A convolutional neural network (CNN) is a deep learning model specifically designed for processing structured grid data like images. Comprising convolutional, pooling, and fully connected layers, CNNs excel at detecting intricate patterns within images. Convolutional layers employ filters to extract features, while pooling layers reduce spatial dimensions, enhancing computational efficiency and reducing overfitting. Fully connected layers integrate extracted features for classification or regression tasks. During training, CNNs employ backpropagation and optimization algorithms to minimize a predefined loss function, adjusting parameters to improve performance. CNNs have transformed fields such as computer vision, enabling breakthroughs in image classification, object detection, and segmentation. Renowned architectures like LeNet-5, AlexNet, and ResNet have set standards for performance and efficiency. Beyond image analysis, CNNs find applications in medical imaging, autonomous driving, and facial recognition, underscoring their versatility and effectiveness in learning complex patterns from visual data.

```
import tensorflow as tf
from tensorflow.keras import layers, models

def create_cnn_model(input_shape):
    model = models.Sequential([
        layers.Conv2D(32, (3, 3), activation='relu', input_shape=input_shape),
        layers.MaxPooling2D((2, 2)),
        layers.Conv2D(64, (3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Conv2D(128, (3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Flatten(),
        layers.Dense(128, activation='relu'),
        layers.Dense(1, activation='sigmoid')
    ])
```

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

    return model

input_shape = (128, 128, 3)
cnn_model = create_cnn_model(input_shape)

cnn_model.summary()
```