1. History

1.1 Perceptron

Threshold Unit
$f[w,b](x) = \text{sign}(x\cdot w + b)$ where $x\cdot w := \sum_{i=1}^n x_i w_i$

Decision Boundary
$x\cdot w + b \neq 0 \Leftrightarrow \frac{x\cdot w}{\|w\|} + \frac{b}{\|w\|} \neq 0$

$x\cdot w + b = \begin{bmatrix}x\\1\end{bmatrix}\cdot\begin{bmatrix}w\\b\end{bmatrix} =: \tilde x\cdot\tilde w,\quad \tilde x,\tilde w\in\mathbb{R}^{n+1}$

Geometric Margin
$\gamma[w,b](x,y) := \frac{y(x\cdot w+b)}{\|w\|}$

Maximum Margin Classifier
$(w^*,b^*) \in \text{argmax}_{w,b}\,\gamma[w,b](S)$ with $\gamma[w,b](S) := \min_{(x,y)\in S}\gamma[w,b](x,y)$

Perceptron Learning
if $f[w,b](x)\neq y$: update $w\leftarrow w+yx$, and $b\leftarrow b+y$
$w_0\in\text{span}(x_1,\dots,x_s)\Rightarrow w_t\in\text{span}(x_1,\dots,x_s)(\forall t)$

Convergence
$\exists w, \|w\|=1$, that $\gamma[w](S)=\gamma>0 \Rightarrow w_t\cdot w\geq t\gamma$.
$R=\max_{x\in S}\|x\| \Rightarrow \|w_t\|\leq R\sqrt t$
$\cos\angle(u,w_t) = \frac{u\cdot w_t}{\|w_t\|} \geq \frac{\sqrt t\gamma}{tR} = \frac{\gamma}{R\sqrt t} \leq 1 \Rightarrow t\leq\frac{R^2}{\gamma^2}$

Covers Theorem
$C(s+1,n) = 2\sum_{i=0}^{n-1}\binom{s}{i}$
$C(S,n)$: Number of ways to separate S with n dimensions. $C(s,n)=2s$ for $s\leq n$
Phase transition at $s=2n$. For $s>2n$ empty version space is the exception.

1.2 Hopfield Networks

Hopfield Model
$E(X) = -\frac12\sum_{i\neq j} w_{ij}X_iX_j + \sum_i b_iX_i$ where $X_i\in\{-1,+1\}$
$w_{ij}=w_{ji}\ (\forall i,j)$, $w_{ii}=0\ (\forall i)$: Interaction strengths

Hebbian Learning
$x^t\in\{\pm1\}^n\ (1\leq t\leq s)$, $w_{ij} = \frac1n\sum_{t=1}^s x_i^t x_j^t$, $W=\frac1n\sum_{t=1}^s x^t(x^t)^\top$

2. Feedforward Networks

2.1 Linear Models

Linear regression
$h[w](S) = \frac{1}{2s}\|Xw-y\|^2$, $\nabla h = 2X^\top Xw - 2X^\top y$

Moore-Penrose inverse solution
$w^* = X^+y \in \text{argmin}_w\, h[w]$ where $X^+ := \lim_{\epsilon\to0}(X^\top X+\epsilon I)^{-1}X^\top$

SGD update
$w_{t+1} := w_t + \eta(y_{i_t} - w_t^\top x_{i_t})x_{i_t}$ with $i_t \overset{\text{iid}}{\sim}$ Uniform$(1,\dots,s)$

Gaussian noise model
$y_i = w^\top x_i+\epsilon_i$, $\epsilon_i\sim\mathcal N(0,\sigma^2)$. Least squares $\equiv$ neg log likelihood.

