```python
import numpy as np
import cv2
from glob import glob
from tqdm import tqdm
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.model_selection import GridSearchCV

train_path = "/kaggle/input/waste-classification-data/DATASET/TRAIN/"
test_path = "/kaggle/input/waste-classification-data/DATASET/TEST/"

def preprocess_images(image_paths):
    x_data = []
    y_data = []
    for category in image_paths:
        label = 0 if category.endswith('R') else 1  # 0 for
Recyclable, 1 for Organic
        for file in tqdm(glob(category + '/*')):
            img_array = cv2.imread(file, cv2.IMREAD_GRAYSCALE)
            img_array = cv2.resize(img_array, (64, 64))
            # Thêm vector 1 chiều vào x_data thay vì làm phẳng
            x_data.append(img_array)
            y_data.append(label)
    # Chuyển đổi x_data thành mảng numpy và chuyển từ mảng 4D
sang mảng 2D
    x_data = np.array(x_data).reshape(len(x_data), -1)
    return x_data, np.array(y_data)

# Chuẩn bị dữ liệu huấn luyện và kiểm tra
train_image_paths = glob(train_path + '*')
test_image_paths = glob(test_path + '*')

x_train, y_train = preprocess_images(train_image_paths)
x_test, y_test = preprocess_images(test_image_paths)

print("Shape of x_train:", x_train.shape)
print("Shape of x_test:", x_test.shape)
```

```
100%|██████████| 9999/9999 [01:06<00:00, 151.27it/s]
100%|██████████| 12565/12565 [01:25<00:00, 147.53it/s]
100%|██████████| 1112/1112 [00:05<00:00, 189.16it/s]
100%|██████████| 1401/1401 [00:07<00:00, 176.57it/s]

Shape of x_train: (22564, 4096)
Shape of x_test: (2513, 4096)
```

```
# Chuẩn hóa dữ liệu bằng StandardScaler
scaler = StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)

# Áp dụng PCA để giảm chiều dữ liệu
pca = PCA(n_components=100)  # Chọn số thành phần chính là 100
x_train_pca = pca.fit_transform(x_train_scaled)
x_test_pca = pca.transform(x_test_scaled)

print("Shape of x_train after PCA:", x_train_pca.shape)
print("Shape of x_test after PCA:", x_test_pca.shape)
```

```
Shape of x_train after PCA: (22564, 100)
Shape of x_test after PCA: (2513, 100)
```

```
# Định nghĩa các tham số bạn muốn tinh chỉnh
param_grid = {
    'kernel': ['linear', 'rbf', 'poly', 'sigmoid']
}

# Khởi tạo GridSearchCV với mô hình SVC và param_grid
grid_search = GridSearchCV(SVC(random_state=42), param_grid, cv=5,
verbose=2)

# Huấn luyện GridSearchCV trên dữ liệu huấn luyện đã qua PCA
grid_search.fit(x_train_pca, y_train)

# Lấy ra mô hình tối ưu
best_model = grid_search.best_estimator_

# Đánh giá độ chính xác của mô hình trên tập kiểm tra
y_pred_test = best_model.predict(x_test_pca)

test_acc = accuracy_score(y_test, y_pred_test)

print(f"Testing Accuracy with best model: {test_acc}")
print("\nClassification Report:\n", classification_report(y_test,
y_pred_test))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred_test))
```

```
Fitting 5 folds for each of 4 candidates, totalling 20 fits
[CV] END ......................................kernel=linear; total
time=46.7min
[CV] END ......................................kernel=linear; total
time=46.9min
[CV] END ......................................kernel=linear; total
time=46.5min
[CV] END ......................................kernel=linear; total
time=46.7min
[CV] END ......................................kernel=linear; total
```

```
time=47.6min
[CV] END .........................................kernel=rbf; total
time=  38.4s
[CV] END .........................................kernel=rbf; total
time=  38.3s
[CV] END .........................................kernel=rbf; total
time=  38.7s
[CV] END .........................................kernel=rbf; total
time=  38.3s
[CV] END .........................................kernel=rbf; total
time=  38.8s
[CV] END ........................................kernel=poly; total
time=  34.6s
[CV] END ........................................kernel=poly; total
time=  35.6s
[CV] END ........................................kernel=poly; total
time=  36.5s
[CV] END ........................................kernel=poly; total
time=  36.8s
[CV] END ........................................kernel=poly; total
time=  35.2s
[CV] END .....................................kernel=sigmoid; total
time=  27.8s
[CV] END .....................................kernel=sigmoid; total
time=  27.5s
[CV] END .....................................kernel=sigmoid; total
time=  27.5s
[CV] END .....................................kernel=sigmoid; total
time=  27.7s
[CV] END .....................................kernel=sigmoid; total
time=  27.3s
Testing Accuracy with best model: 0.7433346597692002

Classification Report:
               precision    recall  f1-score   support

           0       0.79      0.58      0.67      1112
           1       0.72      0.88      0.79      1401

    accuracy                           0.74      2513
   macro avg       0.75      0.73      0.73      2513
weighted avg       0.75      0.74      0.74      2513


Confusion Matrix:
 [[ 642  470]
 [ 175 1226]]
```