Teacher-Student Convolutional Neural Network (CNN) enhanced by Mixture of Experts (MoE) fusion

```
import pandas as pd
import numpy as np
import os

base_path = "/kaggle/input/x-ray-images-of-fractured-and-healthy-
bones/X-ray Imaging Dataset for Detecting Fractured vs. Non-Fractured
Bones/Augmented Dataset/"
categories = ["Fractured", "Non-Fractured"]

image_paths = []
labels = []

for category in categories:
    category_path = os.path.join(base_path, category)
    for image_name in os.listdir(category_path):
```

```
        image_path = os.path.join(category_path, image_name)
        image_paths.append(image_path)
        labels.append(category)

df = pd.DataFrame({
    "image_path": image_paths,
    "label": labels
})

df.head()
```

	image_path	label
0	/kaggle/input/x-ray-images-of-fractured-and-he...	Fractured
1	/kaggle/input/x-ray-images-of-fractured-and-he...	Fractured
2	/kaggle/input/x-ray-images-of-fractured-and-he...	Fractured
3	/kaggle/input/x-ray-images-of-fractured-and-he...	Fractured
4	/kaggle/input/x-ray-images-of-fractured-and-he...	Fractured

```
df.tail()
```

	image_path	label
9295	/kaggle/input/x-ray-images-of-fractured-and-he...	Non-Fractured
9296	/kaggle/input/x-ray-images-of-fractured-and-he...	Non-Fractured
9297	/kaggle/input/x-ray-images-of-fractured-and-he...	Non-Fractured
9298	/kaggle/input/x-ray-images-of-fractured-and-he...	Non-Fractured
9299	/kaggle/input/x-ray-images-of-fractured-and-he...	Non-Fractured

```
df.shape
```

```
(9300, 2)
```

```
df.columns
```

```
Index(['image_path', 'label'], dtype='object')
```

```
df.duplicated().sum()
```

```
0
```

```
df.isnull().sum()
```

```
image_path    0
label         0
dtype: int64
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9300 entries, 0 to 9299
Data columns (total 2 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
```

```
 0   image_path  9300 non-null   object
 1   label       9300 non-null   object
dtypes: object(2)
memory usage: 145.4+ KB
```

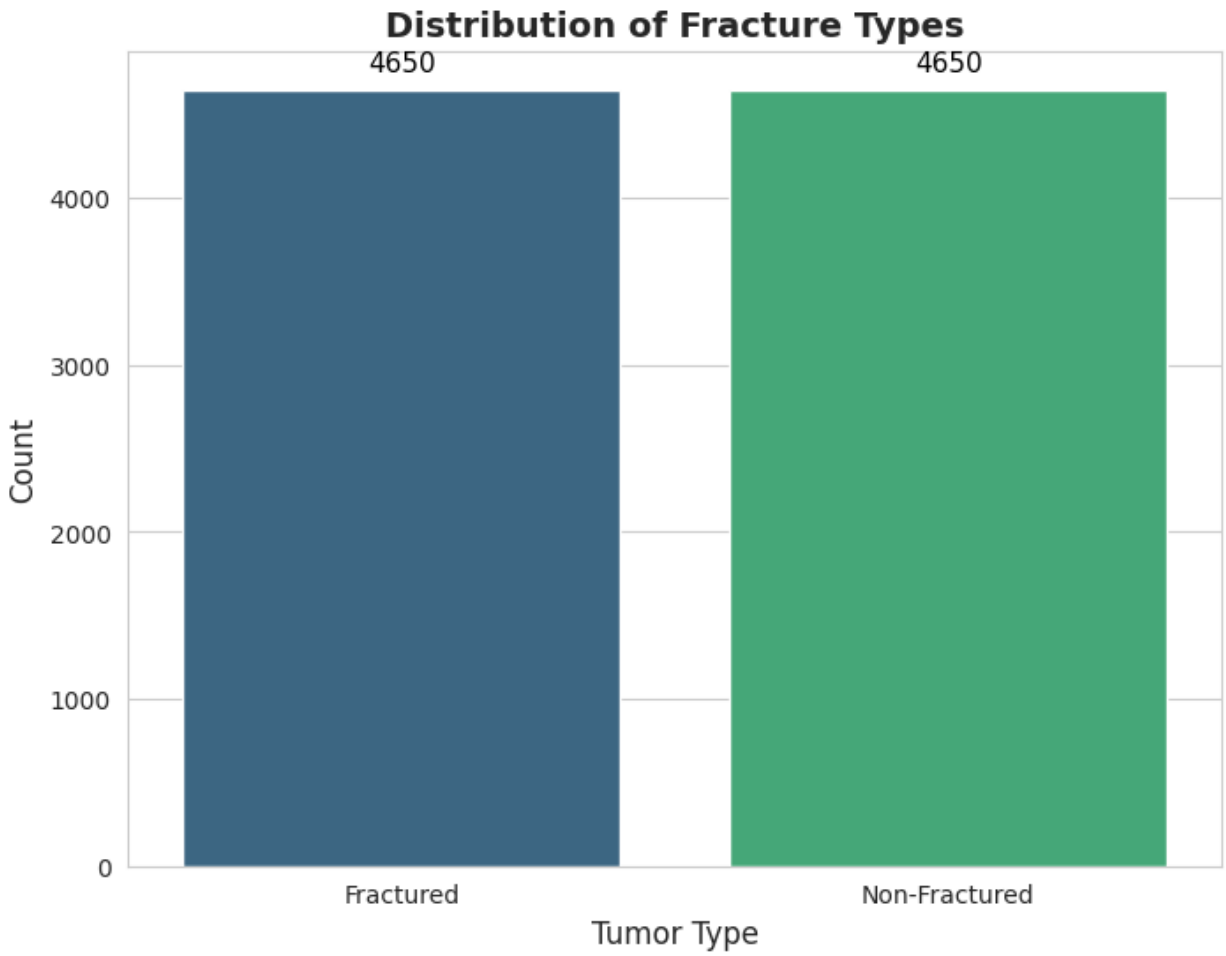
```
df['label'].unique()
```

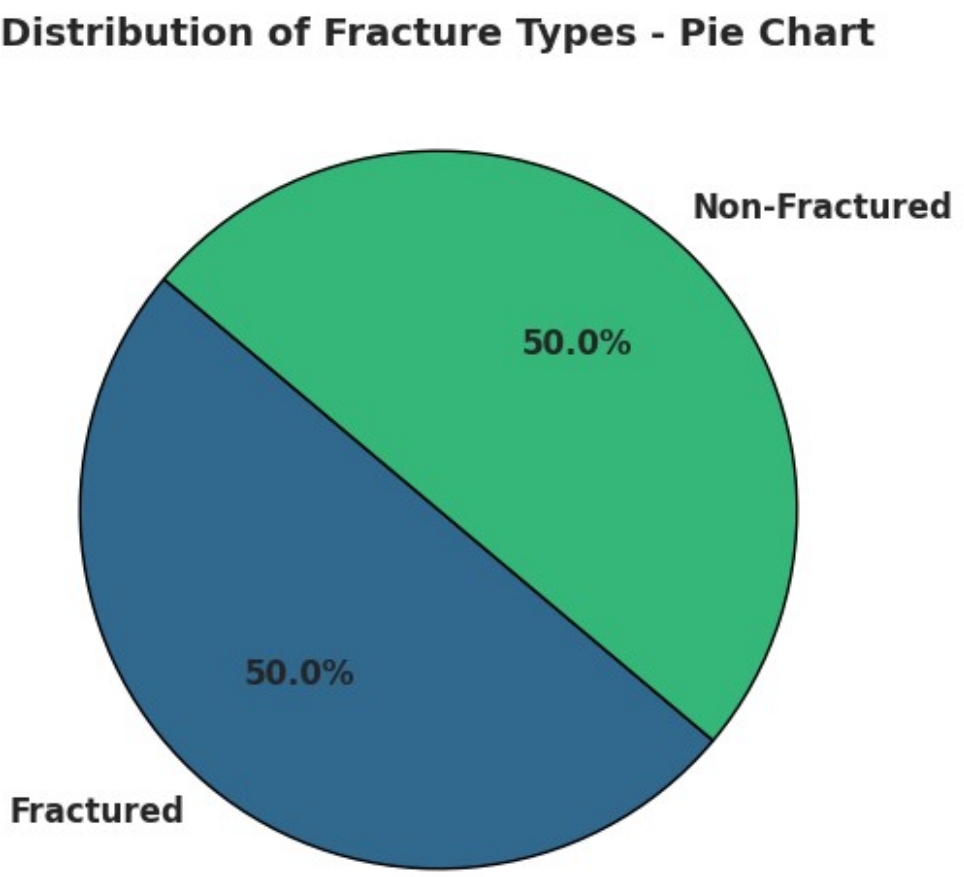
```
array(['Fractured', 'Non-Fractured'], dtype=object)
```

```
df['label'].value_counts()
```

```
label
Fractured        4650
Non-Fractured    4650
Name: count, dtype: int64
```

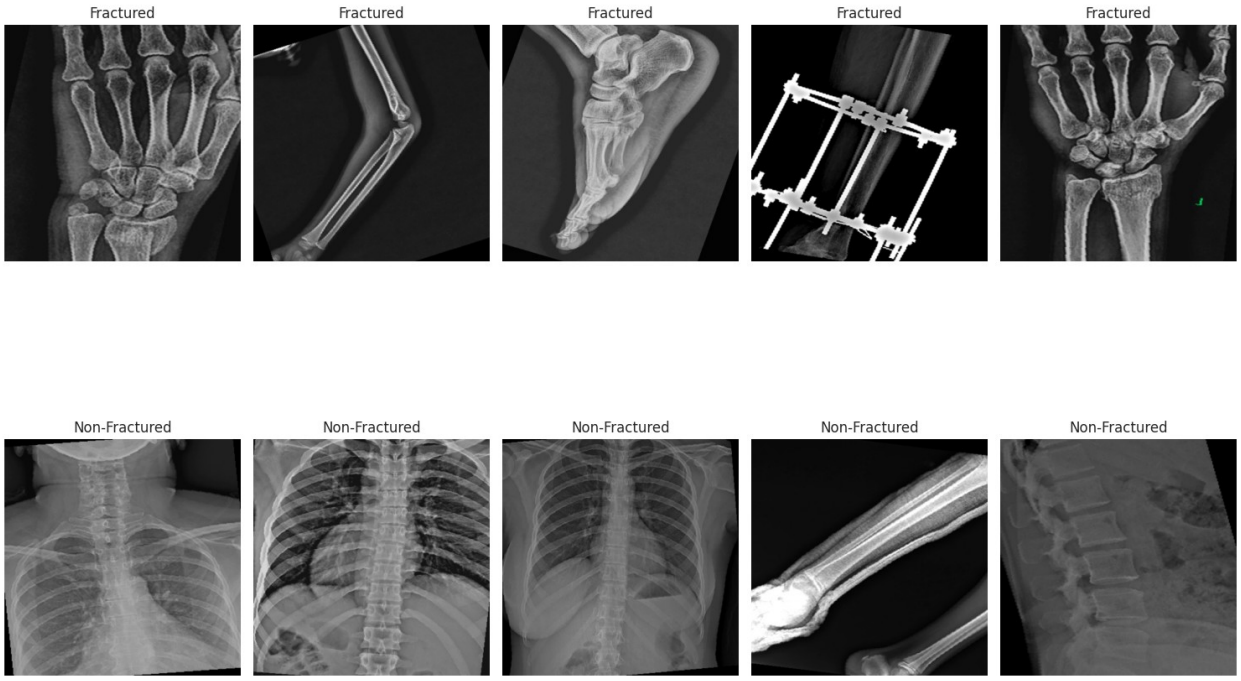
```
import seaborn as sns
import matplotlib.pyplot as plt

sns.set_style("whitegrid")

fig, ax = plt.subplots(figsize=(8, 6))
sns.countplot(data=df, x="label", palette="viridis", ax=ax)

ax.set_title("Distribution of Fracture Types", fontsize=14,
fontweight='bold')
ax.set_xlabel("Tumor Type", fontsize=12)
ax.set_ylabel("Count", fontsize=12)

for p in ax.patches:
    ax.annotate(f'{int(p.get_height())}',
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='bottom', fontsize=11, color='black',
                xytext=(0, 5), textcoords='offset points')

plt.show()

label_counts = df["label"].value_counts()

fig, ax = plt.subplots(figsize=(8, 6))
colors = sns.color_palette("viridis", len(label_counts))

ax.pie(label_counts, labels=label_counts.index, autopct='%1.1f%%',
       startangle=140, colors=colors, textprops={'fontsize': 12,
'weight': 'bold'},
       wedgeprops={'edgecolor': 'black', 'linewidth': 1})

