Connectionist Computing
COMP 30230/41390

Gianluca Pollastri

office: E0.95, Science East.

email: gianluca.pollastri@ucd.ie

Credits

- **Geoffrey Hinton, University of Toronto.**
  - **borrowed some of his slides for "Neural Networks" and "Computation in Neural Networks" courses.**

- **Ronan Reilly, NUI Maynooth.**
  - **slides from his CS4018.**

- **Paolo Frasconi, University of Florence.**
  - **slides from tutorial on Machine Learning for structured domains.**

Lecture notes on Brightspace

- **Strictly confidential...**
- **Slim PDF version will be uploaded later, typically the same day as the lecture.**
- **If there is demand, I can upload onto Brightspace last year's narrated slides.. (should be very similar to this year's material)**

Books

- No book covers large fractions of this course.

- Parts of chapters 4, 6, (7), 13 of Tom Mitchell's "Machine Learning"
- Parts of chapter V of Mackay's "Information Theory, Inference, and Learning Algorithms", available online at:

http://www.inference.phy.cam.ac.uk/mackay/itprnn/book.html

- Chapter 20 of Russell and Norvig's "Artificial Intelligence: A Modern Approach", also available at:

http://aima.cs.berkeley.edu/newchap20.pdf

- More materials later..

Marking

- **3 landmark papers to read, and submit a 10-line summary on Brightspace about: each worth 6-7%**
- **a connectionist model to build and play with on some sets, write a report: 30%**
- **Final Exam in the RDS (50%)**

Programming assignment

- Implement a Multi-Layer Perceptron, test it.
- The description on Brightspace.
- Submit through Brightspace code and test results by Dec the 5th at 23:59, any time zone of your choice (Baker Island?).
- 30% of the overall mark
- One third of a grade down every day late, that is: if you deserve an A and you're 1 day late you get an A-, 2 days late a B+, etc.

Assignment 3

- Read the paper "Reducing the Dimensionality of Data with Neural Networks", by Hinton and Salakhutdinov (2006).
- Don't panic!
- The paper is on Brightspace.
- Submit through Brightspace a 400 word MAX summary by Nov 22nd at 23:59, any time zone of your choice (Baker Island?).
- 7% of the overall mark.
- One third of a grade down every 2 days late, that is: if you deserve an A and you're 1-2 days late you get an A-, 3-4 days late a B+, etc.

Priors: P(M)

- **The use of priors is often criticised because it might introduce an arbitrary element into the inference mechanism.**
- **In reality:**
  - **Often, the more data we have, the less important priors become.**
  - **Non-informative priors (e.g. uniform) can often be derived.**
  - **Priors are really always used, even when they aren't mentioned, so it is better to deal with them explicitly.**
  - **In the Bayesian framework we can at least assess their impact.**

Likelihood: P(D|M)

- **This can be dealt with easily if our model can assess naturally the probability of an observation D (e.g. if the outputs of a softmax MLP could be interpreted as probabilities..), or *generates* data with a certain probability (think of Boltzmann machines).**

- **In NNs we used Error functions; how can we translate them into likelihoods? We will need to make some assumption on how the error is distributed.**

Using the Bayesian machinery

- **We can try to select the model that minimises the following error:**
- **$E = - \log P(M|D) = -\log P(D|M) - \log P(M) + \log P(D)$**
- **This is called MAP (maximum a posteriori) estimate, because we are maximising the posterior.**

How to do the M part of MAP and ML

- **ML and MAP give us an objective, but not a method to achieve it.**
- **Practically performing the maximisation can be a rough task, and there are a lot of techniques to do it. For instance gradient descent (but also simulated annealing, etc. etc.)..**

MAP/ML, NNs and supervised learning

- **The terms expressing the probability of the inputs don't depend on the weights of the model.**
- **What we are interested in minimising is:**

$$-\sum_{i=1}^{N} logP(t_i|d_i, w) - logP(w)$$

The prior

- **If we assume that P(w) has a Gaussian distribution, -log P(w) contributes a term proportional to $-w^2$ to the error. This is precisely *weight decay*.**
- ***Weight sharing* can also be included into the prior.**

The likelihood: linear case

- Let's assume that we have a MLP with linear output.
- We want to evaluate P(t|d,w). It is natural to express (d,w) as what the model does with d, i.e. the output of the model to d: y(d)=y. We can rewrite:
- P(t|d,w) = P(t|y)

Linear output: Gaussian assumption

- **The "natural" distribution of t given y in the linear case is Gaussian.**

$$P(t|d,w) = \prod_{j=1}^{K} \frac{1}{\sqrt{2\pi}\sigma_j} e^{-\frac{(t_j - y_j)^2}{2\sigma_j^2}}$$

Error for Gaussian assumption

- **If we assume that the target is distributed like a gaussian variable with mean y, then the log likelihood becomes:**

$$-logP(t|d,w) = \sum_{j=1}^{K}\Big[\frac{(t_j - y_j)^2}{2\sigma_j^2} - \frac{1}{2}log(2\pi\sigma_j^2)\Big]$$

Natural assumptions

- For the linear output case (regression), if we assume that P(t|y) is distributed like a Gaussian, the associated likelihood really looks just like a squared error.
- This means that minimising the squared error really means doing ML.
- [And if we add a bit of weight decay we can call it MAP]

The likelihood: binary classification

- **In this case we have two classes and we want to assign each example to either.**
- **We can use a binomial model for P(t|d,w):**

$$P(t|d,w) = y^t(1-y)^{1-t}$$

- **This also implies that y(d) is an estimate of the probability that d is in either class (also: y must be in [0,1]).**

Error for binary classification

- **In this case, the log likelihood becomes the relative (mutual) entropy between target and output:**

$$-logP(t|d,w) = -tlogy - (1-t)log(1-y)$$

Natural assumptions 2

- For the binary classification case, if we assume that P(t|y) is distributed binomially, the associated likelihood really looks just like a relative (mutual) entropy error.

- If we implement the output using a sigmoid, taking the derivative of the error wrt the weights the only output term remaining is proportional to (t-y).

Multinomial case

- **In the more general case of multi-class classification we can assume a multinomial model for the likelihood:**

$$P(t|d,w) = \prod_{i=1}^{K} y_i^{t_i}$$

- **This also implies that y(d) estimates the probability of d being in any class (also: y_i must sum to 1).**

Error in the multinomial case

- **In this case the log-likelihood becomes:**

$$-logP(t|d,w) = -\sum_{i=1}^{K} t_i logy_i$$

- **This is again a relative (cross) entropy.**

Natural assumptions 3

- For the multi-class classification case, if we assume that P(t|y) is distributed multinomially, the associated likelihood really looks just like a relative (cross) entropy error.

- We saw before that if we implement the output using softmax, taking the derivative of the error wrt the weights the only output term remaining is proportional to (t-y).

Summary: NNs and ML

	probabilistic model for P(t|y)	natural output function	error (- log likelihood)
regression	gaussian	linear	squared
binary classification	binomial	sigmoid	relative (mutual) entropy
multi-class classification	multinomial	softmax	relative (cross) entropy