Inteligencia Artificial en la inspección de defectos de aerogeneradores

Información de los Autores

Dr. Ricardo Carreño Aguilera
Universidad del Istmo Campus Tehuantepec
carrenoricardo04@gmail.com
https://orcid.org/0000-0002-6240-0152

Ing. Israel Mendoza Vásquez
Universidad del Istmo Campus Tehuantepec
Email:israel.mendoza.v@gmail.com

Dr. Miguel Patiño Ortiz
Instituto Politécnico Nacional
Email: mpatino2002@ipn.mx
https://orcid.org/0000-0002-5630-8077

Nombre: Dr. Julián Patiño Ortiz
Instituto Politécnico Nacional
Email: jpatinoo@ipn.mx
https://orcid.org/0000-0001-8106-9293

Email: Mauro Mijangos Dominguez
Instituto Politécnico Nacional
mijangos0987@gmail.com
https://orcid.org/0009-0002-3997-1800

***Resumen*— El presente trabajo propone aprovechar técnicas del aprendizaje automático, para el desarrollo de un modelo neuronal en la detección, y reconocimiento de defectos visuales físicos en álabes de aerogeneradores en tiempo real, aplicando los modelos neuronales existentes del aprendizaje profundo, dado que estos son de gran importancia para la detección de objetos en imágenes y vídeos, por medio de convoluciones, filtros y algoritmos de propuestas regionales que ayudan en el aprendizaje del modelo a desarrollar, los defectos a detectar son los siguientes: erosión, grieta en concha, grieta en coat, guarda polvo, impacto de rayo, mala reparación y suciedad..**

***Palabras Clave* — Aprendizaje automático, aerogeneradores, álabes, convoluciones.**

Abstract- The present work proposes to take advantage of machine learning techniques for the development of a neural model for the detection and recognition of physical visual defects in wind turbine blades in real time, applying existing neural models of deep learning, since these are of great importance for the detection of objects in images and videos, through convolutions, filters and algorithms of regional proposals that help in learning the model to develop, the defects to be detected are the following: erosion, crack in shell, crack in coat, dust guard, lightning impact, bad repair and dirt.

***Keywords* — Machine learning, wind turbines, blades, convolutions.**

I. INTRODUCCIÓN

En el campo de la inteligencia artificial (IA), la visión artificial se destaca como un sector que permite a las máquinas entender y tomar decisiones basadas en información visual extraída de imágenes y videos. Básicamente, se trata de enseñar a las máquinas a "ver" y comprender su entorno de manera

similar a la visión humana. Si bien deben aprender estas habilidades más rápido y usando cámaras, datos y algoritmos en lugar de órganos visuales, su capacidad puede superar rápidamente las habilidades humanas en tareas específicas. Este adelanto ha llevado a la inteligencia artificial al sector eólico, mejorando la eficiencia y fiabilidad del sistema al monitorear y analizar aerogeneradores mediante cámaras estratégicamente ubicadas, al hacer la detección de fallas de los aerogeneradores ocupando un sistema inteligente como machine learning [1].

Este sector eólico al ser una fuente renovable, tiene una tendencia próspera en la producción de energía eléctrica, dando lugar a nuevas instalaciones de parques eólicos en diferentes lugares de México, en especial en el Istmo de Tehuantepec Oaxaca, esto ha provocado una demanda en el mantenimiento, análisis y estudios de aerogeneradores instalados, para satisfacer estas necesidades se aplican las nuevas tecnologías existentes una de ellas es la inteligencia artificial [2].

Esta tecnología llamada inteligencia artificial (IA) ha tenido un avance creciente en la industria eólica, en especial en la detección y prevención de daños en los aerogeneradores, por medio de redes neuronales artificiales, para detectar fallas de las turbinas, reconocimiento en la supervisión, control y adquisición de datos, clasificación de fisuras de álabes, predicción del estudio del viento, entre otras aplicaciones existentes. Estas son apoyadas por los aprendizajes: automático y profundo, que emplean algoritmos, filtros, técnicas esenciales y modelos neuronales para el reconocimiento de objetos [3].

Se propone implementar técnicas de aprendizaje automático para desarrollar un modelo de red neuronal capaz de detectar y reconocer defectos físicos en perfiles de álabes de aerogeneradores en tiempo real. Esto se logrará utilizando modelos de aprendizaje profundo como lo es YOLOv7 (You Only Look Once) [4], el cual es esencial para la detección de objetos en imágenes y aplicaciones en tiempo real.

