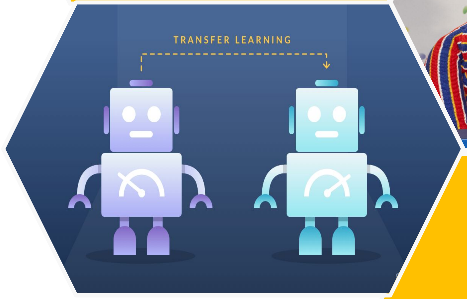
PGM in Modern AI Approaches

Eric Walzthöny
Shanshan Zhang

2022-05-24

Contents

Transfer Learning



Deep Reinforcement Learning

Transformer

Deep Generative Models

Meta Learning

Transformers

Multi-Head Self-Attention

$A_h = \text{Softmax}(\alpha Q_h K_h^\top) V_h$

~1/2

Layer-norm and residual connection

$X = \text{LayerNorm}(F_S(X)) + X$

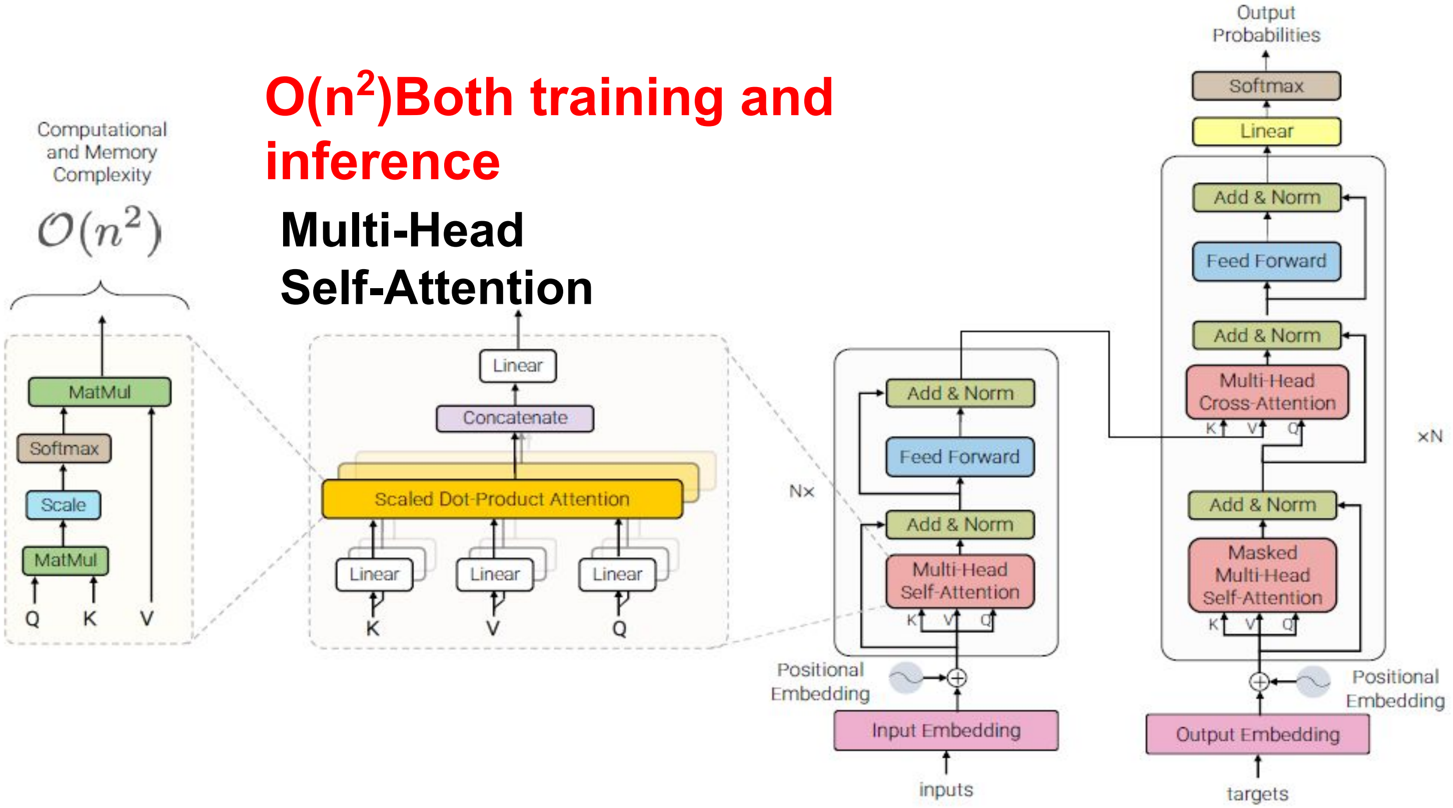
Position-wise Feed-

$F_2(ReLU(F_1(X_A)))$

~1/2

Layer-norm and residual connection

$X_B = \text{LayerNorm}(X_A)) + X_A$

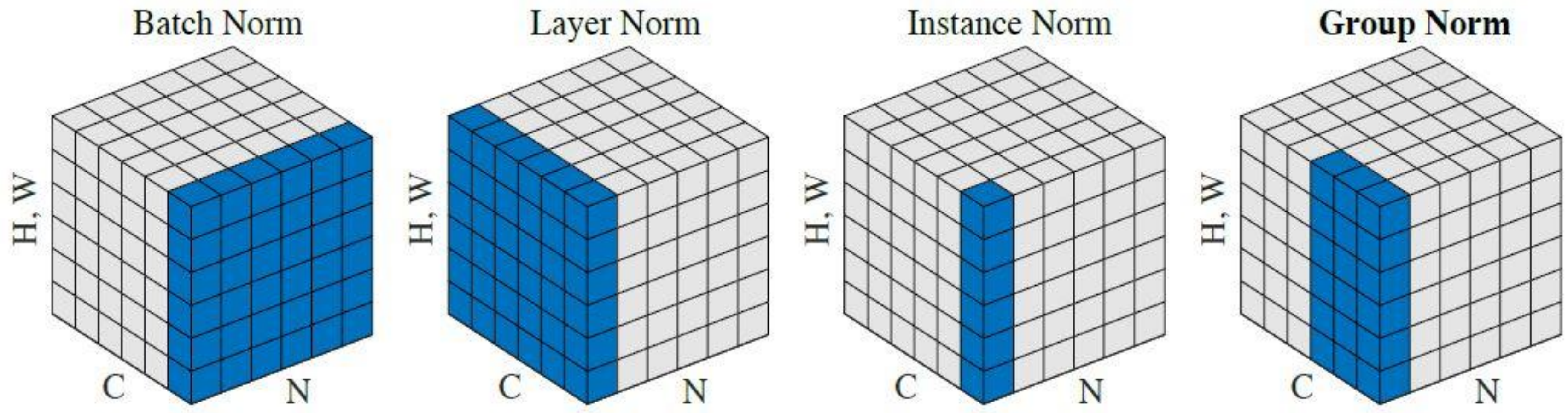


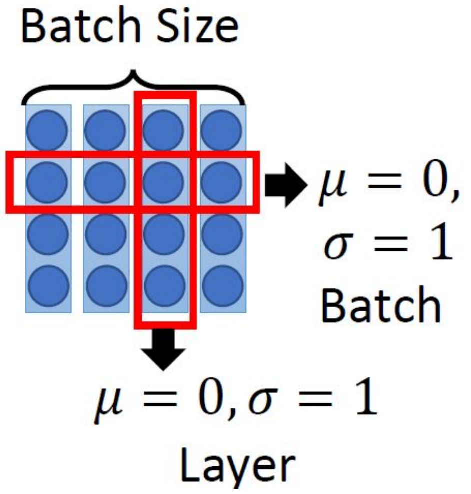
O(n^2)Both training and inference

Multi-Head Self-Attention

Attention is all you need https://arxiv.org/pdf/1706.03762.pdf

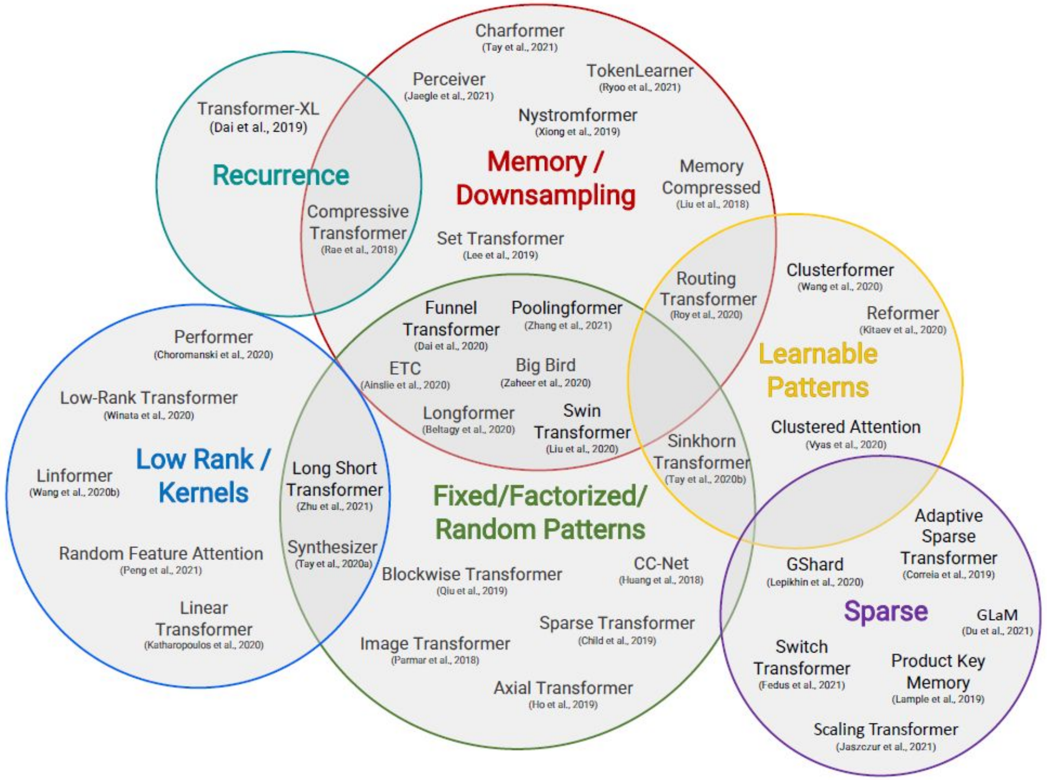
Different Normalization





BN VS. LN

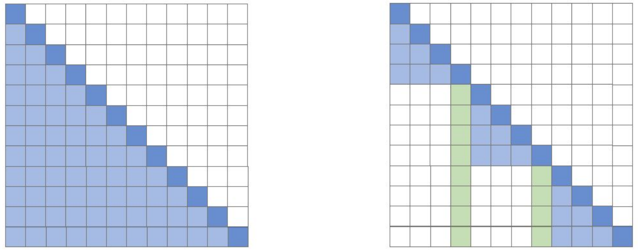
Efficient Transformers



Charformer (Tay et al., 2021)
Perceiver (Jaegle et al., 2021)
TokenLearner (Ryoo et al., 2021)
