Construcción y análisis de desempeño de una Red Neuronal Convolucional (CNN)

Juan Carlos Martínez Zacarías

Resumen

Las Redes Neuronales Convolucionales (CNN) son redes que conforman un sistema de procesamiento computacional enfocado a imágenes. Algunas de sus características más notables son el hecho de integrar capas de convolución, de Pooling y densas, principalmente, por las que un conjunto de datos fluye y les permite adquirir aprendizajes para encontrar las características que identifican a un tipo de imagen. Algunas arquitecturas están constituidas por varias decenas de capas, como la ResNet-50, con hasta 48 capas convolucionales. Sin embargo, estos 3 tipos de capas son tan solo algunas de las variables que definen un modelo CNN, puesto que, tan solo en la capa convolucional, el diseñador debe elegir una cantidad de filtros y el tamaño de estos, entre otros parámetros. Además, en el entorno de entrenamiento es posible variar la forma en la que una red se ajusta internamente, la cantidad de veces que realiza el entrenamiento, el tamaño del lote de datos que procesa en cada iteración, y el preprocesamiento aplicado a estos, por mencionar algunos ejemplos.

Dada la cantidad de variables, es importante entender los efectos que estas tienen en el desempeño de un modelo, pero también en su eficiencia.

Palabras clave: CNN, arquitectura, hiper parámetros, preprocesamiento de datos.

Abstract

Convolutional Neural Networks (CNN) are networks that make up a computational processing system focused on image processing. Some of their most notable features are the fact that they integrate convolution, pooling and dense layers through which a dataset flows and allows them to learn how to find the characteristics that identify an image class. Some architectures are made up of dozens of layers, such as ResNet-50, with up to 48 convolutional layers. However, these 3 types of layers are just some of the variables that define a CNN model since just in the convolutional layer the designer must choose a number of filters and their size, among other parameters. Furthermore, in the training environment it is possible to

vary the way in which a network is adjusted internally, the number of times it performs training, the size of the batch of data it processes in each iteration, and the preprocessing applied to the images, which are just a few examples.

Given the number of variables, it is important to understand the effects they have on the performance of a model, but also on its efficiency.

Keywords: CNN, architecture, hyperparameters, data preprocessing.

Introducción

Las Redes Neuronales Artificiales (ANNs) son sistemas de procesamiento computacional inspirados por sistemas nerviosos biológicos. Este tipo de redes comprenden un alto número de nodos interconectados conocidos como neuronas, las cuales aprenden de forma colectiva dada una entrada con el fin de optimizar la salida final (O'shea & Nash, 2015).

Su estructura básica puede ser modelada con una capa de entrada que recibe un vector multidimensional. Los datos serán distribuidos a capas ocultas y estas tomarán decisiones a partir de la capa anterior, ponderando cómo un cambio estocástico dentro de sí mismas perjudica o mejora el resultado final. En esto consiste el proceso de aprendizaje y, al tener múltiples capas ocultas apiladas, se está hablando de aprendizaje profundo.

De este modo, las Redes Neuronales Convolucionales (CNN) son análogas a las ANNs tradicionales, siendo la única diferencia notable el hecho de que una red convolucional es principalmente utilizada en el campo del reconocimiento de patrones dentro de las imágenes.

En las CNNs es posible encontrar tres tipos de capas principales: capa convolucional, en la que en múltiples ocasiones se toman grupos de píxeles cercanos de la imagen de entrada para operar matemáticamente contra una pequeña matriz llamada kernel y “dibujar” ciertas características de la imagen original; capa de pooling, que busca reducir gradualmente la dimensionalidad de la representación; y la capa densa, que contiene neuronas conectadas directamente a las neuronas de las dos capas adyacentes, encargándose de la tarea de clasificación.

Otro concepto que también cabe señalar es el sobreajuste, que ocurre cuando una red es incapaz de aprender efectivamente debido a que se ajusta exactamente a sus datos de entrenamiento y da malos resultados para los datos nuevos.

Es importante comprender el funcionamiento de las herramientas que utilizamos, en este caso, para el procesamiento de imágenes, así como las situaciones a las que pueden estar expuestas y que afectan su rendimiento, dando paso a una toma de decisiones encaminada a la construcción de una arquitectura robusta que requiera un poder computacional y un tiempo de entrenamiento razonables.

