PNN Framework - 2023

Anton Vlasjuk

Explanation for the implemented framework - PNN2023

Chair for Computer Science VI
University of Würzburg
Programming with Neural Networks

Frank Puppe, Amar Hekalo, Christian Hauptmann, Adrian Krenzer
Würzburg, September 7, 2023

1. Structure and Functionalities

The source code has the following structure and functionalities:

- src
  - data/MNIST/raw
    - .*?(.gz)?: all the raw data downloaded via pytorch
  - layer
    - ActivationLayer.py: Can be initialised with the following activation functions (by passing the string mentioned in the brackets):
      - ReLu ('relu')
      - Sigmoid ('sigmoid')
      - Tanh ('tanh')
    - Conv2DLayer.py: Takes four initial integers:
      - x - the size of the rows
      - y - the size of the columns
      - channels - the number of channels
      - numFilters - the total number of filters to be applied

      Each filter is initialised according to a uniform distribution between -1.0f and 1.0f and a bias of 0.
    - FlattenLayer.py: A utility layer that takes the expected incoming shape for initialisation; flattens incoming tensors of the given shape into a one-dimensional vector (or a two-dimensional one with shape (1, n)).
    - FullyConnectedLayer.py: Takes an integer each for the input's shape and the wanted output's shape. The weight matrix' values are initialised randomly (according to a uniform distribution) between -1.0f and 1.0f. The bias is initialised with zeros.
    - InputLayer.py: Expects a list (or any stacking of lists) of floats and turns it into a corresponding list of Tensors.
    - Layer.py: Abstract class that defines all needed functions of each layer.
    - LossLayer.py: Incorporates two types of losses:
      - Cross entropy loss
      - Mean squared error loss
    - Pooling2DLayer.py: Takes the shape of the filter and the respective stride for initialisation. Aggregates according to the filter and stride in two possible ways:
      - Taking the maximum of the overlapped filter
      - Taking the average of the overlapped filter
    - SoftmaxLayer.py: The usual softmax.

- model
  - Tensor.py: Describes a layer's values, deltas, and shape.
- network
  - Network.py: Describes the neural network. After initialising an empty network, the aforementioned layers can be added progressively to create a fully functional network. The corresponding input and label(s) have to be passed to the forward function to get a result/loss for an input. During this process, a cache is created which saves the results of the forward pass. The cache runs the respective updates on each layer when running the backward pass (which needs a learning rate). Aside from that, there is an optional debug flag (on initialisation) to enable prints of the state of the net during the respective passes.
  - SGDTrainer.py: Applies the standard gradient descent to a neural network with the given data. The hyperparameters (e.g. number of epochs, learning rate, etc.) can be adjusted on initialisation. Furthermore, just like the neural net, there is a debug flag to enable print statements and another loss flag that prints the current loss at a rate given by the loss_batch parameter (the loss flag has a higher priority than the debug flag). (There is also another flag, the dev flag, which enables an early break if the improvement from the last epoch is too little.)
  - UpdateMechanism.py: Enum taken from the lecture.
- Dummy.py: Messing around with neural nets and layers.
- LoadDataMNIST.py: Loads the MNIST dataset via torch(vision). There is a variant each for the CNN and the FNN as they work with different dimensionalities.
- MNIST-CNN.py: Runs a certain CNN on the MNIST dataset and evaluates it via accuracy; usually runs on one certain config (1 epoch, 0.01 learning rate). Due to the long training time, there are some optional functions:
  - pickle_run: Loads an already trained net and runs through the evaluation of the net.
  - save_current_net: Saves one trained net that has the best performance among all other ones after the complete training procedure.
- MNIST-FFNN.py: Runs a certain FFNN on the MNIST dataset and evaluates it via accuracy; has some configs to run multiple times which should be adjusted one's need in the first few lines of the main function.
- MNIST-Torch.py: Torch playground to create nets and run them through MNIST.

- test
  - test_activation.py: Unit tests for the relu and sigmoid activation layer as well as the softmax layer.
  - test_layer.py: Unit tests for the fully connected, convolutional, and pool layer.
  - test_losses.py: Unit tests for the loss layers, e.g. cross entropy and mean squared error loss.