Nystromformer (Xiong et al., 2019)
Transformer-XL (Dai et al., 2019)
Recurrence
Memory / Downsampling
Memory Compressed (Liu et al., 2018)
Compressive Transformer (Rae et al., 2018)
Set Transformer (Lee et al., 2019)
Routing Transformer (Roy et al., 2020)
Clusterformer (Wang et al., 2020)
Reformer (Kitaev et al., 2020)
Learnable Patterns
Funnel Transformer (Dai et al., 2020)
Poolingformer (Zhang et al., 2021)
Performer (Choromanski et al., 2020)
ETC (Ainslie et al., 2020)
Big Bird (Zaheer et al., 2020)
Low-Rank Transformer (Winata et al., 2020)
Longformer (Beltagy et al., 2020)
Swin Transformer (Liu et al., 2020)
Sinkhorn Transformer (Tay et al., 2020b)
Clustered Attention (Vyas et al., 2020)
Linformer (Wang et al., 2020b)
Low Rank / Kernels
Long Short Transformer (Zhu et al., 2021)
Fixed/Factorized/ Random Patterns
Adaptive Sparse Transformer (Correia et al., 2019)
Random Feature Attention (Peng et al., 2021)
Synthesizer (Tay et al., 2020a)
Blockwise Transformer (Qiu et al., 2019)