Algunas de las arquitecturas conocidas son: LeNet, entrenada para el reconocimiento de dígitos, con 5 capas convolucionales y capas densas individuales; AlexNet, con alrededor de 5 capas convolucionales y 2 capas densas para el aprendizaje de características; VGG-16, con 13 capas convolucionales y 3 capas densas, alcanzando 92.7% de precisión en el conjunto de datos ImageNet; y ResNet-50, con 48 capas convolucionales, además de 1 max pool y 1 average pool (Shyam, 2021).

El propósito de este trabajo es construir una CNN partiendo de una arquitectura sencilla para aplicar modificaciones y observar sus cambios con respecto al desempeño, buscando mejorar tanto su precisión como su pérdida en la clasificación de imágenes analizando las implicaciones computacionales, además de considerar conceptos como el sobreajuste.

Metodología

La investigación consiste en 4 experimentos. En el primero se construyó una red neuronal convolucional con una arquitectura básica, mientras que en el segundo se realizaron modificaciones a dicha arquitectura. Posteriormente, las modificaciones implementadas giraron en torno al ajuste de hiper parámetros y, finalmente, se aplicaron algunas técnicas avanzadas.

Para la construcción del modelo y sus diferentes versiones se utilizó el marco de trabajo TensorFlow/Keras con los módulos referentes al modelado de capas para datos de dos dimensiones (Conv2D, MaxPooling2D), correspondientes a la dimensionalidad de las imágenes, al igual que otros como Dense, Flatten y Dropout.

El conjunto de datos utilizado es CIFAR-10 (TensorFlow, 2022), conformado por 60000 imágenes en color de 32x32 distribuidas en 10 clases, con 6000 imágenes por clase. Del total, 50000 son de entrenamiento y 10000 de prueba.

Para cada experimento se recolectaron la pérdida y la precisión del modelo en tanto al conjunto de datos de entrenamiento y de validación a lo largo de las épocas de entrenamiento, siendo el conjunto de validación construido a partir de los datos de prueba.

Experimentos

La configuración base del entorno de entrenamiento fue de una tasa de aprendizaje (Learning Rate) de 0.001, un tamaño de lote (Batch Size) de 16 y 20 épocas de entrenamiento.

Experimento 1:

El modelo base (v1) tiene una arquitectura conformada por una capa convolucional de 32 filtros, con un tamaño de kernel de 3x3 y ReLU como función de activación. Tiene una capa de MaxPooling de 2x2, una capa de aplanamiento, una capa densa de 128 unidades con activación ReLU y una más de salida con activación SoftMax.

Experimento 2:

El modelo base fue modificado 4 veces para analizar los efectos de una variedad de cambios en la arquitectura:

- Primera modificación (v2): se añadió una capa convolucional de 32 filtros, tamaño de kernel de 3x3 y activación ReLU tras la primera capa convolucional y antes de la capa de MaxPooling. También se añadió un nuevo bloque con dos capas convolucionales de 64 filtros, kernel de 3x3 y activación ReLU, seguidas de otra capa de MaxPooling de 2x2.
- Segunda modificación (v3): para cada bloque de capas se añadió una capa de Dropout, de modo que, tras cada MaxPooling, y antes de la capa densa de salida, existen capas Dropout del 20%.
- Tercera modificación (v4): para esta versión del modelo, la cantidad de filtros de todas las capas convolucionales fue reducida a la mitad. Así, las primeras dos capas resultaron con 16 filtros, y la siguientes con 32 filtros.

- Cuarta modificación (v5): para observar el efecto contrario a la versión 4 del modelo, en esta cuarta modificación todas las capas convolucionales tienen 64 filtros.

Experimento 3:

Para este experimento se tomaron el modelo base y la quinta versión del modelo para observar cómo influye la variación de hiper parámetros como la tasa de aprendizaje, el tamaño del lote y la cantidad de épocas de entrenamiento.