ax.set_title("Distribution of Fracture Types - Pie Chart",
fontsize=14, fontweight='bold')

plt.show()
```


Distribution of Fracture Types
4650
4650
4000
3000
2000
1000
0
Count
Fractured
Non-Fractured
Tumor Type




```
import cv2

num_images = 5

plt.figure(figsize=(15, 12))

for i, category in enumerate(categories):
    category_images = df[df['label'] == category]
['image_path'].iloc[:num_images]

    for j, img_path in enumerate(category_images):

        img = cv2.imread(img_path)
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

        plt.subplot(len(categories), num_images, i * num_images + j +
1)
        plt.imshow(img)
        plt.axis('off')
        plt.title(category)
```

```
plt.tight_layout()
plt.show()
```




```
df
```

	image_path	label
0	/kaggle/input/x-ray-images-of-fractured-and-he...	Fractured
1	/kaggle/input/x-ray-images-of-fractured-and-he...	Fractured
2	/kaggle/input/x-ray-images-of-fractured-and-he...	Fractured
3	/kaggle/input/x-ray-images-of-fractured-and-he...	Fractured
4	/kaggle/input/x-ray-images-of-fractured-and-he...	Fractured
...	...	...
9295	/kaggle/input/x-ray-images-of-fractured-and-he...	Non-Fractured
9296	/kaggle/input/x-ray-images-of-fractured-and-he...	Non-Fractured
9297	/kaggle/input/x-ray-images-of-fractured-and-he...	Non-Fractured
9298	/kaggle/input/x-ray-images-of-fractured-and-he...	Non-Fractured
9299	/kaggle/input/x-ray-images-of-fractured-and-he...	Non-Fractured

```
[9300 rows x 2 columns]

!pip install torchviz

Collecting torchviz
  Downloading torchviz-0.0.3-py3-none-any.whl.metadata (2.1 kB)
Requirement already satisfied: torch in
/usr/local/lib/python3.11/dist-packages (from torchviz) (2.6.0+cu124)
Requirement already satisfied: graphviz in
/usr/local/lib/python3.11/dist-packages (from torchviz) (0.21)
Requirement already satisfied: filelock in
```

```
/usr/local/lib/python3.11/dist-packages (from torch->torchviz)
(3.18.0)
Requirement already satisfied: typing-extensions>=4.10.0 in
/usr/local/lib/python3.11/dist-packages (from torch->torchviz)
(4.14.0)
Requirement already satisfied: networkx in
/usr/local/lib/python3.11/dist-packages (from torch->torchviz) (3.5)
Requirement already satisfied: jinja2 in
/usr/local/lib/python3.11/dist-packages (from torch->torchviz) (3.1.6)
Requirement already satisfied: fsspec in
/usr/local/lib/python3.11/dist-packages (from torch->torchviz)
(2025.5.1)
Collecting nvidia-cuda-nvrtc-cu12==12.4.127 (from torch->torchviz)
  Downloading nvidia_cuda_nvrtc_cu12-12.4.127-py3-none-
manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cuda-runtime-cu12==12.4.127 (from torch->torchviz)
  Downloading nvidia_cuda_runtime_cu12-12.4.127-py3-none-
manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cuda-cupti-cu12==12.4.127 (from torch->torchviz)
  Downloading nvidia_cuda_cupti_cu12-12.4.127-py3-none-
manylinux2014_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cudnn-cu12==9.1.0.70 (from torch->torchviz)
  Downloading nvidia_cudnn_cu12-9.1.0.70-py3-none-
manylinux2014_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cublas-cu12==12.4.5.8 (from torch->torchviz)
  Downloading nvidia_cublas_cu12-12.4.5.8-py3-none-
manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cufft-cu12==11.2.1.3 (from torch->torchviz)
