CODATA – RDA

Data Schools

Artificial Neural Networks 3 : Teaching Neural Networks

Roger Barlow `Roger.Barlow@hud.ac.uk`

Advice on Teaching Neural Networks

This is advice, not instructions

- I don't know your students
  - their prior knowledge
  - their ages
  - their homogeneity
  - their attitude
  - the constraints they're under
  - their equipment
- I don't know the framework: school, college, evening classes...
- I don't know how long you've got: a couple of lessons or a whole course.
- I don't know about any assessment requirements.

Make your own choices, don't just follow me – or anyone else

Get them doing it!

Don't just talk at them about what neural networks are and what they can do
Do get them running a real network, training it and using it

- They learn by doing.
- They'll engage and enjoy it
- It empowers them. They learn an ANN is something they can run and use - not something for remote specialists
- It provides a clear, direct and motivational way of assessment, if needed

Learning Neural Networks is like learning to ride a bicycle: a little instruction and a lot of experience.

Being able to teach by doing may seem unusual, but it's a great opportunity

Don't be afraid to leave stuff out

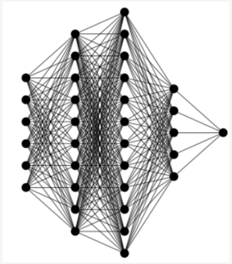
Better to thoroughly learn a little than to fail to learn a lot
When planning your course, don't feel compelled to include everything you know

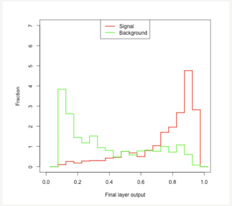
Confession! When teaching you about neural networks I omitted:

1. The threshold in the sigmoid function. $U_i = \frac{1}{1+e^{T_i - \sum w_{ij} U_j}}$
2. Neural Network Regression
3. Dedicated NN hardware
4. NNs as pipelines

What are the essentials?

Network topology
Nodes
Inputs and outputs and hidden layers
Each line is a weight
Data moving through the network





Outputs for Signal and Background (or whatever)
Training: start with random weights
Full overlap to begin with
Pull apart as training continues

Should they write their own or use a package?

Very much depends on who you're teaching and what you're trying to do
Teaching someone to drive a car can involve engine, steering, electrics.... or just: this is the accelerator, this is the brake.

Make a careful choice. Be aware of dangers

Writing their own
Can get bogged down in compiler syntax
Will not be as performant
Will not have as many features

Using a package
Can become a black box
Little to go on if something doesn't work as expected
Over-emphasis on weights

If you are teaching programming, writing an ANN program is a great example.

What language should you use for Neural Networks?

R, C, C++, Python, Matlab, Java, Fortran...

It absolutely doesn't matter

ANNs fit neatly into pretty much any language (except COBOL, maybe)

Work with the standard language used by your students

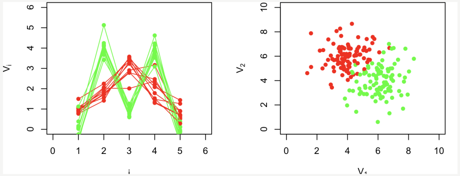
If they have no standard language, use your favourite

What about TensorFlow^{TM} and SageMaker^{TM}?

Very powerful but complicated. Too advanced for education?

Data Tips

1. Use data that is relevant to the students' specialisation, if they have that in common. Don't feel you have to follow standard examples.
2. Simulated (fake) data may be better than real (messy) data for teaching purposes. Don't feel guilty about using it.
3. Use data that can be visualised. Either by displaying or by plotting. Numeric is better than logical/categorical



4. Don't be tempted to use students' personal data. It would make it relevant – but you're going to upset someone.

Final Thoughts

- This is not hard! Even though it may look daunting to a newcomer. You learnt about ANNs in a single day - going from novices to giving presentations.
- Tasks that result in plots are generally more satisfying than tasks that result in numbers. ANNs present opportunities you can exploit.
- Your students will enjoy it - ANNs are still new, and they can do meaningful work with a simple PC
- Good luck!