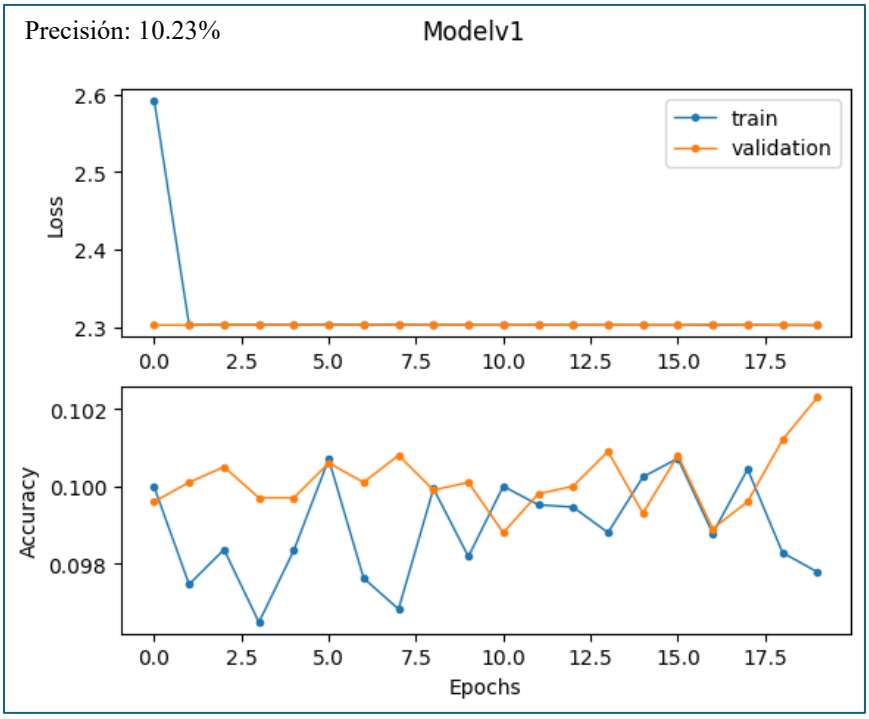
- Tasa de aprendizaje: para el modelo base se consideraron los valores de 0.01 y 0.1, mientras que para la versión 5 del modelo se tomaron los valores de 0.0003 y 0.004.
- Tamaño del lote: tanto el modelo base como su quinta versión fueron entrenados con un Batch Size de 8 y 32.
- Cantidad de épocas. la cantidad de épocas de entrenamiento se amplió a 40 para ambos modelos.

Experimento 4:

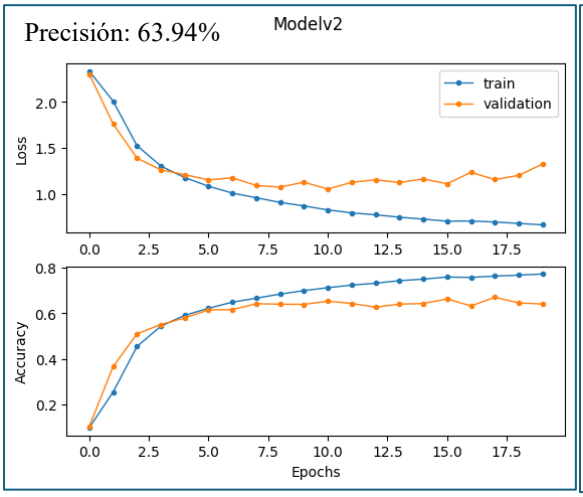
En este experimento se implementó la técnica de aumentación de datos (Data Augmentation) sobre el conjunto de entrenamiento. Se realizó una aumentación sencilla con un 10% de traslado aleatorio horizontal y vertical en las imágenes, junto a un volteado aleatorio horizontal. El nuevo conjunto de datos fue utilizado para entrenar el modelo base y la quinta versión del modelo.