Ridge regression
$h_\lambda[w] := h[w]+\frac\lambda2\|w\|^2$, $w^*=(XX^\top+\lambda I)^{-1}X^\top y$

Logistic function
$\sigma(z)=\frac{1}{1+e^{-z}}$, $\sigma(z)+\sigma(-z)=1$, $\sigma'=\sigma(1-\sigma)$

Cross entropy loss
$\ell(y,z) = -y\log\sigma(z) - (1-y)\log(1-\sigma(z)) = -\log\sigma((2y-1)z)$

Logistic regression gradient
$\nabla\ell_i = [\sigma(w^\top x_i)-y_i]x_i$

2.2 Feedforward Networks

Generic feedforward layer
$F:\mathbb R^{m(n+1)}\times\mathbb R^n\to\mathbb R^m$, $F[\theta](x) := \varphi(Wx+b)$, $\theta := \vec{(W,b)}$

Composition of layers
$G = F_L[\theta_L]\circ\cdots\circ F_1[\theta_1]$ where $F_l[W^l,b^l](x) := \varphi_l(W^lx+b^l)$

Layer activations
$x^l := (F_l\circ\cdots\circ F_1)(x) = F_l(x^{l-1})$, $x^0=x$, $x^L=F(x)$

Softmax function
$\text{softmax}(z)_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$, $\ell(y;z) = \left[-zy+\log\sum_j e^{z_j}\right]\frac{1}{\ln 2}$

Residual layer
$F[W,b](x) = x+[\varphi(Wx+b)-\varphi(0)]$, therefore $F[0,0]=\text{id}$

2.3 Sigmoid Networks

Sigmoid/Tanh activations
$\sigma(z)=\frac{1}{1+e^{-z}}$, $\tanh(z)=2\sigma(2z)-1$, $\tanh'(z)=1-\tanh^2(z)$

Barron's Theorem
For f with finite $C_f := \int\|\omega\||\hat f(\omega)|d\omega$, ∃ MLP g with width m: $\int_B(f-g_m)^2\mu(dx)\leq O(1/m)$

2.4 ReLU Networks

ReLU activation
$\varphi(z) := (z)_+ := \max\{0,z\}$. ReLU networks are universal function approximators.

Zaslavsky: Connected regions
$R(H)\leq\sum_{i=0}^{\min\{n,m\}}\binom{m}{i} := R(m)$

Montufar: Connected regions
$R(m,L)\geq R(m)\lfloor\frac mn\rfloor^{n(L-1)}$ (L: layers, m: width)

3. Gradient-Based Learning

3.1 Backpropagation

Parameter derivatives
$\frac{\partial x_i^l}{\partial w_{ij}^l} = \dot\varphi_i^l x_j^{l-1}$, $\frac{\partial x_i^l}{\partial b_i^l} = \dot\varphi_i^l$ where $\dot\varphi_i^l := \varphi'^l((w_i^l)^\top x^{l-1}+b_i^l)$

Loss derivatives
$\frac{\partial h}{\partial w_{ij}^l} = \delta_i^l\dot\varphi_i^l x_j^{l-1}$, $\frac{\partial h}{\partial b_i^l} = \delta_i^l\dot\varphi_i^l$ with $\delta_i^l = \frac{\partial h}{\partial x_i^l}\dot\varphi_i^l$

3.2 Gradient Descent

GD update & flow
$\theta_{t+1}=\theta_t-\eta\nabla h(\theta_t)$, ODE: $\frac{d\theta}{dt} = -\nabla h(\theta)$

L-smoothness
$\|\nabla h(\theta_1)-\nabla h(\theta_2)\|\leq L\|\theta_1-\theta_2\|$, $\lambda_{\max}(\nabla^2h)\leq L$
$\ell(w)-\ell(w')\leq\nabla\ell(w')^\top(w-w')+\frac L2\|w-w'\|_2^2$

Polyak-Łojasiewicz
$\frac12\|\nabla h(\theta)\|^2\geq\mu(h(\theta)-\min h)\ (\forall\theta)$

Convergence rate
$\eta=1/L\Rightarrow t=\frac{2L}{\epsilon^2}(h(\theta^0)-\min h)$ for ϵ-critical. With PL: $h(\theta^t)-\min h\leq(1-\frac\mu L)^t(h(\theta^0)-\min h)$

3.3 Acceleration and Adaptivity

Heavy ball momentum
$\theta_{t+1}=\theta_t-\eta\nabla h(\theta_t)+\beta(\theta_t-\theta_{t-1})$

Nesterov acceleration
$\tilde\theta_{t+1}=\theta_t+\beta(\theta_t-\theta_{t-1})$, $\theta_{t+1}=\tilde\theta_{t+1}-\eta\nabla h(\tilde\theta_{t+1})$