2. Example Usage

a) FFNN

```
# Assumption of correctly imported classes etc.



# initialises the standard gradient descent trainer with certain
# hyperparameters and the debug flag
# --> forces print statements during the network optimisation
sgd = SGDTrainer(learningRate=1, amountEpochs=3, debug=True, loss=False)


# initialising an empty network and adding consecutive layers
# one after another
net = Network()
# fully connected layer of 3 rows and 3 columns
net.addLayer(FullyConnectedLayer(3, 3))
# activation layer sigmoid by passing the according string
net.addLayer(ActivationLayer('sigmoid'))
# fully connected layer of 3 rows and 2 columns
net.addLayer(FullyConnectedLayer(3, 2))
# softmax layer
net.addLayer(SoftmaxLayer())
# cross entropy loss
net.addLayer(CrossEntropyLoss())


# the data the net will be trained on with sgd
# only consists of one pair (input, labels)
data = [([0.4183, 0.5209, 0.0291], [0.7095, 0.0942])]


# trains the net with the sgd trainer
sgd.optimize(net, data, [], [])
```

b) CNN

```
# Assumption of correctly imported classes etc.



# initialises the standard gradient descent trainer with certain
# hyperparameters and the debug flag
# --> forces print statements during the network optimisation
sgd = SGDTrainer(learningRate=0.01, amountEpochs=1, debug=True, loss=False)


# initialising an empty network and adding consecutive layers
# one after another
net = Network()
# convolutional layer of 9 rows, 9 columns, and 1 channel
# while using 4 filters
net.addLayer(Conv2DLayer(9, 9, 1, 4))
# activation layer relu by passing the according string
net.addLayer(ActivationLayer('relu'))
# pooling layer that uses averaging as aggregation mechanism
# filter of size 4x4 and a stride of 4x4
net.addLayer(AvgPooling(filter_shape=(4, 4), stride=(4, 4)))
# flattening layer to transform the result into 1d
net.addLayer(FlattenLayer(input_shape=(4, 5, 5)))
# fully connected layer of 100 rows and 10 columns
net.addLayer(FullyConnectedLayer(100, 10))
# softmax layer
net.addLayer(SoftmaxLayer())
# cross entropy loss
net.addLayer(CrossEntropyLoss())


# the data the net will be trained on with sgd
# only consists of one pair (input, labels)
data = [(np.array([[[0.        , 0.        , 0.        , 0.        , 0.
                                             ,
                     0.        , 0.        , 0.        , 0.        , 0.
                                                                  ,
                     0.        , 0.        , 0.        , 0.        , 0.
                                                                  ,
                     0.        , 0.        , 0.        , 0.        , 0.
                                                                  ,
                     0.        , 0.        , 0.        , 0.        , 0.
                                                                  ,
                     0.        , 0.        , 0.        ],
                    [0.        , 0.        , 0.        , 0.        , 0.
                                                                 ,
                     0.        , 0.        , 0.        , 0.        , 0.
                                                                  ,
                     0.        , 0.        , 0.        , 0.        , 0.
                                                                  ,
```

```
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         ],
[0.         , 0.         , 0.         , 0.          , 0.
                                                 ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         ],
[0.         , 0.         , 0.         , 0.          , 0.
                                                 ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         ],
[0.         , 0.         , 0.         , 0.          , 0.
                                                 ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         ],
[0.         , 0.         , 0.         , 0.          , 0.
                                                 ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         , 0.          , 0.
                                                   ,
 0.         , 0.         , 0.         ],
[0.         , 0.         , 0.         , 0.          , 0.
                                                 ,
```

```
 0.        , 0.        , 0.        , 0.        , 0.
                                                  ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                  ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                  ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                  ,
 0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.32941177, 0.72549021, 0.62352943, 0.
                                                  59215689,
 0.23529412, 0.14117648, 0.        , 0.        , 0.
                                                  ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                  ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                  ,
 0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.87058824, 0.99607843, 0.99607843, 0.
                                                  99607843,
 0.99607843, 0.94509804, 0.7764706 , 0.7764706 , 0.
                                                  7764706 ,
 0.7764706 , 0.7764706 , 0.7764706 , 0.7764706 , 0.
                                                  7764706 ,
 0.66666669, 0.20392157, 0.        , 0.        , 0.
                                                  ,
 0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.26274511, 0.44705883, 0.28235295, 0.
                                                  44705883,
 0.63921571, 0.89019608, 0.99607843, 0.88235295, 0.
                                                  99607843,
 0.99607843, 0.99607843, 0.98039216, 0.89803922, 0.
                                                  99607843,
 0.99607843, 0.54901963, 0.        , 0.        , 0.
                                                  ,
 0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                  ,
 0.        , 0.06666667, 0.25882354, 0.05490196, 0.
                                                  26274511,
 0.26274511, 0.26274511, 0.23137255, 0.08235294, 0.
                                                  9254902 ,
```

```
0.99607843, 0.41568628, 0.        , 0.        , 0.
                                                   ,
0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                                  ,
0.        , 0.        , 0.        , 0.        , 0.
                                                   ,
0.        , 0.        , 0.        , 0.        , 0.
                                                   ,
0.        , 0.        , 0.        , 0.32549021, 0.
                                           99215686,
0.81960785, 0.07058824, 0.        , 0.        , 0.
                                                   ,
0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                                  ,
0.        , 0.        , 0.        , 0.        , 0.
