Redes neuronales convolucionales

Contents

Una red convolucional (Convolutional Neural Network, CNN) es un tipo de red neuronal especialmente diseñada para procesar datos con una estructura de cuadrícula, como imágenes. Las CNN son particularmente eficaces en tareas de clasificación, detección y segmentación de imágenes, gracias a su capacidad para aprender patrones espaciales.

Una de las operaciones fundamentales en una red neuronal convolucional es la **convolución**, la cual permite extraer características locales de una imagen, como **bordes**, **texturas** o **esquinas**.

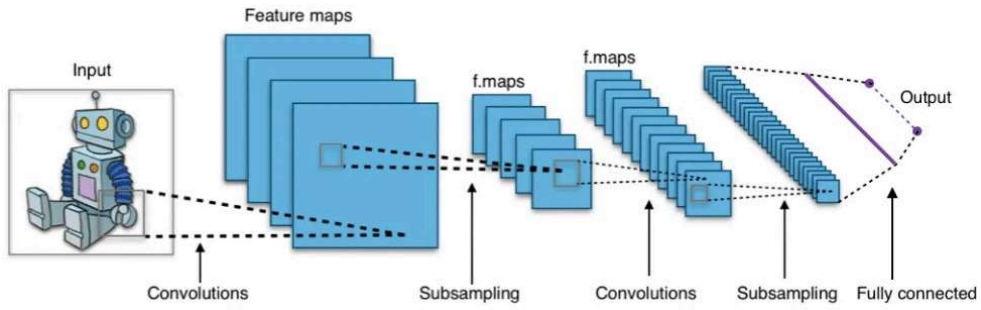

Fig.1.CNN. Imagen tomada de Taye MM

La convolución consiste en aplicar un pequeño conjunto de pesos, conocido como **kernel** o **filtro**, sobre distintas regiones de la imagen. Esta operación se expresa mediante la siguiente fórmula:

$$O(i,j) = I(i+m, j+n) \cdot \sum_m \sum_n K(m,n)$$

En esta expresión:

- $I(i+m, j+n)$ representa el valor de un píxel de la **imagen original** en la posición $(i+m, j+n))$$
- $K(m,n)$ es el valor del **filtro** en la posición (m,n)

Lo que hace la convolución es **multiplicar** cada elemento del filtro por el valor del píxel correspondiente de la imagen en una región específica, y luego **sumar todos esos productos** para obtener un nuevo valor. Este resultado se almacena en una nueva matriz llamada **mapa de características** (*feature map*).

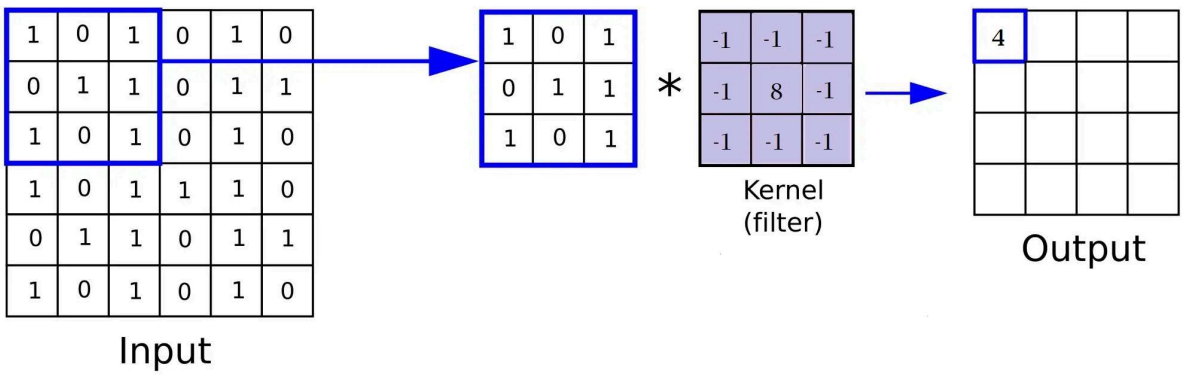

Fig.2. Convolución

$$\begin{aligned}\text{Output}(i,j) &= (1 \cdot -1) + (0 \cdot -1) + (1 \cdot -1) \\ &+ (0 \cdot -1) + (1 \cdot 8) + (0 \cdot -1) \\ &+ (1 \cdot -1) + (0 \cdot -1) + (1 \cdot -1) \\ &= -1 + 0 - 1 + 0 + 8 + 0 - 1 + 0 - 1 \\ &= 4\end{aligned}$$