AdaGrad
$\theta_{t+1}^i=\theta_t^i-\eta_t^i\partial_ih(\theta^t)$, $\nu_t^i=\nu_{t-1}^i+[\partial_ih]^2$, $\eta_t^i=\frac{\eta}{\sqrt{\nu_t^i+\epsilon}}$

Adam
$g_t^i=\beta g_{t-1}^i+(1-\beta)\partial_ih$, $\nu_t^i=\alpha\nu_{t-1}^i+(1-\alpha)[\partial_ih]^2$, $\theta_{t+1}^i=\theta_t^i-\frac{\eta}{\sqrt{\nu_t^i+\epsilon}}g_t^i$

3.4 SGD

SGD update & variance
$\theta_{t+1}=\theta_t-\eta\nabla h(\theta^t)(x_{i_t},y_{i_t})$, $V[\theta]=\frac1s\sum_{i=1}^s\|\nabla h(S)-\nabla h(x_i,y_i)\|^2$

SGD convergence
General: $O(1/\sqrt t)$, Strongly convex: $O(\log t/t)$, Additionally smooth: $O(1/t)$

3.5 Function properties

Convexity
$\ell(\lambda w+(1-\lambda)w')\leq\lambda\ell(w)+(1-\lambda)\ell(w')$, $\ell''(x)\geq0$

Strong convexity
$\ell(w)\geq\ell(w')+\nabla\ell(w')^\top(w-w')+\frac\mu2\|w-w'\|_2^2$, $\ell''(x)\geq\mu$

4. Convolutional Networks

4.1 Convolutions

Convolution definition
$(f*g)(u) := \int_{-\infty}^\infty g(u-t)f(t)dt = \int_{-\infty}^\infty f(u-t)g(t)dt$

Fourier property
$\mathcal F(f*g)=\mathcal F(f)\cdot\mathcal F(g)$

Discrete convolution
$(f*g)[u] := \sum_{t=-\infty}^\infty f[t]g[u-t]$

Cross-correlation
$(g\star f)[u] := \sum_{t=-\infty}^\infty g[t]f[u+t]$

Toeplitz matrices
$(f*g)=\text{Toeplitz-Matrix}(g)f$

4.2 Convolutional Networks

Conventions
Padding: Add zeros around input. Stride: Step size of convolution.

Max-Pooling
Take maximum value in windows (size r)

ConvNets for Images
$y[r][s,t]=\sum_u\sum_{s',t'}w[r,u][s',t']\cdot x[u][s+s',t+t']$ (r: output channel, u: input channel)

Parameters count
$D=\#r\cdot\#u\cdot\#s'\cdot\#t'$ (channels × window size)

4.3 NLP with ConvNets

Word embedding
word $\omega\mapsto x_\omega\in\mathbb R^n$

Conditional log-bilinear model
$P(\nu|\omega)=\frac{\exp[x_\omega^\top y_\nu]}{\sum_\mu\exp[x_\omega^\top y_\mu]}$, $\ell_{\omega,\nu} = -x_\omega^\top y_\nu+\ln\sum_\mu\exp[x_\omega^\top y_\mu]$

Negative sampling
$\tilde\ell_{\omega,\nu}=-\ln\sigma(x_\omega^\top y_\nu)-\beta\mathbb E_{\mu\sim D}\ln(1-\sigma(x_\omega^\top y_\mu))$

5. Recurrent Networks

5.1 Simple RNNs

Time evolution
$z_t := F[\theta](z_{t-1},x_t)$, $z_0:=0$

Output map
$\hat y_t := G[\psi](z_t)$

RNN parameterization
$F[U,V](z,x) := \varphi(Uz+Vx)$, $G[W](z) := \Phi(Wz)$

BPTT
$\frac{\partial h}{\partial v_{ij}} = \sum_{t=1}^T\frac{\partial h}{\partial z_i^t}\dot\varphi_i^t x_j^t$, $\frac{\partial h}{\partial u_{ij}}=\sum_{t=1}^T\frac{\partial h}{\partial z_i^t}\dot\varphi_i^t z_j^{t-1}$

Spectral norm
$\|A\|_2=\max_{x:\|x\|=1}\|Ax\|_2=\sigma_1(A)$

Gradient norms
$\frac{\partial z_T}{\partial z_0} = \dot\Phi^TU\cdots\dot\Phi^1U$. Vanishes if $\sigma_1(U)<1/\kappa$, explodes if $\sigma_1(U)$ too large.

