Model Architecture

- **VGGNet Class**: The class defines a convolutional neural network model inspired by the VGG architecture.
    - **Architecture Configuration**: Defined by the `arch` list which includes convolutional layers (with ReLU activations and batch normalization) and max-pooling layers.
    - **Fully Connected Layers**: After the convolutional layers, there are three fully connected layers with dropout regularization to reduce overfitting.

Hyperparameters and Settings

- **Device Configuration**: The model uses a GPU if available, otherwise defaults to CPU.
- **Batch Sizes**: `TRAIN_BATCH_SIZE` is set to 64, and `VAL_BATCH_SIZE` is set to 16.
- **Epochs**: Training runs for 20 epochs.
- **Data Loading**: CIFAR-100 dataset is used, and the data is loaded using `DataLoader` for both training and validation sets with basic transformations (converting to tensor).

Training and Validation

- **Loss Function**: Cross-Entropy Loss is used as the criterion.
- **Optimizer**: Stochastic Gradient Descent (SGD) with momentum, learning rate of 0.01, and weight decay of 5e-4.
- **Learning Rate Scheduler**: `ReduceLROnPlateau` scheduler adjusts the learning rate based on validation loss with a patience of 10.

Training and Validation Loop

- **Epoch Loop**: For each epoch, the model trains on the training set and evaluates on the validation set.
- **Training Step**:
    - The model is set to training mode.
    - For each batch, the model computes the loss, performs backpropagation, and updates the weights.
    - The training loss is accumulated and printed after each epoch.
- **Model Checkpointing**: Model weights are saved every 5 epochs.
- **Validation Step**:
    - The model is set to evaluation mode.
    - Validation loss is computed over the validation set.
    - Training and validation accuracy are calculated using the `calculate_accuracy` function.

  - Learning rate scheduler updates the learning rate based on validation loss.

Results Reporting

- **Loss Metrics**: Prints training and validation loss for each epoch.
- **Accuracy Metrics**: Prints training and validation accuracy for each epoch.

Task-02

Model and Data Preparation

- **Pre-trained Model**:
  - The code uses the VGG-16 model pre-trained on ImageNet. This model is then modified to classify 100 classes (to suit CIFAR-100) by changing the output layer of the classifier to `nn.Linear(4096, 100)`.
  - The model is moved to a GPU if available.
- **Data Transformations**:
  - **Training Transformations**: Includes random cropping, horizontal flipping, and normalization with CIFAR-100-specific mean and standard deviation.
  - **Testing Transformations**: Includes normalization with the same mean and standard deviation as the training data.
- **Data Loading**:
  - **CIFAR-100 Dataset**: The training and testing sets are loaded with the defined transformations.
  - **DataLoader**: The data is loaded using PyTorch's `DataLoader` with appropriate batch sizes and shuffling.

Training and Evaluation Setup

- **Loss Function**: Cross-Entropy Loss is used.
- **Optimizer**: Stochastic Gradient Descent (SGD) with momentum, learning rate of 0.01, and weight decay of 5e-4.

Training Function

- **train_model Function**:
  - Trains the model for a specified number of epochs (50 by default).

    - For each epoch, iterates over the training data, computes the loss, performs backpropagation, and updates the model parameters.
    - Prints the average training loss after each epoch.

Evaluation Function

- **evaluate_model Function**:
    - Evaluates the model on the test data.
    - Computes the loss and accuracy by iterating over the test data without updating the model parameters.
    - Returns the accuracy and average loss.

Training and Evaluation Results

- **Training**:
    - The model is trained for 50 epochs.
    - Average training loss is printed after each epoch to monitor the training progress.
- **Evaluation**:
    - The model's performance is evaluated on both the training and test datasets.
    - Accuracy and loss are computed and printed for both datasets.

Results

After training for 50 epochs, the results are printed:

- **Training Accuracy and Loss**: Indicating how well the model performs on the training data.
- **Test Accuracy and Loss**: Indicating the model's generalization ability on unseen data.

Task-3

Model and Data Preparation

- **Pre-trained Model**:
    - **VGG-16**: Uses the VGG-16 model pre-trained on ImageNet.

