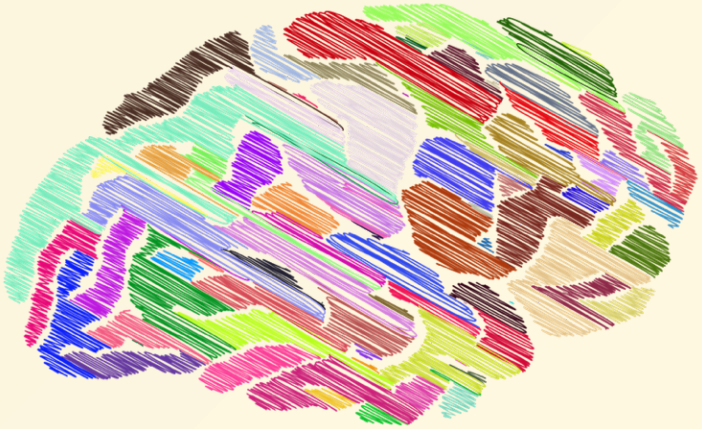
Project Ideas



Jeremy Bernstein
bernstein@caltech.edu

Agenda for today:

1. Lecture recap
2. Discuss project ideas

L7: Neural Architecture Design

Different architectures: MLP, CNN, transformer

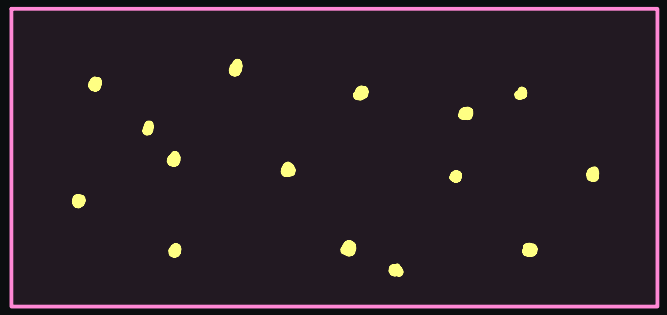
Ways to choose architecture:
- match symmetries in the data.
- architecture search (computationally heavy)

Local principles of architecture design including weight constraints — you can think of a neural net as a Cartesian product of hyperspheres.

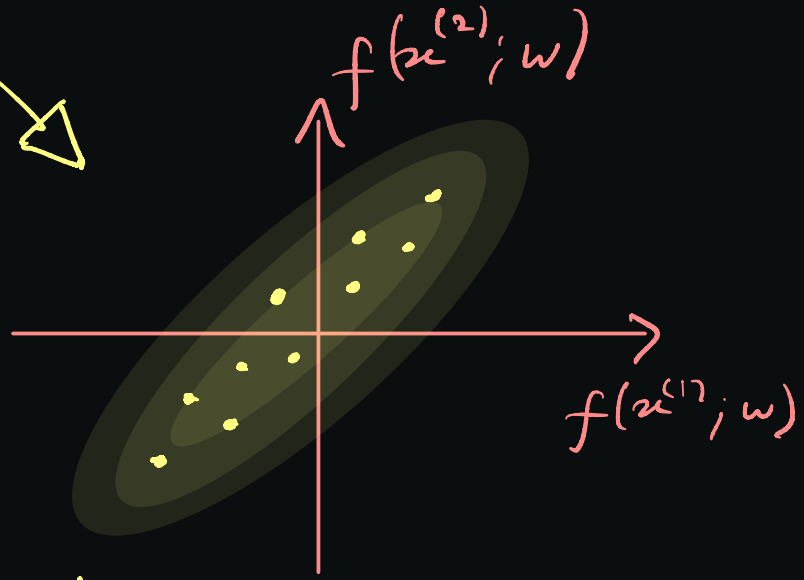
Introduced perturbation bounds on network output in terms of relative perturbation to each layer.

L8: Network Function Spaces

Studied how randomly sampling weight vectors in weight space



Leads to a Gaussian distribution of network outputs on any finite set of inputs...



... for wide enough networks.

L9: Network Optimisation

Introduced the Adam optimiser, which
- removes gradient scales
- retains the relative size of successive minibatch gradients

Discussed first order optimisation theory
which relies on a good smoothness model of the loss

Modelled the smoothness of neural networks
via perturbation bounds on the network architecture

e.g. $$\frac{\|f(x;W+\Delta W)-f(x;W)\|_2}{\|f(x;W)\|_2} \leq C \prod_{L=1}^{L} \left(1+\frac{\|\Delta W_L\|}{\|W_L\|}\right) - 1 .$$

L10: Statistical Learning Theory

Proved a VC-style generalisation bound:

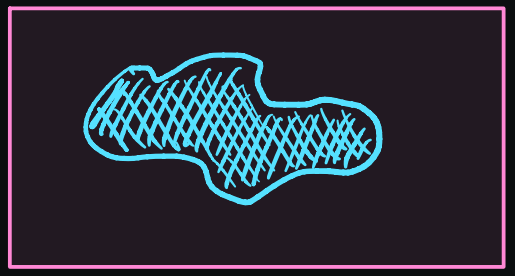
$$P\left[\begin{array}{c}\text{for any} \\ f \in F\end{array} \;\middle|\; \begin{array}{c}\text{train} \\ \text{error}\end{array} - \begin{array}{c}\text{test} \\ \text{error}\end{array} \middle| \geq \varepsilon\right] \leq \frac{2\,\mathbb{E}_X[N(F, X)]}{\exp\left(m\frac{\varepsilon^2}{2}\right)}$$

- applies to all classifiers f in the function class F.
- involves counting the number of dichotomies $N(F, X)$ that the classifier can realise on a sample X of $2m$ datapoints.

L11: PAC-Bayesian Theory

Given a dataset and a network architecture, define the

$$\text{evidence} = \frac{\text{volume of solutions}}{\text{volume of weight space}}$$



i.e. the probability of randomly sampling a weight vector that fits the training data.

Then PAC-Bayes says that for $(1-\delta)$ iid training sets,

$$\underset{\text{over solutions}}{\text{average}} \left[\text{population error} \right] \leq \frac{\log \frac{1}{\text{evidence}} + \log \frac{2m}{\delta}}{m-1}.$$