dipvit

November 26, 2024

1 New section

2 New section

3 New section

[ ]:
```
!pip install datasets
import os
import pandas as pd
from datasets import load_dataset
from transformers import ViTFeatureExtractor, ViTForImageClassification,␣
 ↪TrainingArguments, Trainer
from torchvision import transforms
from sklearn.metrics import accuracy_score, precision_recall_fscore_support
from PIL import Image
import torch
```

```
Collecting datasets
  Downloading datasets-3.1.0-py3-none-any.whl.metadata (20 kB)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
packages (from datasets) (3.16.1)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-
packages (from datasets) (1.26.4)
Requirement already satisfied: pyarrow>=15.0.0 in
/usr/local/lib/python3.10/dist-packages (from datasets) (17.0.0)
Collecting dill<0.3.9,>=0.3.0 (from datasets)
  Downloading dill-0.3.8-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages
(from datasets) (2.2.2)
Requirement already satisfied: requests>=2.32.2 in
/usr/local/lib/python3.10/dist-packages (from datasets) (2.32.3)
Requirement already satisfied: tqdm>=4.66.3 in /usr/local/lib/python3.10/dist-
packages (from datasets) (4.66.6)
Collecting xxhash (from datasets)
  Downloading
xxhash-3.5.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata
(12 kB)
```

```
Collecting multiprocess<0.70.17 (from datasets)
  Downloading multiprocess-0.70.16-py310-none-any.whl.metadata (7.2 kB)
Collecting fsspec<=2024.9.0,>=2023.1.0 (from
fsspec[http]<=2024.9.0,>=2023.1.0->datasets)
  Downloading fsspec-2024.9.0-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.10/dist-
packages (from datasets) (3.11.2)
Requirement already satisfied: huggingface-hub>=0.23.0 in
/usr/local/lib/python3.10/dist-packages (from datasets) (0.26.2)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-
packages (from datasets) (24.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-
packages (from datasets) (6.0.2)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (2.4.3)
Requirement already satisfied: aiosignal>=1.1.2 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (1.3.1)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-
packages (from aiohttp->datasets) (24.2.0)
Requirement already satisfied: frozenlist>=1.1.1 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (1.5.0)
Requirement already satisfied: multidict<7.0,>=4.5 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (6.1.0)
Requirement already satisfied: propcache>=0.2.0 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (0.2.0)
Requirement already satisfied: yarl<2.0,>=1.17.0 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (1.17.2)
Requirement already satisfied: async-timeout<6.0,>=4.0 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (4.0.3)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.23.0->datasets)
(4.12.2)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.32.2->datasets)
(3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests>=2.32.2->datasets) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.32.2->datasets)
(2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.32.2->datasets)
(2024.8.30)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.10/dist-packages (from pandas->datasets) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas->datasets) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-
```

```
packages (from pandas->datasets) (2024.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.8.2->pandas->datasets) (1.16.0)
Downloading datasets-3.1.0-py3-none-any.whl (480 kB)
                           480.6/480.6 kB
9.1 MB/s eta 0:00:00
Downloading dill-0.3.8-py3-none-any.whl (116 kB)
                           116.3/116.3 kB
9.3 MB/s eta 0:00:00
Downloading fsspec-2024.9.0-py3-none-any.whl (179 kB)
                           179.3/179.3 kB
13.2 MB/s eta 0:00:00
Downloading multiprocess-0.70.16-py310-none-any.whl (134 kB)
                           134.8/134.8 kB
10.0 MB/s eta 0:00:00
Downloading
xxhash-3.5.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (194 kB)
                           194.1/194.1 kB
14.9 MB/s eta 0:00:00
Installing collected packages: xxhash, fsspec, dill, multiprocess,
datasets
  Attempting uninstall: fsspec
    Found existing installation: fsspec 2024.10.0
    Uninstalling fsspec-2024.10.0:
      Successfully uninstalled fsspec-2024.10.0
ERROR: pip's dependency resolver does not currently take into account all
the packages that are installed. This behaviour is the source of the following
dependency conflicts.
gcsfs 2024.10.0 requires fsspec==2024.10.0, but you have fsspec 2024.9.0 which
is incompatible.
Successfully installed datasets-3.1.0 dill-0.3.8 fsspec-2024.9.0
multiprocess-0.70.16 xxhash-3.5.0
```

[ ]:
```python
import tarfile
import os

# Define paths
tar_path = "/content/DataSets.tar"
extract_path = "/content/dataset"

# Extract the .tar file
if tarfile.is_tarfile(tar_path):
    with tarfile.open(tar_path, "r") as tar:
        tar.extractall(path=extract_path)
        print(f"Extracted dataset to {extract_path}")
```

```
else:
    print(f"{tar_path} is not a valid tar file")
```

```
Extracted dataset to /content/dataset
```

[ ]:
```
from datasets import load_dataset

# Load the dataset
dataset = load_dataset("imagefolder", data_dir="/content/dataset")

# Print the dataset structure
print(dataset)
```

```
Generating test split: 0 examples [00:00, ? examples/s]
```

```
DatasetDict({
    test: Dataset({
        features: ['image', 'label'],
        num_rows: 6
    })
})
```

[ ]:
```
import pandas as pd

# Load calorie data
calorie_df = pd.read_csv("/content/calorie_dataset.csv")

# Verify the structure
print(calorie_df.head())
```

```
           Food            FoodSubcategory  Quantity ServingSize(g)  Fat(g)  \
0        Burger              Cheese Burger         1          100 g   27.50
1        Burger                Beef Burger         1          100 g   10.09
2        Burger             Chicken Burger         1          100 g   14.81
3        Burger              Veggie Burger         1          100 g   12.48
4  French Fries  French Fries, Oven Heated         1          117 g    3.39

   Calories
0       297
1       264
2       286
3       261
4       133
```

[ ]:
```
import os
print(os.listdir("/content/dataset"))
```

```
['food_photos', 'test_photos']
```

[ ]:
```
print(dataset)  # Check the available keys (splits)
```

```
DatasetDict({
    test: Dataset({
        features: ['image', 'label'],
        num_rows: 6
    })
})
```

[ ]:
```
import os # imports the 'os' module which provides functions for interacting␣
 ↪with the operating system.
import tarfile # imports the 'tarfile' module which provides functions for␣
 ↪working with tar archives

data_path = "/content/DataSets.tar"
extract_path = "/mnt/data/ExtractedDataSets"

if not os.path.exists(extract_path):
    with tarfile.open(data_path, 'r') as tar:
        tar.extractall(path=extract_path)
```

[ ]:
```
import pandas as pd # Imports the pandas library and assigns it the alias 'pd'.

calorie_df = pd.read_csv("/content/calorie_dataset.csv")
```

[ ]:
```
!pip install torchvision # Installs the torchvision library, which includes the␣
 ↪'transforms' module.
import torchvision.transforms as transforms # Imports the 'transforms' module␣
 ↪from torchvision and assigns it the alias 'transforms'.

transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.5, 0.5, 0.5], std=[0.5, 0.5, 0.5])
])
```

```
Requirement already satisfied: torchvision in /usr/local/lib/python3.10/dist-
packages (0.20.1+cu121)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages
(from torchvision) (1.26.4)
Requirement already satisfied: torch==2.5.1 in /usr/local/lib/python3.10/dist-
packages (from torchvision) (2.5.1+cu121)
Requirement already satisfied: pillow!=8.3.*,>=5.3.0 in
/usr/local/lib/python3.10/dist-packages (from torchvision) (11.0.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
packages (from torch==2.5.1->torchvision) (3.16.1)
Requirement already satisfied: typing-extensions>=4.8.0 in
```

```
/usr/local/lib/python3.10/dist-packages (from torch==2.5.1->torchvision)
(4.12.2)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-
packages (from torch==2.5.1->torchvision) (3.4.2)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages
(from torch==2.5.1->torchvision) (3.1.4)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages
(from torch==2.5.1->torchvision) (2024.9.0)
Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.10/dist-
packages (from torch==2.5.1->torchvision) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
/usr/local/lib/python3.10/dist-packages (from
sympy==1.13.1->torch==2.5.1->torchvision) (1.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.10/dist-packages (from jinja2->torch==2.5.1->torchvision)
(3.0.2)
```

[ ]:
```python
import torchvision.datasets as datasets  # Import the datasets module from␣
↪torchvision
import torchvision.transforms as transforms
from sklearn.model_selection import train_test_split # Import train_test_split␣
↪from sklearn.model_selection

# ... (rest of your code) ...

image_dataset = datasets.ImageFolder(root="/mnt/data/ExtractedDataSets",␣
↪transform=transform)
train_data, val_data = train_test_split(image_dataset, test_size=0.2,␣
↪random_state=42)
```

[ ]:
```python
from torch.utils.data import DataLoader # Import the DataLoader class from␣
↪torch.utils.data

train_loader = DataLoader(train_data, batch_size=32, shuffle=True)
val_loader = DataLoader(val_data, batch_size=32, shuffle=False)
```

[ ]:
```python
!pip install transformers
import torch
from transformers import ViTForImageClassification # Import the␣
↪ViTForImageClassification class

# ... rest of your code ...

# Define the device
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

model = ViTForImageClassification.from_pretrained("google/vit-base-patch16-224")
```

```
model.to(device)
```

```
Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-
packages (4.46.2)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
packages (from transformers) (3.16.1)
Requirement already satisfied: huggingface-hub<1.0,>=0.23.2 in
/usr/local/lib/python3.10/dist-packages (from transformers) (0.26.2)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-
packages (from transformers) (1.26.4)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from transformers) (24.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-
packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.10/dist-packages (from transformers) (2024.9.11)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (from transformers) (2.32.3)
Requirement already satisfied: safetensors>=0.4.1 in
/usr/local/lib/python3.10/dist-packages (from transformers) (0.4.5)
Requirement already satisfied: tokenizers<0.21,>=0.20 in
/usr/local/lib/python3.10/dist-packages (from transformers) (0.20.3)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-
packages (from transformers) (4.66.6)
Requirement already satisfied: fsspec>=2023.5.0 in
/usr/local/lib/python3.10/dist-packages (from huggingface-
hub<1.0,>=0.23.2->transformers) (2024.9.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.10/dist-packages (from huggingface-
hub<1.0,>=0.23.2->transformers) (4.12.2)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests->transformers) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers)
(2024.8.30)
```

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_auth.py:94:
UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab
(https://huggingface.co/settings/tokens), set it as secret in your Google Colab
and restart your session.
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access
```

```
public models or datasets.
  warnings.warn(

config.json:   0%|          | 0.00/69.7k [00:00<?, ?B/s]

model.safetensors:   0%|          | 0.00/346M [00:00<?, ?B/s]
```

```
[ ]: ViTForImageClassification(
       (vit): ViTModel(
         (embeddings): ViTEmbeddings(
           (patch_embeddings): ViTPatchEmbeddings(
             (projection): Conv2d(3, 768, kernel_size=(16, 16), stride=(16, 16))
           )
           (dropout): Dropout(p=0.0, inplace=False)
         )
         (encoder): ViTEncoder(
           (layer): ModuleList(
             (0-11): 12 x ViTLayer(
               (attention): ViTSdpaAttention(
                 (attention): ViTSdpaSelfAttention(
                   (query): Linear(in_features=768, out_features=768, bias=True)
                   (key): Linear(in_features=768, out_features=768, bias=True)
                   (value): Linear(in_features=768, out_features=768, bias=True)
                   (dropout): Dropout(p=0.0, inplace=False)
                 )
                 (output): ViTSelfOutput(
                   (dense): Linear(in_features=768, out_features=768, bias=True)
                   (dropout): Dropout(p=0.0, inplace=False)
                 )
               )
               (intermediate): ViTIntermediate(
                 (dense): Linear(in_features=768, out_features=3072, bias=True)
                 (intermediate_act_fn): GELUActivation()
               )
               (output): ViTOutput(
                 (dense): Linear(in_features=3072, out_features=768, bias=True)
                 (dropout): Dropout(p=0.0, inplace=False)
               )
               (layernorm_before): LayerNorm((768,), eps=1e-12,
     elementwise_affine=True)
               (layernorm_after): LayerNorm((768,), eps=1e-12,
     elementwise_affine=True)
             )
           )
         )
         (layernorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
       )
       (classifier): Linear(in_features=768, out_features=1000, bias=True)
```

```
)
```

[ ]:
```
optimizer = torch.optim.AdamW(model.parameters(), lr=5e-5)
criterion = torch.nn.CrossEntropyLoss()
```

[ ]:
```
def train_model(model, train_loader, val_loader, epochs=5):
    for epoch in range(epochs):
        model.train()
        train_loss = 0.0
        for images, labels in train_loader:
            images, labels = images.to(device), labels.to(device)
            optimizer.zero_grad()
            outputs = model(images).logits
            loss = criterion(outputs, labels)
            loss.backward()
            optimizer.step()
            train_loss += loss.item()
```

[ ]:
```
def estimate_calories(image_path, model, calorie_df, feature_extractor):
    image = Image.open(image_path).convert("RGB")
    inputs = feature_extractor(images=image, return_tensors="pt").to(device)

    outputs = model(**inputs).logits
    predicted_class = outputs.argmax(dim=1).item()
    food_item = image_dataset.classes[predicted_class]

    calories = calorie_df[calorie_df['food_item'] == food_item]['calories'].
 ↪values[0]
    return food_item, calories
```

[ ]:
```
!pip install transformers
from transformers import ViTFeatureExtractor, ViTForImageClassification
from PIL import Image

