IAN GOODFELLOW - Deeplearning

14 : Autoencoders
15 : Representation learning
20.10.B/4 : Deep gen models -
vaes & GANS

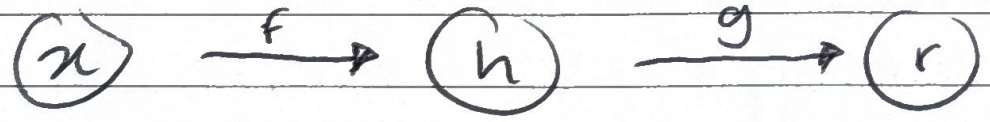
Autoencoders

An Autoencoder is a nn that is trained to attempt to copy its input to its output

internally, a hidden layer h describes a code used to represent the input
↳ latent

Network of two parts:

① An encoder $h = F(x)$
② A decoder $r = g(h)$



the ability to copy x exactly is not useful

Autoencoders are designed to be unable to learn to copy perfectly

copy to resemble train data

traditionally (historically) Autoencoders were used for dim reduction or feature learning

Recent, theory connecting Autoencoders & latent variable models

Autoencoders may be thought as a special case of feedforward networks

↳ & may be trained w/ All same techs
↳ Minibatch gradient descent, Backprop

14.1 undercomple Autoencoders
14.2 Regularized Autoencoders
14.2.1 Sparse Autoencoders
14.2.2 Denoising Autoencoders
14.2.3 regularize by penalizing Derivatives
14.3 rep power, layer size & depth
14.4 Stochastic encoders & decoders
14.5 Denoising Autoencoders
14.6 learning manifolds w/ Autoencoders
14.7 Contractive Autoencoders
14.8 Predictive Sparse Decomposition
14.9 Applications of Autoencoders

14.1 undercomplete Autoencoders

whilst achieving a copy from input to output may sound pointless

we hope that the intermediary form of h will take on some useful properties

$x \longrightarrow h \longrightarrow r$

encode decode

one way to make it useful is to constrain h to be smaller than x

↳ this is why it is called undercomplete

undercomplete forces the Autoencoder to capture the most salient features

learn progress → min loss function

$$L(x, g(f(x)))$$

loss func penalizes $g(f(x))$ for being dissimilar to x

encoder & decoder can't be allowed too much capacity otherwise the learning will just be as a mapping w/ no features learnt

14.2 Regularized Autoencoders

- undercomplete
- code dimension less than input
- learn salient features
- will not learn anything if encoder + decoder = too much capacity

Same issue occurs if hidden code has same dimensions as input

- or hidden has more dims than input = overcomplete

Ideally, train Autoencoder successfully by choosing code dims & capacity based on complexity of dist to be modeled

Reg AutoCoder provide ability

rather than limiting the capacity by keeping the en/de shallow

- Reg Autoencoders use a loss function that encourages the model to have other properties besides the ability to copy

A regularized autoencoder can still learn something useful about the data even if overcomplete

VAEs (variational autoencoders) are regularized

14.2.1 Sparse Autoencoders

An Autoencoder that includes a sparsity penalty $\Omega(h)$ on the code/hidden layer

$$L(x, \underbrace{g(f(x))}_{\text{g() decoder output}} + \Omega(h)$$

Sparse encoders learn features to then be applied for another task
↳ classification FE

MUST respond to unique statistical features rather than becoming an identity func

thus training to perform the copy task w/ a sparsity penalty can yield a model that has learned

take a feedforward network whose task is to copy the input

↳ add a sparse reg term

Regularized Autoencoders are not Bayesian as the reg depends on the data

rather than think of sparsity penalty as a reg for the copy task

think of entire sparse Autoencoder as approximating Max likelihood of train a generative Model that has latent variables

---

14.3 Representational Power, layer Size & Depth

often Autoencoders =
Single layer encode + Single layer Decode

However, this is not a requirement

using Deep encoders & Decoders has Many Advantages

Note: Feedforward NN's benefit from Depth

↳ Autoencoders are feedforwards
↳ encoders & Deceders are FFs
↳ All components benefit from Depth

