrada Original_Image (OI) y Labelled_Image (LI) al igual que en la etapa anterior, y se agrega ALML que corresponde a un archivo de salida de ejecutar las métricas. La salida corresponde a la evaluación de contorno para todas las caras de la imagen detectadas como full y una métrica del diámetro promedio de la imagen. Las funciones utilizadas son:

- E-AC: tiene como principal objetivo obtener el contorno de las caras detectadas. El resultado esperado es un arreglo que contiene el diámetro de cada cara detectada. Cada columna dentro del arreglo contiene la información del diámetro obtenido para una contorno detectado. Por columna se entrega información de las coordenadas del centro del cuadro detectado para x e y, ancho, alto, categoría, tag, probabilidad de detección, radio de la adivinanza inicial, radio del contorno.
- load_data: Obtiene los datos desde los archivos .txt entregados por la red y los entrega como un arreglo. Los archivos .txt deben ser obtenidos previamente con la ejecución de Execute_CNN.
- Filter_ALML: Tiene como principal objetivo filtrar los resultados del emparejamiento para los correspondientes a la cara Full. Entrega como resultado un arreglo que contiene el diámetro de cada cara detectada. Cada columna dentro del arreglo contiene el diámetro del contorno obtenido sobre la cara detectada. Por columna se entrega información de x_centro, y_centro, ancho, alto, categoría, tag, probabilidad de detección, radio de la adivinanza inicial, radio del contorno.
- execute_contour_metric: Tiene como objetivo obtener el contorno de las caras detectadas. Entrega como resultado un documento de texto que contiene el diámetro de cada cara detectada. Cada columna dentro del arreglo contiene el diámetro del contorno obtenido sobre el prior de contorno activo, utilizando contorno activo y para los datos de marcaje. Adicionalmente se entrega el radio promedio de cada imagen.

Diseño del Modo On-line

Bajo el nombre de Modo On-line se hará referencia al sistema basado en imágenes para la estimación de radio (píxeles) y número de troncos sobre bancos de madera cargados en camiones utilizando procesamiento de imágenes. El Modo on-line tiene como salidas principales la cantidad de troncos por carga y una lista con el diámetro de cada una de las detecciones. Las variables adicionales que se entregan en el modo on-line están orientadas a describir las características de la carga (por ejemplo el diámetro promedio).

La operación del Modo On-line es utilizando como entrada una imagen de la carga del camión. El modelo utiliza actualmente imágenes de un Portal de Logmeter5000, con imágenes capturadas con cámaras de la marca Vivotek, modelo IB9371-HT. Este módulo incluye la ejecución de la red neuronal convolucional, que tiene la información de los pesos de la red ya integrados. Para su ejecución se debe mantener la estructura de carpetas presentada.

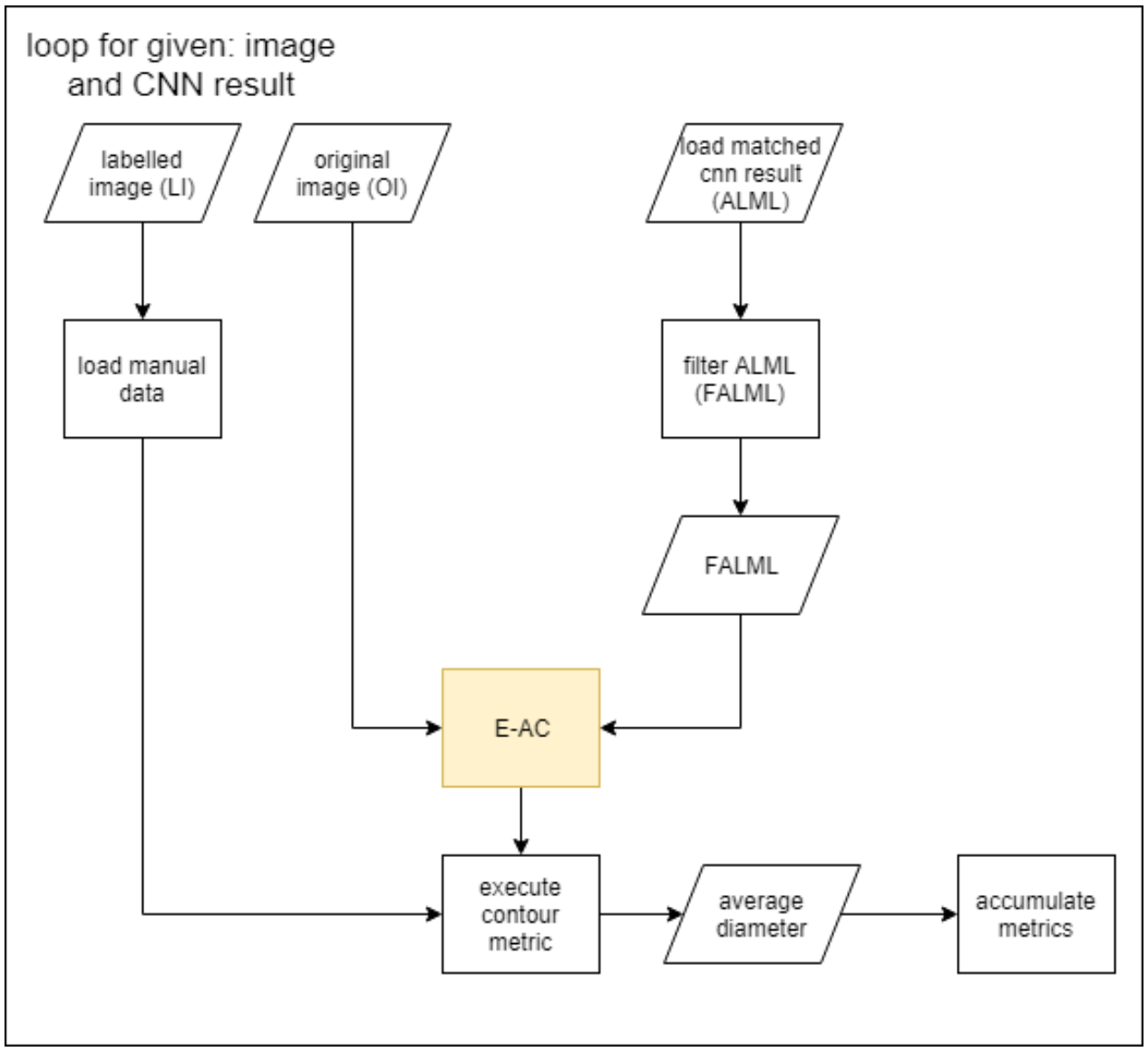

Figura 2.8: Diagrama detallado de la Etapa de Ajuste de Contorno

En el diagrama 2.9 se muestra la interacción de roles del Modo On-line. Se describen tres funciones principales; E-CNN (Execute Convolutional Neural Network), E-AC (Execute Active Contour) y E-LNE (Execute Log Number Estimator). A continuación se presenta la descripción de cada uno de las funciones del Modo On-line, sus objetivos y los resultados esperados. Las salidas de este modulo es el número de troncos (number of logs) y el promedio de diámetros (average diameter) para la imagen.

Roles

E-CNN tiene como principal objetivo la detección y localización de caras dentro de la imagen. El resultado esperado es un documento de texto que contiene las detecciones de la red. Cada línea en el archivo de salida corresponde a una detección de la red. Cada línea

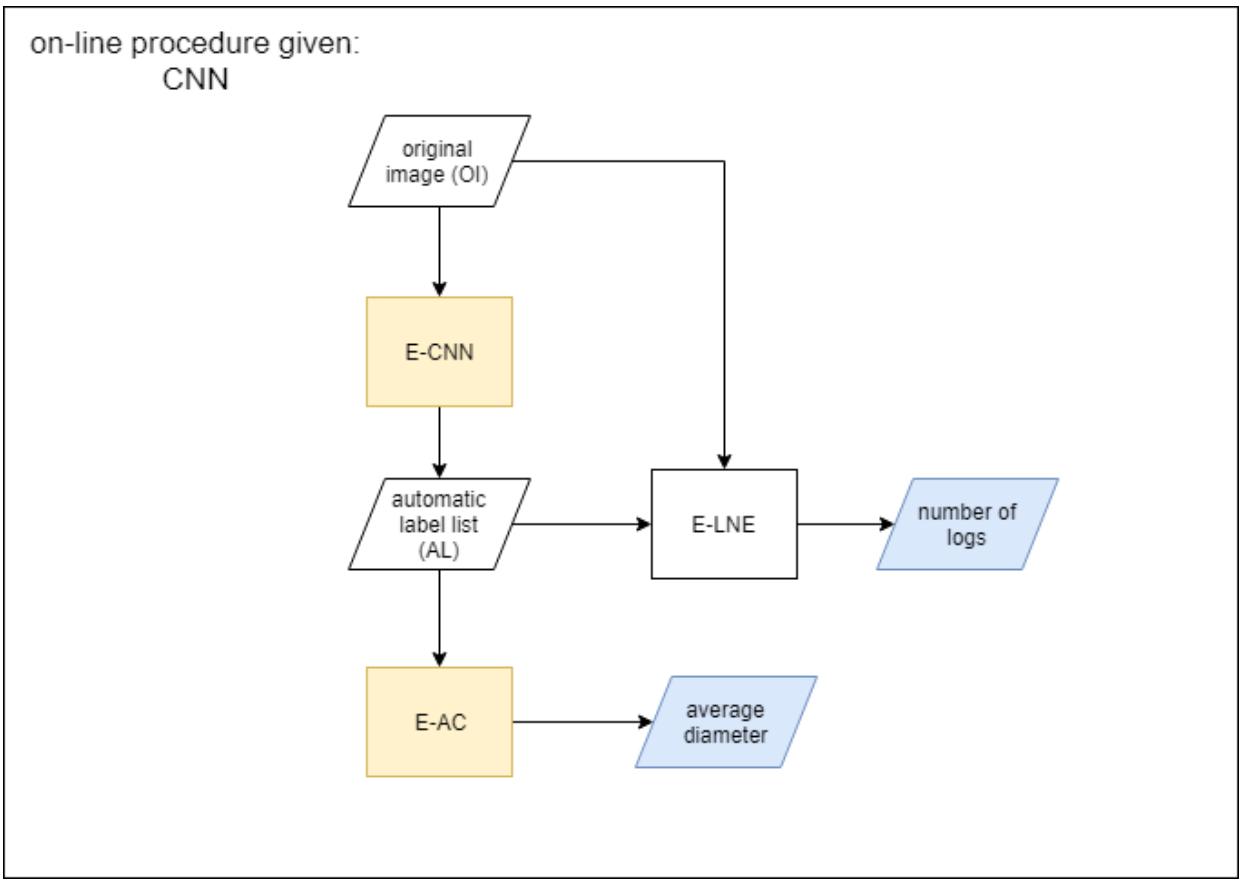

Figura 2.9: Diagrama detallado del Modulo On-line

contiene información de la clase, las coordenadas del centro del cuadro detectado para x e y, alto, ancho, el número indicador del tronco (se puede ver en las imágenes de salida), la probabilidad de la detección y el tiempo de ejecución.

E-AC tiene como principal objetivo obtener el contorno de las caras detectadas. El resultado esperado es un arreglo que contiene el diámetro de cada cara detectada. Cada columna dentro del arreglo contiene la información del diámetro obtenido para una contorno detectado. Por columna se entrega información de las coordenadas del centro del cuadro detectado para x e y, ancho, alto, categoría, tag, probabilidad de detección, radio de la adivinanza inicial, radio del contorno.

E-LNE tiene como principal objetivo retornar el número de caras detectadas. Esta función toma como entrada las detecciones de la red con detecciones mayor a un umbral conocido. El resultado esperado es la cantidad de troncos dentro de la imagen. Adicionalmente se entrega una imagen con los recuadros correspondientes a las detecciones de la red, mostrando en color cian los recuadros de categoría Full y en blanco los de categoría Partial. En la imagen se muestra además el número de referencia para cada tronco.

2.2. Métricas de Evaluación

Las métricas seleccionadas buscan abordar los dos temas principales: evaluar el rendimiento de la detección de caras y segmentación de bordes. Se complementan Medidas de Precisión Media (mAP) para Detección de objetos con métricas de Análisis de Sensibilidad y Especificidad. Cada métrica evalúa criterios diferentes. Por ejemplo la exactitud mide la certeza de las predicciones, es decir, el porcentaje de las predicciones que es correcto. La medida de Recall mide que tan bien se es capas de encontrar todos los positivos.

2.2.1. Métricas para Detección de Caras

Durante el entrenamiento es posible ver algunos indicadores propios de la red neuronal convolucional YOLO, estos corresponden a las primeras métricas utilizadas para la verificación del rendimiento de la red. Durante la etapa de entrenamiento, por cada cien épocas, se muestra: la pérdida total, el promedio de error de pérdida, taza de aprendizaje actual, el tiempo total empleado en el batch y el total de imágenes empleadas.

Para entender las métricas, es necesario definir que se entiende por objeto real y detección. El objeto real corresponde a los datos marcados inicialmente, por un usuario experto, incluyendo la posición y la categoría de la cara. Consideramos como detección a los resultados del modelo sobre la imagen. Definiremos, adicionalmente tres métricas para evaluar sobre los box detectados. La exactitud mide la certeza de las predicciones. El valor de Recall mide que tan bien encuentra todos los positivos e Intersección sobre Unión (IoU, Intersection Over Union) es una medida del traslape entre el contorno real de un objeto y el valor predicho.

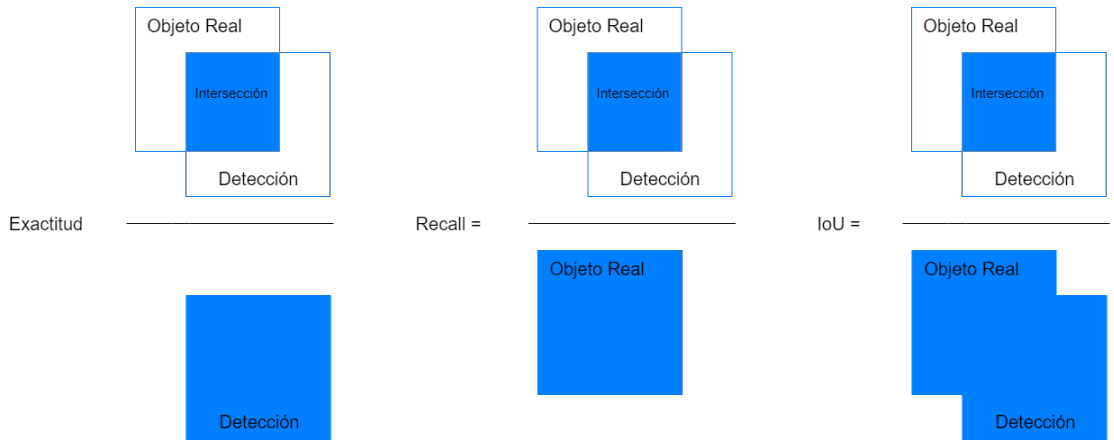

Figura 2.10: Métricas de evaluación del desempeño de la red

Como se observa en la figura 2.10, la medida de Exactitud relaciona la intersección entre el objeto real y el detectado sobre la detección. La medida de Recall relaciona la intersección entre el objeto real y el objeto detectado sobre el objeto real. En IoU se considera la misma intersección del objeto real con el objeto detectado, pero ahora, sobre la unión de ambos recuadros. Si IoU alcanza el 100 % se obtendrán detecciones que coincidan completamente con el objeto real.

Para evaluar los resultados es posible utilizar los índices que se muestran en la tabla 2.11. Para obtenerlos se fija un umbral de discriminación, que corresponde al valor a partir del cual se decide que un caso es positivo. Por lo tanto, las detecciones que superen el umbral de discriminación se clasificaran como VP y las que no como FN. Estos parámetros permiten realizar una representación gráfica de la sensibilidad frente a la especificidad para un sistema clasificador binario.