# ... your existing code for model loading and training ...

def estimate_calories(image_path, model, calorie_df, feature_extractor):
    image = Image.open(image_path).convert("RGB")

    # Initialize the feature extractor here:
    feature_extractor = ViTFeatureExtractor.from_pretrained("google/
 ↪vit-base-patch16-224")

    inputs = feature_extractor(images=image, return_tensors="pt").to(device)

    outputs = model(**inputs).logits
    predicted_class = outputs.argmax(dim=1).item()
```

```
    # ... rest of your function ...
```

```
Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-
packages (4.46.2)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
packages (from transformers) (3.16.1)
Requirement already satisfied: huggingface-hub<1.0,>=0.23.2 in
/usr/local/lib/python3.10/dist-packages (from transformers) (0.26.2)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-
packages (from transformers) (1.26.4)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from transformers) (24.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-
packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.10/dist-packages (from transformers) (2024.9.11)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (from transformers) (2.32.3)
Requirement already satisfied: safetensors>=0.4.1 in
/usr/local/lib/python3.10/dist-packages (from transformers) (0.4.5)
Requirement already satisfied: tokenizers<0.21,>=0.20 in
/usr/local/lib/python3.10/dist-packages (from transformers) (0.20.3)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-
packages (from transformers) (4.66.6)
Requirement already satisfied: fsspec>=2023.5.0 in
/usr/local/lib/python3.10/dist-packages (from huggingface-
hub<1.0,>=0.23.2->transformers) (2024.9.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.10/dist-packages (from huggingface-
hub<1.0,>=0.23.2->transformers) (4.12.2)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests->transformers) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers)
(2024.8.30)
```

[ ]:
```
import torch
from torchvision import transforms
from transformers import ViTForImageClassification, ViTImageProcessor
from PIL import Image
import pandas as pd

# Step 1: Load the Calorie Dataset
```

```
calorie_df = pd.read_csv("/content/calorie_dataset.csv")
calorie_mapping = calorie_df.set_index("Food")["Calories"].to_dict()

# Step 2: Define the Image Preprocessing
preprocess = transforms.Compose([
    transforms.Resize((224, 224)),  # Resize to 224x224 for ViT
    transforms.ToTensor(),          # Convert to PyTorch tensor
    transforms.Normalize(           # Normalize as per ViT requirements
        mean=[0.485, 0.456, 0.406],
        std=[0.229, 0.224, 0.225]
    ),
])

# Step 3: Load the Pre-trained ViT Model
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model = ViTForImageClassification.from_pretrained("google/vit-base-patch16-224")
model.to(device)
model.eval()

# Define a mapping of class indices to food items (update this based on your␣
 ↪dataset)
class_to_food = {0: "Pizza", 1: "Burger", 2: "Salad", 3: "Pasta", 4: "Sandwich"}

# Step 4: Process the Image
def process_image(image_path):
    image = Image.open('/content/pic_002.jpg').convert("RGB")
    input_tensor = preprocess(image).unsqueeze(0)  # Add batch dimension
    return input_tensor.to(device)

# Step 5: Predict Food Item and Calories
def predict_calories(image_path):
    # Process the image
    input_tensor = process_image(image_path)

    # Run inference
    with torch.no_grad():
        outputs = model(input_tensor).logits  # Calculate outputs
        predicted_class = torch.argmax(outputs, dim=1).item()  # Get predicted␣
 ↪class

    # Map class to food item
    food_item = class_to_food.get(predicted_class, "Unknown")

    # Fetch calorie value
    calories = calorie_mapping.get(food_item, "Calorie data not available")

    # Debugging prints
```

```
    print("Logits:", outputs)
    print("Predicted Class ID:", predicted_class)

    return food_item, calories

# Step 6: Use the System
image_path = "/content/pic_002.jpg"  # Replace with the correct path
food_item, calories = predict_calories(image_path)

print(f"Recognized Food Item: {food_item}")
print(f"Estimated Calories: {calories}")
```

```
Logits: tensor([[ 4.9355e-02,  1.9270e+00, -1.7401e+00,  3.7952e-01,
-7.1113e-01,
          8.1342e-02, -3.1654e-01, -7.3184e-01, -2.4319e-01, -1.1480e+00,
         -1.6180e+00, -1.7898e+00, -1.1489e+00, -9.3738e-01, -1.3830e+00,
         -1.8446e+00, -5.3947e-01, -3.3788e-01, -6.7697e-01, -1.6844e+00,
         -1.0728e+00,  2.0483e-01, -8.3274e-01, -7.8007e-01, -7.6632e-01,
         -6.9818e-01, -5.4468e-01,  4.1681e-01, -6.9453e-01, -1.3012e-01,
         -4.7862e-01, -9.2269e-01,  1.7592e-01, -2.3999e-01,  2.7386e-01,
         -9.1653e-01, -3.4281e-01, -8.0176e-01, -9.6174e-01, -1.7852e+00,
         -6.5785e-01, -1.1774e+00, -1.1787e+00, -1.0944e+00, -3.4741e-01,
         -2.8202e-01, -1.5513e+00, -6.0279e-01, -9.9559e-01, -7.1564e-01,
          2.7780e-01,  2.8914e-01, -8.0736e-01, -2.6192e-01,  1.7956e-01,
         -1.4034e+00, -8.7790e-01, -8.3739e-01, -9.4629e-01, -9.4100e-01,
          1.0814e+00, -1.3996e+00, -8.5766e-01, -1.3924e+00, -9.4839e-01,
         -9.9199e-01, -1.6576e-01,  2.2275e-01,  6.1043e-01,  8.4929e-01,
         -7.6218e-01, -8.2327e-01, -7.5616e-01,  2.8924e-02,  1.6915e-01,
         -9.5943e-01, -7.6790e-01, -3.0132e-01,  7.2183e-01, -8.5211e-02,
          5.4199e-02, -1.1606e+00, -8.8651e-01, -4.9808e-01, -4.9188e-01,
         -1.5117e-01, -8.0659e-01, -2.5561e-01, -1.3048e+00,  7.3945e-02,
         -1.4890e+00, -1.2108e+00, -1.8987e+00, -9.9000e-01, -4.6181e-01,
         -9.9285e-01, -1.2463e+00, -5.0889e-01, -3.4943e-01, -1.0882e+00,
         -8.2767e-01, -3.4559e-01, -7.8062e-01, -7.7702e-01, -2.1933e-01,
          1.2505e+00,  5.9142e-02, -1.8730e-01, -2.2288e-01, -9.3156e-02,
         -1.0879e+00, -3.6166e-01,  1.2668e+00, -1.3480e-01,  2.9693e-01,
         -5.3342e-01, -8.3514e-02,  1.8140e+00,  6.0002e-01, -9.3957e-02,
         -8.2399e-01,  2.5702e+00,  1.2729e+00,  2.6514e-01,  1.2536e+00,
         -4.2350e-01, -8.9139e-01,  1.5123e-01, -1.7596e-01,  6.8149e-01,
         -1.0624e+00, -7.1497e-01, -2.6696e-01, -2.0227e+00, -5.0072e-01,
         -4.8801e-01,  1.6052e-01, -5.6549e-03, -8.8193e-01, -1.7102e+00,
         -1.0707e+00, -8.0588e-02, -8.6835e-01, -5.6791e-01, -6.0839e-02,
          6.7146e-01, -4.7156e-01, -3.8032e-01, -8.8724e-01, -1.1618e-01,
         -6.7124e-01, -7.0045e-01,  3.8739e-01,  2.4151e-01,  2.6278e-01,
         -1.2709e+00, -1.6140e+00, -2.8299e-02, -9.3022e-01, -4.5723e-01,
         -4.0534e-01, -1.5662e+00, -1.4292e+00, -6.5250e-01, -1.2533e+00,
         -9.3870e-01, -2.3274e+00, -1.9628e+00, -5.3039e-01, -1.0226e-01,
```

```
-1.1180e+00, -9.8388e-02, -1.8224e+00, -1.9132e+00, -1.1523e+00,
-1.7829e+00, -1.1040e+00, -7.4623e-01, -1.3095e+00, -9.9288e-01,
-1.0480e+00, -9.6264e-01, -2.2346e+00, -7.0809e-02, -4.3830e-01,
-5.0287e-01, -1.3328e+00, -1.2213e+00, -1.6393e+00, -4.6042e-01,
-3.4526e-01, -5.6926e-01, -2.0862e+00, -1.5526e+00, -6.3212e-01,
 3.5434e-01,  6.7005e-01,  4.9092e-01,  6.5222e-02,  3.3304e-01,
-1.2159e+00, -1.2536e+00, -1.2293e+00, -9.2062e-02, -1.2864e+00,
-5.1133e-01, -4.3074e-01, -1.5245e+00, -2.4872e+00, -5.6198e-01,
-8.8182e-01, -5.8700e-01, -4.9215e-01,  1.4508e-01, -3.4315e-01,
-1.0645e+00, -1.3236e+00, -8.0087e-01, -1.6635e+00, -9.3172e-01,
-5.7405e-01, -1.0595e+00, -7.3191e-01, -6.3217e-01,  2.3184e-01,
-1.0296e-01, -6.2153e-01, -4.7179e-01, -7.7975e-01,  5.2990e-01,
 4.7021e-02, -7.5327e-01, -7.7983e-01, -4.6103e-01, -1.1079e+00,
-5.6566e-01, -2.4456e-01,  1.9080e-01, -1.2234e+00, -7.0587e-01,
-1.0347e+00, -1.4250e+00, -4.5566e-01, -1.1760e+00, -1.2008e+00,
-1.2912e-01, -1.9042e+00, -1.4840e+00, -9.0982e-01, -2.7746e-01,
-2.0669e-01, -1.2829e+00, -1.1619e+00, -5.5032e-01, -1.2060e+00,
-1.2410e+00, -1.2069e-01, -6.7211e-01, -1.2386e-02,  5.2055e-02,
-3.5387e-01, -6.4947e-01, -5.2304e-01, -9.0620e-01, -1.6430e+00,
 7.1617e-02, -1.5293e+00, -1.4171e+00, -1.6552e+00, -1.3154e+00,
-1.2647e+00, -1.7857e+00, -1.3462e+00, -6.5342e-01, -1.7472e+00,
-2.3145e+00, -2.2113e+00, -1.6922e+00, -2.5230e+00, -1.3760e+00,
-2.3947e+00, -2.7248e+00, -1.9695e+00, -4.4399e-01, -2.4972e-01,
-1.9308e+00, -1.0376e+00, -1.3956e+00,  3.6069e-01, -5.6631e-02,
-7.4957e-02, -1.5186e-01, -1.5432e+00, -6.8868e-01,  5.9930e-02,
-5.1589e-02, -1.6090e-01,  6.5956e-01, -1.9108e+00, -1.5641e+00,
-1.1857e+00, -2.3124e-01, -6.4739e-01, -8.3409e-01,  4.4540e-01,
-6.1991e-01, -1.0067e+00, -1.8719e-01, -5.5808e-01, -8.7695e-02,
 2.5375e-01, -4.4845e-01, -1.1742e+00, -8.9917e-01,  2.0905e-02,
-9.9289e-01, -3.4495e-01, -4.6849e-01, -5.7760e-01, -5.1922e-01,
-9.1136e-01, -1.3642e+00, -5.8131e-01, -9.1720e-01, -9.8010e-01,
-6.4383e-01, -7.5129e-01,  1.5133e+00,  1.5849e-01, -1.0197e+00,
-5.2969e-01, -1.2220e+00, -3.8466e-01, -1.1097e+00, -6.1904e-01,
-6.8921e-01, -1.2380e+00, -5.9703e-01, -1.8617e+00, -5.5634e-01,
-1.2669e-01, -1.1863e+00, -6.4247e-01, -4.8141e-01, -1.4245e+00,
-4.2123e-01,  5.0289e-02, -2.1587e-01, -1.2854e+00, -1.3372e+00,
-1.0655e+00,  4.7293e-01, -8.8375e-01, -6.3958e-01, -3.2926e-01,
-6.8739e-01, -2.0109e+00, -1.6337e+00, -1.6860e+00, -2.1037e+00,
-1.5863e+00,  6.1323e-01, -1.2196e+00, -3.2024e-01, -7.8418e-01,
-8.3066e-01, -8.3600e-01, -1.2241e+00, -1.0980e+00, -5.5047e-01,
-1.3203e+00, -1.1391e+00, -6.7387e-01, -9.1312e-01, -1.2123e+00,
-5.3462e-01, -2.6786e-01, -1.1597e+00, -1.4165e+00, -7.4759e-01,
 1.1341e-01, -1.4117e+00, -8.5494e-01, -2.6683e-01, -3.5902e-01,
-1.1292e+00, -8.0525e-01, -9.4508e-01, -9.7987e-01,  5.0248e-01,
-5.1523e-01, -4.2477e-01,  5.1746e-01, -8.0354e-01, -3.3561e-01,
 1.4065e-01, -4.9667e-01, -2.5358e-01,  1.7241e+00,  3.1665e-01,
 2.3679e-01, -1.1114e-01,  6.4214e-01, -7.9579e-01, -5.0759e-01,
 1.0496e+00,  4.8165e-01,  3.7254e-01, -4.2979e-01,  1.8448e+00,
```

