PROYECTO FIN DE GRADO

TÍTULO: Sistema de conteo de larvas de rodaballos mediante mapas de densidad

AUTOR: Guillermo Armesto Parra

TITULACIÓN: Electrónica de Comunicaciones

TUTORA: Juana María Gutiérrez Arriola

DEPARTAMENTO: Ingeniería Telemática y Electrónica

VºBº TUTORA

Miembros del Tribunal Calificador:

PRESIDENTE: Rubén Fraile Muñoz

TUTORA: Juana María Gutiérrez Arriola

SECRETARIO: Nicolás Sáenz Lechón

Fecha de lectura: 23 de julio de 2024

Calificación:

El Secretario

Resumen

En el ámbito de la acuicultura, la estimación precisa del número y tamaño de los peces en un tanque es crucial para la viabilidad de las piscifactorías. Este proyecto consiste en el desarrollo de una aplicación para el conteo de larvas de rodaballos mediante el uso de mapas de densidad generados a partir de redes neuronales convolucionales. Esta tecnología se implementó debido a su eficacia demostrada en tareas de conteo de objetos en imágenes.

Este proyecto continúa los resultados obtenidos en el proyecto final de grado de Alejandro Fernández González, quien utilizó el modelo SS-DCNet (Supervised Spatial Divide-and-Conquer Network), conocido por su precisión en el conteo de personas y otros objetos en diversas aplicaciones. En su proyecto, Alejandro adaptó el modelo SS-DCNet para trabajar con imágenes de alta densidad de larvas de rodaballo, logrando una precisión notable con un error medio absoluto inferior al 3.5%. En este proyecto, se amplía el estudio para el cálculo del error utilizando una desviación estándar variable (σ) para conjuntos de datos en función del día de nacimiento de la larva, en lugar de una desviación estándar fija para un conjunto de datos global.

Uno de los principales desafíos abordados en este proyecto fue la variabilidad en los datos de entrenamiento, específicamente en cuanto a las desviaciones estándar utilizadas en los mapas de densidad. Los resultados mostraron que, a pesar de las variaciones en los datos, el uso de una desviación estándar fija de 3 proporcionó el menor error en casi todos los conjuntos de datos evaluados. Esta observación se realizó tras comparar los resultados obtenidos con diferentes desviaciones estándar para los conjuntos de datos correspondientes a los días 23-25, 26, y 27-29. Para el conjunto de datos de los días 27-29, por ejemplo, los errores producidos mediante la desviación estándar de 4 resultaron considerablemente menor que el obtenido con otras desviaciones estándar.

A lo largo del proyecto, se utilizaron varios conjuntos de datos etiquetados manualmente, lo cual implicó un esfuerzo significativo en la preparación y anotación de las imágenes para todos los alumnos del grupo GAMMA (Grupo de Aplicaciones Multimedia y Acústica) . Sin embargo, la precisión alcanzada justifica este esfuerzo, destacando la importancia de los datos de entrenamiento referentes a su calidad en el rendimiento del modelo. Además, se evaluó el impacto de la variación en la densidad de los objetos en las imágenes, concluyendo que el modelo es robusto y capaz de mantener una alta precisión incluso en escenarios con alta densidad de objetos.

Como conclusión, este proyecto demuestra que es más efectivo la utilización de una desviación estándar fija para cada conjunto de datos específico, basados en el día de nacimiento de la larva, en lugar de utilizar una desviación estándar variable. Esto mejora la precisión y consistencia de los conteos automatizados, contribuyendo a la eficiencia y sostenibilidad de las piscifactorías, optimizando tanto el proceso de alimentación como la gestión de recursos acuícolas.

Abstract

In the field of aquaculture, the accurate estimation of the number and size of fish in a tank is crucial for the viability of fish farms. This project involves the development of an application for counting turbot larvae using density maps generated by convolutional neural networks. This technology was implemented due to its proven effectiveness in object counting tasks in images.

This project continues the results obtained in Alejandro Fernández González's final degree project, who used the SS-DCNet model (Supervised Spatial Divide-and-Conquer Network), known for its accuracy in counting people and other objects in various applications. In his project, Alejandro adapted the SS-DCNet model to work with high-density images of turbot larvae, achieving remarkable accuracy with a mean absolute error of less than 3.5%. In this project, the study is expanded to calculate the error using a variable standard deviation (σ) for data sets based on the larva's day of birth, instead of a fixed standard deviation for a global data set.

One of the main challenges addressed in this project was the variability in the training data, specifically regarding the standard deviations used in the density maps. The results showed that despite the variations in the data, using a fixed standard deviation of 3 provided the lowest error in almost all the evaluated data sets. This observation was made after comparing the results obtained with different standard deviations for the data sets corresponding to days 23-25, 26, and 27-29. For the data set from days 27-29, for example, the errors produced using a standard deviation of 4 were significantly lower than those obtained with other standard deviations.

Throughout the project, several manually labeled data sets were used, which involved a significant effort in preparing and annotating the images by all the students in the GAMMA group (Group of Multimedia and Acoustic Applications). However, the achieved accuracy justifies this effort, highlighting the importance of the quality of the training data in the model's performance. Additionally, the impact of the variation in object density in the images was evaluated, concluding that the model is robust and capable of maintaining high accuracy even in scenarios with high object density.

As a conclusion, this project demonstrates that using a fixed standard deviation for each specific data set, based on the larva's day of birth, is more effective than using a variable standard deviation. This improves the accuracy and consistency of automated counts, contributing to the efficiency and sustainability of fish farms, optimizing both the feeding process and the management of aquaculture resources.

Índice de figuras

Índice de Tablas

Lista de acrónimos

Acrónimos	Significado
ADN	ácido desoxirribonucleico
CNN	Convolutional Neural Network
CNN	Convolutional neural network
FNN	feedforward neural network
GAN	Generative Adversarial Networks
GAMMA	Grupo de Aplicaciones Multimedia y Acústica
IA	Inteligencia artificial
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
ML	Machine Learning
MSE	Mean Square Error
ODS	Objetivos de Desarrollo Sostenible
PCA	Principal Component Analysis
ReLu	Rectified Linear Unit
ResNet-50	Residual Network
RGB	Red Green Blue
RNN	Recurrent Neural Network
RMSE	Root Mean Squared Error
SS-DCNET	Supervised Spatial Divide and Conquer Network
SVM	Support vector machine
TFG	Trabajo final de grado
VGG	Visual Geometry Group

Índice de contenidos

1. Introducción

En la acuicultura, obtener una estimación precisa de la biomasa de los peces es esencial para la sostenibilidad económica y ambiental de las operaciones. La biomasa se calcula como el número total de peces en un área multiplicado por su peso promedio. Esta información es fundamental para optimizar recursos, como la cantidad de alimento suministrado, evitando el desperdicio y manteniendo la calidad del agua. Además, permite a las empresas mejorar sus estrategias de negocio, favoreciendo un crecimiento y producción sostenible. Tradicionalmente, la estimación de la biomasa se ha realizado mediante métodos manuales y visuales, que son invasivos, laboriosos y a menudo imprecisos. Estos métodos implican la intervención directa del personal, lo que puede afectar el bienestar de los peces y no proporciona datos precisos en tiempo real.

En las últimas décadas, los avances en tecnología de sensores, visión por computadora y aprendizaje automático han revolucionado la estimación de biomasa en acuicultura. Las técnicas basadas en visión artificial, tanto en espectro visible como infrarrojo, han permitido desarrollar métodos no intrusivos para contar peces. Estas técnicas utilizan cámaras y algoritmos de procesamiento de imágenes para detectar y contar automáticamente los peces en los tanques. Una de las tecnologías más prometedoras es el uso de redes neuronales convolucionales (CNN), que han demostrado ser efectivas en tareas de visión por computadora, como la clasificación de imágenes y la detección de objetos. Las CNN son capaces de procesar grandes volúmenes de datos visuales y extraer características relevantes para identificar y contar objetos específicos dentro de una imagen.

El objetivo principal para este proyecto consiste en la continuación del trabajo de Alejandro Fernández González [29], quien ya demostró la eficacia del modelo SS-DCNet (Supervised Spatial Divide-and-Conquer Network) en la adaptación para el conteo de larvas de rodaballos. Este proyecto se centrará en evaluar la aplicación de una desviación estándar (σ) variable para cada conjunto de datos, basado en el día de nacimiento de las larvas, en lugar de usar una sigma fija para un conjunto de datos global. Se espera que esta metodología proporcione una mayor precisión y consistencia en las estimaciones de biomasa, ajustando dinámicamente los parámetros del modelo según las características específicas de cada conjunto de datos.

La implementación de un sistema automatizado de conteo de peces basado en CNN ofrece múltiples beneficios. Primero, mejora la precisión y la eficiencia de las estimaciones de biomasa, lo que permite una gestión más sostenible de los recursos en la acuicultura. Segundo, reduce la necesidad de intervención humana directa, minimizando el estrés y la perturbación de los peces. Finalmente, proporciona datos en tiempo real que pueden utilizarse para optimizar la alimentación y otras prácticas de manejo, mejorando la productividad y reduciendo los costos operativos. Este proyecto se alinea con las tendencias actuales de digitalización y automatización en la acuicultura, contribuyendo a la transición hacia prácticas más sostenibles y eficientes en el sector.

El presente documento se organiza de la siguiente manera: Introducción, donde se presenta el contexto y la relevancia del proyecto, los objetivos y la estructura del documento. Marco Tecnológico, describiendo las tecnologías y métodos utilizados en la estimación de biomasa en acuicultura, con un enfoque en el aprendizaje automático y las redes neuronales. Especificaciones y Restricciones de Diseño, detallando las especificaciones técnicas y las limitaciones del diseño del sistema propuesto. Descripción de la Solución Propuesta, exponiendo el proceso de desarrollo del modelo SS-DCNet, incluyendo la preparación de datos, el entrenamiento y la evaluación del modelo. Resultados, presentando los resultados obtenidos y analizando la precisión y la robustez del modelo en diferentes escenarios. Conclusiones, resumiendo los hallazgos del proyecto, destacando las contribuciones y las posibles direcciones futuras para la investigación y el desarrollo en este campo. Con esta estructura, el documento ofrece una visión comprensiva y detallada del desarrollo e implementación de un sistema innovador para el conteo de peces en acuicultura, basado en tecnologías avanzadas de aprendizaje automático y visión por computadora.

2. Marco tecnológico

En este apartado exploraremos los diferentes métodos y tecnologías utilizados para el conteo de peces en el ámbito de la acuicultura, desde técnicas manuales hasta avanzadas tecnologías basadas en sensores. Se profundiza en el aprendizaje máquina, destacando su aplicación en la acuicultura, y se describen las redes neuronales, sus tipos y funcionamiento. Además, se detallan las redes neuronales convolucionales (CNN), su arquitectura y su uso específico en el conteo de objetos, con ejemplos aplicados al conteo de rodaballos.

2.1 Métodos y tecnologías usadas en conteo para Acuicultura

En la acuicultura, el conteo preciso de las poblaciones de peces es esencial para la optimización de los recursos disponibles y de la mejora de la eficiencia productiva. A continuación, se presentan los métodos y tecnologías más comunes utilizados para este propósito, junto con sus ventajas y desventajas.

2.1.1 Conteo manual

Los métodos tradicionales de conteo manual implican contar físicamente los peces en los tanques. Este método, aunque directo, es propenso a errores humanos y puede ser estresante para los peces, afectando su bienestar y crecimiento. Además, es un proceso laborioso y que consume mucho tiempo, lo que limita su aplicabilidad en operaciones a gran escala [1]

2.1.2 Tecnologías basadas en sensores

2.1.2.1 Radares

Los sistemas de radar utilizan ondas de radio para detectar y contar peces. Estos métodos son no intrusivos y pueden realizar el conteo de peces sin necesidad de manipularlos directamente. La tecnología de radar ofrece la ventaja de funcionar en diversas condiciones ambientales, lo que la hace adecuada para diferentes tipos de acuicultura. Los sistemas de radar pueden penetrar tanto en aguas claras como turbias, permitiendo un monitoreo continuo sin afectar a los peces. Sin embargo, la precisión de estos sistemas puede verse afectada por la interferencia de objetos no deseados y la necesidad de calibraciones regulares para mantener la exactitud de los datos[2]. Estudios actuales han demostrado que el uso de radares en acuicultura puede proporcionar datos precisos sobre la biomasa y el comportamiento de los peces, facilitando una gestión más eficiente de los recursos acuícolas.

2.1.2.2 Sistemas acústicos

Los sistemas acústicos, como el sonar, utilizan ondas sonoras para detectar y contar peces. Estos métodos son no intrusivos y pueden realizar el conteo de peces sin necesidad de manipularlos directamente. La precisión de estos sistemas puede verse afectada por factores como la presencia de otros objetos en el agua. Sin embargo, los avances recientes en algoritmos de aprendizaje profundo han mejorado significativamente la capacidad de análisis

de imágenes de sonar, permitiendo una detección y clasificación más precisa de las especies de peces [3].

Los algoritmos de aprendizaje profundo aplicados al análisis de imágenes de sonar han demostrado ser efectivos en la identificación y clasificación de peces en diversas condiciones ambientales. Estos algoritmos pueden procesar grandes volúmenes de datos y aprender características complejas de las imágenes de sonar, lo que mejora la precisión y robustez del sistema en comparación con métodos tradicionales. Los recientes avances en técnicas de aprendizaje profundo han optimizado la capacidad de los sistemas de sonar para realizar monitoreos precisos y detallados, proporcionando datos valiosos para la gestión y sostenibilidad en la acuicultura.

2.1.2.3 Análisis de ADN

El análisis de ADN mide la concentración de ADN en el agua para estimar el número de peces. Este método es indirecto y puede no ser preciso para conteos instantáneos, ya que depende de la distribución y la degradación del ADN en el medio acuático [4]. No obstante, es útil para estimaciones globales y puede complementar otros métodos de conteo. Investigaciones han demostrado que el uso de ADN ambiental (eDNA) permite la detección de especies presentes en el agua sin necesidad de observación directa, lo que puede ser particularmente útil en entornos difíciles de acceder o donde el estrés de los peces es una preocupación .

2.1.2.4 Visión por computadora

La visión por computadora utiliza cámaras y algoritmos de procesamiento de imágenes para contar peces en tiempo real. Esta tecnología ha demostrado ser efectiva y no intrusiva, permitiendo un monitoreo continuo y preciso de las poblaciones de peces. Los recientes progresos en inteligencia artificial y aprendizaje profundo han llevado al desarrollo de métodos de segmentación de instancias y otras técnicas avanzadas que mejoran significativamente la precisión del conteo de peces.

Un ejemplo destacado de estos avances es el método de conteo en tiempo real basado en la segmentación de instancias. Este método utiliza redes neuronales convolucionales (CNN) avanzadas para identificar y segmentar cada pez en una imagen de manera individual, lo que permite contar con precisión incluso en entornos con alta densidad de población[5]. La segmentación de instancias no solo detecta la presencia de peces, sino que también delimita los contornos de cada individuo, facilitando una estimación precisa del número de peces presentes.

Otro enfoque innovador es el uso de redes de agregación de características multi-escala densas (Dense Multi-scale Feature Aggregation Network). Esta técnica mejora la precisión del conteo de peces mediante la red convolucional ResNet-50 (Residual Network) al combinar información de diferentes escalas de la imagen, permitiendo una mejor detección y clasificación de peces de diferentes tamaños y en diversas condiciones ambientales. La red de agregación multi-escala es capaz de manejar la variabilidad en la apariencia de los peces debido a factores como la iluminación, la turbidez del agua y la orientación de los peces [6].

Además, el método automático de conteo de alevines basado en visión por computadora proporciona una solución eficiente para el monitoreo de poblaciones de peces jóvenes. Este enfoque utiliza algoritmos de segmentación y clasificación para distinguir y contar alevines en imágenes de alta resolución. La automatización del conteo de alevines es particularmente importante en acuicultura, ya que permite un seguimiento preciso del crecimiento y la supervivencia de los peces en las primeras etapas de vida, lo que es crucial para la gestión y planificación de la producción [7].

Estos métodos avanzados de visión por computadora se pueden implementar en sistemas de monitoreo en tiempo real, proporcionando datos continuos sobre la población de peces sin interrumpir su entorno. Esto permite a los operadores de acuicultura tomar decisiones informadas y oportunas sobre la gestión de los recursos, mejorando la eficiencia y sostenibilidad de las operaciones acuícolas .

2.1.2.5 Detección y Clasificación con Datos Temporales

Un enfoque avanzado en la detección y clasificación de peces implica el uso de aprendizaje profundo junto con información temporal para mejorar la precisión y la robustez del sistema. Utilizando técnicas de procesamiento de imágenes y secuencias temporales, los sistemas pueden no solo identificar y contar los peces en una imagen fija, sino también rastrear su movimiento y comportamiento a lo largo del tiempo. Este método aumenta la precisión del conteo al reducir las duplicaciones y omisiones que pueden ocurrir en análisis de imágenes estáticas. Estudios actuales han revelado que la inclusión de información temporal en modelos de aprendizaje profundo puede mejorar significativamente la precisión en la clasificación de especies y la detección de peces en ambientes submarinos[8]. Dicha información temporal consiste en la combinación de modelos de mezcla gaussiana y flujo óptico con la red neuronal YOLO.

2.2 Aprendizaje máquina

El aprendizaje máquina o machine learning (ML) es una subdisciplina de la inteligencia artificial (IA) que se centra en crear algoritmos y modelos que capacitan a las computadoras para aprender y realizar predicciones a partir de datos. En el contexto de la acuicultura, el aprendizaje máquina ha demostrado ser una herramienta poderosa para optimizar diversas operaciones y mejorar la eficiencia y sostenibilidad de las prácticas acuícolas.

2.2.1 Conceptos básicos

El aprendizaje máquina se apoya en el concepto de que los sistemas son capaces de aprender a partir de los datos, reconocer patrones y tomar decisiones con mínima intervención humana [9] [10]. Existen varios tipos de aprendizaje máquina, entre los que destacan:

- **Aprendizaje Supervisado:** Este tipo de aprendizaje se basa en el uso de datos etiquetados para entrenar modelos. Los algoritmos aprenden a partir de un conjunto de entrenamiento y luego aplican lo aprendido a nuevos datos no etiquetados. Algunos de los algoritmos supervisados populares son las redes neuronales, la regresión lineal

y las máquinas de vectores de soporte (SVM). Este método se puede utilizar para el aprendizaje de clasificación y regresión, donde se intenta predecir una etiqueta o un valor continuo, respectivamente, basándose en un conjunto dado de características de entrada.

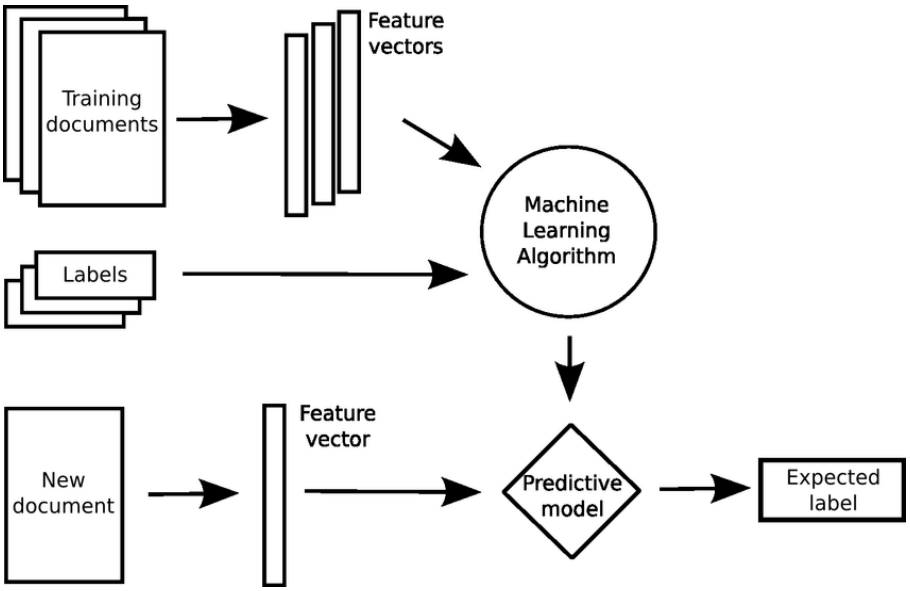


Figura 1: Aprendizaje supervisado.[11]

- **Aprendizaje No Supervisado:** En ese caso, los algoritmos operan en datos no etiquetados y hacen un esfuerzo para encontrar patrones de estructura o regularidad subyacente en los datos. Algunos de los algoritmos no supervisados más comunes son la agrupación y el análisis de componentes principales (PCA). El aprendizaje no supervisado se aplica en la segmentación de clientes, la detección de anomalías y la reducción de la dimensionalidad.