Hay cuatro posibles resultados para un clasificador binario. Cuando la prueba declara un valor positivo y es realmente positivo se conoce como verdadero positivo o VP. Cuando la prueba declara positivo y en realidad es negativos, se conoce como falsos positivos o FP. Verdadero negativo, o VN, indica el casos en que la prueba declara negativo y que es realmente negativos. Falsos negativos, o FN, es cuando la prueba declara negativo y en realidad es positivos.

		Resultados de Clasificación	
		Objeto Presente	Objeto No Presente
Instancias Reales	Objetos Presentes	Verdaderos Positivos	Falsos Negativos
	Objetos no Presentes	Falsos Positivo	Verdaderos Negativos

Figura 2.11: Matriz de contingencia

Con los valores descritos es posible realizar el análisis de la curva ROC (Características Operativas del Receptor o ROC por sus siglas en ingles, Receiver Operating Characteristic). Con éste análisis es posible determinar los modelos óptimos. La utilidad de la curva ROC es en la generación de estadísticos que resumen el rendimiento del clasificador. Nos permitirá comparar dos pruebas diferentes al comparar el área bajo la curva (AUC) con valores entre 0,5 (prueba sin capacidad discriminatoria) y uno (valor de diagnóstico perfecto).

La curva ROC muestra la razón de verdaderos positivos (VPR) y de falsos positivos (FPR). VPR es equivalente a la sensibilidad y mide hasta que punto se pueden clasificar los casos positivos de manera correcta, de entre todos los casos positivos. FPR es 1-especificidad, mide los casos incorrectos de entre todos los casos negativos disponibles. Los valores de VPR (o sensibilidad) y FPR (o 1-especificidad) se muestran en las ecuaciones 2.1 y 2.2 respectivamente.

$$Sensibilidad = VPR = \frac{VP}{VP + FP} = \frac{VP}{Total\ Resultados\ Positivos} \tag{2.1}$$

$$Especificidad = SPC = 1 - FPR = \frac{VN}{VN + FP} = \frac{VN}{Total\ Casos\ Negativos} \tag{2.2}$$

Es posible obtener algunas métricas adicionales durante el entrenamiento de la red como la iteración actual de entrenamiento, la pérdida total, el error de pérdida promedio, la taza de aprendizaje, el tiempo toral de procesamiento de la subdivisión actual y el total de imágenes utilizadas durante el entrenamiento. El error de pérdida promedio debe ser tan pequeño como

sea posible, un buen criterio de detención del entrenamiento es cuando es menor a 0,06 avg. La cantidad de imágenes varia por data augmentation.

Durante la ejecución de la red, se muestran algunas métricas adicionales como el promedio de IoU, clase, count y promedio de Recall. El promedio de IoU entrega el promedio de la métrica para todas las imágenes evaluadas en la subdivisión. La clase en la que fue clasificada la imagen. Count corresponde a la cantidad de imágenes dentro de la subdivisión para la que se detectaron objetos. El promedio de Recall ($recall/count$) es una métrica del total de positivos correctamente detectados.

2.2.2. Métricas para ajuste de contorno fino

Para evaluar los resultados sobre contornos activos se utilizó un primer criterio de inspección visual, el cuál evaluó que la curva detectada como borde no se aleje demasiado en ningún punto del contorno real. En el ejemplo de la imagen 2.12 se ejemplifica el criterio evaluado, en la imagen de la izquierda se tiene un punto en la detección donde la curva detectada se aleja de la curva real. A la derecha se observa un ejemplo de detección positiva, con una curva muy cercana al contorno real.

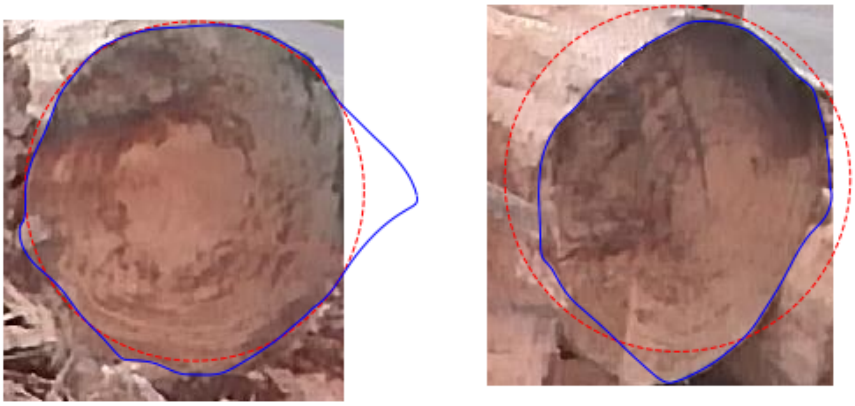

Figura 2.12: Inspección visual sobre detección de contornos. Ejemplo de un mal ajuste de contorno (izq) y de un buen ajuste (der)

Además, se utiliza un segundo criterio, que nombraremos como el diámetro promedio. Para esto, comparamos el diámetro obtenido de las curvas de contornos marcadas manualmente con el diámetro estimado utilizando detectores de bordes. Para obtener el diámetro promedio sobre los datos de marcaje, se debe obtener, en primer lugar, un arreglo que contenga todos los puntos de la marca manual. Al ser una imagen en blanco y negro, al aplicar directamente contornos activos, se ajusta a la curva marcada en un 99.5 % de los casos.

El diámetro promedio se calcula fijando el centro de la imagen y promediando la distancia de cada uno de los puntos del arreglo al centro propuesto. Este procesamiento se realiza para obtener el diámetro promedio de la marca y el de los datos obtenidos con el detector de bordes. La ecuación 2.3 muestra la forma de agregar una tolerancia al comparar los diámetros obtenidos. Se considerará un umbral (ρ) que define un rango para el cual la diferencia entre

el radio manual (r_m) y el radio del detector de bordes (r_d) será despreciado.

$$f(n) = \begin{cases} (r_m - r_\mathrm{d}) - \rho \cdot r_m & \text{si } |r_m - r_\mathrm{d}| > \rho \cdot r_m \ \textit{con } r_m > r_\mathrm{d} \\ (r_m - r_\mathrm{d}) + \rho \cdot r_m & \text{si } |r_m - r_\mathrm{d}| > \rho \cdot r_m \ \textit{con } r_m < r_\mathrm{d} \\ 0 & \text{si } |r_m - r_\mathrm{d}| \leq \rho \cdot r_m \end{cases} \tag{2.3}$$

Con la comparación de los diámetros calculados se obtendrá el error en la medición para cada una de las caras detectadas como categoría Full. Para obtener el promedio del error, por imagen, se promediara el error de cada una de las caras evaluadas dentro de la imagen. Para evaluar el promedio del error sobre un conjunto (Prueba o Validación) se considerara cada cara por separado, sin hacer distinción de la imagen a la que pertenece. Los errores mencionados podrán tomar tanto valores negativos como positivos.

Con la comparación del diámetro promedio, el cálculo del promedio del error y la desviación estándar sobre un conjunto de imágenes no es posible comparar los puntos de coincidencia entre ambas curvas y tampoco detectar puntos donde la curva detectada se aleje considerablemente de la curva real. Sin embargo, si nos entrega información general del comportamiento del detector y la cercanía de los radios estimados al valor de radio real.

$$e = \textit{Error absoluto} = \sqrt{\sum \frac{(\textit{Valor medido} - \textit{Valor exacto})^2}{N \cdot (N-1)}} \tag{2.4}$$

Para obtener la incertidumbre de la medición utilizaremos el error absoluto y el error relativo. El error absoluto corresponde a la diferencia entre el valor medido y el considerado como exacto. La ecuación que se debe utilizar corresponde la ecuación 2.4. En esta ecuación se tiene como numerador la suma de cada valor medido menos el valor exacto de los datos y en el denominador el valor de N corresponde al número de datos. Este error absoluto corresponde al error del medidor.

$$E = \textit{Error relativo} = \frac{e}{\textit{Valor exacto}} \cdot 100 \tag{2.5}$$

El error absoluto corresponde al valor mayor entre el error del medidor y el error del aparato. Consideraremos el error del aparato como despreciable y por consiguiente solo se considerara como error absoluto el valor del error del medidor. El error relativo es el error absoluto sobre el valor exacto (ecuación 2.5). Este valor se mide en porcentaje y se utiliza para determinar la precisión de la medición, ya que entrega la proporción entre el error y el valor exacto.

2.3. Descripción de Hardware

La transferencia de conocimiento permite disminuir el tiempo de entrenamiento, utilizando redes que ya fueron entrenadas para clasificación y que por lo tanto sus capas no requieren de grandes modificaciones para agregar una nueva clase. Los ejemplos de uso son muy diversos, en [12] se utiliza YOLO9000 para la detección de equipos de protección personal(EPP) utilizando para su entrenamiento un ordenador de 16 GB de memoria, tarjeta gráfica NVIDIA GeForce GTX 1080 TI y un procesador Intel Core i5.

	YOLO	**Memoria**	**Tarjeta gráfica**	**Procesador**
[12]	YOLO9000	16GB	NVIDIA GeForce GTX 1080 TI	Intel Core i5
[11]	YOLOv3	12GB	NVIDIA GeForce GTX TITAN XP x4	No informado
[26]	YOLOv3	8GB	NVIDIA GeForce GTX 1080 TI	No informado

Tabla 2.3: Comparación de Hardware para diferentes proyectos que utilizan YOLOv3

Otro ejemplo de implementación utilizando YOLOv3, en [11] se utiliza YOLOv3 para detección de vehículos utilizando para entrenamiento cuatro NVIDIA GeForce GTX TITAN XP GPU con 12GB de memoria. En [26] se evalúa el estado de los rieles utilizando YOLOv3 las características de entrenamiento son 32GB RAM, 3.2 Hz CPU y GTX 1080 GPU con 8 GB de memoria. Existen diversos ejemplos de utilización de YOLO, con los que podemos estimar un punto de partida para el Hardwere utilizado en la implementación.

En resumen, para el aprendizaje profundo es altamente recomendable disponer de GPU, ya que permite mejorar hasta en 100 veces la velocidad de ejecución. Las cualidades de la tarjeta dependerán de su uso y el presupuesto. Si se busca eficiencia, partir desde la 980 Ti hacia arriba. También es necesario considerar que si se tienen varios procesos simples simultáneamente, es mejor tener varias de menor performance. En cuanto a memoria 32 Gb hacia arriba es una cantidad suficiente.

Se dispuso de un computador exclusivo para el entrenamiento de la red que presenta las siguientes características: se utilizó una tarjeta NVIDIA GeForce GTX 2080 ti que cuenta con Baffer de 11 GB GDDR5X, Memoria 8 GB GDDR5 (256 bit), bus PCI Express3.0x16, frecuencia Core 1607 MHz, frecuencia Memorias 2002 MHz. El servidor utilizado cuenta con CPU Intel(R) Core(TM) i7-6700K CPU @ 4.00GHz Y Memoria KINGSTON 8G, DDR4, 2133 MT/s X2.

Para validar los resultados obtenidos por la red, se incluye una etapa de caracterización de la red. Adicionalmente se crea un marco de trabajo que permite ejecutar el modo online del proyecto. Para lo anteriormente señalado se utilizo un segundo servidor que cuenta con las siguientes características: CPU Intel(R) Core(TM) i7-6700K CPU @ 4.00GHz. GPU ZOTAC -GeForce GTX 950. 2Gbps, Boost Clock de 1652 MHz y memoria KINGSTON 8G, DDR4, 2133 MT/s X2.

Capítulo 3

Resultados

3.1. Base de datos

Las imágenes se encuentran en un sistema desarrollado por Woodtech, XMeterR, que gestiona las librerías, módulos y herramientas de los diferentes productos que pertenecen a la marca. Cada imagen es generada por un portal de Logmeter5000 y se encuentran divididas por reporte de eventos. Se define como evento el paso de la carga por un punto de medición, o portal Logmeter5000. Cada evento contienen entre 40 a 70 capturas secuenciales.

Figura 3.1: Ejemplo de las tres cámaras para un mismo instante de tiempo: Cámara uno (izq), Cámara dos (cen), Cámara tres (der)

Los eventos son almacenados como un conjunto de múltiples imágenes de la carga, obtenidas desde tres posiciones diferentes para cada captura. En la imagen 3.1 se muestra un ejemplo de las tres cámaras para un mismo instante de tiempo. Se utilizaron las dos cámaras con enfoques laterales de la carga. Para la creación de la base de datos se seleccionaron 118 eventos. Para cada evento se seleccionaron a mano las dos imagen (cámaras laterales) que muestran la mayor cantidad de caras, dando un total de 236 imágenes.

Las imágenes seleccionadas para la base de datos se dividen en tres conjuntos fundamentales: entrenamiento, prueba y validación. El 81.4 % de los datos se utilizan en el entrenamiento de la red, estos datos son los que posibilitan que los modelos aprendan. El 9.3 % se utiliza en test, estos datos permiten seleccionar el mejor de los modelos entrenados. El otro 9.3 %

corresponde a validación, y permite obtener un error real para el modelo seleccionado. En total se utilizaron 236 imágenes cuya distribución se muestra en la figura 3.2.

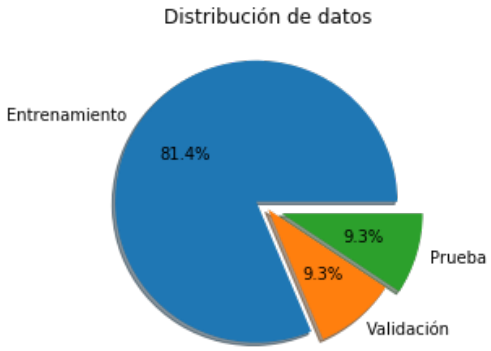

Figura 3.2: Distribución de los conjuntos fundamentales de la base de datos

Desde ahora nos referiremos al conjunto de imágenes seleccionadas manualmente desde los eventos para las dos cámaras laterales como DS_carga. En la imagen 3.2 se muestra la distribución de las imágenes en los conjuntos mencionados. Del total de imágenes en DS_carga, 192 corresponden a datos de entrenamiento con 3.694 caras, 22 a datos de prueba con 405 caras y 22 a datos de validación con 403 caras. Cada imagen contiene en promedio 19 caras de troncos marcadas.

Figura 3.3: Ejemplo de imágenes pertenecientes a la categoría Full

Figura 3.4: Ejemplo de imágenes pertenecientes a la categoría Partial

Las caras presentes en las imágenes de DS_carga pueden requerir de preprocesamiento cuando los contornos no son fácilmente distinguibles. Se definieron dos conjuntos de datos

con características similares para facilitar el ajuste de los algoritmos utilizados. Se proponen dos clases: 'Full' y 'Partial'. La clase 'Full' corresponde a las caras de troncos que tienen más del 90 % visible, se muestran ejemplos de esta clase en la imagen 3.3. La categoría 'Partial' corresponden a las que se ve menos del 90 % de la cara, se presentan ejemplos de esta clase en la imagen 3.4.