Bidirectional RNNs
$\hat y_t=\Phi(Wz_t+\tilde W\tilde z_t)$

5.2 Gated Memory

LSTM
$z_t := \sigma(F\tilde x_t)\odot z_{t-1}+\sigma(G\tilde x_t)\odot\tanh(V\tilde x_t)$, $\tilde x_t := [x_t,\ell_t]$, $\ell_{t+1}=\sigma(H\tilde x_t)\odot\tanh(Uz_t)$

GRU
$z_t=(1-\sigma)\odot z_{t-1}+\sigma\odot\tilde z_t$, $\sigma := \sigma(G[x_t,z_{t-1}])$, $\tilde z_t := \tanh(V[\ell_t\odot z_{t-1},x_t])$

5.3 Linear Recurrent Models

Linear state evolution
$z_{t+1}=Az_t+Bx_t$

Diagonal form
$A=P\Lambda P^{-1}$, $\Lambda := \text{diag}(\lambda_1,\dots,\lambda_m)$, $\lambda_i\in\mathbb C$

Stability
$\max_j|\lambda_j|\leq1$

Initialization
$\lambda_i=\exp(-\exp(\xi_i)+i\theta_i)$, $\theta_i\sim$ Uni$[0;2\pi]$, $r_i\sim$ Uni$[I]$

Advantages
(i) clear long/short range dependencies (ii) no channel mixing required (iii) parallelizable training

6. Attention and Transformers

6.1 Attention

Attention mixing

$y_s := \sum_t a_{st} W x_t$, $a_{st} \geq 0$, $\sum_t a_{st} = 1$, $Y = WXA^\top$

Query-key matching

$Q = U_Q X$, $K = U_K X$, $Q^\top K = X^\top U_Q^\top U_K X$ (rank $\leq q$)

Softmax attention

$A = \text{softmax}(\beta Q^\top K)$, $a_{st} = \frac{e^{\beta [Q^\top K]_{st}}}{\sum_r e^{\beta [Q^\top K]_{sr}}}$, usually $\beta = 1/\sqrt{q}$

Feature transformation

$X \mapsto Y \mapsto F(Y)$, $F(\theta)(Y) = (F(y_1), \ldots, F(y_T))$

Positional encoding

$p_{tk} = \begin{cases} \sin(t\omega_k) & k \text{ even} \\ \cos(t\omega_k) & k \text{ odd} \end{cases}$, $\omega_k = C^{k/K}$

Transformer architecture

Self-attention: attend to its own values in the past. Cross-attention: decoder attends to encoder output.

Vision transformer patch embedding

$\text{patch}_t \mapsto x_t := V\vec{(\text{patch}_t)} \in \mathbb{R}^n$ with $V \in \mathbb{R}^{n \times (qp^2)}$

GELU activation

$\varphi(z) = z \Pr(z \leq Z)$, $Z \sim \mathcal{N}(0, 1)$

7. Geometric Deep Learning

7.1 Sets and Points

Order-invariance

$f(x_1, \ldots, x_M) = f(x_{\pi_1}, \ldots, x_{\pi_M})$ $\forall \pi \in S_M$

Equivariance

$f(x_{\pi_1}, \ldots, x_{\pi_M}) = (y_{\pi_1}, \ldots, y_{\pi_M})$

Deep Sets model

$f(x_1, \ldots, x_M) = f^l(\sum_{m=1}^M \varphi(x_m))$

Equivariant map

$f^l : \mathbb{R} \times \mathbb{R}^N \to Y$, $(x_m, \sum_{k=1}^M \varphi(x_k)) \mapsto y_m$

7.2 Graph Conv Networks

Feature & adjacency

$X = [x_1^\top; \ldots; x_M^\top]$, $A = (a_{nm})$ with $a_{nm} = 1$ if $\{v_n, v_m\} \in E$