II. METODOLOGÍA/DESARROLLO

La metodología adoptada se fundamenta en el trabajo de Hernández et al. [5] y sigue el proceso del descubrimiento de conocimiento en bases de datos (KDD, por sus siglas en inglés). Esta metodología fue elegida para la detección de imágenes debido a su capacidad para extraer información valiosa de grandes volúmenes de datos de manera eficiente. A lo largo de sus diversas fases, se optimiza el uso de imágenes con el fin de facilitar el entrenamiento de modelos de inteligencia artificial. Su estructura organizada favorece la limpieza y preparación de los datos, lo que incrementa la calidad de las características extraídas, aspecto fundamental para asegurar que los modelos de detección operen con datos precisos y relevantes. La Figura 1 muestra la adaptación de esta metodología para alcanzar el objetivo deseado.

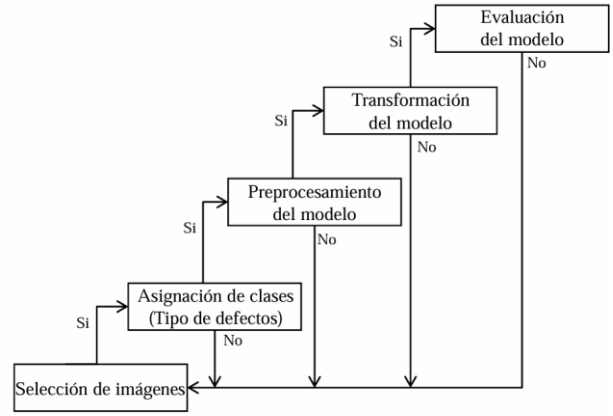

Figura 1.- Adaptación de la metodología del descubrimiento de conocimiento en base de datos.

En la Figura 1, se observan los pasos de cada etapa a continuación se describen de una manera más detallada.

- Selección de imágenes.

El primer paso fue la identificación de defectos en los álabes, para lograr esto, se realizó una investigación en el entorno laboral, lo que permitió clasificar los defectos en siete categorías, los cuales se detallan en la Figura 2.

Figura 2.- Efectos empleados para el banco de imágenes.

Al tener definidas las categorías se crea el banco de imágenes que sirve como una herramienta clave para el entrenamiento del modelo de aprendizaje automático, y también para la validación del modelo al estar entrenando, en la Figura 3, se presenta un muestreo del banco de imágenes.

Figura 3.- Muestreo del banco de imágenes.

Se llevó a cabo el proceso de etiquetado del banco de imágenes utilizando el software LabelImg, con el propósito de delimitar únicamente las áreas de interés que el modelo requiere aprender.

- Asignación de clases.

La distribución del banco de imágenes para la asignación de clases se realizó de la siguiente manera: un 80% de las imágenes fue destinado al entrenamiento y el 20% a la validación, siguiendo el criterio propuesto por Goodfellow en 2016 [6]. Es importante destacar que este criterio porcentual se aplica de forma equitativa a cada categoría. En la Figura 4, se ilustra dicha distribución, que comprende un total de 5963 imágenes, de las cuales 4772 se utilizaron para el entrenamiento y 1191 para la validación.

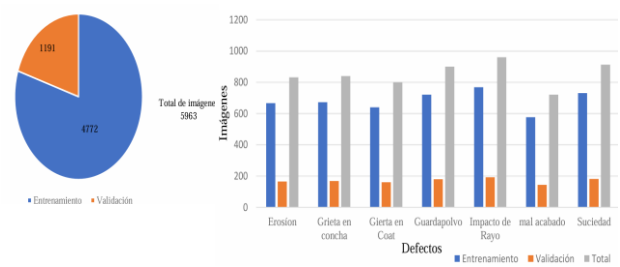

Figura 4.- Distribución de los defectos de los álabes siguiendo el criterio de Goodfellow.

- Preprocesamiento del modelo

En esta sección, se realizó el preprocesamiento requerido antes de hacer el entrenamiento del modelo. Para esto, fue necesario crear archivos de configuración en formato YAML y establecer la estructura de directorios en la que se alojaría el modelo YOLOv7. Los pasos seguidos se encuentran en la Figura 5.

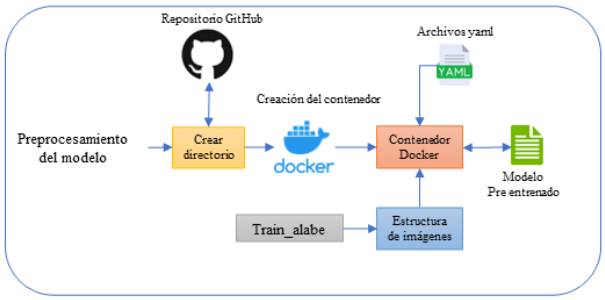

Figura 5.- Esquema representative del preprocesamiento del modelo.