  Downloading nvidia_cufft_cu12-11.2.1.3-py3-none-
manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-curand-cu12==10.3.5.147 (from torch->torchviz)
  Downloading nvidia_curand_cu12-10.3.5.147-py3-none-
manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cusolver-cu12==11.6.1.9 (from torch->torchviz)
  Downloading nvidia_cusolver_cu12-11.6.1.9-py3-none-
manylinux2014_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cusparse-cu12==12.3.1.170 (from torch->torchviz)
  Downloading nvidia_cusparse_cu12-12.3.1.170-py3-none-
manylinux2014_x86_64.whl.metadata (1.6 kB)
Requirement already satisfied: nvidia-cusparselt-cu12==0.6.2 in
/usr/local/lib/python3.11/dist-packages (from torch->torchviz) (0.6.2)
Requirement already satisfied: nvidia-nccl-cu12==2.21.5 in
/usr/local/lib/python3.11/dist-packages (from torch->torchviz)
(2.21.5)
Requirement already satisfied: nvidia-nvtx-cu12==12.4.127 in
/usr/local/lib/python3.11/dist-packages (from torch->torchviz)
(12.4.127)
Collecting nvidia-nvjitlink-cu12==12.4.127 (from torch->torchviz)
  Downloading nvidia_nvjitlink_cu12-12.4.127-py3-none-
```

```
manylinux2014_x86_64.whl.metadata (1.5 kB)
Requirement already satisfied: triton==3.2.0 in
/usr/local/lib/python3.11/dist-packages (from torch->torchviz) (3.2.0)
Requirement already satisfied: sympy==1.13.1 in
/usr/local/lib/python3.11/dist-packages (from torch->torchviz)
(1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
/usr/local/lib/python3.11/dist-packages (from sympy==1.13.1->torch-
>torchviz) (1.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.11/dist-packages (from jinja2->torch->torchviz)
(3.0.2)
Downloading torchviz-0.0.3-py3-none-any.whl (5.7 kB)
Downloading nvidia_cublas_cu12-12.4.5.8-py3-none-
manylinux2014_x86_64.whl (363.4 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 363.4/363.4 MB 4.6 MB/s eta
0:00:00:00:0100:01
anylinux2014_x86_64.whl (13.8 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.8/13.8 MB 92.2 MB/s eta
0:00:00:00:010:01
anylinux2014_x86_64.whl (24.6 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24.6/24.6 MB 70.4 MB/s eta
0:00:00:00:0100:01
e_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl (883 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 883.7/883.7 kB 42.3 MB/s eta
0:00:00
anylinux2014_x86_64.whl (664.8 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 664.8/664.8 MB 2.5 MB/s eta
0:00:00:00:0100:01
anylinux2014_x86_64.whl (211.5 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 211.5/211.5 MB 5.9 MB/s eta
0:00:00:00:0100:01
anylinux2014_x86_64.whl (56.3 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 56.3/56.3 MB 30.8 MB/s eta
0:00:00:00:0100:01
anylinux2014_x86_64.whl (127.9 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 127.9/127.9 MB 13.6 MB/s eta
0:00:0000:0100:01
anylinux2014_x86_64.whl (207.5 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 207.5/207.5 MB 2.0 MB/s eta
0:00:00:00:0100:01
anylinux2014_x86_64.whl (21.1 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21.1/21.1 MB 89.3 MB/s eta
0:00:00:00:0100:01
e-cu12, nvidia-cuda-nvrtc-cu12, nvidia-cuda-cupti-cu12, nvidia-cublas-
cu12, nvidia-cusparse-cu12, nvidia-cudnn-cu12, nvidia-cusolver-cu12,