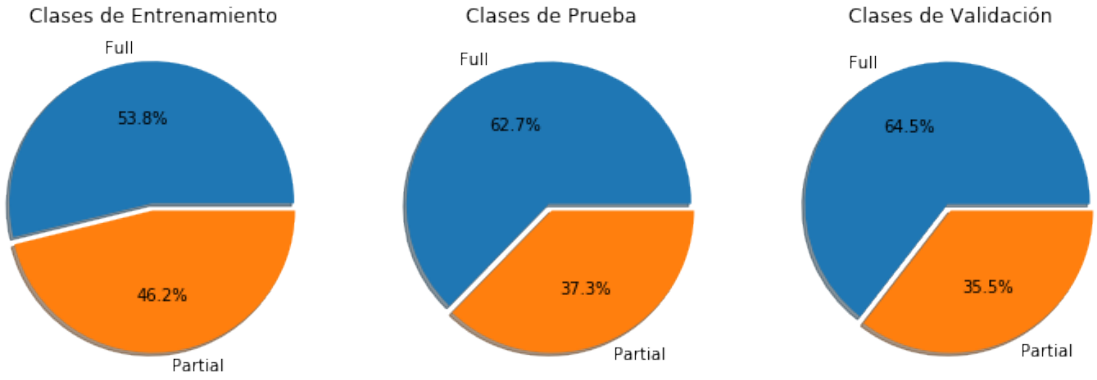

Figura 3.5: Distribución de las clases de DS_carga para cada subconjunto (de izquierda a derecha, clases del conjunto de entrenamiento, prueba y validación

Del total de caras de la muestra, el 55.58 % corresponde a caras de la categoría 'Full' y 44.42 % a caras de la categoría 'Partial'. La distribución de las clases, para cada subconjunto, se muestra en la figura 3.5, de izquierda a derecha se presenta la distribución de los datos de entrenamiento, prueba y validación respectivamente. Adicionalmente, en la tabla 3.1 se muestra un resumen con las características mencionadas como la cantidad de imágenes, número total de troncos y la cantidad de troncos en cada clase, para cada conjunto.

Conjunto	Imágenes	Número de troncos	Full	Partial
Total	236	4502	2502	2000
Entrenamiento	192	3694	1988	1706
Prueba	22	405	254	151
Validación	22	403	260	143

Tabla 3.1: Características de los conjuntos de Entrenamiento, Test y Validación

Para DS_carga se marcó a mano la posición de cada cara utilizando OpenLabeling y se guardó la información de categoría, centro en x, centro en y, ancho y alto para cada cara. Finalmente DS_carga contiene, para el total de imagenes, un archivo de texto con la posición de todas las caras presentes, ordenando las características de cada cada marcada en una línea del archivo de texto. Las instrucciones para crear la base de datos se encuentran en el Anexo C y se muestran ejemplos del archivo mencionado en Anexo F.

Para la detección de contornos fue necesario implementar una segunda base de datos, que llamaremos desde ahora DS_contorno. Esta base de datos contiene una carpeta por cada imagen del conjunto de prueba y validación. La carpeta cuenta con una imagen con fondo blanco, el contorno en negro y con el mismo tamaño de la imágen inicial, para cada cara. Los contornos fueron marcados a mano utilizando la aplicación sketch. En la figura 3.6 se muestran algunos ejemplos.

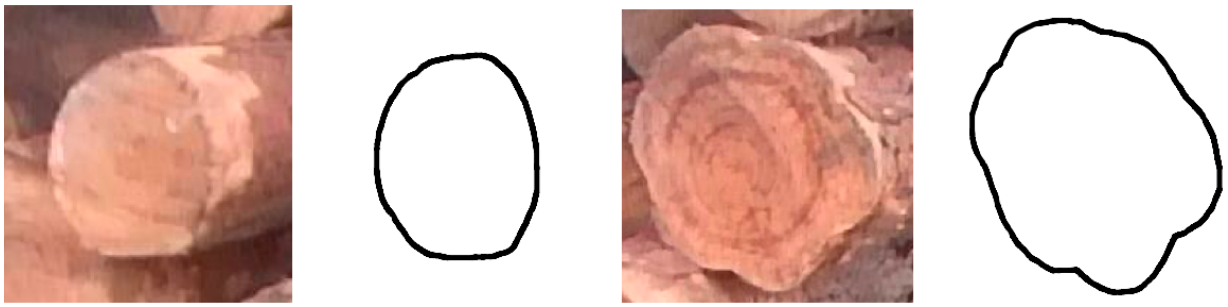

Figura 3.6: DS Contorno: Ejemplo de caras Full y su correspondiente contorno en negro

En resumen, DS_carga contiene las imágenes y los datos de posición de las caras dentro de la imagen. Nos entrega los datos necesarios para el entrenamiento de la red y la verificación de sus resultados, por lo que será especialmente útil en la etapa de desarrollo, 'Detección de caras'. La finalidad de DS_contorno es validar los resultados obtenidos en la etapa 'Ajuste de contornos' y, en consecuencia, de la implementación de los algoritmos de segmentación sobre las imágenes.

3.2. Definición de pruebas a realizar

Dentro de la implementación del prototipo, que permita estimar el número de caras de troncos para una imagen y entregar el diámetro correspondiente, se definen diferentes etapas. Se implementa la etapa de 'Detección de caras' para obtener las regiones de interés. Sobre las regiones de interés detectadas se implementa la etapa de 'Ajuste de contorno fino'. Para las dos etapas mencionadas, se ejecutara un proceso paralelo de verificación del algoritmo. Para el proceso de desarrollo se definen las siguientes pruebas:

- Determinar la red a utilizar para la detección de las regiones de interés dentro de la imagen. Considerando las características de la red y de la base de datos.
- Seleccionar el umbral de la red que permita maximizar los verdaderos positivos y al mismo tiempo minimizar los falsos positivos y falsos negativos.
- Considerando los dos objetivos principales, conteo de troncos y obtención de diámetros, evaluar la mejor época de entrenamiento para cada uno de ellos.
- Verificar las métricas obtenidas para el primer objetivo, conteo de caras, y revisar los principales casos de interés.
- Determinar la mejor configuración de filtros a utilizar como preprocesamiento. Evaluando el efecto de modificar sus parámetros.
- En la etapa de segmentación, evaluaremos diferentes modelos y el efecto de las etapas de preprocesamiento sobre el ajuste de contornos.
- Evaluar el desempeño del modo online de ejecución y los tiempos de procesamiento.

3.3. Resultados y Discusión

3.3.1. Etapa: Detección de caras

Como se mencionó en el capítulo 1, la gran versatilidad de redes neuronales y la posibilidad de actuar dentro de todas las etapas del preprocesamiento de imágenes. Se decidió utilizar YOLO para la detección de las zonas de interés. YOLO detecta una gran variedad de objetos, sin embargo los segmentos de caras de troncos no están incluido en la base de datos. El primer paso será decidir la configuración de red a utilizar. Posteriormente se caracterizará el funcionamiento de la red.

Elección de la red a utilizar

En la sección 1.2.3, en la tabla 1.1 se muestran diferentes topologías de YOLO. Como la implementación futura considera la obtención de imágenes a través de cámaras del modelo BFLY-PGE-50S5C-C, consideraremos solo las que permitan valores superiores a 20fps. Las topologías evaluadas se muestran en la tabla 3.2. Las métricas mostradas en la tabla corresponden a las descritas en la sección 2.2.

red	época	total loss	avg loss error	rate	seconds	images	tiempo
yolov2.cfg	20000	29.212	29.241	0.001	1.518	20000	34
yolov2-tiny	20000	7.927	7.654	0.001	0.009	1280000	2
yolov3.cfg	10910	0.582	0.769	0.001	5.735	698240	21
yolov3-tiny	20000	3.547	2.931	0.001	0.597	1280000	12

Tabla 3.2: Comparación de diferentes topologías de YOLO

De las configuraciones mostradas yolov2 y yolov3 divergen después de alcanzar el máximo, tienen los mayores tiempos de entrenamiento y el menor número de imágenes de entrenamiento. Al utilizar yolov2-tiny y yolov3-tiny se observa que el mejor desempeño corresponde a la red yolov3-tiny, sin embargo, para modelo de prueba utilizaremos yolov2-tiny. La utilidad de usar yolov2-tiny es un menor tiempo de entrenamiento en desmedro de aumentar el error total.

Valor de Threshold

El valor de umbral o threshold permite limitar las predicciones de clase fijando un valor de probabilidad mínimo. Para todas las detecciones de la red se mantendrán solo las detecciones con probabilidad mayor al valor del umbral. El threshold de la red, por defecto, permite que sólo las caras detectadas con 25 % de certeza o superior sean consideradas como detecciones correctas. Es posible modificar su valor dependiendo de la precisión que se desee obtener con el clasificador y el caso de uso.

Para los ejemplos de la figura 3.7, se observan variaciones en el valor del Threshold. De

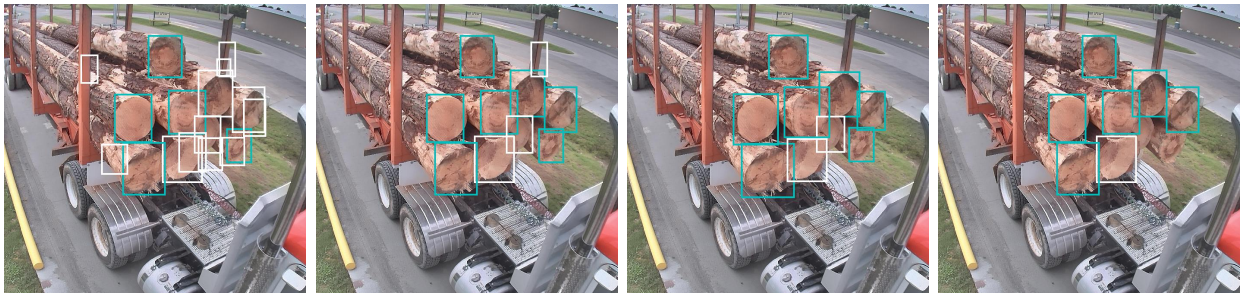

Figura 3.7: Variación de Threshold en la Red, (de izquierda a derecha 1 %, 25 %, 50 % y 75 % respectivamente)

izquierda a derecha se varían los umbrales, se muestran ejemplos para 1 %, 25 %, 50 % y 75 %. Se observa que para umbrales mayores al valor por defecto (umbral de 1 %) se detectan el mayor número de caras, pero también se obtiene un número mayor de detecciones erróneas. Si se realiza la implementación considerando un umbral menor al sugerido es necesario realizar un procesamiento adicional para eliminar las detecciones erróneas.

Para el valor de umbral sugerido (umbral de 25 %) se mantiene la cantidad de caras detectadas y disminuyen las detecciones erróneas (FP). Un umbral mayor al valor predeterminado, como 50 %, presenta una buena detección minimizando los falsos positivos. Para valores aún mayores, como 75 %, se eliminan algunas de las detecciones correctas (FN). Un umbral de 50 % permite dar confiabilidad a la red, ya que se minimizan las detecciones de falsos positivos al mismo tiempo que se mantienen las detecciones correctas. Para todos los desarrollos siguientes se considera un umbral de 50 % en la detección.

Elección de las categorías

Los datos de entrenamiento utilizados, descritos en la sección 2, incluyen información de las dos categorías principales; Full y Partial. Para la red seleccionada se utilizaron dos arquitecturas diferentes. Las arquitecturas propuestas buscan evaluar el efecto de la cantidad de clases en la correcta segmentación de imágenes. Consideraremos, para todas los entrenamientos, inicializar los pesos con darknet53.conv.74.

Las variaciones consisten en modificar el número de clase, se denominará Red_1 a la red que considera todos los datos marcados pertenecientes a la categoría Full. Se considera Red_2 a la red que considera, en la etapa de entrenamiento, dos categorías diferentes Full y Patial. Esta modificación permite identificar el efecto de utilizar todas las caras de una imagen para entrenar la categoría Full o utilizar sólo las que presentan sobre un 90 % visible. Adicionalmente, para cada red, se realizan pruebas variando su configuración interna.

Para las dos redes entrenadas se muestra el gráfico de caracterización del entrenamiento. Se incluyen métricas de evaluación del rendimiento de la red como Precisión, Recall e IoU para las veinte mil primeras épocas. También se muestra las métricas de VP, VN y FP en cada red. Se muestran los resultados de las métricas de evaluación de rendimiento (izq) y de la variación de VP, FP y FN (der) en las imágenes siguientes.

Para los gráficos mostrados, se calculó cada una de las métricas sobre el conjunto de datos de Prueba. Los valores de las métricas se calcularon para el total de caras detectadas, sin hacer distinción por imagen. En el gráfico 3.8 se muestran los resultados obtenidos para la Red_1 y en 3.9 se muestran los resultados para la Red_2.

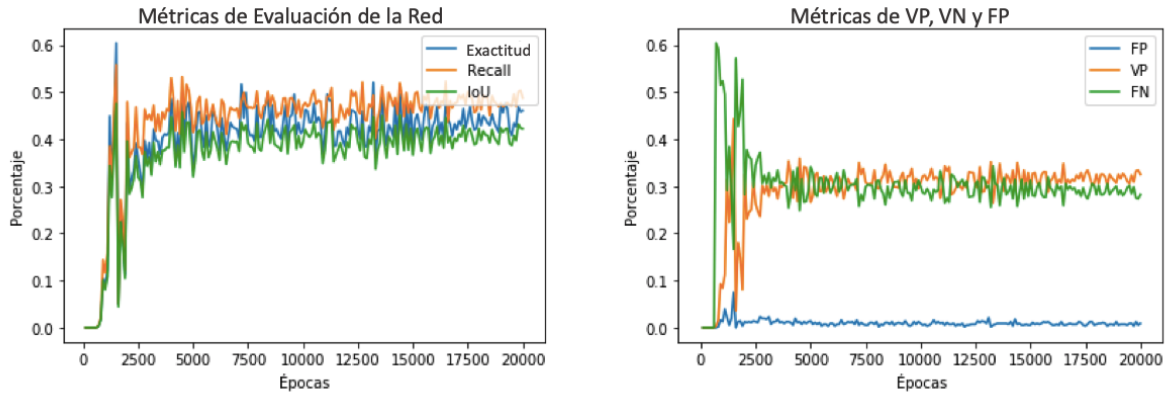

Figura 3.8: Red 1: Métricas de evaluación de rendimiento (izq) y variación de VP, FP y FN (der).

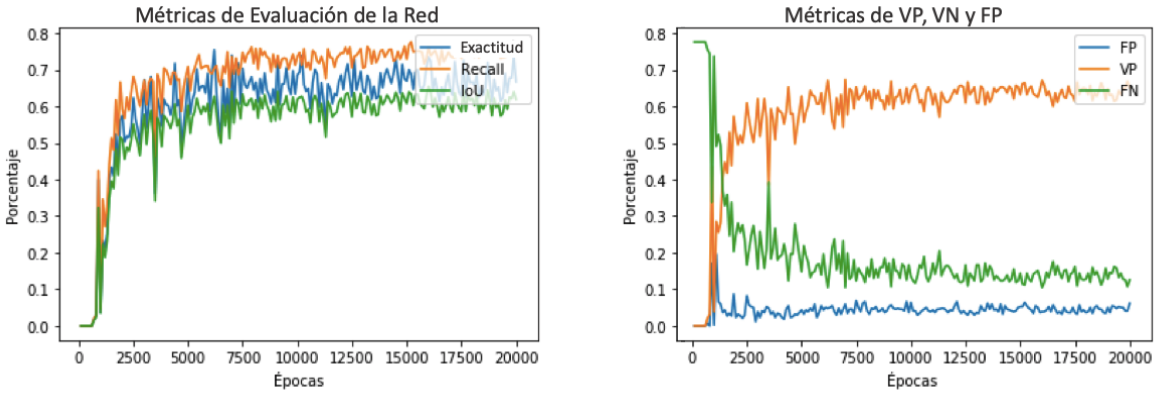

Figura 3.9: Red 2: Métricas de evaluación de rendimiento (izq) y variación de VP, FP y FN (der).

Para ambas redes, en la gráfica de Métricas de evaluación, podemos observar que en las primeras cinco mil épocas las curvas se ajustan en mayor grado que en las épocas siguientes. Todos los máximos se alcanzan antes de las primeras 16.000 épocas de entrenamiento. Para la Red_1 el máximo de Exactitud y Recall se obtiene en la misma época de entrenamiento, a las 1.500 épocas. Para la Red_2 ambos máximos se presentan en épocas diferentes, el máximo de Exactitud en la época 6200 y el máximo de Recall en la época 15.200.