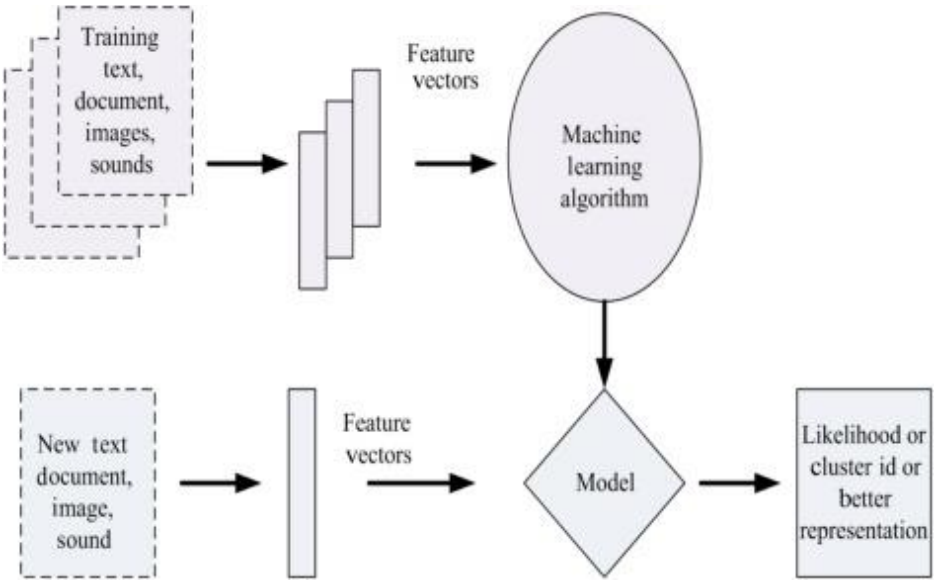


Figura 2: Aprendizaje no supervisado. [12]

- **Aprendizaje Semi-Supervisado**: Este tipo de aprendizaje combina elementos de los enfoques supervisado y no supervisado. Se utiliza un pequeño conjunto de datos etiquetados junto con un gran conjunto de datos no etiquetados para entrenar los modelos. Este enfoque es especialmente útil cuando la obtención de datos etiquetados es costosa o laboriosa.

- **Aprendizaje por Refuerzo**: En este enfoque, los algoritmos aprenden a tomar decisiones mediante la interacción con un entorno dinámico. Las acciones tomadas por el agente son recompensadas o castigadas, y el objetivo es maximizar la recompensa acumulada a lo largo del tiempo. Ejemplos de aplicaciones incluyen los sistemas de recomendación, los robots autónomos y los juegos.

2.2.2 Aplicaciones en Acuicultura

El aprendizaje máquina tiene numerosas aplicaciones en la acuicultura, mejorando significativamente la eficiencia y sostenibilidad de las operaciones. A continuación, se describen algunas de las aplicaciones más destacadas:

1. Predicción de la Calidad del Agua

El monitoreo y la predicción de la calidad del agua son cruciales para mantener un entorno saludable para los peces. Los modelos de aprendizaje máquina pueden analizar datos históricos y en tiempo real de sensores para predecir cambios en parámetros clave como la temperatura, el pH y los niveles de oxígeno disuelto. Esto permite a los acuicultores tomar medidas preventivas para evitar condiciones adversas.

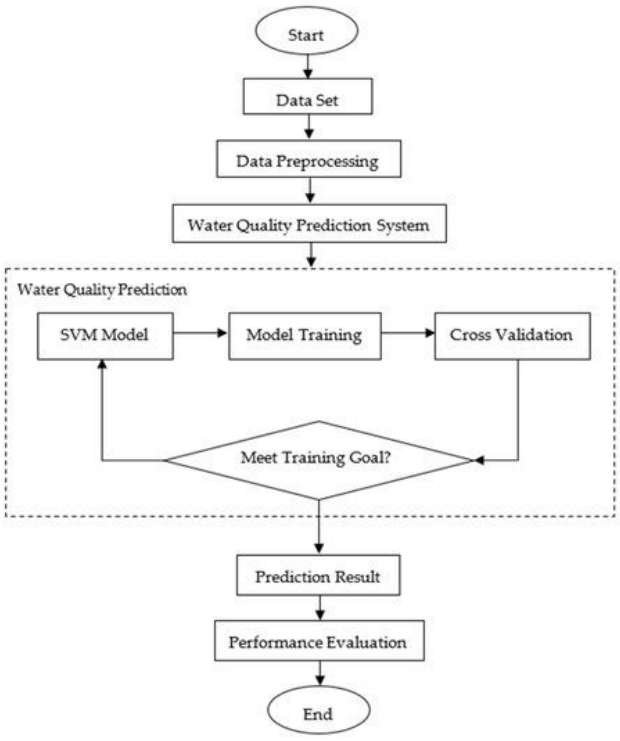


Figura 3: Esquema de Sistema de Monitoreo de Calidad del Agua. [14]

2. Detección de Enfermedades

El aprendizaje máquina se utiliza para detectar enfermedades en peces a partir de imágenes y datos de comportamiento. Los algoritmos pueden identificar patrones que indican la presencia de enfermedades antes de que sean visibles a simple vista, lo que permite una intervención temprana y reduce la mortalidad. Los sistemas de detección temprana basados en aprendizaje máquina pueden analizar imágenes de alta resolución y datos de comportamiento de los peces para identificar signos de estrés o enfermedad, facilitando una gestión proactiva de la salud de los peces.

3. Optimización de la Alimentación

La alimentación representa uno de los costos más significativos en la acuicultura. Los sistemas de aprendizaje máquina pueden optimizar los regímenes de alimentación al examinar los datos sobre el crecimiento de los peces, la ingesta de alimento y las condiciones ambientales. Esto ayuda a minimizar el desperdicio y maximizar la eficiencia del alimento.

4. Conteo y Clasificación de Peces

Como se mencionó en secciones anteriores, la visión por computadora y el aprendizaje profundo se utilizan para el conteo y la clasificación de peces en tiempo real. Esto es esencial para monitorear el crecimiento de las poblaciones y gestionar adecuadamente los recursos. Los sistemas basados en visión por computadora pueden procesar imágenes en tiempo real, permitiendo un conteo preciso y una clasificación automática de los peces en diferentes categorías según el tamaño, la especie y otros atributos.

5. Modelado del Crecimiento de los Peces

Los modelos de aprendizaje máquina pueden predecir el crecimiento de los peces basándose en datos históricos y condiciones actuales. Estos modelos ayudan a los acuicultores a planificar mejor la cosecha y optimizar los tiempos de producción. Al analizar datos como la temperatura del agua, la calidad del alimento y las condiciones ambientales, los modelos pueden proporcionar estimaciones precisas del crecimiento y la salud de los peces, lo que permite una gestión más eficiente de la producción.

2.3 Redes Neuronales

Las redes neuronales son una tecnología esencial en el aprendizaje automático y la inteligencia artificial. Basadas en la estructura y el funcionamiento del cerebro humano, estas redes consisten en unidades interconectadas conocidas como neuronas artificiales, que colaboran para procesar información y aprender a partir de los datos.

2.3.1 Funcionamiento de las Redes Neuronales

Las redes neuronales trabajan transmitiendo información a través de múltiples capas de neuronas artificiales. Cada neurona recibe varias entradas, las procesa y genera una salida que se envía a las neuronas de la siguiente capa. El aprendizaje en las redes neuronales consiste

en ajustar los pesos de las conexiones entre neuronas para reducir al máximo el error en las predicciones.

Neuronas Artificiales: Cada neurona artificial calcula una suma ponderada de sus entradas y aplica una función de activación para producir una salida **(figura 5).**

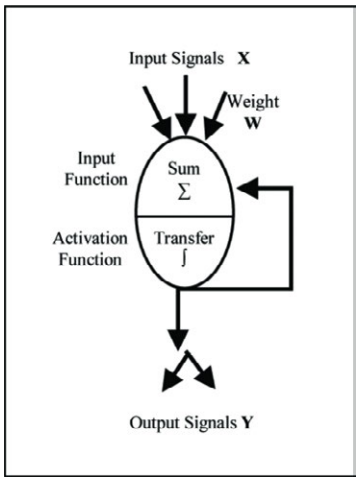


Figura 4: Esquema de una neurona artificial.[15]

La fórmula básica (1) es una representación matemática fundamental en el modelo de perceptrón, utilizado en redes neuronales para tareas de clasificación y predicción que representa la función $f(x)$ de salida para una asignacion x_i en su entrada

$$f(x) = \Phi\left(\sum_{i=1}^{n} w_i x_i + b\right)(1)$$

Donde la suma $\sum_{i=1}^{n} w_i x_i$ representa la combinación lineal de las entradas ponderadas de los pesos, el término '*b*' se añade para ajustar la salida al modelo y Φ corresponde a la función escalonada de Heaviside (o función escalón unitarios) cuyo valor corresponde a 0 para valores de argumento negativo y 1 para valores de argumento positivo.

Capas de la Red: Las redes neuronales se estructuran en capas, que incluyen una capa de entrada, una o más capas ocultas y una capa de salida. Cada capa oculta extrae características de los datos y las pasa a la siguiente capa.

Entrenamiento: Durante el entrenamiento, se emplean algoritmos de optimización, como el descenso por gradiente, para ajustar los pesos de las conexiones. La función de pérdida *L* (2) mide la diferencia entre las predicciones de la red $\hat{y}_i$ y los valores reales $\hat{y}_i$.

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

El objetivo del entrenamiento es minimizar esta función de pérdida. Los pesos se actualizan usando la regla del descenso por gradiente (3):

$$w_{i+1} = w_i - \eta \frac{\partial L}{\partial w_i} \quad (3)$$

Donde η es la tasa de aprendizaje.

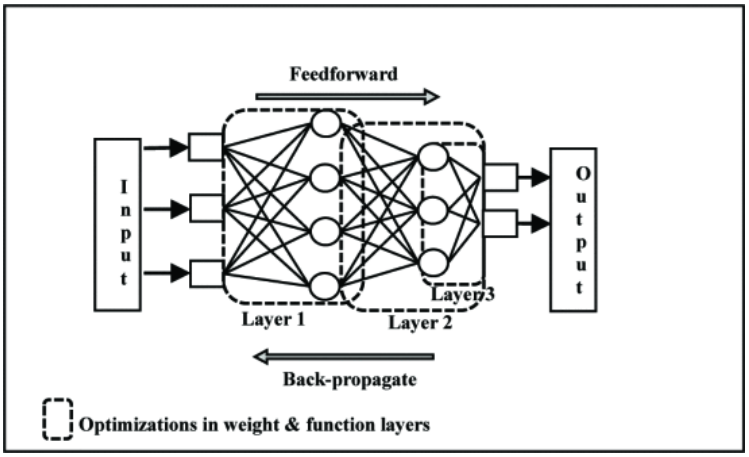


Figura 5: Proceso de aprendizaje de una red neuronal.[15]

2.3.2 Tipos de Redes Neuronales

Existen varios tipos de redes neuronales, cada una diseñada para manejar diferentes tipos de datos y tareas:

1. **Redes Neuronales Feedforward (FNN):** Son las redes neuronales más simples, donde la información se mueve en una sola dirección, desde la capa de entrada hasta la capa de salida.

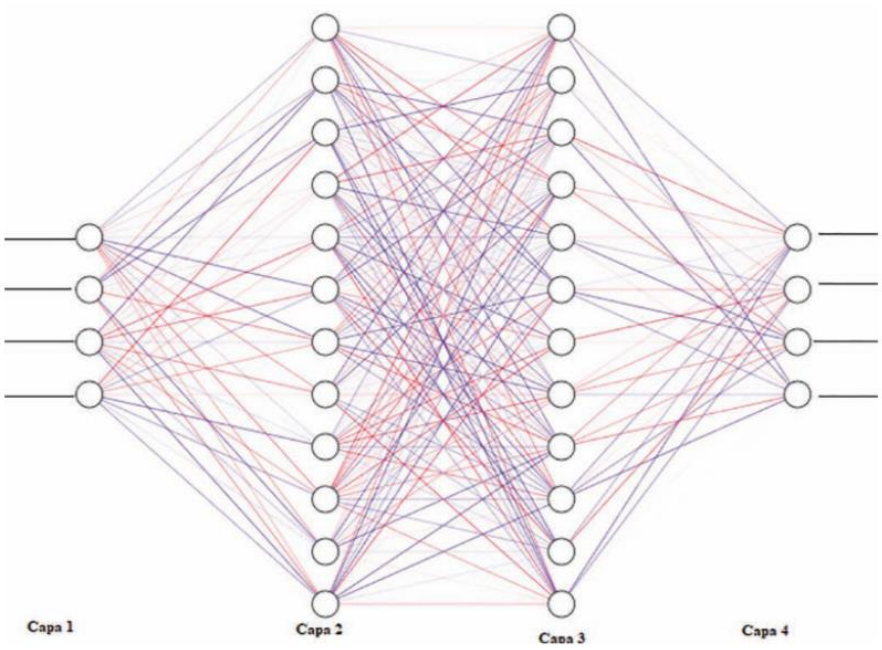


Figura 6: Ejemplo de red neuronal artificial completamente conectada.[16]

2. **Redes Neuronales Recurrentes (RNN):** Estas redes tienen conexiones que forman bucles, permitiendo que la información persista. Son adecuadas para tareas que implican secuencias de datos, como el procesamiento del lenguaje natural.

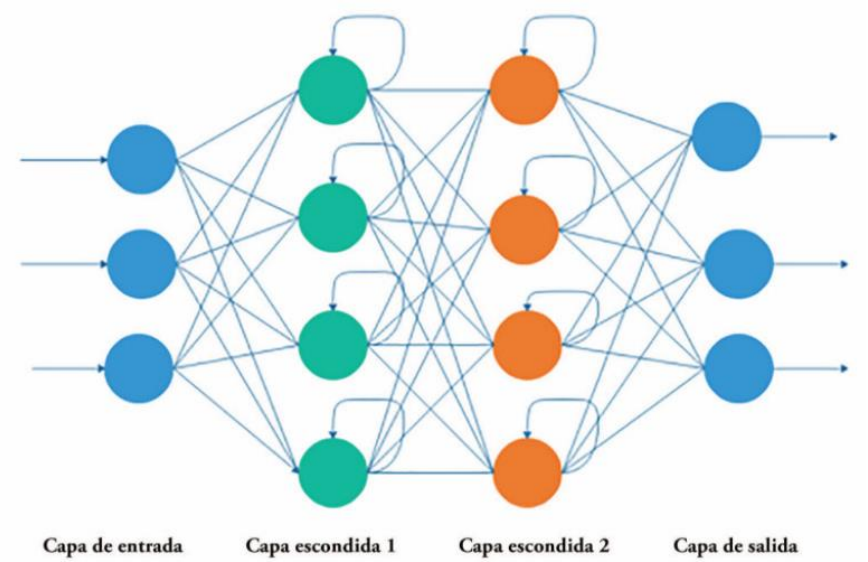


Figura 7: Ejemplo de red neuronal recurrente.[16]

3. **Redes Neuronales Convolucionales (CNN):** Especializadas en el procesamiento de datos en forma de cuadrícula, como imágenes. Utilizan operaciones de convolución para extraer características locales de las entradas.

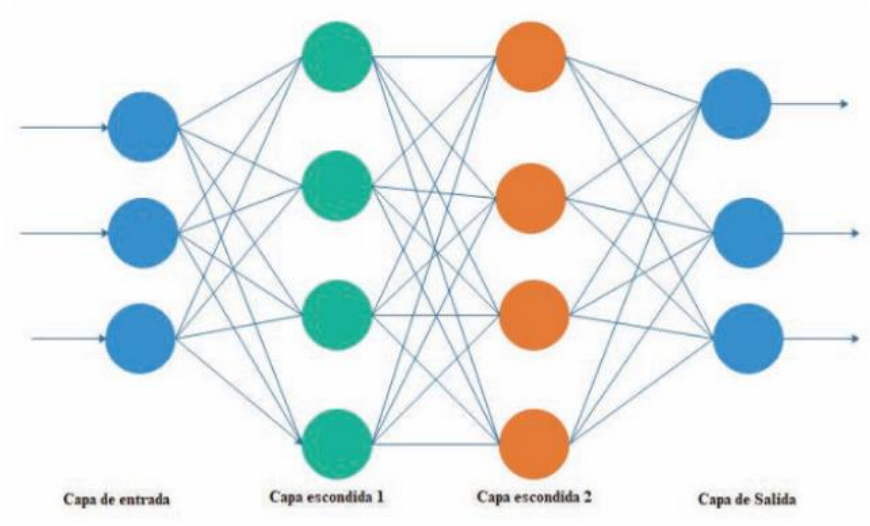


Figura 8: Ejemplo de red neuronal convolucional.[16]

4. **Redes Generativas Antagónicas (GAN)**: Compuestas por dos redes neuronales que compiten entre sí, una generadora y una discriminadora. Son utilizadas para generar datos sintéticos que son similares a los datos reales.

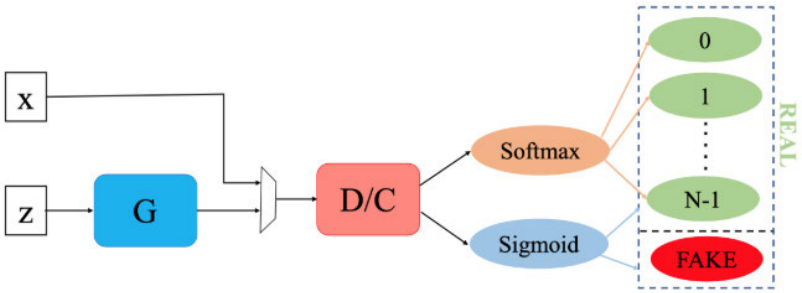


Figura 9: Ejemplo de red neuronal tipo SGAN.[17]

2.3.3 Ventajas y desventajas

2.3.3.1 Ventajas:

- Capacidad de Aprendizaje: Las redes neuronales pueden aprender y modelar relaciones complejas y no lineales en los datos.
- Escalabilidad: Pueden manejar grandes volúmenes de datos y escalarse para aprovechar la potencia de procesamiento de las GPU.
- Versatilidad: Son aplicables a una amplia gama de problemas, desde el reconocimiento de voz hasta la visión por computadora.

2.3.3.2 Desventajas:

- Requisitos Computacionales: El entrenamiento de redes neuronales profundas requiere una gran cantidad de recursos computacionales y tiempo.
- Datos de Entrenamiento: Necesitan grandes cantidades de datos etiquetados para entrenarse efectivamente.
- Interpretable: Las redes neuronales a menudo se consideran "cajas negras" debido a la dificultad para interpretar sus decisiones y el proceso de aprendizaje.

2.4 Redes Neuronales Convolucionales

Las Redes Neuronales Convolucionales (CNN) son una clase especial de redes neuronales desarrolladas para procesar datos que tienen una disposición específica en forma de red, por ejemplo, imágenes. Representan una de las nuevas revoluciones en el campo de la visión artificial y se aplican en diferentes áreas como el reconocimiento de objetos, la segmentación de imágenes y la detección de rostros. En comparación con una red neuronal estándar con capas completamente conectadas, las CNN se componen de capas convolucionales que actúan en una entrada con filtros para lograr un cálculo de características locales. Son muy poderosos en el reconocimiento de patrones visuales.

2.4.1 Conceptos básicos

El proceso de convolución es fundamental para el funcionamiento de las CNN. En una capa convolucional, se aplican múltiples filtros (o kernels) a la imagen de entrada. Cada filtro se desplaza sobre la imagen y calcula el producto punto entre los valores del filtro y la imagen en

cada posición, generando un mapa de características. Estos mapas de características destacan diferentes aspectos de la imagen, como bordes, texturas y patrones más complejos.

A continuación, se muestra la ecuación (4) que representa la convolución discreta matemática formulada de la siguiente manera:

$$(I * K)(x, y) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} I(x + i, y + j)K(i, j)(4)$$

donde I representa la imagen de entrada, K es el filtro, y (x,y) son las coordenadas de la salida. Esta operación es capaz de capturar características espaciales en diferentes niveles de abstracción, dependiendo de la profundidad de la red, obteniendo así mapas de características. Estas características son posibles ubicaciones del filtro con respecto a la imagen original.

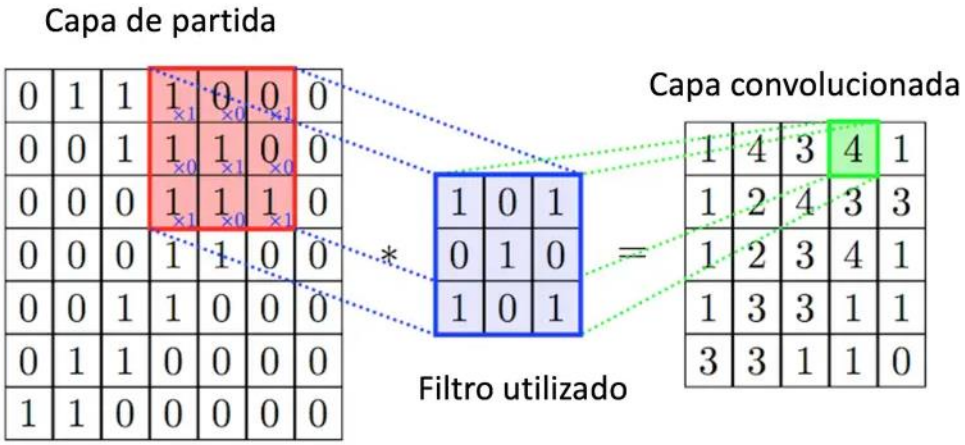


Figura 10: Proceso de convolución.[18]

Una vez que se han generado los mapas de características, se aplica una función de activación para introducir no linealidades en el modelo. La función de activación más comúnmente utilizada es la ReLU (Rectified Linear Unit), que se define como:

$$ReLu(x) = \max(0, x)$$

La ReLU convierte todas las entradas negativas en cero y mantiene las entradas positivas sin cambios, lo que ayuda a la red a aprender características no lineales[19]. Esta no linealidad es crucial para que la red pueda modelar relaciones complejas en los datos.

Después de la activación, las características extraídas pasan a través de capas de pooling, también conocidas como submuestreo o downsampling. El objetivo del pooling es reducir la dimensionalidad de los mapas de características, lo que disminuye la cantidad de parámetros y el costo computacional, y también hace que la red sea más robusta a pequeñas variaciones en la entrada. Las dos técnicas de pooling más comunes son el max pooling y el average pooling. En el max pooling, se selecciona el valor máximo en cada ventana de la matriz de características (8):

$$Y(x,y) = \max\{I(x+i, y+j)\}(8)$$

para i,j∈{0,…,k−1}en una ventana de tamaño k×k. En el average pooling, en lugar de seleccionar el valor máximo, se calcula el promedio de los valores en la ventana.

