Contenido

Introduction

Some of the facilities that the library allows is that you can use your own datasets, add more benchmark datasets, add your own models or pruning methods.

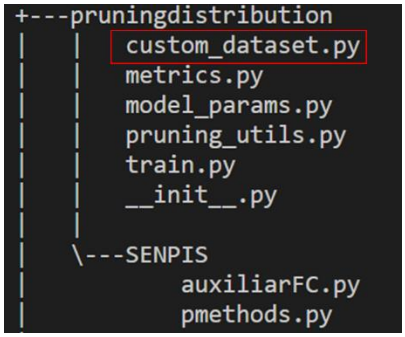
To achieve this, it is not to use pip install pruningdistribution==0.1.0 otherwise:

```
git clone https://github.com/DEEP-CGPS/PruningDistribution
cd PruningDistribution
pip3 install -r requirements.txt
```

Since some changes require adding your new components in the library classes.

How to Use Your Own Dataset

By default, in the example notebooks only the CIFAR-10 benchmark dataset is used, however, many times you want to use your own datasets. Inside pruningdistribution, you will find the file custom_dataset.py which contains a class that allows you to use custom datasets.



For both benchmark and custom datasets, the same function get_dataset, found inside train.py, will always be used:

Here is an example of how to use your own dataset:

First define your own arguments, your dataset must be stored in a folder named data, so if your dataset is named Date_Fruit_7classes, the path will be data/ Date_Fruit_7classes.

```
args.method = 'SenpisFaster'
args.dataset = "Date_Fruit_7classes"
args.eval_metric = "f1_score"
custom_split = 1    # 1: if dataset is already divided into train and test folders,
                    # 0: if all the images of the dataset are in a single folder
```

Then you can get your own dataset, it should be noted that in the case of using SenpisFaster, it returns an additional parameter.

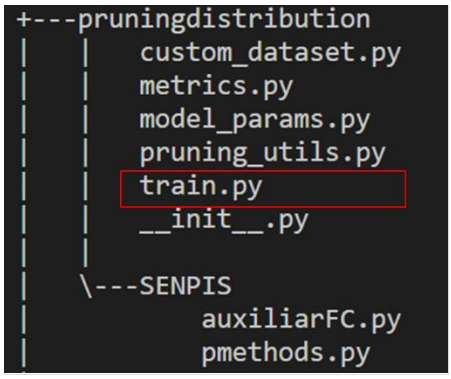
Get Model, DATASET and TRAIN

```
if args.method != 'SenpisFaster':
    train_loader, test_loader, num_classes, _ = get_dataset(args, custom_split = custom_split)
    trainset = None
else:
    train_loader, test_loader, num_classes, trainset = get_dataset(args, custom_split = custom_split)
```

The rest of the process can be the same as for notebooks.

How to Add New Benchmark Datasets

Currently the library only has by default CIFAR-10, CIRFAR-100 and FashionMNIST. But many times, it will be necessary to use other datasets. for this it is necessary to first go to the train.py file.



Then, in the get_dataset function, there is a comment (line 83-89) that indicates how to add a new benchmark dataset:

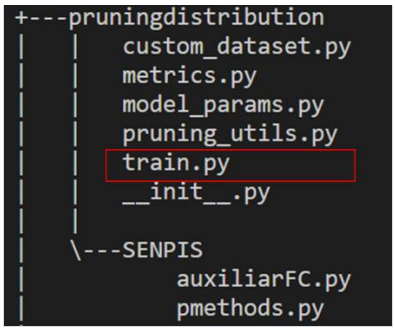
```
59  def get_dataset(args, custom_split = 0):
60      transform = transforms.Compose(
61          [transforms.Resize((224,224)),
62           transforms.ToTensor(),
63           transforms.Normalize((0.4914, 0.4822, 0.4465), (0.2023, 0.1994, 0.2010))])
64
65      if args.dataset == "CIFAR10":
66          trainset = torchvision.datasets.CIFAR10(root='./data', train=True,
67                                                  download=True, transform=transform)
68          testset = torchvision.datasets.CIFAR10(root='./data', train=False,
69                                                 download=True, transform=transform)
70          num_classes = len(trainset.classes)
71      elif args.dataset == "CIFAR100":
72          trainset = torchvision.datasets.CIFAR100(root='./data', train=True,
73                                                   download=True, transform=transform)
74          testset = torchvision.datasets.CIFAR100(root='./data', train=False,
75                                                  download=True, transform=transform)
76          num_classes = len(trainset.classes)
77      elif args.dataset == "FashionMNIST":
78          trainset = torchvision.datasets.FashionMNIST(root='./data', train=True,
79                                                       download=True, transform=transform)
80          testset = torchvision.datasets.FashionMNIST(root='./data', train=False,
81                                                      download=True, transform=transform)
82          num_classes = len(trainset.classes)
83      # Add more datasets as needed
84      # Example: elif args.dataset == "AnotherDataset":
85      #              trainset = torchvision.datasets.AnotherDataset(root='./data', train=True,
86      #                                                             download=True, transform=transform)
87      #              testset = torchvision.datasets.AnotherDataset(root='./data', train=False,
88      #                                                            download=True, transform=transform)
89      #              num_classes = len(trainset.classes)
90
```

To know which datasets you could add, go to pytorch documentation:

https://pytorch.org/vision/stable/datasets.html

How to Add New Models

The process to add a new model is very similar to adding a new dataset. First it is necessary to go to the train.py file:



