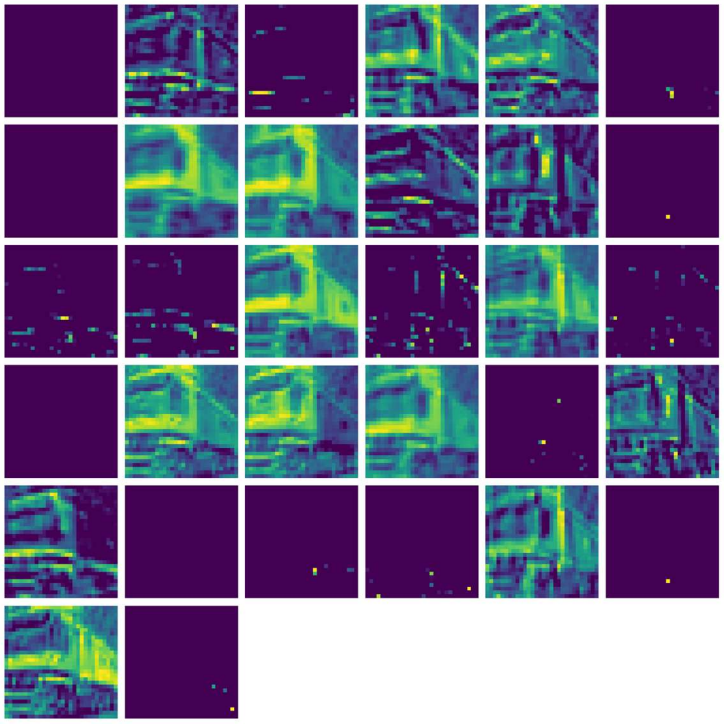
Unit 10 – CNN tutorial

```
import tensorflow as tf
from tensorflow.keras import models, layers
from tensorflow.keras.datasets import cifar10
import matplotlib.pyplot as plt
import numpy as np

# Load CIFAR-10 dataset
(train_images, _), (_, _) = cifar10.load_data()
train_images = train_images / 255.0

# Create a simple CNN model
model = models.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 3)),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu')
])
# Select an image to visualize
img = np.array([train_images[1]])

# Model summary
model.summary()

# Function to visualize feature maps
def visualize_feature_maps(model, image):
    layer_outputs = [layer.output for layer in model.layers if 'conv' in
layer.name]
    activation_model = tf.keras.models.Model(inputs=model.input,
outputs=layer_outputs)
    activations = activation_model.predict(image)

    for layer_idx, activation in enumerate(activations):
        num_filters = activation.shape[-1]
        size = int(np.ceil(np.sqrt(num_filters)))  # Dynamic grid size
        plt.figure(figsize=(size * 2.5, size * 2.5))
        for i in range(num_filters):
            plt.subplot(size, size, i + 1)
            plt.imshow(activation[0, :, :, i], cmap='viridis')
            plt.axis('off')
        plt.tight_layout()
        plt.show()
visualize_feature_maps(model, img)
```

```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_12 (Conv2D)          (None, 30, 30, 32)        896

 max_pooling2d_8 (MaxPoolin  (None, 15, 15, 32)        0
 g2D)

 conv2d_13 (Conv2D)          (None, 13, 13, 64)        18496

 max_pooling2d_9 (MaxPoolin  (None, 6, 6, 64)          0
 g2D)

 conv2d_14 (Conv2D)          (None, 4, 4, 64)          36928

=================================================================
Total params: 56320 (220.00 KB)
Trainable params: 56320 (220.00 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
WARNING:tensorflow:6 out of the last 6 calls to <function Model.mak
1/1 [==============================] - 0s 68ms/step
```



We can see here that the image being detected is of a truck.

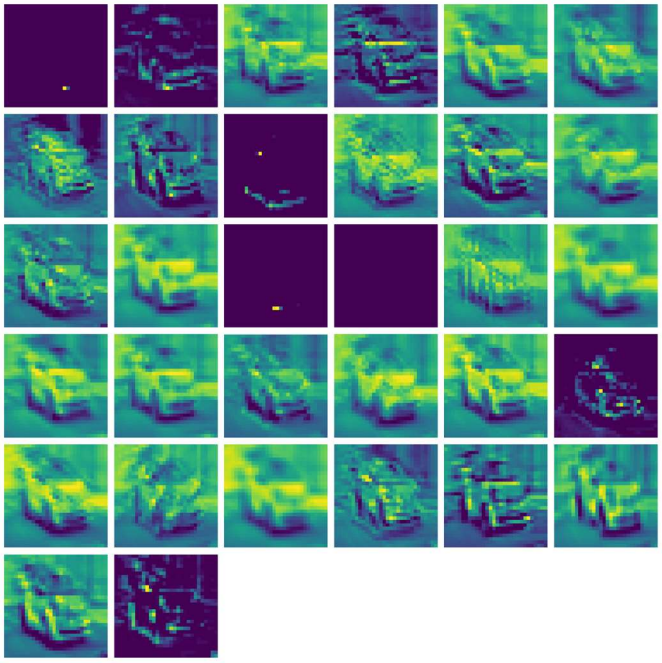
The next step will be to detect a familiar image, below is some code to change the image path, to an image of a car. Let's see if the CNN can detect the image and recognise it.

```
from tensorflow.keras.preprocessing import image
from PIL import Image
# necessary libraries to have the image uploaded from an external source
```

```
image_path = 'car_stock.jpg'
img = load_and_process_image(image_path)
```

```
def load_and_process_image(image_path):
    # Load the image file, ensuring it's resized to 32x32 pixels
    img = Image.open(image_path)
    img = img.resize((32, 32))

    # Convert the image to an array and normalize it
    img = image.img_to_array(img)
    img /= 255.0  # Normalize to [0, 1]

    # Expand the dimensions so the image is (1, 32, 32, 3)
    img = np.expand_dims(img, axis=0)
    return img
```

```
visualize_feature_maps(model, img)
#visualising the imported image that the CNN would be familiar with
```



When changing the image path to something unfamiliar, in this example a rollercoaster has been used, it appears as if the CNN has trouble recognising the image and could be identifying the image as something else that the CNN has been trained to identify within the CIFAR-01 dataset

```
image_path = 'rollercoaster.jpg'
img = load_and_process_image(image_path)
```

```
def load_and_process_image(image_path):
    # Load the image file, ensuring it's resized to 32x32 pixels
    img = Image.open(image_path)
    img = img.resize((32, 32))

    # Convert the image to an array and normalize it
    img = image.img_to_array(img)
    img /= 255.0  # Normalize to [0, 1]

    # Expand the dimensions so the image is (1, 32, 32, 3)
    img = np.expand_dims(img, axis=0)
    return img
```

```
visualize_feature_maps(model, img)
#visualising the imported image that the CNN would NOT be familiar with
```