torchviz
  Attempting uninstall: nvidia-nvjitlink-cu12
    Found existing installation: nvidia-nvjitlink-cu12 12.5.82
```

```
    Uninstalling nvidia-nvjitlink-cu12-12.5.82:
      Successfully uninstalled nvidia-nvjitlink-cu12-12.5.82
  Attempting uninstall: nvidia-curand-cu12
    Found existing installation: nvidia-curand-cu12 10.3.6.82
    Uninstalling nvidia-curand-cu12-10.3.6.82:
      Successfully uninstalled nvidia-curand-cu12-10.3.6.82
  Attempting uninstall: nvidia-cufft-cu12
    Found existing installation: nvidia-cufft-cu12 11.2.3.61
    Uninstalling nvidia-cufft-cu12-11.2.3.61:
      Successfully uninstalled nvidia-cufft-cu12-11.2.3.61
  Attempting uninstall: nvidia-cuda-runtime-cu12
    Found existing installation: nvidia-cuda-runtime-cu12 12.5.82
    Uninstalling nvidia-cuda-runtime-cu12-12.5.82:
      Successfully uninstalled nvidia-cuda-runtime-cu12-12.5.82
  Attempting uninstall: nvidia-cuda-nvrtc-cu12
    Found existing installation: nvidia-cuda-nvrtc-cu12 12.5.82
    Uninstalling nvidia-cuda-nvrtc-cu12-12.5.82:
      Successfully uninstalled nvidia-cuda-nvrtc-cu12-12.5.82
  Attempting uninstall: nvidia-cuda-cupti-cu12
    Found existing installation: nvidia-cuda-cupti-cu12 12.5.82
    Uninstalling nvidia-cuda-cupti-cu12-12.5.82:
      Successfully uninstalled nvidia-cuda-cupti-cu12-12.5.82
  Attempting uninstall: nvidia-cublas-cu12
    Found existing installation: nvidia-cublas-cu12 12.5.3.2
    Uninstalling nvidia-cublas-cu12-12.5.3.2:
      Successfully uninstalled nvidia-cublas-cu12-12.5.3.2
  Attempting uninstall: nvidia-cusparse-cu12
    Found existing installation: nvidia-cusparse-cu12 12.5.1.3
    Uninstalling nvidia-cusparse-cu12-12.5.1.3:
      Successfully uninstalled nvidia-cusparse-cu12-12.5.1.3
  Attempting uninstall: nvidia-cudnn-cu12
    Found existing installation: nvidia-cudnn-cu12 9.3.0.75
    Uninstalling nvidia-cudnn-cu12-9.3.0.75:
      Successfully uninstalled nvidia-cudnn-cu12-9.3.0.75
  Attempting uninstall: nvidia-cusolver-cu12
    Found existing installation: nvidia-cusolver-cu12 11.6.3.83
    Uninstalling nvidia-cusolver-cu12-11.6.3.83:
      Successfully uninstalled nvidia-cusolver-cu12-11.6.3.83
Successfully installed nvidia-cublas-cu12-12.4.5.8 nvidia-cuda-cupti-
cu12-12.4.127 nvidia-cuda-nvrtc-cu12-12.4.127 nvidia-cuda-runtime-
cu12-12.4.127 nvidia-cudnn-cu12-9.1.0.70 nvidia-cufft-cu12-11.2.1.3
nvidia-curand-cu12-10.3.5.147 nvidia-cusolver-cu12-11.6.1.9 nvidia-
cusparse-cu12-12.3.1.170 nvidia-nvjitlink-cu12-12.4.127 torchviz-0.0.3
```

```
import torch
import torch.nn as nn
import torch.nn.functional as F
import torchvision
import torchvision.transforms as transforms
from torch.utils.data import Dataset, DataLoader
```

```
from torch.optim.lr_scheduler import CosineAnnealingLR
from torchvision.models import resnet18, ResNet18_Weights
from torchinfo import summary
from torchviz import make_dot
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.model_selection import train_test_split
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from PIL import Image
import pandas as pd

NUM_CLASSES = 2
HIDDEN_DIM = 512
NUM_EXPERTS = 2
NUM_EPOCHS = 5
BATCH_SIZE = 128
LEARNING_RATE = 1e-3
WEIGHT_DECAY = 1e-4
LAMBDA_ST = 0.1