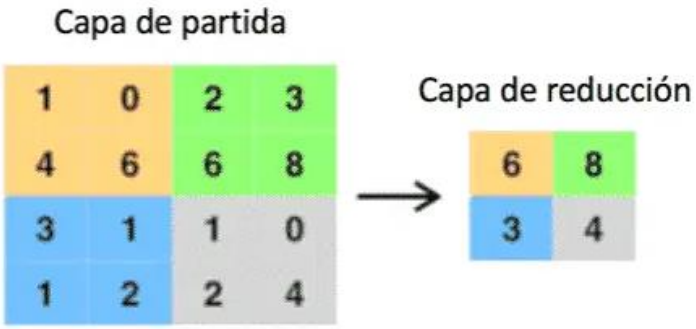


Figura 11: Ejemplo de reducción dimensional.[18]

A medida que los datos pasan por varias capas convolucionales y de pooling, la CNN aprende a capturar características cada vez más complejas y abstractas de la imagen. Las primeras capas pueden detectar bordes simples y texturas, mientras que las capas más profundas pueden capturar patrones más complejos y específicos de alto nivel, como formas y objetos completos.

Al final de la red, las características extraídas se aplanan y se pasan a través de una o más capas completamente conectadas (fully connected layers). En estas capas, cada neurona está conectada a todas las neuronas de la capa anterior, similar a las redes neuronales tradicionales. La ecuación (9) representa la salida final de la red, que suele pasar por una capa softmax, produciendo una distribución de probabilidad sobre las clases de salida:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}} (9)$$

Esta función Softmax transforma un vector z de k números reales en otro vector de k valores que caen dentro del rango [0, 1], de manera que la suma de estos valores es igual a 1. En esencia, estima la distribución de probabilidad de los k números originales. Lo logra calculando la exponencial de cada valor del vector y luego dividiendo cada resultado por la suma de todas las exponenciales, normalizando así los valores para que sumen 1 y formen una distribución de probabilidad.

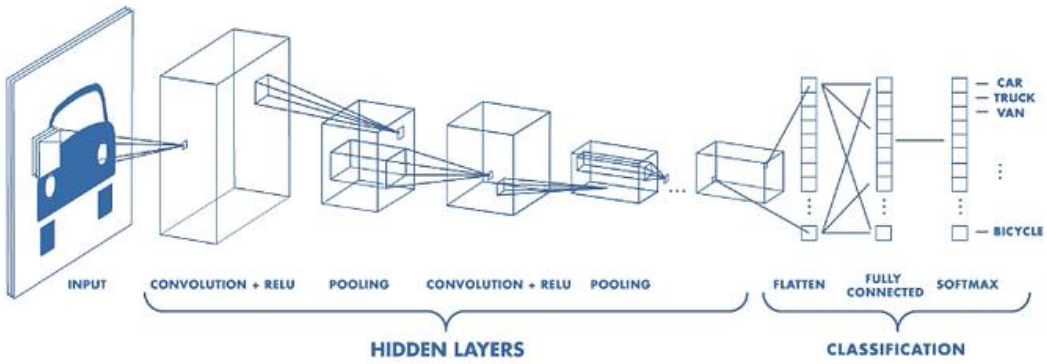


Figura 12: Arquitectura CNN[20]

El proceso general en una CNN puede resumirse en los siguientes pasos: Primero, la imagen de entrada se alimenta a la red. Luego, la imagen pasa a través de múltiples capas convolucionales y de activación, que extraen características esenciales de la imagen. Posteriormente, las características se reducen en dimensionalidad mediante capas de pooling. Después de pasar por varias de estas capas, las características aplanadas se procesan en capas completamente conectadas. Finalmente, la salida se produce, que puede ser una clasificación, una detección de objeto o cualquier otra tarea específica de la aplicación.

2.4.2 Arquitecturas comunes

Las arquitecturas de CNN han evolucionado significativamente desde sus inicios, con varias arquitecturas influyentes que han establecido nuevos estándares en el campo de la visión por computadora. A continuación, se describen algunas de las arquitecturas más comunes y sus variantes, junto con sus características, ventajas y ejemplos de uso.

2.4.2.1 LeNet-5

LeNet-5 fue una de las primeras CNN exitosas, desarrollada en la década de 1990 para el reconocimiento de dígitos escritos a mano en el conjunto de datos MNIST [21]. LeNet-5 consta de dos capas convolucionales seguidas de capas de pooling y tres capas totalmente conectadas, lo que forman un total de siete capas en su arquitectura. Esta arquitectura estableció las bases para el diseño de redes más complejas.

Cada capa convolucional utiliza filtros de 5x5 y es seguida por una capa de submuestreo (pooling) que reduce la dimensionalidad

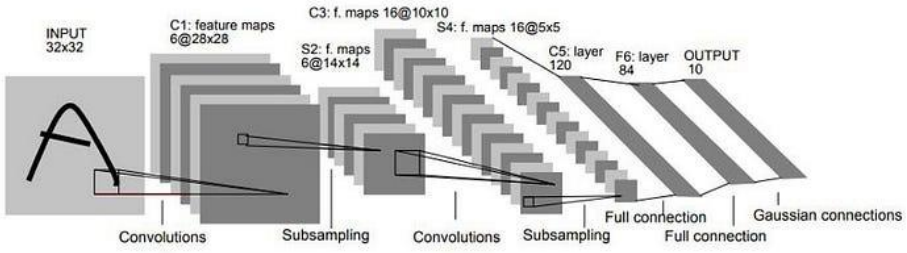


Figura 13: Arquitectura LeNet-5.[21]

2.4.2.2 AlexNet

AlexNet ganó el concurso ImageNet en 2012 y demostró el poder de las redes neuronales profundas y el uso de GPUs para el entrenamiento. AlexNet tiene cinco capas convolucionales, algunas seguidas de capas de pooling, y tres capas totalmente conectadas [22]. Introdujo el uso de la función de activación ReLU y técnicas de regularización como el dropout .

La arquitectura incluye convoluciones con filtros de 11x11, 5x5 y 3x3, con ReLU aplicada después de cada convolución y pooling.

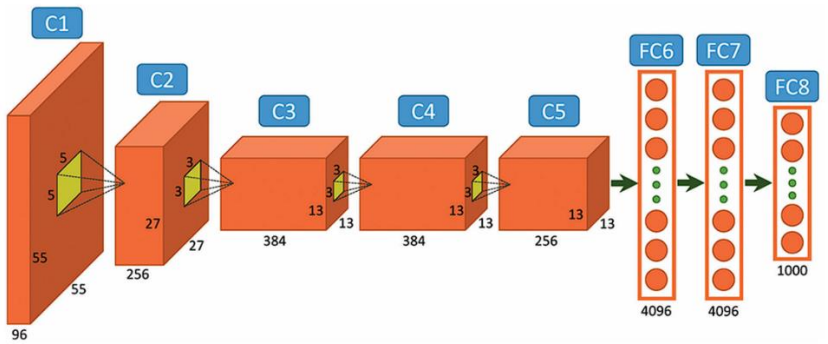


Figura 14: Arquitectura AlexNet.[23]

2.4.2.3 VGG

La arquitectura VGG es conocida por su simplicidad y profundidad . Utiliza exclusivamente capas convolucionales de 3x3 píxeles y tiene un número fijo de filtros en cada capa . Las arquitecturas más comunes son VGG-16 y VGG-19, que tienen 16 y 19 capas de profundidad, respectivamente[24] .

Las capas convolucionales de 3x3 se apilan, seguidas de capas de pooling por cada convolución .

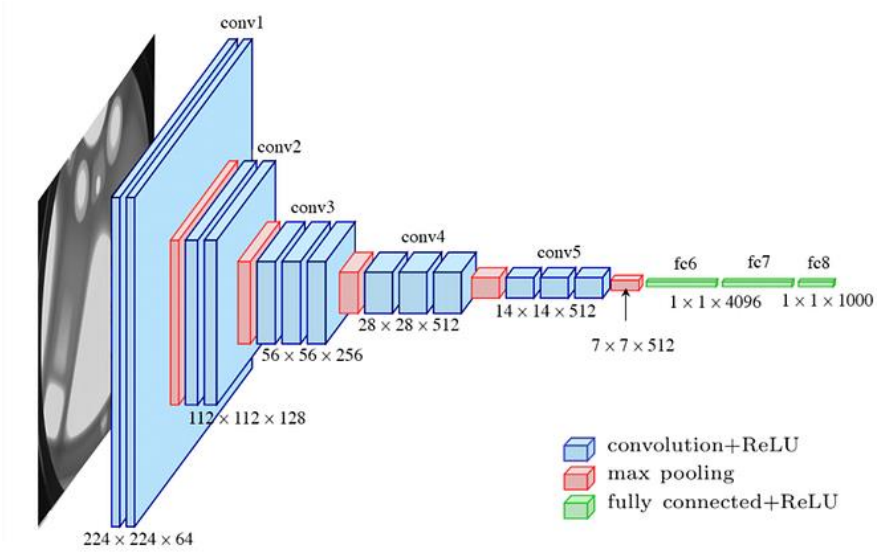


Figura 15: Arquitectura VGG-Net.[24]

2.4.2.4 ResNet

ResNet, o Redes Residuales, aborda el problema de la degradación en redes profundas . Las conexiones residuales permiten el entrenamiento de redes extremadamente profundas, con más de 100 capas, sin sufrir de la degradación del rendimiento [25].

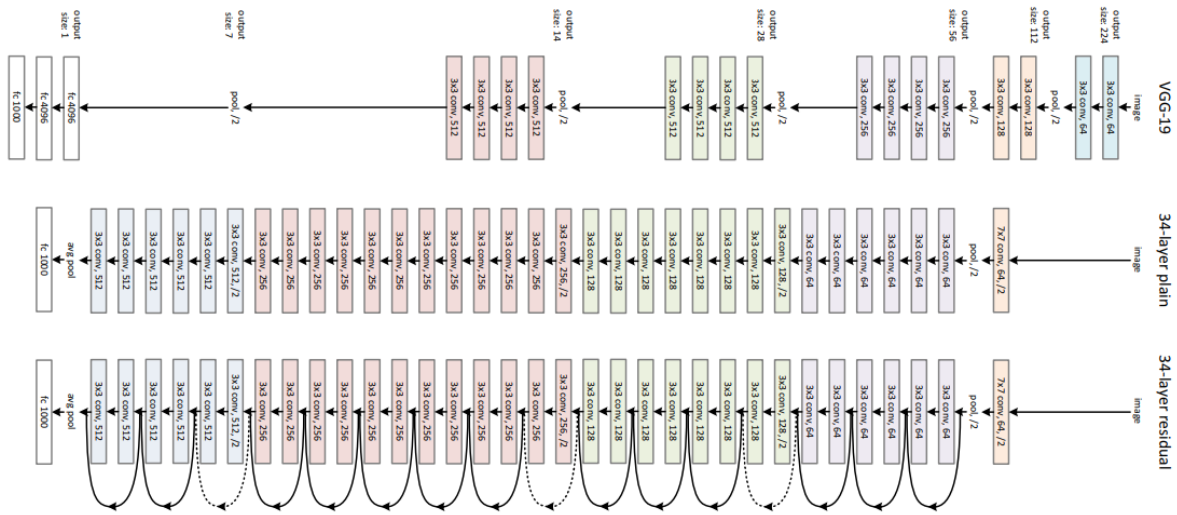


Figura 16: Arquitectura ResNet.[25]

2.4.2.5 Inception (GoogleNet)

Inception, también conocida como GoogleNet, utiliza bloques de Inception que aplican múltiples convoluciones en paralelo, permitiendo una extracción de características más eficiente . Cada bloque Inception realiza convoluciones con diferentes tamaños de filtro (1x1, 3x3, 5x5) y los resultados se concatenan[26] .

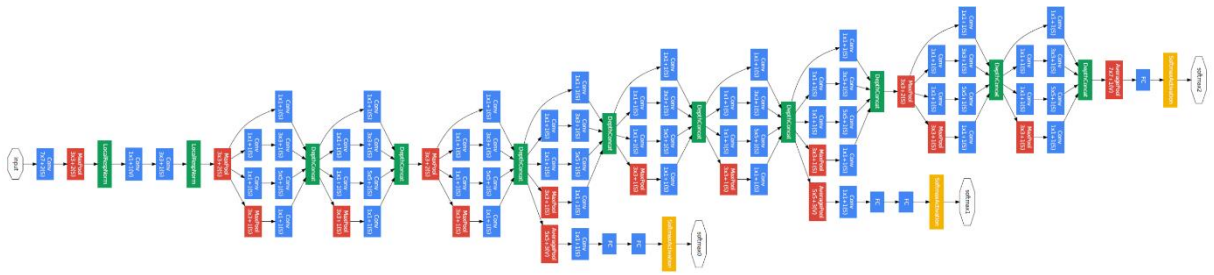


Figura 17: Arquitectura GoogleNet.[26]

2.4.3 Aplicaciones en conteo de objetos

Las CNN son extremadamente efectivas para el conteo de objetos en imágenes. Al segmentar y clasificar cada objeto en una imagen, las CNN pueden contar automáticamente el número de instancias de cada categoría. Esta capacidad es particularmente útil en aplicaciones como el monitoreo de tráfico, la gestión de inventarios y la acuicultura.

1. **Mapas de densidad:** Las CNN pueden generar mapas de densidad que representan la concentración de objetos en diferentes áreas de una imagen. Estos mapas son útiles para el conteo en entornos densos. La siguiente fórmula (10) representa la distribución de combinaciones Gaussianas normales centradas en diferentes puntos $(x_i y_i)$ para una desviación estándar σ.

$$Z(x,y) = \sum_{i=1}^{N} N(x_i, y_i; \sigma)$$

2. **Segmentación de Instancias**: Permite identificar y contar cada instancia individual de un objeto en una imagen, proporcionando un conteo preciso incluso en escenarios complejos.

3. Especificaciones y restricciones de diseño

El objetivo principal del proyecto consiste en el desarrollo de una aplicación basada en redes neuronales capaz de realizar de forma efectiva el conteo de crías de rodaballos a partir de videos e imágenes.

3.1 Especificaciones del diseño

- La aplicación diseñada está basada en un sistema de conteo de larvas de rodaballo utilizando CNN para generar mapas de densidad.
- El sistema fue entrenado con un conjunto de imágenes con zoom que muestran larvas de rodaballo en varios estados de desarrollo.
- El sistema está diseñado para estimar el número de larvas de rodaballo presentes en las imágenes proporcionadas.
- El desarrollo de la aplicación fue realizado en Python.
- El conjunto de datos utilizado para el entrenamiento y la evaluación del sistema es propiedad de GAMMA.

3.2 Restricciones del diseño

- La limitada cantidad de imágenes disponibles para cada estadio de desarrollo de las larvas de rodaballo, lo que puede afectar la variabilidad y la generalización del modelo.
- La base de datos de las imágenes utilizadas corresponde a diferentes empresas del sector acuícola.
- El tiempo y personal disponible para etiquetar las imágenes de larvas de rodaballo fue insuficiente, lo que pudo haber afectado la calidad y precisión de los etiquetados
- Las imágenes fueron etiquetadas por personal no especializado, lo que introduce la posibilidad de errores humanos en el proceso de etiquetado, afectando la exactitud de los datos utilizados para el entrenamiento y evaluación del modelo.

4. Descripción de la solución propuesta

Este proyecto de Trabajo de Fin de Grado (TFG) tiene como objetivo expandir un trabajo previo centrado en el conteo de objetos en imágenes RGB utilizando redes convolucionales con división espacial [29]. Para lograr esto, se empleará un modelo avanzado conocido como SS-DCNet, basado en redes neuronales convolucionales (CNN) que utilizan mapas de densidad, destacándose por su eficacia en el conteo de personas (principalmente). Este proyecto busca adaptar y optimizar dicho modelo para estimar con precisión el número de larvas de rodaballo en diferentes estadios de desarrollo a partir de imágenes con zoom.

El presente TFG también se propone clasificar las larvas de rodaballo según los días de nacimiento, introduciendo un valor sigma variable para mejorar la precisión del modelo. El sistema se entrenará con un conjunto de datos proporcionado por GAMMA, que incluye imágenes etiquetadas de larvas de rodaballo. Este enfoque permitirá evaluar la eficacia del modelo SS-DCNet en acuicultura, ofreciendo una herramienta precisa y no invasiva para el monitoreo y gestión de la biomasa en los tanques de cría.

4.1 Modelo SS-DCNet a utilizar

El modelo SS-DCNet ofrece una solución avanzada y eficaz para el conteo de objetos en imágenes, especialmente en escenarios con alta densidad de objetos. Aborda el desafío de conteo objetos en imágenes densamente pobladas mediante una estrategia de "divide y vencerás" espacial, dividiendo iterativamente las imágenes en subregiones más pequeñas y manejables. Al hacerlo, SS-DCNet mejora significativamente la precisión y eficiencia del conteo, permitiendo un análisis detallado y preciso de las escenas complejas. Esta metodología permite no solo una mejor gestión de los recursos computacionales, sino también una mayor precisión en el conteo al reducir la complejidad de las imágenes originales, tal como se muestra en la figura 19. Este enfoque permite manejar imágenes complejas dividiéndolas iterativamente en subregiones más pequeñas hasta que cada subregión contenga un número manejable de objetos[27]. Esta técnica es fundamental para mejorar la precisión del modelo en el conteo de objetos en escenas densamente pobladas. La propiedad de conteo garantiza que cada subimagen, una vez dividida, contiene un número preciso de objetos, facilitando así el conteo global.

Figura 18: División espacial de imágenes.

La arquitectura del SS-DCNet se basa en un codificador-decodificador tal como se muestra en la **figura 19**. El codificador está basado en la red VGG16, una arquitectura reconocida por su eficacia en la extracción de características detalladas y de alta calidad de las imágenes. El codificador elimina las capas totalmente conectadas de VGG16, conservando únicamente las capas convolucionales para extraer un mapa de características de la imagen de entrada. Estas capas convolucionales utilizan filtros de diferentes tamaños para capturar tanto características finas como globales de la imagen. Por otro lado, el decodificador, similar a U-Net, toma el mapa de características generado por el codificador y lo decodifica para obtener una representación de alta resolución de la imagen original, facilitando la tarea de conteo en subregiones. Utiliza convoluciones transpuestas para aumentar la resolución de los mapas de características y recuperar la estructura espacial de la imagen original. La arquitectura de codificador-decodificador permite una segmentación y conteo precisos de los objetos en las imágenes, lo que es crucial para aplicaciones en las que la precisión del conteo es fundamental.

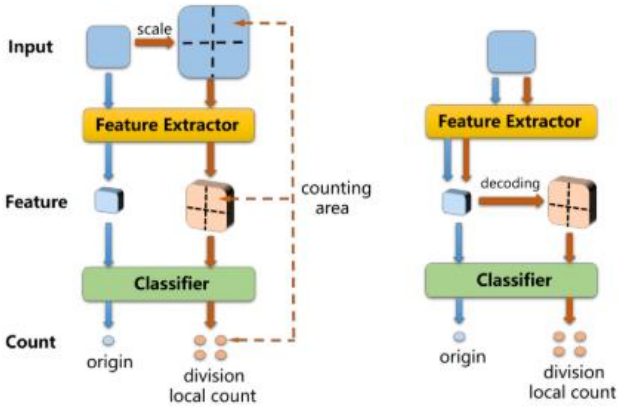


Figura 19: Codificador (izquierda) y decodificador (derecha)

En lugar de predecir valores de conteo continuos, SS-DCNet discretiza los valores de conteo en un conjunto de intervalos y utiliza un clasificador para predecir estos intervalos. Esta técnica transforma el problema de regresión en un problema de clasificación, mejorando la precisión del modelo al simplificar la tarea de predicción. El clasificador asigna etiquetas discretas a cada subimagen, indicando el número de objetos presentes. Si una subimagen contiene demasiados objetos, se subdivide nuevamente y se vuelve a clasificar. Este proceso asegura que el modelo maneje eficientemente tanto imágenes con alta como baja densidad de objetos. La implementación de un clasificador de intervalos de conteo es una innovación clave que permite al SS-DCNet gestionar de manera efectiva la variabilidad en la densidad de objetos, mejorando la precisión y robustez del modelo.

La estrategia de división espacial es esencial para manejar la alta densidad de objetos en las imágenes. El proceso comienza con la división de la imagen de entrada en subregiones de tamaño fijo, típicamente de 64x64 píxeles. Si el número de objetos en una subregión excede un umbral predefinido, esa subregión se subdivide nuevamente en parches más pequeños. Este proceso recursivo continúa hasta que cada subimagen contiene un número de objetos que el clasificador del modelo puede manejar con precisión. La máscara de división suave

asigna valores entre 0 y 1 a cada subregión, indicando la probabilidad de que una subregión necesite ser dividida. Un valor cercano a 1 implica que la subregión debe subdividirse, mientras que un valor cercano a 0 indica que no es necesaria una subdivisión adicional. Esta técnica permite manejar eficientemente imágenes con alta densidad de objetos, asegurando que cada subregión sea adecuadamente analizada. La aplicación de una máscara de división suave mejora la eficiencia del proceso de conteo, permitiendo una mejor gestión de los recursos computacionales y un análisis más preciso de las subregiones densamente pobladas.

