Getting Started

How to set up your first network

- Set up your first Network

Set up your first Network

With Deepr it is quite easy to start making your own neural network model. This chapter will show you how to make your first neural network using Deepr. The goal of this project will be to sum two floating point numbers, lets get started!

First we have to install DeeprAI, navigate to a comand line and enter as follows:

```
pip install deeprai
```

Now that we have downloaded the package, we will be able to start making our neural network. For the next step make a new python file and write the following:

```
from deeprai import models

network = models.FeedForward()
```

This imports the models folder from Deepr, which is the standard practice for making a model. Then we create the network object which is initializing our FeedForward class. Our next step is to set up our network's dense layers. Because our goal is to sum two floating point numbers, we will have the layer structure of 2,5,1 neurons (1 inputs layer with 2 neurons, 1 hidden layer with 5 neurons , and 1 output layer with 1 neuron). Lets make it!

```
network.add_dense(2)

network.add_dense(5, activation='linear')

network.add_dense(1, activation='linear')
```

Great! we set the activation function to a linear function. The activation function on default will be sigmoid if nothing is specified. If you don't know what that is, don't worry about it. If you do know, there are more advanced options. Before we can train our neural network we need to set up training data it can learn from. Navigate to the top of you file and add these to the import list:

```
# ...
import numpy as np
import random
```

We are importing numpy for its array data-type, and we are importing random to generate random training data. Lets generate some training data. We need two arrays, one to be a 2D list of two floating point numbers, the second will have the summed values. For example Array 1:[[0.2, 0.6],[0.1, 0.3]], Array 2: [[.8], [.4]]. Now that we have that inplace lets build it!, Enter this below the import statements.

```
inputs = np.array([[random.random()/2 for _ in range(2)] for _ in range(5000)])
expected = np.array([[i[0] + i[1]] for i in inputs])
```

Now we have all the data, we have all the things we need to train the network. Lets train it. Navigate back down to the bottom of the file and add this line:

```
network.train_model(train_inputs=inputs,train_targets=expected,test_inputs=inputs,test_targets=expected,
epochs=20)
```

Awsome! We set up connected our inputs and expected values into the neural network, we also set the epochs to 20 (how many times the neural network will run through the training data). One last thing, if we want to test out our network we can ask it a question based on its training data. Lets quiz our network with a simple addition problem and output the results:

```
output = network.run(np.array([.3,.1]))
print(output)
```

Now we can click run and see what happens!

(Note: In PyCharm, in run config check "Emulate terminal in output console" if you want the correct loading animations.)

Here is the full code up to this point:

```
import numpy as np
import random
import deeprai.models as model

inputs = np.array([[random.random()/2 for _ in range(2)] for _ in range(3000)])
expected = np.array([[i[0] + i[1]] for i in inputs])

network = model.FeedForward()

network.add_dense(2)

network.add_dense(5, activation='linear')

network.add_dense(1, activation='linear')

network.train_model(train_inputs=inputs,train_targets=expected,test_inputs=inputs,test_targets=expected,epo
chs=20)

output = network.run(np.array([.3,.1]))
```

```
print(output)
```

This was pretty easy, in the next tutorial we will do something a bit more difficult. We will train on the mnist dataset!