Graph invariance

$f(X, A) = f(PX, PAP^\top)$ $\forall P$

Graph equivariance

$f(X, A) = Pf(PX, PAP^\top)$ $\forall P$

Message passing

$\varphi(x_m, X_m) = \varphi(x_m, \bigoplus_{X_m} \Phi(x))$, $\bigoplus$ permutation-invariant

Normalized adjacency

$\bar{A} = D^{-1/2}(A + I)D^{-1/2}$, $D = \text{diag}(d_m)$, $d_m = 1 + \sum_n a_{nm}$

GCN layer

$X^+ = \sigma(\bar{A}XW)$

7.3 Spectral Graph Theory

Graph Laplacian

$L = D - A$, $(Lx)_n = \sum_m a_{nm}(x_n - x_m)$

Normalized Laplacian

$\tilde{L} = I - D^{-1/2} A D^{-1/2}$

Graph Fourier transform

$L = U\Lambda U^\top$, convolution: $x * y = U((U^\top x) \odot (U^\top y))$

Polynomial kernels

$U(\sum_{k=0}^K \alpha_k \Lambda^k)U^\top = \sum_{k=0}^K \alpha_k L^k$

7.4 Attention GNNs

Attention coupling

$q_{ij} = \text{softmax}(f^l(u^\top(Vx_i; Vx_j; x_{ij})))$ s.t. $\sum_j A_{ij} q_{ij} = 1$

Attention propagation

$X^+ = \sigma(QXW)$

8. Tricks of the Trade

8.1 Initialization

Random initialization

$\theta_i^0 \sim \mathcal{N}(0, \sigma_i^2)$ or $\theta_i^0 \sim \text{Uniform}[-\sqrt{3}\sigma_i; \sqrt{3}\sigma_i]$

LeCun initialization

$w_{ij} \sim \text{Uniform}[-a; a]$, $a := 1/\sqrt{n}$, $b_i = 0$. Stabilizes variance.

Glorot initialization

$w_{ij} \sim \text{Uniform}[-\sqrt{3}\epsilon; \sqrt{3}\epsilon]$, $\epsilon := 2/(n + m)$. Stabilizes gradient variance.

He initialization

$w_{ij} \sim \mathcal{N}(0, \epsilon)$ or $\text{Uniform}[-\sqrt{3}\epsilon; \sqrt{3}\epsilon]$, $\epsilon := 2/n$. For ReLU (half units active).

Orthogonal initialization

$\frac{1}{\sqrt{m}} W \sim \text{Uniform}(O(m))$ s.t. $W^\top W = WW^\top = mI$

8.2 Weight Decay

L_2 regularization

$\mu(\theta) = \frac{\mu}{2} \|\theta\|^2$

GD with weight decay

$\dot{\theta} = -\eta \nabla E(\theta) - \eta\mu\theta$

Local loss landscape

$\theta_\mu^* = (H + \mu I)^{-1} H\theta^*$. Minimum shrunk along small eigenvalue directions.

Optimal weight decay

$\mu = \sigma^2/u^2$. Inverse proportional to signal-to-noise ratio.

8.3 Dropout

Dropout as Ensembling

$p(y|x) = \sum_{b \in \{0,1\}^R} p(b) p(y|x; b)$, $p(b) = \prod_i \phi_i^{b_i}(1 - \phi_i)^{1-b_i}$

Weight scaling for inference

$\tilde{w}_{ij} \leftarrow \phi_{ij} w_{ij}$

8.4 Normalization

Batch normalization

$\bar{f} = \frac{f - \mathbb{E}[f]}{\sqrt{\mathbb{V}[f]}}$, $\bar{f}[\mu, \epsilon] = \mu + \epsilon \bar{f}$

Weight normalization

$w := \frac{\epsilon}{\|v\|_2} v$, $\partial_\epsilon E = \nabla_w E \cdot \frac{v}{\|v\|_2}$

Layer normalization

$\tilde{f}_i = \frac{f_i - \mathbb{E}[f]}{\sqrt{\mathbb{V}[f]}}$, $\mathbb{E}[f] = \frac{1}{m} \sum_i f_i$, $\mathbb{V}[f] = \frac{1}{m} \sum_i (f_i - \mathbb{E}[f])^2$