GAMMA_MOE = 0.0005
MISS_PROB = 0.0

class MoEFeatureFusion(nn.Module):
    def __init__(self, input_dim, hidden_dim, num_experts, top_k=1):
        super(MoEFeatureFusion, self).__init__()
        self.input_dim = input_dim
        self.hidden_dim = hidden_dim
        self.num_experts = num_experts
        self.top_k = top_k
        self.gate = nn.Linear(input_dim, num_experts)
        self.experts = nn.ModuleList([
            nn.Sequential(
                nn.Linear(input_dim, hidden_dim * 2),
                nn.GELU(),
                nn.Linear(hidden_dim * 2, hidden_dim)
            ) for _ in range(num_experts)
        ])
        self.shared_expert = nn.Sequential(
            nn.Linear(input_dim, hidden_dim * 2),
            nn.GELU(),
            nn.Linear(hidden_dim * 2, hidden_dim)
        )
        self.layer_norm = nn.LayerNorm(hidden_dim)
        for m in self.modules():
            if isinstance(m, nn.Linear):
                nn.init.kaiming_normal_(m.weight, mode='fan_out',
nonlinearity='relu')
                m.weight.data *= 0.1
```

```
    def forward(self, x):
        batch_size, num_features, _ = x.shape
        gate_scores = F.softmax(self.gate(x), dim=-1)
        topk_scores, topk_indices = gate_scores.topk(self.top_k, dim=-
1)
        out = torch.zeros(batch_size, num_features, self.hidden_dim,
device=x.device)
        for i in range(self.top_k):
            expert_idx = topk_indices[:, :, i]
            expert_scores = topk_scores[:, :, i]
            for m in range(self.num_experts):
                mask = (expert_idx == m).float()
                if mask.sum() > 0:
                    expert_out = self.experts[m](x)
                    out += mask.unsqueeze(-1) *
expert_scores.unsqueeze(-1) * self.layer_norm(expert_out)
        shared_out = self.shared_expert(x)
        out += self.layer_norm(shared_out)
        gate_mean = gate_scores.mean(dim=(0, 1))
        fraction = (gate_scores > 0).float().mean(dim=(0, 1))
        moe_loss = (gate_mean * fraction).sum()
        return out, moe_loss

class TeacherStudentCNN(nn.Module):
    def __init__(self, num_classes, hidden_dim, num_experts):
        super(TeacherStudentCNN, self).__init__()
        self.hidden_dim = hidden_dim
        self.num_classes = num_classes
        self.conv_backbone =
resnet18(weights=ResNet18_Weights.DEFAULT)
        self.conv_backbone.fc = nn.Identity()
        for param in self.conv_backbone.parameters():
            param.requires_grad = False
        for param in self.conv_backbone.layer2.parameters():
            param.requires_grad = True
        for param in self.conv_backbone.layer3.parameters():
            param.requires_grad = True
        for param in self.conv_backbone.layer4.parameters():
            param.requires_grad = True
        self.teacher_feature = nn.Sequential(
            nn.Linear(512, hidden_dim),
            nn.BatchNorm1d(hidden_dim),
            nn.ReLU(),
            nn.Dropout(0.4)
        )
        self.student_feature = nn.Sequential(
            nn.Linear(512, hidden_dim),
            nn.BatchNorm1d(hidden_dim),
            nn.ReLU(),
```

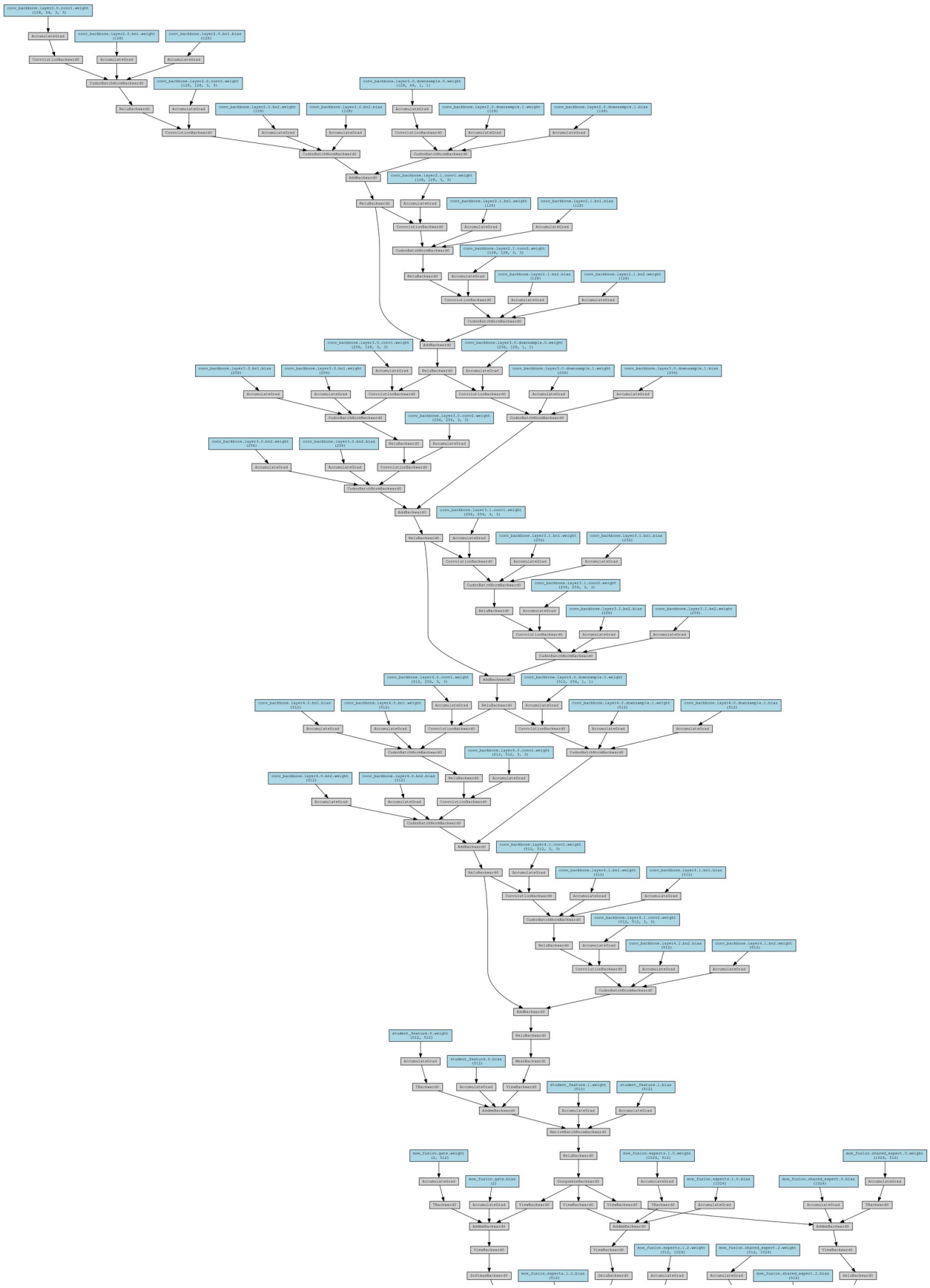
```
            nn.Dropout(0.4)
        )