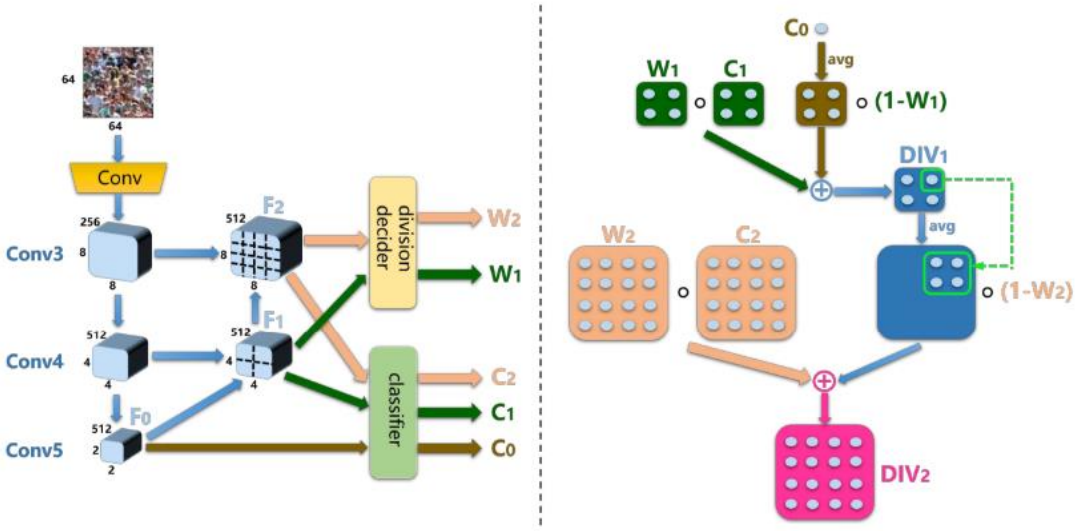


Figura 20: Arquitectura modelo SS-DCNet

La función de pérdida en SS-DCNet incluye varias componentes para optimizar el proceso de entrenamiento. Estas componentes incluyen la pérdida del contador, la pérdida por fusión, la pérdida por división y la pérdida de sobremuestreo. La combinación de estas pérdidas asegura que el modelo no solo aprenda a contar objetos de manera precisa, sino que también mantenga la coherencia y la exactitud en todo el proceso de conteo.

- **Pérdida del contador:** penaliza las diferencias entre el conteo predicho y el conteo real de objetos en las subregiones. Esta pérdida se calcula utilizando el error cuadrático medio (MAE)(11), que mide la diferencia entre los valores predichos y los valores reales.

$$\frac{1}{Z}\sum_{i=1}^{z}\left|{C_i}^{pre}-{C_i}^{gt}\right| (11)$$

  Donde Z es el número de puntos de datos no faltantes, ${C_i}^{pre}$ son las observaciones actuales de las series de tiempo y ${C_i}^{gt}$ es la serie de tiempo estimada o pronosticada.

Pérdida por Fusión: se utiliza para penalizar la incorrecta fusión de parches subdivididos. Esta pérdida asegura que los conteos de objetos en las subregiones se combinen de manera coherente y precisa. La pérdida por fusión se calcula como la

- diferencia entre el conteo total predicho y el conteo total real , sumada sobre todas las subregiones.
- **Perdida por división:** penaliza la subdivisión innecesaria de parches. Esta pérdida asegura que el modelo no realice divisiones adicionales cuando no son necesarias, optimizando así el uso de los recursos computacionales y mejorando la eficiencia del modelo (12).

$$L_{div}^{1} = -\mathbb{I}\{C_0^{gt} > C_{max}\}\, x \log\left(\max(W_1)\right) (12)$$

- **Pérdida por sobremuestreo:** penaliza el sobreajuste del modelo a los datos de entrenamiento. Esta pérdida se calcula como la diferencia entre los valores predichos y los valores reales en las subregiones (13), asegurando que el modelo generalice bien a nuevos datos.

$$U_i^{gt} = \frac{C_i^{gt}}{C_{i-1}^{gt} \otimes 1_{2x2}} (13)$$

A la suma ponderada de todas las anteriores se le llama Función de pérdida total.

En conclusión, el modelo SS-DCNet representa un avance significativo en el conteo de objetos en imágenes. Su estrategia de división espacial, junto con su arquitectura de codificador-decodificador y su clasificador de intervalos de conteo, permite una precisión elevada en el conteo, optimizando el uso de recursos computacionales y mejorando la gestión de escenas densamente pobladas. Por ello, se adaptará el código fuente para la realización de una aplicación de conteo de larvas de rodaballo de diferentes etapas de nacimiento.

4.2 Mapas de densidad

Los mapas de densidad son una herramienta esencial en la visión por computadora para el conteo de objetos, especialmente en escenarios con alta densidad, como el monitoreo de multitudes o el conteo de células en imágenes microscópicas. Esta técnica convierte el problema de contar objetos discretos en un problema de regresión continua[28], donde cada píxel de la imagen se asocia con una densidad estimada que indica la probabilidad de presencia de un objeto en ese punto.

La creación de mapas de densidad generalmente implica los siguientes pasos:

- Anotación de Puntos: Los objetos en la imagen son marcados con puntos. Por ejemplo, en una imagen de una multitud, cada persona se marca con un punto en el centro de su cabeza.
- Generación del Mapa de Densidad: Los puntos anotados se convierten en un mapa de densidad mediante la convolución con un núcleo Gaussiano. Esta convolución difumina cada punto sobre un área, creando una "colina" de densidad que indica la presencia de un objeto. La suma de todas las densidades en el mapa debe coincidir con el número total de objetos anotados. La siguiente ecuación (14) muestra la representación para estimar la densidad:

$$D(x) = \sum_{i=1}^{N} G(x - x_i)(14)$$

Donde D(x) es la densidad en el punto x, N es el número de objetos, x_i representa la posición del objeto i, y G es el núcleo Gaussiano definido por la siguiente ecuación (15):

$$G(x) = \frac{1}{2\pi\sigma^{\sigma}} e^{-\frac{x^2}{2\sigma^2}}$$

El parámetro sigma (σ) en la función Gaussiana es crucial porque determina el ancho del núcleo Gaussiano utilizado para generar el mapa de densidad. La elección de sigma afecta significativamente a la calidad del mapa de densidad y, por ende, a la precisión del conteo de objetos.

Un sigma pequeño genera picos de densidad más estrechos y altos, lo que puede ser útil para imágenes donde los objetos están claramente separados y tienen un tamaño uniforme. Sin embargo, si los objetos están muy cerca, esto puede resultar en subestimación del conteo debido a que los picos no se solapan adecuadamente.

Un sigma grande genera picos de densidad más anchos y bajos, lo que facilita el solapamiento de los picos de densidad cuando los objetos están cerca uno del otro. Esto puede mejorar la precisión del conteo en escenarios densamente poblados, pero puede causar sobreestimación si los objetos están dispersos.

La elección de sigma debe adaptarse a las características específicas de las imágenes y la distribución espacial de los objetos. En el contexto del conteo de larvas de rodaballo, se ha observado que diferentes etapas de desarrollo de las larvas requieren diferentes valores de sigma para obtener un mapa de densidad óptimo.

4.3 Proceso de etiquetado manual

El etiquetado manual de imágenes con zoom de Rodaballos consistirá en nuestra primera etapa para el desarrollo de nuestra aplicación. Este proceso de etiquetado es especialmente importante debido a la implicación que va a tener a lo largo de nuestro trabajo. Dicho proceso consiste en marcar el centro del Rodaballo para tener así el conjunto de coordenadas de todos los rodaballos que forman dicha imagen. La precisión del etiquetado manual mejorará tanto la calidad del modelo de entrenamiento y su validación en el aprendizaje como los mapas de densidad que se generarán a partir de ellos, intentando reflejar de forma fiel y precisa la distribución real de Rodaballos.

4.3.1 Método de etiquetado

El desarrollo del etiquetado de Rodaballos se realiza mediante el *script* de MATLAB TurbotClassify_v1_3_1.mlapp desarrollado por el grupo GAMMA. Dicho script cargará una interfaz de usuario mostrada en la **figura 22**, para mayor comodidad del usuario.

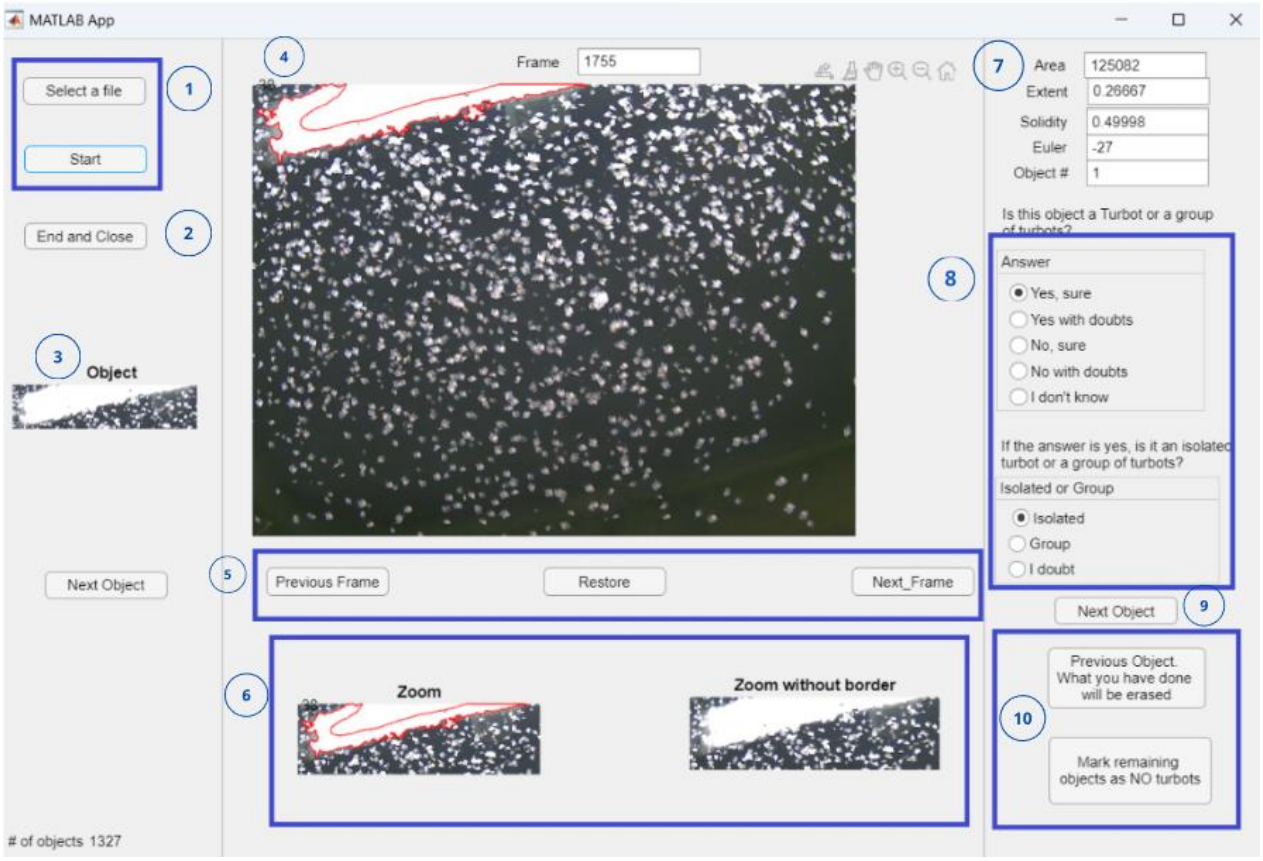


Figura 21: Interfaz etiquetado grupo GAMMA.

El primer paso para el etiquetado de nuestros fotogramas consiste en abrir el vídeo seleccionando el botón “Select a file” ①. Una vez abierto, la aplicación asignará un fotograma específico de dicho video y le daremos a Start ①para comenzar nuestro etiquetado.

Una vez abierto, aparecerá en la interfaz de usuario la imagen a etiquetar. La aplicación utiliza una segmentación de imágenes por grupos de rodaballos de más grande a más pequeño, y podremos observar en todo momento cual es el objeto segmentado actualmente ③ (aumentado en tamaño, para mejorar su visualización) así como su evolución en el tiempo, pudiendo ver justo el instante anterior al frame en el que nos encontramos y el inmediatamente posterior⑤. Esto nos será de gran ayuda, ya que, en muchos grupos de Rodaballos, se superponen unos a otros, y con esta herramienta podremos navegar a través de los frames más cercanos a nuestra imagen para comprobar el movimiento que realizan dichos Rodaballos y poder discriminarlos más fácilmente.

En todo momento podremos observar tanto la segmentación que se está realizando sobre el total de la imagen ④, como el número y características de la segmentación actual ⑦.

El siguiente paso consiste en indicar a la interfaz que el tipo de grupo de rodaballo asignado a la segmentación actual, para ello tendremos tres opciones⑩:

- Isoleted: Se considera que es un Rodaballo aislado.
- Group: Se considera que es un grupo de Rodaballos.

- I doubt: No se sabe si corresponde con un grupo de rodaballos o a un Rodaballo aislado.

Para cualquiera de las tres opciones mencionadas anteriormente, habrá que indicar también el tipo de certeza que tenemos frente a dicha segmentación, para ello habrá que seleccionar entre los cinco tipos de opciones que nos ofrece la aplicación:

- Yes, sure
- Yes with doubts
- No, sure
- No with doubts
- I don't know

Una vez seleccionado el tipo de Rodaballo y el tipo de confianza frente a esa segmentación, debemos seleccionar el botón "Next Object"⑨. Si anteriormente indicamos que nuestra segmentación actual se considera un grupo de Rodaballos, necesitaremos seleccionar el centro de cada uno de los Rodaballos independientes que forman dicho grupo de Rodaballos. Para ello, seleccionaremos con la cruceta ,que aparecerá al apretar el botón "Next Object", el centro de nuestros Rodaballos que aparecen en la ventana "object"③. Una vez finalizada la selección del centro del Rodaballo, pulsaremos el botón "Esc" de nuestro teclado para navegar hacia la siguiente fragmentación.

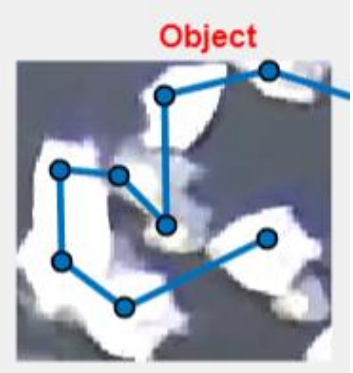


Figura 22: Ejemplo de selección de grupo de Rodaballos

En todo momento podremos volver al frame anterior, seleccionando el botón "Previous Object"⑩.

Al finalizar el etiquetado de todos los objetos segmentados, obtendremos una imagen semejante a la mostrada en la **figura 24**:

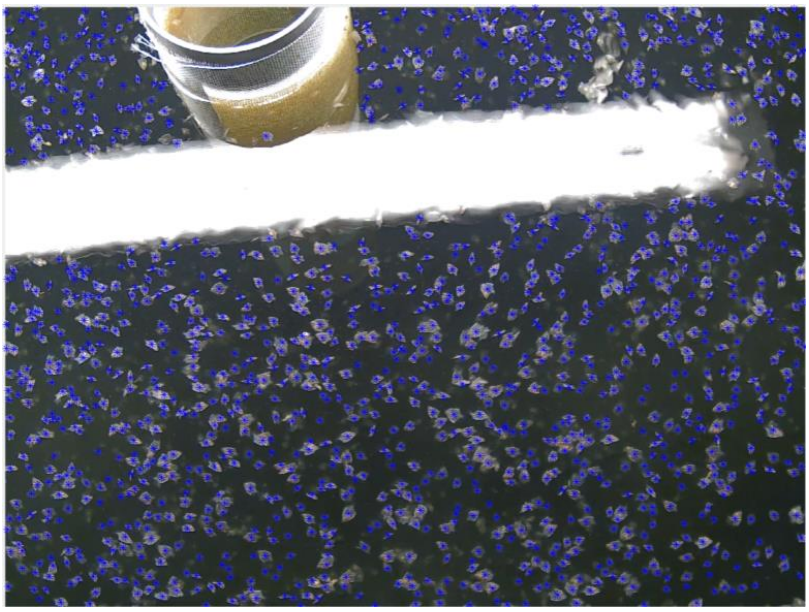


Figura 23: Imágenes etiquetada manualmente

Una vez finalizado todo el etiquetado de los Rodaballos del frame en cuestión, la aplicación generará un archivo .mat en el que incluirá diferentes variables enfocadas a su posterior uso para creación de mapas de densidad y entrenamientos de nuestras redes neuronales, tal como se aprecia en la **figura 25**.

Import	Name	Size	Bytes	Class
☑	BS	371x1	646808	cell
☑	LS	1920x25...	39321600	double
☑	count...	1x1	8	double
☑	iFram...	1x1	8	double
☑	mrPr...	11x371	32648	double
☑	nS	1x1	8	double
☑	turbo...	298x3	7152	double

Figura 24: Variables obtenidas a partir del etiquetado manual.

Las variables almacenadas son las siguientes:

- BS: Define el contorno de los objetos segmentados. Cada entrada consta de una matriz con dos columnas con las coordenadas X e Y de los puntos del borde.
- LS: Indica los píxeles formados por cada objeto.
- countObjectS: Contiene el ultimo objeto etiquetado, con el fin de continuar el etiquetado en caso de cerrar la aplicación.
- iFrameS: Número del fotograma del que se está etiquetando.
- mrPropsS: Contiene información sobre cada objeto etiquetado, incluyendo el número asignado a dicho objeto, sus características morfológicas y la certeza de la identificaciñon del objeto.
- nS: Indica el número total de objetos segmentados a etiquetar.

- turbotCentroidS: Contiene una matriz de 3 columnas que contienen información sobre cada rodaballo de la imagen.
    - Columna 1: Número del objeto al que corresponde
    - Columna 2: Coordenada X del centroide del Rodaballo
    - Columna 3: Coordenada Y del centroide del Rodaballo.

4.4 Preparación de los datos

Los conjuntos de datos que se utilizaron para el entrenamiento de nuestra aplicación consisten en tres grandes divisiones, creadas en función del estado de la larva. Para ello obtendremos conjuntos de datos para larvas comprendidas entre 23 y 25 días de nacimiento, conjunto de datos para larvas de día 26 de nacimiento y conjunto de datos para larvas comprendidas entre 27 y 29 días de nacimiento. Estos conjuntos de datos corresponden a imágenes de una resolución de 2560x192 obtenidas a partir de los videos con zoom del grupo GAMMA. Estos videos no captan el total de Rodaballos en los tanques, debido a que se generaron mediante una cámara con zoom, con lo que no se podrá obtener la totalidad de densidad de Rodaballos en cada tanque, pero si una estimación de la densidad por cada fotograma obtenido de los videos.

Figura 25: Imagen con zoom del conjunto de datos de días 23-25.

El total de estos conjuntos de imágenes separados por el estado de la larva es reducido. Esto se debe al limitado conjunto de imágenes etiquetadas por días de nacimiento. El total de imágenes etiquetadas utilizadas son las siguientes:

- Días de nacimiento comprendidos entre el 23 y 25: 14 imágenes
    - 11 imágenes para entrenamiento
    - 3 imágenes para validación
- Días de nacimiento numero 26: 10 imágenes
    - 8 imágenes para entrenamiento
    - 2 imágenes para validación
- Días de nacimiento comprendidos entre el 27 y 29: 10 imágenes

  - 8 imágenes para entrenamiento
  - 2 imágenes para validación
- Días de nacimiento 25-26 (completo): 174 imágenes
  - 139 imágenes para entrenamiento
  - 35 imágenes para validación

Con el fin de reducir la cantidad de submuestreos realizados de cuadrados de 64x64 (tal como se incida en el apartado 4.1), se ha implementado un script de Matlab llamado "createDataset.m" capaz de generar nuestros conjuntos de datos con un factor de reducción del 2.5 con respecto a la imagen original, obteniendo así imágenes cuya resolución es de 1024x768. Dicho script proporcionara una repartición de los datos asignando, de forma completamente aleatoria, el 80% de ellos para el entrenamiento de nuestra red neuronal y el 20% restantes para test.

El conjunto de datos utilizado para el día 26 "completo" se ha propuesto para comprobar la diferencia de error entre un set compuesto de un mayor número de muestras de imágenes etiquetadas frente a un set compuesto por un número menor de imágenes etiquetadas.

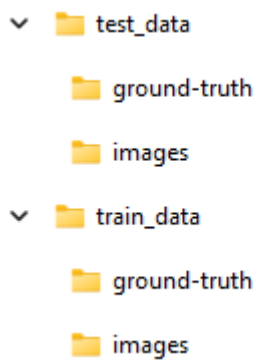


Figura 26: Sistema de carpetas de conjunto de datos.

La **figura 27** muestra la división en carpetas de los datos generados. Para cada grupo de datos (entrenamiento y test) se generan dos carpetas diferentes:

- Ground-truth: Está compuesto de archivos .m que contienen información de las coordenadas de los centroides de Rodaballos para cada imagen asignada a ese conjunto de datos. Dichos archivos tendrán el siguiente formato de nombre: GT_IMG_*, donde * corresponde al número de imagen para ese conjunto de datos.
- Imágenes: Está compuesto de las imágenes con zoom de Rodaballos, tanto para el conjunto de imágenes de entrenamiento como para el conjunto de imágenes de test.

4.5 Parámetros de la aplicación

4.5.1 Desviación estándar

Para la realización del proyecto se ha utilizado en valor variable para la desviación estándar (sigma σ) en la creación de los mapas de densidad para cada conjunto de datos. Esta

desviación estándar va a variar en función del día de nacimiento correspondiente a la larva del Rodaballo.