Advantage of Depth: Universal Approx theorem gaurantees that a feedforw network w/ atleast one hidden layer can represent an approx of any func

to an arbitary degree of accuracy provided enough hidden units

Arbitary here means any

↳ there exists a num of hid units to acheive the selected accuracy

if an encoder is shallow - no Hidden - cannot acheive arb constraints

A deep encoder can

Depth can reduce ~~cost~~ comp cost

↳ requires less weights and bias
↳ layers build up features
↳ low → med → high
↳ Shallow needs many units to acheive this
↳ this costs more weights & bias = comp cost ↑
→ layers can be seen as modular & reuse-able

14.4 Stochastic Encoders & Decoders

Autoencoders are just feedforward networks
- ↳ same loss funcs can be used
- ↳ same output unit types

In trad feednet, $p(y|x)$
- ↳ y is the target given x
- ↳ y = vector of target such as class labels

in an autoencoder, x (input) is also a target for the decoder $h \rightarrow y(x)$

$$p_{decoder}(x|h)$$

train, min $-\log_{decoder}(x|h)$

Stochastic Architecture →

$x \longrightarrow h \longrightarrow r$

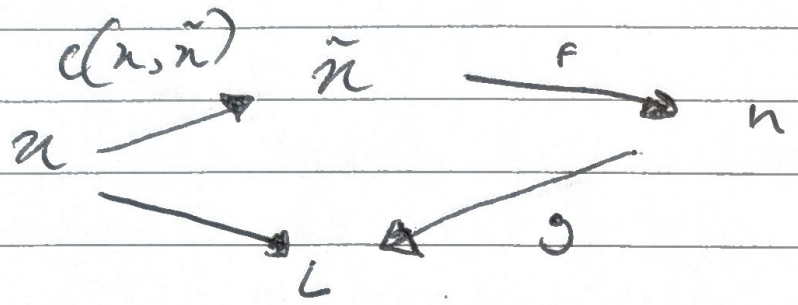
$p_{encoder}(h|x)$ $p_{decoder}(x|h)$

both encoder & decoder are not simple functions
- ↳ instead involve some noise injections
  - ↳ thus, output can be seen as being sampled FROM a distribution

Any latent variable model defines a stochastic encoder/decoder

14.5 Denoising Autoencoders DAE

An A DAE recieves a corrupted data point as its input & is trained to predict the original, uncorrupted data point



corruption process $C(\tilde{x} \mid x)$

dates back to LeCun (1987)

Denoising Autoencoders are just MLPs trained to denoise

"Denoising Autoencoders" as a specific type of model

↳ not just merely denoise input
↳ but learn good internal representation

goal is to allow the creation of a high capacity encoder

↳ but avoid identity function

14.6 Manifolds

~~Contractive Autoencoders~~

the manifold hypothesis says

↳ "Many high dimensional datasets that occur in the real world actually lay upon some low dimensional manifold within that high dimensional space"

2-D example of a Manifold
↳ line, circle

3-D Manifolds
↳ Plane sphere

0-D = Dot

goal of A NN is to stretch & Morph ~~these~~ & Disentangle these manifolds

such that we can finally seperate ~~them~~ them using a hyperplane in the final layer

14.7 Contractive Autoencoders

14.8 Predictive sparse decomosition

hybrid of Sparse & Parametric

Parametric encoder = trained to predict the output of iterative inference

14.9 Applications of Autoencoders

Autoencoders have been successfully applied to dimension reduction & information retrieval tasks

Dim red = one of first applications of ~~rep~~ representation learning & DL

↳ early study motivs for Autoencoders

Why is dimension reduction good?

↳ lower dims improve performance on many tasks
↳ less memory & runtime

Many forms of Dim reduction place semantically related example near each other

the hints provided by mapping to lower dimensions aid generalization