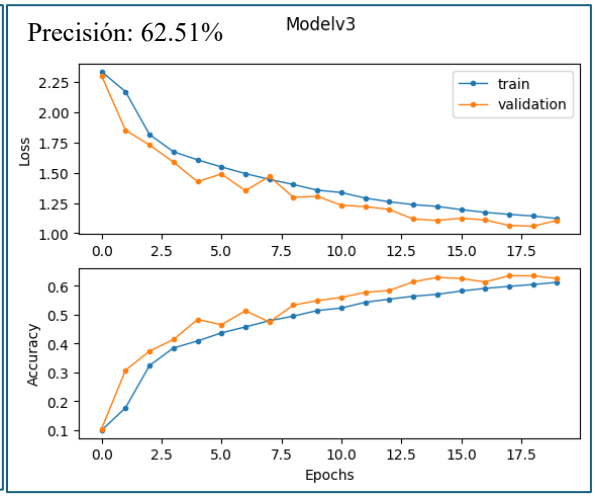
Resultados

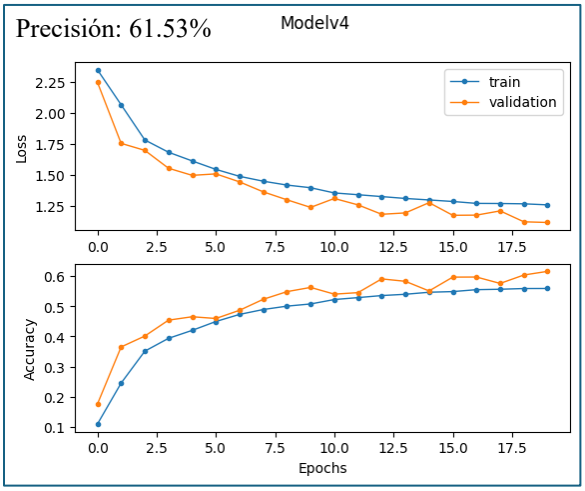
Experimento 1:

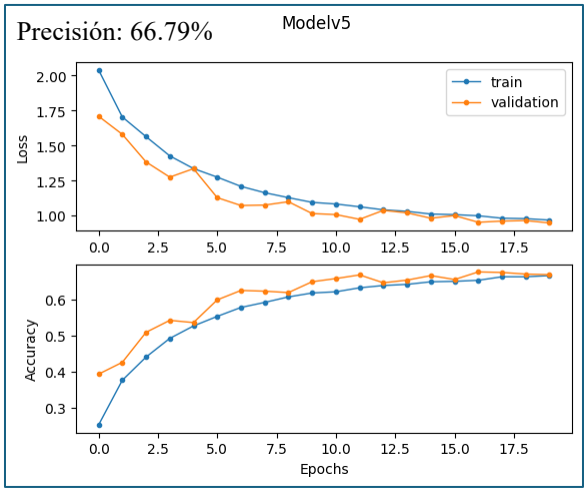


Experimento 2:

