UNIVERSIDAD DEL VALLE DE GUATEMALA

CC3045 - Inteligencia Artificial

Sección 10

Ing. Alberto Suriano

Proyecto Final - Redes Neuronales

Diego Leiva #21752

Maria Ramirez #21342

Gustavo Gonzalez #21438

Jose Pablo Orellana #21970

Guatemala, 20 de mayo del 2,024

Descripción del problema

Las problemáticas que se tenían que afrontar en este proyecto incluyen desafíos técnicos y prácticos específicos que son fundamentales para mejorar procesos en sectores clave como la manufactura, la seguridad pública y el sector de la salud.

En la industria manufacturera, la principal dificultad radica en la necesidad de realizar inspecciones visuales automatizadas que sean capaces de identificar defectos imperceptibles al ojo humano. Esto no solo implica una mejora en la calidad de los productos finales, sino que también ofrece una oportunidad para optimizar la eficiencia operativa y reducir costos significativos asociados a defectos de fabricación y desperdicio de material.

En cuanto a la seguridad pública, el reto consiste en desarrollar sistemas que puedan monitorear eficazmente espacios urbanos y públicos para detectar comportamientos o actividades sospechosas en tiempo real.

En el sector de la salud, el desafío se centra en el análisis preciso y rápido de imágenes médicas. Estos sistemas deben ser capaces de identificar signos tempranos de enfermedades a partir de imágenes diagnósticas como radiografías o análisis de muestras de tejido, lo cual es crucial para mejorar las tasas de diagnóstico temprano y, por ende, los resultados de tratamientos médicos.

Cada uno de estos desafíos refleja la naturaleza de la problemática que nuestro proyecto busca abordar, aunque en un contexto distinto. Así como en los sectores industriales, de seguridad y de salud, nuestro proyecto enfrenta la necesidad de desarrollar soluciones tecnológicas robustas capaces de adaptarse a variaciones significativas en los datos de entrada. Similar a estas aplicaciones, nuestro enfoque también requiere sistemas que minimicen los errores y maximicen la precisión, dado que una identificación incorrecta podría llevar a interpretaciones erróneas o decisiones inadecuadas.

Análisis

El proyecto se enfoca en identificar de forma automática y precisa ciertas especies de dinosaurios en base a imágenes digitales, este problema requiere de una solución que pueda gestionar tanto la complejidad visual a las imágenes de los dinosaurios como las expectativas de aplicaciones prácticas mencionadas anteriormente, como seguridad, automatización y salud.

Dentro de los factores clave dentro del problema tenemos la variabilidad visual, ya que las especies de dinosaurios, incluso dentro de una misma categoría, pueden exhibir una considerable diversidad en términos de tamaño, color, y forma. Esto plantea un desafío significativo en términos de reconocimiento automático, ya que el sistema debe ser capaz de generalizar a partir de características comunes sin confundirse con variaciones intraespecíficas.

Otro factor importante puede ser la calidad y diversidad de de datos, debido a que el modelo depende de un conjunto de datos que consiste en 100 imágenes por cada una de las 16 especies de dinosaurios. La calidad de estas imágenes, su resolución, iluminación y el ángulo desde el cual fueron tomadas son cruciales para el entrenamiento eficaz de cualquier modelo de reconocimiento de imágenes.

En este proyecto y dadas las características que presenta se pueden pensar en diversas soluciones como modelos basados en características, redes neuronales o ensamble de modelos.

Propuesta de solución

Para poder abordar el problema de identificar automáticamente especies de dinosaurios a partir de imágenes digitales, proponemos una solución integral que capitaliza las fortalezas de las redes neuronales convolucionales para el procesamiento y clasificación de imágenes. Esta propuesta se fundamenta en la capacidad demostrada de las redes neuronales para manejar la variabilidad visual y la complejidad inherente a las imágenes de dinosaurios, proporcionando una solución robusta y escalable que puede adaptarse a las demandas de aplicaciones prácticas en otros ámbitos de la vida cotidiana.

El utilizar redes neuronales tiene beneficios a la hora de realizar el software que se encargará de reconocer imágenes. Dentro de estos beneficios tenemos los siguientes:

Extracción Automática de Características: Las redes neuronales son excepcionales en la identificación y extracción automática de características importantes desde imágenes crudas. A diferencia de los métodos tradicionales, que requieren selección manual y diseño de características, las redes neuronales aprenden estas características de manera autónoma durante el proceso de entrenamiento, adaptándose específicamente a las complejidades del conjunto de datos en cuestión.