CC-Net (Huang et al., 2018)
GShard (Lepikhin et al., 2020)
Sparse
GLaM (Du et al., 2021)
Linear Transformer (Katharopoulos et al., 2020)
Sparse Transformer (Child et al., 2019)
Image Transformer (Parmar et al., 2018)
Switch Transformer (Fedus et al., 2021)
Product Key Memory (Lample et al., 2019)
Axial Transformer (Ho et al., 2019)
Scaling Transformer (Jaszczur et al., 2021)

Efficient Transformers: https://arxiv.org/pdf/2009.06732.pdf

Sparse Transformer



(a) Transformer

(b) Sparse Transformer

Figure 4: Illustration of patterns of the attention matrix for dense self-attention in Transformers and sparse fixed attention in Sparse Transformers. Blue in the right diagram represents the local self-attention while green represents the strided component of the sparse attention.

Axical Transformer

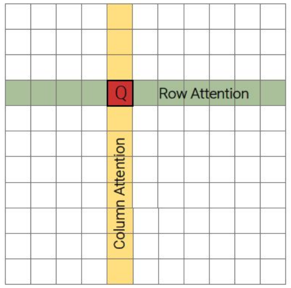
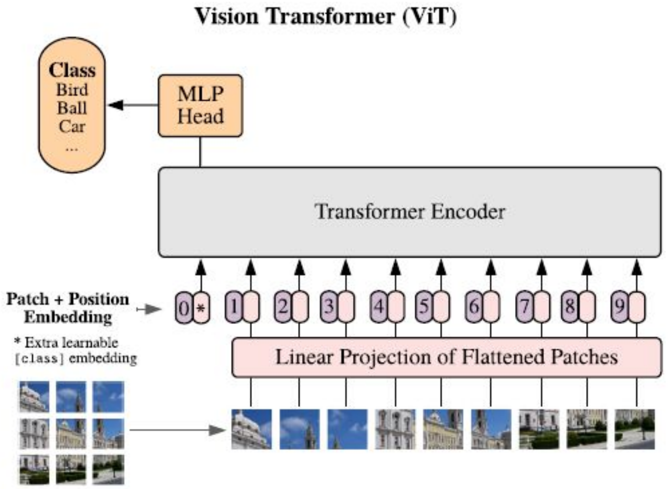


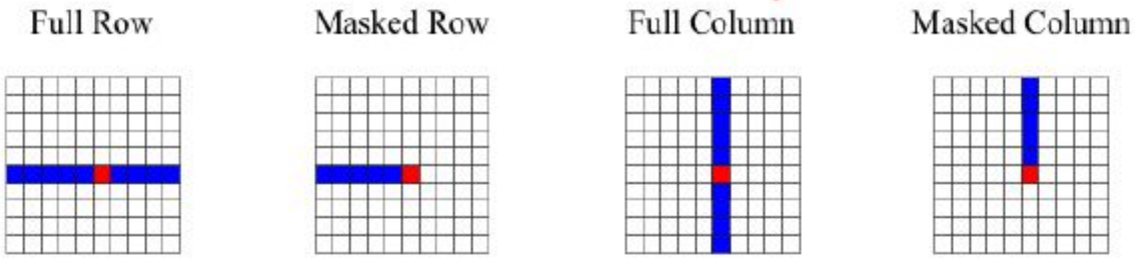
Figure 5: Attention span in Axial Transformer on a two-dimensional input.

