# Generador de descripciones de imágenes

**Abel Valle Chávez**

0231889@up.edu.mx

Repositorio Github: dl-image-description-gen

Cuaderno de Colaboratory: image-caption-gen-abelvalle.ipynb

## Introducción

La intención de este trabajo es poder integrar distintas redes neuronales (NN) y procesamiento de lenguaje natural (NLP). El problema que seleccioné para cumplir con dicha intención se originó en el artículo "Show and Tell: A Neural Image Caption Generator" de colaboradores de Google [1]. El objetivo es realizar una implementación para entrenar un modelo de deep learning que permita proporcionar una descripción textual en inglés que vaya de acuerdo a una imagen dada. Las NN que se usan en la implementación son Red Neuronal Convolucional (CNN) complementando con la Red Neuronal Recurrente (RNN) Long Short Term Memory (LSTM).

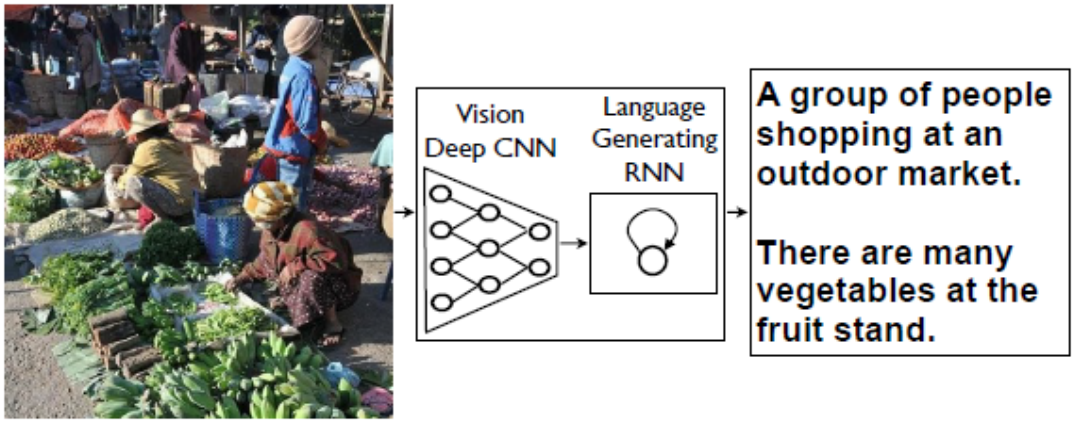



Figura 1. Modelo que consiste en una CNN de visión seguida de una RNN generadora de lenguaje a partir de una imagen de entrada. Imagen obtenida de [1].

## 1. El conjunto de datos

El conjunto de datos utilizado es el de Flickr_8K [2] que cuenta con 8091 imágenes de distintas dimensiones y se ha convertido en un estándar en el problema de descripción textual de imágenes. Dentro de los datos se proporciona un archivo de texto (Flickr8k.token) que contiene el nombre del archivo de imagen y su descripción.

Aunque se incluyen los datos necesarios, se hace referencia a los víclulos directos de descarga:

- flickr8k_dataset
- flickr8k_text
- Github de Jason Brownlee con otros datasets

## 2. Las redes neuronales CNN y LSTM

Las CNN son redes neuronales profundas especializadas en procesar datos cuya entrada sea una matrix 2D, por lo tanto son útiles para procesar imágenes. Las CNN se utilizan para clasificación, por ejemplo, para identificar si una imagen es una motocicleta, un avión, un auto, etc. En idea, lo que hace es recorrer la imagen de izquierda a derecha, de arriba hacia abajo obteniento características relevantes y las combina para clasificar. Puede manejar imágenes que estén trasladadas, rotadas o escaladas.

Las redes recurrentes LSTM son apropiadas para problemas donde los eventos suceden en secuencia. Bajo el principio mencionado, es posible predecir la palabra siguiente dato un texto previo. A diferencia de una red recurrente tradicional es que la LSTM puede contener información relevante a través del procesamiento de entradas con un filtro de "olvido", el cual descarta la información no relevante.