Generalización Sobre Variaciones de Imágenes: Las redes neuronales convolucionales manejan bien las variaciones en las imágenes, como cambios en la orientación, escala y deformaciones parciales, gracias a su arquitectura jerárquica que procesa la información en múltiples niveles de abstracción. Esto significa que el modelo puede reconocer un dinosaurio en una imagen nueva que puede no ser idéntica a las imágenes de entrenamiento.

Eficiencia en Tiempo Real: Una vez entrenadas, las redes neuronales pueden realizar predicciones en fracciones de segundo, lo que las hace ideales para aplicaciones en tiempo real.

Robustez y Escalabilidad: Las CNN son inherentemente robustas a las perturbaciones en los datos de entrada y pueden escalar eficientemente a grandes conjuntos de datos y problemas complejos. A medida que se disponga de más imágenes de dinosaurios,

el modelo puede continuar mejorando sin necesidad de cambios significativos en la arquitectura.

Descripción de la solución

Como se mencionó anteriormente la solución a la problemática para identificar automáticamente imágenes digitales se basa en la implementación de una red neuronal convolucional.

Primero, se llevó a cabo la preparación de datos utilizando imágenes de un dataset que se fue generando a lo largo del proyecto. A estas imágenes se les aplicaron modificaciones previas, como agregar padding, convertirlas al formato .bmp y escalarlas adecuadamente. Además, se emplearon generadores de imágenes para manejar el flujo de datos durante el entrenamiento, aprovechando la técnica de Data Augmentation. Esta técnica modifica ligeramente la imagen original para crear distintas versiones de la misma, incrementando así los datos disponibles para entrenamiento, validación y pruebas, y asegurando que las imágenes se procesen eficientemente en lotes. Esto facilita el manejo de grandes volúmenes de datos sin sobrecargar la memoria.

El aumento de datos mediante Data Augmentation incluye técnicas como rotación, cambio de escala y volteo horizontal, las cuales se emplean para ampliar la diversidad del conjunto de entrenamiento. Estas técnicas mejoran la robustez del modelo al hacerlo más capaz de generalizar frente a nuevas y variadas imágenes.

Posteriormente se diseñó la arquitectura de la red neuronal, la cual se compone de múltiples capas convolucionales con reguladores L2 y capas de dropout, estas últimas destinadas a mitigar el sobreajuste. Posteriormente, se incorporaron capas de pooling para reducir la dimensionalidad de los datos, manteniendo las características más relevantes. Se optó por emplear activaciones ReLU debido a su eficacia para introducir no linealidad, lo que facilita al modelo el aprendizaje de patrones complejos en los datos. Finalmente, se incluyó una capa aplanadora que convierte la matriz resultante a una sola dimensión, seguida de una capa densa para realizar la clasificación final y generar el output de probabilidades.

Luego se hace una optimización y ajuste de hiperparámetros, en este se utiliza un afinador automático (Keras Tuner) para explorar un rango amplio de configuraciones y determinar la mejor combinación de hiperparámetros como el número de filtros, la tasa de dropout y los coeficientes de regularización.

Una vez tenemos eso se hace una evaluación del modelo, el rendimiento de nuestro modelo, se evalúa utilizando métricas estándar como la precisión, el recall y el F1-score. Además, se emplean matrices de confusión para visualizar el desempeño del modelo en cada clase. Las visualizaciones de las predicciones correctas e incorrectas proporcionan insights

adicionales sobre cómo el modelo está realizando predicciones y en qué áreas necesita mejoras.

Para luego culminar se da la implementación y pruebas, se hacen pruebas iniciales con el conjunto de validación para ajustar parámetros y evaluar la generalización del modelo. Luego el modelo se prueba exhaustivamente con un conjunto de test independiente para asegurar que su rendimiento es robusto y consistente.

Herramientas aplicadas

1. TensorFlow
    a. Descripción:
    TensorFlow es una plataforma de código abierto desarrollada por Google Brain para el aprendizaje automático. Facilita la creación de modelos de machine learning y deep learning mediante una estructura flexible y eficiente que soporta una amplia gama de algoritmos y aplicaciones.
    b. Características clave:
        i. Eficiencia: Es altamente optimizado para la computación en múltiples CPUs y GPUs.
        ii. Flexibilidad: Permite la construcción de modelos complejos de machine learning con un control granular sobre las operaciones matemáticas y el flujo de datos.
        iii. Escalabilidad: Adecuado para aplicaciones que van desde experimentos pequeños hasta grandes despliegues en producción.