Sparse vs. axial attention

Vision Transformer https://arxiv.org/pdf/2010.11929.pdf
Sparse Transformer https://arxiv.org/pdf/1912.12180.pdf
Axial Transformer https://arxiv.org/pdf/1912.12180.pdf

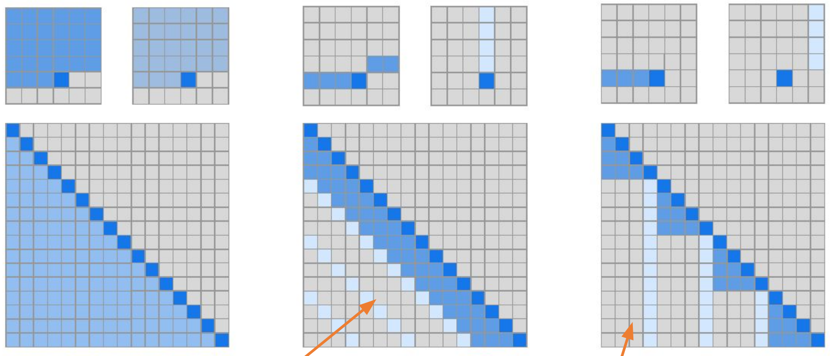

Vision Transformer (ViT)
Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
0
*
1
2
3
4
5
6
7
8
9
Linear Projection of Flattened Patches

Axial attention over axis k can be implemented by transposing all axes except k to the batch axis, calling standard attention as a subroutine


Full Row Masked Row Full Column Masked Column

~~$O(N^{(d-1)/d})$~~

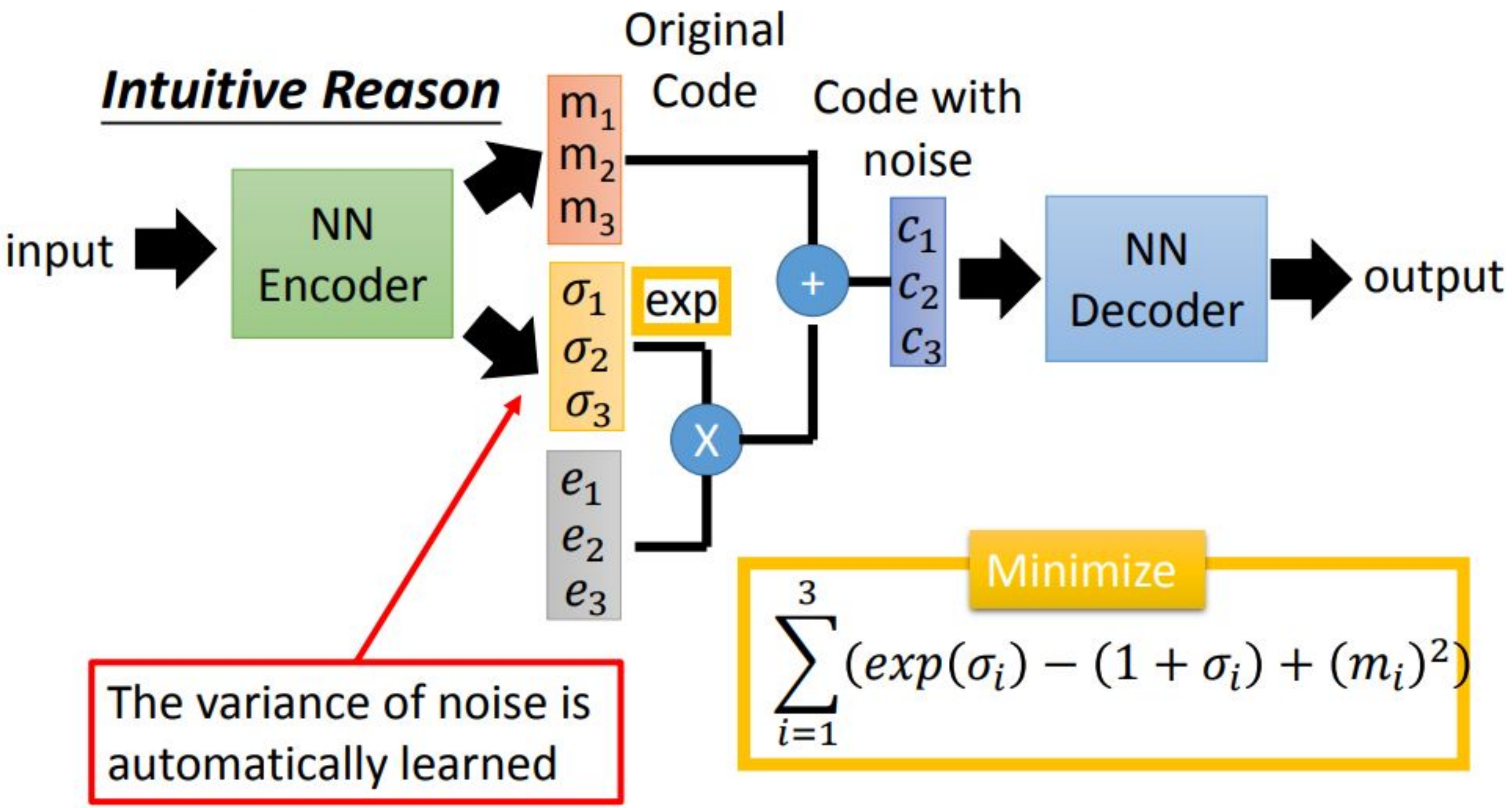
$O(n\sqrt[p]{n})$


(a) Transformer (b) Sparse Transformer (strided) (c) Sparse Transformer (fixed)

Formally, $A_i^{(1)} = \{t, t+1, ..., i\}$ for $t = \max(0, i-l)$ and $A_i^{(2)} = \{j : (i-j) \bmod l = 0\}$. This pattern can be visualized in Figure 3(b).

Formally, $A_i^{(1)} = \{j : (\lfloor j/l \rfloor = \lfloor i/l \rfloor)\}$, where the brackets denote the floor operation, and $A_i^{(2)} = \{j : j \bmod l \in \{t, t+1, ..., l\}\}$, where $t = l - c$ and c is a hyperparameter.