Resulta útil enmarcar los valores de Exactitud y Recall según los objetivos planteados. Los dos principales objetivos para este proyecto son: conteo de troncos sobre una carga y la obtención de diámetros sobre la carga. Si el objetivo es determinar la cantidad de caras en una imagen, buscamos una mayor exactitud. Un mayor Recall permitirá obtener recuadros

mejor ajustados a la cara real, por lo que para calcular diámetro utilizaremos la red con mayor Recall.

	Épocas	Exact.	Recall	IoU	FP	VP	FN	E_2	R_2
Red_1	1500	0.60	0.56	0.48	42	248 (72.9 %)	92 (27.1 %)	0.86	0.73
Red_2	6200	0.75	0.70	0.63	21	294 (86.5 %)	46 (13.5 %)	0.93	0.86
Red_2	12600	0.73	0.74	0.64	13	292 (85.9 %)	48 (14.1 %)	0.96	0.86
Red_2	15200	0.69	0.78	0.63	28	291 (85.6 %)	49 (14.4 %)	0.91	0.86
Red_2	16000	0.73	0.76	0.65	17	294 (86.5 %)	46 (13.5 %)	0.95	0.86

Tabla 3.3: Resumen métricas de evaluación de la red

En la gráfica de Variación de VP, FP y FN se aprecia mejor la convergencia de la red. Si bien la Red_1 presenta valores altos de las métricas para una época temprana del entrenamiento, se observa que la cantidad de verdaderos positivos (curva naranja) es muy cercana a la cantidad de falsos negativos (curva verde) en las épocas posteriores. En la Red_2 se observa más claramente la existencia de una separación entre verdaderos positivos y falsos negativos. Las detecciones convergen a cerca de 300 caras detectadas correctamente de un total de 405 caras evaluadas.

En la tabla 3.3 se muestra las épocas en las que alguna de las métricas alcanza su máximo, utilizando el conjunto de imágenes de prueba. Las métricas de Exactitud y Recall consideran la definición en base a imágenes. Se agrega E_2 y R_2 que consideran la definición de Exactitud y Recall en base la cantidad de detecciones correctas. La mejor Red para realizar el conteo de caras corresponde a la Red_2 en la época de entrenamiento 12.600, con E_2 (exactitud) de 0,96. Para el cálculo de diámetro la red recomendada es la Red_2 en la época 15.200, con Recall de 0,78.

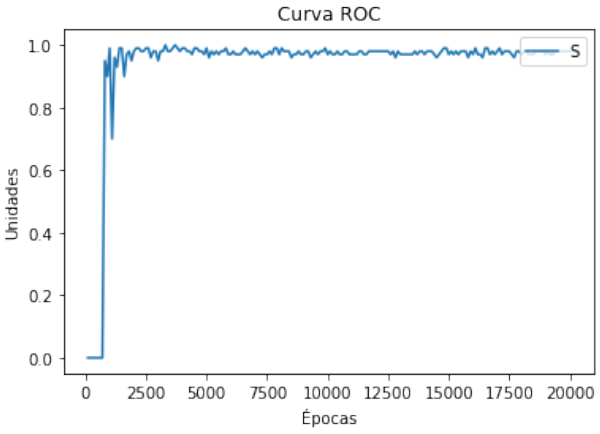

Figura 3.10: Curva ROC de la Red_2 sobre el conjunto de prueba

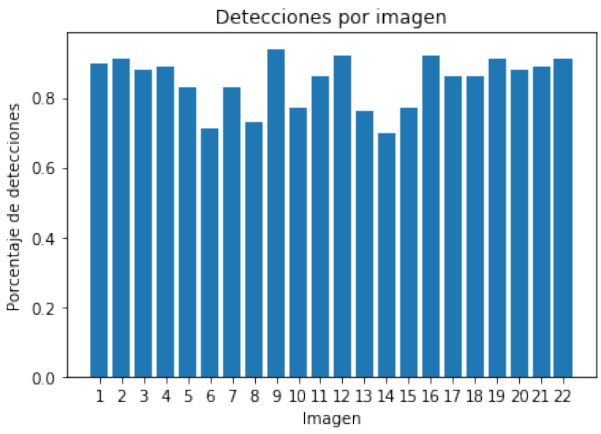

Figura 3.11: Gráfico de detecciones por imagen sobre el conjunto de validación

Como ya se menciono el espacio ROC representa los intercambios entre verdaderos positivos y falsos positivos. El mejor método de predicción estará situado en la esquina superior izquierda del espacio ROC, y recibe el nombre de clasificación perfecta. Una clasificación aleatoria tomaría puntos sobre la diagonal (esquina inferior izquierda a la superior derecha).

En la imagen 3.10 se muestra la curva ROC de la Red_2, se observa que la posición de la curba esta por sobre una clasificación aleatoria y cercana a la clasificación perfecta.

Los resultados de la detección del área de interés para las imágenes del conjunto de validación se presentan en la Figura 3.11. Se muestra en el eje de las ordenadas el nombre de cada imagen y en el eje de las abscisas el porcentaje de las caras detectadas para la imagen. Se obtiene una media aritmética de 0.85, mediana de 0.86, desviación estándar de 0.07 y varianza de 0.005. La imagen para la que se tienen mejor resultado es para la imagen 9 con un Recall de 0.94. La imagen con el desempeño más bajo es la número 14 con un recall de 0.7.

Se realizó un entrenamiento adicional con imágenes que contenían sólo la sección de la carga. Para generar el conjunto de datos se reajustaron las dimensiones de las imágenes utilizando la categoría 'carga' para definir la posición de la carga dentro de la imagen. El entrenamiento realizado con estas imágenes no entregó las métricas satisfactorias, al aumentar las épocas de entrenamiento el resultado no mejora significativamente, por lo que se descartó obtener la zona de interés previo al entrenamiento de la Red Neuronal.

Figura 3.12: Ejemplo de detecciones, para la red seleccionada, con mejor Recall

3.3.2. Análisis de casos

La red seleccionada presenta un desempeño de 0,86 de recall. Para algunas imágenes es posible obtener un valor por encima del estimado para la red. Dentro del conjunto de datos de validación, las imágenes para las que se obtiene mejor desempeño se presentan en la figura 3.12. Estas imágenes presentan valores de Recall superiores a 0,9. Las imágenes mostradas presentan buenas detecciones a pesar de la cantidad de troncos. Las fotografías muestran que pequeñas porciones de la carga son tapadas por la estructura del camión.

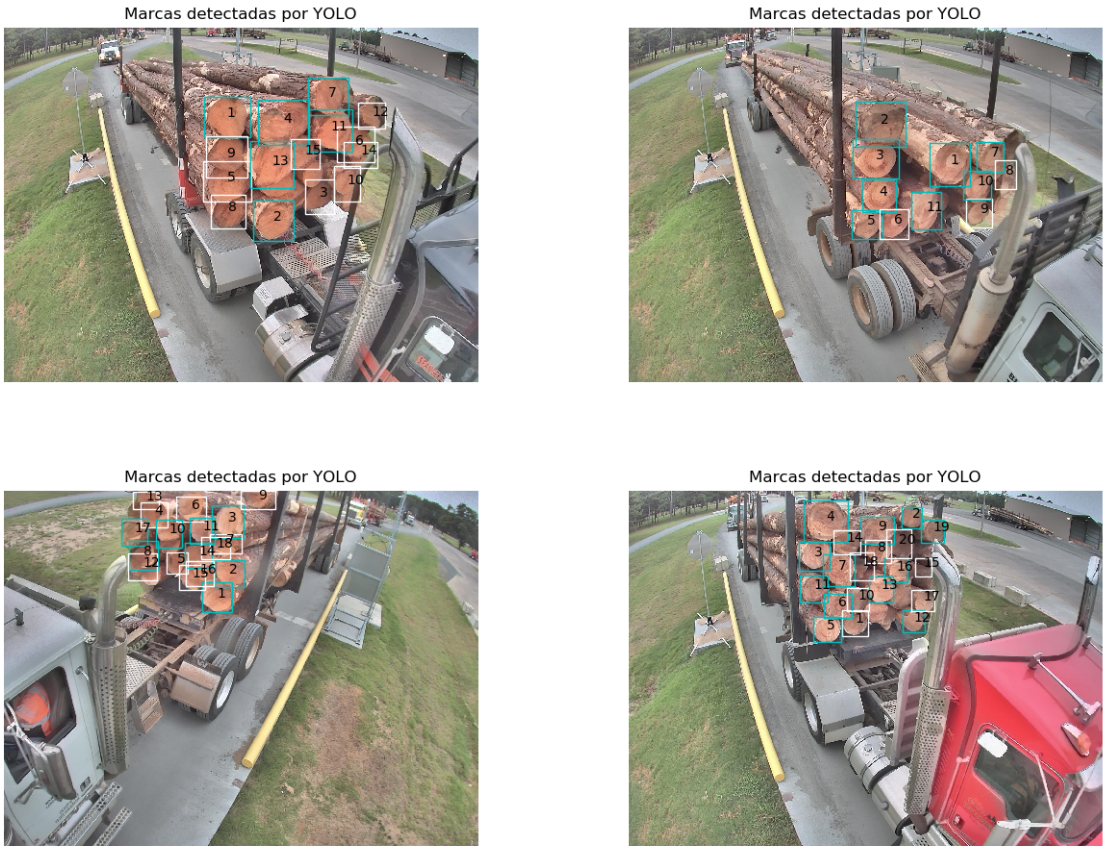

Figura 3.13: Ejemplo de detecciones, para la red seleccionada, con menor Recall

De las imágenes de las que se obtienen resultados menos favorables para el mismo entrenamiento, se muestran ejemplos en la figura 3.13. Los valores de recall de estas imágenes son cercanos a 0,7. De las imágenes mostradas es posible notar que como factores que influyen en la detección esta el orden de la carga y el bajo contraste entre las caras. La caras de tronco con menor contraste corresponden, a aquella que no tienen corteza, o que la parte de la corteza esta ocluida por otra cara de tronco, dificultando el conteo.

Las principales dificultades de detección corresponden a caras parciales cuya imagen originalmente marcada es tan pequeña que no es posible efectuar la detección de manera exitosa. Cuando parte del camión tapa la carga o cuando la imagen está lo suficientemente inclinada respecto al plano de la cámara no es posible ver las caras más alejadas de manera clara. A

pesar de lo anterior, la red es capaz de detectar caras incluso en casos difíciles de observar a simple vista (se muestran algunos ejemplos en la figura 3.14).

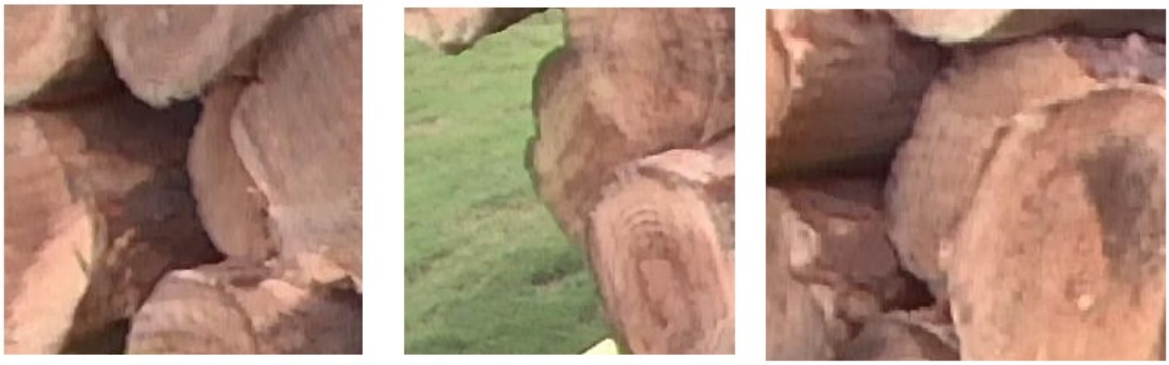

Figura 3.14: Imágenes con detección exitosa y que están parcialmente ocluidas dentro de la imagen.

Analizaremos, además, algunos casos de interés para la ejecución de la red. Como ya se vio, la red presenta buenos resultados para imágenes pertenecientes al conjunto de validación, que corresponden a imágenes con las mismas características de las imágenes de entrenamiento. Se muestran en la imagen 3.15 algunos ejemplos para imágenes con caras de troncos, pero con características diferentes a las ya analizadas. Las principales variaciones son el diámetro de la carga, el contraste de luces y el fondo.

Figura 3.15: Ejemplo de detecciones, para la red seleccionada, con imágenes fuera de la base de datos

En la imagen se observan tres falsos positivos en la parte superior. Las detecciones correctas, verdaderos positivos, corresponden a 33 caras de un total de 44. Si obtenemos las métricas para esta imagen, se obtiene un 70 % de exactitud y un 68 % de recall. Para el cálculo realizado, al igual que en las métricas anteriores, se considera la distinción por clases.

3.3.3. Preprocesamiento de imágenes

La etapa de preprocesamiento permite mejorar y ajustar los datos para la segmentación. El primer preprocesamiento realizado sobre la imagen corresponde a la detección de la zona de interés, utilizando la Red Neuronal YOLO, descritas en el apartado anterior (capítulo 3.3.1). Esta segunda etapa considera un preprocesamiento adicional sobre las imágenes del conjunto de Prueba y validación que se detectaron por la red como pertenecientes a la categoría Full. El nuevo preprocesamiento esta enfocado al realce de la imagen y al manejo de ruido.

Para la evaluación de ecualización de histograma, en la imagen 3.16 de izquierda a derecha se presenta la imagen inicial y la imagen ecualizada con sus histogramas correspondientes. Se observa que es posible ajustar los rangos de intensidad dentro de la imagen, partiendo de un histograma inicialmente centrado. No se obtiene un histograma bimodal y, por lo tanto, no se obtiene una segmentación clara sólo con la aplicación de esta técnica.

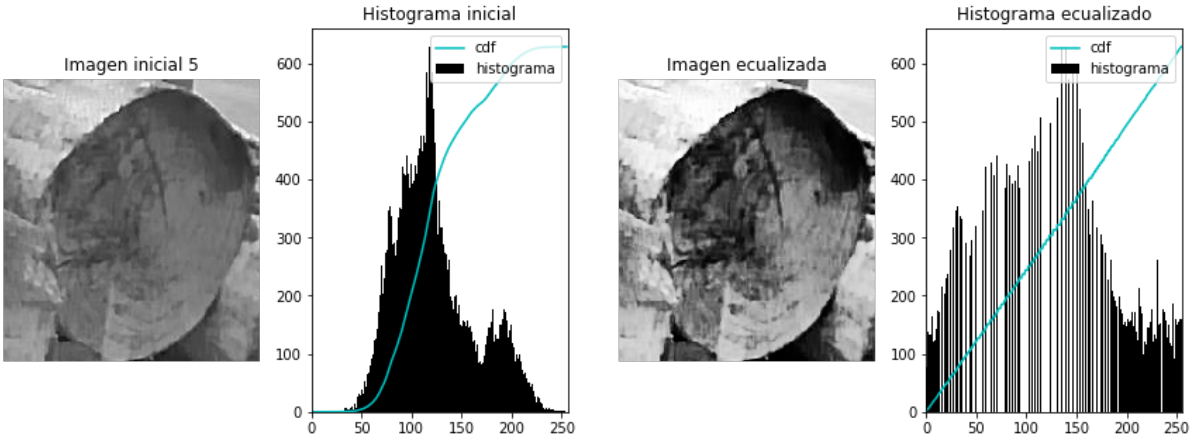

Figura 3.16: Ecualización de histograma para un ejemplo de cara detectado por la red

3.3.4. Segmentación de imágenes

Recordemos que en esta etapa, los ajustes específicos para cada prueba deben ser incorporados manualmente. Se muestra a continuación el detector de bordes Sobel. En la figura 3.17 se observa a la izquierda el resultado del detector de bordes Sobel para x e y, sobre una imagen inicial en escala de grises. A la derecha, se muestra el resultado de aplicar previamente un filtro gausiano. Se observa un mayor realce de bordes al utilizar filtro gaussiano previamente.

