Deep Learning Cheat Sheet

Authors

January 2026

1 History

1.1 Perceptron

Threshold Unit
$f[w,b](x) = \mathrm{sign}(x^\top w + b)$

Decision Boundary
$x^\top w + b \neq 0 \Leftrightarrow \frac{x*w}{\|w\|} + \frac{b}{\|w\|} \neq 0$
$x^\top w - b = \begin{bmatrix} x & -1 \end{bmatrix} \cdot \begin{bmatrix} w \\ b \end{bmatrix} =: \tilde{x}^\top \tilde{w}, \quad \tilde{x}, \tilde{w} \in \mathbb{R}^{n+1}.$

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

Maximum Margin Classifier
$(w^*, b^*) \in \mathrm{argmax}_{w,b} \gamma[w,b](S)$,
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 $b \leftarrow b + y$
$w_0 \in \mathrm{span}(x_1, ..., x_s) \Rightarrow w_t \in \mathrm{span}(x_1, ..., x_s)$ (forall t)

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

Cover's 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
$C(s,n) = 2^s$ for $s \leq n$
Phase transition at $s = 2n$. For $s > 2n$ empty version space is the exception, otherwise the rule.

1.1.1 Hopfield Networks

Hopfield Model
$E(X) = -\frac{1}{2}\sum_{i\neq j} w_{ij} X_i X_j + \sum_i b_i X_i$
where $X_i \in \{-1, +1\}$
$w_{ij} = w_{ji}$ (forall i, j), $w_{ii} = 0$ (forall i): Interaction strengths

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

1.2 Feedforward Networks

1.2.1 Linear Models

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

Moore-Penrose inverse solution
$w^* = X^* y \in \mathrm{argmin}_w h[w]$
where $X^* := \lim_{\delta\to 0} \left(X^\top X + \delta I\right)^{-1} X^\top$

Stochastic gradient descent update
$w_{t+1} := w_t + \eta\left(y_{i_t} - w_t^\top x_{i_t}\right) x_{i_t}$
with $i_t \sim \mathrm{Uniform}(1, ..., s)$

Gaussian noise model
$y_i = w^\top x_i + \varepsilon_i, \varepsilon_i \sim \mathrm{N}(0, \sigma^2)$
Least squares equivalent to negative log likelihood of gaussian noise model

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

Logistic function
$\sigma(z) = \frac{1}{1+e^{-z}}$
$\sigma(z) + \sigma(-z) = 1$
$\sigma' = \sigma(1-\sigma), \sigma'' = \sigma(1-\sigma)(1-2\sigma)$

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

Logistic regression with cross entropy loss
$\nabla \ell_i = [\sigma(w^\top x_i) - y_i] x_i$

1.2.2 Feedforward Networks

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

Composition of layers
$G = F^{L}[\theta^L] @ ... @ F^1[\theta^1]$
where $F^l[W^l, b^l](x) := \varphi^{l(W^l x + B^l)}$

Layer activations
$x^l := \left(F^l @ ... @ F^1\right)(x) = F^{l(x^{l-1})}$
identifying $x^0 = x, x^L = F(x)$

Softmax function
$\mathrm{softmax}(z)_i = \frac{e^{z_i}}{\sum_j} e^{z_j}, \mathrm{softmax}(A)_{ij} = \frac{e^{A_{ij}}}{\sum_k} e^{A_{ik}}$
$\ell(y,z) = \frac{-z_y + \log\sum_j e^{z_j}}{\ln 2}$

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

Skip connection: Concatenate previous layer back in

1.2.3 Sigmoid Networks

Sigmoid activation
$\varphi(z) := \sigma(z) = \frac{1}{1+e^{-z}}$

Hyperbolic tangent activation
$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} = 2\sigma(2z) - 1$
$\tanh'(z) = 1 - \tanh^2(z)$

Baron's Theorem: Approximation error
For f with finite $C_f := \int \|\omega\| \, |\hat{f}(\omega)| \, \mathrm{d}\omega < \infty$ there exists MLP g with one hidden layer of width m that:
$\int_B \left(f(x) - g_{m(x)}\right)^2 \mu(\mathrm{d}x) \leq O\left(\frac{1}{m}\right)$

1.2.4 ReLU Networks

ReLU activation
$\varphi(z) := (z)_+ := \max\{0, z\}$

ReLU networks are universal function approximators

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

Montufar: Connected regions in ReLU network
$R(m, L) \geq R(m) \left(\frac{m}{n}\right)^{n(L-1)}$, L: layers, m: width

1.3 Gradient-Based Learning

1.3.1 Backpropagation

Parameter derivatives for ridge function layers
$\frac{\partial x_i^l}{\partial w_{ij}^l} = \dot{\varphi}_i^l x_j^{l-1}$,
$\dot{\varphi}_i^l := \dot{\varphi}^l\left(\left(w_i^l\right)^\top x^{l-1} + b_i^l\right)$
$\frac{\partial x_i^l}{\partial b_i^l} = \dot{\varphi}_i^l$