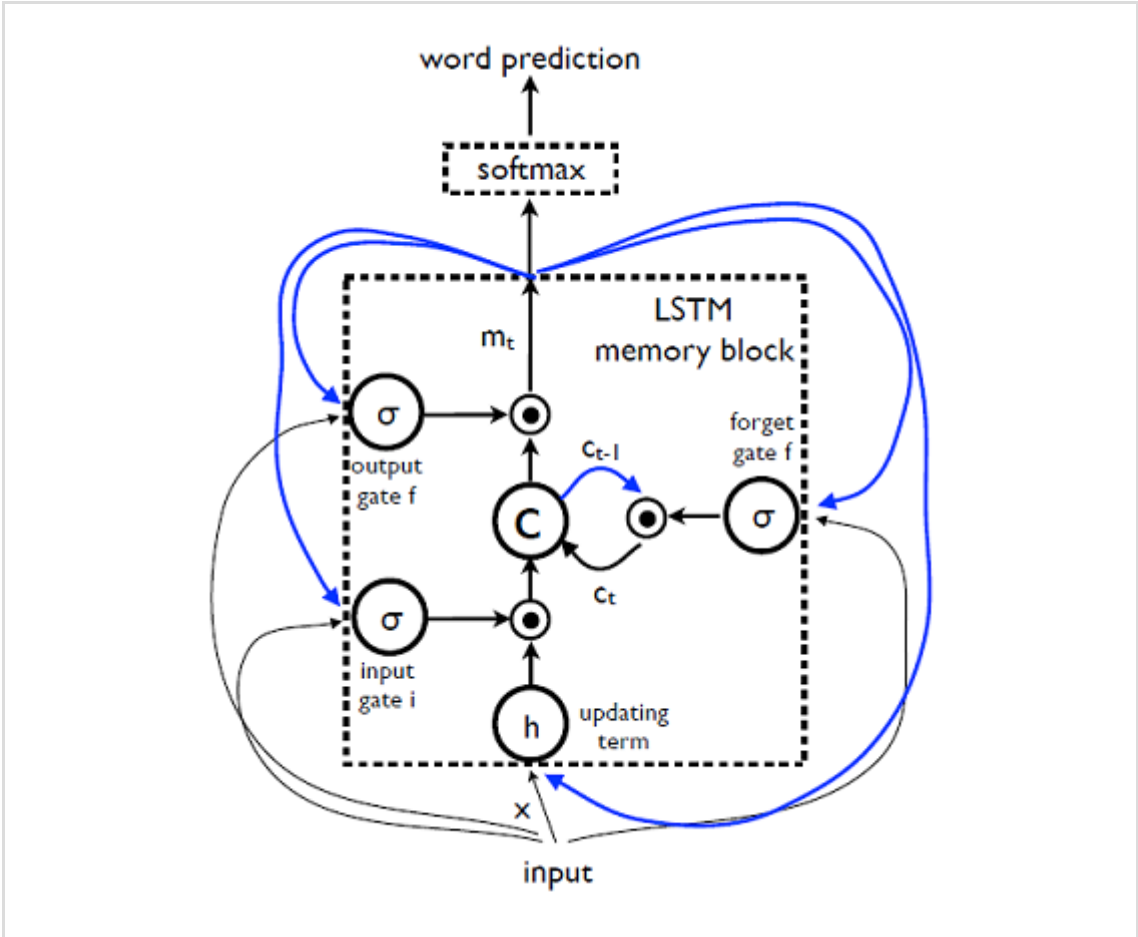



Figura 2. Bloque de memoria LSTM. Contiene una celda c que es controlada por 3 compuertas. En azul se muestran las conexiones recurrentes. La salida del bloque termina en Softmax para la predicción de palabra. Imagen obtenida de [1].

## 3. El modelo del generador de descripción de imagen

Para implementar el modelo generador de descripciones se unirán las arquitecturas CNN y LSTM, a dicha unión también se le denomina modelo CNN-RNN.

- La CNN se usa para extraer características de la imagen. Aquí se utiliza el modelo entrenado previamente denominado Xception.
- La LSTM usa la información de la CNN para generar la descripción de la imagen.