VAE in NN perspective



The variance of noise is automatically learned

Minimize

$$\sum_{i=1}^{3}(exp(\sigma_i) - (1 + \sigma_i) + (m_i)^2)$$

Deep Generative MODELS

Explicit probabilistic models

Provide an explicit parametric specification of the distribution of $\boldsymbol{x}$

l Tractable likelihood function p#($\boldsymbol{x}$)

l E.g., Deep generative model parameterized with NNs **(e.g., VAEs)**

l

$$\boldsymbol{P_{\emptyset}(x|z)\ = N(x;\ \mu_{\emptyset}, \sigma)}$$

$$\mathbf{P(z) = \boldsymbol{N(x; 0, I)}}$$

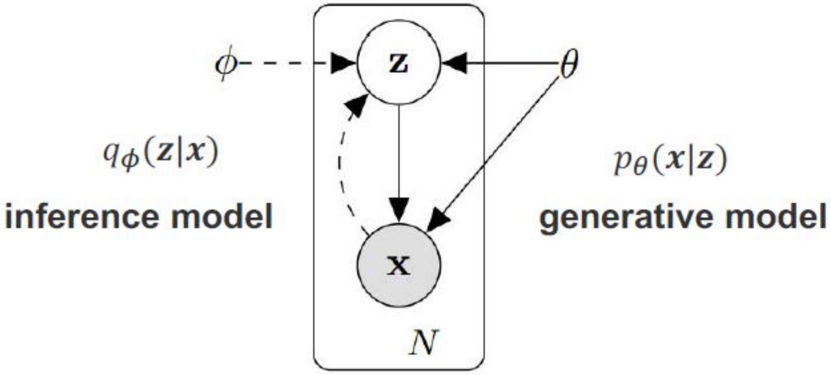

ϕ
θ
z
x
N
$q_{\phi}(\mathbf{z}|\mathbf{x})$
inference model
$p_{\theta}(\mathbf{x}|\mathbf{z})$
generative model

Figure courtesy: Kingma & Welling, 2014

VAE: https://arxiv.org/pdf/1312.6114.pdf

Variational Inference in VAE

Maximize log likelihood

$$\log[Pr(\mathbf{x}|\boldsymbol{\phi})] = \log\left[\int Pr(\mathbf{x},\mathbf{h}|\boldsymbol{\phi})d\mathbf{h}\right]$$

Jensen's inequality

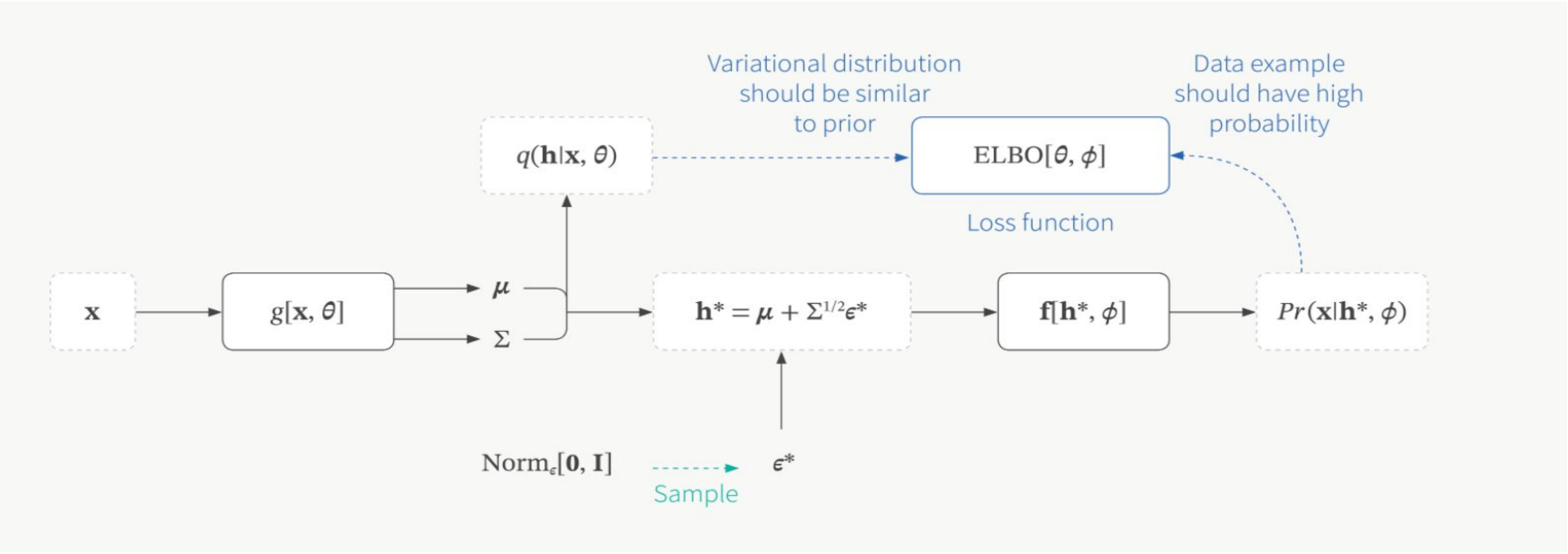
$$\mathrm{ELBO}[\boldsymbol{\theta},\boldsymbol{\phi}] = \int q(\mathbf{h}|\boldsymbol{\theta}) \log\left[\frac{Pr(\mathbf{x},\mathbf{h}|\boldsymbol{\phi})}{q(\mathbf{h}|\boldsymbol{\theta})}\right]d\mathbf{h}.$$

$$= \boxed{\int q(\mathbf{h}|\mathbf{x},\boldsymbol{\theta}) \log[Pr(\mathbf{x}|\mathbf{h},\boldsymbol{\phi})]d\mathbf{h}} - \boxed{D_{KL}[q(\mathbf{h}|\mathbf{x},\boldsymbol{\theta}), Pr(\mathbf{h})]}$$