Loss derivatives
$\frac{\partial h[\theta](x,y)}{\partial w_{ij}^l} = \frac{\partial h^{l[\theta]}(x^l, y)}{\partial x_i^l} \frac{\partial x_i^l}{\partial w_{ij}^l} = \delta_i^l \dot{\varphi}_i^l x_j^{l-1}$,
$\frac{\partial h[\theta](x,y)}{\partial b_i^l} = \frac{\partial h^{l[\theta]}(x^l, y)}{\partial x_i^l} \frac{\partial x_i^l}{\partial b_i^l} = \delta_i^l \dot{\varphi}_i^l$
with $\delta_i^l = \frac{\partial h}{\partial x_i^l} \dot{\varphi}_i^l$

1.3.2 Gradient Descent

Gradient descent update
$\theta_{t+1} = \theta_t - \eta\nabla h(\theta_t)$

Gradient flow ODE
$d\frac{\theta}{\mathrm{d}}t = -\nabla h(\theta)$

L-smoothness
$\|\nabla h(\theta_1) - \nabla h(\theta_2)\| \leq L\|\theta_1 - \theta_2\|$ (forall θ_1, θ_2)
$\lambda_{\max(\nabla^2 h)} \leq L$
$\ell(w) - \ell(w') \leq \nabla\ell(w')^\top (w - w') + \frac{L}{2}\|w - w'\|^2$
$\ell''(x) \leq L$

Polyak-Lojasiewicz condition
$\frac{1}{2}\|\nabla h(\theta)\|^2 \geq \mu(h(\theta) - \min h)$ (forall θ)

Convergence rate
$\eta = \frac{1}{L}$
$\Delta t = \frac{2L}{\varepsilon^2}(h(\theta_0) - \min h)$ for ε-critical point
$\Delta h(\theta_t) - \min h \leq \left(1 - \frac{\mu}{L}\right)^t (h(\theta_0) - \min h)$

1.3.3 Acceleration and Adaptivity

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

Nesterov acceleration
$\tilde{\theta}_{t+1} = \theta_t + \beta(\theta_t - \theta_{t-1})$
$\theta_{t+1} = \tilde{\theta}_{t+1} - \eta\nabla h\left(\tilde{\theta}_{t+1}\right)$
More theoretical grounding than heavy ball

AdaGrad updates
$\theta_{i,t+1} = \theta_{i,t} - \eta_i^t \frac{\partial h}{\partial \theta_i}(\theta_t)$,
$\gamma_i^t = \gamma_i^{t-1} + \left(\frac{\partial h}{\partial \theta_i}(\theta_t)\right)^2$,
$\eta_i^t = \frac{\eta}{\sqrt{\gamma_i^t + \delta}}$

Adam updates
$g_i^t = \beta g_i^{t-1} + (1-\beta)\frac{\partial h}{\partial \theta_i}(\theta_t)$
$\gamma_i^t = \alpha\gamma_i^{t-1} + (1-\alpha)\left(\frac{\partial h}{\partial \theta_i}(\theta_t)\right)^2$
$\theta_{i,t+1} = \theta_{i,t} - \eta_i^t g_i^t, \eta_i^t := \frac{\eta}{\sqrt{\gamma_i^t + \delta}}$

RMSprop
Adam without momentum term

1.3.4 Stochastic Gradient Descent

Stochastic gradient descent update
$\theta_{t+1} = \theta_t - \eta\nabla h(\theta_t)\left(x_{i_t}, y_{i_t}\right)$

SGD variance
$V[\theta](S) = \frac{1}{s}\sum_{i=1}^{s} \|\nabla h[\theta](S) - \nabla h[\theta](x_i, y_i)\|^2$

SGD convergence rate
$E[h(|(\theta)_t)] - \min h \leq O\left(\frac{1}{\sqrt{t}}\right)$ (general)
$E[h(|(\theta)_t)] - \min h \leq O\left(\log\frac{t}{t}\right)$ (strongly convex)
$E[h(|(\theta)_t)] - \min h \leq O\left(\frac{1}{t}\right)$ (additionally smooth)

1.3.5 Function Properties

Convexity
$\ell(\lambda w + (1-\lambda)w') \leq \lambda\ell(w) + (1-\lambda)\ell(w')$
$\ell''(x) \geq 0$ forall x

Convexity and differentiability
$\ell(w) \geq \ell(w') + \nabla\ell(w')^\top (w - w')$
Implies convexity for differentiable functions and vice versa

Strong convexity and differentiability
$\ell(w) \geq \ell(w') + \nabla\ell(w')^\top(w - w') + \frac{\mu}{2}\|w - w'\|^2$
$\ell''(x) \geq \mu$ forall x