        self.moe_fusion = MoEFeatureFusion(hidden_dim, hidden_dim,
num_experts)
        self.teacher_classifier = nn.Linear(hidden_dim, num_classes)
        self.student_classifier = nn.Linear(hidden_dim, num_classes)
        for m in self.modules():
            if isinstance(m, nn.Linear):
                nn.init.kaiming_normal_(m.weight, mode='fan_out',
nonlinearity='relu')
                m.weight.data *= 0.1

    def forward(self, x, is_training=True):
        batch_size = x.shape[0]
        x_teacher = x
        x_student = x
        feat_teacher = self.conv_backbone(x_teacher)
        feat_student = self.conv_backbone(x_student)
        teacher_features = self.teacher_feature(feat_teacher)
        student_features = self.student_feature(feat_student)
        teacher_input = torch.stack([teacher_features,
student_features], dim=1)
        student_input = student_features.unsqueeze(1)
        teacher_fused, moe_loss_teacher =
self.moe_fusion(teacher_input)
        student_fused, moe_loss_student =
self.moe_fusion(student_input)
        teacher_fused = teacher_fused.mean(dim=1)
        student_fused = student_fused.mean(dim=1)
        teacher_out = self.teacher_classifier(teacher_fused)
        student_out = self.student_classifier(student_fused)
        return student_out, teacher_out, moe_loss_teacher +
moe_loss_student

    def inference(self, x):
        feat = self.conv_backbone(x)
        feat = self.student_feature(feat)
        feat = feat.unsqueeze(1)
        feat, _ = self.moe_fusion(feat)
        feat = feat.mean(dim=1)
        return self.student_classifier(feat)

class FractureDataset(Dataset):
    def __init__(self, df, transform=None):
        self.df = df
        self.transform = transform
        self.label_map = {'Non-Fractured': 0, 'Fractured': 1}

    def __len__(self):
        return len(self.df)
```

```
    def __getitem__(self, idx):
        img_path = self.df.iloc[idx]['image_path']
        label = self.df.iloc[idx]['label']
        label = self.label_map.get(label, label) if isinstance(label,
str) else label
        image = Image.open(img_path).convert('RGB')
        if self.transform:
            image = self.transform(image)
        return image, label

train_transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.RandomCrop(224, padding=28),
    transforms.RandomHorizontalFlip(p=0.5),
    transforms.ColorJitter(brightness=0.2, contrast=0.2,
saturation=0.2),
    transforms.RandomRotation(15),
    transforms.ToTensor(),
    transforms.Lambda(lambda x: x.repeat(3, 1, 1) if x.size(0) == 1
else x),
    transforms.Normalize((0.485, 0.456, 0.406), (0.229, 0.224, 0.225))
])

test_transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
    transforms.Lambda(lambda x: x.repeat(3, 1, 1) if x.size(0) == 1
else x),
    transforms.Normalize((0.485, 0.456, 0.406), (0.229, 0.224, 0.225))
])

train_df, test_df = train_test_split(df, test_size=0.2,
stratify=df['label'], random_state=42)

train_dataset = FractureDataset(train_df, transform=train_transform)
test_dataset = FractureDataset(test_df, transform=test_transform)
train_loader = DataLoader(train_dataset, batch_size=BATCH_SIZE,
shuffle=True, num_workers=2)
test_loader = DataLoader(test_dataset, batch_size=BATCH_SIZE,
