10-708: 08/07/2020

L15: Statistical and Algorithmic Foundations of DL

Ex: PGMS provide an interesting perspective on DL, particularly on new methods
- High-level coverage of connections between PGMs, DL
- Invite guest lecturers to present frontier work.

Ex: Broader community outside ML drawn to DL
- cover 1st 2 parts; rest - after class review

(*) Perception and Neural Nets

- McCulloch & Pitts (1943): mathematical model of biological prototype
- biological neuron network → ANN
- Rosenblatt perceptron

(*) 'Combined logistic models'

- biological NN - in principle can measure intermediate output
- in ANNs: only see input and output; don't see intermediate (?) (?)
NNs model or process?

(*) Backprop

- ANN as a computational graph.
- input, output; 'middle' subject to design
- we have derivatives of output not input
- use chain rule
- A comput. procedure to relay gradient into different layers.

$$\frac{\partial f_n}{\partial x} = \sum_{i_1 \in \pi(n)} \frac{\partial f_n}{\partial f_{i_1}} \frac{\partial f_{i_1}}{\partial x} = \sum_{i_1 \in \pi(n)} \frac{\partial f_n}{\partial f_{i_1}} \sum_{i_2 \in \pi(n)} \frac{\partial f_{i_1}}{\partial f_{i_2}} \frac{\partial f_{i_2}}{\partial x} = \cdots$$

- If functions are stochastic → stochastic backprop
(*) Modern packages → library of derivatives; reverse-mode differentiation

(*) Modern building blocks

- Activations
- Layers
- Loss functions

- arbitrary combos of building blocks
- can include loss inside if you want

Ex: No has proved that the whole network can be 'trained'; parameters estimated given enough data

Representational learning → layers of progressively more abstract rep. (the received interpret.)

Ex: Don't give too much weight to this idea (lots of meaningless nodes too)

(*) Similarities/differences of PGMs, NNs.

· NN is a graph of computation

- lots of time in PGMs to correctness of inference algorithms
- inference not studied or a space of exploration in DL (space is architect.)
- many of these do not need noting - v. high level at which to apprec.
- PGMs, through structure, can inspire approximations

(*) Graphical models vs deep nets

- network → complex dec. hypotheses (large use projection, aggregation)

Ex. DL literature nomenclature decontextualises old ideas

- Build unified vocab to show connections (historical)

- NN as graphical models:-

(*) Boltzmann Machines (Hinton & Sejnowski 1983)

(*) Restricted Boltzmann Machines (Smolensky 1986)

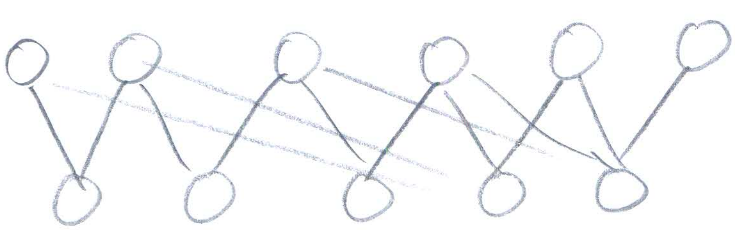
(*) Learning and inference in sigmoid belief networks (Neal 1992)

(*) Fast learning in deep belief networks (Hinton, Osindero, Teh, 2006)

(*) Deep Boltzmann Machines (Salakhutdinov, Hinton 2009)

(*) Restricted Boltzmann Machines

- RBM is MRF with bi-partite graph
- All nodes in one layer/part of graph (fully connected)



(A) weight, factor
- factor graph

Joint distn: $p(v,h) = \frac{1}{Z} \exp\left\{ \sum_{ij} w_{ij} v_i h_i + \sum_i b_i v_i + \sum_j c_j h_j \right\}$

log-like: (single data point) $\log L(v) = \log \sum_h \exp\left\{ \sum_{ij} w_{ij} v_i h_i + \sum_i b_i v_i + \sum_j c_j h_j - \log(Z) \right\}$

- unobs. marginalised out.

gradient of log-like. $\frac{\partial}{\partial w_{ij}} \log L(v) = \sum_h p(h|v) \frac{\partial}{\partial w_{ij}} p(v,h) - \sum_{v,h} p(v,h) \frac{\partial}{\partial w_{ij}} p(v,h)$

? AI review - clarity?