    - **Layer Freezing**: All fully connected layers except the last one are frozen to leverage pre-trained weights while training only the last layer for CIFAR-100 classification.
    - **Classifier Modification**: The final layer is modified to have 100 output nodes (one for each class in CIFAR-100).
- **Device Configuration**:
    - The model is transferred to a GPU if available, otherwise, it defaults to CPU.

Data Transformations

- **Training Transformations**:
    - Random cropping and horizontal flipping for data augmentation.
    - Normalization with CIFAR-100-specific mean and standard deviation values.
- **Testing Transformations**:
    - Only normalization is applied to ensure consistent preprocessing during evaluation.

Data Loading

- **CIFAR-100 Dataset**:
    - Training and testing datasets are loaded with the defined transformations.
    - DataLoaders are created for both datasets with appropriate batch sizes and shuffling.

Loss Function and Optimizer

- **Loss Function**: Cross-Entropy Loss, which is standard for classification tasks.
- **Optimizer**: Stochastic Gradient Descent (SGD) with momentum, a learning rate of 0.01, and weight decay of 5e-4.

Training Function

- **train_model Function**:
    - Trains the model for 50 epochs.
    - For each epoch, iterates over the training data, computes the loss, performs backpropagation, and updates the model parameters.
    - Prints the average training loss after each epoch to monitor progress.

Evaluation Function

- **evaluate_model Function**:
    - Evaluates the model on the given data loader (either training or test).
    - Computes the loss and accuracy by iterating over the data without updating the model parameters.

- Returns the accuracy and average loss.

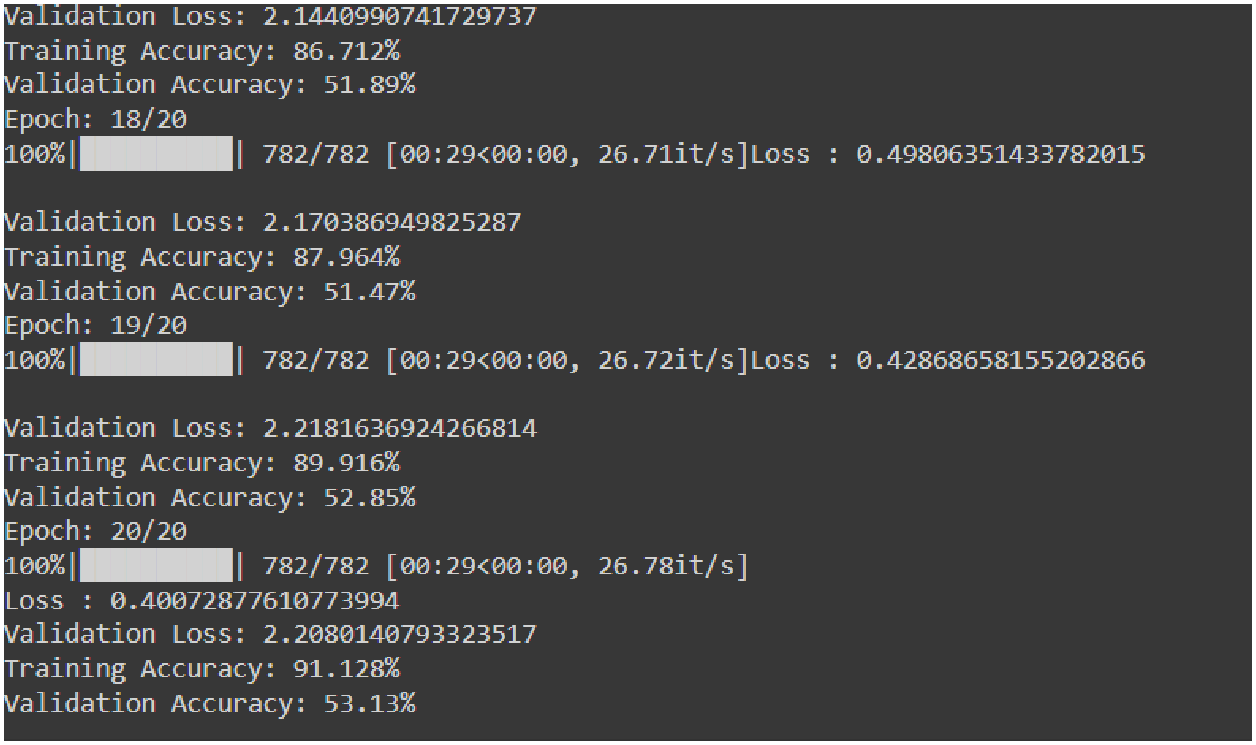
Results

After training for 50 epochs, the code evaluates and prints the following metrics for both the training and test datasets:

- **Training Accuracy and Loss**: Indicates how well the model performs on the training data.
- **Test Accuracy and Loss**: Indicates the model's generalization ability on unseen data.