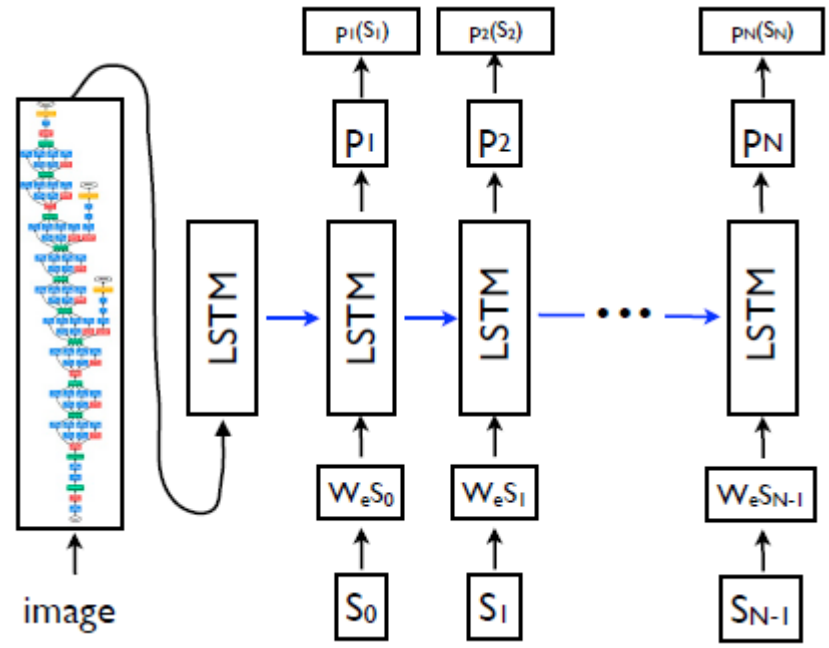



Figura 3. Modelo que consiste en una CNN de visión seguida de una RNN generadora de lenguaje a partir de una imagen de entrada. Imagen obtenida de [1].

## 4. Extracción del vector de características de las imágenes

Para esta etapa se aplica la técnica de *transfer learning*, que consiste en utilizar un modelo previamente entrenado en conjuntos de datos grandes de donde se extraen características que posteriormente utilizamos en nuestro procesamiento. El modelo se reutiliza es Xception, que ha sido entrenado en un conjunto de imágenes que tiene 1000 clases. El modelo se importa directamente de keras.applications. Se observó que el modelo Xception trabaja con imágenes de tamaño (299 x 299 x 3) como entrada, por lo tanto, las imágenes se ajustan al tamaño requerido.

## 5. Carga de datos para entrenamiento del modelo

Se tomaron 6000 imagenes para entrenamiento. La carga de datos se resume en los siguientes puntos.

- Se carga un archivo de texto con los 6000 nombres de cada archivo de imagen.
- Se crea un diccionario que contiene descripciones para cada foto. Se agregan tokens y para identificar dónde comienza y termina cada descripción.
- Se crea un diccionario con los nombres de imagen y el vector de características que previamente se ha extraído del modelo Xception.

## 6. Tokenización de vocabulario

A cada palabra se le asocia un índice numérico único. Para lo anterior, Keras proporciona el objeto *Tokenizer*, que se usa para crear los tokens a partir del vocabulario y se guardan en el archivo *tokenizer.p*.

## 7. Creación del generador de descripciones

Para generar las descripciones, mediante aprendizaje supervisado, se le proporciona al modelo la entrada y la salida para el entrenamiento. Se entrenan 6000 imágenes, cada una con su vector de 2048 características y su descripción codificada en forma numérica. Para el propósito se utiliza *yield* de Python. Al usar *yield*, interrumpimos la ejecución en determinado punto, conservando la instancia para su uso posterior, así hasta que hayamos terminado. Por ejemplo, la entrada de nuestro modelo es [*x1, x2*] y la salida será *y*. donde *x1* es el vector de características, *x2* es la secuencia de la descripción y *y* es el la palabra de salida que el modelo va a predecir (ver Tabla 1).

Tabla 1. Ejemplo de generación de cadena.

| x1 | x2 | y (predicción de palabra) |
|---|---|---|
| [feature-vector] | start | two |
| [feature-vector] | start two | men |
| [feature-vector] | start two men | playing |
| [feature-vector] | start two men playing | basketball |
| [feature-vector] | start two men playing basketball | end |

## 8. El modelo CNN-RNN

La definición del modelo consiste en las siguientes partes.