```
-4.3707e-01,  6.1779e-02,  6.4020e-01, -4.8998e-01,  1.3406e+00,
 4.4813e+00,  6.6532e-01, -5.4765e-01, -1.3303e-01,  6.9174e-01,
 1.7240e-01,  7.9442e-01, -4.0718e-01, -1.9168e-01,  5.7407e-01,
-5.0142e-01,  9.4194e-01,  1.2681e+00, -3.7723e-01,  9.5602e-03,
-4.0250e-01,  1.3372e-01, -4.5444e-01,  1.1988e-01, -1.3564e-01,
 2.7461e-02,  8.2845e-02,  2.5362e-01,  1.5099e+00,  9.1986e-01,
 1.5247e+00,  5.9364e-01,  1.0381e+00,  8.5583e-01, -3.8461e-01,
 1.7176e-01, -1.3083e-01, -2.8359e-01, -2.7765e-01, -3.1008e-01,
 1.3212e+00,  2.8628e-01,  7.2681e-01,  3.1393e-01, -3.2675e-02,
 1.4992e+00,  4.4760e-01, -1.2313e-01,  5.4204e-01,  2.6004e-01,
-1.2458e+00,  1.2720e+00, -9.1132e-01,  8.0391e-01,  2.2097e+00,
 9.6485e-01, -1.2652e+00,  1.6466e+00, -3.6513e-01,  1.2956e+00,
 3.4938e+00, -1.2251e+00, -1.0157e+00,  5.1141e-01, -7.8575e-01,
-4.1936e-01,  1.0570e+00,  1.0380e-01,  2.0241e-01, -2.7298e-01,
 7.2900e-01,  5.0477e-01,  2.2948e-01, -6.3945e-01, -8.9473e-02,
-7.4049e-02, -7.7390e-01,  4.1257e-01,  6.3012e-01, -1.2133e-01,
 7.1546e-01, -8.1463e-01,  7.5490e-01,  6.1227e-01,  1.4422e+00,
 2.8692e-01,  1.3811e+00,  9.8607e-01,  1.3162e+00,  1.7020e+00,
-4.3668e-01,  2.1025e-01,  3.2561e-01,  2.3877e-01,  8.5689e-01,
 1.7696e+00,  1.5228e+00,  1.4575e+00, -8.1036e-01,  1.3171e+00,
-3.9260e-01,  6.0008e-02,  1.7796e+00, -4.9884e-01, -5.0243e-01,
 1.3885e-01,  2.6095e-01,  9.7082e-01,  3.1077e-01,  5.1366e-02,
 1.2313e-01,  1.0669e+00,  1.5115e-01, -4.9210e-01,  3.8088e-01,
-4.4411e-01, -5.8325e-01, -5.8320e-01, -2.5762e-01,  1.5032e+00,
 9.5346e-01,  1.1482e-01,  6.2133e-01,  4.1087e-01,  4.8612e-02,
 7.0351e-01, -4.4909e-01, -1.0658e+00,  2.7849e-01,  4.8385e-01,
-1.6584e+00,  5.9835e-01,  6.0141e-01, -1.6512e-01,  1.6850e+00,
-1.5039e-01,  3.4249e-01, -7.6093e-01, -7.9419e-01,  6.1657e-01,
 1.3424e+00,  1.4975e+00,  3.7160e-01,  1.7788e-01, -3.7160e-01,
-1.3754e-02,  1.1428e+00,  1.4090e+00,  7.7176e-01,  7.5578e-01,
-3.9935e-02, -7.5442e-02,  1.9465e-01,  1.0803e+00, -2.8690e-01,
-1.4569e-01,  6.1429e-02,  2.9868e+00,  1.2040e-01, -6.0485e-01,
 3.0894e-01,  6.7006e-01,  9.1899e-01,  5.3540e-01,  4.1685e-01,
-1.7485e-01,  3.6086e-01,  1.5370e+00,  4.3530e-02, -6.2403e-02,
-5.7106e-01, -2.7141e-01,  2.3811e+00,  2.7403e-01,  1.3418e+00,
 5.4251e-01,  3.5100e-01,  9.4829e-01,  1.5122e+00,  6.2721e-01,
 8.8093e-01,  8.8803e-02, -1.2857e-02, -1.2500e-01, -3.2035e-02,
-7.6271e-01, -1.6037e-01,  4.0476e-01, -7.0479e-01,  1.7227e+00,
 6.8532e-01,  2.1923e-01,  8.5967e-01, -1.2877e-01,  3.9174e-01,
 6.0360e-01,  3.3442e-01,  1.1113e+00,  7.7524e-01, -1.2239e+00,
-7.2519e-01,  1.8434e+00,  2.7956e-01,  3.2556e-01, -1.1883e-01,
 4.3105e-01, -5.8655e-01,  4.8623e-01,  1.2459e+00,  8.3735e-01,
-4.5573e-01, -1.3935e-01,  1.3004e+00,  8.8254e-01,  9.1922e-01,
-6.8676e-01,  1.3616e+00,  6.0279e-01, -3.9522e-02,  1.3682e+00,
-8.6430e-01, -1.2026e-01, -5.2377e-01,  7.6381e-01, -1.2573e+00,
 1.2221e+00,  1.6941e+00,  3.4519e-01,  2.7868e-01,  7.6577e-02,
-3.8942e-01,  1.6139e+00,  2.0376e-01,  1.1660e+00,  1.4264e+00,
-1.4737e-01,  1.4974e+00,  4.5906e-01,  1.0494e+00,  4.4859e-01,
```

```
 2.8789e-01,  1.5975e+00,  9.0323e-01,  4.8979e-01, -4.3672e-01,
-9.2141e-01, -1.3244e+00,  8.5121e-01,  5.5864e-01,  1.6491e+00,
 5.0833e-01, -6.3686e-01,  2.8190e-01,  1.1140e+00, -5.4455e-01,
-1.1078e+00,  1.0708e+00,  1.6438e-01,  1.7631e-01, -6.9314e-01,
-1.2410e+00, -6.0519e-01, -7.4582e-01,  2.6086e-01,  8.3709e-01,
-6.8256e-01, -7.1112e-01,  3.7628e-01,  3.7522e-01,  6.8729e-01,
 1.1319e+00, -7.8211e-01,  9.0751e-01,  6.7976e-02,  5.2301e-01,
-1.8147e-01,  1.2771e+00,  8.1682e-01,  1.0358e+00, -1.0644e+00,
-3.2979e-01, -1.1783e-01,  2.4111e+00, -7.6687e-01, -1.0699e+00,
-5.1829e-01,  3.2368e-01, -6.8552e-01,  8.6148e-01,  1.3107e+00,
 1.7734e+00, -1.9457e-01,  9.9788e-01, -1.5143e+00, -4.9337e-01,
-7.3314e-01,  5.1991e-01,  5.4702e-01,  1.2075e+00,  1.1462e+00,
 7.1322e-01, -2.8192e-02,  2.0343e+00,  6.5899e-01,  8.5035e-01,
 9.6039e-01, -4.7040e-01, -6.8128e-01,  8.5581e-01,  3.9485e-01,
 1.1786e+00,  7.8180e-01,  9.7662e-01,  7.4694e-01, -7.7551e-01,
 9.2742e-01, -2.9245e-01, -8.1922e-01, -6.6267e-01,  9.0401e-01,
 3.7294e-01, -1.0654e-02,  2.1798e-01,  2.5093e+00,  6.7083e-01,
-4.2903e-01,  6.9173e-01,  9.0300e-01,  1.1550e+00,  2.5453e-01,
 5.6456e-01,  9.1382e-01, -8.8052e-01,  3.1846e-01, -1.8660e-01,
-4.0287e-01,  1.6445e+00, -5.6400e-02,  1.7891e+00,  1.1038e+00,
-2.6823e-01,  2.5190e-02,  6.5836e-01, -1.0020e+00,  3.4182e-01,
-4.6459e-01, -4.5645e-01, -1.7253e-01, -1.3872e-02,  1.2330e+00,
 1.3461e+00, -3.3123e-01,  3.7087e+00,  5.4120e-01, -5.8285e-02,
-1.1272e+00,  9.1908e-01,  1.1426e+00, -2.4662e-03,  1.4780e+00,
-1.1914e-01,  3.4965e-01,  1.6614e+00,  9.3736e-01, -3.1602e-02,
-3.8709e-01,  1.5007e-01,  4.6054e-01,  6.7700e-01,  7.0924e-01,
-9.3720e-01,  1.1092e+00,  5.7237e-01,  4.7074e-01,  7.1475e-01,
 1.1487e-02,  9.4331e-01,  1.2549e+00,  2.6711e-01, -1.5369e-01,
 8.0142e-01,  1.1211e+00,  2.5995e-01,  8.4020e-01, -7.0467e-01,
-3.7644e-01,  5.0085e-01,  5.8202e-02,  8.6176e-01,  4.6135e-01,
 2.3134e+00, -8.2769e-01,  2.8595e-01,  1.1113e-01,  5.6403e-01,
-2.2373e-01, -1.4076e+00,  9.8619e-01,  1.8322e+00,  4.2702e-01,
-5.5289e-01,  2.9575e-01, -7.7234e-02,  2.9694e+00,  3.1718e-01,
-6.4007e-02, -1.0380e+00, -6.4046e-01,  5.3811e-01, -2.6168e-01,
-5.3951e-01,  1.0711e+00,  1.3116e+00,  1.1778e+00,  1.9425e-01,
-7.7104e-01,  6.6479e-01,  1.0855e+00,  7.8241e-01,  9.8975e-01,
 1.9232e-01, -9.3943e-02,  5.8768e-01, -1.1746e+00,  5.2983e-02,
 6.5368e-01,  5.2846e-01, -8.5888e-02,  2.8321e-01, -1.4889e-01,
-1.2794e+00, -4.0831e-01, -1.5799e-01, -9.3868e-01,  7.4030e-01,
 7.7965e-01,  7.2574e-01,  2.7949e-02,  2.4043e-01,  3.6256e-01,
-3.3642e-01,  1.9004e-01,  5.3702e-01, -1.2676e+00,  6.3054e-01,
 1.0525e+00,  6.9434e-01,  6.8766e-01,  6.4942e-03,  8.2977e-01,
 1.7038e+00,  2.0383e-01,  5.1688e-01,  1.2683e+00, -5.8216e-01,
 3.9049e-01,  6.7690e-01, -1.2807e+00,  2.7270e+00, -2.0535e-01,
-5.3956e-01, -5.9522e-01, -3.8708e-02, -4.3594e-01,  9.2142e-01,
-4.5581e-02, -1.9167e-01,  9.9870e-01, -9.7471e-01, -8.0738e-01,
-6.1810e-01,  1.8601e-01, -6.3623e-01,  9.4361e-02,  1.2841e+00,
 6.1458e-01,  1.8145e+00,  1.6536e+00,  3.3617e-01, -3.8466e-01,
```

```
          -2.9933e-01,  2.2698e+00,  5.5722e-03,  4.7272e-01,  3.5369e-01,
          -5.0063e-01,  1.2451e+00,  9.9699e-01,  7.9239e-01,  1.4392e+00,
           6.0982e-01,  1.4248e-01,  9.4452e-01, -3.0732e-01, -1.9954e-01,
          -5.9298e-01,  4.4030e-01,  9.1355e-01, -2.6681e-01,  1.2014e+00,
           1.1969e+00,  1.7399e-01, -1.1907e+00,  6.2447e-03, -8.2014e-01,
          -6.1113e-01,  4.0423e-01,  2.5732e-01,  6.3412e-01,  1.9309e+00,
           1.3842e+00,  6.0811e-02,  1.5702e+00,  3.3525e+00,  6.4051e-01,
          -3.6171e-01,  1.3255e+00,  3.6808e+00,  1.5412e+00, -3.4876e-02,
           2.9624e+00,  3.5363e+00,  3.2336e+00,  2.1716e+00,  3.0114e+00,
           7.4788e-02, -3.1107e-01,  6.3696e-01,  5.2592e-01,  1.7024e+00,
           1.7902e+00,  5.3629e-01,  1.9472e+00,  9.4360e-03, -6.8297e-01,
           1.8841e+00, -1.3072e+00,  5.9471e-01,  1.1118e+00,  3.5786e+00,
           1.8841e+00,  1.3751e+00,  1.5593e+00,  1.6164e+00,  1.3390e+00,
           4.6854e-01,  1.4715e+00,  1.9049e+00, -2.9193e+00,  3.1339e+00,
           1.4137e+00,  3.0831e+00,  1.6817e+00,  1.2486e+01,  3.2403e+00,
           1.2221e+00,  2.0545e+00,  1.3362e+00,  1.5351e+00,  3.1112e+00,
          -2.0474e-01,  1.5839e+00, -9.8723e-01, -5.0235e-02,  1.8752e-01,
           8.0310e-01,  1.4266e-01, -1.6200e-02,  4.1712e-02, -5.7352e-01,
           2.9426e-01, -4.5754e-01,  6.0879e-01, -5.5425e-01, -6.8887e-01,
          -4.9249e-01, -1.2048e+00,  1.0009e+00,  6.3239e-01,  3.6786e-01,
           2.7396e-01, -3.1446e-01,  8.3168e-01, -7.9736e-01, -3.8105e-01,
           1.9371e-01,  1.4207e+00, -4.0655e-01,  1.5168e+00,  3.7392e-02]])
Predicted Class ID: 963
Recognized Food Item: Unknown
Estimated Calories: Calorie data not available
```

[ ]:
```python
import tarfile
import os
from datasets import load_dataset

# Paths
tar_path = "/content/DataSets.tar"
extract_dir = "/content/dataset"

# Extract if not already done
if not os.path.exists(extract_dir):
    with tarfile.open(tar_path, "r") as tar:
        tar.extractall(path=extract_dir)

# Check the directory structure
for root, dirs, files in os.walk(extract_dir):
    print(f"Directory: {root}")
    for file in files:
        print(f"  File: {file}")

