Week 6 Learning Activities

Explain the concept of transfer learning. How does it differ from training a model from scratch?

Transfer learning is a technique where a model trained on one task is reused for a different, but related, task. Instead of starting from scratch, the model leverages learned features from a pretrained model and fine-tunes it for the new task, making it faster and more efficient.

The key differences between transfer learning and learning from scratch lies in the training time, the ability to work better with small dataset and the performance or efficiency.

What is fine-tuning in the context of transfer learning, and why is it useful?

Fine-tuning in transfer learning refers to adjusting or retraining parts of a pretrained model for a new, specific task. After transferring the general knowledge from the pretrained model, only the last few layers (or sometimes more) are modified to adapt to the new dataset.

Fine tuning is extremely useful in transfer learning since not only does it improve the model performance on the test but also reduces the training time and demanded dataset size.

Why is it important to freeze the convolutional base during feature extraction?

Freezing the convolutional base during feature extraction in transfer learning means preventing the weights of the pretrained layers (typically convolutional layers in image models) from being updated during training. This is to preserve the learning features, eliminate the risk of overfitting and speed up the training process.

Why use data augmentation?

Data augmentation serves the purpose of preventing overfitting by introducing variations in the training data, increasing dataset size without collecting new data and improving model performance by making it more adaptable to real-world variations, which enhances model robustness to handle diverse inputs.

Screenshots

Code snippet where the pre-trained MobileNetV2 model is loaded without the top classification layers.

```
# Create the base model from the pre-trained model MobileNet V2
IMG_SHAPE = IMG_SIZE + (3,)
base_model = tf.keras.applications.MobileNetV2(input_shape=IMG_SHAPE,
                                               include_top=False,
                                               weights='imagenet')
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/mobilenet_v2/mo
9406464/9406464 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
```

Portion of code where the pre-trained model is set to be non-trainable for feature extraction purposes.

```
] base_model.trainable = False
```

Data augmentation layers defined in the model.

```
] data_augmentation = tf.keras.Sequential([
    tf.keras.layers.RandomFlip('horizontal'),
    tf.keras.layers.RandomRotation(0.2),
  ])
```

Code that shows the addition of the new classifier layers on top of the base model.

```
] inputs = tf.keras.Input(shape=(160, 160, 3))
  x = data_augmentation(inputs)
  x = preprocess_input(x)
  x = base_model(x, training=False)
  x = global_average_layer(x)
  x = tf.keras.layers.Dropout(0.2)(x)
  outputs = prediction_layer(x)
  model = tf.keras.Model(inputs, outputs)
```