Para el detector de bordes Canny se consideraron diferentes valores de límite superior e inferior. Se muestran los resultados obtenidos en la figura 3.18, de izquierda a derecha se muestran los resultados de configurar V_{min} igual a V_{max} igual a 100, 200, 300 y 400 respectivamente. Podemos observar que para menores límites superior e inferior la imagen contiene más información de contorno. Para límites mayores se logra eliminar mas ruido y se mantiene claro el contorno del tronco.

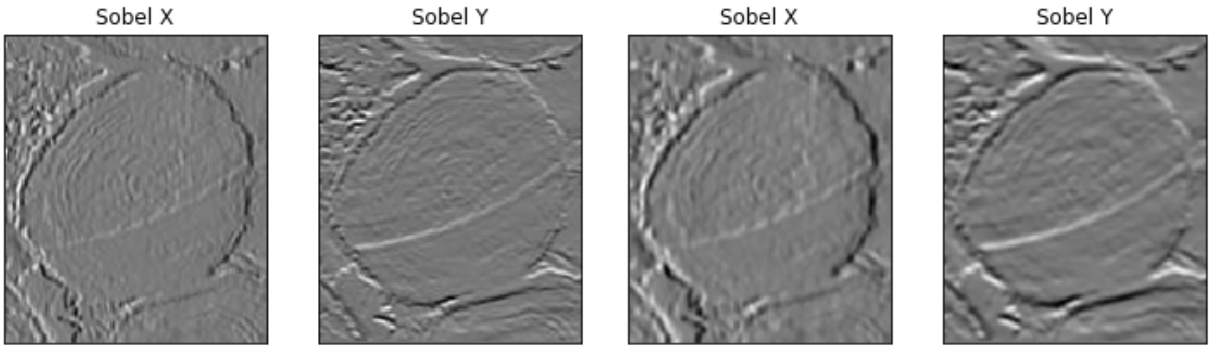

Figura 3.17: Detector de bordes Sobel x, y: Imagen inicial en escala de grises (izq) y con filtro gaussiano (der), para imágenes de entrada preprocesadas con un filtro Gaussiano.

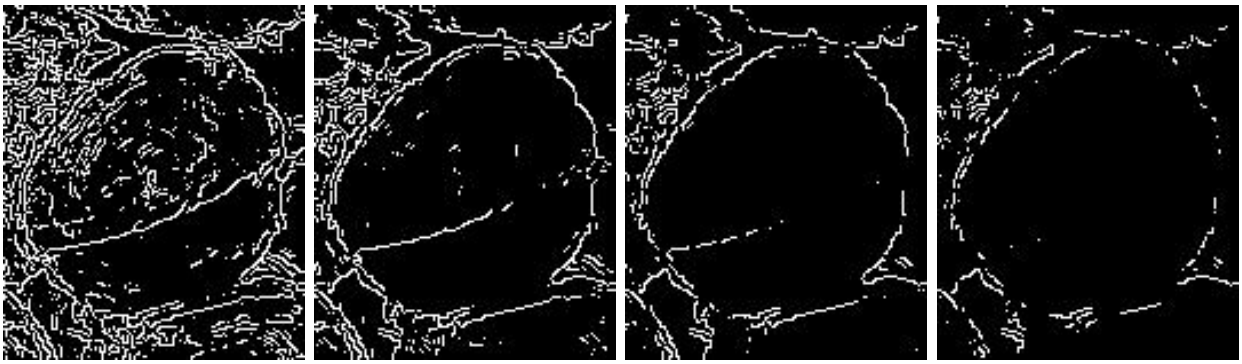

Figura 3.18: Detector de bordes Canny: Para limite inferior y superior igual a 100, 200, 300 y 400 (izq a der)

Figura 3.19: Detector de bordes Canny posterior a filtro gaussiano: Para limite inferior y superior igual a 100, 200, 300 y 400 (izq a der)

Si se aplica Canny posterior a un filtro gaussiano de 3x3, se observan bordes más claros desde límites menores (imagen 3.19). Para definir el valor de umbral mínimo y máximo del filtro Canny que mejor demarcan el contorno, es necesario evaluar los resultados posterior al preprocesamiento. En la imagen 3.20 de izquierda a derecha, se muestra un ejemplo partiendo con un filtro gaussiano, detector de bordes Canny, los contornos sobre la imagen y el resultado de buscar contornos cerrados. El borde encontrado no corresponde a la cara.

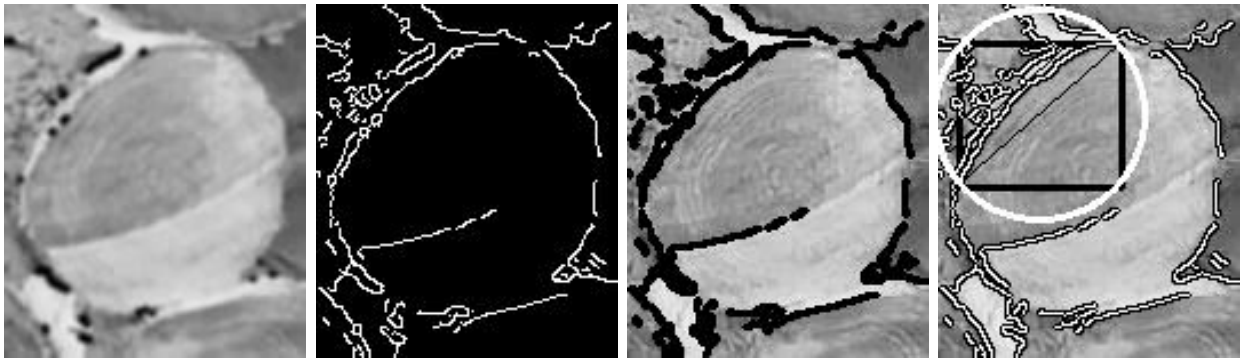

Figura 3.20: Detección de contornos cerrados aplicando filtro gaussiano y Canny

Modelos deformables: Contornos Activos

Con el modelo de contorno activo evaluaremos la segmentación de la cara de un tronco a través de una curva cerrada. Considerando que la imágenes de entrada corresponden a caras de troncos bien delimitadas, se utilizan las características del recuadro como parámetros del prior de contorno activo. El radio de la circunferencia inicial tomará el valor del lado mayor de la región de interés. En caso de que el resultado de aplicar Hough sobre la imagen entrege un resultado satisfactorio, se utilizará éste valor como prior del algoritmo de contorno activo.

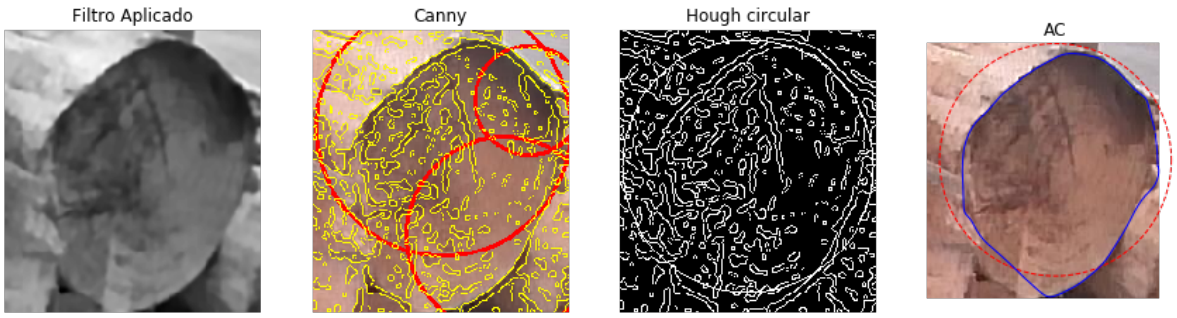

Figura 3.21: Diferentes algoritmos de segmentación sobre una imagen; Filtro aplicado, Canny, Hough y Contornos Activos

El prior se modifica por cada iteración del algoritmo hasta minimizar el funcional de energía sobre la imagen. Al realizar una inspección visual de los resultados, la curva se ajusta en gran medida al contorno de la cara. Para obtener una comparación entre los algoritmos vistos, en la imagen 3.21, se muestra la imagen filtrada (izq). Posteriormente el resultado de aplicar Canny para realzar los bordes en amarillo y los contornos cerrados encontrados en rojo. El resultado de Hough y el método de contornos activos (der) con la linea roja que representa el prior y en azul el contorno encontrado.

Como se observa en la figura anterior, la imagen pasa por un proceso de filtrado para realzar los contornos y facilitar la detección entes de la segmentación. Realizando una inspección visual de los diferentes algoritmos de segmentación utilizados (Canny, Hough y Contornos Activos) se comprobó que en todos los casos la detección del contorno fue más cercana al

contorno real al utilizar contornos activos.

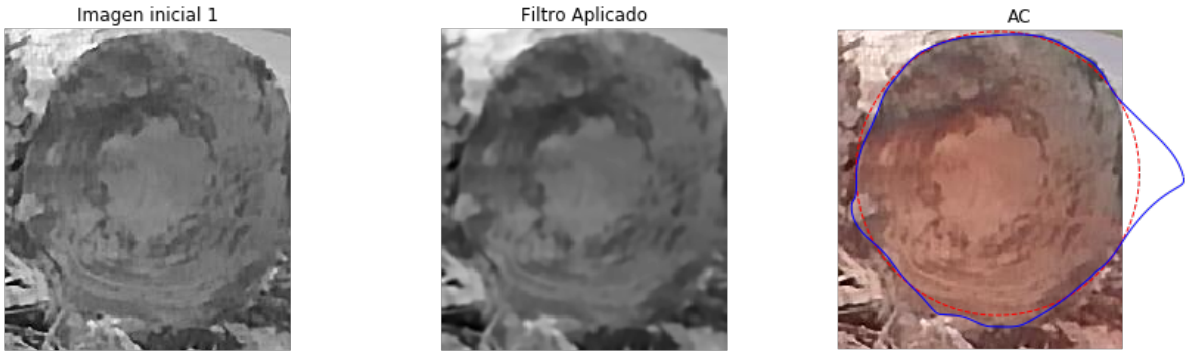

Figura 3.22: Ejemplo AC: Imagen inicial (izq), filtrada (centro) y contornos activos (der)
.

Figura 3.23: Ejemplo AC: Imagen inicial (izq), filtrada (centro) y contornos activos (der)

Al determinar el borde fino utilizando AC para cada recuadro, se pierde información del resto de la imagen. Los bordes pueden ampliarse por la zona blanca en lugar de detenerse al encontrar otros máximos, ejemplo figura 3.22. Para evitar el desborde del contorno, se utilizo AC sobre la imagen completa. La imagen 3.23 permite visualizar un ejemplo de segmentación exitosa utilizando AC. Se muestra el resultado de Houg (der) utilizando los parámetros del recuadro de entrada (en rojo) y los resultado de AC (azul).

Los resultados sobre contornos activos se comparan con los contornos de la base de datos llamada 'DS_contornos'. La base de datos contiene imágenes, del mismo tamaño que la original, para cada cara de tronco. La imagen tiene fondo blanco y en ella sólo se ve el contorno fino de cada cara en color negro. Para comparar los resultados de contornos activos con los datos manuales se aplican las métricas descritas en la sección 2.2.2, partiendo por el diámetro promedio.

Configuraciones

En la sección anterior se evaluó el preprocesamiento de imágenes utilizando filtros por separado. Ahora corresponde evaluar el desempeño de las configuraciones de filtros seleccio-

nadas (descritas en la sección 2.1.2). Para evaluar su desempeño se ejecutó el algoritmo de segmentación, contornos activos, para imágenes preprocesadas con las cinco configuraciones de filtros. Se evaluó el diámetro promedio de las caras de troncos y se comparo con el diámetro promedio utilizando DS_contornos.

La comparación realizada permitió obtener el sesgo de la estimación del borde fino y la desviación estándar de esa diferencia. La evaluación se utilizó sobre las 198 caras del conjunto de imágenes de prueba que fueron detectadas por la red como pertenecientes a la categoría Full. Para las caras evaluadas se tiene que el radio promedio de las marcas manuales de $79{,}5 + -0{,}3$ y un radio promedio de las estimaciones de la red de $86{,}03 + -0{,}28$. Ambos valores no varían para los filtros evaluados.

Filtro	radio estimado	promedio del error	desviación estándar
Filtro 1	84.62	5.12	5.80
Filtro 2	83.43	3.93	4.75
Filtro 3	84.73	5.23	5.46
Filtro 4	82.70	3.2	4.65
Filtro 5	82.99	3.48	4.69

Tabla 3.4: Resultados de contornos activos para diferentes configuraciones de preprocesamiento

El radio estimado entregado en la tabla corresponde al radio promedio. Para obtenerlos se calcula por separado el radio de cada cara detectada del conjunto de imágenes de prueba y se promedian. La primera columna corresponde a la combinación de filtros aplicados, la segunda al radio promedio estimado utilizando contornos activos. La tercera columna entrega el promedio del error y la cuarta la desviación estándar de ese error.

El promedio del error, utilizando las estimaciones de la red, es de 6.24 y desviación estándar de 5.82. Para los valores de promedio del error y desviación estándar de AC, mostrados en la tabla 3.4, se observa una disminución en el promedio del error para todos los casos, en comparación al promedio del error de la red. El promedio del error menor es de 3.2 y el mayor de 5.23.

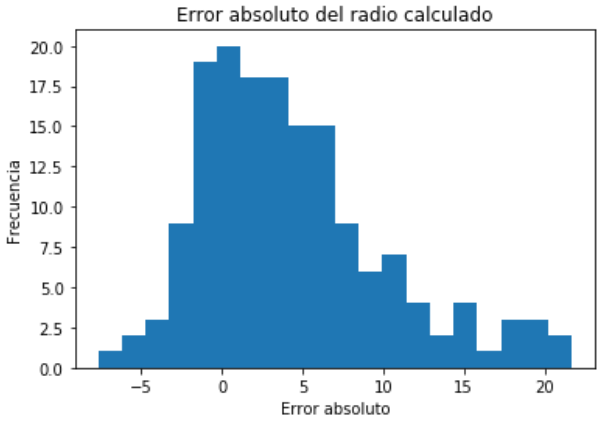

Figura 3.24: Detecciones de YOLO (izq) y Filtro uno (der) para imagen de ejemplo

El mejor resultado de filtro, sobre los datos de prueba corresponden al filtro cuatro. Se

obtiene un radio promedio estimado de 82.7, con promedio del error de 3.2 y desviación estándar de 4.65. El gráfico 3.24 muestra el histograma del promedio del error obtenido para todas las caras del conjunto de prueba. Se utilizó el Filtro cuatro previo a la segmentación. El 75 % de los datos presenta error positivo, debido a que el filtro de contornos activos esta configurado para iniciar la búsqueda del contorno fino desde afuera hacia adentro en la imagen.

Filtro	Radio estimado	Tiempo Red	Tiempo Contorno	Tiempo Total
Filtro 1	80.25	0.713	119.827	120.489
Filtro 2	80.29	0.626	117.234	117.809
Filtro 3	79.71	1.071	115.85	116.870
Filtro 4	79.52	0.843	94.573	95.363
Filtro 5	79.89	0.675	104.656	105.278

Tabla 3.5: Resultados de diferentes configuraciones para la imagen de ejemplo

Probaremos las diferencias de los filtros sobre una imagen. Utilizamos la imagen 3.25 (izq), con r_marca de 79.19 píxeles. Se muestran en la tabla 3.5 el radio promedio utilizando el filtro indicado en cada fila y contornos activos. El tiempo de la red, corresponde a la detección de las caras y el tiempo de contorno a la segmentación de contorno fino. El tiempo total corresponde a la ejecución del modo on line.