1.4 Convolutional Networks

1.4.1 Convolutions

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

Fourier transform convolution property
$F(f * g) = F(f) * F(g)$

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

1.4.2 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
$y[r][s,t] = \sum_u \sum_{\Delta s, \Delta t} w[r,u][\Delta s, \Delta t] * x[u][s + \Delta s, t + \Delta t]$
r: output channel, u: input channel

Number of parameters of a convolutional layer
$D = (|r| * |u|) * (|\Delta s| * |\Delta t|)$
fully connected · window size

1.4.3 Natural Language Processing with ConvNets

Word embedding
$\Omega : w \mapsto x_w \in \mathbb{R}^n$

Conditional log-bilinear model
Prediction of output word μ given word w in neighborhood
$P(\mu \mid w) = \frac{\exp(x_w^\top y_\mu)}{\sum_\mu} \exp(x_w^\top y_\mu)$

$h(\{x_w\}, \{y_\mu\}) = \sum_{(w,\mu)} \ell_{w\mu}$
$\ell_{w,\mu} = -x_w^\top y_\mu + \ln \sum_\mu \exp(x_w^\top y_\mu)$

Negative sampling
$\tilde{\ell}_{w,\mu} = -\ln \sigma(x_w^\top y_\mu) - \beta E_{\mu \sim D} \ln(1 - \sigma(x_w^\top y_\mu))$

1.5 Recurrent Networks

1.5.1 Simple Recurrent Networks

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

Output map
$\hat{y}_t := G[\xi](z_t)$

RNN parameterization
$F[U,V](z,x) := \varphi(Uz + Vx)$
$G[W](z) := \psi(Wz)$, $W \in \mathbb{R}^{q \times m}$

Backpropagation through time
$\frac{\partial h}{\partial z_i^t} = \sum_{s=t}^{T} \delta_k^s \sum_{j=1}^{m} \frac{\partial \hat{y}_k^s}{\partial z_j^s} \frac{\partial z_j^s}{\partial z_i^t}$,
$\frac{\partial \hat{y}_k^s}{\partial z_j^s} = \dot{\psi}_k^s w_{kj}$
$\frac{\partial h}{\partial v_{ij}} = \sum_{t=1}^{T} \frac{\partial h}{\partial z_i^t} \dot{\varphi}_i^t x_j^t$
$\frac{\partial h}{\partial u_{ij}} = \sum_{t=1}^{T} \frac{\partial h}{\partial z_i^t} \dot{\varphi}_i^t z_j^{t-1}$

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

Gradient norms
$\frac{\partial z^T}{\partial z^0} = \dot{\Phi}^T U * \ldots * \dot{\Phi}^1 U$
The norm of gradients either:
1. Vanishes exponentially if $\sigma_1(U) < \frac{1}{|(\alpha)}$: $\left\|\frac{\partial z^t}{\partial z^0}\right\|_2 \leq (|(\alpha)\sigma_1(U))^t \to \infty$
2. Explodes if $\sigma_1(U)$ is too large

Bidirectional RNNs
$\hat{y}_t = \psi\left(Wz_t + \widetilde{W}\tilde{z}_t\right)$

1.5.2 Gated Memory

LSTM
$z_t := \sigma(F\tilde{x}_t) * z_{t-1} + \sigma(G\tilde{x}_t) * \tanh(V\tilde{x}_t)$
$\tilde{x}_t := \text{mat}(x_t; h_t)$, $h_{t+1} = \sigma(H\tilde{x}_t) * \tanh(Uz_t)$

GRU
$z_t = (1 - \sigma) * z_{t-1} + \sigma * \tilde{z}_t$,
$\sigma := \sigma(G[x_t, z_{t-1}])$
$\tilde{z}_t := \tanh(V[r_t * z_{t-1}, x_t])$
$r_t := \sigma(H[z_{t-1}, x_t])$

1.5.3 Linear Recurrent Models

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

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

Stability condition
$\max_j |\lambda_j| \leq 1$

Initialization
$\lambda_i = \exp(-\exp(\kappa_i) + i\varphi_i)$,
$e^{\kappa_i} = -\ln r_i$
$\varphi_i \sim \text{Uni}[0; 2\pi]$, $r_i \sim \text{Uni}[I]$, $I \subset [0;1]$

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

1.6 Attention and Transformers

1.6.1 Attention

Attention mixing
$\xi_s := \sum_t \alpha_{st} W x_t$, $\alpha_{st} \geq 0$, $\sum_t \alpha_{st} = 1$
$A = (a_{st}) \in \mathbb{R}^{T \times T}$, s.t. $\Xi = WXA^\top$

Query-key matching
$Q = U_Q X$, $K = U_K X$
$(U_Q, U_K \in \mathbb{R}^{q \times n})$
$Q^\top K = X^\top U_Q^\top U_K X$ rank $\leq q$
$(Q^\top K \in \mathbb{R}^{T \times T})$

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 $\beta = \frac{1}{\sqrt{q}}$

