Product Image Classifier

Objective:

The objective of this task is to develop a machine learning model capable of automatically categorizing product images into predefined groups such as fashion, nutrition, accessories, etc. This model will enhance the intelligence of an e-commerce application, providing users with more efficient browsing and search functionalities.

Dataset:

The dataset consists of product images provided by the e-commerce platform. Images can be accessed through the application or downloaded from the provided link. The dataset includes images from various product categories, and each image is labeled with its corresponding category.

Steps:

1. **Dataset Download**
   The dataset was sourced from various categories available on the Slash app. It is categorized into four distinct classes:
   - 0: Fashion
   - 1: Accessories
   - 2: Beauty
   - 3: Home

2. **Data Preparation**
   The dataset was divided into train and test, each label in the train data contain 10 images to make the data balanced and each label in the test data contain 5 images.

   Preparing the training and testing data for the model. by using the 'ImageDataGenerator' class to preprocess the images by rescaling their pixel values. Then, it creates data generators (train_generator and test_generator) from the image directories (train_path and test_path) that yield batches of images and their corresponding labels. Here, class_mode='sparse' indicates that the labels are provided as integers.

3. **Model Building**
   To address the challenge posed by the limited dataset size, experimentation with several pre-trained models was conducted, including Xception, VGG16, and MobileNet.

   We tried the three models with its basic structure and adding a final dense layer with softmax activation for multi-class classification

4. **Training**
   We compile the model using the Adam optimizer and sparse categorical cross-entropy loss function. Then, it trains the model for 5 epochs and the result was as follow:

Model	Accuracy
Xception	95%
VGG16	92%
MobileNet	88%

5. **Validation**
   The Xception model have the best performance so we retrain it in used it on the test data and it achieved 100% on the train and 94% on the test.

6. **Fine-Tuning**
   It seems like our model is overfitted so we will try make it generalize well by reconstruct our model, the base Xception model on the top acts as a feature extractor, followed by a flatten layer, two dense layers with dropout regularization, and a final dense layer with softmax activation for multi-class classification.

   The number of epochs was increased to 7

   The Xception model demonstrated superior performance, achieving 100% accuracy on both training and testing datasets.

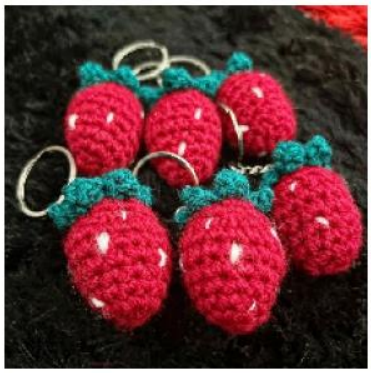
7. **Saving the model**
   We save the model architecture, weights, and optimizer state, allowing the model to be loaded and used later for inference or further training.

8. **Inference**

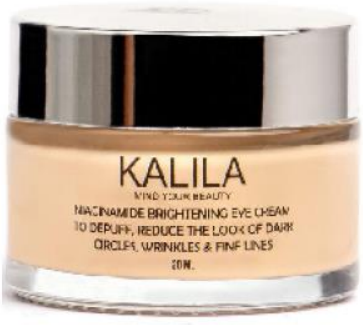
We loaded the saved model for inference and performs predictions on a set of images. then visualize the predictions by displaying the images along with their predicted class labels.

the images and their predicted class labels are plotted using matplotlib.

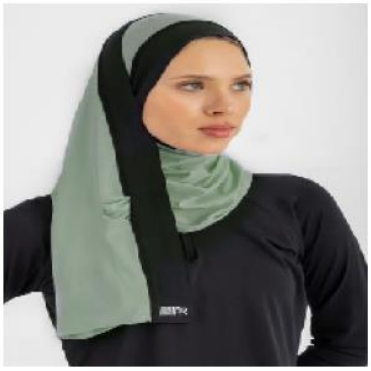
Predicted Class: Accessories



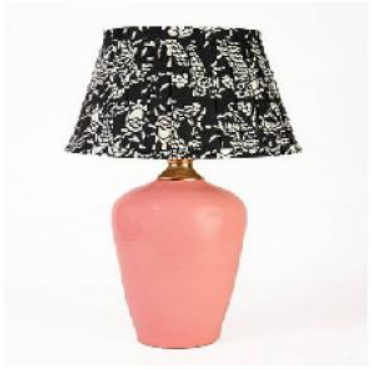
Predicted Class: Beauty



Predicted Class: Fashion



Predicted Class: Home



9. **Deployment**

   The model was deployed using Streamlit, an open-source Python library that simplifies the creation and sharing of custom web apps for machine learning and data science.

   You can interact with the web app by uploading the image and pass it to the model to make the prediction