8.5 Model Distillation

Tempered cross entropy

$\ell(x) = \sum_y \frac{q \exp[F_y/T]}{\sum_\nu \exp[F_\nu/T]} [\frac{1}{T} G_y - \ln \sum_\nu \exp[G_\nu/T]]$

Distillation gradient

$\frac{\partial \ell}{\partial G_y} = \frac{1}{T}[\frac{e^{qF_y/T}}{\sum_\nu e^{F_\nu/T}} - \frac{qe^{G_y/T}}{\sum_\nu e^{G_\nu/T}}]$

9. Theory

9.1 Infinite Width (NTK)

Neural tangent kernel

$K(\mathbf{x}, \mathbf{x}') := \langle \nabla_\theta F(\mathbf{x}; \theta), \nabla_\theta F(\mathbf{x}'; \theta) \rangle$

Gradient flow

$\dot{F}_t = -K_t(F_t - Y)$, $K_t := K(X, X; \theta_t)$

Linearized network

$\bar{F}(x) = F(x; \theta_0) + \nabla_\theta F(x; \theta_0)^\top(\theta - \theta_0)$

Lazy training

If width $n \to \infty$: $K_t \to K_0$ (deterministic), $F_t \to \bar{F}_t$ (linear dynamics).

Kernel regression solution

$\bar{F}_\infty = K_0(K_0 + \lambda I)^{-1} Y$

Function space

$\bar{F} \in \mathcal{H}_K$ (RKHS), $\|\bar{F}\|_{\mathcal{H}_K}^2 = \theta^\top \theta$

9.2 Bayesian DNNs

Prior

$p(\theta) = \mathcal{N}(0, \sigma_p^2 I)$

Likelihood

$p(y|x, \theta) = \mathcal{N}(F(x; \theta), \sigma^2)$

Posterior (Laplace approx.)

$p(\theta|\mathcal{D}) \approx \mathcal{N}(\theta^*, H^{-1})$, $H = \nabla^2 \mathcal{L}(\theta^*)$

Predictive distribution

$p(y|x, \mathcal{D}) = \int p(y|x, \theta) p(\theta|\mathcal{D}) d\theta$

9.3 GPs & Infinite Width

NNGP kernel

$K_{\text{NNGP}}(\mathbf{x}, \mathbf{x}') = \mathbb{E}_\theta[F(\mathbf{x}; \theta) F(\mathbf{x}'; \theta)]$ as width $\to \infty$

GP prior

$F \sim \mathcal{GP}(0, K_{\text{NNGP}})$

Recursive kernel

$K^{(l+1)} = \sigma_w^2 \mathbb{E}_{(u,v) \sim \mathcal{N}(0, \Sigma^{(l)})}[\phi(u)\phi(v)] + \sigma_b^2$

GP posterior

$F|\mathcal{D} \sim \mathcal{GP}(\mu_*, K_*)$, $\mu_* = K_{*X}(K_{XX} + \sigma^2 I)^{-1} Y$

9.4 Statistical Learning Theory

Generalization gap

$\mathcal{R}(f) - \hat{\mathcal{R}}(f) = \mathbb{E}_{x,y}[\ell(f(x), y)] - \frac{1}{n} \sum_i \ell(f(x_i), y_i)$

PAC bound

$\mathbb{P}[\mathcal{R}(f) - \hat{\mathcal{R}}(f) \leq \epsilon] \geq 1 - \delta$

VC dimension

Largest n s.t. $\exists$ data shattered by $\mathcal{F}$. For linear: $d + 1$.

Rademacher complexity

$\mathcal{R}_n(\mathcal{F}) = \mathbb{E}_{\sigma,S}[\sup_{f \in \mathcal{F}} \frac{1}{n} \sum_i \sigma_i f(x_i)]$

Generalization bound

$\mathcal{R}(f) \leq \hat{\mathcal{R}}(f) + 2\mathcal{R}_n(\mathcal{F}) + \sqrt{\frac{\ln(1/\delta)}{2n}}$

Bias-variance tradeoff

$\mathbb{E}[(f(x) - y)^2] = \text{Bias}^2 + \text{Var} + \sigma^2$

Double descent

Test error: U-shaped in classical regime, second descent in overparameterized regime.