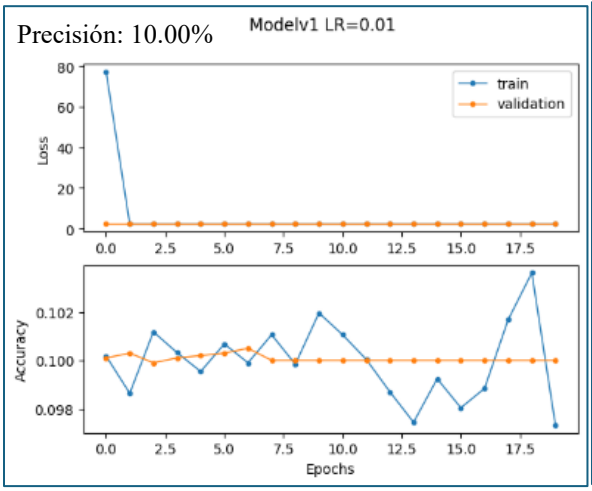


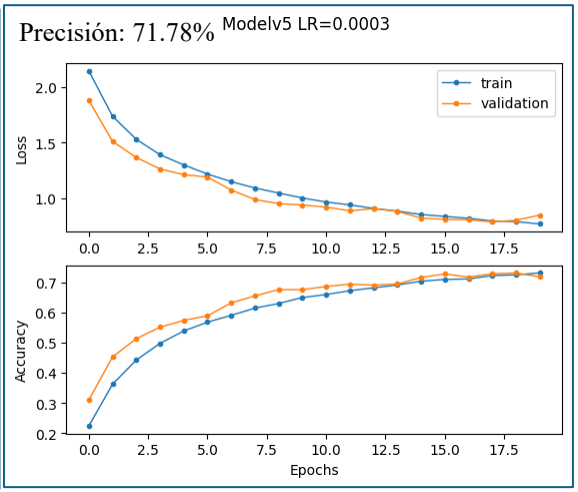


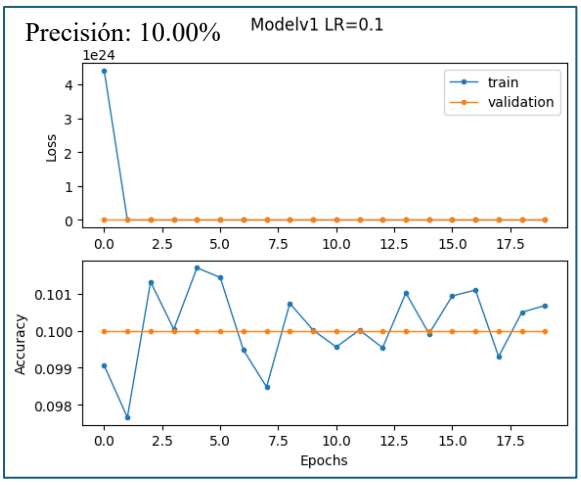


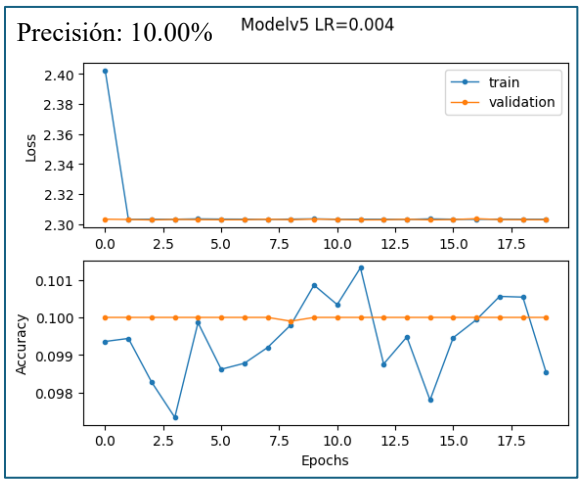


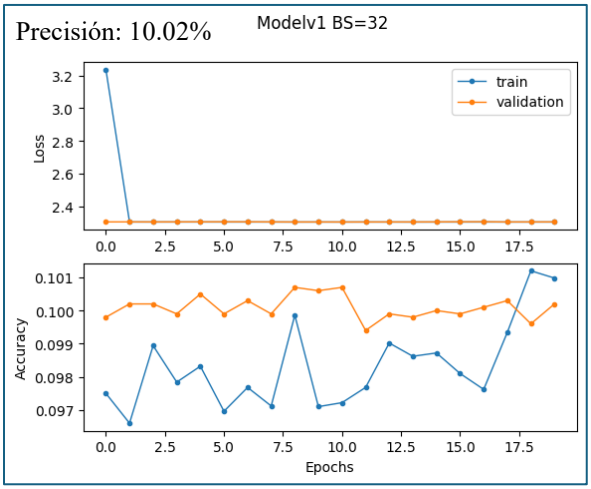
Experimento 3:

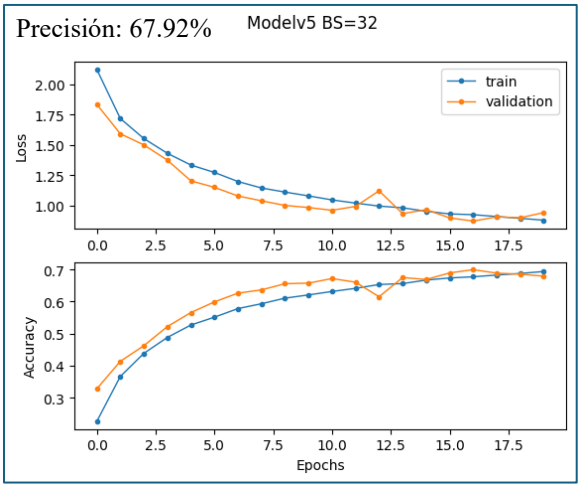


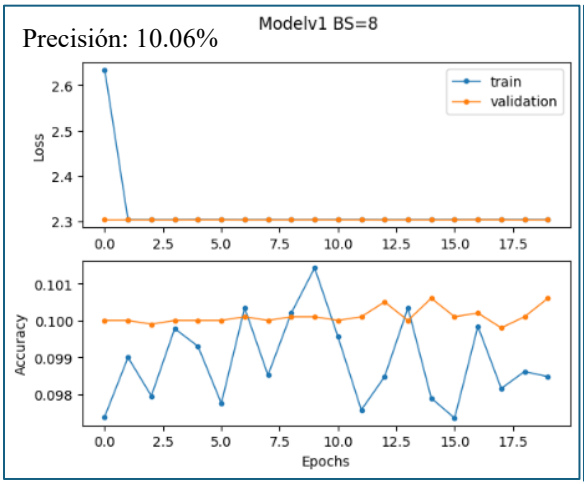


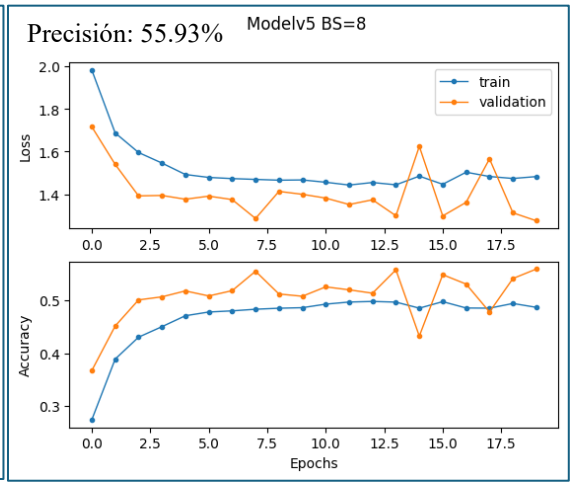


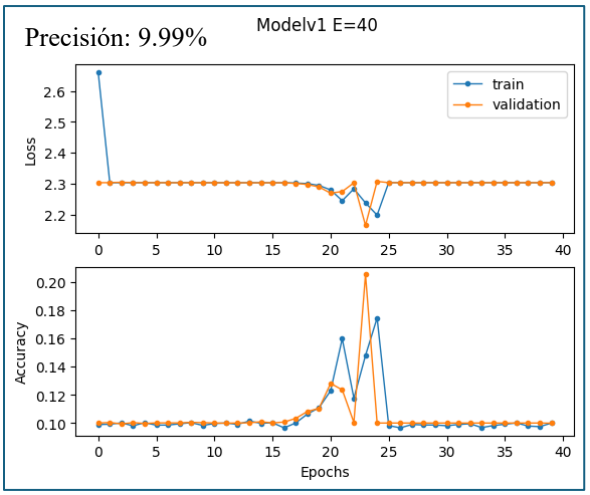


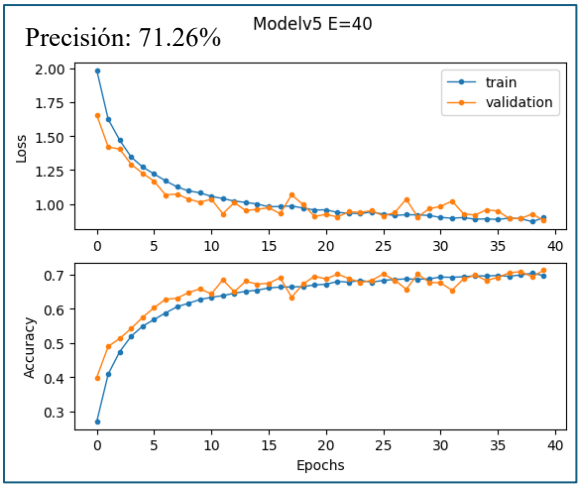




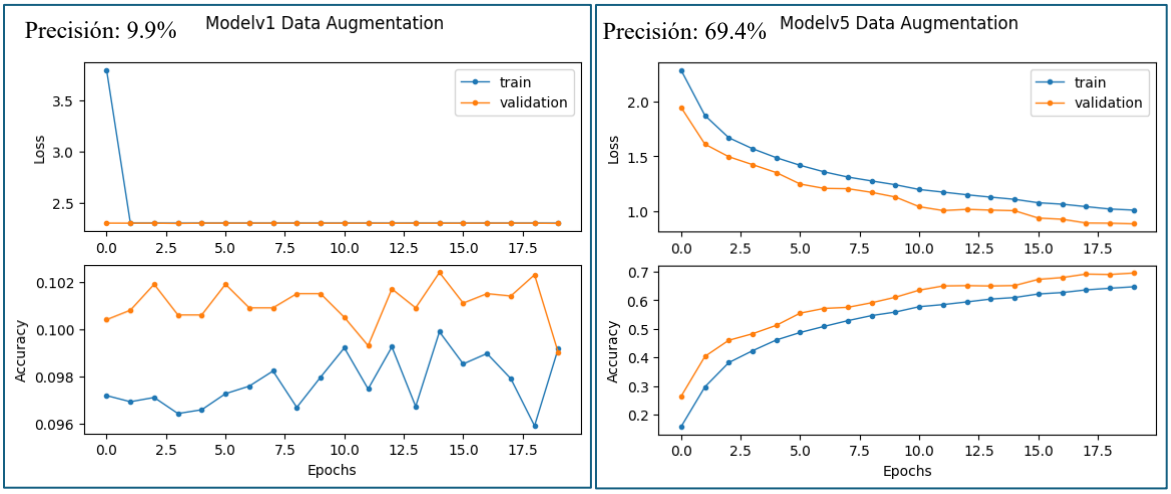








Experimento 4:



Discusión

En términos generales, el modelo base parece ser bastante simple como para aprender de las características que recibe del conjunto de datos dado que su precisión se mantiene aproximadamente en 10% a pesar de las modificaciones internas (arquitectura) y externas (hiper parámetros, técnicas avanzadas).

Con respecto a las demás versiones, si bien se logró alrededor de un 60% de precisión, es notable el comportamiento que experimentó el modelo en su segunda versión puesto que la precisión de entrenamiento empezó a diferir considerablemente de la precisión de validación, reflejando indicios de sobreajuste, no obstante, al agregar capas de Dropout, como podría esperarse, se consiguió corregir el problema al mismo tiempo que el porcentaje subió a 65 con apoyo de un “barrido” más minucioso sobre las imágenes al haber más filtros.

Por su parte, someter al modelo a un ajuste cuidadoso de pesos a través de una tasa de aprendizaje baja, así como a un procesamiento de imágenes más exhaustivo tanto por medio de lotes más grandes como por procedimientos que incrementan la cantidad y variabilidad del conjunto de datos, además de un mayor número de iteraciones, parecen ser ciertamente beneficiosos para su desempeño, llegando hasta un 70% de precisión.

Conclusión

