DS 203 Deep Learning Assignment 2021

Deadline: November 6, 2021 11:59 PM

Go through these instructions carefully to avoid any confusions.

1 Tutorials

Some relevant tutorials to familiarise yourself to pytorch:

- https://pytorch.org/tutorials/beginner/pytorch_with_examples.html
- https://youtu.be/6SlgtELqOWc?t=3077
- Pytorch Documentation : https://pytorch.org/docs/stable/index.html

Jargon you should be comfortable with : Tensors, Optimiser, Models, Autograd or Backward Pass, Activations, ...

2 XNOR

2.1 Data Generation

Firstly generate 10K points in $[1, 1] \times [1, 1]$ using np.random.seed(0)

data = 2*np.random.uniform(size=(10000, 2)) - 1

Label points according to their quadrant. First and Third quadrant being labelled 1 and Second and Fourth quadrant being labelled 0.

2.2 Excercises

1. Write a Dataset module for the XOR data (3 sets train, validation and test,70:15:15 respectively)
    - Inherit from `torch.utils.data.Dataset`

- Define `__init__`, `__getitem__`, `__len__`

2. Define the Dataloader with batchsize of 16
   - Go through all the arguments of dataloader like `drop_last`,`shuffle`,`batch_size`
3. Define the Neural Network Model
   - Inherit from `torch.nn.module`
   - 2 Linear layers, ReLU activation after first layer, single numeric output
   - Variable hidden layer size as input to model (4 as default hidden layer size)
4. Define loss function as `torch.nn.CrossEntropyLoss`
   - Explore other possible loss functions as `LogSoftmax` + `NLLLoss` vs `CrossEntropyLoss` and `MSELoss`
5. Optimizer: Use SGD optimizer with learning rate 1e-3
   - See `zero_grad()`, `step()`
6. Write the main training loop and validation loops for n epochs
   - See `loss.backward()`, `model.forward()`
   - Use `model.train()`, `model.eval()`,`torch.no_grad()`
   - Validate for each epoch, run for 100 epochs
7. Plot the following:
   (a) Training and Validation loss vs epoch in a single plot
   (b) Training and Validation accuracy vs epoch in a single plot
   (c) Best Validation loss vs Hidden Layer size (use hidden size to be (2,4,6,8,10))
   (d) Best Validation loss vs learning rate used (use learning rates in (1e-5,1e-4,1e-3,1e-2,1e-1)) for max number of 20 epochs
   (e) Plot test set predicted labels for best validation model. Report accuracy and loss for the same.

3 Fine-tune Distilbert on IMDB Reviews Dataset

1. Download dataset from
   `http://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz`
2. Split it into train and validation in the ratio 80:20
3. Install transformers library of huggingface
4. From transformers import DistilBertTokenizerFast tokenizer of pretrained model "distilbert-base-uncased"
5. Tokenize the text in the dataset using this tokenizer
6. Inherit from `torch.utils.data.Dataset`
   - Define `__init__`,`__getitem__`,`__len__`
7. Import `DistilBertModel` From `transformers` library
8. Add a dropout layer followed by Linear Layer
9. Use an appropriate loss function and optimizer for the classification task and finetune the model on the training data
10. Plot train loss vs epoch, validation loss vs epoch, validation accuracy vs epoch
    Note that the use of `DistilBertForSequenceClassification` is not allowed. You can use `DistilBertTokenizer` and `DistilBertModel`

4 Submission Details

Create 2 notebooks for the assignment, namely `xnor.ipynb` and `distilbert.ipynb`. For submission download these notebooks as `.pdf`, `.ipynb` and `.py` with code and plots intermixed (make sure to generate all the plots and report all numbers instead of just the providing code for the same). Your submission should look something like this :

1. rollnumber

(a) xnor.pdf

(b) distilbert.pdf

(c) xnor.py

(d) distilbert.py

(e) xnor.ipynb

(f) distilbert.ipynb

Zip or tar.gz this folder and name it `rollnumber.zip` or `rollnumber.tar.gz` appropriately.