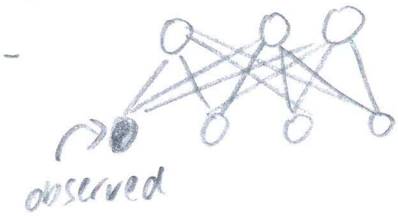
- averaging over post / joint

- AH form: $\frac{\partial}{\partial w_{ij}} \log L(v) = \underset{(i)}{\mathbb{E}_{p(h|v)}\left[\frac{\partial}{\partial w_{ij}} p(v,h) \right]} - \underset{(ii)}{\mathbb{E}_{p(v,h)}\left[\frac{\partial}{\partial w_{ij}} p(v,h) \right]}$

(*) Approximate expect. via sampling

(i) sampling from posterior is exact (RBM factorises over h given v)

- Note: for UGMs :-



- when common neighbour is observed, all others d-separate.
- can sample one by one (trivial)

(ii) nontrivial, as cannot condition on evidence.

- Have to sample from entire joint distribution.
- Sampling from joint is approximate → opens up whole space of lit.
- via MCMC. (e.g. Gibbs sampling

NN lit: clamped - unclamped, wake-sleep, pos-neg.

(*) connect very deeply to GANs and VAE.

(*) learning/param est is possible

II. Sigmoid Belief Nets

(*) (A1) - see slides for architec. ex: Not the best peformance

- Directed GMS
- widely used in medical diagnosis

(*) SBMs - are BNS over binary variables with CPDS rep by sigmoid functions

$$p(x_i|\pi(x_i)) = \sigma\left(x_i \sum_{x_j \in \pi(x_i)} w_{ij} x_j\right)$$

- Practical difficulty of training at bottom layer (v-structure)
- inference of one particular r.v; explaining away ⟶ coupling of that with all other nodes in the hidden layer. (could be thousands)
- Distinct from RBMS in the tractability of inference (d-sep in that case)

SBMs - estimation, inference

(O1) · v-structure/explaining away yields insight on complexity of inference.

- HO (*) slow convergence

(*) RBMS as infinite belief networks

(*) Tie RBM and SBMs

- Can sample joint using Gibbs procedure - alternates between different subsets of r.v.s.
- vanilla Gibbs - sample every single r.v given the rest.
- Block Gibb sampling - group r.v.s. of interest ⟶ blocks ; sample block conditioned on other blocks
- introduce observed r.v. ; given as input, sample hiddens - segregate into 2 blocks - top and bottom
- then given hidden, sample output ?
- A natural Gibbs Sampling step

(*) Gibbs sampling → alternate between sampling hidden and visible/observed variables

(*) Conditional distr's. $P(v|h)$ and $P(h|v)$ rep. by sigmoids.

(*) Gibbs sampling from joint → top down propagation in infm deep SBN (?)

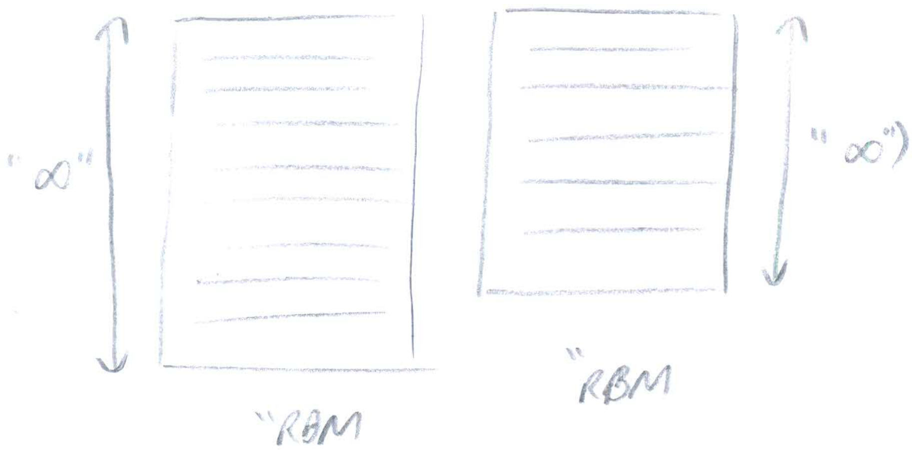
EX: Every path is a single network from input to output
each episode of sampling breaking down into concatenated sigmoid fns;
sample just once (?)