Para el cálculo de la desviación estándar variable, se suele utilizar el método del cálculo basado en la distancia media a los vecinos más cercanos. Para este proyecto, encontramos la dificultad de calcular la media de los vecinos más cercanos, ya que las imágenes de las que disponemos para los diferentes conjuntos de datos no están distribuidas de forma uniforme a lo largo de toda la imagen, tal como se aprecia en la **figura 28.**

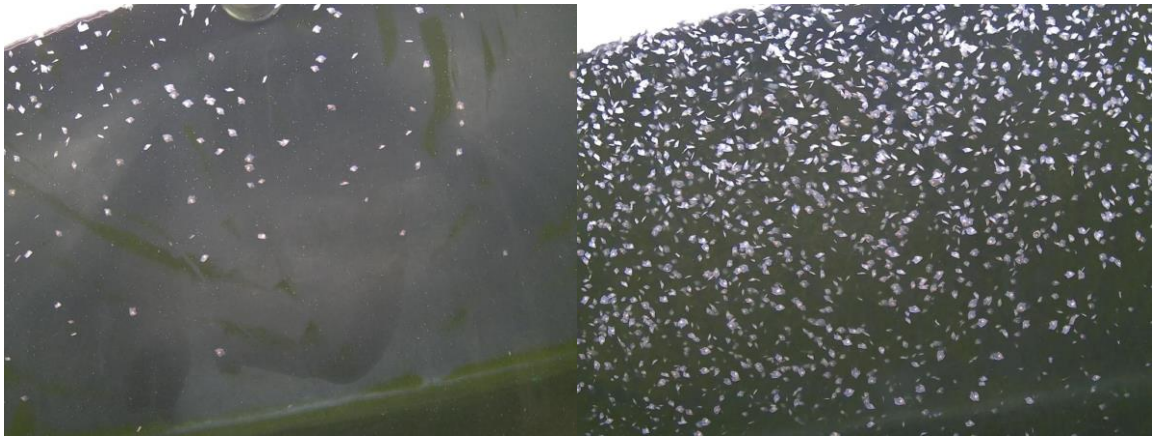


Figura 27: Diferencia de densidades en conjunto de datos día 23-25

Por ello, dicha elección del parámetro sigma (σ) en la generación de mapas de densidad es importante para la precisión del conteo de objetos en imágenes, especialmente cuando las larvas de rodaballo no están distribuidas de manera uniforme. Un sigma pequeño genera picos de densidad más estrechos y altos, lo que es beneficioso en áreas donde las larvas están claramente separadas, permitiendo una identificación precisa de cada una. Sin embargo, en áreas donde las larvas están muy juntas, esto puede resultar en una subestimación del conteo, ya que los picos de densidad pueden no solaparse adecuadamente.

Por otro lado, una sigma grande genera picos de densidad más anchos y bajos, lo que facilita el solapamiento de los picos en áreas densamente pobladas y mejora la precisión del conteo en estas zonas. No obstante, en áreas donde las larvas están claramente separadas, una sigma grande puede causar una sobreestimación del conteo, al extenderse demasiado los picos de densidad y crear la ilusión de que hay más larvas de las que realmente están presentes. Además, una sigma grande puede perder detalles finos en la distribución de las larvas, reduciendo la capacidad de detectar pequeñas variaciones en la densidad. Uno de los objetivos a cumplir a futuro consiste en la adaptación de sigma al tamaño de las larvas de rodaballo.

En la **figura 29** podemos observar la diferencia de usar diferentes sigmas para una única imagen de entrenamiento para el estado de larva comprendido entre 23 y 25 días de nacimiento.

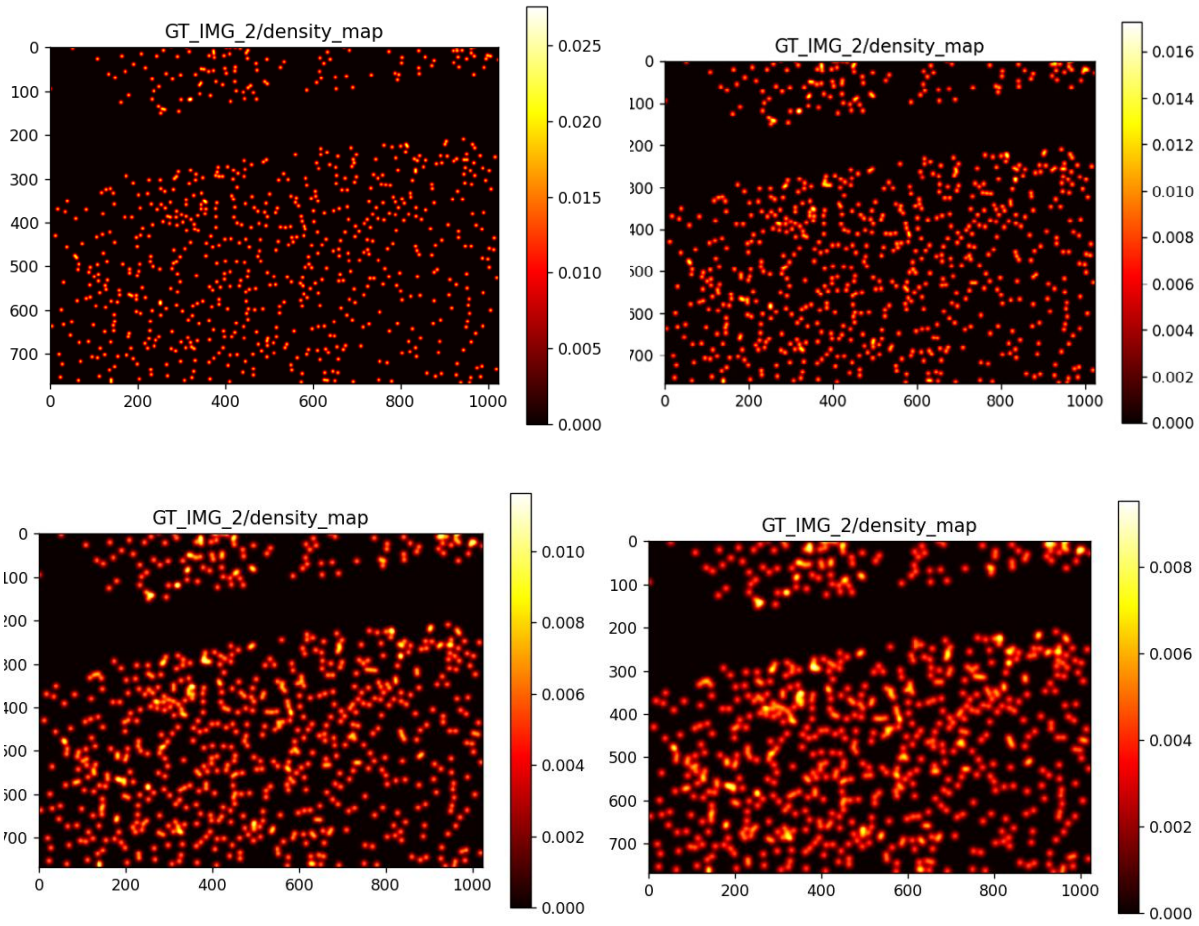


Figura 28: Mapas de densidad para imagen 2 del conjunto de entrenamiento de días 23-25. Sigma 3 (arriba izquierda), sigma 4 (arriba derecha), sigma 5 (abajo izquierda), sigma 6 (abajo derecha).

4.5.2 Dimensión del clasificador

Partiendo de los resultados obtenidos en el anterior proyecto de conteo de rodaballos mediante mapas de densidad, se estipula una dimensión del clasificador correspondiente al valor 5 rodaballos por cuadrado de 64x64 en las imágenes de 1024x768 [29]. Se ha tomado la decisión de establecer una dimensión del clasificador fija para cada grupo de conjunto de datos, ya que la finalidad es ver la varianza del conteo a partir de la variación del parámetro sigma. El proceso de muestreo de rodaballos mediante dimensión del clasificador variable se dejará pendiente para futuros proyectos.

5. Resultados

Para cada conjunto de datos nombrado en el apartado 4.4, se ha utilizado un clasificador $C_{max} = 5$. Adicionalmente, y a diferencia de resultados anteriores, se ha estipulado un numero de reiteraciones de entrenamiento mayor (siendo este de 1000), para compensar la exactitud con el escaso número de fotos etiquetadas usadas para el cálculo de los siguientes resultados. Previamente al entrenamiento de los conjuntos de datos reducidos, se ha realizado el entrenamiento del modelo con el conjunto de datos completo para el día 25-26 usado en el anterior proyecto [29]. Esto se debe a que se ha tenido que utilizar el código fuente [27] debido a incompatibilidades de versiones anteriores en Python.

Para poder cuantificar los errores de predicción del modelo, se utilizaron las siguientes métricas:

- **Mean Absolute Error (MAE)**
  El Mean Absolute Error (MAE) es una métrica utilizada para medir la precisión de un modelo de predicción. Representa el promedio de los errores absolutos entre las predicciones y los valores reales. Un error absoluto es la diferencia absoluta entre el valor predicho y el valor real [30] [31].

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

- **Mean Squared Error (MSE)**
  El Mean Squared Error (MSE) es otra métrica para evaluar la precisión de un modelo de predicción. Calcula el promedio de los cuadrados de los errores, es decir, la diferencia cuadrada entre los valores predichos y los valores reales. MSE penaliza más los errores grandes que los pequeños debido al cuadrado de las diferencias [31] [30].

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

- **Mean Absolute Percentage Error (MAPE)**
  El Mean Absolute Percentage Error (MAPE) mide la precisión de un modelo de predicción en términos porcentuales. Es el promedio de los errores porcentuales absolutos entre los valores predichos y los valores reales. MAPE es útil porque es fácil de interpretar y comunicar, ya que expresa el error como un porcentaje de los valores reales [32].

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right|$$

5.1 Conjunto de datos día 26 completo.

Con el fin de comprobar la resolución de los resultados obtenidos con la modificación del código fuente frente al código usado en proyectos anteriores, se ha realizado la medida de los errores del conjunto de datos original de 25-26 días compuesto por 139 imágenes para el entrenamiento del modelo y 35 imágenes para validación. Con este set de datos, la comparación con trabajos anteriores son los siguientes:

Tabla 1: Resultados proyecto anterior [29].

Valor de σ	MAE	MSE	MAPE
3	9,00	18,82	3,52%
6	11,66	19,46	4,04%
9	11,05	19,22	3,56%
12	9,66	18,20	3,48%

Tabla 2: Resultados de código fuente modificado [27].

Valor de σ	MAE	MSE	MAPE
3	7,83	9,51	4,48%
6	4,50	8,12	0,33%
9	5,21	7,63	1,14%
12	5,47	7,58	1.89%

Tal como podemos observar, la modificación del código fuente resulta de gran utilidad para los conjuntos de datos creados por GAMMA, ya que mejora visiblemente los errores producidos por nuestro modelo de predicción, siendo 6 el valor de sigma que arroja mejor resultado (con un MAPE del 0.33%) frente al valor 12 de sigma utilizado en el modelo del proyecto anterior [29] cuyo error es del 3,48%.

5.2 Conjunto de datos días 23-25

A continuación, se muestra una tabla resumen de los resultados al variar el valor de sigma. Se puede observar que el error relativo aumenta a medida que incrementa el valor de la desviación estándar (σ). Específicamente, el MAE pasa de 77,86 a 103,93, el MSE de 91,63 a 121,55 y el MAPE de 14,48% a 19,20% cuando sigma varía de 3 a 6. Estos resultados indican que un mayor valor de sigma conduce a un incremento en todos los tipos de error considerados. Por lo tanto, se ha decidido desestimar el uso de valores de sigma más grandes, como sigma 12, para evitar un incremento en los errores y asegurar una mayor precisión en las estimaciones.

Tabla 3: Resultados conjunto de entrenamiento días 23/25

Valor de σ	MAE	MSE	MAPE
3	77,86	91,63	14,48%
4	89,26	108,1	17,48%
5	92,62	110,35	18,00%
6	103,93	121,55	19,20%

En los apartados siguientes, se desglosan detalladamente los resultados obtenidos para cada experimento realizado. Para cada uno, se presenta un ejemplo ilustrativo de una predicción generada por el modelo. Además, se analiza la evolución de los errores MAE, MSE y MAPE a lo largo del proceso de entrenamiento, mostrando gráficos y tablas que representen visualmente cómo varían estos errores a medida que avanza el entrenamiento del modelo. Esto permitirá observar tendencias y posibles puntos de mejora en la precisión del modelo. Se incluirán comparaciones entre diferentes configuraciones del modelo, destacando el valor de sigma.

Desviación estándar 3

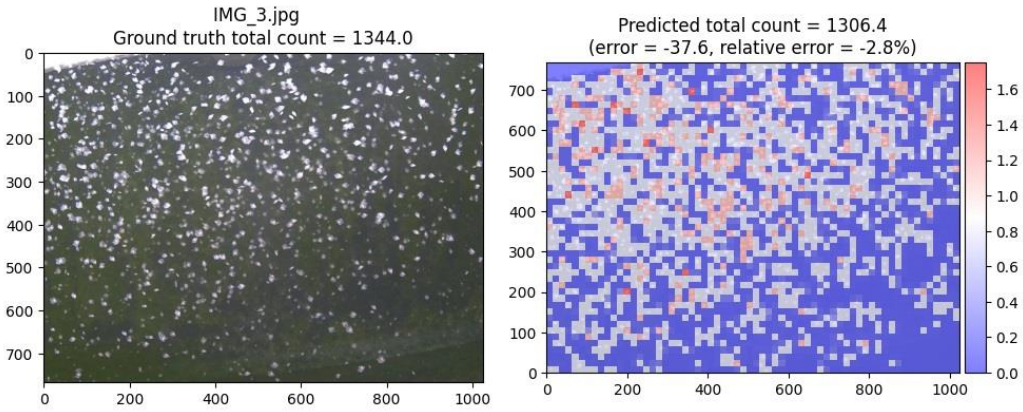


Figura 29: Ejemplo resultado para conjunto de datos de días 23/25 con desviación estándar 3

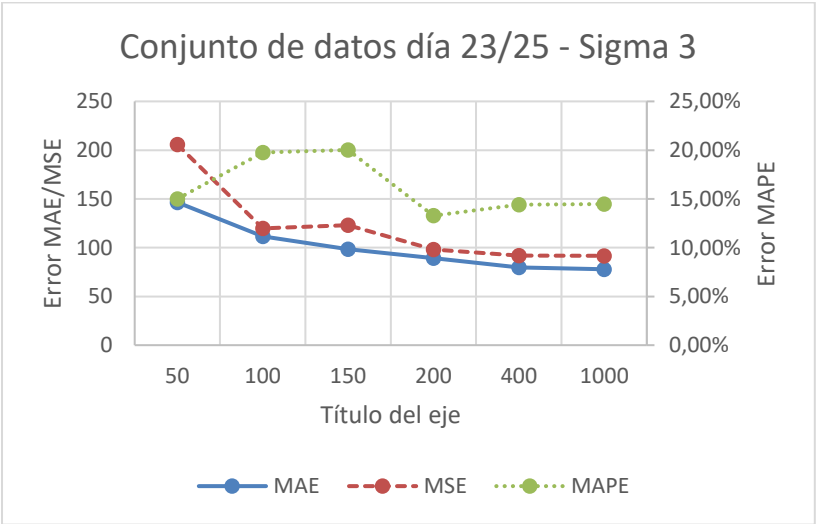


Figura 30: Proceso de los errores durante el entrenamiento para conjunto de datos de días 23/25 con desviación estándar 3

Desviación estándar 4

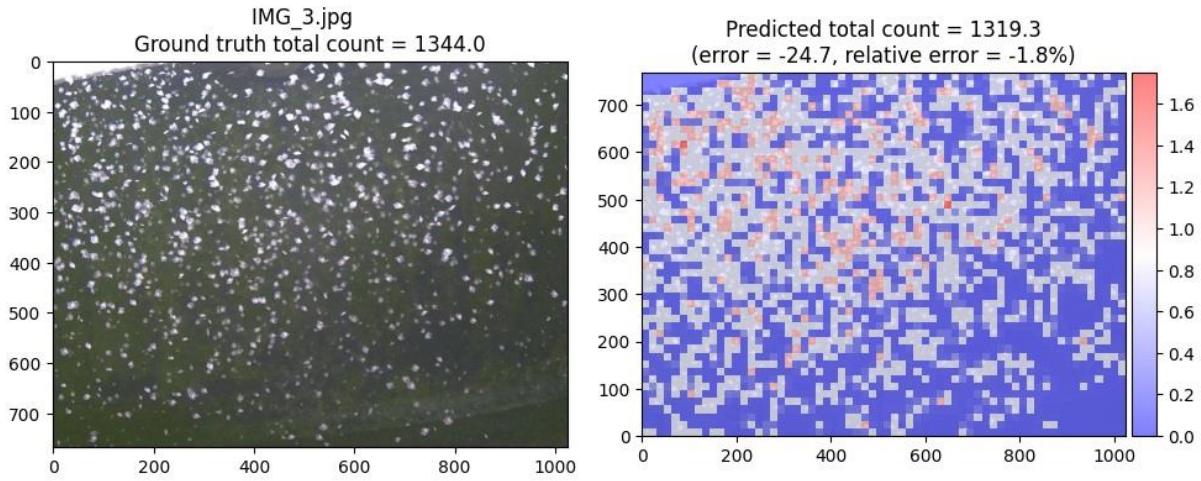


Figura 31:Ejemplo resultado para conjunto de datos de días 23/25 con desviación estándar 4

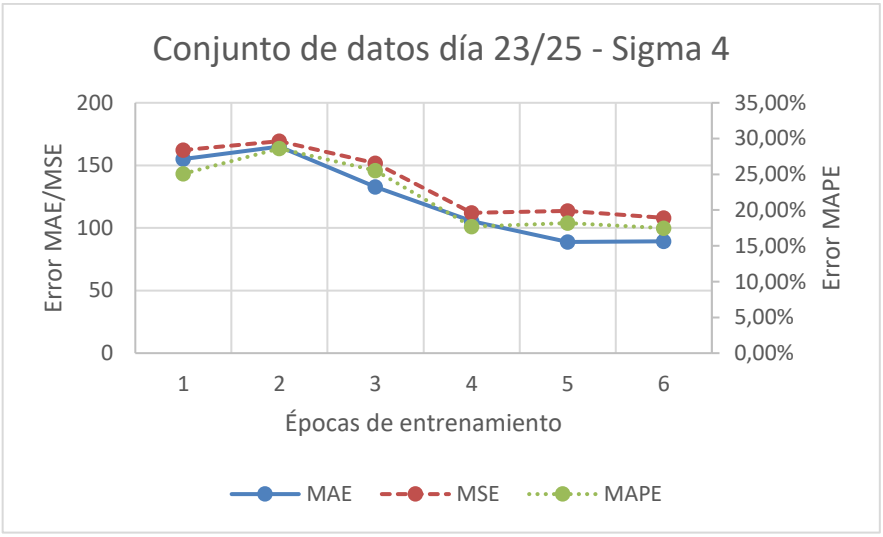


Figura 32: Proceso de los errores durante el entrenamiento para conjunto de datos de días 23/25 con desviación estándar 4

Desviación estándar 5

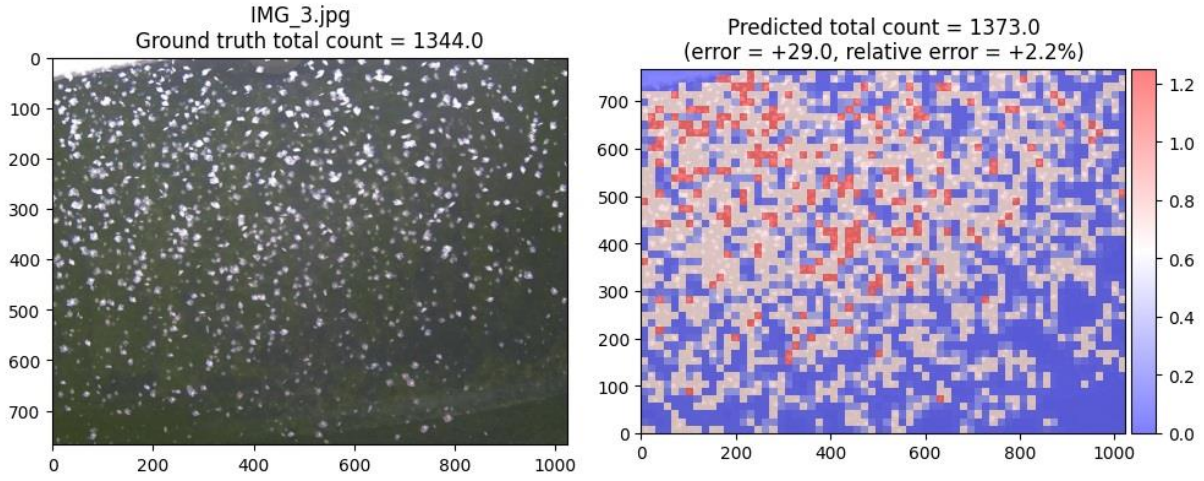


Figura 33: Ejemplo resultado para conjunto de datos de días 23/25 con desviación estándar 5