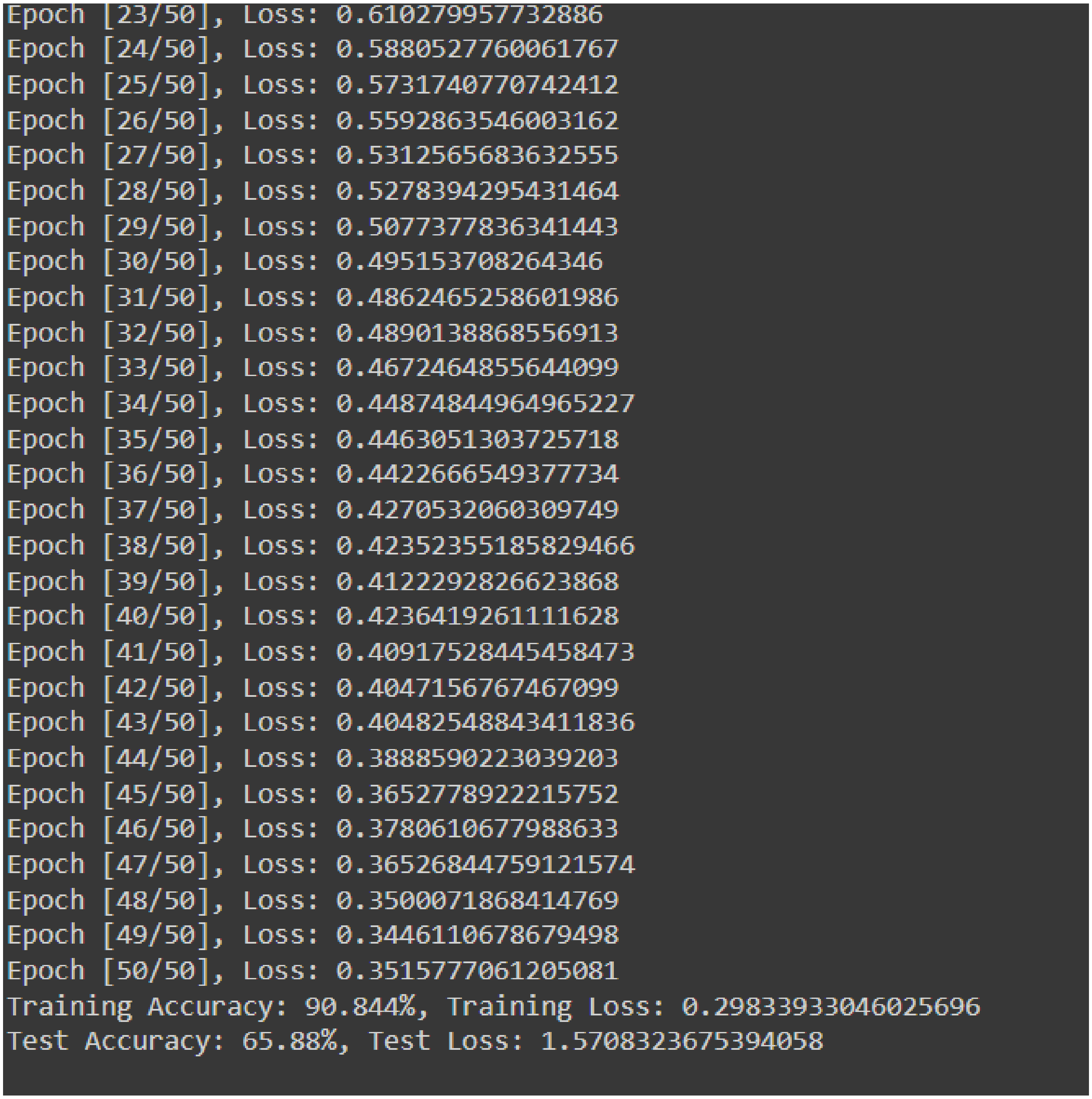
All task results are given below

Task 1 result



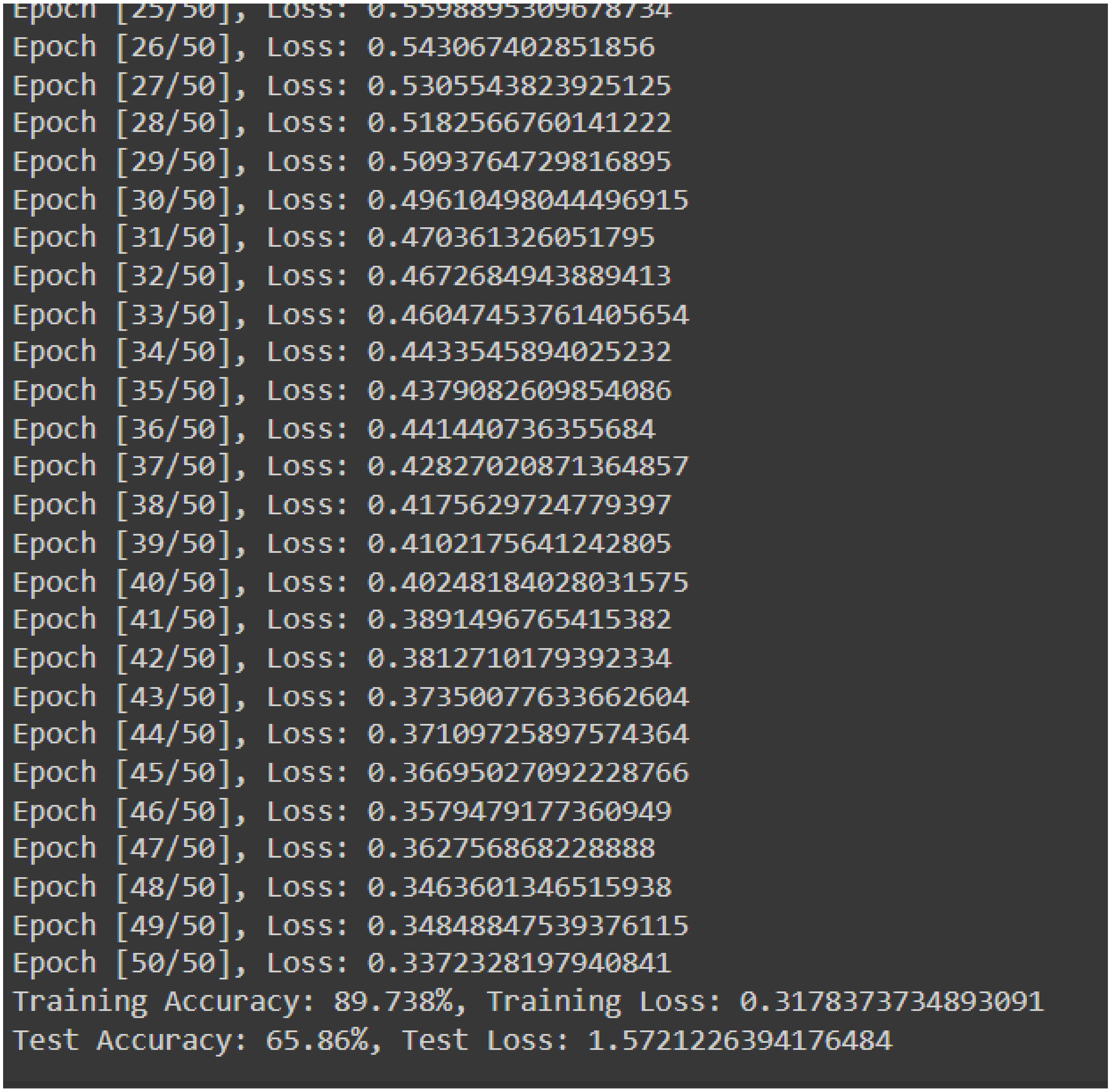
```
Validation Loss: 2.1440990741729737
Training Accuracy: 86.712%
Validation Accuracy: 51.89%
Epoch: 18/20
100%|██████████| 782/782 [00:29<00:00, 26.71it/s]Loss : 0.49806351433782015

Validation Loss: 2.170386949825287
Training Accuracy: 87.964%
Validation Accuracy: 51.47%
Epoch: 19/20
100%|██████████| 782/782 [00:29<00:00, 26.72it/s]Loss : 0.42868658155202866

Validation Loss: 2.2181636924266814
Training Accuracy: 89.916%
Validation Accuracy: 52.85%
Epoch: 20/20
100%|██████████| 782/782 [00:29<00:00, 26.78it/s]
Loss : 0.40072877610773994
Validation Loss: 2.2080140793323517
Training Accuracy: 91.128%
Validation Accuracy: 53.13%
```