Minimize Reconstruction Error

Minimize

$$\sum_{i=1}^{3}(exp(\sigma_i) - (1+\sigma_i) + (m_i)^2)$$


Variational distribution should be similar to prior
Data example should have high probability
$q(\mathbf{h}|\mathbf{x},\theta)$
$\mathrm{ELBO}[\theta,\phi]$
Loss function
$\mathbf{x}$
$\mathrm{g}[\mathbf{x},\theta]$
μ
Σ
$\mathbf{h}^* = \mu + \Sigma^{1/2}\epsilon^*$
$\mathbf{f}[\mathbf{h}^*,\phi]$
$Pr(\mathbf{x}|\mathbf{h}^*,\phi)$
$\mathrm{Norm}_\epsilon[\mathbf{0},\mathbf{I}]$
Sample
ϵ^*

DEEP GENERATIVE MODEL

Mode Collapse?!

Motivating entropy regularization!! Adding the entropy of G(z)) to the objective

Implicit probabilistic models – **GANs**

- Defines a stochastic process to simulate data $\mathrm{x} = G_\theta\,(\mathrm{z})$
- Define an implicit distribution over x: $P_{g_\theta}(x)$
- Do not require tractable likelihood function

$$\min_G \max_D V(D,G) = E_{x \sim p_{data}(x)}\left[\log D(x)\right] + E_{z \sim p_z(z)}\left[\log\left(1 - D\left(G\left(z\right)\right)\right)\right].$$

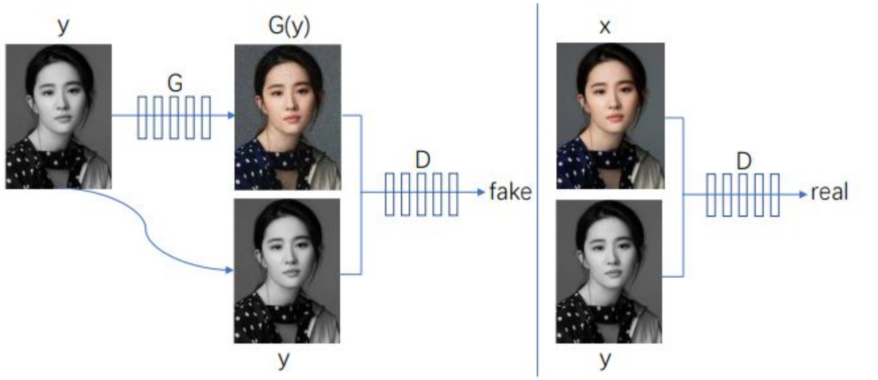
- Generate data from a deterministic equation given parameters and random noise $\mathrm{z} \sim \mathrm{N}(0,\ \mathrm{I})$

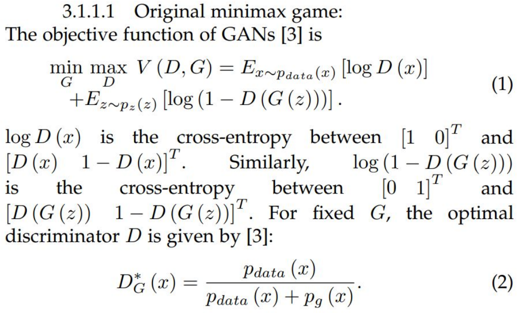
- Intractable to evaluate likelihood

Goodfellow et al., 2014
https://arxiv.org/pdf/1406.2661.pdf

GAN Variants

Pixel2Pixel





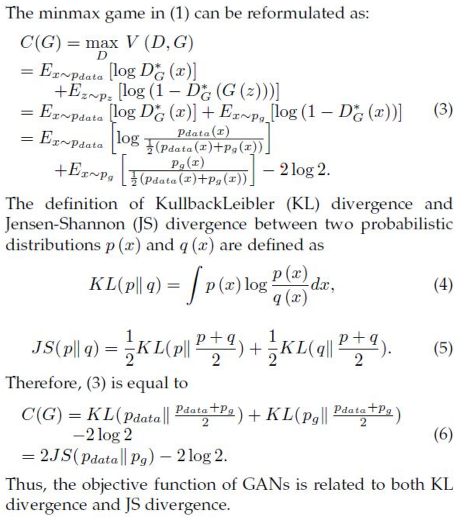
3.1.1.1 Original minimax game:

The objective function of GANs [3] is

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}\left[\log D(x)\right] + E_{z\sim p_z(z)}\left[\log\left(1 - D(G(z))\right)\right]. \tag{1}$$