Feature transformation
$X \mapsto \Xi \mapsto F(\Xi)$,
$F(\theta)(\Xi) = (F(\xi_1), \ldots, F(\xi_T))$

Positional encoding
$p_{tk} = \text{cases}(\sin(t\omega_k), k \text{ even}; \cos(t\omega_k), k \text{ odd})$,
$\omega_k = C^{\frac{k}{K}}$

Transformer architecture
Self-attention: attend to its own values in the past
Cross-attention: E.g. decoder attends to encoder output (query from decoder, key and value from encoder)

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

GELU activation
$\varphi(z) = z\ \text{Prob}(z \leq Z)$, $Z \sim \text{N}(0,1)$

1.7 Geometric Deep Learning

1.7.1 Sets and Points

Function over sets
$\{x_1, \ldots, x_M\} \subset \mathbb{R}$, $f : 2^{\mathbb{R}} \to Y$

Order-invariance property
$f(x_1, \ldots, x_M) = f\left(x_{\pi(1)}, \ldots, x_{\pi(M)}\right)$ forall $\pi \in S_M$

Equivariance property
$f(x_1, \ldots, x_M) = (y_1, \ldots, y_M) \Rightarrow$
$f\left(x_{\pi(1)}, \ldots, x_{\pi(M)}\right) = \left(y_{\pi(1)}, \ldots, y_{\pi(M)}\right)$

Permutation invariant sum
$\sum_{m=1}^{M} x_m = \sum_{m=1}^{M} x_{\pi(m)}$, forall M, forall $\pi \in S_M$

Deep Sets model
$f(x_1, \ldots, x_M) = \rho\left(\sum_{m=1}^{M} \varphi(x_m)\right)$

Max pooling variant
$f(x_1, \ldots, x_M) = \rho\left(\max_{m=1}^{M} \varphi(x_m)\right)$

Equivariant map construction
$\rho : \mathbb{R} \times \mathbb{R}^N \to Y$,
$\left(x_m, \sum_{k=1}^{M} \varphi(x_k)\right) \mapsto y_m$

1.7.2 Graph Convolutional Networks

Feature and adjacency matrices
$X = \text{mat}(x_1^\top; \ldots; x_M^\top)$, $A = (a_{nm})$ with $a_{nm} =$ cases$(1$, if $\{v_n, v_m\} \in E; 0$, otherwise$)$

Permutation matrix constraints
$P \in \{0,1\}^{M \times M}$ s.t.
$\sum_{n=1}^{M} p_{nm} = \sum_{n=1}^{M} p_{mn} = 1$ (forall m)

Graph invariance definition
$f(X, A) \neq f(PX, PAP^\top)$, forall $P \in \Pi_M$

Graph equivariance definition
$f(X, A) \neq Pf(PX, PAP^\top)$, forall $P \in \Pi_M$

Node neighborhood features
$X_m := \{\{x_n : \{v_n, v_m\} \in E\}\}$, $\{\{\text{cdot}\}\} =$ multiset

Message passing scheme
$\varphi(x_m, X_m) = \varphi\left(x_m, m_{X_m} \psi(x)\right)$
m is a permutation-invariant operation

Normalized adjacency matrix
$|(A) = D^{-\frac{1}{2}}(A + I)D^{-\frac{1}{2}}$
$D = \text{diag}(d_1, \ldots, d_M)$, $d_m = 1 + \sum_{n=1}^{M} a_{nm}$

GCN layer
$X^+ = \sigma(|(A)XW)$, $W \in \mathbb{R}^{M \times N}$

Two-layer GCN
$Y = \text{softmax}(|(A)(|(A)XW^0)W^1)$

1.7.3 Spectral Graph Theory

Laplacian operator
$\Delta f := \sum_{n=1}^{N} \frac{\partial^2 f}{\partial x_n^2}$, $f : \mathbb{R}^N \to \mathbb{R}$

Graph Laplacian
$L = D - A$, $(Lx)_n = \sum_{m=1}^{M} a_{nm}(x_n - x_m)$

Normalized Laplacian
$\tilde{L} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} = D^{-\frac{1}{2}}(D - A)D^{-\frac{1}{2}}$

Graph Fourier transform
$L = D - A = U \Lambda U^{\top}$,
$\Lambda := \mathrm{diag}(\lambda_1, ..., \lambda_M), \lambda_i \geq \lambda_{i+1}$

Convolution
$x * y = U((U^{\top} x) \text{ odot } (U^{\top} y))$

Filtering operation
$G_{\theta(L)} x = U G_{\theta(\Lambda)} U^{\top} x$

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

Polynomial kernel network layer
$x_i^{l+1} = \sum_j p_{ij}(L) x_j^l + b_i$,
$p_{ij}(L) = \sum_{k=0}^{K} \alpha_{ijk} L^k$

1.7.4 Attention GNNs

Attention coupling matrix
$Q = (q_{ij})$,
$q_{ij} = \mathrm{softmax}(\rho(u^{\top}(V x_i; V x_j; x_{ij})))$
s.t. $\sum_j A_{ij} q_{ij} = 1$