2. Keras
    a. Descripción:
    Keras es una API de alto nivel para construir y entrenar modelos de aprendizaje profundo. Es compatible con múltiples backends, incluido TensorFlow, y se integra estrechamente con TensorFlow 2.x.
    b. Características Clave:
        i. Simplicidad y Facilidad de Uso: Keras proporciona una interfaz simple y coherente para construir modelos de deep learning de manera rápida y eficiente.
        ii. Modularidad: Los modelos en Keras se construyen como una secuencia de módulos independientes que pueden combinarse fácilmente.
        iii. Extensibilidad: Permite la creación de capas personalizadas y modelos complejos mediante la subclasificación de las clases base.

3. Componentes Utilizados de Keras y TensorFlow:
    a. Modelos y Capas:
        - Sequential: Modelo secuencial para apilar capas linealmente.

- Conv2D: Capas convolucionales para la extracción de características de las imágenes.
- MaxPooling2D: Capas de agrupamiento para la reducción de dimensionalidad.
- Dense: Capas densamente conectadas para la clasificación.
- Dropout: Capas de regularización para prevenir sobreajuste.
- BatchNormalization: Capas de normalización para acelerar el entrenamiento y mejorar la estabilidad.
- Flatten: Capas para aplanar la entrada antes de la capa completamente conectada.
- Activation: Aplicación de funciones de activación como ReLU.

b. Optimización y Pérdida:
   - Adam: Optimizador adaptativo para la actualización eficiente de los pesos.
   - CategoricalCrossentropy: Función de pérdida para clasificación categórica multiclase.

c. Callbacks:
   - EarlyStopping: Callback para detener el entrenamiento cuando no hay mejora en la pérdida.
   - ReduceLROnPlateau: Callback para reducir la tasa de aprendizaje cuando no hay mejora en la pérdida.

d. Regularización:
   - l2: Regularización L2 para prevenir el sobreajuste al penalizar los pesos grandes.

e. Keras Tuner:
   - Descripción: Librería para la optimización de hiper parámetros mediante métodos como Hyperband.
   - Uso: Afinamiento de hiper parámetros como el número de filtros en las capas convolucionales, la tasa de regularización y la tasa de dropout.

4. ImageDataGenerator (Keras):
   a. Descripción:
      Herramienta para el aumento y procesamiento de datos de imágenes.
   b. Uso en el Proyecto:
      Generación de lotes de datos de entrenamiento y validación con aumentos de datos para mejorar la robustez del modelo.

5. os:
   a. Descripción:
      Librería estándar de Python para el manejo de rutas y operaciones del sistema de archivos.

b. Uso en el Proyecto:
   Manejo de rutas para la carga y almacenamiento de datos.

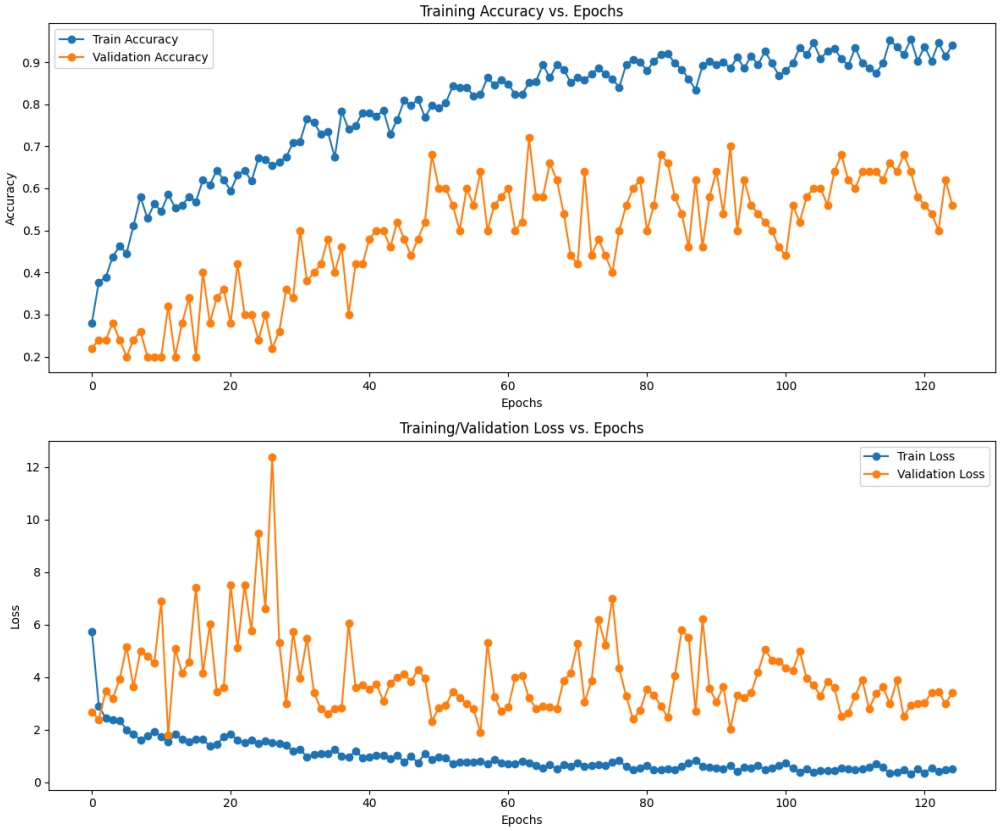
6. numpy:
   a. Descripción:
      Librería para operaciones matemáticas y manejo de matrices en Python.
   b. Uso en el Proyecto:
      Visualización de resultados de entrenamiento y validación, y gráficos de precisión y pérdida.