# Load the dataset
dataset = load_dataset("imagefolder", data_dir=extract_dir)
```

```
print(dataset)
```

```
Directory: /content/dataset
Directory: /content/dataset/food_photos
Directory: /content/dataset/food_photos/Pizza
  File: pic_044.jpg
  File: pic_053.jpg
  File: pic_299.jpg
  File: pic_255.jpg
  File: pic_119.jpg
  File: pic_368.jpg
  File: pic_376.jpg
  File: pic_069.jpg
  File: pic_329.jpg
  File: pic_055.jpg
  File: pic_148.jpg
  File: pic_289.jpg
  File: pic_209.jpg
  File: pic_228.jpg
  File: pic_149.jpg
  File: pic_340.jpg
  File: pic_252.jpg
  File: pic_147.jpg
  File: pic_399.jpg
  File: pic_326.jpg
  File: pic_108.jpg
  File: pic_146.jpg
  File: pic_243.jpg
  File: pic_051.jpg
  File: pic_198.jpg
  File: pic_239.jpg
  File: pic_173.jpg
  File: pic_301.jpg
  File: pic_063.jpg
  File: pic_043.jpg
  File: pic_378.jpg
  File: pic_398.jpg
  File: pic_336.jpg
  File: pic_248.jpg
  File: pic_263.jpg
  File: pic_379.jpg
  File: pic_020.jpg
  File: pic_360.jpg
  File: pic_374.jpg
  File: pic_369.jpg
  File: pic_177.jpg
  File: pic_201.jpg
```

File: pic_058.jpg
File: pic_081.jpg
File: pic_375.jpg
File: pic_284.jpg
File: pic_088.jpg
File: pic_316.jpg
File: pic_377.jpg
File: pic_060.jpg
File: pic_303.jpg
File: pic_387.jpg
File: pic_310.jpg
File: pic_335.jpg
File: pic_271.jpg
File: pic_182.jpg
File: pic_238.jpg
File: pic_397.jpg
File: pic_037.jpg
File: pic_192.jpg
File: pic_079.jpg
File: pic_314.jpg
File: pic_214.jpg
File: pic_074.jpg
File: pic_405.jpg
File: pic_095.jpg
File: pic_174.jpg
File: pic_098.jpg
File: pic_296.jpg
File: pic_280.jpg
File: pic_211.jpg
File: pic_036.jpg
File: pic_282.jpg
File: pic_266.jpg
File: pic_062.jpg
File: pic_240.jpg
File: pic_234.jpg
File: pic_229.jpg
File: pic_311.jpg
File: pic_120.jpg
File: pic_094.jpg
File: pic_184.jpg
File: pic_362.jpg
File: pic_288.jpg
File: pic_396.jpg
File: pic_191.jpg
File: pic_236.jpg
File: pic_167.jpg
File: pic_054.jpg
File: pic_223.jpg

File: pic_006.jpg
File: pic_219.jpg
File: pic_264.jpg
File: pic_170.jpg
File: pic_124.jpg
File: pic_168.jpg
File: pic_111.jpg
File: pic_373.jpg
File: pic_012.jpg
File: pic_007.jpg
File: pic_085.jpg
File: pic_386.jpg
File: pic_207.jpg
File: pic_216.jpg
File: pic_390.jpg
File: pic_328.jpg
File: pic_106.jpg
File: pic_171.jpg
File: pic_276.jpg
File: pic_353.jpg
File: pic_225.jpg
File: pic_169.jpg
File: pic_254.jpg
File: pic_190.jpg
File: pic_321.jpg
File: pic_164.jpg
File: pic_187.jpg
File: pic_218.jpg
File: pic_337.jpg
File: pic_140.jpg
File: pic_159.jpg
File: pic_090.jpg
File: pic_152.jpg
File: pic_027.jpg
File: pic_128.jpg
File: pic_350.jpg
File: pic_257.jpg
File: pic_357.jpg
File: pic_338.jpg
File: pic_341.jpg
File: pic_364.jpg
File: pic_199.jpg
File: pic_231.jpg
File: pic_297.jpg
File: pic_099.jpg
File: pic_325.jpg
File: pic_403.jpg
File: pic_391.jpg

```
File: pic_359.jpg
File: pic_129.jpg
File: pic_195.jpg
File: pic_402.jpg
File: pic_186.jpg
File: pic_157.jpg
File: pic_204.jpg
File: pic_165.jpg
File: pic_389.jpg
File: pic_112.jpg
File: pic_068.jpg
File: pic_003.jpg
File: pic_156.jpg
File: pic_127.jpg
File: pic_352.jpg
File: pic_077.jpg
File: pic_142.jpg
File: pic_244.jpg
File: pic_256.jpg
File: pic_313.jpg
File: pic_083.jpg
File: pic_277.jpg
File: pic_117.jpg
File: pic_064.jpg
File: pic_331.jpg
File: pic_115.jpg
File: pic_285.jpg
File: pic_046.jpg
File: pic_367.jpg
File: pic_401.jpg
File: pic_322.jpg
File: pic_320.jpg
File: pic_334.jpg
File: pic_023.jpg
File: pic_269.jpg
File: pic_155.jpg
File: pic_188.jpg
File: pic_014.jpg
File: pic_073.jpg
File: pic_305.jpg
File: pic_032.jpg
File: pic_347.jpg
File: pic_087.jpg
File: pic_160.jpg
File: pic_306.jpg
File: pic_047.jpg
File: pic_135.jpg
File: pic_233.jpg
```

File: pic_270.jpg
File: pic_097.jpg
File: pic_382.jpg
File: pic_249.jpg
File: pic_251.jpg
File: pic_033.jpg
File: pic_274.jpg
File: pic_084.jpg
File: pic_392.jpg
File: pic_042.jpg
File: pic_067.jpg
File: pic_130.jpg
File: pic_133.jpg
File: pic_388.jpg
File: pic_361.jpg
File: pic_045.jpg
File: pic_125.jpg
File: pic_071.jpg
File: pic_039.jpg
File: pic_035.jpg
File: pic_049.jpg
File: pic_002.jpg
File: pic_121.jpg
File: pic_089.jpg
File: pic_351.jpg
File: pic_212.jpg
File: pic_342.jpg
File: pic_262.jpg
File: pic_005.jpg
File: pic_018.jpg
File: pic_144.jpg
File: pic_241.jpg
File: pic_319.jpg
File: pic_123.jpg
File: pic_183.jpg
File: pic_122.jpg
File: pic_292.jpg
File: pic_339.jpg
File: pic_022.jpg
File: pic_242.jpg
File: pic_034.jpg
File: pic_281.jpg
File: pic_370.jpg
File: pic_265.jpg
File: pic_300.jpg
File: pic_293.jpg
File: pic_093.jpg
File: pic_024.jpg

File: pic_139.jpg
File: pic_344.jpg
File: pic_131.jpg
File: pic_109.jpg
File: pic_103.jpg
File: pic_291.jpg
File: pic_327.jpg
File: pic_260.jpg
File: pic_138.jpg
File: pic_253.jpg
File: pic_154.jpg
File: pic_250.jpg
File: pic_295.jpg
File: pic_278.jpg
File: pic_193.jpg
File: pic_172.jpg
File: pic_287.jpg
File: pic_158.jpg
File: pic_226.jpg
File: pic_298.jpg
File: pic_345.jpg
File: pic_307.jpg
File: pic_004.jpg
File: pic_385.jpg
File: pic_161.jpg
File: pic_220.jpg
File: pic_208.jpg
File: pic_162.jpg
File: pic_395.jpg
File: pic_333.jpg
File: pic_366.jpg
File: pic_259.jpg
File: pic_050.jpg
File: pic_082.jpg
File: pic_076.jpg
File: pic_215.jpg
File: pic_279.jpg
File: pic_092.jpg
File: pic_028.jpg
File: pic_273.jpg
File: pic_041.jpg
File: pic_065.jpg
File: pic_175.jpg
File: pic_309.jpg
File: pic_200.jpg
File: pic_358.jpg
File: pic_308.jpg
File: pic_286.jpg

File: pic_141.jpg
File: pic_096.jpg
File: pic_101.jpg
File: pic_056.jpg
File: pic_246.jpg
File: pic_318.jpg
File: pic_104.jpg
File: pic_145.jpg
File: pic_114.jpg
File: pic_026.jpg
File: pic_275.jpg
File: pic_189.jpg
File: pic_163.jpg
File: pic_196.jpg
File: pic_304.jpg
File: pic_290.jpg
File: pic_019.jpg
File: pic_178.jpg
File: pic_110.jpg
File: pic_009.jpg
File: pic_151.jpg
File: pic_365.jpg
File: pic_116.jpg
File: pic_052.jpg
File: pic_029.jpg
File: pic_075.jpg
File: pic_031.jpg
File: pic_272.jpg
File: pic_283.jpg
File: pic_381.jpg
File: pic_008.jpg
File: pic_197.jpg
File: pic_294.jpg
File: pic_206.jpg
File: pic_166.jpg
File: pic_258.jpg
File: pic_237.jpg
File: pic_132.jpg
File: pic_356.jpg
File: pic_348.jpg
File: pic_078.jpg
File: pic_371.jpg
File: pic_030.jpg
File: pic_086.jpg
File: pic_070.jpg
File: pic_176.jpg
File: pic_349.jpg
File: pic_355.jpg

File: pic_235.jpg
File: pic_224.jpg
File: pic_372.jpg
File: pic_245.jpg
File: pic_137.jpg
File: pic_317.jpg
File: pic_015.jpg
File: pic_217.jpg
File: pic_227.jpg
File: pic_267.jpg
File: pic_312.jpg
File: pic_330.jpg
File: pic_143.jpg
File: pic_011.jpg
File: pic_102.jpg
File: pic_404.jpg
File: pic_013.jpg
File: pic_203.jpg
File: pic_230.jpg
File: pic_113.jpg
File: pic_202.jpg
File: pic_150.jpg
File: pic_315.jpg
File: pic_332.jpg
File: pic_134.jpg
File: pic_383.jpg
File: pic_016.jpg
File: pic_247.jpg
File: pic_185.jpg
File: pic_061.jpg
File: pic_302.jpg
File: pic_136.jpg
File: pic_059.jpg
File: pic_180.jpg
File: pic_261.jpg
File: pic_038.jpg
File: pic_010.jpg
File: pic_021.jpg
File: pic_384.jpg
File: pic_268.jpg
File: pic_091.jpg
File: pic_232.jpg
File: pic_126.jpg
File: pic_393.jpg
File: pic_363.jpg
File: pic_181.jpg
File: pic_179.jpg
File: pic_105.jpg

```
  File: pic_343.jpg
  File: pic_025.jpg
  File: pic_221.jpg
  File: pic_222.jpg
  File: pic_205.jpg
  File: pic_066.jpg
  File: pic_194.jpg
  File: pic_210.jpg
  File: pic_072.jpg
  File: pic_040.jpg
  File: pic_107.jpg
  File: pic_057.jpg
  File: pic_017.jpg
  File: pic_100.jpg
  File: pic_380.jpg
  File: pic_080.jpg
  File: pic_213.jpg
  File: pic_346.jpg
  File: pic_324.jpg
  File: pic_118.jpg
Directory: /content/dataset/food_photos/VegBurger
  File: pic_044.jpg
  File: pic_053.jpg
  File: pic_521.jpg
  File: pic_299.jpg
  File: pic_255.jpg
  File: pic_119.jpg
  File: pic_463.jpg
  File: pic_616.jpg
  File: pic_368.jpg
  File: pic_594.jpg
  File: pic_376.jpg
  File: pic_069.jpg
  File: pic_329.jpg
  File: pic_519.jpg
  File: pic_683.jpg
  File: pic_596.jpg
  File: pic_532.jpg
  File: pic_448.jpg
  File: pic_055.jpg
  File: pic_148.jpg
  File: pic_709.jpg
  File: pic_424.jpg
  File: pic_289.jpg
  File: pic_514.jpg
  File: pic_209.jpg
  File: pic_228.jpg
  File: pic_721.jpg
```