Attention propagation
$X^{+} = \sigma(QXW)$

Weisfeiler-Lehman test

1.8 Tricks of the Trade

1.8.1 Initialization

Random initialization
$\theta_i^0 \sim \mathrm{N}(0, \sigma_i^2)$, or
$\theta_i^0 \sim \mathrm{Uniform}\left(-\sqrt{3}\sigma_i; \sqrt{3}\sigma_i\right)$

LeCun initialization
$w_{ij} \sim \mathrm{Uniform}[-a; a], a := \frac{1}{\sqrt{n}}, b_i = 0$
Stabilizes variance

Glorot initialization
$w_{ij} \sim \mathrm{Uniform}\left[-\sqrt{3}\gamma; \sqrt{3}\gamma\right]$,
$\gamma := \frac{2}{n+m}$
Stabilizes variance of gradients in backpropagation

He initialization
$w_{ij} \sim \mathrm{N}(0, \gamma)$ or $w_{ij} \sim \mathrm{Uniform}\left[-\sqrt{3}\gamma; \sqrt{3}\gamma\right]$,
$\gamma := \frac{2}{n}$
In ReLU networks typically only $\frac{n}{2}$ units active

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

1.8.2 Weight Decay

L2 regularization
$\Omega_{\mu(\theta)} = \frac{\mu}{2} \|\theta\|^2, \mu \geq 0$

Gradient descent with weight decay
$\Delta\theta = -\eta \nabla E(\theta) - \eta \nabla \Omega_{\mu(\theta)} = -\eta \nabla E(\theta) - \eta \mu \theta$

Weight decay for multiple layers
$\theta = (\mathrm{vec}(W^1), \mathrm{vec}(W^2), ..., \mathrm{vec}(W^L))$,
$\Omega_{\mu(\theta)} = \sum_{l=1}^{L} \mu_l \|W^l\|_F^2$

Local loss landscape
$\theta_\mu^* = (H + \mu I)^{-1} H \theta^*, H = Q^{\top} \Lambda Q$
$(\Lambda + I)^{-1} \Lambda = \mathrm{diag}\left(\frac{\lambda_i}{\lambda_i + \mu}\right)$
The minimum θ^* is shrunk along directions with small eigenvalues

Generalization
$\mu = \frac{\sigma^2}{u^2}$, u: teacher "sign"al
Optimal weight decay inverse proportional to the "sign"al-to-noise ratio

1.8.3 Dropout

Probability φ_i of keeping a unit

Dropout as Ensembling
$p(y \mid x) = \sum_{b \in \{0,1\}^R} p(b) p(y \mid x; b)$
with $p(b) = \mathrm{prod}_{i=1}^{R} \varphi_i^{b_i} (1 - \varphi_i)^{1 - b_i}$

Weight scaling for inference
$\tilde{w}_{ij} \leftarrow \varphi_j w_{ij}$

1.8.4 Normalization

Batch normalization
E and V from minibatches or population statistics
$|(f) = \frac{f - E[f]}{\sqrt{V[f]}}, E[|(f)] = 0, V[|(f)] = 1$
$|(f)[\mu, \gamma] = \mu + \gamma \ |(f)$

Weight normalization
$f(v, \gamma)(x) = \varphi(w^{\top} x), w := \frac{\gamma}{\|v\|_2} v$
Gradient descent with respect to decoupled γ and v:
$\frac{\partial E}{\partial \gamma} = \nabla_w E * \frac{v}{\|v\|_2}$
$\nabla_v E = \frac{\gamma}{\|v\|}\left(I - \frac{w w^{\top}}{\|w\|^2}\right) \nabla_w E$

Layer normalization
$\tilde{f}_i = \frac{f_i - E[f]}{\sqrt{V[f]}}$,
$E[f] = \frac{1}{m} \sum_{i=1}^{m} f_i$
$V[f] = \frac{1}{m} \sum_{i=1}^{m} (f_i - E[f])^2$
Using population averages across units in a layer

1.8.5 Model Distillation

Tempered cross entropy loss for distillation
$\ell(x) = \sum_{y=1}^{K} \frac{\exp\left[\frac{F_{y(x)}}{T}\right]}{\sum_{\mu=1}^{K} \exp\left[\frac{F_{\mu(x)}}{T}\right]} \left[\frac{1}{T} G_{y(x)} - \right.$
$\left.\ln \sum_{\mu=1}^{K} \exp\left[\frac{G_{\mu(x)}}{T}\right]\right]$
$T > 0$, F_y: teacher logits, G_y: student logits

Gradient of distillation loss
$\frac{\partial \ell}{\partial G_y} = \frac{1}{T}\left[\frac{e^{\frac{F_y}{T}}}{\sum_\mu e^{\frac{F_\mu}{T}}} - \frac{e^{\frac{G_y}{T}}}{\sum_\mu e^{\frac{G_\mu}{T}}}\right]$