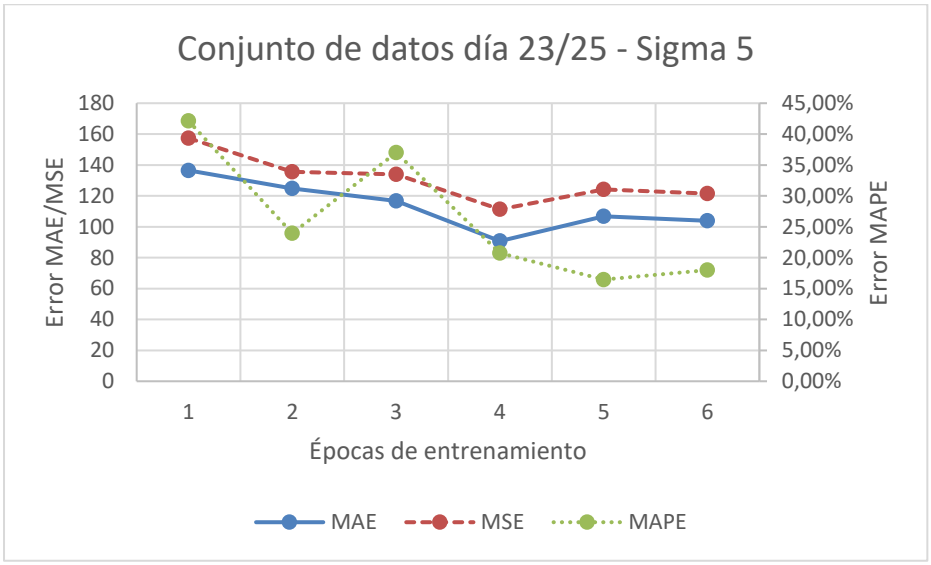


Figura 34: Proceso de los errores durante el entrenamiento para conjunto de datos de días 23/25 con desviación estándar 5

Desviación estándar 6

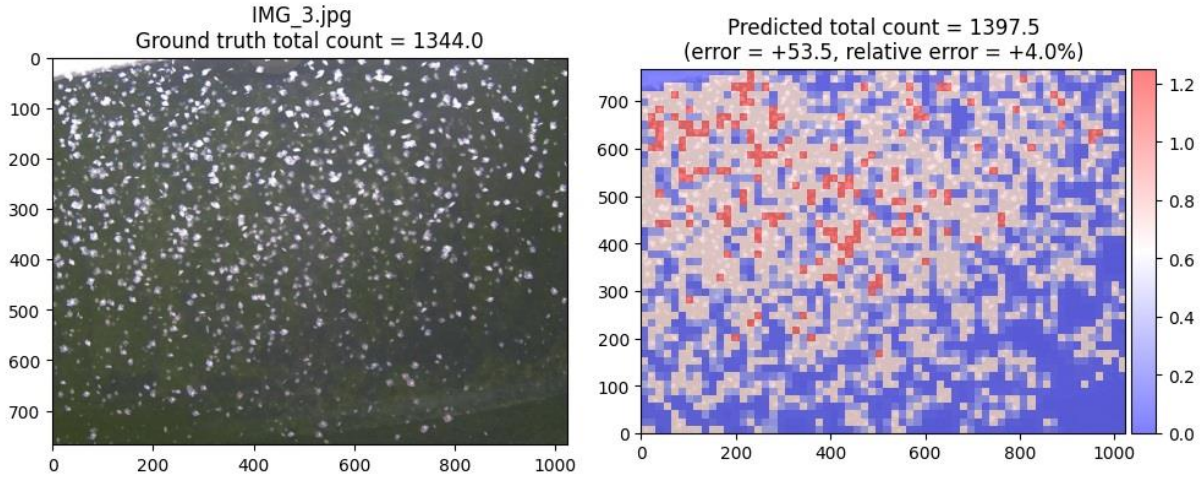


Figura 35: Ejemplo resultado para conjunto de datos de días 23/25 con desviación estándar 6

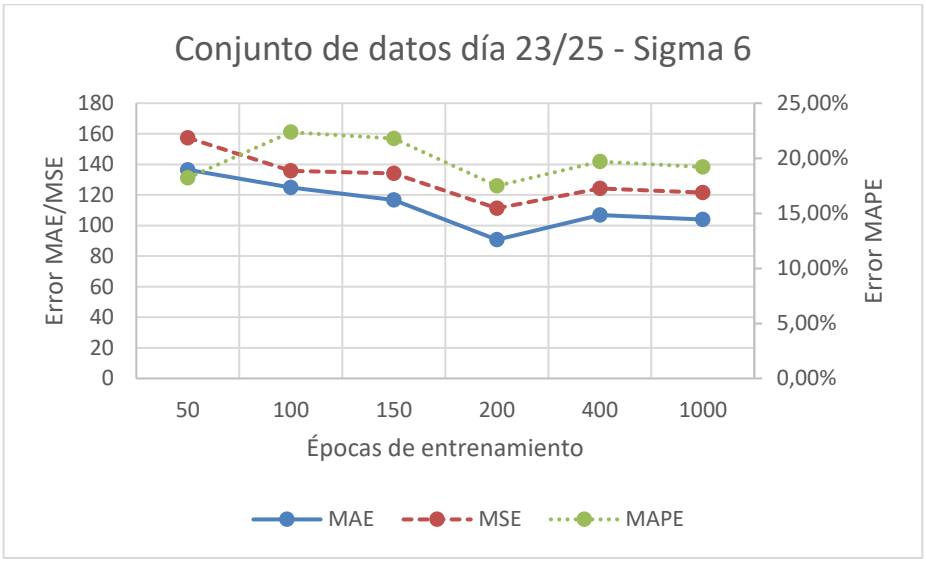


Figura 36: Proceso de los errores durante el entrenamiento para conjunto de datos de días 23/25 con desviación estándar 6

5.3 Conjunto de datos día 26

A continuación, se muestra una tabla resumen de los resultados obtenidos al variar el valor de sigma para las larvas nacidas el día 26. En esta tabla, se presentan los errores MAE, MSE y MAPE para diferentes valores de sigma.

Tabla 4: Resultados conjunto de entrenamiento día 26

Valor de σ	MAE	MSE	MAPE
3	86,41	86,9	6,09%
4	135	135,14	9,85%
5	165,79	165,79	11,96%
6	176,24	176,83	12,49%

Se puede observar que el error relativo aumenta a medida que incrementa el valor de la desviación estándar. En particular, el MAE pasa de 86,41 a 176,24, el MSE de 86,90 a 176,83, y el MAPE de 6,09% a 12,49% al variar σ de 3 a 6. Estos resultados indican que mayores valores de σ conducen a un incremento significativo en todos los tipos de error medidos.

En comparación con los datos obtenidos para las larvas nacidas entre los días 23 y 25, se observa una tendencia similar donde los errores aumentan con valores de σ más altos. Esto refuerza la decisión de no utilizar valores grandes de σ, ya que incrementan la imprecisión en las estimaciones. La selección de una σ más pequeña parece ser más efectiva para mantener bajos los errores y mejorar la precisión de las predicciones en el conteo de larvas, siendo la desviación estándar de 3 la más oportuna para dicho conjunto de datos.

Desviación estándar 3

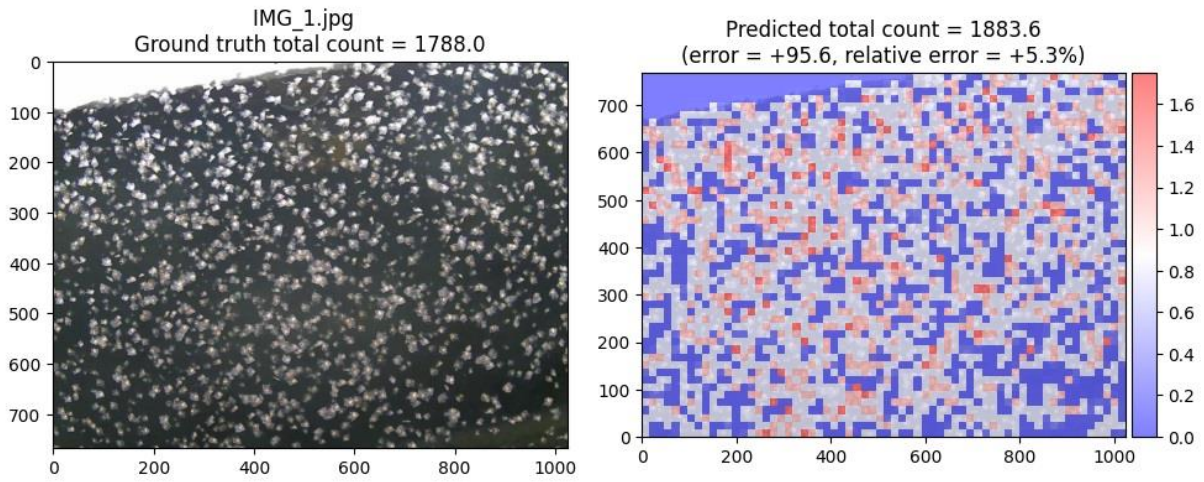


Figura 37: Ejemplo resultado para conjunto de datos del día 26 con desviación estándar 3

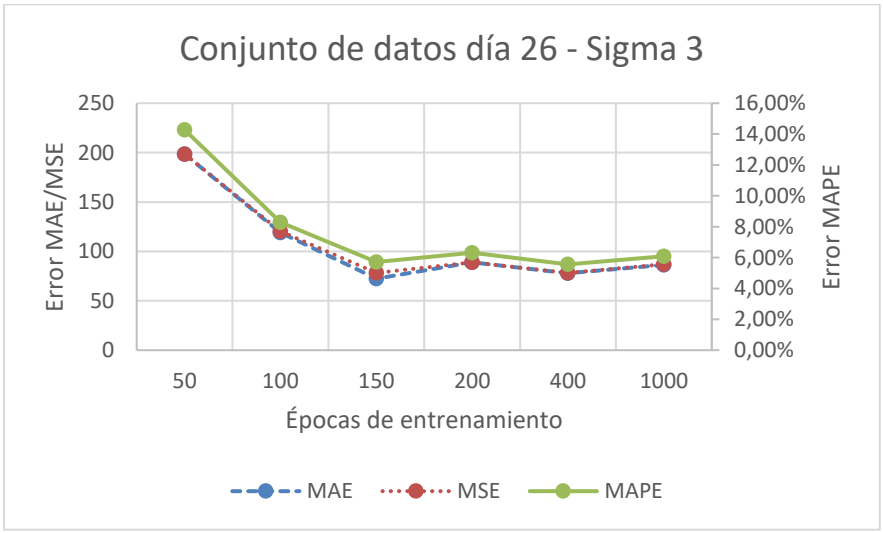


Figura 38: Proceso de los errores durante el entrenamiento para conjunto de datos del día 26 con desviación estándar 3

Desviación estándar 4

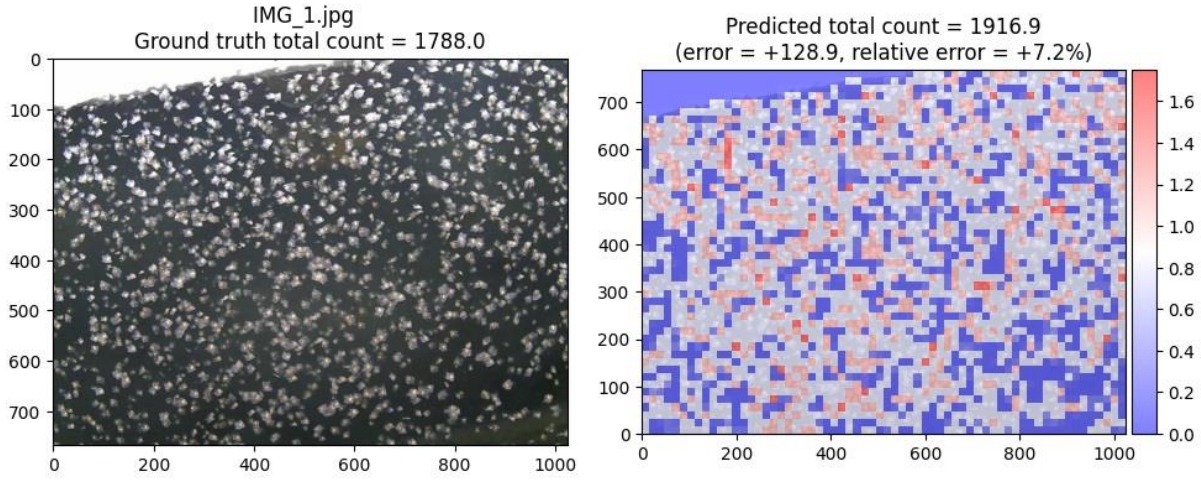


Figura 39: Ejemplo resultado para conjunto de datos del día 26 con desviación estándar 4

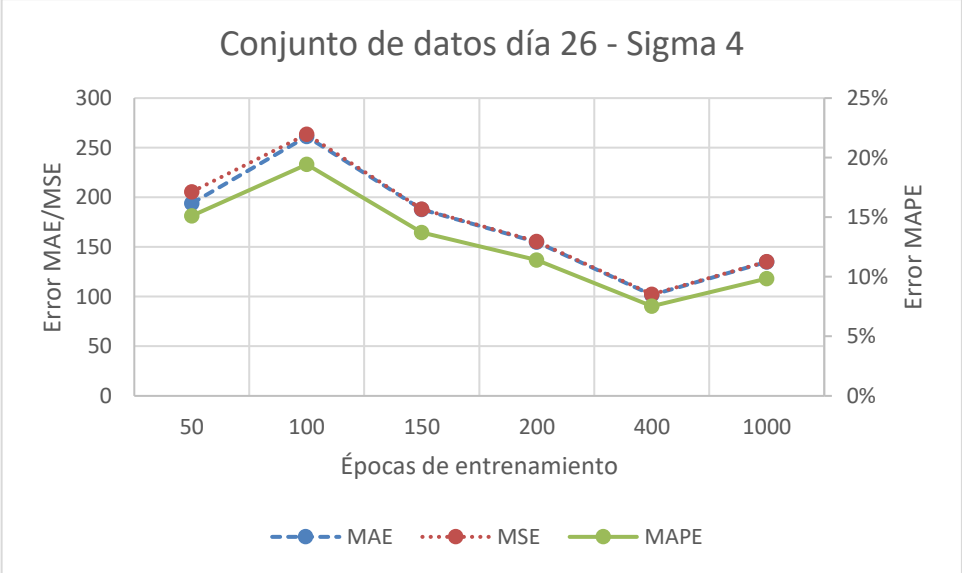


Figura 40: Proceso de los errores durante el entrenamiento para conjunto de datos del día 26 con desviación estándar 4

Desviación estándar 5

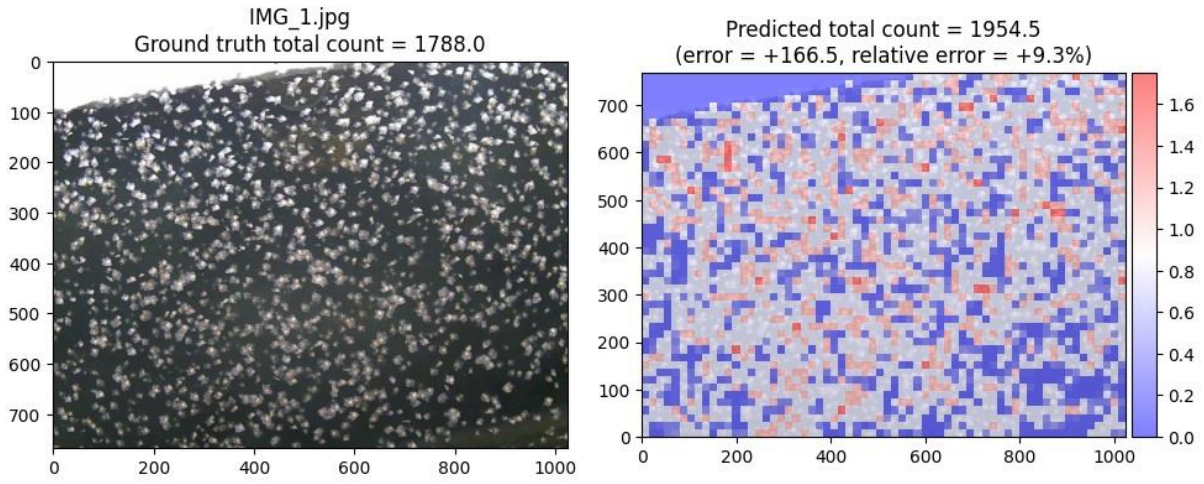


Figura 41: Ejemplo resultado para conjunto de datos del día 26 con desviación estándar 5

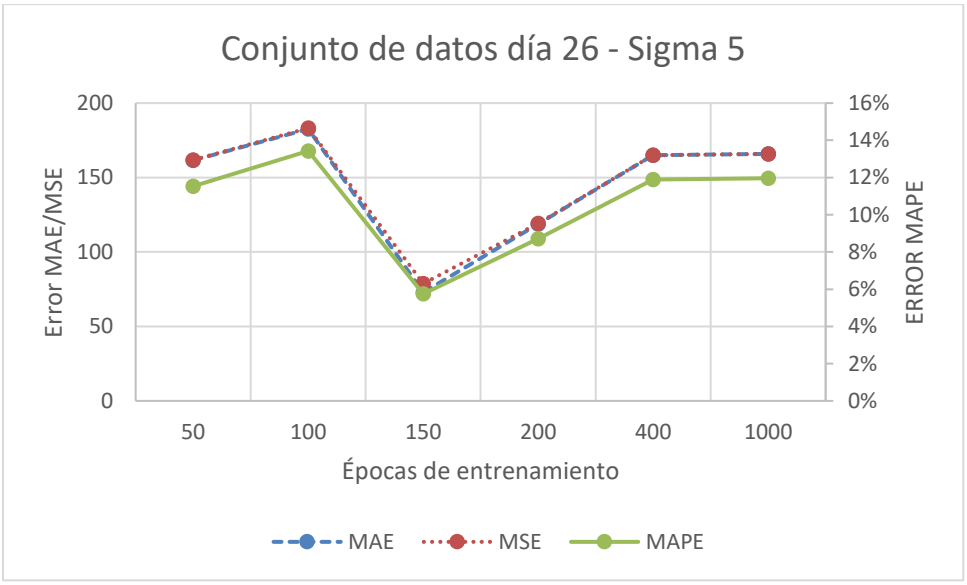


Figura 42: Proceso de los errores durante el entrenamiento para conjunto de datos del día 26 con desviación estándar 5

Desviación estándar 6

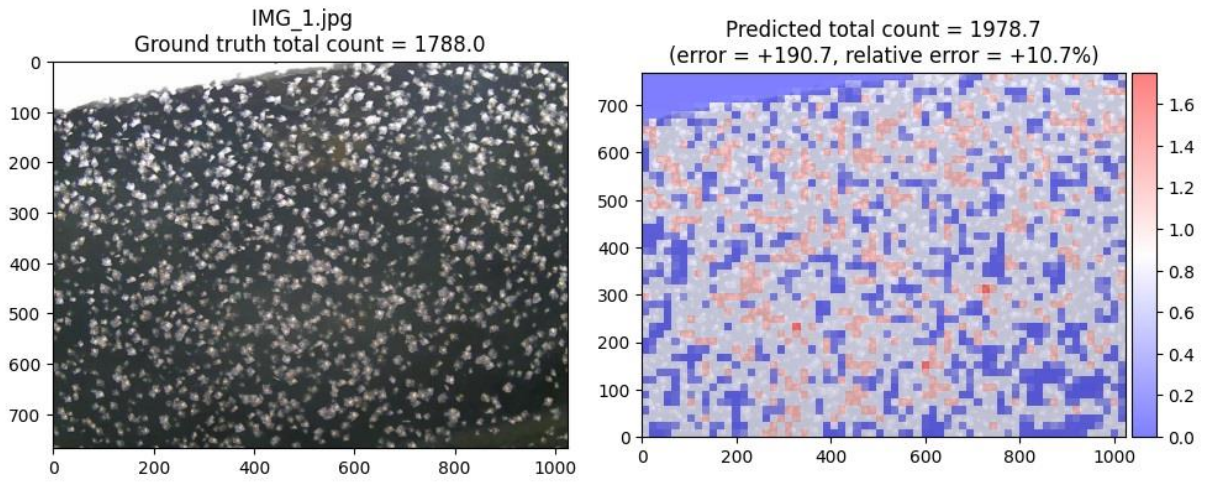


Figura 43: Ejemplo resultado para conjunto de datos del día 26 con desviación estándar 6

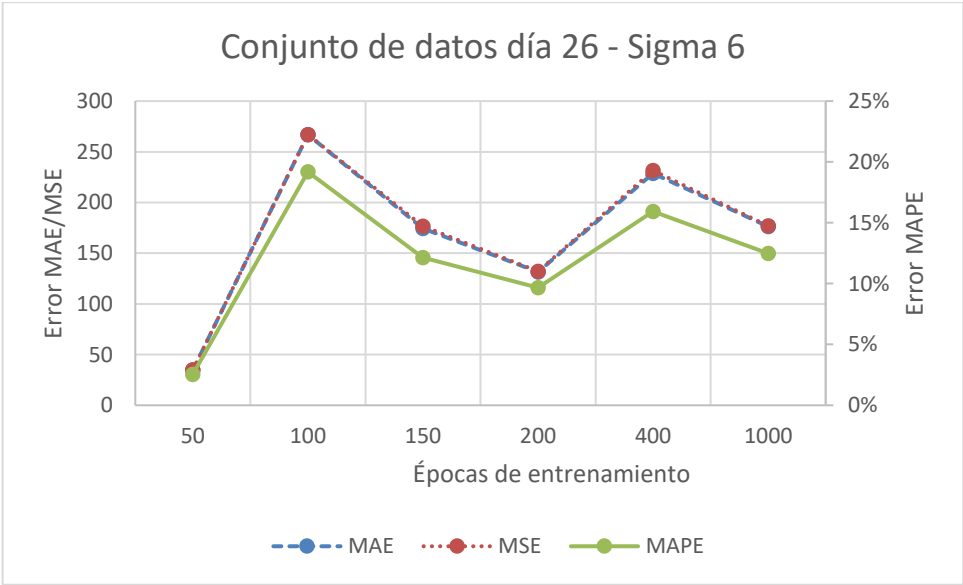


Figura 44: Proceso de los errores durante el entrenamiento para conjunto de datos del día 26 con desviación estándar 6

5.4 Conjunto de datos días 27-29

A continuación, se muestra una tabla resumen de los resultados obtenidos al variar el valor de sigma para las larvas nacidas entre los días 27 y 29. En esta tabla, se presentan los errores MAE, MSE y MAPE para diferentes valores de sigma.