En la imagen 3.25 (der) se observa el resultado de la ejecución del modo on line utilizando el filtro uno. En la imagen 3.26 se muestra la ejecución del Filtro dos (izq) y Filtro tres (der). La imagen 3.27 muestra los resultados del Filtro cuatro (izq) y cinco (der).

Figura 3.25: Detecciones de YOLO (izq) y Filtro uno (der) para imagen de ejemplo

3.4. Integración modelo on line

Los pasos para integrar en un modelo completo se dividen en dos etapas. La primera tiene como objetivo obtener los parámetros de las caras detectadas por la red. La evaluación de la red considera la obtención de los valores correspondientes a cada detección. La segunda etapa

Figura 3.26: Filtro dos (izq) y Filtro tres (der) para imagen de ejemplo

Figura 3.27: Filtro tres (izq) y Filtro cuatro (der) para imagen de ejemplo

corresponde al procesamiento posterior a la red. Éste procesamiento tiene como objetivo obtener métricas del funcionamiento de la red, comparar la información con los datos de marcaje y obtener el ajuste fino o segunda segmentación.

Primero analizaremos el comportamiento para una imagen. El número de imagen seleccionada es la imagen número 10 del evento 27990, obtenida por la tercera cámara. Los resultados de YOLO se obtuvieron para la época de entrenamiento 15200 de la Red número 3. Para esta imagen, la precisión de los recuadros detectados es de 86 %, la precisión en la cantidad de caras detectadas correctamente es de 100 %. El valor de Recall de la detección de la red es de 90 % y la precisión en la cantidad de caras detectadas es de 100 %.

En la imagen 3.28 a la izquierda se muestran los datos de marcaje manual. A la derecha se muestran los recuadros detectados por la Red 2. Para ambas imágenes los recuadros de Blancos corresponden a datos de la categoría 'Partial' y los de color Cian a datos de la categoría 'Full'. El número corresponde a un rotulo utilizado para diferenciar cada cara. Para cada recuadro entregado por la red con categoría Full, se busca el contorno especifico de la cara. Para la imagen seleccionada, el número de caras Full detectados es mayor que los

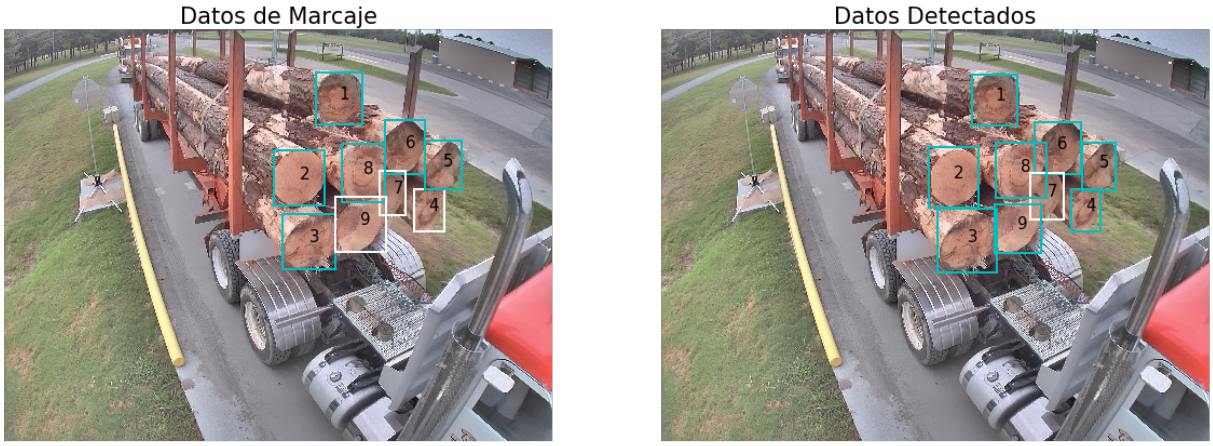

Figura 3.28: Datos de marcaje (izq) y detectados (der) para la imagen seleccionada

Figura 3.29: Adivinanza inicial (izq) y ajuste de contornos activos (der) para la imagen seleccionada

marcados.

Para obtener el contorno fino de cada cara se parte por generar una adivinanza inicial para el algoritmo de contornos activos. En la imagen 3.29 se muestra a la izquierda la adivinanza inicial. A la derecha se muestra el ajuste del contorno realizado por contornos activos.

3.5. Tiempo de procesamiento

En esta sección se describe el tiempo requerido para la ejecución del algoritmo completo, se muestran las etapas en la figura 3.30. Para esto debemos aclarar que existen etapas que requieren de una sola ejecución como el marcaje de datos y el entrenamiento de la red. Adicionalmente, también existen etapas que requerirán ser ejecutadas por cada nueva imagen que se desea procesar como la detección de caras, el ajuste de contorno fino y cálculo de número de caras y diámetro.

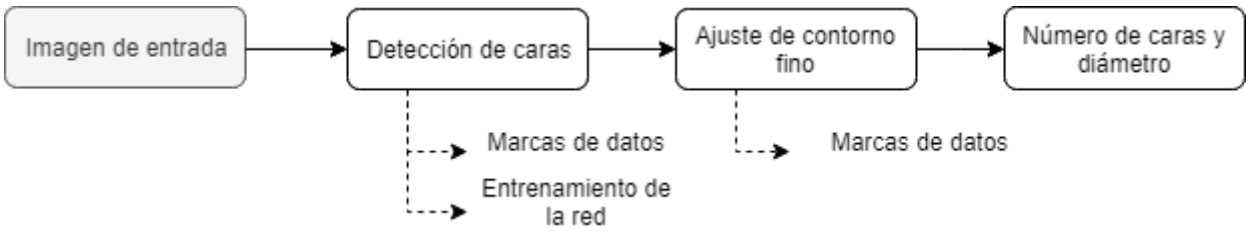

Figura 3.30: Etapas de ejecución de algoritmo

Si se modifica conjunto de datos de entrenamiento, se debe realizar nuevamente la etapa de marcaje. Para el desarrollo de este trabajo se marcaron 236 imágenes, en total 4.502 caras de troncos. Se utilizó OpenLabeling y un tiempo estimado en nueve horas, para los datos de la red. Se utilizó sketch para marcar los contornos en un total de nueve horas. Se sugiere modificar el conjunto de datos de entrenamiento en caso de utilizar imágenes provenientes de plantas diferentes, para minimizar los efectos de posición o cambio de luz de las cámaras, también es necesario si se requiere probar el entrenamiento utilizando nuevas categorías.

La creación de nuevas bases de datos para entrenamiento, podría tomar menor tiempo del mencionado si se utiliza como punto de partida las detecciones de la red ya entrenada. Modificando solo las imágenes mal detectadas. Como la red ya tiene una exactitud de 0.86, en el mejor de los casos solo sería necesario disponer de 70 minutos.

La etapa que requiere de mayor tiempo es el entrenamiento de la red. El tiempo de entrenamiento dependerá de las prestaciones y características del hardware utilizado (sección 2.3), el tamaño del set de entrenamiento (sección 3.1), la configuración de la red y varios otros factores adicionales. El tiempo utilizado para las características de red ya descritas en el desarrollo de este trabajo están en el orden de horas de entrenamiento (4-8 horas) hasta días de entrenamiento (hasta cuatro días).

Cuando ya se fija una configuración de red y se entrena, es necesario determinar en que época del entrenamiento se obtiene el mejor rendimiento. Para esta etapa las características de hardware se describen en 2.3. Para el conjunto de validación de 22 imágenes con 403 caras en total, se obtienen los resultados de la red ya entrenada en menos de un segundo en promedio por imagen. Los datos de la red son comparados con los de marcaje para obtener las métricas de evaluación y tarda en promedio 7.75 minutos.

Finalmente la implementación del modo online considera la obtención de los datos de YOLO, a partir de una red ya entrenada, y la obtención de diámetros con AC de todas las caras detectadas clasificadas en la categoría Full. El promedio de tiempo empleado en la detección de la red es menor a un segundos por imagen. El tiempo de la ejecución de AC sobre las detecciones realizadas por la red dependerá del número de detecciones y de las iteraciones máximas que se permite. Para un máximo de mil iteraciones se demora por cada imagen menos de un minuto.

Conclusión

En el diseño, implementación y evaluación de modelos matemáticos para estimar diámetro y número de troncos sobre bancos de madera cargados sobre camión, mediante imágenes se trataron dos aspectos importantes: el primer aspecto consistió en obtener el número de caras en la imagen, el segundo aspecto importante fue el ajuste de contorno de las caras detectadas y calcular su diámetro.

El primer aspecto consistió en obtener el número de caras en la imagen, realizando segmentación y localización con redes neuronales convolucionales. Se utilizaron varias configuraciones de YOLO y se completaron pruebas durante el entrenamiento para determinar la arquitectura que proporcione los mejores resultados.

El segundo aspecto partió con el ajuste de contorno de las caras detectadas para obtener el radio. Éste aspecto consistió en evaluar e implementar una etapa de procesamiento de imágenes. Se ajustó el contorno de cada cara de tronco detectada y se determinó el diámetro. Se implementaron pruebas de diferentes configuraciones de filtros y algoritmos de segmentación para obtener una solución más exacta.

En relación a los resultados obtenidos se presentan las siguientes conclusiones:

- Arquitecturas más livianas de redes neuronales convolucionales requieren de menor tiempo y recursos computacionales para generar resultados con pérdidas de error medio aceptables. (se considera como criterio de detención del entrenamiento que la pérdida de error medio sea menor a uno)
- Arquitecturas más complejas y con mayor número de capas requieren de mayor tiempo de entrenamiento. El tiempo de entrenamiento de una red liviana es menor al 5 % del tiempo utilizado por las otras arquitecturas.
- La elección de la mejor red a utilizar dependerá del objetivo. Si la implementación se centra en el conteo de caras, se utiliza la con mayor Exactitud. Por otro lado, si se centra en la obtención de diámetro, se utiliza la con mejor Recall.
- La tercera versión de YOLO mejora el total de pérdida y el promedio de pérdida de error para todas sus configuraciones. Además, disminuye el tiempo de procesamiento por cada imagen.
- En relación al hardware utilizado para el entrenamiento, la utilización de CUDA permitió obtener resultados en menor tiempo. Para tareas de detección, es posible realizar la detección sin CUDA en contraposición a un mayor tiempo de detección.
- Para la etapa de entrenamiento es posible reducir los tiempos utilizando redes ya en-

trenadas. Los datos de marcas manuales para el entrenamiento de la red seleccionada son rápidos de obtener, agilizando el proceso.

- El entrenamiento y evaluación de yolov2-tiny utilizando dos clases ('Full' y 'Partial') permitió la obtención de mejores resultados en Exactitud, Recall e IoU, en comparación a utilizar sólo una categoría.

Para la detección de caras de troncos en bancos de madera se concluye que utilizar técnicas de Deep Learning, en particular Redes Neuronales Convolucionales, permite obtener resultados satisfactorios. La mejor red implementada considera valores de Recall de 0.84 y Exactitud de 0.97 para el conjunto de validación. La red seleccionada permite tener tiempos de detección cercanos a 0.5 segundos, congruentes con un modelo de ejecución online. Los tiempos de entrenamiento son aún elevados, sin embargo, ésta etapa sólo es necesaria para agregar las nuevas clases a la red.

En la etapa de segmentación, las primeras pruebas muestran que las dificultades en esta etapa se originan principalmente en: el bajo contraste de algunas zonas del borde de las caras de troncos por falta de corteza, la textura de cada cara, la superposición de troncos, problema de enfoque y las deformaciones angulares. Se implementaron diferentes configuraciones de filtros y de algoritmos de segmentación para determinar el modelo que mejor responda a éstas dificultades. En relación a los resultados del ajuste de contorno se concluye lo siguiente:

- El procesamiento de imágenes requiere de mayor ajuste de parámetros de manera manual en comparación al uso de redes neuronales. Los parámetros se deben revisar en caso de cambiar la base de datos.
- La configuración de filtros que permite obtener un menor promedio del error en la detección de diámetro es el que combina transformación a escala de grises, filtro gaussiano, umbral y mediana.
- Evaluar los diámetros de la carga, utilizando como radio el promedio entre el ancho y el largo de la red, tiene un promedio del error de 6.24 con desviación estándar del error de 5.82. Es posible obtenerlo con los mismos resultados de la red.
- Utilizando contornos activos es posible minimizar el error, en comparación al error obtenido con los datos de la red. Utilizando contornos activos se obtiene un promedio del error de 3.2 para la mejor implementación, con desviación estándar de 4.65.
- La aplicación de contornos activos sobre las caras detectadas por la red toma, en promedio, 59 segundos por carga.

Las curvas de contorno se ajustan a la cara de tronco a través de curvas deformables que se acercan al contorno real minimizando el funcional de energía. Los resultados de aplicar AC, combinadas con el procesamiento de imágenes seleccionado (escala de grises, filtro gaussiano, umbral y mediana) minimiza el promedio del error (de 3.2) en comparación a las otras configuraciones. Se acerca más al diámetro marcado que los obtenidos con Sobel, Canny y Hough. En el proceso completo, ésta etapa es la que requiere de mayor tiempo de ejecución, en promedio toma 59 segundos por carga.

Las etapas de desarrollo consideran una primera detección de las regiones de interés y una segunda etapa de segmentación. En relación a los datos de marcaje y considerando que el etiquetado de imágenes para clasificación es mucho más costoso que el etiquetado de imágenes

para localización y que las marcas de contorno fino sobre la imagen son más costosas que las de localización se concluye que las etapas propuestas responden a facilitar el proceso de marcaje, considerando la posibilidad de replicar su uso para otra plantas.

Los sistemas de visión artificial orientados a permanecer en el futuro, consideran implementaciones con posibilidad de adaptar los resultados a tareas cambiantes. En este contexto, se optimizó la etapa de marcaje, se entrenó en base a dos categorías que permiten mejorar la medición de diámetro y se consideraron los resultados de la etapa de clasificación para inicializar los parámetros de la etapa de segmentación. Con lo anterior, se minimizan los parámetros que requieran de ajustes al modificar las características de la imagen o la base de datos.

Se logro evaluar el impacto de diferentes algoritmos de preprocesamiento en la estimación del radio de las caras de troncos en la imágenes, en base a la desviación estándar de la diferencia entre la marca manual y la estimada, teniendo presente que la detección de diámetro es una de las etapas más importantes en la caracterización de la carga.

Dentro de los aprendizajes adquiridos en el proceso de desarrollo del presente trabajo está el uso de diferentes herramientas. Scrum, corresponde a un marco de trabajo para desarrollo de proyectos de software. Permite priorizar los módulos que aportan mayor valor a la empresa, con exposición de avances de manera regular. Gitlab es un servicio web de control de versiones y desarrollo de software colaborativo basado en Git.

3.6. Trabajos futuros

Evaluar la extensión de la base de datos a imágenes de diferentes plantas permitiría tener una red robusta y universal, capas de detectar exitosamente en diferentes contextos, intensidades de luz y ángulos de captura. Permitiría, además, que las mejoras del algoritmo fueran rápidamente extensibles a todas las plantas.

La solución implementada considera una red rápida en desmedro de mayor certeza en las detecciones. Cuando el tiempo de entrenamiento no sea un factor crítico, como es el caso de un procesamiento posterior de los datos, se sugiere seleccionar una red con menor pérdida de error promedio.