1.9 Theory

1.9.1 Neural Tangent Kernel

Linearized DNN taylor approximation
$h(\beta)(x) = f(x) + \beta * \nabla f(x)$
with $\beta \approx \theta - \theta_0, f(x) := f(\theta_0)(x)$

Kernel of gradient feature maps
$k(x, \xi) = \nabla f(x) * \nabla f(\xi), \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$

Dual representation
$h(\alpha)(x) = f(x) + \sum_{i=1}^{s} \alpha_i \nabla f(x_i) * \nabla f(x)$

Squared loss
$E(\alpha) = \frac{1}{2s} \sum_{i=1}^{s} \left(\sum_{j=1}^{s} \alpha_j \nabla f(x_j) * \nabla f(x_i) + f(x_i) - y_i\right)^2$

Optimal solution of linearized DNN
$K = \left[k(x_i, x_j)\right]_{i,j=1}^{n} \in \mathbb{R}^{n \times n}$
$\alpha^* = K^{+}(y - f)$,
$h^{*(x)} = k(x) K^{+}(y - f)$

Neural Tangent Kernel NTK
$k(\theta)(x, \xi) := \nabla f(\theta)(x) * \nabla f(\theta)(\xi)$

Quadratic loss
$E(\theta) = \frac{1}{2} \|f(\theta) - y\|^2, y := (y_1, ..., y_s)^{\top}$

Gradient flow ODE
$\dot{\theta} := \mathrm{d}\frac{\theta}{\mathrm{d}} t = \sum_{i=1}^{s} \left(y_i - f_{i(\theta)}\right) \nabla f_{i(\theta)}$

Functional gradient flow
$\dot{f}_j = \nabla f_j * \dot{\theta} = \sum_{i=1}^{s} (y_i - f_i) k(\theta)(x_i, x_j)$
$\dot{f} = K(\theta)(y - f)$

Infinite width limit
$w_{ij}^l = \frac{\sigma_w}{\sqrt{m_l}} \varepsilon_{ij}^l$,
$b_i^l = \frac{\sigma_b}{\sqrt{m_l}} \beta_i^l$,
$\varepsilon_{ij}^l, \beta_i^l \sim \mathrm{N}(0, 1)$
$k(\theta) \to k_\infty$ for $m_l \to \infty$
Initial NTK converges to deterministic limit

NTK constancy
$d\frac{k(\theta(t))}{\mathrm{d}} t = 0$
$f_{\infty(x)} = k(x) K^{+}(y - f), k = k_\infty$
NTK remains constant when training in infinite width limit

Vanishing curvature
$\frac{\|\nabla^2 f(\theta_0)\|_2}{\|\nabla f(\theta_0)\|_2^2} \ll 1$

Near-constancy
$\|k(\theta_0) - k(\theta_t)\|_F \in O\left(\frac{1}{m}\right), m = m_1 = ... = m_L$

1.9.2 Bayesian DNNs

Bayesian predictive distribution
$f(x) = \int f(\theta)(x) p(\theta \mid S) d\theta$

Bayes rule
$p(\theta \mid S) = \frac{p(\theta) p(S \mid \theta)}{p(S)}$,
$p(S) = \int p(\theta) p(S \mid \theta) d\theta$

Parameter priors (Gaussian)
$p(\theta) = \mathrm{prod}_{i=1}^{d} p(\theta_i), \theta_i \sim \mathrm{N}(0, \sigma_i^2)$
$-\log p(\theta) = \frac{1}{2\sigma^2} \|\theta\|^2 + \mathrm{const}$
Essentially a weight decay term

Likelihood (Gaussian noise)
$-\log p(S \mid \theta) = \frac{1}{2\gamma^2} \|y - f(\theta)\|^2 + \mathrm{const.}$
with $y_i = f^{*(x_i)} + \nu_i, \nu_i \sim \mathrm{N}(0, \gamma^2)$

Posterior
$-\log p(\theta \mid S) = E(\theta) + \mathrm{const}$,
$E(\theta) = \frac{1}{2\gamma^2} \|y - f\|^2 + \frac{1}{2\sigma^2} \|\theta\|^2$

Bayesian ensembling (post hoc)
$f(\Theta)(x) = \sum_{i=1}^{n} \frac{\exp[-E(\theta_i)]}{\sum_{j=1}^{n} \exp[-E(\theta_j)]} f(\theta_i)(x)$
Relative posterior weighting

Markov chain monte carlo (MCMC)
$\theta_0, \theta_1, \theta_2, ...$,
$\theta_{t+1} \mid \theta_t \sim \Pi$
$p(\theta_1 \mid S) \Pi(\theta_2 \mid \theta_1) = p(\theta_2 \mid S) \Pi(\theta_1 \mid \theta_2)$