$\log D(x)$ is the cross-entropy between $[1 \quad 0]^T$ and $[D(x) \quad 1 - D(x)]^T$. Similarly, $\log(1 - D(G(z)))$ is the cross-entropy between $[0 \quad 1]^T$ and $[D(G(z)) \quad 1 - D(G(z))]^T$. For fixed G, the optimal discriminator D is given by [3]:

$$D_G^*(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)}. \tag{2}$$



The minmax game in (1) can be reformulated as:

$$\begin{aligned} C(G) &= \max_D V(D,G) \\ &= E_{x\sim p_{data}}\left[\log D_G^*(x)\right] \\ &\quad + E_{z\sim p_z}\left[\log\left(1 - D_G^*(G(z))\right)\right] \\ &= E_{x\sim p_{data}}\left[\log D_G^*(x)\right] + E_{x\sim p_g}\left[\log\left(1 - D_G^*(x)\right)\right] \\ &= E_{x\sim p_{data}}\left[\log \tfrac{p_{data}(x)}{\frac{1}{2}(p_{data}(x)+p_g(x))}\right] \\ &\quad + E_{x\sim p_g}\left[\tfrac{p_g(x)}{\frac{1}{2}(p_{data}(x)+p_g(x))}\right] - 2\log 2. \end{aligned} \tag{3}$$

The definition of KullbackLeibler (KL) divergence and Jensen-Shannon (JS) divergence between two probabilistic distributions $p(x)$ and $q(x)$ are defined as

$$KL(p\| q) = \int p(x) \log \frac{p(x)}{q(x)} dx, \tag{4}$$

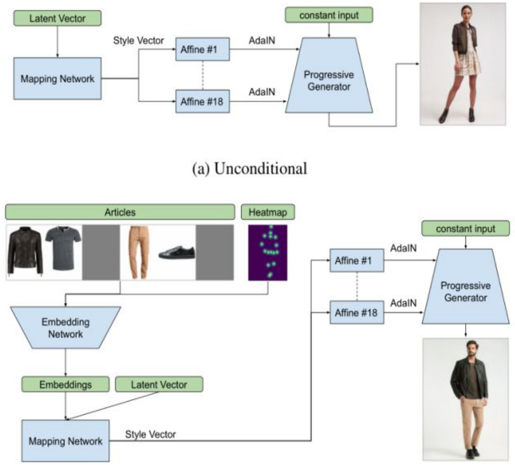
$$JS(p\| q) = \frac{1}{2} KL(p\| \frac{p+q}{2}) + \frac{1}{2} KL(q\| \frac{p+q}{2}). \tag{5}$$

Therefore, (3) is equal to

$$\begin{aligned} C(G) &= KL(p_{data}\| \tfrac{p_{data}+p_g}{2}) + KL(p_g\| \tfrac{p_{data}+p_g}{2}) \\ &\quad -2\log 2 \\ &= 2JS(p_{data}\| p_g) - 2\log 2. \end{aligned} \tag{6}$$

Thus, the objective function of GANs is related to both KL divergence and JS divergence.

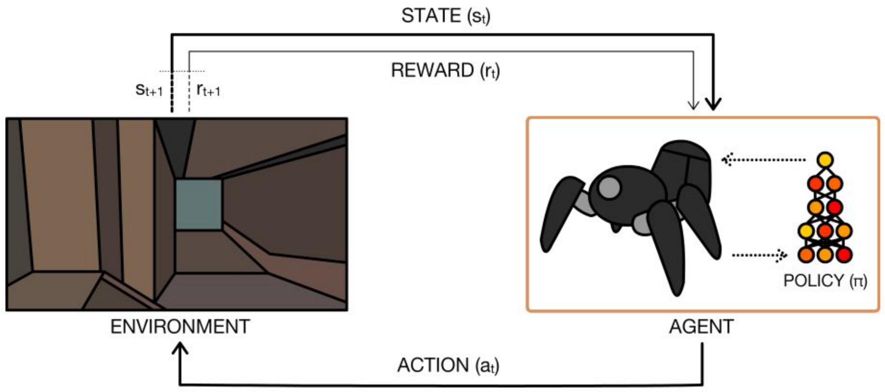
Conditional



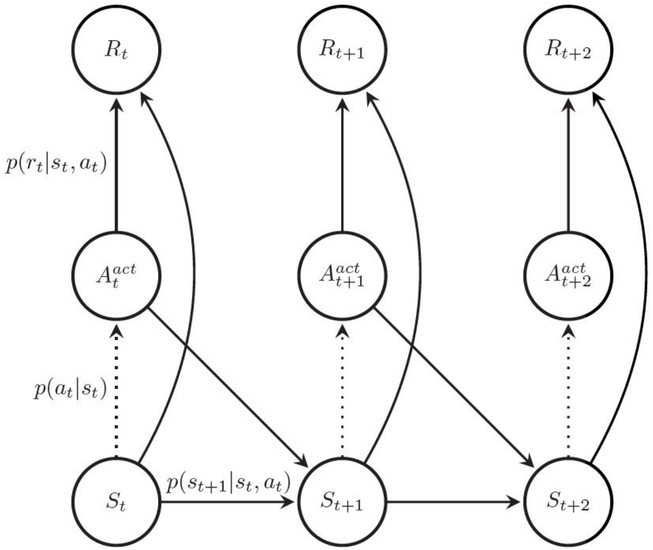
(a) Unconditional

GAN Survey (including 400+ variants): https://arxiv.org/pdf/2001.06937.pdf

Deep Reinforcement learning



Directed Acyclic Graph
For Markov Decision Process



There are four key ingredients for a RL system:

Environment: Physical world in which the agent operates

State: Current situation of the agent

Reward: Feedback from the environment

Policy: Method to map agent's state to actions

DRL+PGM survey: https://arxiv.org/pdf/1906.10025.pdf

Partially Observable MDP

Partially Observable Markov Decision process with its DAG representation shows that the agent could only observe the state partially by observing $\boldsymbol{O_t}$ through a non invertible function of the next state $\boldsymbol{S_{t+1}}$ and the action $\boldsymbol{a_t}$, as indicated the Figure by $\boldsymbol{P(O_t|\, S_{t+1}, a_t)}$

