How to configure Weights & Biases for you ML project

What is Weights & Biases?

Weights & Biases (wandb from now on) is a platform for AI/ML development. A set of tools it provides can help you keep track of your model's training. This can be very useful if you want to check on how's your model training since you can access wandb remotly from your phone or home computer.

wandb set up

1. Create an account on wandb website.
2. Create a new project and set it's visibility.
   Python_mac
   Image not found or type unknown
   Python_mac
   Image not found or type unknown
3. Activate your Python enviroment.
4. Install wandb package from PyPi:
   `pip install wandb`
5. Login into your wandb account from console:
   `wandb login` Python_mac
   Image not found or type unknown

How to use wandb in your AI/ML project

The simplest use case for wandb is to use it to track your training progress. You can monitor training and validation loss values, test accuracy, and even see what data is being fed into your model during training and validation.

You can do this in four simple steps:

1. Import wandb library
2. Initialize wandb process with your project name, you can specify details abouth the training run, like batch size and learning rate.
3. Log the training information after every epoch.
4. Stop the process after training is finished.

You can refer to this block of code for step 2-4.

```
num_epochs = 25

if wandb and wandb.run is None:
    experiment_dict = {}
    experiment_dict['batch_size']=batch_size
    experiment_dict["learning_rate"]=learning_reate
    experiment_dict["epochs"]=num_epochs

    wandb_run = wandb.init(config=experiment_dict, resume=False,
                           project="Sample_WnB_project",
                           name="Test Run")

#We will execute our training inside of a loop. Each iteration is a new epoch.
for epoch in range(num_epochs):
    print('Epoch:', epoch)

    total_train_loss, model, optimizer = train(model, optimizer, loss_fn, train_dataloader)

    train_loss = total_train_loss/len(train_dataloader)

    print('Train loss: ', train_loss) #average training loss for current epoch

    total_test_loss, total_test_acc = evaluate(model, test_dataloader, loss_fn)

    test_loss = total_test_loss/len(testDataset)
    test_acc = total_test_acc/len(testDataset)

    #Average evaluation loss and evaluation accuracy for this epoch
    print('Test loss: ', test_loss)
    print('Test accuracy: ', test_acc)

    wandb.log({"acc": test_acc, "train_loss": train_loss, "test_loss": test_loss})

wandb.finish()
```

You can get the full code from:

https://git.arts.ac.uk/ipavlov/WikiMisc/blob/main/SimpleCNN_tweak.ipynb

Data can be found here

Revision #7
Created 11 April 2024 09:14:00 by Ilia Pavlov

Updated 29 April 2024 22:22:38 by Ilia Pavlov