Advanced Image Processing and Computer Vision Assignment-2

Semester - 8 (Spring)

CS60052

Pranil Dey (20CS30038)

Vikas Vijaykumar Bastewad (20CS10073)

1. Image Classification Problem

The objective of this project is to categorize images from the CIFAR-10 dataset employing two advanced deep learning models: ResNet-50 and EfficientNet-b0.

CIFAR10 dataset:

CIFAR-10 is a widely used dataset in the field of computer vision and machine learning. It consists of **60,000 32x32** color images in **10** classes, with 6,000 images per class. The dataset is divided into **50,000** training images and **10,000** testing images.

Preparation:

i. Dataset Overview: CIFAR-10 comprises **10** classes, each with **6,000** images. These classes include airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks.

ii. Train-Validation Split: The training data is divided into train-val splits with an **80-20** ratio. Specifically, **80%** of the data is allocated for training and **20%** for validation. This split ensures that each class has **80%** of its images in the training set and **20%** in the validation set. This means that each class has **4800** images in the train dataset and **1200** in the test. We combine the default datasets and then split.

Deep Learning Model:

i. Pretrained ResNet-50: ResNet-50, a variant of the **ResNet** architecture, is utilized for training. The **ResNet-50** model is available as a pretrained model in the **PyTorch** library. It consists of **50** layers and has shown exceptional performance in various computer vision tasks.

ii. Pretrained EfficientNet-B0: EfficientNet-B0, another powerful architecture, is also employed for training. Similar to **ResNet-50, EfficientNet-B0** is available as a pretrained model in the **PyTorch** library. The EfficientNet architecture aims to achieve better performance with **fewer parameters** compared to traditional architectures.

Negative Log Likelihood (NLL) Loss:

Negative Log Likelihood (NLL) Loss: The NLL loss is chosen as the loss function for model training. It is well-suited for classification tasks and measures the likelihood of the

model's predictions given the true labels.NLL loss is calculated as the negative log probability of the predicted class for each pixel.

Test Metric:

i. Percentage Accuracy on Test Set: The primary metric for evaluating model performance is the percentage accuracy achieved on the test set. Accuracy reflects the proportion of correctly classified images out of the total number of images in the test set.
The accuracy is calculated by **100*correct/total**.

ii. Model Architecture Comparison: After training both the **ResNet-50** and **EfficientNet-B0** models, their performances will be compared based on their accuracy on the test set. This comparison will provide insights into the effectiveness of each architecture for the **CIFAR-10** dataset. The accuracy of the **ResNet-50** model is **63.812%** and accuracy of the **EfficientNet-b0** model is **60.235%** after running for **50 epochs**.
This shows that even though EfficientNet-b0 is faster than ResNet-50 it gives lower accuracy on the same number of epochs. Though if we give them the same amount of time the efficient net model will give a better accuracy.

Implementation:

Framework: PyTorch is used as the deep learning framework for implementing the models.
Training Procedure:
- Initialize the pretrained ResNet-50 and Efficient-b0 and let it run for about 50 epochs (more epochs would give better accuracy but we have limited time and resources).
- Define the loss function as a combination of NLL loss.
Testing Procedure:
- Evaluate the trained model on the test dataset.
- Calculate accuracy based on performance.

2. Image Segmentation Problem

Our study centers on employing the Cityscapes dataset and leveraging a U-Net architecture to address the image segmentation challenge.

Cityscapes dataset:

The Cityscapes dataset is a popular benchmark dataset for urban scene understanding, particularly semantic and instance-level segmentation tasks. It comprises around **5000** fine annotated images and **20000** coarse annotated images, providing rich annotations for diverse urban scenes.

Data Preparation:

i. Dataset Overview: The dataset includes fine annotated images with detailed pixel-level annotations, as well as coarse annotated images with broader annotations. This diversity allows for comprehensive training and evaluation of segmentation models.

ii. Train-Test Split: From the fine annotated images, 3500 images are randomly selected for training, while 1500 images are reserved for testing. This split ensures a diverse representation of urban scenes in both training and testing sets, enabling robust model evaluation.

Deep Learning Model:

U-Net Architecture: The U-Net architecture is chosen for semantic segmentation. U-Net is well-suited for segmentation tasks due to its ability to capture both global context and fine-grained details through skip connections. It comprises an encoder-decoder architecture with skip connections to preserve spatial information.

Here's a brief overview of the U-Net architecture:

Contracting Path (Encoder):

- The contracting path resembles a traditional CNN and is composed of multiple convolutional and pooling layers.
- Each convolutional layer is followed by a rectified linear unit (ReLU) activation function, which introduces non-linearity into the network.
- The purpose of the contracting path is to capture the context and extract features from the input image.

Expansive Path (Decoder):

- The expansive path is designed to enable precise localization and segmentation of objects in the image.
- It consists of upsampling and concatenation operations, followed by convolutional layers. The upsampling operations are used to increase the spatial resolution of the feature maps.
- Concatenation is performed with feature maps from the contracting path to provide detailed localization information. The expansive path gradually recovers spatial information lost during the contracting path.

Skip Connections:

- One of the key features of the U-Net architecture is the use of skip connections.Skip connections connect the contracting path to the corresponding layers in the expansive path.
- These connections help in preserving spatial information and gradients during training, which can improve segmentation accuracy, especially for objects of varying sizes.

Final Layer:

The final layer of the U-Net architecture typically consists of a convolutional layer with a softmax activation function.
This layer produces the segmentation mask for each pixel in the input image, assigning each pixel to one of the predefined classes.

Loss Function:

i. Negative Log Likelihood (NLL) Loss with DICE Loss: The chosen loss function combines the NLL loss with the DICE loss. The NLL loss measures the negative log likelihood of the model's predictions given the ground truth labels, while the DICE loss

quantifies the similarity between predicted and ground truth segmentation masks. Combining these losses helps optimize the model for accurate segmentation while encouraging precise delineation of object boundaries.

Test Metric:

i. DICE Score on Test Set: The DICE score is computed on the test set to evaluate the overlap between predicted and ground truth segmentation masks. A higher DICE score indicates better segmentation performance and alignment with ground truth annotations. The dice score of the plain UNet model was **0.483563** on running 25 epochs.

Formula: DICE loss is calculated as 1 minus twice the intersection over the union (IoU) of predicted and ground truth masks.

ii. Mean Average Precision (mAP) on Test Set: The mAP metric is calculated on the test set to assess the model's ability to precisely localize objects in the segmented images. It considers both precision and recall, providing a comprehensive evaluation of segmentation quality.

Implementation:

Framework: PyTorch is used as the deep learning framework for implementing the segmentation model.
Training Procedure:
- Initialize the U-Net model architecture.
- Define the loss function as a combination of NLL loss and DICE loss.
- Use stochastic gradient descent (SGD) or Adam optimizer for training.
- Train the model on the training dataset for a number of epochs, in our case 25.

Testing Procedure:
- Evaluate the trained model on the test dataset.
- Calculate DICE score and mAP to assess segmentation performance.