UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Estudios Superiores
Acatlán

Practica 6:

Redes Neuronales Convolucionales

Aprendizaje de maquina y minería de datos avanzados

Integrantes:

Gustavo Adolfo Alvarez Hernández

Profesor:

Eduardo Eloy Loza Pacheco

FES Acatlán, 20 de abril 2023

Objetivo

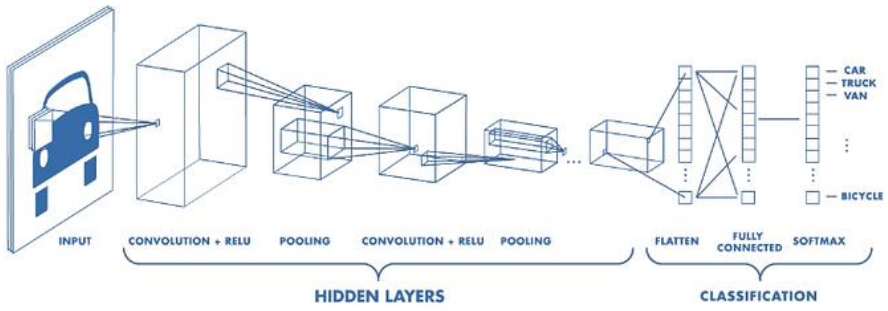
Que el alumno aprenda a utilizar la operación de convolución con el fin de extraer características de las imágenes, lo cual acompañado de su entendimiento de redes neuronales para tareas de clasificación, poder clasificar imágenes de manera satisfactoria.

Resumen

En la presente practica se utilizara una red neuronal convolucional sobre un conjunto de datos de imágenes de perros y gatos, con el fin de entrenar la red neuronal y que sea capaz de identificar si en una imagen si se muestra un perro o un gato.

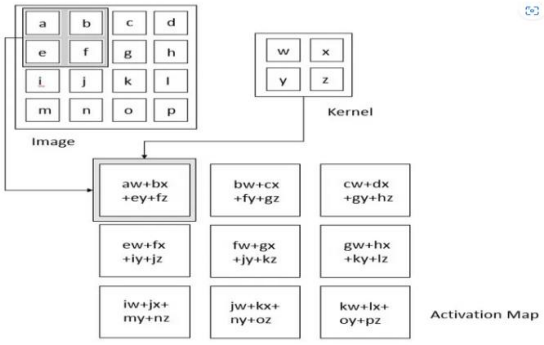
Antecedentes

Una red neuronal convolucional (la cual abreviaremos por sus siglas en ingles CNN) generalmente se separa en 3 componentes principales que define su arquitectura y son utilizadas cuando los datos se presentan de manera matricial, esto es un solo registro se representa por una matriz como lo son las imágenes:



Los 3 principales componentes se ordenan de la siguiente manera:

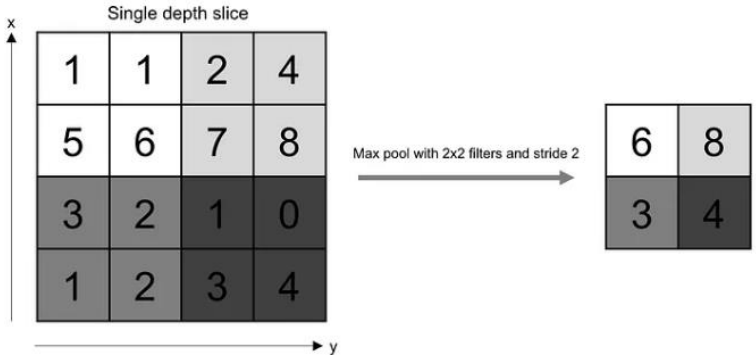
1) Capa convolucional:
   a) Esta es la primera capa que realiza la convolución, la cual consta de el producto de dos matrices que son el dato en sí y el kernel el cual varía, pero es una matriz que tiene funcionalidades como detectar bordes o líneas.



   b) Una de las ventajas de esta operación es la reducción de dimensión, como en el ejemplo de la imagen pasamos de una representación de 4*4 a una de 3*3, lo cual es relevante a la hora de trabajar con imágenes al ser matrices muy grandes.

