Deep neural network processing of DEER data

hands-on session instructions

If you plan to submit assessed work, **questions highlighted in blue** must be answered in your report; there is no target word count – be reasonable and succinct. If you are not required to submit a report, simply explore those questions until you find satisfactory answers.

1. Before we begin, a few stage setting operations must be performed:

(a) On the device that you are using during the tutorial, Install *VNC Viewer* from https://www.realvnc.com/en/connect/download/viewer

(b) Sign into *NMRBox* portal here:

https://nmrbox.nmrhub.org/signin

and then connect to one of the *NMRBox* instances listed here:

https://nmrbox.nmrhub.org/hardware?filter_vms=2023.35&filter_vms=2023.37

It is a good idea to pick an instance that shows low utilisation of CPU and GPU.

(c) Go to upper left corner menu of the Linux instance that you get, then to Settings / Display and increase the resolution to a maximum that fits on the screen of your device.

(d) Open a terminal window and start *Matlab* by typing `matlab` and pressing Enter. Make sure that the version of *Matlab* listed in the splash screen is R2023a Update 4 or newer. In the Matlab command prompt, type `activate_spinach`.

(e) Make sure that no errors appear. If there are errors, follow the manual *Spinach 2.8* download and installation instructions here:

(A) Google up and explain the concept of PATH in operating systems and large packages such as *Matlab*. Why is it essential to get paths right when installing anything? [5 marks]

In the tutorial below, we will trust *Spinach* to have all the complicated quantum mechanics implemented already. In this workshop, we simply tell *Spinach* what to do. If you would like further information on the technicalities, look into the source code of the functions we are going to call, and read the three papers in the workshop directory on *NMRBox*.

2. To obtain default *DEERNet* training database parameters, type:

```
parameters=library_dd()
```

Matlab will report a number of parameters to the console. To see their physical meaning, type

```
edit library_dd
```

and take a look at the code comments. Further explanations are provided in the *Materials and Methods* section of *deernet_paper_a.pdf* file located in the workshop directory on *NMRBox*.

(B) Given what you have learned in the lecture about the neural network training process, how many distinct simulated question-answer pairs do you expect to be necessary for unambiguous training of a fully connected neural network with 10^6 parameters? [10 marks]

The network we are going to train is much smaller; we will only need about 50,000 question-answer pairs with questions (time domain) and answers (distance domain) 256 points long. We will also restrict the network to DEER experiment and not train it to process data from a related RIDME

experiment. We therefore set the following additional parameters (type these lines into the *Matlab* console and press Enter to run them):

```
parameters.ntraces=50000;
parameters.expt='deer';
parameters.np_time=256;
parameters.np_dist=256;
```

(C) Google up and summarise the essential difference between DEER and RIDME experiments in EPR spectroscopy. Which one is more appropriate for a system containing two nitroxide radicals? Which one is best for a system with a rapidly relaxing electron on a metal ion? Why? [10 marks]

The following commands will start *Matlab*'s parallel pool (this takes about a minute) and then generate the training database:

```
parpool(4)
library=deer_lib_gen([],parameters)
```

Using only four parallel workers makes sure that you and your friends do not overload and crash any *NMRBox* nodes. If *Matlab* complains about not being able to write a cache file, point the file browser (on the left of *Matlab* window) to your home directory and run the second command again.

Inspect the resulting library object. It contains a vector of time axis ticks for the time-domain data, a vector of distance axis ticks for the distance distribution data, and 50000 instances of DEER data.

(D) Plot a few of them (use *Matlab*'s `plot` command, read the manual to find out how to use it) to get an idea of the time and distance scale of DEER experiment. [5 marks]

(E) Type `whos` and take a look at the amount of memory used by the library array. How much memory would a library with 10^{10} DEER traces consume? How did the authors of DEERNet work around this storage problem? [10 marks]

(F) Google up and describe the physical origin of the oscillation frequency of the time-domain DEER signal. Why is the signal so short? Enclose a plot of one time-domain and its corresponding distance-domain DEER dataset with your report. [10 marks]

3. To create an untrained fully connected two-layer neural network, paste the following command into *Matlab*'s command line:

```
layers=[featureInputLayer(256);
        fullyConnectedLayer(128);
        batchNormalizationLayer();
        softplusLayer('Name','SP1');
        fullyConnectedLayer(256);
        batchNormalizationLayer();
        softplusLayer('Name','SP2');
        renormLayer();
        regressionLayer()]
```

(G) Google up and describe briefly the mathematical nature of a fully connected layer and a batch normalisation layer. Renormalisation layer makes sure that the signal integrates to 1. Why is this necessary for DEER data? [15 marks]

4. To train the network, paste the following commands into *Matlab*'s command line:

```
train_opts=trainingOptions('adam','MaxEpochs',100,'Verbose',true,...
                           'MiniBatchSize',256,'ExecutionEnvironment','cpu',...
                           'Plots','training-progress');

[net,train_stats]=trainNetwork(library.deer_noisy_lib',...
                               library.dist_distr_lib',...
                               layers,train_opts);
```

Watch the training process. Repeat the above steps with:

(a) a larger network (insert one or more additional [fully connected]-[batch norm]-[softplus] triads after the first one);
(b) a larger training database;
(c) a larger batch size;
(d) the SGDM training algorithm (see the manual for `trainingOptions`).

You may find it convenient to place all of the above commands into a script and run the script. Observe the changes in the resulting final root-mean-square error (the lower, the better).

(H) Write up and rationalise your observations. Google up and describe the difference between conventional and stochastic gradient descent. What is the difference between SGDM and ADAM methods? What is batch size and how does it influence the training? [20 marks]

5. To use the resulting neural network for data processing, use `predict` command. For example, to process the first dataset in the library:

```
ans=predict(net,library.deer_noisy_lib(:,1)')
```

(I) Plot a few network predictions (`ans` variable from above, use "`hold on`" command to overlay the next plot on the previous one) and compare them to the known right answers from the library (contained in the corresponding columns of the `library.dist_distr_lib` array). [10 marks]

(J) Compare the performance of your network with one of the highly trained networks supplied with *Spinach*. Use one of the networks from the following location:

`/usr/software/spinach/experiments/deernet/netset/net_distan_dd/deer-(256)-256-512`

Its output is 512 points long; use `linspace` command to make a distance axis manually. [20 marks]

(K) From the run time of your training process and the memory utilisation of the training library array, estimate the computing resources required to train one production-grade DEERNet network (six FC-BN-SP triads, 10^{10} question-answer pairs in the database). [10 marks]

6. Run a few examples from `~/spinach/examples/deernet` directory and inspect the output. Read the three papers from the workshop folder.