Autoencoder

25 November 2024 09:45

Autoencoder vs PCA

Autoencoder

low dimen^n representn → $Z = f(W_e \cdot X + b_e)$ (encoder weight, bias; X: original I/P vector)

reconstructed I/P vector → $X' = f(W_d \cdot Z + b_d)$ (decoder weight, bias)

PCA

$X = U \Sigma V^T$ (X: original data matr.; U: matrix of eigenvectors of covariance; V^T: transpose matrix of principle component.; Σ: diagonal matrix of variance expl. by principle component.)

Can learn non-linear represen^n of data (use non-linear activ^n in stacked autoencoder)

not interpretable

Linear technique. Principal components are linear combinations of original variable.

linear combinations are interpretable.

Self supervised / unsupervised.

Uses classification or regression loss to do the learning, predict^n.

Labels are implicit in data.

Coming up w/ method to find implicit label = self supervised.

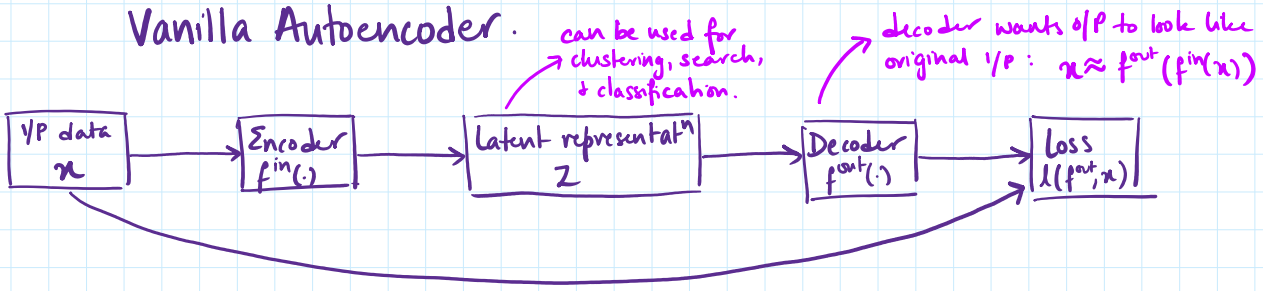
- autoencoder: predict I/P from I/P
- image inpainting: predict missing parts.
- image sorting: break image into parts, then sort.

Unsupervised

Uses of AE (not source agnostic)

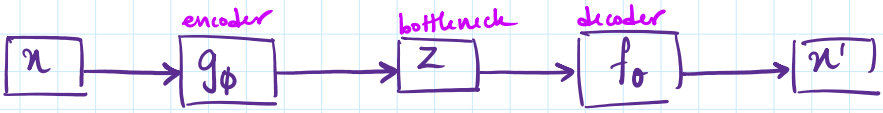
- dimensionality red^n
- anomaly detect^n
- data compression
- image denoising.

Vanilla Autoencoder.



- encoder compresses I/P into latent space representation
- decoder reconstructs I/P from z.
- objective: reconstruct I/P as accurately as possible.
- latent space dimensionality ≤ input dimensionality

Undercomplete Autoencoder.



Cost function: $L(\theta, \phi) = \frac{1}{n} \sum_{i=1}^{n} [x^{(i)} - f_\theta(g_\phi(x^{(i)}))]^2$

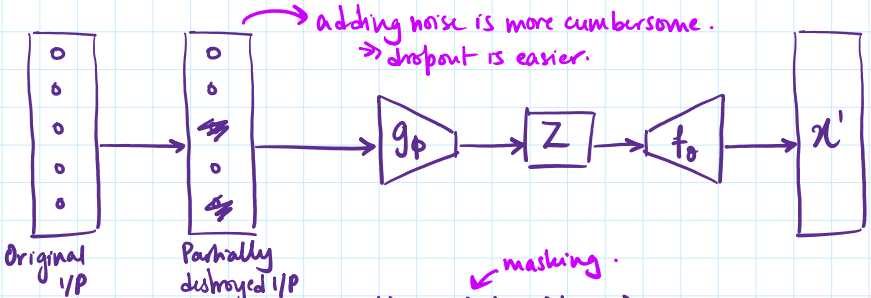
$(y_{true} - y_{pred})^2$ → basically MSE.

$\frac{1}{n}\sum_{i=1}^{n}$ ... $(y_{true} - y_{pred})^2$ → basically MSE.

- objective: learn compact + meaningful represent^n of data.
- latent space dimensionality < I/P dimensionality.

should be small otherwise latent rep. may not be effective ∴ (it's just copying the I/P)

Denoising Autoencoder.



masking.

$$\tilde{x}^{(i)} \sim f(\tilde{x}^{(i)} | x^{(i)})$$

Loss: $L(\theta, \phi) = \frac{1}{n}\sum [x^{(i)}_{\text{clean}} - f_\theta(g_\phi(\tilde{x}^{(i)}_{\text{after noise}}))]^2$

We FINALLY get a denoised version of the original input.
+
network learns to remove noise
= more robust model.