Metropolis-Hastings
$\Pi(\theta_1 \mid \theta_2) = \tilde{\Pi}(\theta_1 \mid \theta_2) A(\theta_1 \mid \theta_2)$
$A(\theta_1 \mid \theta_2) = \min\left\{1, \frac{p(\theta_1 \mid S) \tilde{\Pi}(\theta_2 \mid \theta_1)}{p(\theta_2 \mid S) \tilde{\Pi}(\theta_1 \mid \theta_2)}\right\}$
Modified transition probability with acceptance step A

Hamiltonian monte carlo
$E(\theta) = -\sum_{x,y} \log p(y \mid x; \theta) - \log p(\theta)$
$H(\theta, v) = E(\theta) + \frac{1}{2} v^{\top} M^{-1} v$
with $p(\theta, v)$ propto $\exp[-H(\theta, v)]$
$\dot{v} = -E(\theta), \dot{\theta} = v$
$\theta_{t+1} = \theta_t + \eta v_t$
$v_{t+1} = v_t - \eta \nabla E(\theta_t)$

Langevin dynamics
$\dot{\theta} = v$
$dv = -\nabla E(\theta)\, \mathrm{d}t - Bv\, \mathrm{d}t + N(0, 2B\, \mathrm{d}t)$
$\theta_{t+1} = \theta_t + \eta v_t$
$v_{t+1} = (1 - \eta\gamma) v_t - \eta \int \nabla \tilde{E}(\theta) + \sqrt{2\gamma\eta} N(0, I)$

1.9.3 Gaussian Processes

Gaussian process
$(f(x_1), ..., f(x_s)) \sim N$
$\sum_{i=1}^{s} \alpha_i f(x_i) \sim N$, forall $\alpha \in \mathbb{R}^s$

Mean and covariance functions
GPs are completely defined by first and second order statistics
$\mu(x) := E_{x[f(x)]}$
$k(x, \xi) := E_{x,\xi}[f(x)f(\xi)] - \mu(x)\mu(\xi)$
$K_{\mu\nu} = k(x_\mu, x_\nu), K \in \mathbb{R}^{s\times s}$

Example kernels
$k(x, \xi) = x^\top \xi, k(x, \xi) = e^{-\gamma\|x-\xi\|^2}$

GPs in DNN
Treating parameters as random variables. Each unit in a DNN becomes a random function.

Linear Layer
$w \sim \mathrm{N}\left(0, \frac{\sigma^2}{n} I_{n\times n}\right)$
$E[y_i y_j] = \frac{\sigma^2}{n} x_i^\top x_j$

Deep layers
$W^{l+1} X^l, l \geq 1$
No longer normal as products break normality, but near-normal for high dimensional inputs.

Non-linear activations
$\mu(x^{l+1}) = E[\varphi(W^l x^l)]$
Kernel recursion
$K_{\mu\nu}^l = E\left[\varphi\left(x_{i\mu}^{l-1}\right)\varphi\left(x_{i\nu}^{l-1}\right)\right]$
$= \sigma^2 E[\varphi(f_\mu)\varphi(f_\nu)]$
$f \sim \mathrm{GP}(0, K^{l-1})$

Kernel regression
Mean of bayesian predictive distribution
$f^{*(x)} = k(x)^\top K^+ y$
$E\left[\left(f(x) - f^{*(x)}\right)^2\right] = K(x, x) - k(x)^\top K^+ k(x)$

1.9.4 Statistical Learning Theory

VC learning theory
$L_t = -\frac{\left\|m(x_t, x_0, t) - m_{\theta(x_t, t)}\right\|^2}{2\sigma_t^2} + \text{const.}$
$\text{VC-dim}(F) := \max_s \sup_{|S|=s} 1[\![F(S)| = 2^s]\!]$

VC inequality
$P\left(\sup_F |\hat{E}(f) - E(f)| > \varepsilon\right) \leq 8\ |F(s)|\ e^{-s\frac{\varepsilon^2}{32}}$

Double descent
Beyond the interpolation point, models start to learn and eventually may level out at a lower generalization error.

Generalization gap
$\Delta := \max\left(0, E - \hat{E}\right)$
E: expected population error, $\hat{E}$: empirical error

KL divergence
$D_{\mathrm{KL}}(p \parallel q) =$
$\int p(x) \log\left(\frac{p(x)}{q(x)}\right) \mathrm{d}x = E_{x\sim p}\left[\ln\left(\frac{p(x)}{q(x)}\right)\right]$

PAC-Bayesian theorem
For fixed E and any Q over s samples:
$E_{Q[E(f)]} -$
$E_{Q[\hat{E}(f)]} \leq \sqrt{\frac{2}{s}\left[\mathrm{KL}(Q \parallel P) + \ln\left(\frac{1}{2\sqrt{s}\varepsilon}\right)\right]}$
Ensures general rate $\tilde{O}\left(\frac{1}{\sqrt{s}}\right)$