Se estableció un directorio llamado "yolov7", donde se clona el repositorio de Yolov7, utilizando el siguiente comando:

git clone https://github.com/WongKinYiu/yolov7.git

Una vez definido el directorio, se procedió a crear y ejecutar el contenedor Docker, el cual alojará el modelo preentrenado YOLOv7, junto con el banco de imágenes (train_alabe) organizadas por clase, así como los archivos de configuración en formato YAML; para ejecutar e iniciar Docker, se utilizó el comando:
sudo docker run --name yolov7_res -it -v $PWD:/yolov7 --shm-size=64g nvcr.io/nvidia/pytorch:21.08-py3

- Transformación del modelo.

Durante la transformación del modelo, los archivos de configuración generados en la etapa de preprocesamiento contienen tanto las posiciones como las categorías de los defectos en cada imagen. Estos archivos son fundamentales para el proceso de entrenamiento, ya que constituyen las entradas al modelo de aprendizaje YOLOv7. Este modelo se distingue por implementar estrategias avanzadas de asignación de etiquetas y propagación de gradientes, lo que le permite superar limitaciones previas y ofrecer un rendimiento superior en diversas aplicaciones. A continuación, se presenta el pseudocódigo de YOLOv7.

```
# Inicialización del modelo y carga de pesos preentrenados
model = load_yolov7_model(weights_path)

# Preprocesamiento de la imagen de entrada
def preprocess_image(image):
# Redimensionar la imagen a la entrada requerida por
#el modelo (por ejemplo, 640x640)
resized_image = resize (image, target_size= (640, 640))
# Normaliza los valores de los píxeles a un rango de [0, 1]
normalized_image = normalize(resized_image)
return normalized_image

# Inferencia del modelo
def infer (model, preprocessed_image):
# Realizar la inferencia del modelo
output = model.predict(preprocessed_image)
return output

# Postprocesamiento de las detecciones
def postprocess_output(output, confidence_threshold,
nms_threshold):
# Filtrar las detecciones por el umbral de confianza
detections = filter_by_confidence(output,
threshold=confidence_threshold)

# Aplicar Supresión de No Máximos (Non-Maximum
Suppression, NMS)
para eliminar duplicados
final_detections = non_maximum_suppression(detections,
threshold=nms_threshold)
return final_detections

# Función principal de detección de objetos
def detect_objects(image, model, confidence_threshold=0.5,
nms_threshold=0.4):
```

```
# Preprocesar la imagen
preprocessed_image = preprocess_image(image)

# Realizar la inferencia
output = infer (model, preprocessed_image)

# Postprocesar las detecciones
detections = postprocess_output(output,
confidence_threshold, nms_threshold)

return detections

# Ejemplo de uso
if __name__ == "__main__":
# Cargar la imagen de entrada
image = load_image("input_image.jpg")

# Cargar el modelo YOLOv7 con pesos preentrenados
model = load_yolov7_model("yolov7_weights.pth")

# Detectar objetos en la imagen
detections = detect_objects(image, model)

# Mostrar o guardar las detecciones en la imagen
display_detections(image, detections)
```

- Evaluación

En esta etapa, se realiza la evaluación del modelo congelado que se generó previamente del entrenamiento, esto con el objetivo de realizar las predicciones del modelo, con imágenes distintas al banco de imágenes. Las predicciones realizadas se muestran en la Figura 6.

a) Detección erosión. b) Detección grieta en coat. c) Detección grieta en concha.

Figura 6. - Imágenes detectadas del modelo YOLOv7.

III. Resultados y Discusión

A continuación, se presentan los resultados de la implementación del modelo YOLOv7 para la detección de defectos en los álabes de aerogeneradores. Este modelo tiene un avance considerable en el ámbito del reconocimiento visual, destacándose por su capacidad para localizar objetos de manera precisa en imágenes y en tiempo real. En particular, ha sido especialmente adaptado para identificar defectos en los álabes, aprovechando su eficiencia al procesar la información en una sola pasada, lo que le permite detectar los defectos con gran rapidez. Para evaluar la precisión del modelo, se utilizó la matriz de confusión [7], herramienta clave para medir su desempeño.