```
File: pic_149.jpg
File: pic_340.jpg
File: pic_252.jpg
File: pic_147.jpg
File: pic_399.jpg
File: pic_326.jpg
File: pic_108.jpg
File: pic_146.jpg
File: pic_610.jpg
File: pic_243.jpg
File: pic_051.jpg
File: pic_198.jpg
File: pic_601.jpg
File: pic_239.jpg
File: pic_354.jpg
File: pic_526.jpg
File: pic_426.jpg
File: pic_608.jpg
File: pic_450.jpg
File: pic_173.jpg
File: pic_301.jpg
File: pic_615.jpg
File: pic_597.jpg
File: pic_063.jpg
File: pic_573.jpg
File: pic_653.jpg
File: pic_043.jpg
File: pic_551.jpg
File: pic_378.jpg
File: pic_398.jpg
File: pic_650.jpg
File: pic_336.jpg
File: pic_572.jpg
File: pic_703.jpg
File: pic_512.jpg
File: pic_439.jpg
File: pic_248.jpg
File: pic_263.jpg
File: pic_379.jpg
File: pic_577.jpg
File: pic_020.jpg
File: pic_360.jpg
File: pic_412.jpg
File: pic_374.jpg
File: pic_369.jpg
File: pic_715.jpg
File: pic_177.jpg
File: pic_562.jpg
```

File: pic_201.jpg
File: pic_700.jpg
File: pic_058.jpg
File: pic_081.jpg
File: pic_375.jpg
File: pic_284.jpg
File: pic_088.jpg
File: pic_316.jpg
File: pic_377.jpg
File: pic_502.jpg
File: pic_611.jpg
File: pic_060.jpg
File: pic_303.jpg
File: pic_387.jpg
File: pic_310.jpg
File: pic_335.jpg
File: pic_271.jpg
File: pic_182.jpg
File: pic_238.jpg
File: pic_397.jpg
File: pic_037.jpg
File: pic_192.jpg
File: pic_635.jpg
File: pic_079.jpg
File: pic_314.jpg
File: pic_214.jpg
File: pic_592.jpg
File: pic_074.jpg
File: pic_405.jpg
File: pic_470.jpg
File: pic_446.jpg
File: pic_095.jpg
File: pic_174.jpg
File: pic_697.jpg
File: pic_484.jpg
File: pic_565.jpg
File: pic_098.jpg
File: pic_296.jpg
File: pic_547.jpg
File: pic_579.jpg
File: pic_662.jpg
File: pic_280.jpg
File: pic_211.jpg
File: pic_497.jpg
File: pic_036.jpg
File: pic_622.jpg
File: pic_583.jpg
File: pic_469.jpg

File: pic_282.jpg
File: pic_517.jpg
File: pic_266.jpg
File: pic_407.jpg
File: pic_062.jpg
File: pic_684.jpg
File: pic_240.jpg
File: pic_234.jpg
File: pic_229.jpg
File: pic_311.jpg
File: pic_120.jpg
File: pic_576.jpg
File: pic_540.jpg
File: pic_094.jpg
File: pic_184.jpg
File: pic_362.jpg
File: pic_654.jpg
File: pic_288.jpg
File: pic_396.jpg
File: pic_323.jpg
File: pic_191.jpg
File: pic_509.jpg
File: pic_236.jpg
File: pic_167.jpg
File: pic_054.jpg
File: pic_223.jpg
File: pic_006.jpg
File: pic_584.jpg
File: pic_478.jpg
File: pic_623.jpg
File: pic_219.jpg
File: pic_515.jpg
File: pic_612.jpg
File: pic_264.jpg
File: pic_701.jpg
File: pic_434.jpg
File: pic_643.jpg
File: pic_170.jpg
File: pic_660.jpg
File: pic_124.jpg
File: pic_689.jpg
File: pic_168.jpg
File: pic_111.jpg
File: pic_373.jpg
File: pic_542.jpg
File: pic_656.jpg
File: pic_012.jpg
File: pic_007.jpg

File: pic_529.jpg
File: pic_418.jpg
File: pic_559.jpg
File: pic_488.jpg
File: pic_603.jpg
File: pic_085.jpg
File: pic_386.jpg
File: pic_207.jpg
File: pic_588.jpg
File: pic_706.jpg
File: pic_216.jpg
File: pic_637.jpg
File: pic_390.jpg
File: pic_328.jpg
File: pic_106.jpg
File: pic_171.jpg
File: pic_648.jpg
File: pic_411.jpg
File: pic_550.jpg
File: pic_276.jpg
File: pic_413.jpg
File: pic_353.jpg
File: pic_225.jpg
File: pic_169.jpg
File: pic_422.jpg
File: pic_254.jpg
File: pic_190.jpg
File: pic_321.jpg
File: pic_164.jpg
File: pic_187.jpg
File: pic_218.jpg
File: pic_337.jpg
File: pic_140.jpg
File: pic_159.jpg
File: pic_704.jpg
File: pic_090.jpg
File: pic_705.jpg
File: pic_152.jpg
File: pic_027.jpg
File: pic_631.jpg
File: pic_128.jpg
File: pic_708.jpg
File: pic_630.jpg
File: pic_350.jpg
File: pic_257.jpg
File: pic_357.jpg
File: pic_338.jpg
File: pic_621.jpg

File: pic_341.jpg
File: pic_364.jpg
File: pic_604.jpg
File: pic_658.jpg
File: pic_199.jpg
File: pic_231.jpg
File: pic_693.jpg
File: pic_297.jpg
File: pic_099.jpg
File: pic_523.jpg
File: pic_325.jpg
File: pic_403.jpg
File: pic_480.jpg
File: pic_585.jpg
File: pic_391.jpg
File: pic_359.jpg
File: pic_664.jpg
File: pic_537.jpg
File: pic_129.jpg
File: pic_195.jpg
File: pic_402.jpg
File: pic_644.jpg
File: pic_186.jpg
File: pic_157.jpg
File: pic_204.jpg
File: pic_590.jpg
File: pic_165.jpg
File: pic_389.jpg
File: pic_112.jpg
File: pic_068.jpg
File: pic_575.jpg
File: pic_711.jpg
File: pic_473.jpg
File: pic_003.jpg
File: pic_156.jpg
File: pic_127.jpg
File: pic_352.jpg
File: pic_496.jpg
File: pic_077.jpg
File: pic_142.jpg
File: pic_617.jpg
File: pic_437.jpg
File: pic_244.jpg
File: pic_256.jpg
File: pic_313.jpg
File: pic_394.jpg
File: pic_083.jpg
File: pic_277.jpg

File: pic_629.jpg
File: pic_117.jpg
File: pic_064.jpg
File: pic_331.jpg
File: pic_115.jpg
File: pic_285.jpg
File: pic_046.jpg
File: pic_530.jpg
File: pic_367.jpg
File: pic_417.jpg
File: pic_401.jpg
File: pic_322.jpg
File: pic_441.jpg
File: pic_320.jpg
File: pic_334.jpg
File: pic_485.jpg
File: pic_023.jpg
File: pic_651.jpg
File: pic_269.jpg
File: pic_155.jpg
File: pic_493.jpg
File: pic_188.jpg
File: pic_014.jpg
File: pic_073.jpg
File: pic_305.jpg
File: pic_425.jpg
File: pic_460.jpg
File: pic_663.jpg
File: pic_032.jpg
File: pic_582.jpg
File: pic_714.jpg
File: pic_492.jpg
File: pic_347.jpg
File: pic_087.jpg
File: pic_160.jpg
File: pic_455.jpg
File: pic_543.jpg
File: pic_306.jpg
File: pic_047.jpg
File: pic_135.jpg
File: pic_233.jpg
File: pic_614.jpg
File: pic_472.jpg
File: pic_420.jpg
File: pic_270.jpg
File: pic_504.jpg
File: pic_097.jpg
File: pic_382.jpg

File: pic_249.jpg
File: pic_251.jpg
File: pic_464.jpg
File: pic_682.jpg
File: pic_033.jpg
File: pic_676.jpg
File: pic_274.jpg
File: pic_546.jpg
File: pic_695.jpg
File: pic_429.jpg
File: pic_084.jpg
File: pic_458.jpg
File: pic_392.jpg
File: pic_494.jpg
File: pic_525.jpg
File: pic_042.jpg
File: pic_678.jpg
File: pic_558.jpg
File: pic_449.jpg
File: pic_609.jpg
File: pic_634.jpg
File: pic_712.jpg
File: pic_067.jpg
File: pic_513.jpg
File: pic_130.jpg
File: pic_133.jpg
File: pic_680.jpg
File: pic_518.jpg
File: pic_388.jpg
File: pic_657.jpg
File: pic_491.jpg
File: pic_692.jpg
File: pic_361.jpg
File: pic_580.jpg
File: pic_045.jpg
File: pic_468.jpg
File: pic_125.jpg
File: pic_071.jpg
File: pic_039.jpg
File: pic_035.jpg
File: pic_049.jpg
File: pic_476.jpg
File: pic_002.jpg
File: pic_430.jpg
File: pic_431.jpg
File: pic_666.jpg
File: pic_121.jpg
File: pic_089.jpg

```
File: pic_696.jpg
File: pic_351.jpg
File: pic_212.jpg
File: pic_538.jpg
File: pic_694.jpg
File: pic_506.jpg
File: pic_342.jpg
File: pic_566.jpg
File: pic_641.jpg
File: pic_262.jpg
File: pic_461.jpg
File: pic_483.jpg
File: pic_005.jpg
File: pic_018.jpg
File: pic_144.jpg
File: pic_241.jpg
File: pic_319.jpg
File: pic_123.jpg
File: pic_183.jpg
File: pic_122.jpg
File: pic_628.jpg
File: pic_447.jpg
File: pic_655.jpg
File: pic_687.jpg
File: pic_292.jpg
File: pic_495.jpg
File: pic_636.jpg
File: pic_339.jpg
File: pic_539.jpg
File: pic_022.jpg
File: pic_555.jpg
File: pic_242.jpg
File: pic_440.jpg
File: pic_034.jpg
File: pic_281.jpg
File: pic_556.jpg
File: pic_370.jpg
File: pic_265.jpg
File: pic_593.jpg
File: pic_300.jpg
File: pic_293.jpg
File: pic_093.jpg
File: pic_408.jpg
File: pic_024.jpg
File: pic_501.jpg
File: pic_139.jpg
File: pic_344.jpg
File: pic_131.jpg
```

File: pic_406.jpg
File: pic_467.jpg
File: pic_109.jpg
File: pic_103.jpg
File: pic_457.jpg
File: pic_291.jpg
File: pic_327.jpg
File: pic_260.jpg
File: pic_138.jpg
File: pic_253.jpg
File: pic_686.jpg
File: pic_699.jpg
File: pic_154.jpg
File: pic_250.jpg
File: pic_295.jpg
File: pic_278.jpg
File: pic_606.jpg
File: pic_554.jpg
File: pic_193.jpg
File: pic_533.jpg
File: pic_172.jpg
File: pic_287.jpg
File: pic_716.jpg
File: pic_158.jpg
File: pic_226.jpg
File: pic_298.jpg
File: pic_552.jpg
File: pic_345.jpg
File: pic_691.jpg
File: pic_534.jpg
File: pic_307.jpg
File: pic_004.jpg
File: pic_466.jpg
File: pic_595.jpg
File: pic_385.jpg
File: pic_161.jpg
File: pic_220.jpg
File: pic_208.jpg
File: pic_510.jpg
File: pic_162.jpg
File: pic_395.jpg
File: pic_333.jpg
File: pic_366.jpg
File: pic_259.jpg
File: pic_050.jpg
File: pic_082.jpg
File: pic_076.jpg
File: pic_215.jpg

File: pic_456.jpg
File: pic_673.jpg
File: pic_279.jpg
File: pic_092.jpg
File: pic_427.jpg
File: pic_028.jpg
File: pic_273.jpg
File: pic_459.jpg
File: pic_561.jpg
File: pic_477.jpg
File: pic_560.jpg
File: pic_416.jpg
File: pic_633.jpg
File: pic_414.jpg
File: pic_490.jpg
File: pic_041.jpg
File: pic_065.jpg
File: pic_698.jpg
File: pic_175.jpg
File: pic_507.jpg
File: pic_309.jpg
File: pic_571.jpg
File: pic_415.jpg
File: pic_605.jpg
File: pic_200.jpg
File: pic_358.jpg
File: pic_308.jpg
File: pic_581.jpg
File: pic_286.jpg
File: pic_141.jpg
File: pic_096.jpg
File: pic_101.jpg
File: pic_056.jpg
File: pic_720.jpg
File: pic_438.jpg
File: pic_479.jpg
File: pic_246.jpg
File: pic_481.jpg
File: pic_318.jpg
File: pic_443.jpg
File: pic_444.jpg
File: pic_505.jpg
File: pic_104.jpg
File: pic_145.jpg
File: pic_668.jpg
File: pic_114.jpg
File: pic_026.jpg
File: pic_451.jpg

File: pic_275.jpg
File: pic_462.jpg
File: pic_574.jpg
File: pic_516.jpg
File: pic_189.jpg
File: pic_482.jpg
File: pic_163.jpg
File: pic_613.jpg
File: pic_549.jpg
File: pic_536.jpg
File: pic_528.jpg
File: pic_196.jpg
File: pic_649.jpg
File: pic_304.jpg
File: pic_290.jpg
File: pic_019.jpg
File: pic_178.jpg
File: pic_110.jpg
File: pic_647.jpg
File: pic_639.jpg
File: pic_009.jpg
File: pic_710.jpg
File: pic_151.jpg
File: pic_365.jpg
File: pic_116.jpg
File: pic_587.jpg
File: pic_052.jpg
File: pic_624.jpg
File: pic_674.jpg
File: pic_029.jpg
File: pic_646.jpg
File: pic_075.jpg
File: pic_031.jpg
File: pic_272.jpg
File: pic_283.jpg
File: pic_381.jpg
File: pic_508.jpg
File: pic_454.jpg
File: pic_008.jpg
File: pic_197.jpg
File: pic_294.jpg
File: pic_206.jpg
File: pic_487.jpg
File: pic_166.jpg
File: pic_258.jpg
File: pic_237.jpg
File: pic_132.jpg
File: pic_718.jpg

File: pic_598.jpg
File: pic_356.jpg
File: pic_435.jpg
File: pic_632.jpg
File: pic_685.jpg
File: pic_607.jpg
File: pic_348.jpg
File: pic_557.jpg
File: pic_445.jpg
File: pic_471.jpg
File: pic_078.jpg
File: pic_371.jpg
File: pic_030.jpg
File: pic_475.jpg
File: pic_086.jpg
File: pic_671.jpg
File: pic_070.jpg
File: pic_176.jpg
File: pic_349.jpg
File: pic_578.jpg
File: pic_355.jpg
File: pic_235.jpg
File: pic_224.jpg
File: pic_659.jpg
File: pic_372.jpg
File: pic_245.jpg
File: pic_432.jpg
File: pic_545.jpg
File: pic_137.jpg
File: pic_627.jpg
File: pic_317.jpg
File: pic_015.jpg
File: pic_541.jpg
File: pic_217.jpg
File: pic_620.jpg
File: pic_498.jpg
File: pic_227.jpg
File: pic_267.jpg
File: pic_312.jpg
File: pic_153.jpg
File: pic_669.jpg
File: pic_048.jpg
File: pic_707.jpg
File: pic_330.jpg
File: pic_702.jpg
File: pic_626.jpg
File: pic_589.jpg
File: pic_465.jpg