A2 - clarity

(*) RBMs are infinite belief nets

(?)

- Have W and W'
- Every layer in infinite stack uses same weights
- Each episode of Gibbs sampling is one pass of 'infinitely' many layers



(*) correspondence between propr PGM which can be 'learnt' using 'correct' techniques and another algorithm you can use.

(*) Inference in RBM is equivalent to one-pass-procedure.

(*) Gives info on how to <u>train</u> RBMs.

- weight updates forward → ? } sigmoid
  top-down → $v|h$

- is equivalent to estimating weights in RBMs (infinite stack), except they are clamped.

A3: Really <u>unclear</u> → needs review.

(*) RBMs and SBMs: RBMs are infinite depth SBMs with all weights clamped/shared across layers.

(*) This equivalence was found by Radford Neal.

(*) A separate model → DBNs

IV. Deep Belief Nets

(*) Hybrid graphical models

- multiple layers of RBMs connected to latent variables/obs through sigmoids.
- technically 'chain graphs'
- difficult to learn ⟶ explaining away.

(14) - See slides ⟶ architecture

DBNs:

Joint p.d.: $P(v, h^1, h^2, h^3) = P(h^2, h^3)\, P(h^1 | h^2) P(v | h^1)$

$P(h^2, h^3)$ - RBM $\quad P(h^1|h^2), P(v|h^1)$ - condit. in sigmoid form

Training: maximise log likelihood for given $\log P(v)$

- Really struggled with RBM, SBM equivalence.
- But it can shed light on layer wise pre-training

(?) 59:09 ⟶ 1:03:37 (?)

- selective on which weights can be updated

EX: You're confused right? ☺ Yes; because you have come from a PGM lecture,
- DL we don't know what we are doing

EX: Trying to illustrate the lack of a principle underlying DL, RBM training
- e.g. not caring about loss function
- care more about comput. procedure rather than other design principles

Copy this RBM process (equivalence?) to finite layers.
infinite

(*) DBN-fine tuning

- fine tune a pretrained DBN.

setting A

(A5) review lecture here / supp with recorded tutorial.

- unsupervised learning
  1. pre-train a stack of RBMs
  2. unroll RBMs ⟶ autoencoder
  3. fine tune params by optim. recons. error

(*) emphasis in NN is operational/scalable comp. procedure to get weights; and a task e.g. reconstruc. loss / class loss etc.

(*) DBNs, Boltzmann Machines

(*) DBMs - fully undirected models (MRFs)
- trained sim. as RBMs via MCMC (Hinton & Sejnowski 1983)
- variational approx of data distr for faster training (Salakhut. + Hinton 2009)

(*) These were 'deep learning' models in early NN lit → to approx. PGM, simplify computation.

(*) They knew <u>where</u> approx introduced, where comp simplified

---

(*) Graph models vs deep networks

(*) Optimisation

Q. what is conn. between NN param estim. and optimisation?
- ↳ 'learning to learn'

- equivalence of PGMs and DL:-
  i) training DL / NN net
  - Treat process of estimating DL weights as equivalent to 1st estimating params of model with infinite layers of NN component, but with clamped weights
  - declamp weights then optimise each layer of weights.

- (*) unfolding an optimisation algorithm
- (*) Consider G.D. → run till convergence ; theory says after infinite steps we will converge (under cond.).
- (*) This corresponds to what we should do in param estim. of RBMs (iterative sampling of visible and hiddens over infinite layers as one pass (?))
- (*) Estimating params of RBMs via Gibbs sampling → one pass 'algo' of an infinite layer network with same weights
- (*) Truncate the pass into 5 layers rather than whole ; allow weights to be untied, estimate weights separately via pre-training

- see diagram of G.D.

- Get 5 sets of weights (no longer tied)
- (*) Extended analogy (everest)
  - optimise way to learn
- (*) Ex: Every NN could be mapped into PGM of some form, but NN/DL architect. is used to download comp. steps to estimate that PGM.
  - Treat computational step as goal itself to optimise (weights in every layer)
  - You may forget ultimately the PGM (e.g. RBM), but worry about
- (*) A little garbled presentation:
- (Q1): Understanding NN as truncating optimisation, optimising every step in optimisation
- (A6) - Review/understand G.D. → equations.

(*) structured prediction

- see papers Domke, Stoyanov etc.
- use this principle to empower previous methods