- Extracción de características. La característica extraída de la imagen tiene un tamaño de 2048, la cuál se reduce a 256.
- Procesamiento de secuencia. Una capa embebida maneja la entrada en forma de texto, seguida de una capa LSTM.
- Decoficador. Uniendo la salida de las dos capas anteriores, se procesa mediante una capa densa para tener como salida la predicción final. La capa final contiene el número de unidades igual al tamaño del vocabulario.

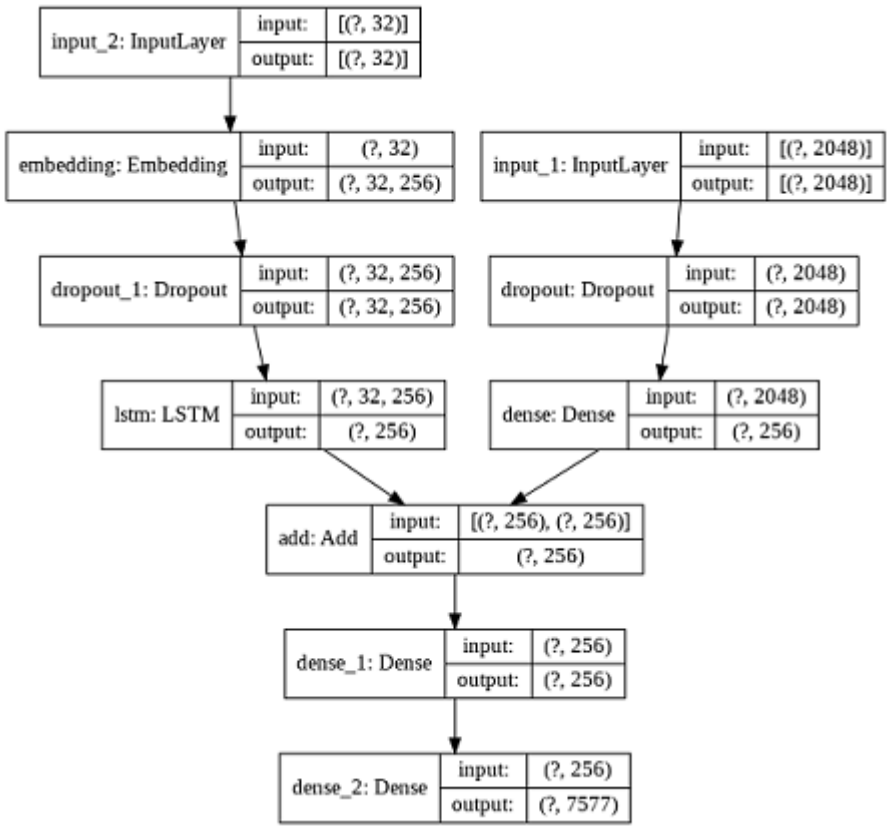



Figura 4. Esquema de modelo. Imagen generada mediante la función *plot_model()* de *keras.utils*.

## 9. Entrenamiento del modelo

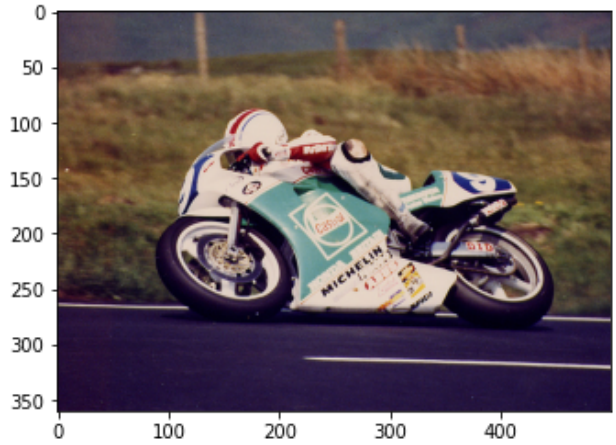
Para el entrenamiento del modelo, se utilizan 6000 imágenes generando las secuencias de entrada y salida en batch, ajustándolas al modelo mediante el método *model.fit_generator()*. El modelo entrenado se guarda como un archivo *model_[epoch].h5* en una carpeta para su futuro uso sin necesidad de volver a entrenar.

## 10. Resultados

A continuación se muestran algunas de las imágenes de prueba que se le proporcionaron a la CNN-RNN y las descripciones obtenidas como resultado.