Task 2 results



```
Epoch [23/50], Loss: 0.610279957732886
Epoch [24/50], Loss: 0.5880527760061767
Epoch [25/50], Loss: 0.5731740770742412
Epoch [26/50], Loss: 0.5592863546003162
Epoch [27/50], Loss: 0.5312565683632555
Epoch [28/50], Loss: 0.5278394295431464
Epoch [29/50], Loss: 0.5077377836341443
Epoch [30/50], Loss: 0.495153708264346
Epoch [31/50], Loss: 0.4862465258601986
Epoch [32/50], Loss: 0.4890138868556913
Epoch [33/50], Loss: 0.4672464855644099
Epoch [34/50], Loss: 0.44874844964965227
Epoch [35/50], Loss: 0.4463051303725718
Epoch [36/50], Loss: 0.4422666549377734
Epoch [37/50], Loss: 0.4270532060309749
Epoch [38/50], Loss: 0.42352355185829466
Epoch [39/50], Loss: 0.4122292826623868
Epoch [40/50], Loss: 0.4236419261111628
Epoch [41/50], Loss: 0.40917528445458473
Epoch [42/50], Loss: 0.4047156767467099
Epoch [43/50], Loss: 0.40482548843411836
Epoch [44/50], Loss: 0.3888590223039203
Epoch [45/50], Loss: 0.3652778922215752
Epoch [46/50], Loss: 0.3780610677988633
Epoch [47/50], Loss: 0.36526844759121574
Epoch [48/50], Loss: 0.3500071868414769
Epoch [49/50], Loss: 0.3446110678679498
Epoch [50/50], Loss: 0.3515777061205081
Training Accuracy: 90.844%, Training Loss: 0.29833933046025696
Test Accuracy: 65.88%, Test Loss: 1.5708323675394058
```

TASK-03



```
Epoch [25/50], Loss: 0.5598895309678734
Epoch [26/50], Loss: 0.543067402851856
Epoch [27/50], Loss: 0.5305543823925125
Epoch [28/50], Loss: 0.5182566760141222
Epoch [29/50], Loss: 0.5093764729816895
Epoch [30/50], Loss: 0.49610498044496915
Epoch [31/50], Loss: 0.470361326051795
Epoch [32/50], Loss: 0.4672684943889413
Epoch [33/50], Loss: 0.46047453761405654
Epoch [34/50], Loss: 0.4433545894025232
Epoch [35/50], Loss: 0.4379082609854086
Epoch [36/50], Loss: 0.441440736355684
Epoch [37/50], Loss: 0.42827020871364857
Epoch [38/50], Loss: 0.4175629724779397
Epoch [39/50], Loss: 0.4102175641242805
Epoch [40/50], Loss: 0.40248184028031575
Epoch [41/50], Loss: 0.3891496765415382
Epoch [42/50], Loss: 0.3812710179392334
Epoch [43/50], Loss: 0.37350077633662604
Epoch [44/50], Loss: 0.37109725897574364
Epoch [45/50], Loss: 0.36695027092228766
Epoch [46/50], Loss: 0.3579479177360949
Epoch [47/50], Loss: 0.362756868228888
Epoch [48/50], Loss: 0.3463601346515938
Epoch [49/50], Loss: 0.34848847539376115
Epoch [50/50], Loss: 0.3372328197940841
Training Accuracy: 89.738%, Training Loss: 0.3178373734893091
Test Accuracy: 65.86%, Test Loss: 1.5721226394176484
```