7. seaborn:
   a. Descripción:
      Librería de visualización de datos basada en matplotlib.
   b. Uso en el Proyecto:
      Mejora de gráficos y visualizaciones estadísticas.

8. sklearn.metrics:
   a. Descripción:
      Módulo de scikit-learn para la evaluación de métricas.
   b. Uso en el Proyecto:
      Generación de reportes de clasificación y matrices de confusión.

9. PIL (Python Imaging Library):
   a. Descripción:
      Librería para la manipulación y procesamiento de imágenes en Python.
   b. Uso en el Proyecto:
      Carga y manipulación de imágenes para la visualización de predicciones correctas.

Resultados (métricas)



Gráfica 1. Desempeño del entrenamiento

	precision	recall	f1-score	support
Brachiosaurus	0.89	0.80	0.84	20
Pteranodon	0.88	0.70	0.78	20
Stegosaurus	0.81	0.65	0.72	20
Triceratops	0.59	0.80	0.68	20
Tyrannosaurus	0.65	0.75	0.70	20
accuracy			0.74	100
macro avg	0.76	0.74	0.74	100
weighted avg	0.76	0.74	0.74	100

Figura 1. Reporte de Clasificación

Interpretación del desempeño del entrenamiento:

- Convergencia del Modelo: Los valores de precisión y pérdida tanto para el entrenamiento como para la validación muestran una tendencia a la convergencia, lo que sugiere que el modelo está aprendiendo correctamente sin sobre ajustarse. La precisión del entrenamiento alcanza un valor muy alto, cerca del 90%, y la precisión de validación es un poco más media alcanzando aproximadamente un 60%.
- Sobreajuste: La diferencia entre la precisión de entrenamiento y la de validación es significativamente distinta, lo que puede indicar que el modelo no está sobre ajustado.

Además, las pérdidas de entrenamiento son bajas y convergen hacia valores similares, lo que refuerza esta conclusión.

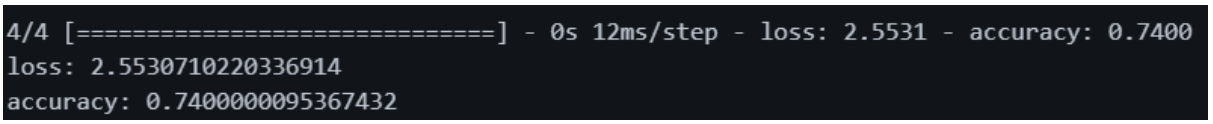


Figura 2. Resultados de la evaluación del modelo

Interpretación de los resultados de la evaluación del modelo:

1. Precisión:
    - 74% de precisión indica que el modelo clasifica correctamente aproximadamente tres cuartas partes de las imágenes de prueba. Esta es una buena precisión, especialmente considerando la posible complejidad y variabilidad en las imágenes de especies de dinosaurios.
2. Pérdida:
    - Pérdida de 2.5531: La pérdida parece ser relativamente alta en comparación con la precisión. En muchos casos, una alta pérdida combinada con una buena precisión puede sugerir que el modelo hace predicciones correctas pero con baja confianza. Esto podría ser debido a la función de pérdida utilizada y la manera en que las predicciones están distribuidas.