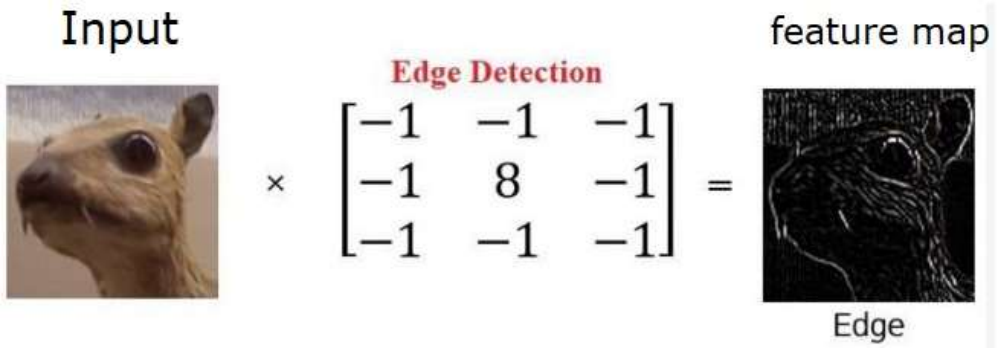

Fig.3.Mapa de características. Imagen tomada de Taye MM

Tipos de Capas

Capa de Entrada

Esta capa recibe los datos de entrada, que son típicamente imágenes. Las imágenes se representan como tensores en 3D, donde las dimensiones corresponden a la altura, la anchura y la profundidad (número de canales, como RGB).

Capas Pooling

Las capas de pooling en una rconvolucional (CNN) reducen la dimensión espacial de los mapas de características, conservando la información más relevante. Su objetivo es disminuir la carga computacional, evitar el sobreajuste y proporcionar invariancia.

- **Capa max Pooling**

Toma el valor máximo dentro de una ventana.

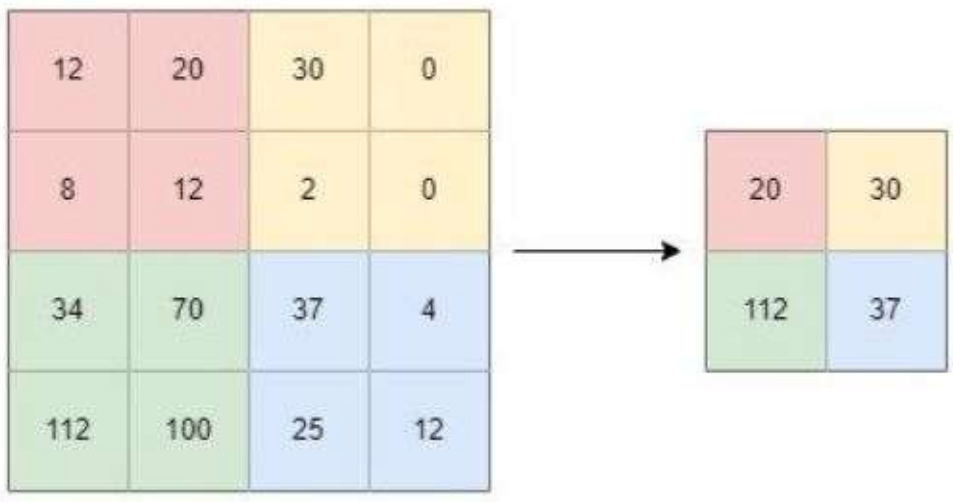

Fig.3.Max pooling. Imagen tomada de Baeldung

- **Capa Average Pooling** Calcula el promedio de los valores dentro de la ventana.