Es importante comprender el panorama del procesamiento de imágenes para poder diseñar redes neuronales que reciban y aprendan de un determinado conjunto de datos. Uno de los

acercamientos altamente conocidos son las redes del tipo convolucional, que implican procesos que buscan “mapear” de la mejor manera posible el conjunto en cuestión para posteriormente reducir su complejidad y al final lograr una clasificación con un alto nivel de precisión.

El verdadero reto surge cuando dichos diseños son sometidos a entrenamiento y pruebas, puesto que puede tratarse de un modelo poco o muy eficiente, con un mal o buen desempeño.

Considerando que en los experimentos realizados solamente hubo mejoras al aumentar la complejidad del modelo mismo y al someterlo a procedimientos computacionalmente más costosos, el encontrar una configuración óptima implica una serie de decisiones que evidencia la necesidad de establecer un equilibrio entre el desempeño y la complejidad de una red de aprendizaje profundo e, inclusive, de encontrar mejores alternativas a este tipo de modelos (Thompson et. al., 2020).

Código fuente

https://github.com/JCarlosMtzZ/CNN1.git

Referencias

O'shea, K., & Nash, R. (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv*:1511.08458.

Shyam, R. (2021). Convolutional neural network and its architectures. *Journal of Computer Technology & Applications,* 12(2), 6-14.

TensorFlow. (2022, diciembre 6). *Cifar10.* TensorFlow. https://www.tensorflow.org/datasets/catalog/cifar10?hl=es-419

Thompson, N. C., Greenewald, K., Lee, K., & Manso, G. F. (2020). The computational limits of deep learning. *arXiv preprint arXiv*:2007.05558.