# Script for Semi\_Supervised\_Transfer Learning- v2

## Import Libraries

* NumPy: Used for efficient array manipulation and mathematical operations, essential for handling image data.
* Keras Image Preprocessing: Provides utilities to load and preprocess images, making it easier to prepare data for model predictions.
* Matplotlib: Utilised for visualising images, which helps in verifying the correctness of image preprocessing and predictions.
* predict\_image Function
* Function Purpose: Predicts the class of an image using a trained model, with optional visualisation.

## Parameters:

* image\_path: Path to the image file. Ensures the function knows which image to process.
* model: The trained Keras model used for making predictions.
* class\_indices: Dictionary mapping class names to their indices, essential for interpreting the model’s output.
* target\_size: Tuple specifying the target size for the image. Ensures images are resized consistently for model input.
* visualize: Boolean flag to control whether the image should be displayed. Useful for debugging and verifying predictions.

## Image Loading and Preprocessing

* Loading Image: Uses Keras’ load\_img to load the image from the specified path, ensuring it matches the model’s expected input size.
* Image to Array: Converts the loaded image to an array format, which is required for model input.
* Expanding Dimensions: Adds a batch dimension to the array, as models expect input in batches.
* Normalisation: Scales pixel values to the range [0, 1], which is a common preprocessing step for neural networks to improve training stability and performance.

## Prediction

* Model Prediction: Uses the model to predict class probabilities for the input image.
* Class Determination: Identifies the class with the highest predicted probability using argmax.
* Class Mapping: Maps the predicted class index back to the class name using class\_indices.

## Visualisation

* Optional Visualisation: If visualize is True, displays the image with its predicted class using Matplotlib. This helps in verifying the accuracy of predictions and understanding model behaviour.

## Exception Handling

* Try-Except Block: Catches and prints any errors that occur during the prediction process. Ensures the function handles unexpected issues gracefully.

## Example Usage

* Example Call: Demonstrates how to use the predict\_image function, providing an image path and class indices from the training generator.
* Result Display: Prints the predicted class, offering a simple interface for users to understand the output.

## Reasoning

* Comprehensive Preprocessing: Ensures images are appropriately prepared for model input, improving prediction accuracy.
* Flexibility: The function can be used with any model and class indices, making it versatile for different classification tasks.
* Visual Verification: Optional visualisation helps in quickly verifying and debugging predictions.
* Error Handling: Robust handling of errors ensures the function is reliable and user-friendly.