CLASIFICACIÓN DE PERROS

Juan David Aranguren Bonil
Jessica Alexandra Cruz Franco

***Abstract*— En este documento se presenta un algoritmo de convolutional neural network, para la clasificación de doce diferentes razas de perros. El código se realizo en python, y se empleó la librería tensor flow.**

I. INTRODUCIÓN

El algoritmo de convolutional neural network es un tipo de red neuronal artificial, donde las neuronas corresponden a campos receptivos muy similares a las neuronas de la corteza visual primaria.
Este tipo de red neuronal es muy utilizado actualmente, dentro de sus aplicaciones se encuentra el reconocimiento de imágenes, la clasificación de imágenes, el análisis de videos, entre otras.
De las aplicaciones presentadas anteriormente la más empleada es la clasificación de imágenes, ya que en este método se puede emplear diferentes arquitecturas, y el porcentaje de exactitud es bastante óptimo.
En este documento se presenta un algoritmo convolutional neural network para la clasificación de doce razas de perros, para el funcionamiento de este, primero se realiza la adquisición de base de datos, esto empleando un programa de búsqueda de imágenes según la raza y se almacena en diferentes carpetas de entrenamiento y de testeo, donde la mayor parte se dejó en la carpeta de entrenamiento. Después se realiza el algoritmo empleando la arquitectura secuencial y se graficó el resultado obtenido, para obtener un mejor resultado empleando la librería de tensor flow se generan más imágenes, pero cambiándoles las características, para así poder adquirir más información relevante para la clasificación.

II. PARTES DEL CÓDIGO

Para realizar el código se realizaron los siguientes pasos:

- Se importan las bibliotecas
- Para iniciar se necesita ubicar la carpeta donde tenemos las imágenes de entrenamiento y testeo. Se separan las direcciones de entrenamiento y testeo
- Se guardan cada uno de los directorios de las razas seleccionadas para entrenamiento y testeo en este caso para las doce razas de perros
- Contamos el número total de imágenes de entrenamiento y testeo.
- Se determina la cantidad de iteraciones, se modifica el tamaño de la imágenes por uno estándar.
- Como las redes neuronales prefieren valores de entrada simplificados, se re escalan los valores de las imágenes de 255 a 1 (valores en pixeles)
- Se ajustan todas las imágenes al mismo tamaño para poder entrenar al sistema.
- Se grafican 5 imágenes de entrenamiento para verificar los ajustes.
- Se sube el modelo a usar, con las respectivas capas y la densidad que cambia con la cantidad de clases
- Se empieza a alimentar el sistema y a realizar las iteraciones, teniendo en cuenta la exactitud y las pérdidas
- Como obtenemos valores bastante bajos en nuestro modelo, se opta por modificar las imágenes a diferentes posiciones y perspectivas para mejorar su rendimiento.

III. RESULTADOS

Empleando el tensor flow se modifican las imágenes para poder adquirir más características sobre la imagen.

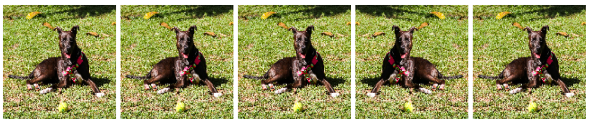

Fig. 1. modificación de imagen

A continuación, se presenta los datos obtenidos con la configuración inicial y final. se presenta el porcentaje de exactitud del entrenamiento y validación y las pérdidas que se presentan con el algoritmo.

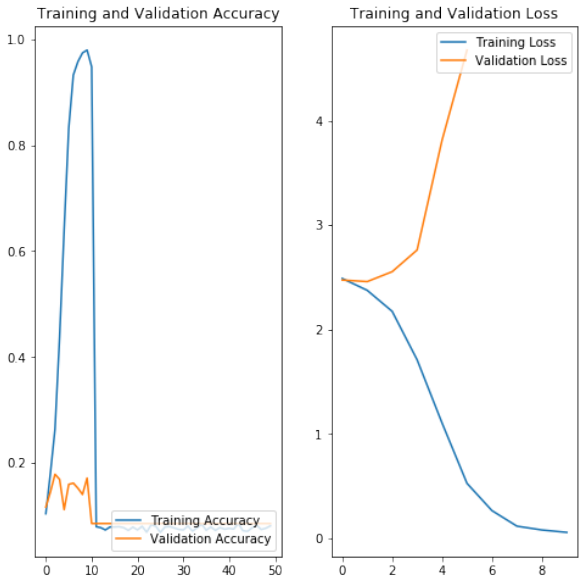

Fig. 2. Resultado inicial

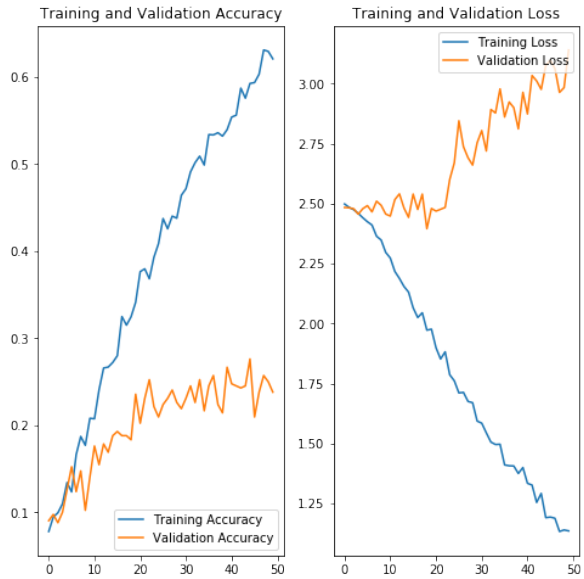

Fig. 3. Resultado final

IV. CONCLUSIONES

-Al implementar un CNN se evidencia que de forma binaria tiende a tener menos errores de clasificado, al agregar más categorías o clases este presenta acumulaciones de pérdidas.

-Para obtener mejores resultados en el entrenamiento y el testeo es necesario cambiar las características de las imágenes, entre más variaciones de imagen, mejor se comporta el sistema.