shuffle=False, num_workers=2)

model = TeacherStudentCNN(num_classes=NUM_CLASSES,
hidden_dim=HIDDEN_DIM, num_experts=NUM_EXPERTS)
optimizer = torch.optim.AdamW(model.parameters(), lr=LEARNING_RATE,
weight_decay=WEIGHT_DECAY)
scheduler = CosineAnnealingLR(optimizer, T_max=NUM_EPOCHS)
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
model.to(device)
```

```
summary(model, input_size=(BATCH_SIZE, 3, 224, 224),
col_names=["input_size", "output_size", "num_params"])

model.eval()
sample_input = torch.randn(2, 3, 224, 224).to(device)
student_out, teacher_out, moe_loss = model(sample_input,
is_training=False)
graph = make_dot(student_out, params=dict(model.named_parameters()))
graph.render("model_architecture", format="png", cleanup=True)
model.train()

def train(model, train_loader, optimizer, scheduler, num_epochs):
    model.train()
    for epoch in range(num_epochs):
        total_loss = 0
        for images, labels in train_loader:
            if torch.isnan(images).any() or torch.isinf(images).any():
                continue
            images, labels = images.to(device), labels.to(device)
            optimizer.zero_grad()
            student_out, teacher_out, moe_loss = model(images)
            ce_loss = F.cross_entropy(teacher_out, labels)
            if torch.isnan(ce_loss):
                continue
            st_loss = F.kl_div(F.log_softmax(student_out, dim=-1),
F.softmax(teacher_out.detach(), dim=-1), reduction='batchmean')
            loss = ce_loss + LAMBDA_ST * st_loss + GAMMA_MOE *
moe_loss
            if torch.isnan(loss):
                continue
            loss.backward()
            torch.nn.utils.clip_grad_norm_(model.parameters(),
max_norm=1.0)
            optimizer.step()
            total_loss += loss.item()
        scheduler.step()
        print(f'Epoch {epoch+1}, Loss: {total_loss /
len(train_loader):.4f}')

def evaluate(model, test_loader):
    model.eval()
    correct = 0
    total = 0
    all_preds = []
    all_labels = []
    with torch.no_grad():
        for images, labels in test_loader:
            images, labels = images.to(device), labels.to(device)
            outputs = model.inference(images)
            _, predicted = torch.max(outputs, 1)
```

```
            total += labels.size(0)
            correct += (predicted == labels).sum().item()
            all_preds.extend(predicted.cpu().numpy())
            all_labels.extend(labels.cpu().numpy())
    accuracy = 100 * correct / total
    print(f'Test Accuracy: {accuracy:.2f}%')
    print("\nConfusion Matrix:")
    cm = confusion_matrix(all_labels, all_preds)
    print(cm)
    print("\nClassification Report:")
    print(classification_report(all_labels, all_preds,
target_names=['Non-Fractured', 'Fractured']))
    plt.figure(figsize=(8, 6))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
xticklabels=['Non-Fractured', 'Fractured'], yticklabels=['Non-
Fractured', 'Fractured'])
    plt.xlabel('Predicted')
    plt.ylabel('True')
    plt.title('Confusion Matrix')
    plt.savefig('confusion_matrix.png')
    plt.close()
    return accuracy

train(model, train_loader, optimizer, scheduler, NUM_EPOCHS)
evaluate(model, test_loader)
```

```
Epoch 1, Loss: 2.1966
Epoch 2, Loss: 0.2839
Epoch 3, Loss: 0.1720