Hasta el momento el valor de radio se determina para las detecciones pertenecientes a la categoría Full. Para la categoría Partial es necesario realizar una estimación de las caras, considerando las zonas parcialmente cubiertas, para obtener el radio. Esta etapa finalizaría el análisis de las categorías detectadas.

Una siguiente etapa del algoritmo sugiere estimaciones sobre las zonas no detectadas considerando las características de la carga del camión y las zonas donde exista alta probabilidad de encontrar una cara pero no exista una detección asociada.

El trabajo realizado se enmarca en el contexto de desarrollo de productos de bajo costo de Woddtech. Para este desarrollo se considera la implementación en paralelo, actualmente

en curso, del uso de imágenes de estereoscopio para estimación de distancias. Adicional a la exploración ya realizada con algoritmos de stitching.

La continuidad de este proyecto incluye la detección de diámetro de caras en unidades de medidas de un sistema de medidas definido. Las pruebas correspondientes a esta etapa se realizaran posterior a la finalización de los dos proyectos mencionados.

Bibliografía

[1] G. J. Augustine, D. Purves, D. Fitzpatrick, W. C. Hall, A. S. Lamantia, J. O. McNamara, and S. M. Williams. Neurociencia. In *Neuroscience, Third edition*, pages 249–308. Editorial médica panamericana S.A, 2010.

[2] A. Baldini. Por un chile forestal sustentable. Technical report, CONAF, Avenida Bulnes 285, Santiago de Chile, Agosto 2013.

[3] V. Caselles and A. Frangi. La segmentación de imágenes. el método de los contornos activos geométricos. In *Encuentros Multidisciplinares*, page 12, 2006.

[4] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contour. In *International Journal of Computer Vision 22*, pages 61–79, 1997.

[5] E. R. Devies. *Computer and Machine Vision: Theory, Algorithms, Practicalities*. Elsevier, fourth edition edition, 2012.

[6] Doxygen. *Threshold types*. Open Source Computer Vision, 3.4.0 edition, Dic 2017.

[7] P. Estévez. In *EL4106 - Inteligencia Computacional*, Primavera 2012.

[8] T. Guo, J. Dong, H. Li, and Y. Gao. Simple convolutional neural network on image classification. 2017.

[9] A. Gupta, R. Kumar, R. Gupta, and P. Wadhwa. Dgw-canny: An improved version of canny edge detector. In *International Symposium on Intelligent Signal Processing and Comunication Systems (ISPACS)*, Dec 2011.

[10] G. Hao, L. Min, and H. Feng. Improved self-adaptive edge detection method based on canny. In *2013 5th International Conference on Intelligent Human-Machine Systems and Cybernetics*, volume 2, pages 527–530, Aug 2013.

[11] K. Kim, P. Kim, Y. Chung, and D. Choi. Performance enhancement of yolov3 by adding prediction layers with spatial pyramid pooling for vehicle detection. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, Nov 2018.

[12] M. Massiris, C. Delrieux, and J.A. Fernandez. Detección de equipos de protección personal mediante red neuronal convolucional yolo. Badajoz, Sep 2018.

[13] M.Kast, A. Witkin, and D. Terzopoulos. Snakes: Active contours models. In *International Journal of Computer Vision 1*, pages 321–331, 1988.

[14] S. Parra and W. Cuervo. Sistemas de visión artificial integrado a plataforma aérea para la detección de personas en tiempo real.

[15] J. Ramírez and M. Chacón. Redes neuronales artificiales para el procesamiento de imágenes, una revisión de la última década. In *RIEEC, Revista de Ingeniería Eléctrica, Electrónica y Computación*, volume 9. RIEEC, Jul 2011.

[16] Redmon, Joseph, Farhadi, and Ali. Yolov3: An incremental improvement. *arXiv*, 2018.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. May 2016.

[18] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. Dec 2016.

[19] Joseph Redmon. Darknet: Open source neural networks in c. `http://pjreddie.com/darknet/`, 2013–2016.

[20] K.H. Schmincke. Las industrias forestales, elemento decisivo para el desarrollo socioeconómico.

[21] S. Suwanmanee, S. Chatpun, and P. Cabrales. Comparison of video image edge detection operators on red blood cells in microvasculature. In *The 6th 2013 Biomedical Engineering International Conference*, pages 1–4, Oct 2013.

[22] Junhua Tong, Hufeng Shi, Chuanyu Wu, Huanyu Jiang, and Taiwei Yang. Skewness correction and quality evaluation of plug seedling images based on canny operator and hough transform. *Computers and Electronics in Agriculture*, 155:461 – 472, 2018.

[23] J. F. Valencia-Murillo, D. A. Poveda-Sendales, and D. F. Valencia-Vargas. Evaluating the impact of image preprocessing on iris segmentation. In *Tecno Lógicas*, volume 17, May 2014.

[24] X. Wang, B. Li, and Q. Geng. Runway detection and tracking for unmanned aerial vehicle based on an improved canny edge detection algorithm. In *4th International Conference on Intelligent Human-Machine Systems and Cybernetics*, volume 2, Aug 2012.

[25] Woodtech. Logmeter, more than 35 installations worldwide. *Timber processing*, pages 14–14, Sep 2015.

[26] S. Yanan, Z. Hui, L. Li, and Z. Hang. Rail surface defect detection method based on yolov3 deep learning networks. In *2018 Chinese Automation Congress (CAC)*, pages 1563–1568, Nov 2018.

[27] B. Zhang. Computer vision vs. human vision. In *9th IEEE International Conference on Cognitive Informatics (ICCI'10)*, July 2010.

[28] H. Zhang, Q. Zhu, and X. Guan. Probe into image segmentation based on sobel operator and maximum entropy algorithm. In *2012 International Conference on Computer Science and Service System*, pages 238–241, Aug 2012.

Apéndice A

Planificación de desarrollo

El tiempo total de planificación considera seis meses de desarrollo. La fecha de inicio parte desde el 7 de enero de 2019 y finalizando el día 15 de Julio del mismo año. Los meses de enero y febrero consideran dedicación de 45 horas semanales, para los meses restantes se consideran 20 horas semanales. Las etapas y tiempos de dedicación de cada una de ellas se presentan en la tabla A.1, donde las primeras ocho subdivisiones consideran el desarrollo y prueba. Las últimas dos tarea consideran la documentación de los códigos.

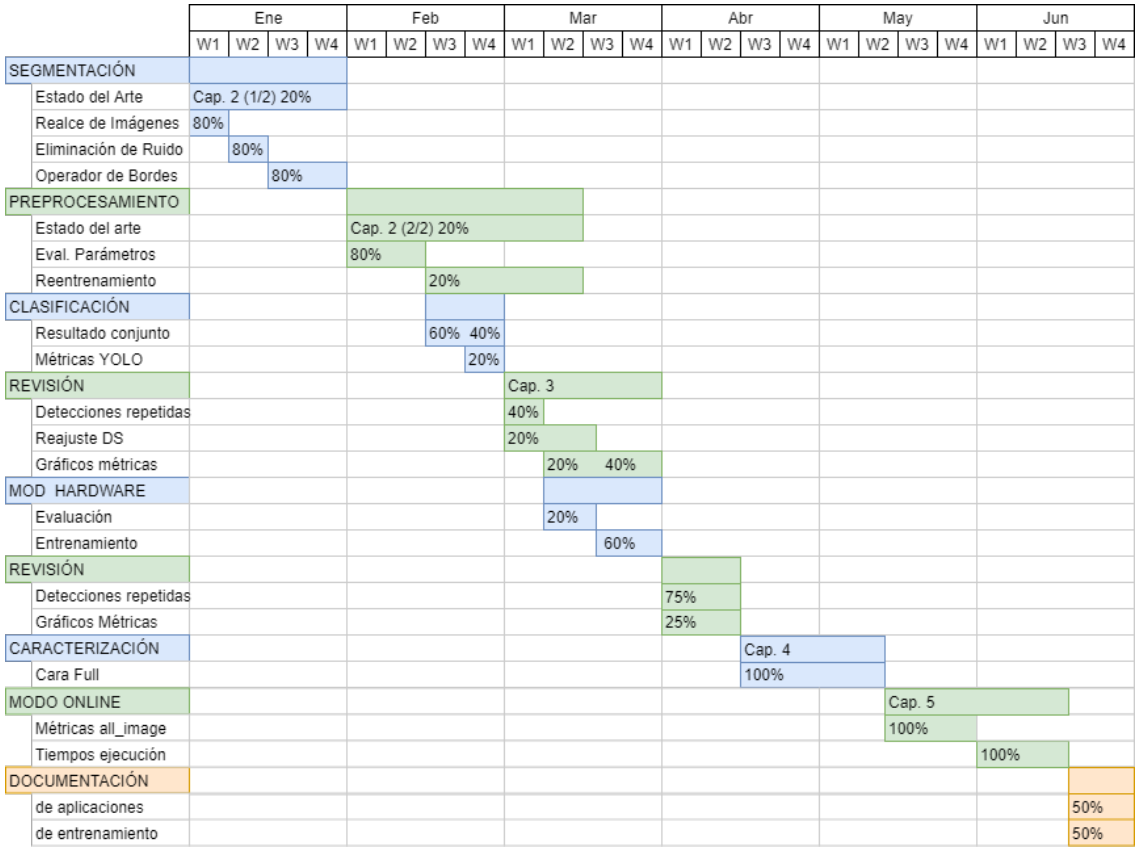

Figura A.1: Planificación desarrollo Memoria

Apéndice B

Ejemplo de imágenes de un Evento

Un evento corresponde al paso del camión por un portal de Logmeter5000. El evento es monitorizados por dos cámaras laterales y una superior, para la cual se guarda una secuencia de imágenes. En la figura B.1 y B.3 se muestra la secuencia de imágenes para la cámara uno y tres, que corresponden a las dos cámaras laterales. En la figura B.2 se muestra la secuencia de imágenes para la cámara superior, o cámara dos.

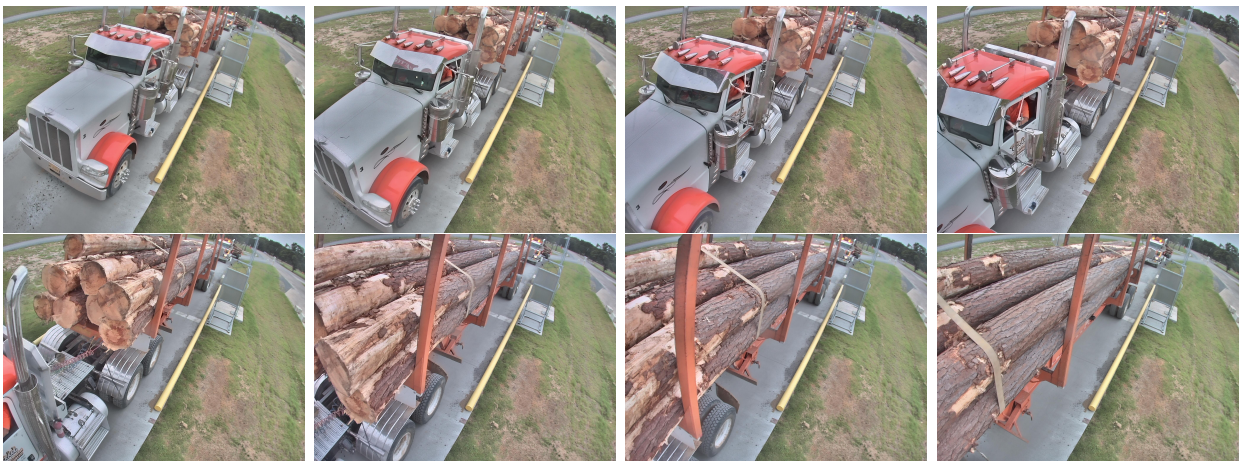

Figura B.1: Ejemplo de evento con cámara 1

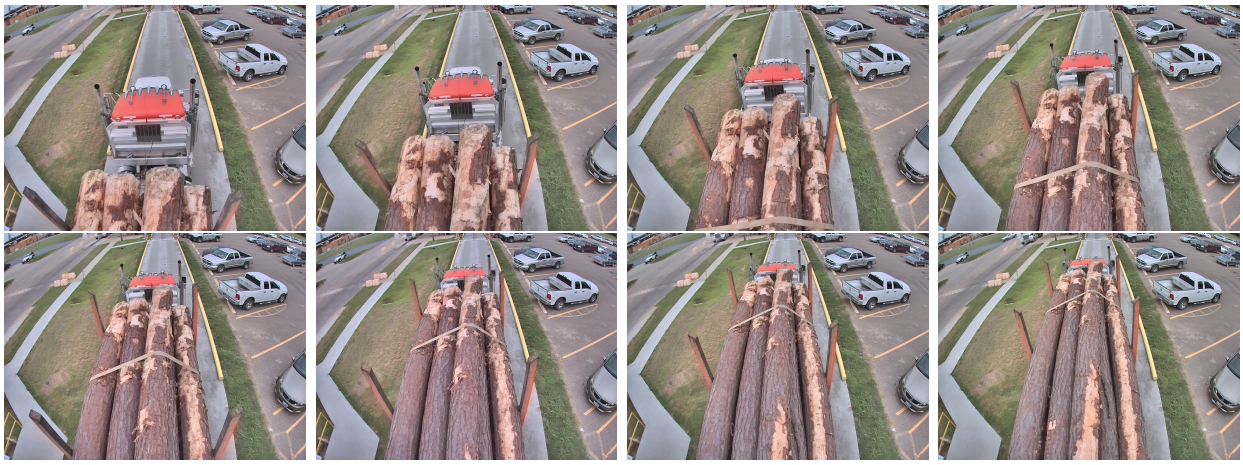

Figura B.2: Ejemplo de evento con cámara 2

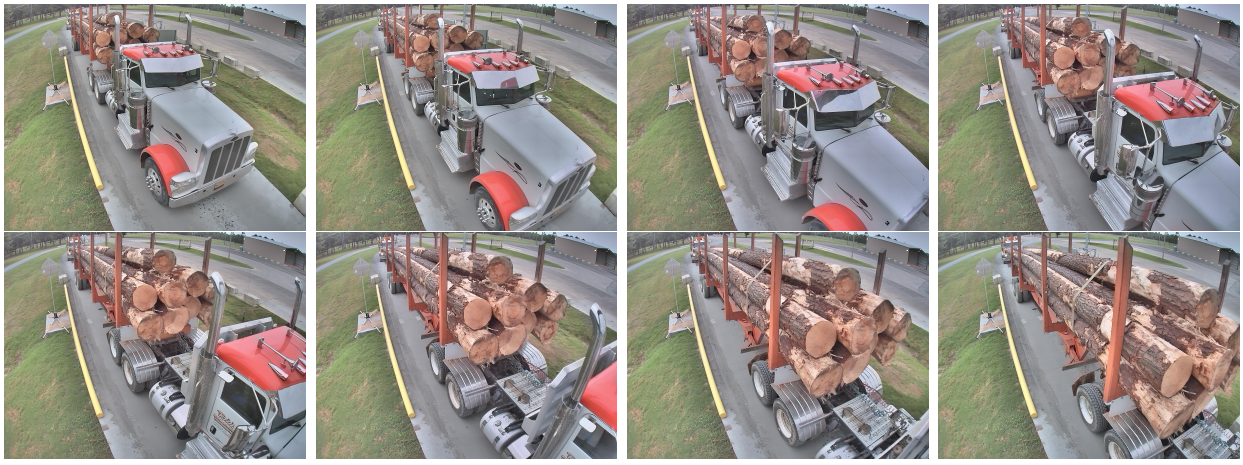

Figura B.3: Ejemplo de evento con cámara 3

Apéndice C

Creación base de datos

Manteniendo el formato de los datos que debe recibir como entrada la red entrenada, para la creación del set de entrenamiento de las clases que es necesario agregar a la red, se utilizó OpenLabeling. Ésta herramienta, opensource permite la segmentación y clasificación de múltiples objetos dentro de una imagen. Las marcas realizadas corresponden a rectángulos y las salidas se entregan en formato PascalVOC y YOLO_Darknet. Las categorías a utilizar se adjuntan en el archivo class_list.txt y contiene solo una categoría por cada línea.