Then, in the get_model function, there are several models supported by the library at the moment: ResNet18, VGG16, DenseNet121, MobileNetV2, AlexNet, VGG11, VGG19. If you want to add more models there is an example (Line 44-48):

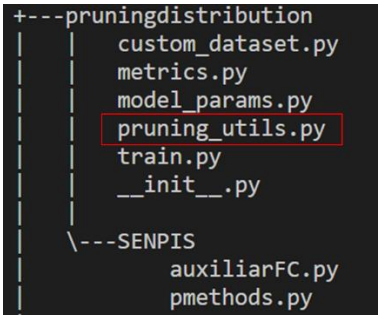
```
18  def get_model(num_classes, args):
19      if args.model_architecture == "ResNet18":
20          model = torchvision.models.resnet18(weights="ResNet18_Weights.DEFAULT")
21          num_fltrs = model.fc.in_features
22          model.fc = nn.Linear(num_fltrs, num_classes)
23      elif args.model_architecture == "VGG16":
24          model = torchvision.models.vgg16_bn(weights="VGG16_BN_Weights.IMAGENET1K_V1")
25          model.classifier[6] = nn.Linear(4096,num_classes)
26      elif args.model_architecture == "DenseNet121":
27          model = torchvision.models.densenet121(weights="DenseNet121_Weights.DEFAULT")
28          num_fltrs = model.classifier.in_features
29          model.classifier = nn.Linear(num_fltrs, num_classes)
30      elif args.model_architecture == "MobileNetV2":
31          model = torchvision.models.mobilenet_v2(weights="MobileNetV2_Weights.DEFAULT")
32          num_fltrs = model.classifier[1].in_features
33          model.classifier[1] = nn.Linear(num_fltrs, num_classes)
34      elif args.model_architecture == "AlexNet":
35          model = torchvision.models.alexnet(weights="AlexNet_Weights.DEFAULT")
36          num_fltrs = model.classifier[6].in_features
37          model.classifier[6] = nn.Linear(num_fltrs, num_classes)
38      elif args.model_architecture == "VGG11":
39          model = torchvision.models.vgg11_bn(weights="VGG11_BN_Weights.IMAGENET1K_V1")
40          model.classifier[6] = nn.Linear(4096, num_classes)
41      elif args.model_architecture == "VGG19":
42          model = torchvision.models.vgg19_bn(weights="VGG19_BN_Weights.IMAGENET1K_V1")
43          model.classifier[6] = nn.Linear(4096, num_classes)
44      # Add more models as needed
45      # Example: elif args.model_architecture == "AnotherModel":
46      #              model = models.another_model(pretrained=True)
47      #              num_fltrs = model.fc.in_features  # Adjust based on the model's architecture
48      #              model.fc = nn.Linear(num_fltrs, num_classes)
49      else:
50          raise ValueError("Model architecture not supported")
51      return model
```

Note:

- Keep in mind that not all pruning methods support all architecture types.
- If you want to use your own model and not a benchmark type, you can add an elif and use torch.load().
- To find out what other models you can add, go to the pytorch documentation: https://pytorch.org/vision/stable/models.html

How to Add Your Own Pruning Method

Currently, the library only supports 3 pruning methods, SenpisFaster, Random and Weight. If you want to add your own pruning method, you must go to the pruning_utils.py file:



In this file there is a function called prune_model, from line 48 to line 51, is the example of where you can add your own pruning method. Please note that the function must modify the model, i.e. reset the weights it considers to zero. It is not necessary to restructure the architecture, the simplify library is used for this.

```python
def prune_model(model: nn.Module, num_classes: int = None, train_loader = None, args = None) -> None:
    """
    Prunes the model and simplifies it.

    Args:
    model (nn.Module): PyTorch model.
    num_classes (int): Number of classes. Defaults to None.
    train_loader: DataLoader for training. Defaults to None.
    args: Additional arguments. Defaults to None.
    """
    if not os.path.exists(f"models/{args.dataset}"):
        os.makedirs(f"models/{args.dataset}")

    torch.manual_seed(args.seed)
    pos = 0
    model.to(args.device)
    model.eval()
    if args.method in ['random','weight']:
        for module in model.modules():
            if isinstance(module, nn.Conv2d):
                if args.method == 'random':
                    prune.random_structured(module, 'weight', amount=args.list_pruning[pos], dim=0)
                elif args.method == 'weight':
                    prune.ln_structured(module, 'weight', amount=args.list_pruning[pos],dim=0,n=2)
                prune.remove(module, 'weight')
                pos += 1
            if isinstance(module, nn.Linear):
                if args.method == 'random':
                    prune.random_structured(module, 'weight', amount=args.list_pruning[pos], dim=0)
                elif args.method == 'weight':
                    prune.ln_structured(module, 'weight', amount=args.list_pruning[pos],dim=0,n=2)
                prune.remove(module,'weight')
                pos += 1

    elif args.method == 'SenpisFaster':
        SenpisFaster(model, num_classes, train_loader, args.list_pruning)

    elif args.method == 'new_method':
        # Implement your new pruning method here
        # Example: prune.new_method(module, 'weight', amount=args.list_pruning[pos], dim=0)
        pass

    simplify.simplify(model, torch.ones((1, 3, 224, 224)).to(args.device), fuse_bn=False)

    torch.save(model, f'models/{args.dataset}/{args.model_architecture}_{args.dataset}_{args.method}_{args.model_type}.pth')
```