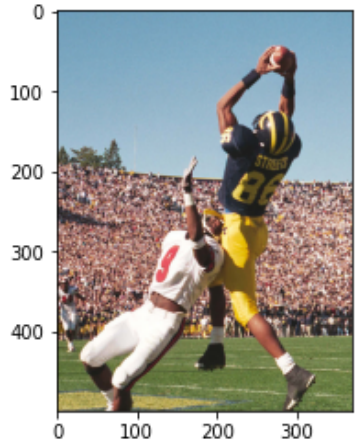
```
1 img_path = base_path + "/flickr8k-dataset/2554531876_5d7f193992.jpg"
2 get_image_description(model, tokenizer, xception_model, max_length, img_path)
```

```
motorcycle racer is riding red motorcycle
```



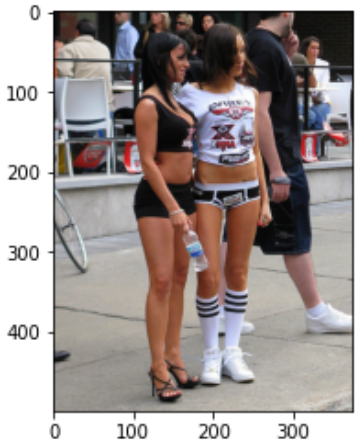

```
1 img_path = base_path + "/flickr8k-dataset/273248777_eaf0288ab3.jpg"
2 get_image_description(model, tokenizer, xception_model, max_length, img_path)
```

```
two men playing baseball
```



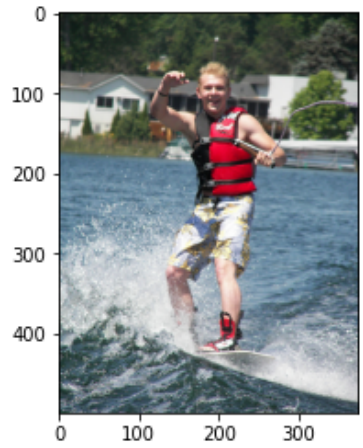
```
1 img_path = base_path + "/flickr8k-dataset/2925760802_50c1e84936.jpg"
2 get_image_description(model, tokenizer, xception_model, max_length, img_path)
```

```
two women in red shirts are playing football on the street
```



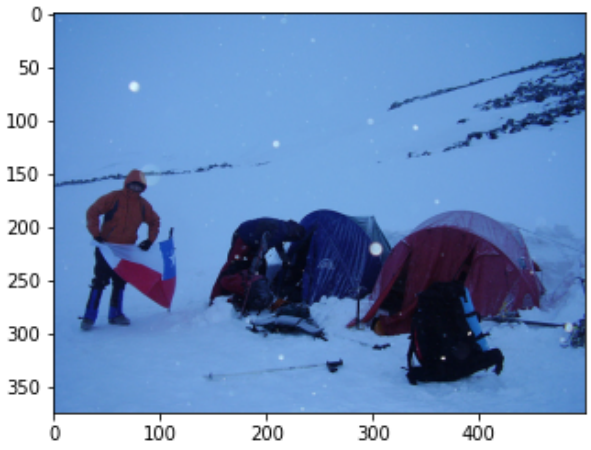

```
1 img_path = base_path + "/flickr8k-dataset/172092461_a9a9762e13.jpg"
2 get_image_description(model, tokenizer, xception_model, max_length, img_path)
```

```
man in wetsuit is surfing
```



```
1 img_path = base_path + "/flickr8k-dataset/1584315962_5b0b45d02d.jpg"
2 get_image_description(model, tokenizer, xception_model, max_length, img_path)
```

two people are walking on snowy mountain



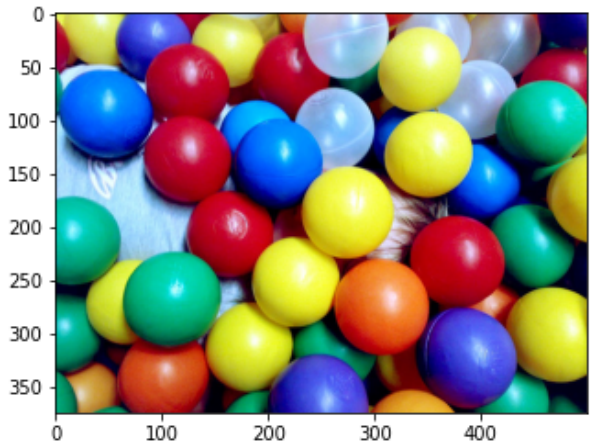

```
1 img_path = base_path + "/flickr8k-dataset/426920445_d07d1fd0f7.jpg"
2 get_image_description(model, tokenizer, xception_model, max_length, img_path)
```