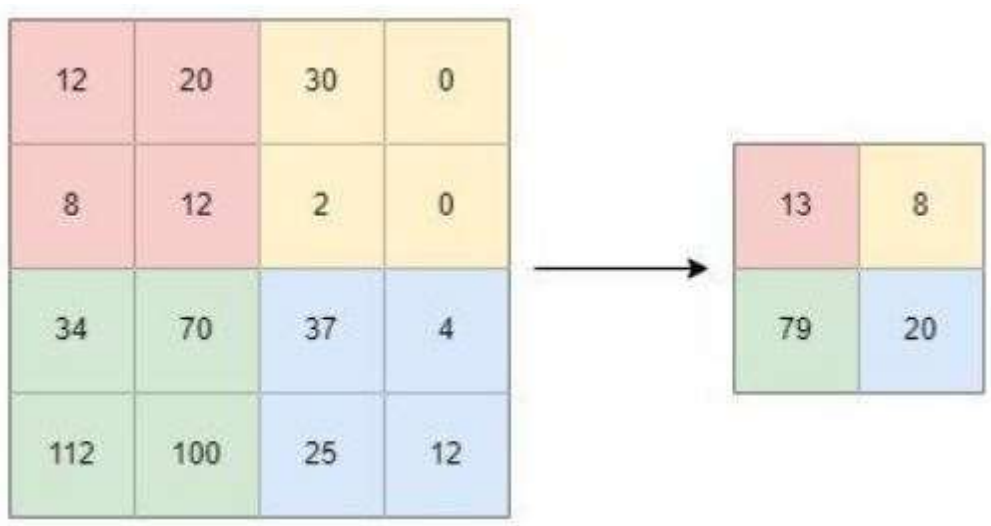

Fig.4.Average pooling. Imagen tomada de Baeldung

Capa Flatten

Convierte una entrada con varias dimensiones, como una imagen en formato 2D o 3D, en un vector lineal de una sola dimensión. Este proceso “aplana” la estructura de datos original para que pueda ser utilizada en las capas densas, que necesitan recibir los datos en forma de un vector unidimensional.

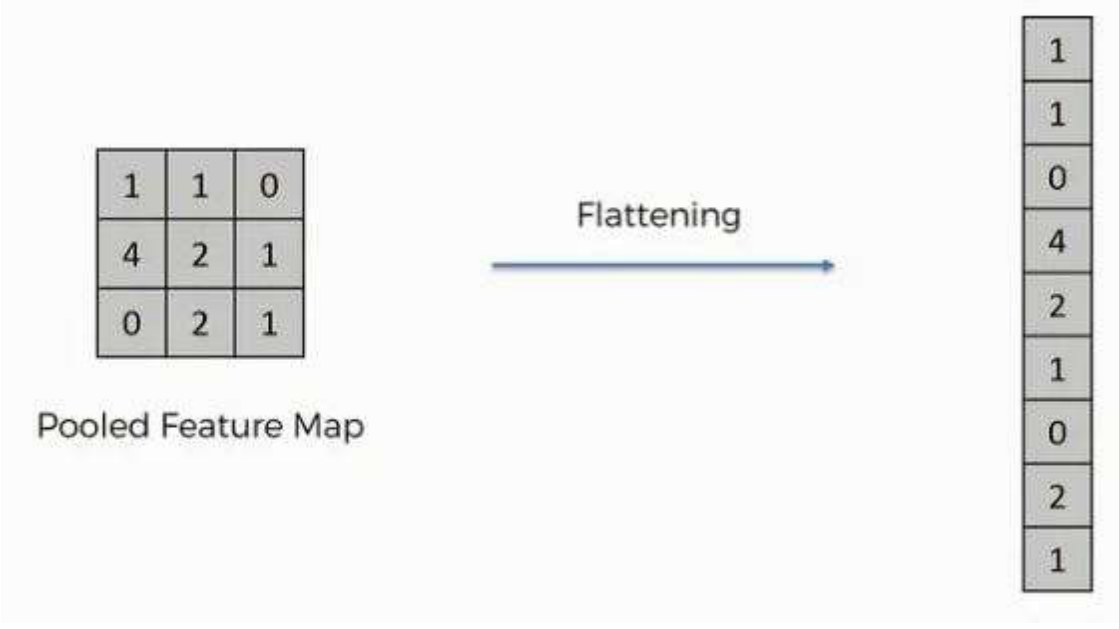

Fig.5.Capa Flatten. Imagen tomada de Bootcampai

Capa Batch Normalization

Se inserta entre una capa oculta y la siguiente en una red neuronal. Su objetivo es normalizar las salidas de la primera capa oculta, manteniendo la media cerca de 0 y la desviación estándar cerca de 1. Esto estabiliza el entrenamiento y acelera la convergencia del modelo.

Padding

Se utiliza en las capas convolucionales para mantener las dimensiones espaciales del mapa de características de salida iguales a las de la entrada. Al añadir filas y columnas adicionales alrededor de la entrada, se permite que las ventanas de convolución se centren en cada mosaico de la entrada.

