

Proyecto final

Artificial Intelligence Developer 2 2020 Global



Nicolas Viera

4.562.028-7

# Implementacion

**Dataset**

En primera instancia lo primero que tuve que hacer fue tratar de balancear mi dataset, tanto a nivel de clases como de tags.

Tuve la necesidad de manipular mi propio dataset y eso me llevo a tener que implementar una clase. La misma hereda de la clase ‘Dataset’, y recibe por parámetros un .csv el cual cuenta con la información que manipulara mi dataset, el path de dónde voy a tener almacenadas mis imágenes, un objeto de la clase ‘transforms’ por si quiero aplicarle alguna transformación a las imágenes que voy a utilizar y por último, tengo una variable llamada ‘ResizedCrop’ la cual paso en true cuando quiero hacer dataAumentation de mis datos.

Sobrescribiendo el método ‘\_\_getitem\_\_’ realizo los parseos necesarios sobre mi .csv y aplico las transformaciones necesarias sobre las imágenes.

**Desafortunadamente no hice un análisis sobre mis imágenes previamente al desarrollo del modelo, esto lo hice durante el testeo del mismo y descubrí que cuento con un 36% de error (revise 30 imágenes de las cuales 11 no concordaban con su data correspondiente).**

**Modelo**

Mi modelo consta con una arquitectura similar a la de LeNet-5. Cuenta con 3 convolutional layers con un stride y un padding de 1, a las cuales les aplico la función de activación Relu. A cada una de las convolutional layers luego de aplicarles la función de activación, también les aplico un maxpooling (2,2). A continuación de estas se encuentran 2 fully connected layers, que luego de hacer pruebas decidí agregarle 6 mas. Una de 4 neuronas para poder realizar la clasificación de categorías, y luego tengo 5 layers más de una neurona las cuales me van a dar una salida binaria (una por cada tag/label)

**Parametrización**

Conte finalmente con una data de 5000 imágenes las cuales repeti mediante transformaciones para el training, y con unas 5000 imágenes para la evaluación. La configuración final conto con:

* Batchs de 4
* 12 epochs
* Learning rate = 0.001
* Momentum = 0.9
* Weight\_decay = 0.01

**Cálculo del loss**

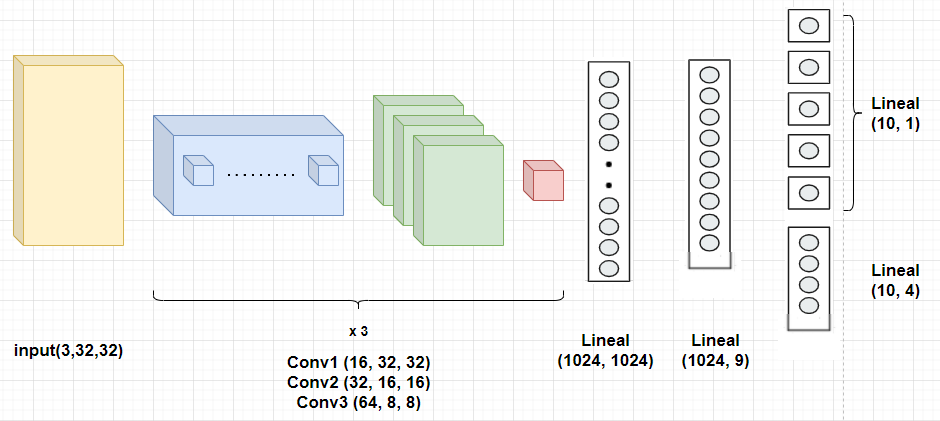
El loss lo calcule aplicándole a cada layer binaria la Binary Cross Entropy Loss y a la layer de categorización la Cross Entropy Loss común.

Todas estas se suman y se le realiza el backward().

Mi modelo luego de entrenarlo quedo con un los de 1.0

**Métricas**

**Diagrama de Arquitectura Final**

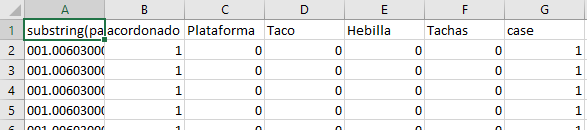


**Conclusiones**

* Verificar que las imágenes de mi dataset concuerden con la información con la que cuento.
* Probar Transfer Learning cuando cuento con poca data para entrenar.
* Seguir indagando en cómo solucionar los problemas que se me presentaron para mejorar el modelo.
* Mejorar mi forma de codificar en Python y el manejo de matrices.

**Instrucciones para uso del modelo**

Para cargar el dataset es necesario especificarle al método ‘Shoes\_Classification\_Dataset’ el path donde están alojadas las imágenes, y el path del csv que se debe cargar. El csv se deberá cargar con el siguiente formato.



La primera columna lleva el nombre de la imagen terminando en .jpg. De la columna 2 a la 6 se completa con 1/0 dependiendo si el articulo posee dicho tag o no; y la ultima columna que es la de clasificación se debe completar con: 0 = bota, 1 = deportivo, 2 = zapato, 3 = sandalia.