                                                   ,
0.        , 0.        , 0.        , 0.        , 0.
                                                   ,
0.        , 0.        , 0.08627451, 0.9137255 , 1.
                                                   ,
0.32549021, 0.        , 0.        , 0.        , 0.
                                                   ,
0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                                  ,
0.        , 0.        , 0.        , 0.        , 0.
                                                   ,
0.        , 0.        , 0.        , 0.        , 0.
                                                   ,
0.        , 0.        , 0.50588238, 0.99607843, 0.
                                           93333334,
0.17254902, 0.        , 0.        , 0.        , 0.
                                                   ,
0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                                  ,
0.        , 0.        , 0.        , 0.        , 0.
                                                   ,
0.        , 0.        , 0.        , 0.        , 0.
                                                   ,
0.        , 0.23137255, 0.97647059, 0.99607843, 0.
                                           24313726,
0.        , 0.        , 0.        , 0.        , 0.
                                                   ,
0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                                  ,
0.        , 0.        , 0.        , 0.        , 0.
                                                   ,
```

```
 0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.52156866, 0.99607843, 0.73333335, 0.
                                                 01960784,
 0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                               ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.03529412, 0.80392158, 0.97254902, 0.22745098, 0.
                                                ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                               ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.49411765, 0.99607843, 0.71372551, 0.        , 0.
                                                ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                               ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                 29411766,
 0.98431373, 0.94117647, 0.22352941, 0.        , 0.
                                                ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                               ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        , 0.07450981, 0.
                                                 86666667,
 0.99607843, 0.65098041, 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        ],
```

```
[0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.        , 0.01176471, 0.79607844, 0.
                                             99607843,
 0.85882354, 0.13725491, 0.        , 0.        , 0.
                                                 ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.        , 0.14901961, 0.99607843, 0.
                                             99607843,
 0.3019608 , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.12156863, 0.87843138, 0.99607843, 0.
                                             4509804 ,
 0.00392157, 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.52156866, 0.99607843, 0.99607843, 0.
                                             20392157,
 0.        , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.        , 0.        , 0.        ],
[0.        , 0.        , 0.        , 0.        , 0.
                                                ,
 0.        , 0.        , 0.        , 0.        , 0.
                                                 ,
 0.23921569, 0.94901961, 0.99607843, 0.99607843, 0.
                                             20392157,
```

```
                                0.        , 0.        , 0.        , 0.        , 0.
                                                                           ,
                                0.        , 0.        , 0.        , 0.        , 0.
                                                                           ,
                                0.        , 0.        , 0.        ],
                               [0.        , 0.        , 0.        , 0.        , 0.
                                                                          ,
                                0.        , 0.        , 0.        , 0.        , 0.
                                                                           ,
                                0.47450981, 0.99607843, 0.99607843, 0.85882354, 0.
                                                                    15686275,
                                0.        , 0.        , 0.        , 0.        , 0.
                                                                           ,
                                0.        , 0.        , 0.        , 0.        , 0.
                                                                           ,
                                0.        , 0.        , 0.        ],
                               [0.        , 0.        , 0.        , 0.        , 0.
                                                                          ,
                                0.        , 0.        , 0.        , 0.        , 0.
                                                                           ,
                                0.47450981, 0.99607843, 0.81176472, 0.07058824, 0.
                                                                           ,
                                0.        , 0.        , 0.        , 0.        , 0.
                                                                           ,
                                0.        , 0.        , 0.        , 0.        , 0.
                                                                           ,
                                0.        , 0.        , 0.        ],
                               [0.        , 0.        , 0.        , 0.        , 0.
                                                                          ,
                                0.        , 0.        , 0.        , 0.        , 0.
                                                                           ,
                                0.        , 0.        , 0.        , 0.        , 0.
                                                                           ,
                                0.        , 0.        , 0.        , 0.        , 0.
                                                                           ,
                                0.        , 0.        , 0.        , 0.        , 0.
                                                                           ,
                                0.        , 0.        , 0.        ]]])
         , [0, 0, 0, 0, 0, 0, 0, 1, 0, 0])]


# trains the net with the sgd trainer
sgd.optimize(net, data, [], [])
```