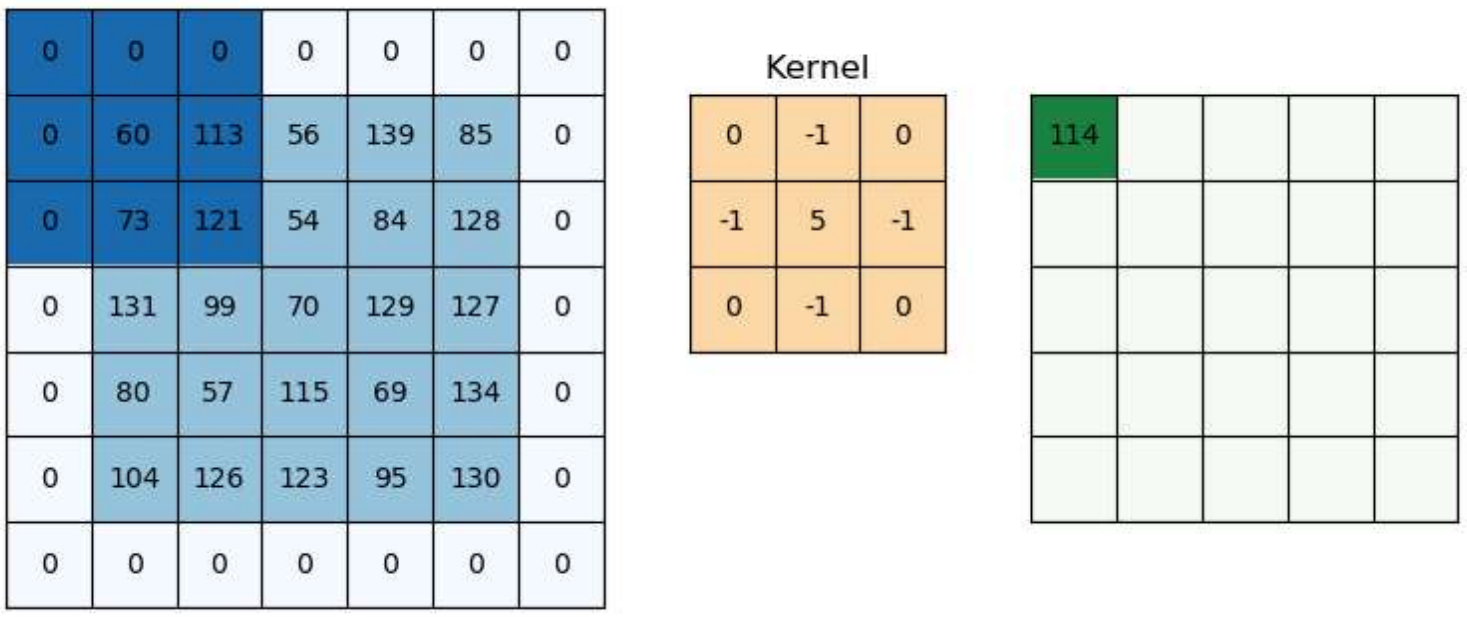

Fig.6.Padding. Imagen tomada de italflux

Zancadas (Strides)

Se refieren a la cantidad de desplazamiento que se aplica a la ventana de convolución mientras se recorre la imagen de entrada. Por defecto, la zancada es 1, lo que significa que la ventana se mueve un píxel a la vez.
Zancadas superiores a 1 resultan en una reducción más rápida de las dimensiones espaciales del mapa de características de salida