```
File: pic_499.jpg
File: pic_618.jpg
File: pic_665.jpg
File: pic_563.jpg
File: pic_143.jpg
File: pic_011.jpg
File: pic_102.jpg
File: pic_675.jpg
File: pic_522.jpg
File: pic_423.jpg
File: pic_404.jpg
File: pic_013.jpg
File: pic_203.jpg
File: pic_535.jpg
File: pic_230.jpg
File: pic_113.jpg
File: pic_202.jpg
File: pic_524.jpg
File: pic_150.jpg
File: pic_315.jpg
File: pic_332.jpg
File: pic_681.jpg
File: pic_486.jpg
File: pic_419.jpg
File: pic_511.jpg
File: pic_489.jpg
File: pic_688.jpg
File: pic_134.jpg
File: pic_383.jpg
File: pic_600.jpg
File: pic_602.jpg
File: pic_016.jpg
File: pic_247.jpg
File: pic_591.jpg
File: pic_672.jpg
File: pic_185.jpg
File: pic_061.jpg
File: pic_717.jpg
File: pic_567.jpg
File: pic_302.jpg
File: pic_410.jpg
File: pic_503.jpg
File: pic_136.jpg
File: pic_453.jpg
File: pic_059.jpg
File: pic_527.jpg
File: pic_569.jpg
File: pic_520.jpg
```

File: pic_452.jpg
File: pic_474.jpg
File: pic_180.jpg
File: pic_261.jpg
File: pic_038.jpg
File: pic_500.jpg
File: pic_677.jpg
File: pic_010.jpg
File: pic_021.jpg
File: pic_384.jpg
File: pic_268.jpg
File: pic_652.jpg
File: pic_642.jpg
File: pic_661.jpg
File: pic_690.jpg
File: pic_091.jpg
File: pic_645.jpg
File: pic_436.jpg
File: pic_232.jpg
File: pic_548.jpg
File: pic_667.jpg
File: pic_126.jpg
File: pic_670.jpg
File: pic_393.jpg
File: pic_442.jpg
File: pic_363.jpg
File: pic_599.jpg
File: pic_553.jpg
File: pic_570.jpg
File: pic_713.jpg
File: pic_544.jpg
File: pic_181.jpg
File: pic_400.jpg
File: pic_679.jpg
File: pic_179.jpg
File: pic_433.jpg
File: pic_105.jpg
File: pic_343.jpg
File: pic_625.jpg
File: pic_025.jpg
File: pic_221.jpg
File: pic_531.jpg
File: pic_222.jpg
File: pic_568.jpg
File: pic_205.jpg
File: pic_564.jpg
File: pic_409.jpg
File: pic_066.jpg

```
  File: pic_428.jpg
  File: pic_194.jpg
  File: pic_210.jpg
  File: pic_072.jpg
  File: pic_040.jpg
  File: pic_107.jpg
  File: pic_057.jpg
  File: pic_421.jpg
  File: pic_017.jpg
  File: pic_100.jpg
  File: pic_380.jpg
  File: pic_640.jpg
  File: pic_080.jpg
  File: pic_213.jpg
  File: pic_346.jpg
  File: pic_619.jpg
  File: pic_638.jpg
  File: pic_324.jpg
  File: pic_586.jpg
  File: pic_118.jpg
Directory: /content/dataset/food_photos/FrenchFries
  File: pic_044.jpg
  File: pic_053.jpg
  File: pic_521.jpg
  File: pic_299.jpg
  File: pic_255.jpg
  File: pic_119.jpg
  File: pic_463.jpg
  File: pic_368.jpg
  File: pic_594.jpg
  File: pic_376.jpg
  File: pic_069.jpg
  File: pic_329.jpg
  File: pic_519.jpg
  File: pic_596.jpg
  File: pic_532.jpg
  File: pic_448.jpg
  File: pic_055.jpg
  File: pic_148.jpg
  File: pic_424.jpg
  File: pic_289.jpg
  File: pic_514.jpg
  File: pic_209.jpg
  File: pic_228.jpg
  File: pic_149.jpg
  File: pic_340.jpg
  File: pic_252.jpg
  File: pic_147.jpg
```

File: pic_399.jpg
File: pic_326.jpg
File: pic_108.jpg
File: pic_146.jpg
File: pic_243.jpg
File: pic_051.jpg
File: pic_198.jpg
File: pic_601.jpg
File: pic_239.jpg
File: pic_354.jpg
File: pic_526.jpg
File: pic_426.jpg
File: pic_450.jpg
File: pic_173.jpg
File: pic_301.jpg
File: pic_597.jpg
File: pic_063.jpg
File: pic_573.jpg
File: pic_043.jpg
File: pic_551.jpg
File: pic_378.jpg
File: pic_398.jpg
File: pic_336.jpg
File: pic_572.jpg
File: pic_512.jpg
File: pic_439.jpg
File: pic_248.jpg
File: pic_263.jpg
File: pic_379.jpg
File: pic_577.jpg
File: pic_020.jpg
File: pic_360.jpg
File: pic_412.jpg
File: pic_374.jpg
File: pic_369.jpg
File: pic_177.jpg
File: pic_562.jpg
File: pic_201.jpg
File: pic_058.jpg
File: pic_081.jpg
File: pic_375.jpg
File: pic_284.jpg
File: pic_088.jpg
File: pic_316.jpg
File: pic_377.jpg
File: pic_502.jpg
File: pic_060.jpg
File: pic_303.jpg

File: pic_387.jpg
File: pic_310.jpg
File: pic_335.jpg
File: pic_271.jpg
File: pic_182.jpg
File: pic_238.jpg
File: pic_397.jpg
File: pic_037.jpg
File: pic_192.jpg
File: pic_079.jpg
File: pic_314.jpg
File: pic_214.jpg
File: pic_592.jpg
File: pic_074.jpg
File: pic_405.jpg
File: pic_470.jpg
File: pic_446.jpg
File: pic_095.jpg
File: pic_174.jpg
File: pic_484.jpg
File: pic_565.jpg
File: pic_098.jpg
File: pic_296.jpg
File: pic_547.jpg
File: pic_579.jpg
File: pic_280.jpg
File: pic_211.jpg
File: pic_497.jpg
File: pic_036.jpg
File: pic_583.jpg
File: pic_469.jpg
File: pic_282.jpg
File: pic_517.jpg
File: pic_266.jpg
File: pic_407.jpg
File: pic_062.jpg
File: pic_240.jpg
File: pic_234.jpg
File: pic_229.jpg
File: pic_311.jpg
File: pic_120.jpg
File: pic_576.jpg
File: pic_540.jpg
File: pic_094.jpg
File: pic_184.jpg
File: pic_362.jpg
File: pic_288.jpg
File: pic_396.jpg

File: pic_323.jpg
File: pic_191.jpg
File: pic_509.jpg
File: pic_236.jpg
File: pic_167.jpg
File: pic_054.jpg
File: pic_223.jpg
File: pic_006.jpg
File: pic_584.jpg
File: pic_478.jpg
File: pic_219.jpg
File: pic_515.jpg
File: pic_264.jpg
File: pic_434.jpg
File: pic_170.jpg
File: pic_124.jpg
File: pic_168.jpg
File: pic_111.jpg
File: pic_373.jpg
File: pic_542.jpg
File: pic_012.jpg
File: pic_007.jpg
File: pic_529.jpg
File: pic_418.jpg
File: pic_559.jpg
File: pic_488.jpg
File: pic_085.jpg
File: pic_386.jpg
File: pic_207.jpg
File: pic_588.jpg
File: pic_216.jpg
File: pic_390.jpg
File: pic_328.jpg
File: pic_106.jpg
File: pic_171.jpg
File: pic_411.jpg
File: pic_550.jpg
File: pic_276.jpg
File: pic_413.jpg
File: pic_353.jpg
File: pic_225.jpg
File: pic_169.jpg
File: pic_422.jpg
File: pic_254.jpg
File: pic_190.jpg
File: pic_321.jpg
File: pic_164.jpg
File: pic_187.jpg

File: pic_218.jpg
File: pic_337.jpg
File: pic_140.jpg
File: pic_159.jpg
File: pic_090.jpg
File: pic_152.jpg
File: pic_027.jpg
File: pic_128.jpg
File: pic_350.jpg
File: pic_257.jpg
File: pic_357.jpg
File: pic_338.jpg
File: pic_341.jpg
File: pic_364.jpg
File: pic_199.jpg
File: pic_231.jpg
File: pic_297.jpg
File: pic_099.jpg
File: pic_523.jpg
File: pic_325.jpg
File: pic_403.jpg
File: pic_480.jpg
File: pic_585.jpg
File: pic_391.jpg
File: pic_359.jpg
File: pic_537.jpg
File: pic_129.jpg
File: pic_195.jpg
File: pic_402.jpg
File: pic_186.jpg
File: pic_157.jpg
File: pic_204.jpg
File: pic_165.jpg
File: pic_389.jpg
File: pic_112.jpg
File: pic_068.jpg
File: pic_575.jpg
File: pic_473.jpg
File: pic_003.jpg
File: pic_156.jpg
File: pic_127.jpg
File: pic_352.jpg
File: pic_496.jpg
File: pic_077.jpg
File: pic_142.jpg
File: pic_437.jpg
File: pic_244.jpg
File: pic_256.jpg

```
File: pic_313.jpg
File: pic_394.jpg
File: pic_083.jpg
File: pic_277.jpg
File: pic_117.jpg
File: pic_064.jpg
File: pic_331.jpg
File: pic_115.jpg
File: pic_285.jpg
File: pic_046.jpg
File: pic_530.jpg
File: pic_367.jpg
File: pic_417.jpg
File: pic_401.jpg
File: pic_322.jpg
File: pic_441.jpg
File: pic_320.jpg
File: pic_485.jpg
File: pic_023.jpg
File: pic_269.jpg
File: pic_155.jpg
File: pic_493.jpg
File: pic_188.jpg
File: pic_014.jpg
File: pic_073.jpg
File: pic_305.jpg
File: pic_425.jpg
File: pic_460.jpg
File: pic_032.jpg
File: pic_582.jpg
File: pic_492.jpg
File: pic_347.jpg
File: pic_087.jpg
File: pic_160.jpg
File: pic_455.jpg
File: pic_543.jpg
File: pic_306.jpg
File: pic_047.jpg
File: pic_135.jpg
File: pic_233.jpg
File: pic_472.jpg
File: pic_420.jpg
File: pic_270.jpg
File: pic_504.jpg
File: pic_097.jpg
File: pic_382.jpg
File: pic_249.jpg
File: pic_251.jpg
```