9.5 Loss Landscape

Critical points

$\nabla \mathcal{L}(\theta^*) = 0$. Local min: $H \succeq 0$. Saddle: H indefinite.

Sharpness

$\lambda_{\max}(H)$. Flat minima $\to$ better generalization.

Mode connectivity

Local minima connected by paths of low loss.

Lottery ticket hypothesis

Sparse subnetworks can match full network performance if initialized correctly.

10. Generative Models

10.1 Variational Auto Encoders

Generative model

$p_\theta(x, z) = p_\theta(x|z)p(z)$, $p(z) = \mathcal{N}(0, I)$

ELBO

$\ln p_\theta(x) \geq \mathbb{E}_{q_\phi(z|x)}[\ln p_\theta(x|z)] - D_{KL}(q_\phi(z|x) \| p(z))$

Encoder

$q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi^2(x))$

Decoder

$p_\theta(x|z) = \mathcal{N}(\mu_\theta(z), \sigma^2 I)$ or Bernoulli

Reparameterization trick

$z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon$, $\epsilon \sim \mathcal{N}(0, I)$

KL divergence (Gaussian)

$D_{KL} = \frac{1}{2} \sum_j (\mu_j^2 + \sigma_j^2 - \ln \sigma_j^2 - 1)$

β-VAE

$\mathcal{L} = \mathbb{E}_q[\ln p(x|z)] - \beta D_{KL}(q\|p)$. $\beta > 1$ for disentanglement.

10.2 Generative Adversarial Networks

Generator

$G : z \mapsto x$, $z \sim p(z)$

Discriminator

$D : x \mapsto [0, 1]$, probability that x is real

GAN objective

$\min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}}[\ln D(x)] + \mathbb{E}_{z \sim p(z)}[\ln(1 - D(G(z)))]$

Optimal discriminator

$D^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}$

Generator loss (non-saturating)

$\mathcal{L}_G = -\mathbb{E}_z[\ln D(G(z))]$

Mode collapse

G produces limited diversity. Solutions: minibatch discrimination, feature matching.

Wasserstein GAN

$\min_G \max_{D \in 1\text{-Lip}} \mathbb{E}_x[D(x)] - \mathbb{E}_z[D(G(z))]$

Gradient penalty (WGAN-GP)

$\lambda \mathbb{E}_{\hat{x}}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2]$, $\hat{x} = \alpha x + (1 - \alpha)G(z)$

10.3 Denoising Diffusion

Forward process

$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$

Marginal

$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I)$, $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$

Reverse process

$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2 I)$

Training objective

$\mathcal{L} = \mathbb{E}_{t,x_0,\epsilon}[\|\epsilon - \epsilon_\theta(x_t, t)\|^2]$

Sampling

$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)) + \sigma_t z$

Score function

$\nabla_x \ln p(x) \approx -\frac{\epsilon_\theta(x,t)}{\sqrt{1-\bar{\alpha}_t}}$

Classifier-free guidance

$\tilde{\epsilon} = (1+w)\epsilon_\theta(x_t, t, c) - w\epsilon_\theta(x_t, t)$

DDIM (deterministic)

$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta}{\sqrt{\bar{\alpha}_t}}) + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon_\theta$

11. Ethics

11.1 Adversarial Examples

Adversarial perturbation

$x' = x + \delta$, $\|\delta\|_p \leq \epsilon$, $F(x') \neq F(x)$

FGSM (Fast Gradient Sign Method)

$\delta = \epsilon \cdot \mathrm{sign}(\nabla_x \mathcal{L}(x, y))$

PGD (Projected Gradient Descent)

$x^{t+1} = \Pi_{B_\epsilon(x)}[x^t + \alpha \cdot \mathrm{sign}(\nabla_x \mathcal{L}(x^t, y))]$

Adversarial training

$\min_\theta \mathbb{E}_{x,y}[\max_{\|\delta\| \leq \epsilon} \mathcal{L}(x+\delta, y; \theta)]$

Certified robustness

Provable guarantees via randomized smoothing, interval bound propagation.

Transferability

Adversarial examples often transfer between models.