Tabla 5: Resultados conjunto de entrenamiento días 27/29

Valor de σ	MAE	MSE	MAPE
3	24,91	24,96	6,00%
4	21,25	21,57	5,88%
5	34,78	34,79	6,81%
6	51,3	51,3	12,89%

En este conjunto de datos, se observa que el valor de sigma que proporciona el menor error es para sigma 4, con un MAE de 21,25, un MSE de 21,57 y un MAPE de 5,88%. Esto contrasta con los resultados obtenidos para los conjuntos de datos anteriores, donde el valor óptimo de sigma era 3. Aunque en este caso particular un sigma de 4 ofrece una mejor precisión, los errores aumentan significativamente con valores de sigma mayores a este.

A pesar de este hallazgo, al ser el único conjunto de datos en el que un sigma de 4 proporciona mejores resultados que uno de 3, se concluye que no merece la pena realizar estudios adicionales con sigma variable. Los resultados indican que un sigma fijo de 3 es generalmente más efectivo para la mayoría de los conjuntos de datos, simplificando el modelo y manteniendo una precisión adecuada sin la necesidad de ajustes adicionales.

Desviación estándar 3

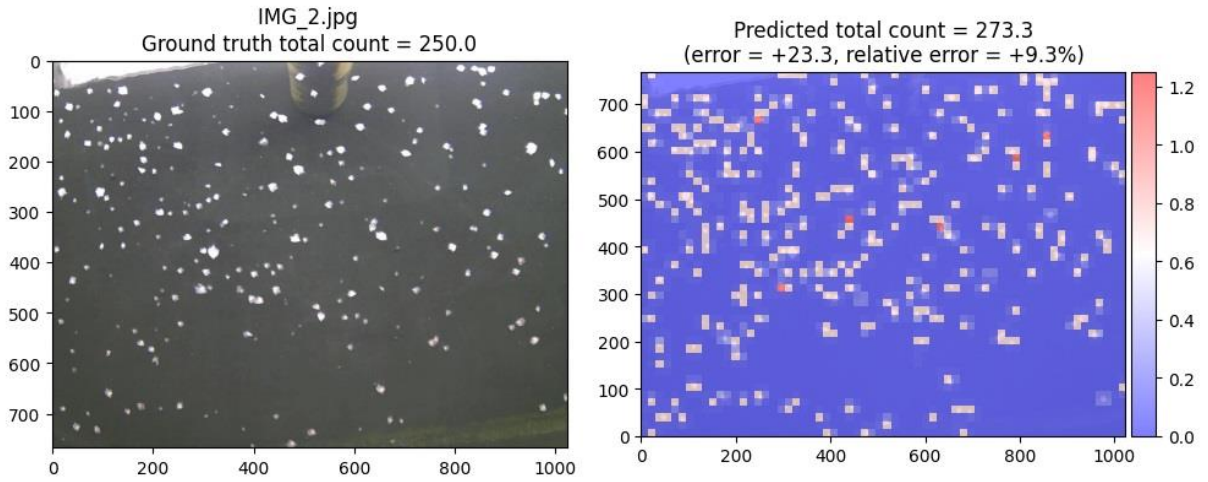


Figura 45: Ejemplo resultado para conjunto de datos de días 27/29 con desviación estándar 3

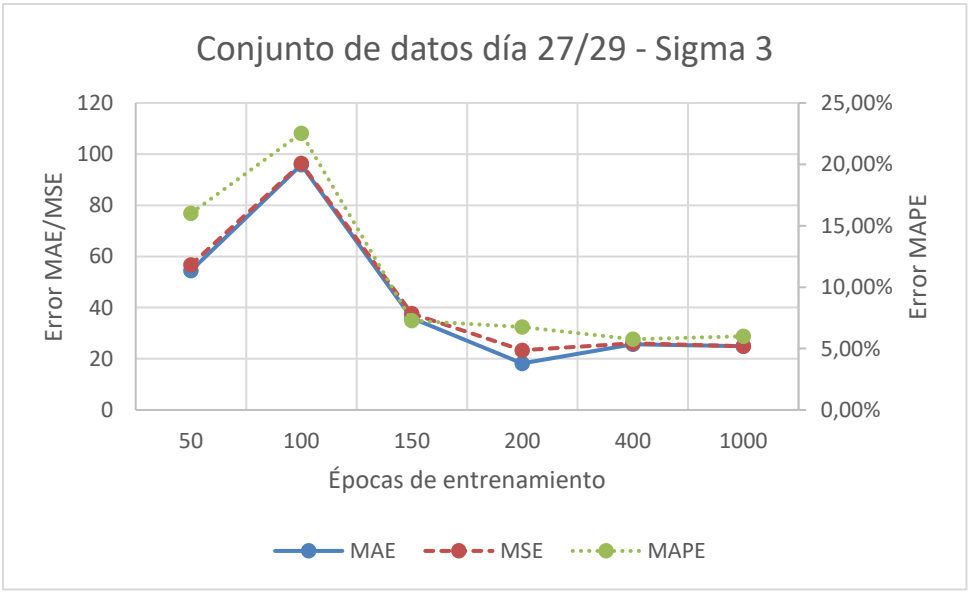


Figura 46: Proceso de los errores durante el entrenamiento para conjunto de datos de días 27/29 con desviación estándar 3

Desviación estándar 4

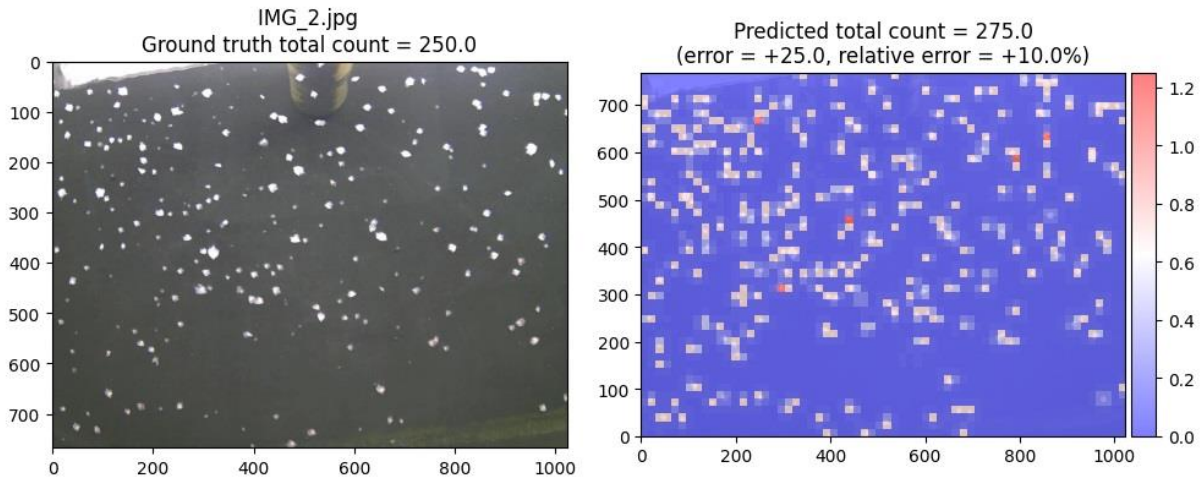


Figura 47: Ejemplo resultado para conjunto de datos de días 27/29 con desviación estándar 4

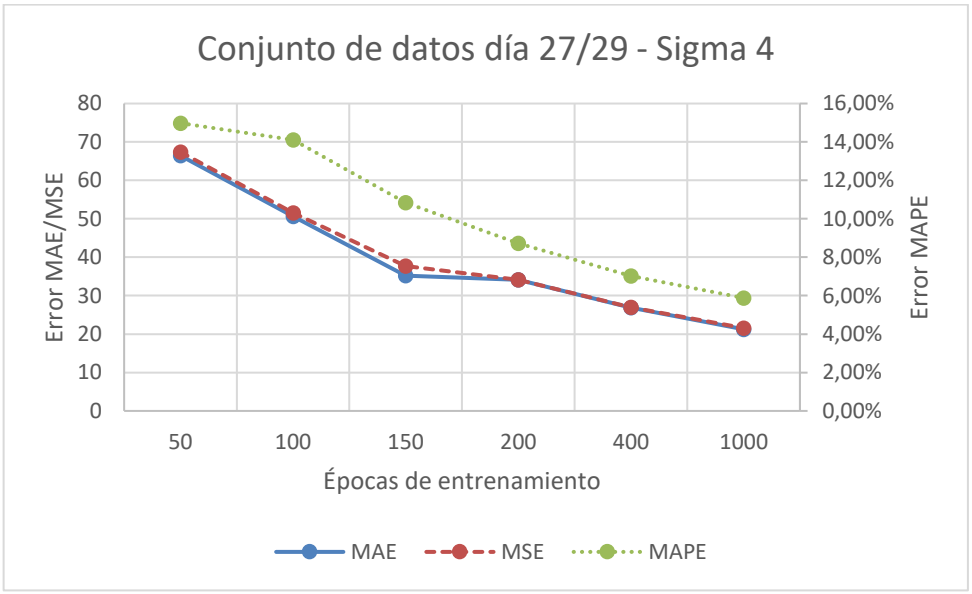


Figura 48: Proceso de los errores durante el entrenamiento para conjunto de datos de días 27/29 con desviación estándar 4

Desviación estándar 5

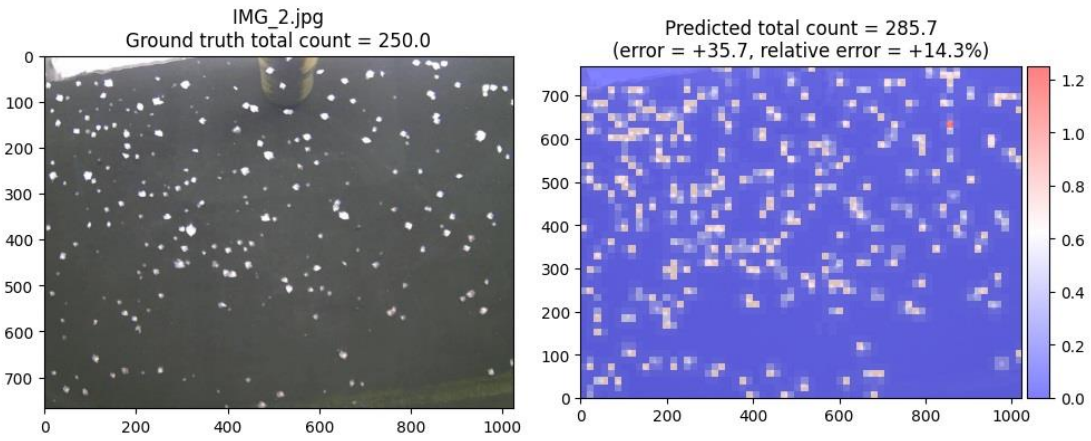


Figura 49: Ejemplo resultado para conjunto de datos de días 27/29 con desviación estándar 5

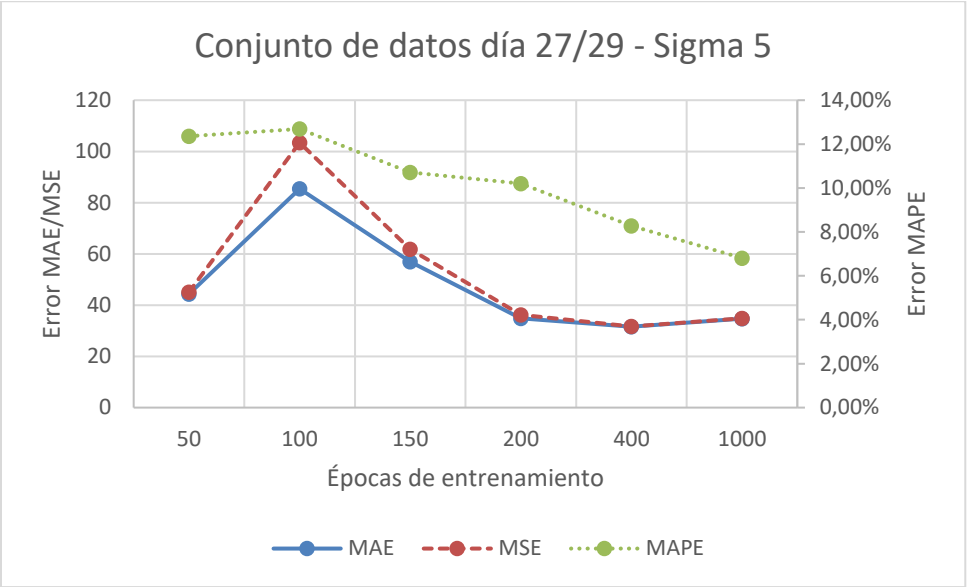


Figura 50: Proceso de los errores durante el entrenamiento para conjunto de datos de días 27/29 con desviación estándar 5

Desviación estándar 6

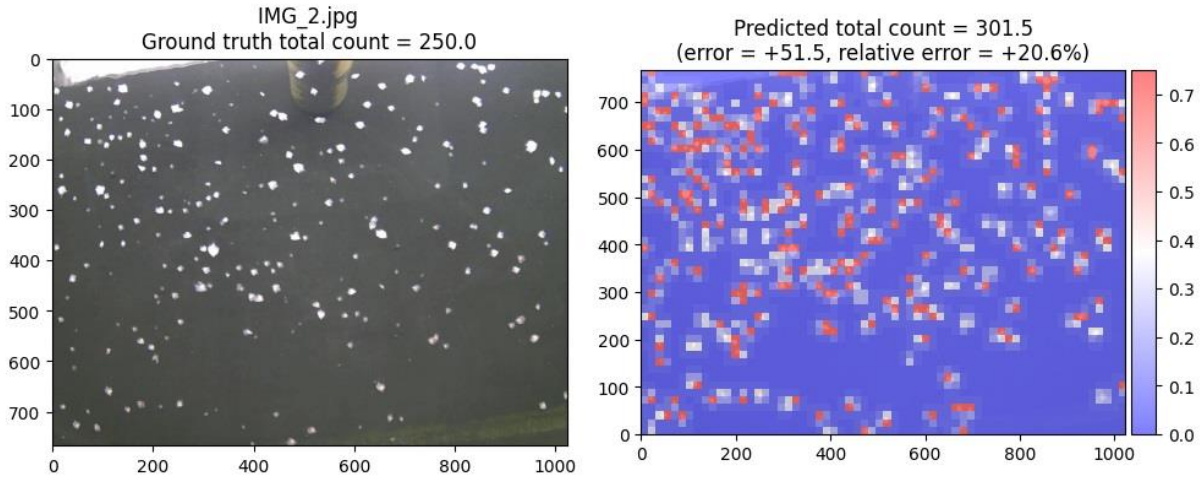


Figura 51: Ejemplo resultado para conjunto de datos de días 27/29 con desviación estándar 6

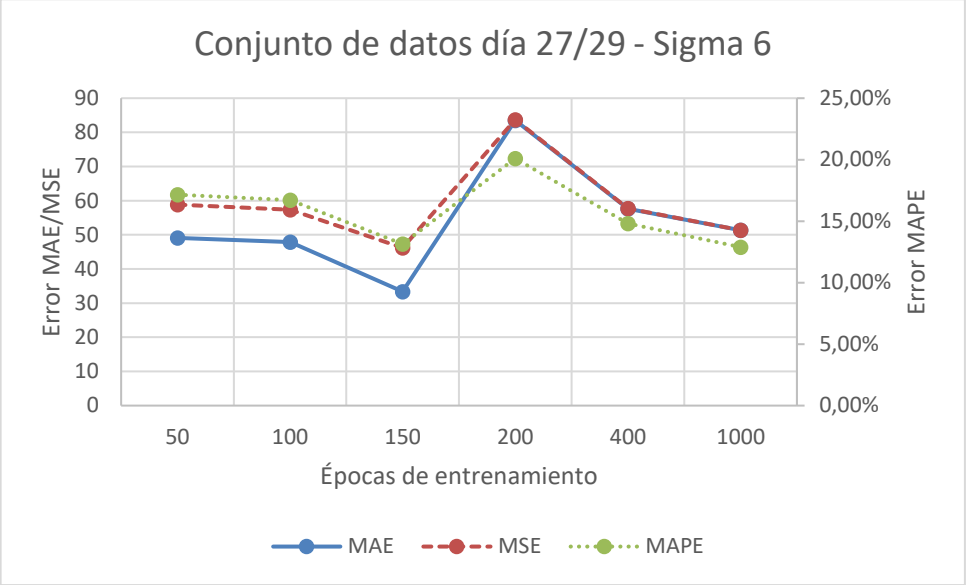


Figura 52: Proceso de los errores durante el entrenamiento para conjunto de datos de días 27/29 con desviación estándar 6

5.5 Síntesis de resultados de los conjuntos de datos

Después de observar los resultados obtenidos para cada conjunto de datos, simplificamos los datos en la siguiente tabla, donde podremos apreciar para cada conjunto de datos y desviación estándar asignada, sus errores en el entrenamiento.

Tabla 6: Resumen resultados de los diferentes conjuntos de datos

	MAE				MSE				MAPE			
	S3	**S4**	**S5**	**S6**	**S3**	**S4**	**S5**	**S6**	**S3**	**S4**	**S5**	**S6**
23-25	77,83	89,26	92,62	103,93	91,63	108,1	110,35	121,55	14,48%	17,48%	18,00%	19,20%
26	86,41	135	165,79	176,24	86,9	135,1	165,79	176,83	6,09%	9,85%	11,96%	12,49%
27-29	24,91	21,25	34,78	51,3	24,96	21,57	34,79	51,3	6,00%	5,88%	6,81%	12,89%

Tal como podemos observar en la **tabla 6**, el análisis de los datos revela que, aunque un valor de sigma 4 ofrece un rendimiento ligeramente mejor en el conjunto de datos 27-29, el valor de sigma 3 proporciona los menores errores en los conjuntos de datos 23-25 y 26. Dado que el valor de sigma 3 funciona adecuadamente en la mayoría de los conjuntos de datos y simplifica la implementación al evitar ajustes adicionales, se concluye que utilizar un valor fijo de sigma 3 es la opción más práctica y efectiva. Esta decisión reduce la complejidad del sistema y garantiza una precisión razonable en las estimaciones de biomasa, optimizando así el monitoreo y gestión en la acuicultura

6. Presupuesto

A continuación, se detalla el presupuesto consumido para elaboración de un sistema de conteo de rodaballos mediante mapas de densidad. Se ha planificado con un total de 340 horas de trabajo, siendo equivalentes a 12 créditos ECTS. El ingeniero de telecomunicaciones encargado, con un coste por hora de 12.5 €, llevará a cabo todas las fases del proyecto, desde la investigación hasta la implementación final, utilizando tanto hardware como software especializado.

Tabla 7: Presupuesto total del proyecto

Partidas	Descripción	Tiempo	Horas Totales	Precio (€/hora)	Precio Total (€)
Definición del Proyecto y Objetivos					
Revisión Bibliográfica	Investigación de técnicas y algoritmos	1 semana	12 horas	12.5 €	150,00 €
Planificación del Proyecto	Diseño del plan de desarrollo	1 semana	12 horas	12.5 €	150,00 €
Búsqueda de Recursos Informativos y Documentación					
Revisión de Materiales y Fuentes	Búsqueda y análisis de documentación	2 semanas	24 horas	12.5 €	300,00 €
Análisis y Ajustes de Modelos Neuronales					
Investigación y Ajuste	Ajustes preliminares	4 semanas	48	12.5 €	600,00 €
Diseño y Entrenamiento del Modelo					
Ingeniero de Telecomunicaciones	1 ingeniero	2,5 meses	70 horas	12.5 €	875,00 €
Análisis y Evaluación del Modelo					
Análisis de Datos	Validación de resultados y ajuste de modelos	2 semanas	32 horas	12.5 €	400,00 €
Conclusiones					
Resumen de Resultados	Compilación y análisis final de resultados	2 semanas	32 horas	12.5 €	400,00 €
Elaboración de la Memoria					
Documentación	Creación de documentación técnica y de usuario	4.5 meses	110 horas	12.5 €	1.375,00 €
Hardware y Software					
Ordenador	PC para desarrollo	1 unidad			1.200,00 €
Licencias de Software	Herramientas de desarrollo y análisis				800,00 €
TOTAL (SIN IVA)			**340 horas**		**6.250,00 €**
TOTAL (CON IVA)					**7.562,50 €**

7. Impacto del proyecto

El desarrollo del sistema de conteo de larvas de rodaballo mediante mapas de densidad y redes neuronales convolucionales tiene múltiples implicaciones positivas en diversos ámbitos, destacándose su impacto en las áreas social, económica, tecnológica, ambiental y en la consecución de los Objetivos de Desarrollo Sostenible (ODS).

Impacto social

En el ámbito social, la implementación de este sistema contribuye significativamente a mejorar las prácticas en la acuicultura. Al automatizar el proceso de conteo de larvas, se reduce la carga de trabajo manual, permitiendo a los trabajadores concentrarse en tareas de mayor valor añadido. Además, la precisión y consistencia del sistema reducen el riesgo de errores humanos, mejorando la calidad del trabajo y el bienestar laboral. Esta tecnología también puede ser aplicada a otras especies acuáticas, ampliando su impacto positivo en diversas comunidades pesqueras.