For POMDP, belief state $\boldsymbol{b_t}$ $\quad \sum_S b(S_t) = 1$

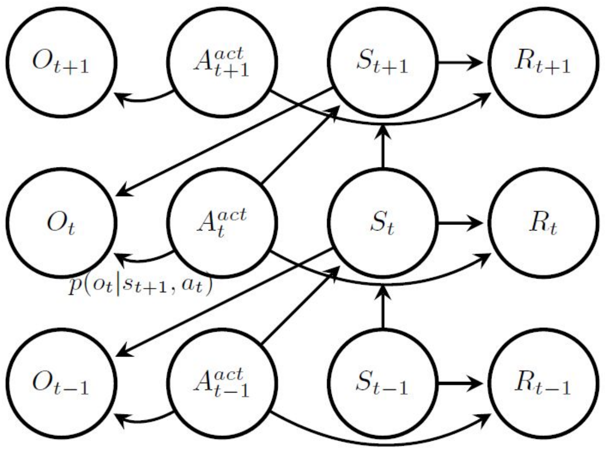


Fig. 4. Probabilistic Graphical Model for POMDP

The distributions on other edges are omitted in last slide

$$
\begin{aligned}
& b_{t+1}(s_{t+1}) \\
=& p(s_{t+1} \mid o_t, a_t, b_t) \\
=& p(o_t \mid s_{t+1}, a_t) \frac{\sum_{s_t} p(s_{t+1} \mid s_t, a_t) p(s_t \mid a_t, b_t)}{p(o_t \mid a_t, b_t)}
\end{aligned}
$$

RBM in DRL

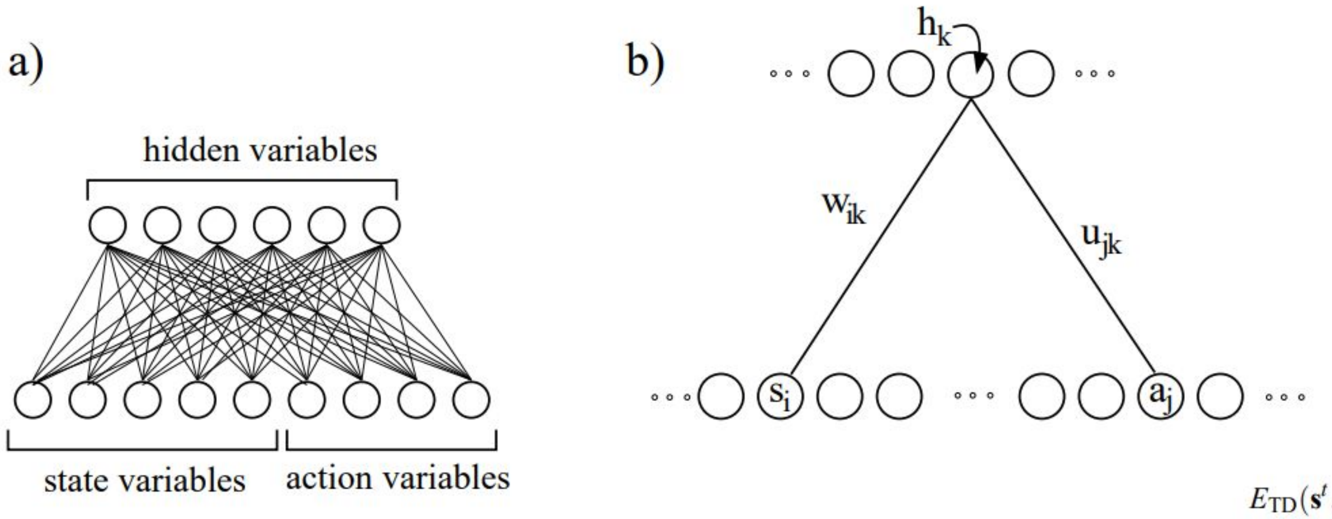
Undirected graph defines a joint probability distribution over state and action pairs through hidden state

$$p(a|s) = 1/Z(s)e^{-F(s,a)/T} = 1/Z(s)e^{Q(s,a)/T}$$

$$P(\mathbf{v},\mathbf{h}) = \frac{\exp(-E(\mathbf{v},\mathbf{h}))}{\sum_{\widehat{\mathbf{v}},\widehat{\mathbf{h}}}\exp(-E(\widehat{\mathbf{v}},\widehat{\mathbf{h}}))},$$

Approximate the value function of an MDP with the negative free energy of the restricted Boltzmann machine.

Function Approximation Q(S, a)



$$\exp(-F(\mathbf{v})) = \sum_{\mathbf{h}} \exp(-E(\mathbf{v},\mathbf{h})),$$

$$P(\mathbf{v}) = \frac{\exp(-F(\mathbf{v}))}{\sum_{\widehat{\mathbf{v}}}\exp(-F(\widehat{\mathbf{v}}))}.$$

Temporal Difference Learning

$$E_{\mathrm{TD}}(\mathbf{s}^t,\mathbf{a}^t) = \left[r^t + \gamma Q(\mathbf{s}^{t+1},\mathbf{a}^{t+1})\right] - Q(\mathbf{s}^t,\mathbf{a}^t). \qquad \frac{\partial F(\mathbf{v})}{\partial w_{ik}} = -v_i \langle h_k \rangle_{P(h_k|\mathbf{v})}$$

The state and action variables will be assumed to be discrete, and will be represented by the visible binary variables of the restricted Boltzmann machine.

The negative free energy to approximate the state action value function Q(s, a).