Autoencoder in Anomaly Detection.

① Bottleneck FORCES autoencoder to learn only impt info from I/P that's impt for reconstruction

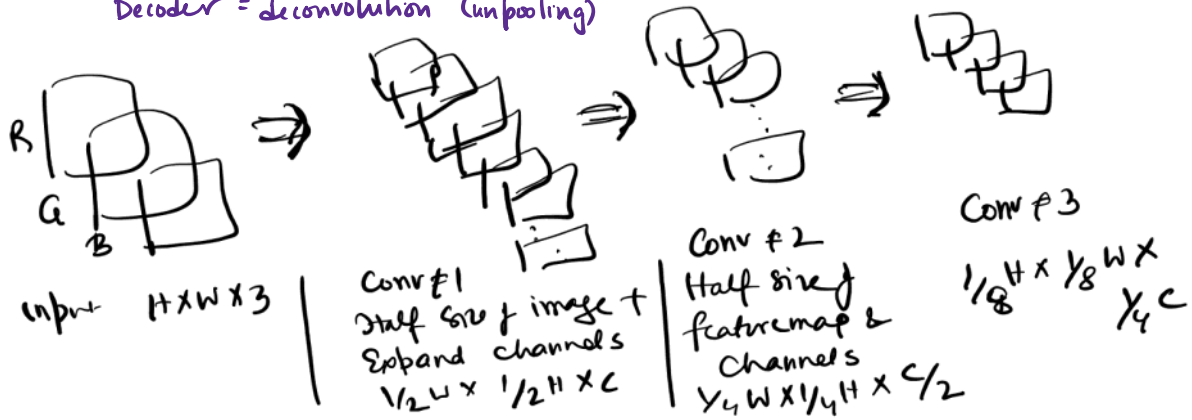
② If trained on normal data → reconstruction loss typical of non-anomalous data.

③ If trained on anomalous data → reconstruction loss super high.

Autoencoders and Convolution.

Encoder = convolution (pooling)
Decoder = deconvolution (unpooling)

* channels C depends on no. of filters.



Deconvolution:

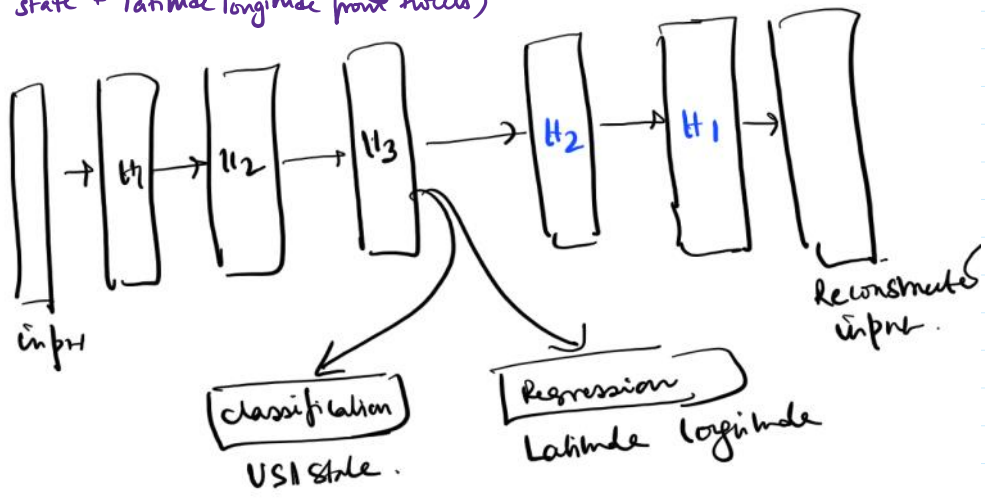
$\frac{1}{8}H \times \frac{1}{8}W \times \frac{1}{4}C \rightarrow \frac{1}{4}H \times \frac{1}{4}W \times \frac{1}{2}C \rightarrow \frac{1}{2}H \times \frac{1}{2}W \times C \rightarrow H \times W \times 3$

Deconvolution :

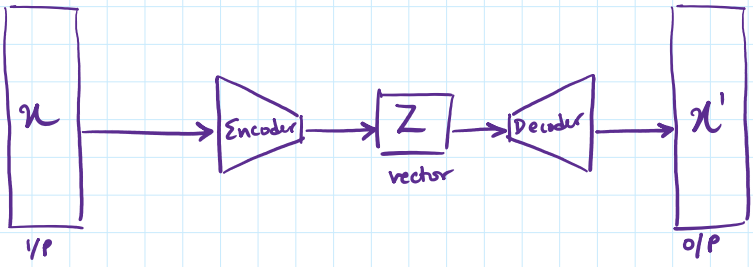
$\frac{1}{8}H \times \frac{1}{8}W \times \frac{1}{4}C \rightarrow \frac{1}{4}H \times \frac{1}{4}W \times \frac{1}{2}C \rightarrow \frac{1}{2}H \times \frac{1}{2}W \times C \rightarrow H \times W \times 3$

Stacked Denoising Autoencoder.

