【条件】
- PyTorch 2.4 以上
- torchvision.models.efficientnet_v2_s(事前学習済み)を使用
- CIFAR-10 データセットで動作
- main() 関数で全体を制御する構成にする
- モデルの定義など設定はconfig.pyとして作成する
- データの読み込みなど実行に関するメイン処理はtrain.pyとして作成する
  - データの読み込みのみsys.argv[1]でコマンドライン引数を適切に使う

追加要件:
- GPU対応(device = "cuda" if torch.cuda.is_available() else "cpu")

EfficientNetV2を使ったCIFAR-10画像分類の学習スクリプトを作成しました。コンパクトな実装を心がけています。

config.py
コード

実行方法

bash

```
python train.py ./data
```

スクリプトの特徴

1. **config.py**: モデル定義と設定を集約
   - EfficientNetV2-S（事前学習済み）を読み込み
   - 最終層をCIFAR-10用（10クラス）に変更

Claudeに返信...

Sonnet 4.5

コピー 公開

```python
import torch
import torch.nn as nn
from torchvision import models, transforms

# モデル設定
NUM_CLASSES = 10
LEARNING_RATE = 0.001
BATCH_SIZE = 64
NUM_EPOCHS = 10
INPUT_SIZE = 224

# デバイス設定
DEVICE = "cuda" if torch.cuda.is_available() else "cpu"

# データ拡張・正規化
TRAIN_TRANSFORM = transforms.Compose([
    transforms.Resize((INPUT_SIZE, INPUT_SIZE)),
    transforms.RandomHorizontalFlip(),
    transforms.RandomRotation(10),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])

TEST_TRANSFORM = transforms.Compose([
    transforms.Resize((INPUT_SIZE, INPUT_SIZE)),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])

def get_model():
    """EfficientNetV2-Sモデルを取得し、最終層をCIFAR-10用に変更"""
    model = models.efficientnet_v2_s(weights=models.EfficientNet_V2_S_Weights.DEFAULT
    model.classifier[1] = nn.Linear(model.classifier[1].in_features, NUM_CLASSES)
    return model.to(DEVICE)
```