2) Capa de agrupación
   a) Esta capa tiene la tarea de agrupar la información para tener una representación aún más pequeña de lo generado por la red, esta capa analiza los resultados en porciones de la matriz y genera un representante por cada cuadrante o zona que genera.

   b) Hay múltiples funciones de agrupación, una de las mas usadas es max pooling que consiste en tomar el valor mayor de cada cuadrante.



3) Capa totalmente conectada
   a) Esta es la última capa, la cual consta de una red neuronal que en general esta enfocada en clasificación, esto es las dos anteriores capas solo sirven para generar una nueva representación de los datos, pero de menor dimensión con las características más representativas.
   b) Como se observo las primeras capas generan operaciones lineales, por lo cual aquí se busca aplicar funciones no lineales para la clasificación, como lo son:
      i) Sigmoide
      ii) Tanh
      iii) ReLu
      iv) Entre otras.

Desarrollo

1) Cargamos los módulos necesarios:

```
from keras.models import Sequential
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
from keras.preprocessing.image import ImageDataGenerator
# Prediccion
import numpy as np
import keras.utils as image
```

2) Generamos la arquitectura del modelo:

```
clasificador = Sequential()
```

   a) Generamos la capa de convulsión

```
clasificador.add(Conv2D(filters=32,kernel_size=(3,3),input_shape=(64,64,3,),activation="relu"))
```

b) Generamos la capa de agrupación:

```
clasificador.add(MaxPooling2D(pool_size=(2,2)))
```

```
clasificador.add(Flatten())
```

c) Generamos las capas de la red neuronal de clasificación:

```
clasificador.add(Dense(units=128,activation="relu"))
# 128 nodos intermedios
```

```
# Capa de salida
clasificador.add(Dense(units=1,activation='sigmoid'))
# 1 --> Salida binaria
```

```
clasificador.compile(optimizer='adam',loss='binary_crossentropy',metrics=['accuracy'])
```

3) Cargamos las imágenes de entrenamiento y de test:

```
]:
training_dataset = train_datagen.flow_from_directory('Gatos y Perros/training_set',
                                                     target_size=(64,64),
                                                     batch_size=32,
                                                     class_mode='binary')
```

```
Found 8005 images belonging to 2 classes.
```

```
]:
testing_dataset = test_datagen.flow_from_directory('Gatos y Perros/test_set',
                                                   target_size=(64,64),
                                                   batch_size=32,
                                                   class_mode='binary')
```

```
Found 2023 images belonging to 2 classes.
```

4) Entrenamos a la CNN durante 32 épocas:

```
5]:
clasificador.fit_generator(training_dataset,
                           steps_per_epoch=20,
                           epochs=32,
                           validation_data = testing_dataset,
                           validation_steps=2000)
```

```
Epoch 1/32
20/20 [==============================] - 3s 136ms/step - loss: 0.6932 - accuracy: 0.5016
Epoch 2/32
20/20 [==============================] - 3s 145ms/step - loss: 0.6933 - accuracy: 0.4844
Epoch 3/32
20/20 [==============================] - 3s 137ms/step - loss: 0.6931 - accuracy: 0.5141
```

5) Generamos una predicción sobre el caso de un gato para saber si hace la predicción de una manera adecuada:

```
:
test_image = image.load_img('Gatos y Perros/test_set/cats/cat.4001.jpg',target_size=(64,64))
test_image = image.img_to_array(test_image)
test_image = np.expand_dims(test_image,axis=0)
result = clasificador.predict(test_image)
training_dataset.class_indices
if result[0][0]==1:
    prediction='dog'
else:
    prediction ='cat'
print(prediction)
```

```
1/1 [==============================] - 0s 23ms/step
cat
```

Conclusiones

Las CNN son un buen modelo para el manejo de imágenes dado que las imágenes son un formato de alta dimensionalidad y patrones difícil de caracterizar, por lo cual el extraer las características y reducirlas a una dimensionalidad manejable permite utilizar una red de clasificación para identificar que hay en la imagen.

También trabajar con imágenes es un proceso de cómputo más demandante, pero en general se tiene un buen desempeño bueno y tiempos de espera aceptables.

Referencias

Mishra, M. (26 de Agosto de 2020). *Towards Data Science*. Obtenido de Convolutional Neural Networks, Explained: https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939

Zhou, Z.-H. (2016). *Machine Learning.* Singapore: Springer Nature Singapore.