(to infer state + latitude longitude from tweets)



Is autoencoder generative? no.



① Z is a vector — doesn't capture semantic rel^n b/w data samples

② AE only learns MAPPING FUNCTION that does COMPRESSION into a vector with a good COMPRESSION LOSS.

③ So it can't really generate anything. It can only try to replicate the I/P.

④ Doesn't learn I/P distribution.

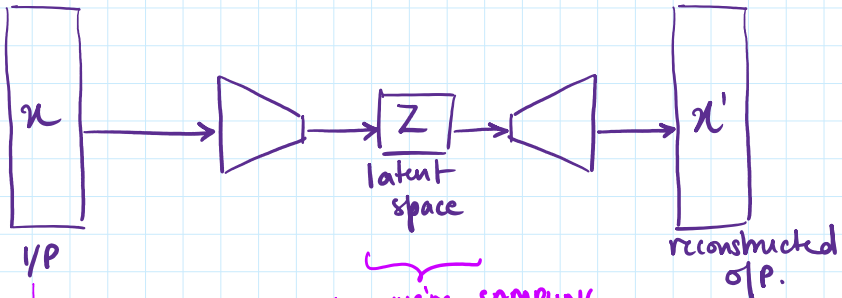
HOWEVER:
an autoencoder w/ generative ability is a variational autoencoder.

Variational Autoencoder. → learns a latent space instead of learning a vector.

represents parameters of multivariate distribution (origin of I/P.)

↓

hence we can generate similar objects.



now we're SAMPLING from a LATENT SPACE to GENERATE new data.

→ conditioned on other random variable z.

model I/P as coming from random variable X → conditioned on other random variable Z.

Z: latent (hidden) → modelled as multivariate gaussian

Now, since X is conditioned on Z:

$$p(X) = \int p(X,z)\,dz$$

$p(X)$: likelihood of our data

$\int p(X,z)\,dz$: marginalize over joint probability WRT latent variable → not possible. Too computationally slow.

we can also write this as: $p(z|X) = \frac{p(X,z)}{p(X)}$

or

$$p(X) = \frac{p(X,z)}{p(z|X)}$$

$p(z|X)$: prob. of latent variable given data. ALSO UNKNOWN.

Issues:

① To have $p(X)$, we need $p(z|X)$. Not computationally tractable.

② To have $p(z|X)$ we need tractable $p(X)$.

③ $p(X) = \frac{p(X,z)}{p(z|X)}$ $\quad p(z|X) = \frac{p(X,z)}{p(X)}$ → So we approximate.

Let true posterior to be found be $p_\theta(z|X)$ (generative params)

Like any probability distribution: $\int q_\psi(z|X)\,dz = 1$.

Then a whole bunch of math.

① log likelihood
② Change approximation to expectation
③ Expand on chain rule of probability

evidence based lower bound.

$$\log p_\theta(X) = \mathcal{ELBO} + D_{KL}(q_\psi(z|X) \,||\, p_\theta(z|X))$$

cuz $\log p_\theta(X) \geq \mathcal{ELBO}$

$$\mathcal{ELBO} = \mathbb{E}_{q_\psi(z|X)}\left[\log \frac{p_\theta(X,z)}{q_\psi(z|X)}\right]$$

maximize this → in order to maximize that.

Maximize reconstruction likelihood of decoder

$D_{KL} \geq 0$ → minimize distance of learnt distribution from prior belief of z.

* $L(\theta, \psi; X)$ optimizn wrt ψ (variational param) and θ (generational param) is a challenge.
→ cuz gradient loss bound wrt ψ is problematic due to stochastic nature.

Now,

→ VAE uses an external source of noise that's combined w/ z.

↓

backprop can't be done over stochastic z.

→ reparameterization

→ Also, actual loss fn $L(\theta, \phi; X)$ = KL divergence of z space & learnt z-space + reconstruction MSE loss.

So, what is the net carry forward from this note,

(a) AutoEncoder is not a generative model but VAE is

(b) z is a vector vs. Z is latent space & sample

(c) from observable data X, we are learning latent space Z and generating X' w.r.t. Z. Hence, it is generative.

(d) Overall maximization of log likelihood of X
i.e. $\log p_\theta(x)$ is turned into a maximization of ELBO that can be broken into 2 parts (that Proof is excluded)
(i) Reconstruction Loss (MSE)
(ii) D_{KL} between the prior Z and learnt Z.

(e) Actual implementation of the "loss function" has a challenge as Gradient Descent on Stochastic ε can be difficult (variance can be large) + so, reparameterization trick is done by taking out source of variance external variable!!

The Trick

The reparameterization trick allows us to bypass this problem by transforming the sampling process into a deterministic one. Here's how it works:

1. **Latent Variables:** Instead of sampling directly from the latent distribution, we sample from a standard normal distribution (which is easy to work with).
2. **Transformation:** We then transform this sample using the mean and standard deviation learned by the encoder network. This transformation is deterministic and allows us to backpropagate gradients through the network.