# Transfer Learning :
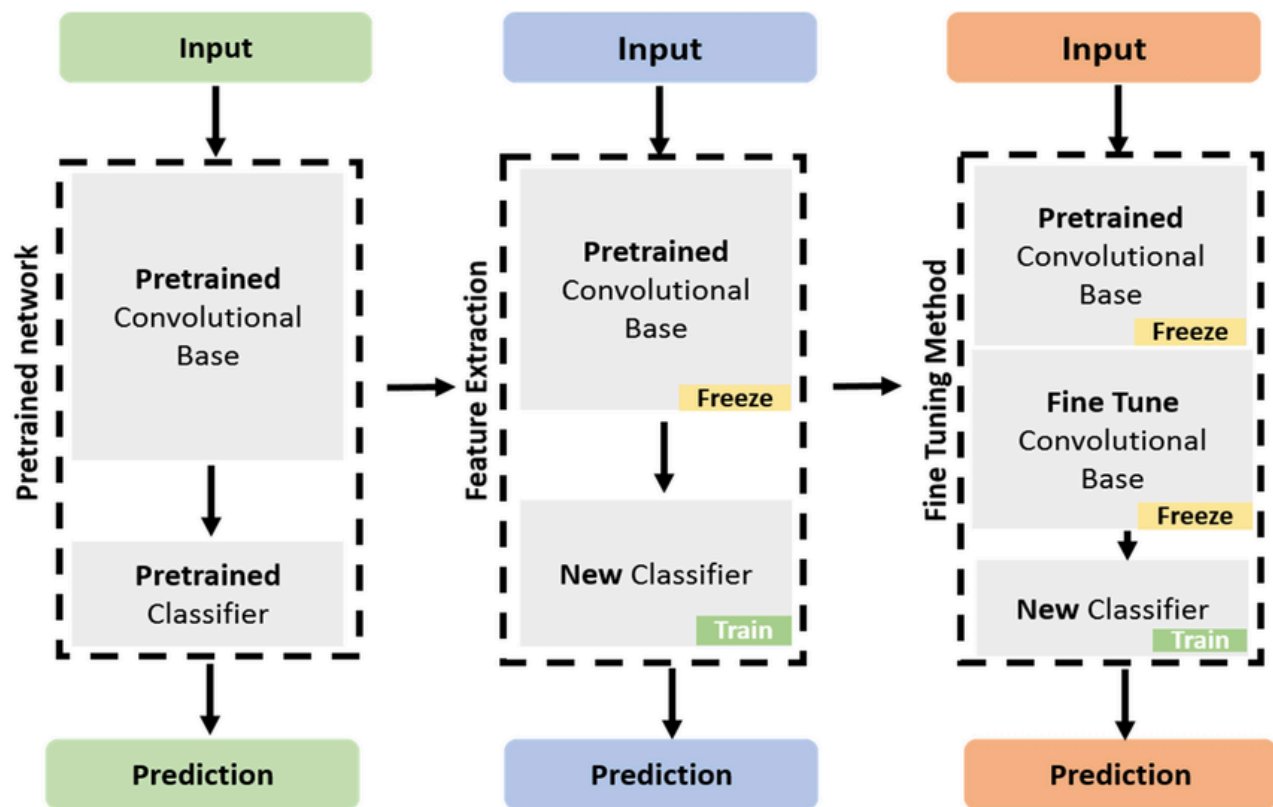
❖ Transfer learning is a machine learning technique where a model trained on one task is reused or adapted as a starting point for a model on a second related task. Instead of starting the learning process from scratch, transfer learning leverages knowledge gained from solving one problem and applies it to a different but related problem.

## Feature Extraction Technique:

**Feature extraction** involves using a pre-trained model to extract relevant features from input data, which are then used as input to a new model trained for a specific task. Here's how it typically works:

1. **Pre-trained Model**: Start with a model that has been trained on a large dataset for a related task. This model has learned to extract hierarchical features from the data.
2. **Feature Extraction**: Use the pre-trained model to extract features from the input data relevant to your task. These features represent abstract representations of the input data that capture meaningful patterns.
3. **New Task Model**: Feed these extracted features into a new model (often a simple classifier or regressor) that is trained specifically for your target task. This new model is usually smaller and faster to train compared to the original pre-trained model.
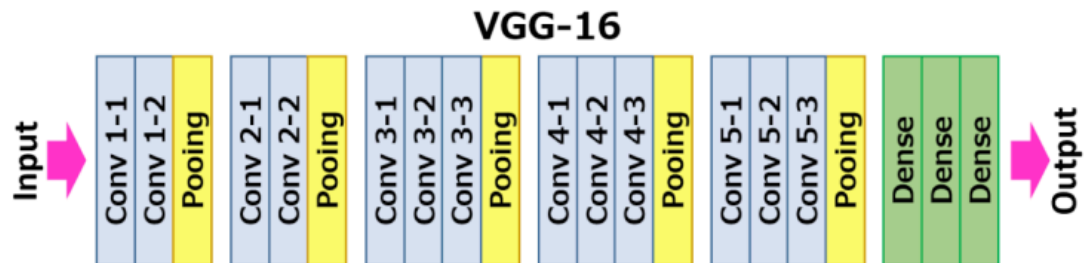
## Fine-tuning Technique:

**Fine-tuning** involves taking a pre-trained model and further training it on a new dataset or task, adjusting its parameters to better fit the new data. Here's how it's typically done:

1. **Pre-trained Model**: Start with a model that has been trained on a large dataset (e.g., ImageNet for vision tasks, or Wikipedia for language tasks).
2. **Transfer Learning Setup**: Replace the final layer(s) of the pre-trained model (typically the classification layer) with new layers suited to your specific task.
3. **Training**: Optionally, freeze the weights of some or all layers of the pre-trained model to prevent them from being updated during training (especially useful when you have limited data). Then, train the entire model (or just the new layers) on your target dataset.

**Reduced Training Time and Cost**: By leveraging pre-trained models, transfer learning reduces the amount of data and computational resources required to train a new model from scratch.

**Improved Performance**: Transfer learning allows models to start from a better initialization point, often resulting in improved performance on the target task compared to training from scratch, especially when data for the target task is limited

# VGGNET Architecture



**VGG-16**

Input → Conv 1-1 | Conv 1-2 | Pooing → Conv 2-1 | Conv 2-2 | Pooing → Conv 3-1 | Conv 3-2 | Conv 3-3 | Pooing → Conv 4-1 | Conv 4-2 | Conv 4-3 | Pooing → Conv 5-1 | Conv 5-2 | Conv 5-3 | Pooing → Dense | Dense | Dense → Output

## The Architecture

The architecture depicted below is VGG16.



$224 \times 224 \times 3$   $224 \times 224 \times 64$

$112 \times 112 \times 128$

$56 \times 56 \times 256$

$28 \times 28 \times 512$   $14 \times 14 \times 512$   $7 \times 7 \times 512$   $1 \times 1 \times 4096$   $1 \times 1 \times 1000$

- convolution+ReLU
- max pooling
- fully connected+ReLU
- softmax