Posibles Razones y Consideraciones:

1. Complejidad del Problema:
   Clasificar imágenes de especies de dinosaurios es un problema complicado debido a las variaciones significativas entre las especies y las similitudes visuales que algunas especies pueden compartir.

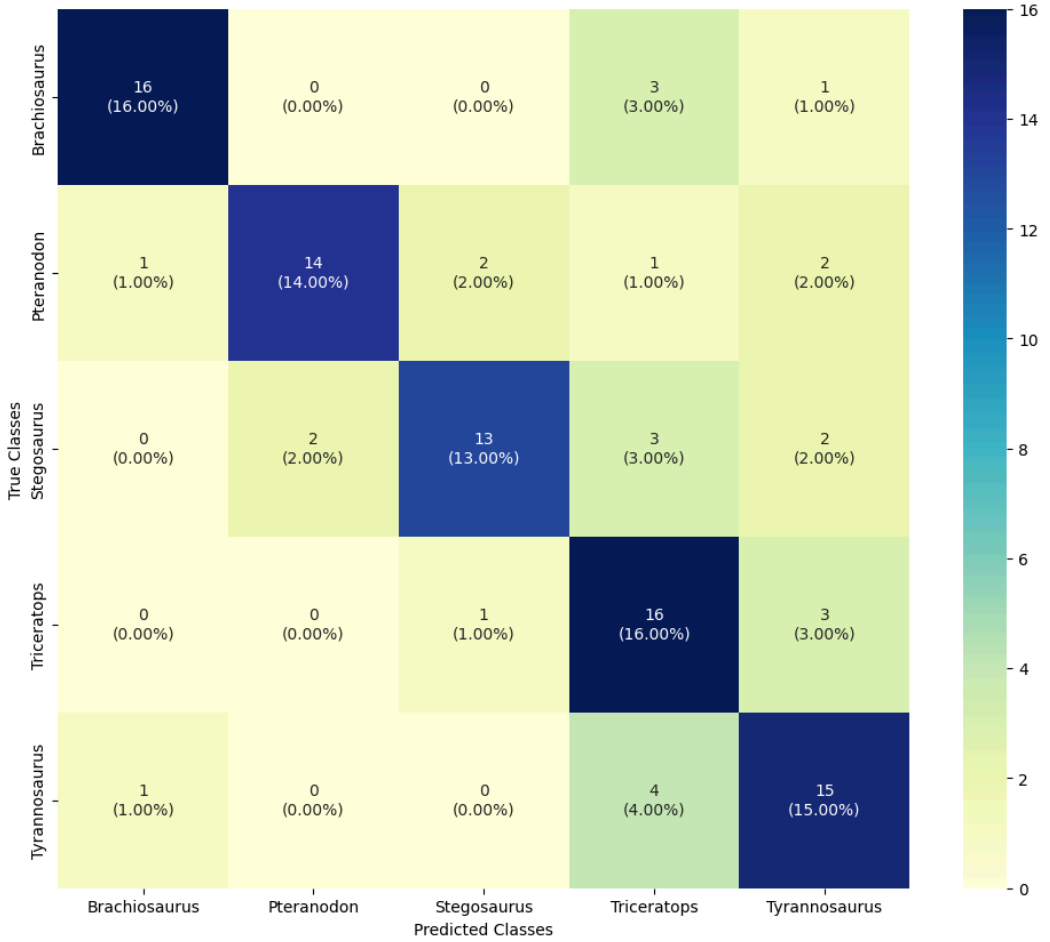
2. Datos de Entrenamiento:
   Cantidad de Datos: Un mayor número de imágenes de entrenamiento podría mejorar la precisión y reducir la pérdida, permitiendo al modelo aprender mejor las características distintivas de cada especie.

   Calidad de Datos: Asegurarse de que las imágenes estén bien etiquetadas y sean representativas de las diferentes clases de dinosaurios.

3. Modelo y Arquitectura:
   Profundidad del Modelo: Un modelo más profundo o con más parámetros podría capturar mejor las características complejas de las imágenes.

   Regularización: Asegurar una adecuada regularización para evitar el sobreajuste y mejorar la capacidad del modelo para generalizar.

4. Optimización y Hiper parámetros:
   Tasa de Aprendizaje: Ajustar la tasa de aprendizaje podría mejorar la convergencia del modelo y reducir la pérdida.
   Afinación de Hiper Parámetros: Continuar utilizando herramientas como Keras Tuner para encontrar los mejores hiper parámetros para el modelo.