Impacto económico

Económicamente, el sistema presenta una oportunidad de optimización de los recursos en la industria de la acuicultura. Al proporcionar datos precisos sobre la cantidad de larvas de rodaballo, los acuicultores pueden tomar decisiones informadas sobre la alimentación, el manejo de los tanques y la cosecha. Esto puede resultar en una mejora significativa en la eficiencia de la producción y una reducción de los costos operativos. La automatización del conteo también permite un uso más eficiente del tiempo y los recursos humanos, incrementando la productividad y la rentabilidad de las operaciones acuícolas.

Impacto tecnológico

Desde una perspectiva tecnológica, este proyecto destaca la aplicación avanzada de redes neuronales convolucionales, especialmente en el campo de la acuicultura. La metodología desarrollada no solo mejora el conteo de larvas de rodaballo, sino que también establece un precedente para la adopción de tecnologías de inteligencia artificial en la monitorización y gestión de recursos naturales. La transferencia de esta tecnología a otros ámbitos de la biología y la agricultura puede impulsar innovaciones adicionales, fomentando un ecosistema de desarrollo tecnológico en estas industrias.

Impacto ambiental

El sistema de conteo automatizado tiene un impacto ambiental positivo al promover prácticas de acuicultura más sostenibles. Al optimizar el manejo de las poblaciones de peces y reducir el desperdicio de alimentos, se disminuye la carga ambiental de las operaciones acuícolas. Una mejor gestión de los recursos acuáticos contribuye a la conservación de los ecosistemas marinos y a la reducción de la sobreexplotación de especies. Además, al minimizar el estrés y la manipulación directa de los peces, se mejora el bienestar animal y se promueve una producción más ética y responsable.

Contribución a los Objetivos de Desarrollo Sostenible (ODS)

Este proyecto contribuye directamente a varios Objetivos de Desarrollo Sostenible de las Naciones Unidas. En particular, apoya el ODS 2: "Hambre cero", al mejorar la eficiencia y sostenibilidad de la producción de alimentos acuáticos. También se alinea con el ODS 9: "Industria, innovación e infraestructura", al fomentar la innovación tecnológica en la acuicultura. Asimismo, contribuye al ODS 12: "Producción y consumo responsables", promoviendo prácticas de acuicultura que optimizan el uso de recursos y minimizan el impacto ambiental. Por último, el proyecto tiene implicaciones positivas para el ODS 14: "Vida submarina", al apoyar la gestión sostenible y la conservación de los ecosistemas marinos.

Figura 53: Objetivos de desarrollo sostenible

En conclusión, el impacto del proyecto de conteo de larvas de rodaballo mediante redes neuronales convolucionales es amplio y multifacético. Su implementación no solo mejora la eficiencia y precisión en la acuicultura, sino que también contribuye a la sostenibilidad ambiental, la innovación tecnológica y el bienestar social y económico. Estos beneficios refuerzan la importancia de continuar desarrollando y aplicando tecnologías avanzadas en la gestión de recursos naturales, promoviendo un futuro más sostenible y eficiente para la industria acuícola y el medio ambiente global.

8. Conclusiones

El desarrollo del sistema de conteo de larvas de rodaballo mediante mapas de densidad utilizando redes neuronales convolucionales ha permitido alcanzar varios objetivos importantes en el ámbito de la acuicultura. A través de este proyecto, se han podido extraer diversas conclusiones que evidencian la viabilidad y eficiencia del modelo propuesto.

Una de las mejoras más destacadas obtenidas en este proyecto se refleja en los resultados comparativos con el conjunto de datos ShanghaiTech, correspondientes a los resultados del código fuente [27]. Al comparar los errores obtenidos, se observó que nuestro modelo presenta una notable reducción en las métricas de error, como el MAE y MSE, cuando se utiliza para el conteo de larvas de rodaballo del conjunto de datos de test completo para un estado de larva del día 25-26. Específicamente, los resultados muestran que el sistema propuesto no solo supera los datos de referencia de ShanghaiTech , sino que también se ha optimizado con respecto a los resultados obtenidos en proyectos anteriores de la misma índole[29].

Los resultados detallados indican que para el conjunto de datos de ShanghaiTech Part A (compuesto de 300 imágenes para entrenamiento y 182 imágenes para test), los valores de MAE y MSE para el conjunto de datos de test fueron de 61.16 y 105.23 respectivamente con una desviación estándar variable de geometría adaptativa y un $C_{max} = 22.0$. Para el conjunto de datos ShanghaiTech Part B (compuesto de 400 imágenes de entrenamiento y 316 para test), los valores de MAE y MSE para el conjunto de datos de test fueron de 8.22 y de 13.65 respectivamente para una desviación estándar de 13 y un CMAX de 7. En contraste, para el conjunto de datos completo para día 25-26 de nacimiento de nuestro proyecto, los errores de conjunto de datos de test se redujeron a valores de MAE y MSE correspondientes a 4.5 y 8.12 respectivamente (lo que supone un 0,33% MAPE) para una desviación estándar de 6 $y\ C_{max} = 5$

Sin embargo, debido al reducido número de muestras de imágenes etiquetadas, se ha observado que no es eficiente el uso de una desviación estándar variable en función de los días de nacimiento. Los resultados para los tres conjuntos de datos (23-25 días, 26, y 27-29 días) indican que el menor error se produce consistentemente con una desviación estándar de 3. Esto se evidenció al comparar los valores de error obtenidos: para los días 23-25, los errores del conjunto de datos de test fueron 77.86 (MAE) y 91.63 (MSE) con un 14.48% MAPE, para el día 26, los valores fueron 86.41 (MAE) y 86.9 (MSE) con un 6.09% MAPE. La excepción la encontramos en conjunto de datos de días 27-29, cuyo mejor resultado se obtuvo para una desviación estándar de 4, obteniendo así unos resultados de 21.25 (MAE) y 21.57 (MSE) con un 5.88% MAPE.

Estos resultados sugieren que mantener una desviación estándar fija de 3 para todos los conjuntos de datos proporciona una mayor consistencia y precisión en el conteo de larvas de rodaballo. Este hallazgo es crucial, ya que simplifica el proceso de modelado y reduce la complejidad computacional sin comprometer la precisión del conteo. La implementación de

una desviación estándar variable no demostró beneficios adicionales significativos y, por lo tanto, no se justifica su uso en el contexto de este proyecto.

En conclusión, el proyecto ha demostrado que el uso de redes neuronales convolucionales para el conteo de larvas de rodaballo es una solución viable y eficiente. Las mejoras en precisión y adaptabilidad del modelo subrayan su potencial para aplicaciones prácticas en la acuicultura. Sin embargo, se recomienda continuar investigando y optimizando el sistema, enfocándose en la expansión del conjunto de datos de entrenamiento y explorando nuevas técnicas de aprendizaje profundo que puedan complementar y mejorar aún más los resultados obtenidos.

9. Referencias

[1] *The State of World Fisheries and Aquaculture 2022*. FAO, 2022. doi: 10.4060/cc0461en.

[2] W. Shen, Z. Peng, y J. Zhang, «Identification and counting of fish targets using adaptive resolution imaging sonar», *J. Fish Biol.*, vol. 104, n.º 2, pp. 422-432, 2024, doi: 10.1111/jfb.15349.

[3] Y. Chai, H. Yu, L. Xu, D. Li, y Y. Chen, «Deep Learning Algorithms for Sonar Imagery Analysis and Its Application in Aquaculture: A Review», *IEEE Sens. J.*, vol. 23, n.º 23, pp. 28549-28563, dic. 2023, doi: 10.1109/JSEN.2023.3324438.

[4] A. G. Checa, J. H. E. Cartwright, I. Sánchez-Almazo, J. P. Andrade, y F. Ruiz-Raya, «The cuttlefish Sepia officinalis (Sepiidae, Cephalopoda) constructs cuttlebone from a liquid-crystal precursor», *Sci. Rep.*, vol. 5, n.º 1, p. 11513, jun. 2015, doi: 10.1038/srep11513.

[5] R. Lin y Y. Liu, «A Real-Time Counting Method of Fish based on the Instance Segmentation», en *2022 China Automation Congress (CAC)*, nov. 2022, pp. 133-138. doi: 10.1109/CAC57257.2022.10054787.

[6] H. Liu, Y. Yu, L. Wang, L. Hao, H. Cui, y X. Ma, «Automatic Fish Counting in Aquaculture with Dense Multi-scale Feature Aggregation Network», jul. 2023, pp. 8246-8251. doi: 10.23919/CCC58697.2023.10241074.

[7] J. Li, J. Sun, X. Cui, B. Jiang, S. Li, y J. Liu, «Automatic Counting Method of Fry Based on Computer Vision», *IEEJ Trans. Electr. Electron. Eng.*, vol. 18, n.º 7, pp. 1151-1159, 2023, doi: 10.1002/tee.23821.

[8] A. Jalal, A. Salman, A. Mian, M. Shortis, y F. Shafait, «Fish detection and species classification in underwater environments using deep learning with temporal information», *Ecol. Inform.*, vol. 57, p. 101088, may 2020, doi: 10.1016/j.ecoinf.2020.101088.

[9] B. Zohuri, «The Evolution of Artificial Intelligence: From Supervised to Semi-Supervised and Ultimately Unsupervised Technology Trends», *Curr. Trends Eng. Sci. CTES*, vol. 3, n.º 5, pp. 1-4, ago. 2023, doi: 10.54026/CTES/1040.

[10] F. Emmert-Streib y M. Dehmer, «Taxonomy of machine learning paradigms: A data-centric perspective», *WIREs Data Min. Knowl. Discov.*, vol. 12, n.º 5, p. e1470, 2022, doi: 10.1002/widm.1470.

[11] M. Mouriño-García, «Clasificación multilingüe de documentos utilizando machine learning y la Wikipedia - Multilingual document classification using machine learning and Wikipedia», 2018. doi: 10.13140/RG.2.2.11966.38723.

[12] «Unsupervised Machine Learning - an overview | ScienceDirect Topics». Accedido: 15 de junio de 2024. [En línea]. Disponible en: https://www.sciencedirect.com/topics/biochemistry-genetics-and-molecular-biology/unsupervised-machine-learning

[13] J. E. van Engelen y H. H. Hoos, «A survey on semi-supervised learning», *Mach. Learn.*, vol. 109, n.º 2, pp. 373-440, feb. 2020, doi: 10.1007/s10994-019-05855-6.

[14] T. Li, J. Lu, J. Wu, Z. Zhang, y L. Chen, «Predicting Aquaculture Water Quality Using Machine Learning Approaches», *Water*, vol. 14, n.º 18, Art. n.º 18, ene. 2022, doi: 10.3390/w14182836.

[15] F. A. Ghani, M. Rivaie, M. Yusoff, y M. Puteh, «A Review of Artificial Neural Network Applications in Variants of Optimization Algorithms», en *2022 International Visualization, Informatics and Technology Conference (IVIT)*, nov. 2022, pp. 115-123. doi: 10.1109/IVIT55443.2022.10033339.

[16] E. Franco, «Aprendizaje de máquina y aprendizaje profundo en biotecnología: aplicaciones, impactos y desafíos», *Cienc. Ambiente Clima*, Accedido: 15 de junio de 2024.

[En línea]. Disponible en: https://www.academia.edu/54953902/Aprendizaje_de_m%C3%A1quina_y_aprendizaje_profundo_en_biotecnolog%C3%ADa_aplicaciones_impactos_y_desaf%C3%ADos

[17] G. Cohen y R. Giryes, «Generative Adversarial Networks». arXiv, 1 de marzo de 2022. Accedido: 15 de junio de 2024. [En línea]. Disponible en: http://arxiv.org/abs/2203.00667

[18] D. Calvo, «Red Neuronal Convolucional CNN», Diego Calvo. Accedido: 16 de junio de 2024. [En línea]. Disponible en: https://www.diegocalvo.es/red-neuronal-convolucional/

[19] A. Krizhevsky, I. Sutskever, y G. E. Hinton, «ImageNet Classification with Deep Convolutional Neural Networks», en *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2012. Accedido: 16 de junio de 2024. [En línea]. Disponible en: https://proceedings.neurips.cc/paper_files/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html

[20] M. Mishra, «Convolutional Neural Networks, Explained», Medium. Accedido: 16 de junio de 2024. [En línea]. Disponible en: https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939

[21] S. Bangar, «LeNet 5 Architecture Explained», Medium. Accedido: 16 de junio de 2024. [En línea]. Disponible en: https://medium.com/@siddheshb008/lenet-5-architecture-explained-3b559cb2d52b

[22] N. Klingler, «AlexNet: A Revolutionary Deep Learning Architecture», viso.ai. Accedido: 16 de junio de 2024. [En línea]. Disponible en: https://viso.ai/deep-learning/alexnet/

[23] T. Ghazal *et al.*, «Early Detection of Autism in Children Using Transfer Learning», oct. 2022.

[24] S. Bangar, «VGG-Net Architecture Explained», Medium. Accedido: 16 de junio de 2024. [En línea]. Disponible en: https://medium.com/@siddheshb008/vgg-net-architecture-explained-71179310050f

[25] A.- I, «Understanding ResNet Architecture: A Deep Dive into Residual Neural Network», Medium. Accedido: 16 de junio de 2024. [En línea]. Disponible en: https://medium.com/@ibtedaazeem/understanding-resnet-architecture-a-deep-dive-into-residual-neural-network-2c792e6537a9

[26] «Understanding GoogLeNet Model - CNN Architecture», GeeksforGeeks. Accedido: 16 de junio de 2024. [En línea]. Disponible en: https://www.geeksforgeeks.org/understanding-googlenet-model-cnn-architecture/

[27] H. Xiong, H. Lu, C. Liu, L. Liu, C. Shen, y Z. Cao, «From Open Set to Closed Set: Supervised Spatial Divide-and-Conquer for Object Counting», *Int. J. Comput. Vis.*, vol. 131, n.º 7, pp. 1722-1740, jul. 2023, doi: 10.1007/s11263-023-01782-1.

[28] G. Ding, M. Cui, D. Yang, T. Wang, S. Wang, y Y. Zhang, «Object Counting for Remote-Sensing Images via Adaptive Density Map-Assisted Learning», *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1-11, 2022, doi: 10.1109/TGRS.2022.3208326.

[29] « A. F. González, "Conteo de objetos en imágenes RGB con redes convolucionales mediante división espacial" Proyecto Final de Grado, Universidad Politécnica de Madrid, Madrid, España, 2023. ».

[30] S. M. Robeson y C. J. Willmott, «Decomposition of the mean absolute error (MAE) into systematic and unsystematic components», *PLOS ONE*, vol. 18, n.º 2, p. e0279774, feb. 2023, doi: 10.1371/journal.pone.0279774.

[31] T. O. Hodson, «Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not», *Geosci. Model Dev.*, vol. 15, n.º 14, pp. 5481-5487, jul. 2022, doi: 10.5194/gmd-15-5481-2022.

[32] «Oracle® Fusion Cloud EPM Trabajo con Planning». Accedido: 30 de junio de 2024. [En línea]. Disponible en:

https://docs.oracle.com/cloud/help/es/pbcs_common/PFUSU/insights_metrics_MAPE.htm#PFUSU-GUID-C33B0F01-83E9-468B-B96C-413A12882334

Anexo

Manual de usuario

Obtención de fotogramas a partir de un video

El archivo de Matlab llamado “obtenerFotograma.m” se encarga de extraer fotogramas específicos de una serie de videos y guardarlos como imágenes JPEG. Todo se realiza automáticamente en el mismo directorio donde se encuentra el archivo MATLAB, los videos, y el archivo Excel que contiene la información de los fotogramas a extraer. Las imágenes extraídas se guardan en una subcarpeta llamada "imágenes".

El archivo Excel contiene un listado de videos separados por días del estado de larva y fotogramas que han sido repartido a los alumnos que participan en GAMMA. La distribución de las columnas más importantes debe de ser la siguiente:

- Columna 1 (Videos): contendrá el nombre del video a partir del cual se obtiene el fotograma deseado.
- Columna 2 (Edad_dias): Esta columna indicará el estado de larva al que corresponde el video.
- Columna 3 (Fotograma): Esta columna indica el fotograma exacto que se desea extraer del video original.

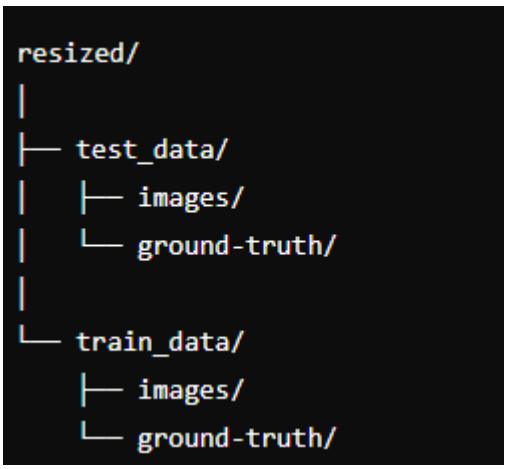
El resto de las columnas carece de importancia para la obtención del fotograma en cuestión.

Las imágenes obtenidas tendrán la misma resolución que el video original y pasaran a llamarse de la siguiente manera: “nombreOriginalDelVideo_FotogramaDeseado”.Ejemplo: para un video con resolución 2560x1920 cuyo nombre es *L05T02-20221121113453-20221121113953_1* y el fotograma deseado es el *2790*, el nombre de la imagen será *L05T02-20221121113453-20221121113953_1_2790*.

Creación de conjunto de datos “datasets”

El archivo “createDataset.m” procesa los archivos .mat obtenidos a partir del proceso de etiquetaje mencionado en el apartado 4.3. Se debe ejecutar sobre la carpeta donde se encuentren las imágenes y los .mat generados del etiquetado. Este generará un conjunto de datos organizado en carpetas para entrenamiento (80%) y test (20%), con imágenes de un factor de reducción de 2.5 (en nuestro caso se realizará un redimensionado de los fotogramas obtenidos a partir de videos cuya resolución original es de 1560x1920, obtendremos así imágenes con una resolución de 1024x768).

La estructura de carpetas que generada para el conjunto de datos es la siguiente:



Generación de mapas de densidad.

El archivo "gen_density_maps.py" es un *script* que genera dos archivos de salida en formato .npz, uno para los datos de conjunto de entrenamiento y otro para los datos de conjunto de test. Los nombres de estos archivos se construyen en base a la configuración del parámetro sigma y al tipo de datos (entrenamiento o test). Específicamente, los nombres de los archivos generados son:

- density_maps_S<sigma>_train.npz
- density_maps_S<sigma>_test.npz

Aquí <sigma> es el valor del parámetro sigma redondeado a un número entero.

Los argumentos que recibe el *script* por parámetros son:

- **dataset_rootdir:** Directorio raíz donde se encuentra el conjunto de datos.
- **knn**: Número de vecinos más cercanos para calcular la distancia
- **max_knn_avg_dist**: Distancia promedio máxima permitida para los vecinos más cercanos.
- **sigma_fixed**: Booleano que determina si el valor de sigma es fijo (true) o relativo a la distancia promedio de los vecinos más cercanos (false).
- sigma: Valor de sigma si es fijo.
- **sigma_coef:** Coeficiente para calcular sigma como sigma_coef * knn_avg_dist
- **sqr_side**: Longitud del lado del cuadrado dentro del cual se calculan los valores Gaussianos.
- **num_proc:** Número de procesos a utilizar para la generación paralela de mapas de densidad.

Ejemplo de uso sigma fija:

```
python  gen_density_maps.py --dataset_rootdir=./dataset/ --sigma_fixed True --sigma 6
```

Ejemplo de uso para estrategia de geometría adaptativa:

```
python gen_density_maps.py --dataset_rootdir=./dataset/ --knn 3 --max_knn_avg_dist 50 --sigma_fixed False --sigma 6 --sigma_coef 0.3 --sqr_side 40 --num_proc 8
```

Lector mapas de densidad

El *script* "lectorMapasDensidad.py" está diseñado para cargar y visualizar imágenes de mapas de densidad almacenadas en archivos .npz. Según el tipo de conjunto de datos (entrenamiento o test), el *script* busca archivos que coincidan con el patrón density_maps_S*_{train|test}.npz, carga los datos y muestra cada mapa de densidad en una ventana separada. Dicho *Script* debe encontrarse en el mismo directorio que los archivos .npz.

Este *Script* recibe los siguientes parámetros mediante la consola de comandos:

- **type**: Indica el archivo del que queremos obtener las imágenes. Train si queremos abrir mapas de densidad del conjunto de datos de conjunto de entrenamiento, test si queremos abrir mapas de densidad de datos de conjunto de test.

Ejemplo de uso:

```
python lectorMapasDensidad.py --type train
```

Entrenamiento del modelo

El *script* "train.py" está diseñado para entrenar a nuestro modelo de red neuronal convolucional utilizando el conjunto de datos específico para cada ocasión. El *script* implementa el flujo completo del entrenamiento, incluyendo la carga de datos, la definición y configuración del modelo, el proceso de entrenamiento, la validación y la gestión de puntos de control (checkpoints).

Por otro lado, El archivo de configuración "config_train_val_test.yaml" define los parámetros necesarios para el entrenamiento, validación y prueba de nuestro modelo.

Ejemplo de uso del entrenamiento:

```
python train.py
```

Evaluación del modelo

El *script* "evaluate.py" se utiliza para evaluar el modelo preentrenado de un conjunto de datos previamente usado para el entrenamiento de nuestra IA. Realiza la inferencia utilizando el modelo, calcula las métricas de error y visualiza las predicciones.

Los parámetros de entrada para dicho *script* son los siguientes:

- **test.visualize:** Booleano que determina si las predicciones del modelo deben visualizarse (true) o no (false).
- **test.trained_ckpt_for_inference:** Ruta donde se encuentra el punto de control (o chekpoint) del modelo que debe cargarse para la inferencia.

Ejemplo de uso:

```
python evaluate.py test.visualize=True test.trained_ckpt_for_inference=outputs/2024-06-17/20-10-22/checkpoints/epoch_0500.pth
```