dog is running through the grass


**Imágenes con descripciones fuera de contexto.**

```
1 img_path = base_path + "/flickr8k-dataset/1433397131_8634fa6664.jpg"
2 get_image_description(model, tokenizer, xception_model, max_length, img_path)
```

young boy in the bathtub with his tongue sticking out

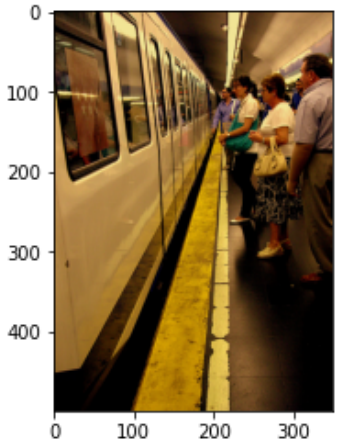



```
1 img_path = base_path + "/flickr8k-dataset/3220140234_e072856e6c.jpg"
2 get_image_description(model, tokenizer, xception_model, max_length, img_path)
```

two girls are playing football on the grass

```
img_path = base_path + "/flickr8k-dataset/2981702521_2459f2c1c4.jpg"
get_image_description(model, tokenizer, xception_model, max_length, img_path)
```

```
man in black shirt and black shirt is sitting on bench
```




Más ejemplos se pueden encontrar en el siguiente vínculo: [image-caption-gen-abelvalle.ipynb]

## 11. Conclusiones

Se implementó una red neuronal que integra dos arquitecturas una CNN para el reconocimiento de imagen y una RNN LSTM para generar descripciones textuales razonables en inglés. La CNN codifica la imagen, seguida de la RNN LSTM que genera la sentencia correspondiente. Los resultados al probar con distintas imágenes son interesantes ya que en el texto se capta parte del contexto de la imagen. Es evidente que conforme se aumente el conjunto de entrenamiento las descripciones proporcionarán más sentido al contexto de la imagen.

Conforme la tarea de descripción de imágenes vaya evolucionando, se puede extender a proporcionar indicios de contexto o características que a simple vista humana se pueden omitir, extendiendo habilidades de un observador de seguridad o bien ayudando a personas con deficiencia visual a escuchar la descripción de una imagen (text-to-speech).

## Referencias

[1] O. Vinyals, A. Toshev, S. Bengio and D. Erhan, "Show and tell: A neural image caption generator", Proceedings of 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3156-3164, 2015.

[2] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. In ACL, 2014.

### Códigos de referencia

https://www.kaggle.com/shadabhussain/automated-image-captioning-flickr8
https://www.kaggle.com/wikiabhi/image-caption-generator

## Organización del repositorio

- **flickr-8k-dataset**: Carpeta con 8091 imágenes. Por motivo de portabilidad, las imágenes se tienen que descomprimir a partir del archivo *flickr-8k-dataset.zip* crear y colocar en la carpeta.
- **flickr-8k-text**: Carpeta que contiene archivos de texto con las descripciones de cada imagen. Por motivo de portabilidad, los archivos txt se tienen que descomprimir a partir del archivo *flickr-8k-text.zip* crear y colocar en la carpeta.
- **models**: Carpeta que contiene el o los modelos entrenados, el modelo listo para usarse se encuentra en el archivo *model_9.h5*.
- **descriptions.txt** – Archivo de texto que tiene los nombres de cada imagen después del preprocesamiento.
- **features.p**: Archivo de objeto *Pickle* que contiene el vector de características obtenidas mediante el modelo CNN pre-entrenado Xception.
- **tokenizer.p**: Contiene los tokens mapeados a un índice (valor numérico).
- **image-caption-gen-abelvalle.ipynb**: Cuaderno de notas Google Colaboratory el cual contiene el código del proyecto.