Gráfica 2. Matriz de Confusión

Interpretación de la Matriz de Confusión:

1. Brachiosaurus:
   - Verdaderos Positivos (TP): 16 (16.00%)
   - Falsos Negativos (FN): 4 (3 para Triceratops, 1 para Tyrannosaurus)
   - Falsos Positivos (FP): 1 (clasificado incorrectamente como Triceratops)
   - Buen desempeño en términos de precisión y recall.

2. Pteranodon:
   - Verdaderos Positivos (TP): 14 (14.00%)
   - Falsos Negativos (FN): 5 (1 para Brachiosaurus, 2 para Stegosaurus, 2 para Tyrannosaurus)

- Falsos Positivos (FP): 0
- El modelo identifica bien a los Pteranodon, pero hay algunas confusiones.

3. Stegosaurus:
    - Verdaderos Positivos (TP): 13 (13.00%)
    - Falsos Negativos (FN): 7 (2 para Pteranodon, 3 para Triceratops, 2 para Tyrannosaurus)
    - Falsos Positivos (FP): 2
    - El modelo tiende a confundir Stegosaurus con otras especies.

4. Triceratops:
    - Verdaderos Positivos (TP): 16 (16.00%)
    - Falsos Negativos (FN): 4 (1 para Stegosaurus, 3 para Tyrannosaurus)
    - Falsos Positivos (FP): 1
    - Similar a Brachiosaurus, con buena precisión y recall.
5. Tyrannosaurus:
    - Verdaderos Positivos (TP): 15 (15.00%)
    - Falsos Negativos (FN): 5 (1 para Brachiosaurus, 2 para Stegosaurus, 2 para Triceratops)
    - Falsos Positivos (FP): 1
    - Buen desempeño general, aunque con algunas confusiones.

Discusión de Resultados:

1. Precisión General:
   La mayoría de las clases muestran una precisión y recall bastante altos, lo cual indica que el modelo está funcionando razonablemente bien en identificar correctamente las especies de dinosaurios.

2. Confusiones Comunes:
   Las confusiones entre clases como Triceratops y Tyrannosaurus o entre Pteranodon y Stegosaurus podrían deberse a características visuales similares en las imágenes utilizadas.

   Stegosaurus parece ser la clase más problemática, con la mayor cantidad de confusiones (FN y FP).

3. Mejoras Potenciales:
   Aumento de Datos: Aumentar el conjunto de datos para tener más ejemplos de cada clase puede ayudar a mejorar nuestra precisión y reducir las confusiones.

   Afinación de Hiper Parámetros: Seguir ajustando los hiper parámetros del modelo para mejorar su capacidad de generalización.

Arquitectura del Modelo: Experimentar con arquitecturas más complejas que puedan captar mejor las diferencias sutiles entre las clases.

Conclusión

El modelo ha mostrado una buena capacidad de clasificación con una precisión del 74%, lo cual es bastante sólido para un problema de clasificación de imágenes complejas como especies de dinosaurios. Sin embargo, la pérdida relativamente alta sugiere que hay margen para mejorar. Se pueden considerar los siguientes pasos para futuras mejoras:

- Aumentar el Conjunto de Datos: Obtener más datos de entrenamiento y asegurarse de que sean de alta calidad.
- Mejorar la Arquitectura del Modelo: Experimentar con arquitecturas más complejas y profundizar en la afinación de hiper parámetros.
- Ajustar la Regularización y Optimización: Asegurarse de que la regularización y la tasa de aprendizaje estén adecuadamente ajustadas.

Enlaces

Link al repositorio en Github
Link al video
Link a la presentación

Bibliografía

TensorFlow. (s.f.). Guía de Keras. TensorFlow. Recuperado el 20 de mayo de 2024, de https://www.tensorflow.org/guide/keras?hl=es

Gulli, A., & Pal, S. (2017). Deep learning with Keras. Packt Publishing Ltd.

Dremio. (s.f.). ReLU Activation Function. Recuperado el 20 de mayo de 2024, de https://www.dremio.com/wiki/relu-activation-function/

Agarap, A. F. (2018). Deep learning using rectified linear units (relu). arXiv preprint arXiv:1803.08375.

Rasamoelina, A. D., Adjailia, F., & Sinčák, P. (2020, January). A review of activation function for artificial neural network. In 2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI) (pp. 281-286). IEEE.

Al-Saffar, A. A. M., Tao, H., & Talab, M. A. (2017, October). Review of deep convolution neural network in image classification. In 2017 International conference on radar, antenna, microwave, electronics, and telecommunications (ICRAMET) (pp. 26-31). IEEE.