Es necesario considerar los siguientes archivos en la carpeta principal:

- class_list.txt: Contiene la lista con las clases que se utilizaran en la etapa de marcaje. En la etapa de marcaje se utilizaron tres categorías Full, Partial y Croop.
- main.py: Contiene el código principal. Requiere de las librerias CV2, numpy y tqdm principalmente.
- input: Imágenes sobre las que se realizara la segmentación.
- output: Carpeta en la que se guardarán los datos de salida. Los archivos se guardan como xml para Pascal_voc y como txt para YOLO_darknet.

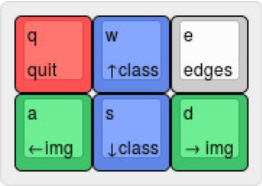

Figura C.1: Keyboard OpenLabelling

Utilizanbdo main.py se obtiene una pantalla de ejecución como la mostrada en la imagen C.2. En la parte superior esta el número de imágenes y se muestran las clases seleccionadas. La creación del recuadro se inicia y termina con un click sobre los puntos seleccionados de la imagen. Utilizando el teclado, como se muestra en la figura C.1 se modifica la clase con la letra W y S, se cambia de imagen con la letra A y D y se elimina la marca realizada con la letra Q.

Para la base de datos de contornos se utilizó la aplicación sketch disponible para bla. Se

Figura C.2: OpenLabelling

marcaron los datos utilizando el tablet bla con lápiz. Esta herramienta corresponde a una aplicación de bla. En la imagen bla se presentan los pasos para obtener la marca manual. Las imágenes son cargadas a la aplicación y se requiere de los siguientes pasos:

- Marcar el contorno de la cara seleccionada.
- Eliminar la imagen de fondo
- guardar imagen

Las imágenes de salida para esta etapa son imágenes con fondo blanco, con las mismas dimensiones que la imagen original, y con el contorno de la cara en color negro.

Apéndice D

Guía de instalación Darknet

Figura D.1: Logo Darknet

Existen dos implementaciones principales para ejecutar YOLO: darknet y darkflow. Con cada una de ellas es posible detectar objetos justo después de su instalación. Darknet está escrito en C con CUDA, así que soporta cómputo con GPU. Corresponde a la implementación oficial de YOLO. Darkflow es recomendado para hacer entrenamiento solamente usando CPU. Sin embargo no ha sido actualizado para YOLOV3.

Se selecciono Darknet por ser un framework open source para redes neuronales que corresponde a la implementación principal de YOLO. Fue desarrollado por Joseph Redmon y soporta computo con GPU. Éste framework ha generado grandes avances en el mundo de Machine Learning y Redes Neuronales. La instalación de darknet permitirá realizar predicciones sobre imágenes utilizando GPU y guardarlas en el disco. La instalación es relativamente fácil y solo requiere de dos dependencias de carácter opcional, OpenCV y CUDA [19].

Referencia	enlace
Darknet	`github.com/pjreddie/darknet`
Instalación Darknet	`pjreddie.com/darknet/install/`
CUDA	`developer.nvidia.com/cuda-downloads`

Tabla D.1: Enlaces para descargar Darknet y sus dependencias

Se utiliza OpenCV si se desea aumentar los formatos de imágenes compatibles. CUDA permite utilizar GPU. Es posible descargar la versión de CUDA adecuada a la tarjeta de memoria en uso, completando los datos de la tarjeta. Si es necesario actualizar una versión de CUDA existente se siguen los pasos de `dhaneshr.net`. Con `modinfo` podemos consultar directamente el módulo de nvidia cargado en el kernel, para verificar que el sistema reconoce la tarjeta. Se muestran los enlaces para descargar Darknet y sus dependencias en la tabla D.1.

Apéndice E

Guía de ejecución algoritmos diseñado

Para utilizar yolo en el servidor 10.1.10.251 es necesario posicionarse en la carpeta darknet y ejecutar los siguientes comandos en el terminal para resolver la incompatibilidad de versiones:

```
export LD_LIBRARY_PATH=/usr/local/lib
export PATH=/usr/local/cuda-10.1/bin:/usr/local/cuda-10.1/NsightCompute-2019.
    1\${PATH:+:\${PATH}}
export LD_LIBRARY_PATH=/usr/local/cuda-10.1/lib64\ ${LD_LIBRARY_PATH:+:${LD
    LIBRARY_PATH}}
```

Para el entrenamiento se considera utilizar un servidor completamente dedicado. Para ejecutar el entrenamiento de la red se deben seguir pasos a continuación:

1. Crear Base de datos siguiendo la guía de instalación en el apéndice **??**
2. Instalar Darknet siguiendo la guía de instalación en el apéndice **??**
3. Seguir la guía de entrenamiento de yolov2 disponible en *https://timebutt.github.io/static/how-to-train-yolov2-to-detect-custom-objects/*
4. Si se desea entrenar yolov3 seguir la guía disponible en *https://medium.com/@manivannan_data/how-to-train-yolov3-to-detect-custom-objects-ccbcafeb13d2*

Las configuraciones adicionales de la arquitectura de la red, dependiendo de la configuración de red a utilizar se resumen en la siguiente tabla:

configuración inicial	sección	linea	modificación
yolov1.cfg	[convolutional]	230	$filtros = (\text{cantidad de clases} + 5) * 5$
	[detection]	248	$classes = \text{cantidad de clases}$
yolov1-tiny.cfg	[convolutional]	230	$filtros = (\text{cantidad de clases} + 5) * 5$
	[detection]	248	$classes = \text{cantidad de clases}$
yolov3.cfg	[convolutional]	603, 689, 776	$filtros = (\text{cantidad de clases} + 5) * 3$
	[yolo]	610, 696	$classes = \text{cantidad de clases}$
yolov3-tiny.cfg	[convolutional]	127, 171, 783	$filtros = (\text{cantidad de clases} + 5) * 3$
	[detection]	135, 177	$classes = \text{cantidad de clases}$

Los resultados se guardan en la carpeta definida en el archivo data.data, para evaluar las imágenes de validación, se ejecuta darknet_2.py. Es necesario modificar las siguientes lineas en el código:

1. Linea 8 - ruta: Dirección que contiene las carpetas data y darknet
2. Linea 9 - train: Nombre asignado al entrenamiento de YOLO actual, contenido en la carpeta data.
3. Linea 10 - entr_: Nombre del archivo .cfg que contiene la configuración de la red a utilizar.
4. Linea 13 - fold: Nombre de la carpeta con las imágenes de validación.
5. Linea 14 - out_fold: Nombre de la carpeta que contendrá los resultados de la ejecución.
6. Line 15 - weight: carpeta que contiene los pesos del entrenamiento.

Para obtener las métricas sobre la red, los resultados de contornos activos, y los diámetros se ejecuta el código python correspondiente, descrito en la siguiente tabla. Adicionalmente se debe crear la carpeta DATA_CNN que contenga tres subcarpetas: data(cuando corresponda, contendrá los datos de marcaje para cada imagen) , yolo (contiene los resultados de la red en carpetas separadas por épocas de entrenamiento. Estos archivos son entregados en ésta estructura por la red), image (Contiene las imágenes).

Apéndice F

Datos de entrada y salida del algoritmo implementado

Dentro de la implementación del prototipo que permita estimar el número de caras de troncos para una imagen y entregar el diámetro correspondiente se considera un proceso de validación para cada etapa funcional. Para el modo on-line, se define e implementa la etapa de detección de regiones de interés. Sobre las regiones de interés detectadas se implementan las etapas de preprocesamiento y segmentación. Al evaluar las etapas de desarrollo inicialmente propuestas se definen las siguientes pruebas sobre el prototipo desarrollado.

Figura F.1: Ejemplo de Original Image (OI)

Las imágenes utilizadas para entrenamiento y validación de la red corresponden a datos de la planta WNB1, generadas por el portal Logmeter5000. Se consideran como datos de entrada Original_Image (OI) que corresponde a la imagen que se desea evaluar (ver imagen F.1 y Labelled_Image (LI) que corresponde a un archivo txt con los datos marcados para cada imagen. Por línea de LI se entrega información de la clase, x_centro, y_centro, ancho y alto de una cara en la imagen, se muestra un ejemplo a continuación.

```
1 0.709228515625 0.3968098958333333 0.04736328125 0.10872395833333333
1 0.775146484375 0.4391276041666667 0.05517578125 0.10221354166666667
1 0.650390625 0.4729817708333333 0.0908203125 0.134765625
0 0.61083984375 0.1669921875 0.083984375 0.12434895833333333
0 0.537841796875 0.361328125 0.09228515625 0.13411458333333334
0 0.555908203125 0.515625 0.09814453125 0.1328125
0 0.6552734375 0.3492838541666667 0.0791015625 0.142578125
0 0.731689453125 0.2854817708333333 0.07373046875 0.130859375
0 0.80078125 0.3291015625 0.06640625 0.11783854166666667
2 0.6630859375 0.3391927083333333 0.3525390625 0.4934895833333333
```

Una salida intermedia de esta etapa son las detecciones de la red. Automati_Label_List (AL) contiene las detecciones de la red indicada para una imagen y contiene información de la categoría, X_centro, Y_centro, ancho, alto, número y el tiempo de detección sobre la imagen. Es posible que sea má rápido obtener los resultados de la red desde un servidor diferente al que se utiliza para obtener las métricas, en tal caso, el archivo AL se considerará como un dato más de entrada en la etapa de detección de caras.

```
Full 1245.374 264.377 182.882 200.001 1 0.999 0.048
Full 1091.697 549.385 218.783 249.268 2 0.999 0.048
Full 1143.577 786.275 253.649 259.345 3 0.999 0.048
Full 1587.277 676.818 123.502 165.578 4 0.998 0.048
Partial 1334.767 742.004 201.512 203.021 5 0.998 0.048
Full 1484.944 436.307 189.710 211.512 6 0.995 0.048
Partial 1436.983 620.377 142.422 182.889 7 0.991 0.048
Full 1331.660 518.027 226.519 249.228 8 0.975 0.048
Full 1638.032 516.462 143.112 186.052 9 0.969 0.048
```

La salida de esta primera etapa de verificación son las métricas de comparación entre las detecciones de la red sobre la imagen de entrada (OI) y los datos de marcas (LI). Contiene información de x_centro, y_cento, ancho, alto y categoría del box detectado y el de marcaje, el valor de precisión, recall, IoU, FP, VP y FN. La información de salida se almacena en el archivo storage.txt. Se muestra a continuación, un ejemplo de datos de salida para la imagen y los datos de marcaje ya señalados.

```
[1149.85, 160.31, 190.74, 199.09, 'Full', 1165.0, 161.0, 172.0, 191.0,
'0', 0.85, 0.98, 0.84, 0, 1, 0, 4]
[1024.95, 675.32, 229.81, 233.69, 'Full', 1038.0, 690.0, 201.0, 204.0,
'0', 0.76, 1.0, 0.76, 0, 1, 0, 6]
[986.99, 439.36, 209.84, 221.35, 'Full', 1007.0, 452.0, 189.0, 206.0,
'0', 0.78, 0.93, 0.73, 0, 1, 0, 5]
```

```
[1239.66, 643.31, 186.16, 188.05, 'Partial', 1239.0, 623.0, 186.0,
207.0, '1', 1.0, 0.91, 0.9, 0, 1, 0, 3]
[1529.5, 592.91, 117.24, 156.33, 'Full', 1531.0, 596.0, 113.0, 157.0,
'1', 0.95, 0.98, 0.93, 0, 1, 0, 2]
[1566.51, 427.98, 132.41, 161.17, 'Full', 1572.0, 415.0, 136.0, 181.0,
'0', 0.97, 0.84, 0.82, 0, 1, 0, 9]
[1382.35, 333.44, 193.92, 204.65, 'Full', 1423.0, 338.0, 151.0, 201.0,
'0', 0.66, 0.86, 0.6, 0, 1, 0, 8]
[1221.59, 410.22, 218.25, 223.7, 'Full', 1261.0, 427.0, 162.0, 219.0,
'0', 0.63, 0.87, 0.58, 0, 1, 0, 7]
[0, 0, 0, 0, 'null', 1404.0, 526.0, 97.0, 167.0, '1', 0, 0, 0, 0, 0, 1, 1]
```

A continuación se presentan los resultados para esta etapa, que corresponden a las salidas de execute_metric sobre una imagen o sobre el total de imágenes, ambos casos para la época de entrenamiento indicada. En la información entregada se agrega el porcentaje de verdaderos positivos y falsos negativos, y el total de caras presentes (verdaderos positivos más falsos negativos).

Imagen	Pres	Rec a	IoU	FP	VP	VP%	FN	FN%	cara	Pre s	Rec
imagen	0.73	0.82	0.68	0	8	88.89	1	11.11	9	1.0	0.89
total	0.64	0.97	0.8	148	1027	76.28	319	23.72	13474	0.87	0.76

Para la validación del módulo de detección de contornos se utilizo una nueva base de datos con marcas de contornos. Para la obtención de las métricas se consideran como datos de entrada Original_Image (OI), Labelled_Image II (LI-II) y ALML (archivo de salida de la detección de caras). La salida corresponde a la evaluación de contorno para todas las caras de la imagen detectadas como full y una métrica del diámetro promedio de la imagen.

En la operación del Modo online se utiliza como entrada una imagen de la carga del camión. Para su ejecución se debe mantener la estructura de carpetas presentada, ya que la mayoría de los módulos incluye referencias a otras carpetas. Como salida se entrega un arreglo que contiene el diámetro de cada cara detectada. Por columna se entrega información de x_centro, y_centro, ancho, alto, categoría, tag, probabilidad de detección, radio de la adivinanza inicial, radio del contorno. Adicionalmente se entrega el número de caras, las que cumplen con el umbral, radio promedio para la imagen y el tiempo de la detección y de la segmentación.

```
(1149.848, 160.307, 190.738, 199.085, 'Full', '1', 1.0, 99.54, 99.09)
(1024.952, 675.319, 229.812, 233.686, 'Full', '2', 1.0, 116.84, 114.85)
(986.9948, 439.364, 209.841, 221.346, 'Full', '3', 1.0, 110.67, 111.13)
(1239.659, 643.310, 186.164, 188.051, 'Partial', '4', 0.996, 94.03, 96.25)
```

```
(1529.503, 592.914, 117.243, 156.330, 'Full', '5', 0.993, 78.17, 78.04)
(1566.512, 427.984, 132.414, 161.172, 'Full', '6', 0.977, 80.59, 82.34)
(1382.346, 333.437, 193.922, 204.650, 'Full', '7', 0.954, 102.33, 100.49)
(1221.592, 410.218, 218.247, 223.702, 'Full', '8', 0.906, 111.85, 110.95)
(1361.327, 542.004, 151.125, 160.863, 'Partial', '9', 0.657, 80.43, 79.26)
```

Caras	Caras P>0.5	Radio Promedio	Tiempo Red	Tiempo Contorno
9	9	96.93	0.895	66.429

Finalmente se guarda red.png y ac.png (imagen F.2). Red.png que corresponde a la imagen inicial con marcas de la red para las dos clases, Full (cian) y Partial (blanco). AC.png corresponde a la imagen inicial con ajuste de contornos activos para ambas clases. EN ambas imágenes, el número corresponde al tag asignado a cada cara detectada.

Figura F.2: Ejemplo de detecciones de la red (izq) y ajuste de contornos para a imágen (der)