- Matriz de confusión

La evaluación de modelos de clasificación es fundamental para determinar su desempeño y capacidad predictiva. En el presente estudio, se empleó un conjunto de datos compuesto por 560 imágenes aleatorias distribuidas entre las siete clases establecidas. Cada clase fue representada por un número específico de imágenes aleatorias asignadas para la evaluación del modelo, como se detalla a continuación: erosión (80 imágenes), grieta concha (80 imágenes), grieta coat (80 imágenes), guarda polvo (80 imágenes), impacto rayo (80 imágenes), mal acabado (80 imágenes), y suciedad (80 imágenes), como se muestra en la Tabla 1.

Tabla 1. Matriz de confusión.

	Clases Predichas						
Clases Reales	Erosión	Grieta concha	Grieta coat	Guarda polvo	Impacto rayo	Mal acabado	Suciedad
Erosión	53	10	0	0	6	3	8
Grieta concha	7	43	14	0	6	0	10
Grieta coat	2	15	48	0	8	0	7
Guarda polvo	0	0	0	75	0	0	5
Impacto rayo	8	6	4	0	47	0	15
Mal acabado	6	0	0	0	0	65	9
Suciedad	4	7	5	2	10	7	45

Una vez obtenido la matriz se obtuvo la exactitud y el error del modelo YOLOV7, a continuación, se muestra las fórmulas correspondientes.

$$\text{Exactitud} = \frac{\sum VP}{\sum Total\ de\ predicciones} \qquad (3.1)$$

$$\text{Tasa de Error} = \frac{\sum FP + \sum FN}{\sum Total\ \text{de predicciones}} \qquad (3.2)$$

Los cálculos detallados se presentan de la siguiente manera.

Suma de verdaderos positivos (VP):	53 + 43 + 48 + 75 + 47 + 65 + 45 = 376
Suma de falsos positivos (FP):	27 + 37 + 31 + 5 + 25 + 15 + 39 = 179
Suma de falsos negativos (FN):	27 + 38 + 24 + 2 + 24 + 7+ 47 = 169
Total, de predicciones:	376 VP+ 179 FP+ 169 FN = 724

Al sustituir en las fórmulas 3.1 y 3.2 los cálculos realizados, queda de la siguiente manera.

$$\text{Exactitud} = \frac{\sum VP}{\sum Total\ de\ predicciones} = \frac{376}{724}$$

$$Exactitud \ = \ 0.5192 \text{ x } 100 \ = 51.92 \text{ aprox } 52 \ \%$$

$$\text{Tasa de Error} = \frac{\sum FP + \sum FN}{\sum Total \text{ de predicciones}} \quad \frac{179 + 169}{724}$$

$$\text{Tasa de Error } = 0.4808 \text{ x } 100 = 48.08 \text{ aprox } 48\%$$

En la gráfica que se muestra en la Figura 7. se observa la precisión media (mAP) del modelo en la detección de objetos. Al calcular el promedio de las precisiones en diferentes niveles de umbral de Intersección sobre Unión (IoU), es decir, para:

- mAP@.5: La precisión media a un IoU de 0.5, significa que una predicción se considera correcta si la IoU es entre la caja predicha y la caja verdadera de al menos 0.5. Se observa la gráfica de color azul se visualiza que en las primeras 100 épocas se acerca al 0.6, y disminuye a 0.5 después de las 200 épocas, dando una precisión del 0.5 en promedio durante el total de épocas, en porcentajes sería un 50%.

- mAP@.5:.95: En el promedio de las precisiones medias en múltiples umbrales de IoU desde 0.5 hasta 0.95 con incrementos de 0.05. Es más rigurosa y proporciona una evaluación más completa de la capacidad del modelo para detectar objetos con precisión en diferentes niveles de tolerancia. Se observa la gráfica de color anaranjado está a diferencia de la anterior, se mantiene en 0.3 desde las primeras 100 épocas y se mantiene así en todo el entrenamiento.

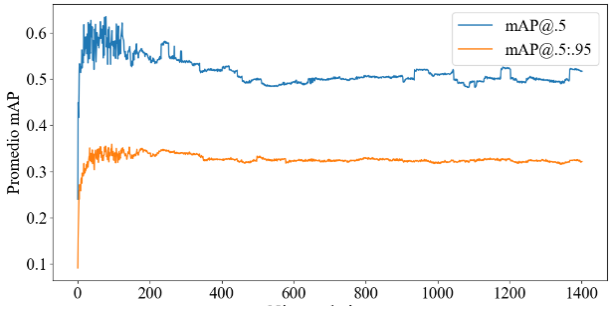

Figura 7.- Gráfica de promedio de Precisión Media mAP@.5 y mAP@.5:.95.