```
File: pic_464.jpg
File: pic_033.jpg
File: pic_274.jpg
File: pic_546.jpg
File: pic_429.jpg
File: pic_084.jpg
File: pic_458.jpg
File: pic_392.jpg
File: pic_494.jpg
File: pic_525.jpg
File: pic_042.jpg
File: pic_558.jpg
File: pic_449.jpg
File: pic_067.jpg
File: pic_513.jpg
File: pic_130.jpg
File: pic_133.jpg
File: pic_518.jpg
File: pic_388.jpg
File: pic_491.jpg
File: pic_361.jpg
File: pic_580.jpg
File: pic_045.jpg
File: pic_468.jpg
File: pic_125.jpg
File: pic_071.jpg
File: pic_039.jpg
File: pic_035.jpg
File: pic_049.jpg
File: pic_476.jpg
File: pic_002.jpg
File: pic_430.jpg
File: pic_431.jpg
File: pic_121.jpg
File: pic_089.jpg
File: pic_351.jpg
File: pic_212.jpg
File: pic_538.jpg
File: pic_506.jpg
File: pic_342.jpg
File: pic_566.jpg
File: pic_262.jpg
File: pic_461.jpg
File: pic_483.jpg
File: pic_005.jpg
File: pic_018.jpg
File: pic_144.jpg
File: pic_241.jpg
```

File: pic_319.jpg
File: pic_123.jpg
File: pic_183.jpg
File: pic_122.jpg
File: pic_447.jpg
File: pic_292.jpg
File: pic_495.jpg
File: pic_339.jpg
File: pic_539.jpg
File: pic_022.jpg
File: pic_555.jpg
File: pic_242.jpg
File: pic_440.jpg
File: pic_034.jpg
File: pic_281.jpg
File: pic_556.jpg
File: pic_370.jpg
File: pic_265.jpg
File: pic_593.jpg
File: pic_300.jpg
File: pic_293.jpg
File: pic_093.jpg
File: pic_408.jpg
File: pic_024.jpg
File: pic_501.jpg
File: pic_139.jpg
File: pic_344.jpg
File: pic_131.jpg
File: pic_406.jpg
File: pic_467.jpg
File: pic_109.jpg
File: pic_103.jpg
File: pic_457.jpg
File: pic_291.jpg
File: pic_327.jpg
File: pic_260.jpg
File: pic_138.jpg
File: pic_253.jpg
File: pic_154.jpg
File: pic_250.jpg
File: pic_295.jpg
File: pic_278.jpg
File: pic_554.jpg
File: pic_193.jpg
File: pic_533.jpg
File: pic_172.jpg
File: pic_287.jpg
File: pic_158.jpg

File: pic_226.jpg
File: pic_298.jpg
File: pic_552.jpg
File: pic_345.jpg
File: pic_534.jpg
File: pic_307.jpg
File: pic_004.jpg
File: pic_466.jpg
File: pic_595.jpg
File: pic_385.jpg
File: pic_161.jpg
File: pic_220.jpg
File: pic_208.jpg
File: pic_510.jpg
File: pic_162.jpg
File: pic_395.jpg
File: pic_333.jpg
File: pic_366.jpg
File: pic_259.jpg
File: pic_050.jpg
File: pic_082.jpg
File: pic_076.jpg
File: pic_215.jpg
File: pic_456.jpg
File: pic_279.jpg
File: pic_092.jpg
File: pic_427.jpg
File: pic_028.jpg
File: pic_273.jpg
File: pic_459.jpg
File: pic_561.jpg
File: pic_477.jpg
File: pic_560.jpg
File: pic_416.jpg
File: pic_414.jpg
File: pic_490.jpg
File: pic_041.jpg
File: pic_065.jpg
File: pic_175.jpg
File: pic_507.jpg
File: pic_309.jpg
File: pic_571.jpg
File: pic_415.jpg
File: pic_200.jpg
File: pic_358.jpg
File: pic_308.jpg
File: pic_581.jpg
File: pic_286.jpg

File: pic_141.jpg
File: pic_096.jpg
File: pic_101.jpg
File: pic_056.jpg
File: pic_438.jpg
File: pic_479.jpg
File: pic_246.jpg
File: pic_481.jpg
File: pic_318.jpg
File: pic_443.jpg
File: pic_444.jpg
File: pic_505.jpg
File: pic_104.jpg
File: pic_145.jpg
File: pic_114.jpg
File: pic_026.jpg
File: pic_451.jpg
File: pic_275.jpg
File: pic_462.jpg
File: pic_574.jpg
File: pic_516.jpg
File: pic_189.jpg
File: pic_482.jpg
File: pic_163.jpg
File: pic_549.jpg
File: pic_536.jpg
File: pic_528.jpg
File: pic_196.jpg
File: pic_304.jpg
File: pic_290.jpg
File: pic_019.jpg
File: pic_178.jpg
File: pic_110.jpg
File: pic_009.jpg
File: pic_151.jpg
File: pic_365.jpg
File: pic_116.jpg
File: pic_587.jpg
File: pic_052.jpg
File: pic_029.jpg
File: pic_075.jpg
File: pic_031.jpg
File: pic_272.jpg
File: pic_283.jpg
File: pic_381.jpg
File: pic_508.jpg
File: pic_454.jpg
File: pic_008.jpg

File: pic_197.jpg
File: pic_294.jpg
File: pic_206.jpg
File: pic_487.jpg
File: pic_166.jpg
File: pic_258.jpg
File: pic_237.jpg
File: pic_132.jpg
File: pic_598.jpg
File: pic_356.jpg
File: pic_435.jpg
File: pic_348.jpg
File: pic_557.jpg
File: pic_445.jpg
File: pic_471.jpg
File: pic_078.jpg
File: pic_371.jpg
File: pic_030.jpg
File: pic_475.jpg
File: pic_086.jpg
File: pic_070.jpg
File: pic_176.jpg
File: pic_349.jpg
File: pic_578.jpg
File: pic_355.jpg
File: pic_235.jpg
File: pic_224.jpg
File: pic_372.jpg
File: pic_245.jpg
File: pic_432.jpg
File: pic_545.jpg
File: pic_137.jpg
File: pic_317.jpg
File: pic_015.jpg
File: pic_541.jpg
File: pic_217.jpg
File: pic_498.jpg
File: pic_227.jpg
File: pic_267.jpg
File: pic_312.jpg
File: pic_153.jpg
File: pic_048.jpg
File: pic_330.jpg
File: pic_589.jpg
File: pic_465.jpg
File: pic_499.jpg
File: pic_563.jpg
File: pic_143.jpg

File: pic_011.jpg
File: pic_102.jpg
File: pic_522.jpg
File: pic_423.jpg
File: pic_404.jpg
File: pic_013.jpg
File: pic_203.jpg
File: pic_535.jpg
File: pic_230.jpg
File: pic_113.jpg
File: pic_202.jpg
File: pic_524.jpg
File: pic_150.jpg
File: pic_315.jpg
File: pic_332.jpg
File: pic_486.jpg
File: pic_419.jpg
File: pic_511.jpg
File: pic_489.jpg
File: pic_134.jpg
File: pic_383.jpg
File: pic_600.jpg
File: pic_016.jpg
File: pic_247.jpg
File: pic_591.jpg
File: pic_185.jpg
File: pic_061.jpg
File: pic_567.jpg
File: pic_302.jpg
File: pic_410.jpg
File: pic_503.jpg
File: pic_136.jpg
File: pic_453.jpg
File: pic_059.jpg
File: pic_527.jpg
File: pic_569.jpg
File: pic_520.jpg
File: pic_452.jpg
File: pic_474.jpg
File: pic_180.jpg
File: pic_261.jpg
File: pic_038.jpg
File: pic_500.jpg
File: pic_010.jpg
File: pic_021.jpg
File: pic_384.jpg
File: pic_268.jpg
File: pic_091.jpg

```
  File: pic_436.jpg
  File: pic_232.jpg
  File: pic_548.jpg
  File: pic_126.jpg
  File: pic_393.jpg
  File: pic_442.jpg
  File: pic_363.jpg
  File: pic_599.jpg
  File: pic_553.jpg
  File: pic_570.jpg
  File: pic_544.jpg
  File: pic_181.jpg
  File: pic_400.jpg
  File: pic_179.jpg
  File: pic_433.jpg
  File: pic_105.jpg
  File: pic_343.jpg
  File: pic_025.jpg
  File: pic_221.jpg
  File: pic_531.jpg
  File: pic_222.jpg
  File: pic_568.jpg
  File: pic_205.jpg
  File: pic_564.jpg
  File: pic_409.jpg
  File: pic_066.jpg
  File: pic_428.jpg
  File: pic_194.jpg
  File: pic_210.jpg
  File: pic_072.jpg
  File: pic_040.jpg
  File: pic_107.jpg
  File: pic_057.jpg
  File: pic_421.jpg
  File: pic_017.jpg
  File: pic_100.jpg
  File: pic_380.jpg
  File: pic_080.jpg
  File: pic_213.jpg
  File: pic_346.jpg
  File: pic_324.jpg
  File: pic_586.jpg
  File: pic_118.jpg
Directory: /content/dataset/test_photos
Directory: /content/dataset/test_photos/Pizza
  File: pizza1.jpeg
  File: notpizza_tea1.jpeg
Directory: /content/dataset/test_photos/VegBurger
```

```
  File: burger1.jpeg
  File: notburger_cake1.jpeg
Directory: /content/dataset/test_photos/FrenchFries
  File: ffries1.jpeg
  File: notffries_noodles1.jpeg
DatasetDict({
    test: Dataset({
        features: ['image', 'label'],
        num_rows: 6
    })
})
```

[ ]:
```
print(dataset.keys())  # Print the available keys in the dataset
print(dataset["test"].column_names)
```

```
dict_keys(['test'])
['image', 'label']
```

[ ]:
```
import tarfile
import os
from datasets import load_dataset, DatasetDict  # Import DatasetDict

# ... (Rest of your code) ...

# Load the dataset with detected splits or a single split
split_names = [x for x in os.listdir(extract_dir) if os.path.isdir(os.path.
 ↪join(extract_dir, x)) and x in ['train', 'test', 'validation']]

if split_names:
    dataset = load_dataset("imagefolder", data_dir=extract_dir,␣
 ↪split=split_names)
else:
    # If no standard split directories are found, assume a single split and␣
 ↪create train/test/validation
    print("Warning: No standard splits found. Creating train/test/validation␣
 ↪splits.")
    dataset = load_dataset("imagefolder", data_dir=extract_dir)
    # If dataset is a DatasetDict (unlikely in this case, but for safety)
    if isinstance(dataset, DatasetDict):  # Use the imported DatasetDict
        # Assume the first key is the unnamed split
        first_key = next(iter(dataset))
        dataset = dataset[first_key].train_test_split(test_size=0.2)  # Create␣
 ↪a train/test split
    else:
        # Assume dataset is a single Dataset object
        dataset = dataset.train_test_split(test_size=0.2)  # Create a train/
 ↪test split
```

```
    # Rename the default split to 'train'
    # Rename the 'image' column to 'image_file' in both train and test splits
    dataset = DatasetDict({  # Use the imported DatasetDict
        'train': dataset['train'].rename_column("image", "image_file"),
        'test': dataset['test'].rename_column("image", "image_file")
    })
    # Create a validation split from the training set
    dataset['validation'] = dataset['train'].
 ↪select(range(int(len(dataset['train']) * 0.2)))
    # Remove the validation data from the training set to avoid overlap
    dataset['train'] = dataset['train'].select(range(int(len(dataset['train'])␣
 ↪* 0.2), len(dataset['train'])))

print(dataset)
```

```
Warning: No standard splits found. Creating train/test/validation splits.
DatasetDict({
    train: Dataset({
        features: ['image_file', 'label'],
        num_rows: 4
    })
    test: Dataset({
        features: ['image_file', 'label'],
        num_rows: 2
    })
    validation: Dataset({
        features: ['image_file', 'label'],
        num_rows: 0
    })
})
```

[ ]:
```
from datasets import DatasetDict, Dataset

# Sample data for demonstration (replace with your actual data)
train_data = {
    "image": ["path/to/image1.jpg", "path/to/image2.jpg", "path/to/image3.
 ↪jpg"],  # Example image paths
    "label": [0, 1, 0]  # Example labels
}

validation_data = {
    "image": ["path/to/image4.jpg", "path/to/image5.jpg"],
    "label": [1, 0]
}

# Create DatasetDict with actual data
dataset = DatasetDict({
```

```
    "train": Dataset.from_dict(train_data),
    "validation": Dataset.from_dict(validation_data)
})
```

[ ]:
```
from datasets import DatasetDict

# Check if dataset has the correct splits
if 'train' not in dataset:
    # If no train split, create one manually
    dataset = dataset.train_test_split(test_size=0.2)  # Split the dataset into␣
 ↪train and test

    # Assign the train and test splits
    dataset = DatasetDict({
        'train': dataset['train'],
        'test': dataset['test'],
    })

    # If needed, create validation from the train data
    dataset['validation'] = dataset['train'].
 ↪select(range(int(len(dataset['train']) * 0.2)))
    dataset['train'] = dataset['train'].select(range(int(len(dataset['train'])␣
 ↪* 0.2), len(dataset['train'])))
```

[ ]:
```
from PIL import Image
from torchvision import transforms

# Define preprocessing for input images
preprocess = transforms.Compose([
    transforms.Resize((224, 224)),  # Resize image to 224x224 as required by ViT
    transforms.ToTensor(),          # Convert image to PyTorch tensor
    transforms.Normalize(
        mean=[0.485, 0.456, 0.406],
        std=[0.229, 0.224, 0.225]
    ),  # Normalize with ViT ImageNet mean and std
])

def preprocess_image(image_path):
    image = Image.open(image_path).convert("RGB")  # Ensure 3 channels
    return preprocess(image).unsqueeze(0)  # Add batch dimension
```

[ ]:
```
import pandas as pd

# Load calorie dataset
calorie_df = pd.read_csv("/content/calorie_dataset.csv")
calorie_mapping = calorie_df.set_index("Food")["Calories"].to_dict()  # Map␣
 ↪food to calories
```

```
# Example: {'Pizza': 285, 'Burger': 354, 'Salad': 152, ...}
```

[ ]:
```
# Define the directory to save the fine-tuned model
model_save_path = "./vit-food-recognition"

# Save the trained model
trainer.save_model(model_save_path)
```

[ ]:
```
import os
print(os.listdir(model_save_path))
```

```
['training_args.bin', 'model.safetensors', 'config.json']
```

[ ]:
```
from transformers import ViTForImageClassification

# Path to the directory containing the saved model
model_path = "./vit-food-recognition"

# Load the model
model = ViTForImageClassification.from_pretrained(model_path)
model.eval()
```

[ ]:
```
ViTForImageClassification(
  (vit): ViTModel(
    (embeddings): ViTEmbeddings(
      (patch_embeddings): ViTPatchEmbeddings(
        (projection): Conv2d(3, 768, kernel_size=(16, 16), stride=(16, 16))
      )
      (dropout): Dropout(p=0.0, inplace=False)
    )
    (encoder): ViTEncoder(
      (layer): ModuleList(
        (0-11): 12 x ViTLayer(
          (attention): ViTSdpaAttention(
            (attention): ViTSdpaSelfAttention(
              (query): Linear(in_features=768, out_features=768, bias=True)
              (key): Linear(in_features=768, out_features=768, bias=True)
              (value): Linear(in_features=768, out_features=768, bias=True)
              (dropout): Dropout(p=0.0, inplace=False)
            )
            (output): ViTSelfOutput(
              (dense): Linear(in_features=768, out_features=768, bias=True)
              (dropout): Dropout(p=0.0, inplace=False)
            )
          )
          (intermediate): ViTIntermediate(
```

