```
import os
import torch
from torchvision import models, transforms
from PIL import Image
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans, SpectralClustering,
AgglomerativeClustering, DBSCAN
from sklearn.cluster import BisectingKMeans
from sklearn.metrics import fowlkes_mallows_score, silhouette_score

dataset_dir = "/content/drive/MyDrive/cropped"

model = models.resnet18(pretrained=True)
model.eval()

def extract_features(image_path):
    transform = transforms.Compose([
        transforms.Resize((224, 224)),
        transforms.ToTensor(),
        transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229,
0.224, 0.225]),
    ])
    image = Image.open(image_path).convert('RGB')
    image = transform(image).unsqueeze(0)
    with torch.no_grad():
        features = model(image)
    return features.flatten().numpy()

features = []
labels = []
class_names = os.listdir(dataset_dir)
for label, class_name in enumerate(class_names):
    class_dir = os.path.join(dataset_dir, class_name)
    for image_name in os.listdir(class_dir):
        image_path = os.path.join(class_dir, image_name)
        features.append(extract_features(image_path))
        labels.append(label)

pca = PCA(n_components=2)
reduced_features = pca.fit_transform(features)

kmeans_random = KMeans(n_clusters=4, init='random',
random_state=42).fit(reduced_features)
kmeans_plus = KMeans(n_clusters=4, init='k-means++',
random_state=42).fit(reduced_features)

bisect_kmeans = BisectingKMeans(n_clusters=4,
random_state=42).fit(reduced_features)

spectral = SpectralClustering(n_clusters=4, random_state=42,
```

```
affinity='nearest_neighbors').fit(reduced_features)

dbscan = DBSCAN(eps=0.5, min_samples=5).fit(reduced_features)

agg_single = AgglomerativeClustering(n_clusters=4,
linkage='single').fit(reduced_features)
agg_complete = AgglomerativeClustering(n_clusters=4,
linkage='complete').fit(reduced_features)
agg_average = AgglomerativeClustering(n_clusters=4,
linkage='average').fit(reduced_features)
agg_ward = AgglomerativeClustering(n_clusters=4,
linkage='ward').fit(reduced_features)

methods = {
    "KMeans Random": kmeans_random.labels_,
    "KMeans Plus": kmeans_plus.labels_,
    "Bisecting KMeans": bisect_kmeans.labels_,
    "Spectral": spectral.labels_,
    "DBSCAN": dbscan.labels_,
    "Agglomerative Single": agg_single.labels_,
    "Agglomerative Complete": agg_complete.labels_,
    "Agglomerative Average": agg_average.labels_,
    "Agglomerative Ward": agg_ward.labels_,
}


evaluation_results = {}
for method, predicted_labels in methods.items():
    fmi = fowlkes_mallows_score(labels, predicted_labels)
    if len(np.unique(predicted_labels)) > 1:
        silhouette = silhouette_score(reduced_features,
predicted_labels)
    else:
        silhouette = -1
    evaluation_results[method] = {"FMI": fmi, "Silhouette":
silhouette}


ranked_by_fmi = sorted(evaluation_results.items(), key=lambda x: x[1]
["FMI"], reverse=True)
ranked_by_silhouette = sorted(evaluation_results.items(), key=lambda
x: x[1]["Silhouette"], reverse=True)

print("Ranking by Fowlkes-Mallows Index:")
for rank, (method, scores) in enumerate(ranked_by_fmi, start=1):
    print(f"{rank}. {method}: FMI={scores['FMI']}")

print("\nRanking by Silhouette Coefficient:")
for rank, (method, scores) in enumerate(ranked_by_silhouette,
```

```
start=1):
    print(f"{rank}. {method}: Silhouette={scores['Silhouette']}")
```

```
/usr/local/lib/python3.10/dist-packages/torchvision/models/
_utils.py:208: UserWarning: The parameter 'pretrained' is deprecated
since 0.13 and may be removed in the future, please use 'weights'
instead.
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/torchvision/models/_utils.py:2
23: UserWarning: Arguments other than a weight enum or `None` for
'weights' are deprecated since 0.13 and may be removed in the future.
The current behavior is equivalent to passing
`weights=ResNet18_Weights.IMAGENET1K_V1`. You can also use
`weights=ResNet18_Weights.DEFAULT` to get the most up-to-date weights.
  warnings.warn(msg)
Downloading: "https://download.pytorch.org/models/resnet18-
f37072fd.pth" to /root/.cache/torch/hub/checkpoints/resnet18-
f37072fd.pth
100%|██████████| 44.7M/44.7M [00:00<00:00, 102MB/s]
```