Por otro lado, en la Figura 8, se muestra las pérdidas que se obtuvieron en el entrenamiento y validación, estas pérdidas fueron de forma exponencial descendientes, lo que quiere decir que el error tiende a cero a mayor número de épocas, lo que significa que las pérdidas van disminuyendo y el modelo esta aprendiendo.

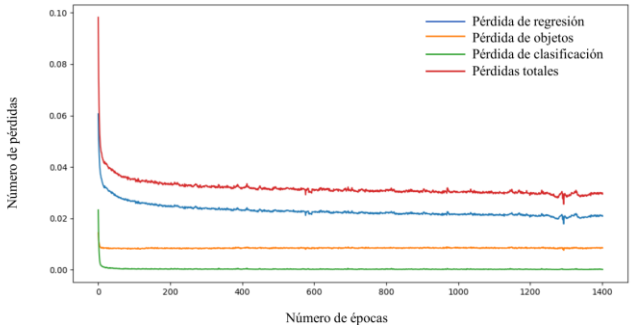

Figura 8.- Gráfica de pérdidas del entrenamiento.

- Detección en tiempo real

Actualmente, la detección de objetos en fotos, videos o en tiempo real es una de las tareas más importantes y desafiantes en la visión artificial. Esto consiste en detectar distintos objetos que aparecen en la imagen, video, o en este caso el entorno real, mediante una cámara web, es importante determinar coordenadas de un cuadro delimitador para cada objeto detectado e indicar a qué clase pertenece.

En la Figura 10 se muestra el resultado final de la detección de objetos de este proyecto al emplear una cámara web en tiempo real. Detectando así múltiples clases a medida que la cámara web recorría todo el perfil del álabe, mostrando los defectos localizados, incluyendo también un porcentaje de confianza para cada clase.

Figura. 10.- Ejemplo de la detección de defectos en un perfil de un álabe con el modelo empleado.

IV. CONCLUSIONES

El modelo YOLOv7 se ha destacado por su efectividad al detectar defectos en los alabes de aerogeneradores, combinando precisión y velocidad de manera excepcional. Su habilidad para encontrar rápidamente defectos en componentes pequeños y complejos como los alabes, incluso en condiciones cambiantes, resulta fundamental para la industria eólica. Esto no solo agiliza las inspecciones al reducir el tiempo necesario para identificar posibles problemas, sino que también contribuye a recortar

costos asociados con el mantenimiento y la operación de parques eólicos.

El objetivo de usar herramientas como mini anaconda, CUDA 12, Python, Docker y PyTorch junto con imágenes específicas para detectar y clasificar defectos visuales en álabes de aerogeneradores con una precisión mínima del 70% se cumplió solo parcialmente. El modelo puede realizar detecciones en tiempo real, pero su precisión está por debajo del 70%. En la detección de imágenes, el modelo alcanzó una exactitud del 52%, y en tiempo real, su precisión fluctuó entre el 27% y el 71%. Esto indica que el entrenamiento del modelo durante 1400 épocas a lo largo de 1.5 meses no fue suficiente para alcanzar la hipótesis planteada.

A pesar de entrenar durante 1400 épocas, es posible que el modelo no haya alcanzado su máximo rendimiento debido a limitaciones en el conjunto de datos. Se necesitan más imágenes para cada clase y entrenar por más tiempo y con más épocas.

REFERENCIAS

[1] J. Doe, Artificial Intelligence in Wind Energy: Enhancing Efficiency through Computer Vision, IEEE Trans. Sustainable Energy, Vol. 12, No. 4, pp. 1234-1245, 2021.

[2] Asociación Mexicana de Energía Eólica, AMDEE - Asociación Mexicana de Energía Eólica, https://amdee.org/es_es/07-proyectos//, 2024.

[3] Zhang, F., Chen, M., Zhu, Y., Zhang, K., y Li, Q., A Review of Fault Diagnosis, Status Prediction, and Evaluation Technology for Wind Turbines, Energies, Vol. 16, No. 3, 2023. https://www.mdpi.com/1996-1073/16/3/1125.

[4] Wang C-Y, Bochkovskiy A y Liao H-YM, YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors, 2022 arXiv (Cornell University), https://arxiv.org/abs/2207.02696.

[5] Sampieri RH, Collado CF y Lucio PB, Metodología de La Investigación, McGraw-Hill (2010).

[6] Goodfellow, I., Bengio, Y., y Courville, A., Deep Learning. Cambridge, MA: MIT Press, 2016. http://www.deeplearningbook.org.

[7] Kuncheva, L. I., Combining Pattern Classifiers: Methods and Algorithms, 1st ed. Hoboken, NJ: Wiley, 2004, pp. 15-18.