```
            (dense): Linear(in_features=768, out_features=3072, bias=True)
            (intermediate_act_fn): GELUActivation()
          )
          (output): ViTOutput(
            (dense): Linear(in_features=3072, out_features=768, bias=True)
            (dropout): Dropout(p=0.0, inplace=False)
          )
          (layernorm_before): LayerNorm((768,), eps=1e-12,
elementwise_affine=True)
          (layernorm_after): LayerNorm((768,), eps=1e-12,
elementwise_affine=True)
        )
      )
    )
    (layernorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
  )
  (classifier): Linear(in_features=768, out_features=2, bias=True)
)
```

[ ]:
```
# Save the model and configuration
model.save_pretrained(model_save_path)
```

[ ]:
```
from transformers import ViTForImageClassification

# Load fine-tuned model
model_path = "./vit-food-recognition"
model = ViTForImageClassification.from_pretrained(model_path)
model.eval()
```

[ ]:
```
ViTForImageClassification(
  (vit): ViTModel(
    (embeddings): ViTEmbeddings(
      (patch_embeddings): ViTPatchEmbeddings(
        (projection): Conv2d(3, 768, kernel_size=(16, 16), stride=(16, 16))
      )
      (dropout): Dropout(p=0.0, inplace=False)
    )
    (encoder): ViTEncoder(
      (layer): ModuleList(
        (0-11): 12 x ViTLayer(
          (attention): ViTSdpaAttention(
            (attention): ViTSdpaSelfAttention(
              (query): Linear(in_features=768, out_features=768, bias=True)
              (key): Linear(in_features=768, out_features=768, bias=True)
              (value): Linear(in_features=768, out_features=768, bias=True)
              (dropout): Dropout(p=0.0, inplace=False)
            )
```

```
            (output): ViTSelfOutput(
              (dense): Linear(in_features=768, out_features=768, bias=True)
              (dropout): Dropout(p=0.0, inplace=False)
            )
          )
          (intermediate): ViTIntermediate(
            (dense): Linear(in_features=768, out_features=3072, bias=True)
            (intermediate_act_fn): GELUActivation()
          )
          (output): ViTOutput(
            (dense): Linear(in_features=3072, out_features=768, bias=True)
            (dropout): Dropout(p=0.0, inplace=False)
          )
          (layernorm_before): LayerNorm((768,), eps=1e-12,
elementwise_affine=True)
          (layernorm_after): LayerNorm((768,), eps=1e-12,
elementwise_affine=True)
        )
      )
    )
    (layernorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
  )
  (classifier): Linear(in_features=768, out_features=2, bias=True)
)
```

[ ]:
```python
import torch

# Map label IDs to food names (update based on your dataset's label names)
id_to_food = {i: name for i, name in enumerate(calorie_df["Food"].unique())}

def predict_calories(image_path):
    # Preprocess the image
    input_tensor = preprocess_image(image_path)

    # Perform inference
    with torch.no_grad():
        outputs = model(input_tensor)
        logits = outputs.logits
        predicted_class = torch.argmax(logits, dim=1).item()

    # Map prediction to food item
    food_item = id_to_food.get(predicted_class, "Unknown")
    calories = calorie_mapping.get(food_item, "Calorie data not available")

    return food_item, calories
```

[ ]:
```
from torchvision import transforms

train_transforms = transforms.Compose([
    transforms.RandomResizedCrop(224),
    transforms.RandomHorizontalFlip(),
    transforms.ColorJitter(),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
])
```

[ ]:
```
from transformers import ViTForImageClassification, ViTImageProcessor
from PIL import Image
import torch

# Load the trained model
model_path = "./vit-food-recognition"
model = ViTForImageClassification.from_pretrained(model_path)

# Load the image processor (preprocessing pipeline)
processor = ViTImageProcessor.from_pretrained("google/
 ↪vit-base-patch16-224-in21k")

# Load the image
image_path = "/content/pic_002.jpg"  # Replace with your image path
image = Image.open(image_path).convert("RGB")

# Preprocess the image: Convert it to pixel_values (tensor format)
inputs = processor(images=image, return_tensors="pt")

# Model inference
model.eval()  # Set model to evaluation mode
with torch.no_grad():
    outputs = model(**inputs)  # Provide the inputs as keyword arguments to the␣
 ↪model

# Get the predicted class index (argmax of logits)
logits = outputs.logits
predicted_class_idx = logits.argmax(-1).item()

# Print the predicted class
print(f"Predicted Class Index: {predicted_class_idx}")
```

```
Predicted Class Index: 1
```

[ ]:
```
#  have a mapping of class indices to food items
class_to_food = {0: "Buger", 1: "Pizza", 2: "Salad", 3: "Pasta", 4: "Sandwich"}
```

```
# Fetch the predicted food item
food_item = class_to_food.get(predicted_class_idx, "Unknown")

# Assuming you have a calorie mapping, fetch the calories for the food item
calories = calorie_mapping.get(food_item, "Calorie data not available")

print(f"Predicted Food Item: {food_item}")
print(f"Estimated Calories: {calories}")
```

```
Predicted Food Item: Pizza
Estimated Calories: 2248
```

[ ]:
```
print(dataset)
```

```
DatasetDict({
    train: Dataset({
        features: ['image', 'label'],
        num_rows: 3
    })
    validation: Dataset({
        features: ['image', 'label'],
        num_rows: 2
    })
})
```

[ ]:
```
{'train': Dataset, 'validation': Dataset}
```

[ ]:
```
{'train': datasets.arrow_dataset.Dataset,
 'validation': datasets.arrow_dataset.Dataset}
```

[ ]:
```
# Manually split the dataset into train and test
dataset = dataset['train'].train_test_split(test_size=0.2)

# Now use the splits
train_dataset = dataset['train']
test_dataset = dataset['test']

print(f"Train Dataset: {len(train_dataset)} samples")
print(f"Test Dataset: {len(test_dataset)} samples")
```

```
Train Dataset: 2 samples
Test Dataset: 1 samples
```

[ ]:
```
from transformers import ViTImageProcessor, ViTForImageClassification
from PIL import Image
import torch
```

```
# Step 1: Load the model and processor
model = ViTForImageClassification.from_pretrained("google/
 ↪vit-base-patch16-224-in21k")
processor = ViTImageProcessor.from_pretrained("google/
 ↪vit-base-patch16-224-in21k")

# Step 2: Load and preprocess the image
image_path = "/content/pic_002.jpg"  # Replace with your image path
image = Image.open(image_path).convert("RGB")

# Step 3: Preprocess the image and get pixel values
inputs = processor(images=image, return_tensors="pt")  # This will return␣
 ↪pixel_values in a tensor format
pixel_values = inputs["pixel_values"]  # This is what you pass to the model

# Step 4: Perform inference
with torch.no_grad():
    outputs = model(pixel_values=pixel_values)
    logits = outputs.logits

# Step 5: Get the predicted class
predicted_class = torch.argmax(logits, dim=-1).item()
print(f"Predicted Class ID: {predicted_class}")
```

```
Some weights of ViTForImageClassification were not initialized from the model
checkpoint at google/vit-base-patch16-224-in21k and are newly initialized:
['classifier.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it
for predictions and inference.
```

```
Predicted Class ID: 1
```

[ ]:
```
from transformers import ViTImageProcessor, ViTForImageClassification
from PIL import Image
import torch
import numpy as np

# Step 1: Load the model and processor
model = ViTForImageClassification.from_pretrained("google/
 ↪vit-base-patch16-224-in21k")
processor = ViTImageProcessor.from_pretrained("google/
 ↪vit-base-patch16-224-in21k")

# Set the model to evaluation mode
model.eval()

# Function to calculate accuracy
```

```
def calculate_accuracy(predictions, labels):
    correct = np.sum(predictions == labels)
    total = len(labels)
    accuracy = correct / total
    return accuracy

# Step 2: Define your test dataset (replace with your actual test data)
# For the sake of example, let's assume a dataset of image paths and␣
 ↪corresponding true labels
# This list should be replaced with your actual image paths and corresponding␣
 ↪labels
image_paths = [ "/content/pic_002.jpg"]  # Example image paths
true_labels = [0, 1]  # Example true labels (0 = Pizza, 1 = Burger, etc.)

# Step 3: Perform inference and calculate accuracy
predictions = []
for image_path in image_paths:
    # Load and preprocess the image
    image = Image.open(image_path).convert("RGB")
    inputs = processor(images=image, return_tensors="pt")
    pixel_values = inputs["pixel_values"]

    # Perform inference
    with torch.no_grad():
        outputs = model(pixel_values=pixel_values)
        logits = outputs.logits
        predicted_class = torch.argmax(logits, dim=-1).item()  # Get predicted␣
 ↪class ID
        predictions.append(predicted_class)

# Step 4: Calculate accuracy
accuracy = calculate_accuracy(np.array(predictions), np.array(true_labels))
print(f"Accuracy: {accuracy:.2f}")
```

```
Some weights of ViTForImageClassification were not initialized from the model
checkpoint at google/vit-base-patch16-224-in21k and are newly initialized:
['classifier.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it
for predictions and inference.
```

```
Accuracy: 0.50
```

[ ]:
```
from torchvision import transforms

train_transforms = transforms.Compose([
    transforms.RandomResizedCrop(224),  # Randomly crop and resize images to␣
 ↪224x224
```

```
    transforms.RandomHorizontalFlip(),  # Random horizontal flip
    transforms.ToTensor(),              # Convert image to tensor
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])␣
 ↪ # Normalize
])

# Apply these transformations during dataset loading
```

[ ]:
```
from transformers import ViTForImageClassification

# Load pre-trained ViT model with fine-tuning
model = ViTForImageClassification.from_pretrained(
    "google/vit-base-patch16-224-in21k",  # Pre-trained model
    num_labels=num_classes,  # Specify number of output classes
    ignore_mismatched_sizes=True  # Ignore mismatch between model and dataset
)
```

```
Some weights of ViTForImageClassification were not initialized from the model
checkpoint at google/vit-base-patch16-224-in21k and are newly initialized:
['classifier.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it
for predictions and inference.
```

[ ]:
```
from transformers import TrainingArguments

# Set up training arguments
training_args = TrainingArguments(
    output_dir="./vit-food-recognition",  # Output directory for the model
    evaluation_strategy="epoch",  # Evaluate after every epoch
    learning_rate=2e-5,  # Learning rate
    per_device_train_batch_size=8,  # Batch size per device
    num_train_epochs=5,  # Number of epochs
    weight_decay=0.01,  # Weight decay to regularize the model
    logging_dir="./logs",  # Directory for logs
    logging_steps=10,  # Log every 10 steps
    load_best_model_at_end=True,  # Load the best model at the end
    save_strategy="epoch"  # Save after every epoch
)
```

```
/usr/local/lib/python3.10/dist-packages/transformers/training_args.py:1568:
FutureWarning: `evaluation_strategy` is deprecated and will be removed in
version 4.46 of  Transformers. Use `eval_strategy` instead
  warnings.warn(
```

[ ]:
```
from torchvision import transforms

train_transforms = transforms.Compose([
```

```
    transforms.RandomResizedCrop(224),  # Randomly crop and resize images
    transforms.RandomHorizontalFlip(),  # Random horizontal flip
    transforms.ColorJitter(brightness=0.2, contrast=0.2, saturation=0.2, hue=0.
 ↪2),  # Randomly adjust brightness, contrast, etc.
    transforms.ToTensor(),  # Convert image to tensor
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])␣
 ↪ # Normalize to ImageNet standards
])

test_transforms = transforms.Compose([
    transforms.Resize(256),
    transforms.CenterCrop(224),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])
```

[ ]:
```
from transformers import TrainingArguments

training_args = TrainingArguments(
    output_dir="./vit-food-recognition",  # Output directory for model␣
 ↪checkpoints
    evaluation_strategy="epoch",  # Evaluate after every epoch
    learning_rate=2e-5,  # Small learning rate for fine-tuning
    per_device_train_batch_size=16,  # Larger batch size
    num_train_epochs=10,  # Train for 10 epochs
    weight_decay=0.01,  # Regularization
    logging_dir="./logs",  # Directory for logs
    logging_steps=10,  # Log every 10 steps
    save_strategy="epoch",  # Save model after each epoch
    load_best_model_at_end=True,  # Load best model after training
)
```

[ ]:
```
from torch import nn

# Adding Dropout to the classifier head of the ViT model
model.classifier = nn.Sequential(
    nn.Dropout(p=0.3),  # Dropout with 30% probability
    nn.Linear(model.config.hidden_size, num_classes)
)
```

[ ]:
```
from transformers import get_scheduler

# Define optimizer and scheduler
optimizer = torch.optim.AdamW(model.parameters(), lr=2e-5)

lr_scheduler = get_scheduler(
    "linear",  # Linear scheduler
```

```
    optimizer=optimizer,
    num_warmup_steps=0,  # No warm-up steps
    num_training_steps=len(train_dataset) * training_args.num_train_epochs
)
```

[ ]:
```
from transformers import ViTImageProcessor
from PIL import Image

# Load an image (replace with your own image path)
image = Image.open("/content/pic_002.jpg")

# Load the ViT image processor
processor = ViTImageProcessor.from_pretrained("google/
 ↪vit-base-patch16-224-in21k")

# Preprocess the image to get pixel_values
inputs = processor(images=image, return_tensors="pt")

# `inputs` will be a dictionary containing the pixel_values tensor
pixel_values = inputs["pixel_values"]

# Print the shape of the pixel_values tensor
print(pixel_values.shape)  # Expected output: torch.Size([1, 3, 224, 224])
```

```
torch.Size([1, 3, 224, 224])
```

[ ]:
```

```

[ ]:
```
print(dataset.keys())  # To see the available keys
```

```
dict_keys(['test'])
```