Connectionist Language Processing — Tutorial #1

Date: Thursday, 18 April 2019 (due at start of next lab)

Name:

1 Prerequisites

For this course, we will be using MESH, our in-house neural network simulator. On the CoLi server, you can find MESH in: `/proj/courses/clp/bin/mesh`. For a better experience, however, it is recommended to add the following alias to the `.bashrc` file in your home directory: `alias mesh="rlwrap /proj/courses/clp/bin/mesh"`, using your favorite editor (e.g., NANO).

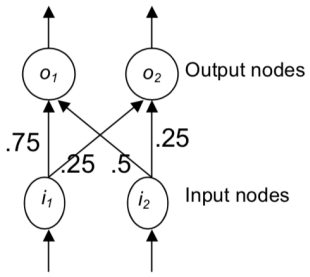
Logout and back in, and test whether it worked by starting MESH from your home directory. This should show you:

```
$ mesh
Mesh, version 0.1.0: http://hbrouwer.github.io/mesh/
  [:>
```

For future reference, the command '`help`' provides help on a variety of topics. For now, just type '`quit`' to exit.

2 Activation propagation using paper and pencil

Copy the files in `/proj/courses/clp/Tutorial1/` to your home directory. The `NetInput.*` files specify the network shown in the figure. Assume a linear activation function on the output nodes. Given the weights shown in the figure, and the input patterns defined in `NetInput.set`, fill out the table below:



I_1	I_2	O_1	O_2

3 Activation propagation using MESH

We can now verify our solutions using MESH. Load the network by typing:

```
$ mesh NetInput.mesh
Mesh, version 0.1.0: http://hbrouwer.github.io/mesh/
...
> Loaded file   [ NetInput.mesh ]
  [NetInput:data>
```

Next, load the predefined weights using the '`loadWeights NetInput.weights`' command, and type '`showMatrix weights input ouput`' to see the effect. Do the weights match those in the figure?

You can now propagate the activation using MESH. The command '`items`' lists all items in the current example set, and the command '`testItem`' either takes the number (first column) or name (second column) of an item as argument. Do the numbers match the ones you calculated by hand?

Explore the network some further, using the commands '`groups`' and '`projections`' (and if you want even more information, try '`inspect`'). Which group was omitted in the figure? Study the weights between this group and its recipient. Was it legitimate to omit in the figure? Finally, type '`weightStats`'. Do these numbers make sense?

4 Training the network

Let us now try to improve the network. Type '`init`' to reset the weights, and use '`weightStats`' to confirm. Now type '`train`', and study the output. What happens to the error? How many times has the network been exposed to each example pattern?

Look at the different items. Has the network improved?

Run '`train`' again. And now? Repeat one more time. And now?