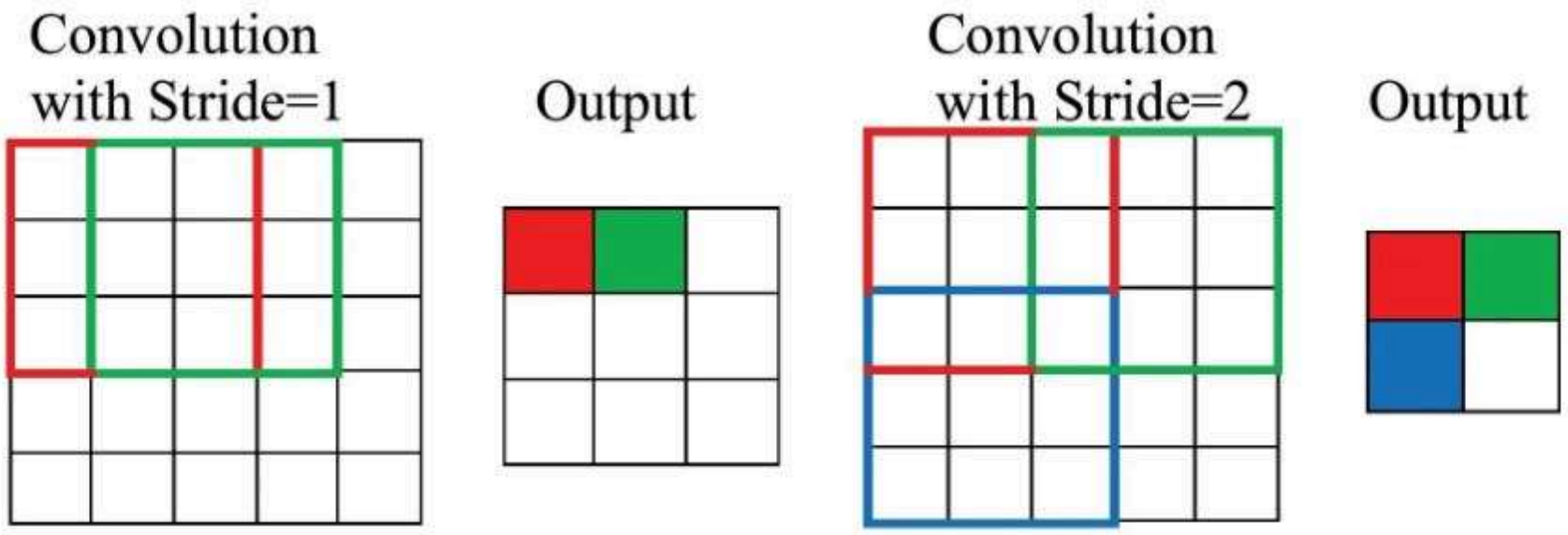

Fig.7.Strides. Imagen tomada de Computer.org

Red neuronal convolucional en Keras

El siguiente codigo,implementa una red neuronal convolucional (CNN) utilizando la API Sequential de Keras, diseñada para procesar imágenes con múltiples bandas (8x8x10). La arquitectura combina diferentes tipos de capas: convolucionales, pooling, normalización, y densas, incorporando diversas funciones de activación y técnicas de regularización para mejorar el rendimiento del modelo. Al final, se compila y entrena para realizar una clasificación multiclase en 4 categorías.

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, AveragePooling2D, Flatten, Dense, Dropout, BatchNormali
from tensorflow.keras.optimizers import Adam

# Construcción del modelo
model_1 = Sequential([
    # Primera capa convolucional con activación ReLU y zancadas de 1
    Conv2D(64, (3, 3), activation='relu', input_shape=(8, 8, 10), padding='same', strides=(1, 1)),
    MaxPooling2D(pool_size=(2, 2)),

    # Segunda capa convolucional con activación Sigmoid y zancadas de 1
    Conv2D(64, (3, 3), activation='sigmoid', padding='same', strides=(1, 1)),
    MaxPooling2D(pool_size=(2, 2)),

    # Tercera capa convolucional con activación Tanh
    Conv2D(32, (3, 3), activation='tanh', padding='same'),

    # Normalización por lotes
    BatchNormalization(),

    # Aplanar para conectar con la capa densa
    Flatten(),

    # Capa densa con activación ReLU
    Dense(64, activation='relu'),

    # Capa densa adicional con activación Sigmoid
    Dense(32, activation='sigmoid'),

    # Capa de salida con activación Softmax (para clasificación multiclase)
    Dense(4, activation='softmax')
])

# Resumen de la arquitectura
model_1.summary()

# Compilación del modelo
model_1.compile(
    optimizer=Adam(learning_rate=0.001),
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy']
)

# Entrenamiento del modelo
history_1 = model_1.fit(
    X_train,
    y_train_encoded,
    epochs=40,
    batch_size=32,
    validation_split=0.1
)
```

Conclusiones

- Las redes neuronales convolucionales son herramientas poderosas para procesar y analizar imágenes. Aprovechan la estructura espacial inherente a los datos en forma de cuadrícula para detectar patrones y reconocer objetos de manera eficiente.
- La operación de convolución, que combina una imagen con un kernel o filtro, es el núcleo de las CNN. Esta operación genera mapas de características que resaltan elementos específicos de la imagen, facilitando su posterior interpretación.

Referencias

- O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks [Preprint]. arXiv. https://doi.org/10.48550/arXiv.1511.08458
- Taye MM. Theoretical Understanding of Convolutional Neural Network: Concepts, Architectures, Applications, Future Directions. Computation. 2023; 11(3):52. https://doi.org/10.3390/computation11030052
- Wang, J., Turko, R., Shaikh, O., Park, H., Das, N., Hohman, F., Kahng, M., & Chau, D. H. P. (n.d.). CNN Explainer: Learn how Convolutional Neural Networks work. Georgia Tech & Oregon State University. https://poloclub.github.io/cnn-explainer/
- Bootcamp AI. (2019, noviembre 23). Intro a las redes neuronales convolucionales. Medium. https://bootcampai.medium.com/redes-neuronales-convolucionales-5e0ce960caf8