Epoch 4, Loss: 0.0877
Epoch 5, Loss: 0.0366
Test Accuracy: 99.09%

Confusion Matrix:
[[920  10]
 [  7 923]]

Classification Report:
              precision    recall  f1-score   support

Non-Fractured       0.99      0.99      0.99       930
    Fractured       0.99      0.99      0.99       930

     accuracy                           0.99      1860
    macro avg       0.99      0.99      0.99      1860
 weighted avg       0.99      0.99      0.99      1860


99.08602150537635
```


conv_backbone.layer2.0.conv1.weight
(128, 64, 3, 3)
AccumulateGrad
conv_backbone.layer2.0.bn1.weight
(128)
conv_backbone.layer2.0.bn1.bias
(128)
ConvolutionBackward0
CudnnBatchNormBackward0
ReluBackward0
conv_backbone.layer2.0.conv2.weight
(128, 128, 3, 3)
conv_backbone.layer2.0.bn2.weight
(128)
conv_backbone.layer2.0.bn2.bias
(128)
conv_backbone.layer2.0.downsample.0.weight
(128, 64, 1, 1)
conv_backbone.layer2.0.downsample.1.weight
(128)
conv_backbone.layer2.0.downsample.1.bias
(128)
AddBackward0
conv_backbone.layer2.1.conv1.weight
(128, 128, 3, 3)
conv_backbone.layer2.1.bn1.weight
(128)
conv_backbone.layer2.1.bn1.bias
(128)
conv_backbone.layer2.1.conv2.weight
(128, 128, 3, 3)
conv_backbone.layer2.1.bn2.bias
(128)
conv_backbone.layer2.1.bn2.weight
(128)
conv_backbone.layer3.0.conv1.weight
(256, 128, 3, 3)
conv_backbone.layer3.0.downsample.0.weight
(256, 128, 1, 1)
conv_backbone.layer3.0.bn1.bias
(256)
conv_backbone.layer3.0.bn1.weight
(256)
conv_backbone.layer3.0.downsample.1.weight
(256)
conv_backbone.layer3.0.downsample.1.bias
(256)
conv_backbone.layer3.0.conv2.weight
(256, 256, 3, 3)
conv_backbone.layer3.0.bn2.weight
(256)
conv_backbone.layer3.0.bn2.bias
(256)
conv_backbone.layer3.1.conv1.weight
(256, 256, 3, 3)
conv_backbone.layer3.1.bn1.weight
(256)
conv_backbone.layer3.1.bn1.bias
(256)
conv_backbone.layer3.1.conv2.weight
(256, 256, 3, 3)
conv_backbone.layer3.1.bn2.bias
(256)
conv_backbone.layer3.1.bn2.weight
(256)
conv_backbone.layer4.0.conv1.weight
(512, 256, 3, 3)
conv_backbone.layer4.0.downsample.0.weight
(512, 256, 1, 1)
conv_backbone.layer4.0.bn1.bias
(512)
conv_backbone.layer4.0.bn1.weight
(512)
conv_backbone.layer4.0.downsample.1.weight
(512)
conv_backbone.layer4.0.downsample.1.bias
(512)
conv_backbone.layer4.0.conv2.weight
(512, 512, 3, 3)
conv_backbone.layer4.0.bn2.weight
(512)
conv_backbone.layer4.0.bn2.bias
(512)
conv_backbone.layer4.1.conv1.weight
(512, 512, 3, 3)
conv_backbone.layer4.1.bn1.weight
(512)
conv_backbone.layer4.1.bn1.bias
(512)
conv_backbone.layer4.1.conv2.weight
(512, 512, 3, 3)
conv_backbone.layer4.1.bn2.bias
(512)
conv_backbone.layer4.1.bn2.weight
(512)
student_feature.0.weight
(512, 512)
student_feature.0.bias
(512)
MeanBackward1
TBackward0
ViewBackward0
student_feature.1.weight
(512)
student_feature.1.bias
(512)
AddmmBackward0
NativeBatchNormBackward0
moe_fusion.gate.weight
(2, 512)
moe_fusion.gate.bias
(2)
StackBackward0
moe_fusion.experts.1.0.weight
(1024, 512)
moe_fusion.experts.1.0.bias
(1024)
moe_fusion.shared_expert.0.weight
(1024, 512)
moe_fusion.shared_expert.0.bias
(1024)
SoftmaxBackward0
moe_fusion.experts.1.2.weight
(512, 1024)
moe_fusion.experts.1.2.bias
(512)
moe_fusion.shared_expert.2.weight
(512, 1024)
moe_fusion.shared_expert.2.bias
(512)

Thanks !!!