PAC-Bayesian bound
$Q := N(\theta, \mathrm{diag}(\sigma_i^2))$
$\mathrm{KL}(Q \parallel P) = \sum_i \log\left(\frac{\lambda}{\sigma_i} + \frac{\sigma_i^2 + \theta_i^2}{2\lambda^2} - \frac{1}{2}\right)$
$E_{\mathrm{PAC}}(Q) :=$
$E_{Q[\hat{E}]} + \sqrt{\frac{2}{s}\left[\mathrm{KL}(Q \parallel P) + \ln\left(\frac{1}{2\sqrt{s}\varepsilon}\right)\right]}$
Favours minima robust to parameter perturbations

PAC-bayesian learning implementation
$\theta_{t+1} = \theta_t - \eta\nabla E_{Q[\hat{E}]} = \theta_t - \eta\nabla\hat{E}\left(\tilde{\theta}\right)$,
with $\tilde{\theta} \sim Q(\theta, \sigma)$
Gradient loss on perturbed parameters

Reparameterization trick
$\tilde{\theta} = \theta + \mathrm{diag}(\sigma_i)\varepsilon, \varepsilon \sim \mathrm{N}(0, I)$
Backpropagation to θ and σ_i

1.10 Generative Models

1.10.1 Variational Autoencoders

Linear autoencoder
$x \mapsto z = Cx, C \in \mathbb{R}^{m\times n}$
$z \mapsto \hat{x} = Dz, D \in \mathbb{R}^{n\times m}$
$E(C, D)(x) = \frac{1}{2}\|x - \hat{x}\|^2 = \frac{1}{2}\|x - DCx\|^2$
$DCX = \hat{X} = U\Sigma_m V^\top$
$\Sigma_m = \mathrm{diag}(\sigma_1, ..., \sigma_m, 0, ..., 0)$
For centered data equivalent to PCA, but generally has non-global minima

Linear factor analysis
Probability Model
$p_{X(x)} = \int p_{Z(z)} p_{X|Z}(x \mid z)\,\mathrm{d}z$
Z: latent variables, X: observed variables

Linear observation model
$x = \mu + Wz + \nu$ with $\nu \sim \mathrm{N}(0, \Sigma)$
$x \sim \mathrm{N}(\mu, WW^\top + \Sigma)$ for $z \sim \mathrm{N}(0, I)$

Posterior mean and covariance
$\mu_{z|x} = W^\top(WW^\top + \Sigma)^{-1}(x - \mu)$
$\Sigma_{z|x} = I - W^\top(WW^\top + \Sigma)^{-1} W$

Pseudoinverse limit
$W^\top(WW^\top + \sigma^2 I)^{-1} \to W^+ \in \mathbb{R}^{m\times n}$
$\mu_{z|x} \to W^+(x - \mu), \Sigma_{z|x} \to 0$

Maximum likelihood estimation
$\mu, W \max \to \log p_{\mu,W}(S)$

Optimality condition for W
$w_i = \rho_i u_i, \rho_i = \max\left\{0, \sqrt{\lambda_i - \sigma^2}\right\}$
With (λ_i, u_i) eigenvalues and eigenvectors of covariance matrix.
For $\sigma = 0$ equivalent to PCA.

Variational autoencoder (VAE)
$z \sim \mathrm{N}(0, I)$
$x = F(\theta)(z) = (F^L @ ... @ F^1)(z)$

Evidence lower bound (ELBO)
$\log p_\theta(x) = \log \int p_\theta(x \mid z)p(z)\,\mathrm{d}z$
$= \log \int q(z)\left[\frac{p_\theta(x \mid z)p(z)}{q(z)}\right]\mathrm{d}z$
$\geq \int q(z) \log p_\theta(x \mid z)\,\mathrm{d}z - \int q(z) \log\left(\frac{q(z)}{p(z)}\right)\mathrm{d}z$
$=: L(\theta, q)(x)$
$\theta \max \to L(\theta, q)(S) = \sum_{i=1}^{s} L(\theta, q)(x_i)$

Inference network
$z \sim \mathrm{N}(\mu(x), \Sigma(x))$
$z = \mu + \Sigma^{\frac{1}{2}}\varepsilon, \varepsilon \sim \mathrm{N}(0, I)$
$\nabla_\mu E[f(z)] = E[\nabla_z f(z)]$
$\nabla_\Sigma E[f(z)] = \frac{1}{2} E[\nabla_z^2 f(z)]$
Integration by parts derivation

1.10.2 Generative Adversarial Networks

GAN objective
$V(G, D) = E_{x_r \sim p_{\text{data}}} D(x_r) + E_{z\sim p_z}(1 - D(G(z)))$

Discriminator Mixture Model
$\tilde{p}_{\theta(x,y)} = \frac{1}{2}\left(yp(x) + (1 - y)p_{\theta(x)}\right)$,
$y \in \{0, 1\}$,
p: true probability, p_θ: model probability

Bayes-optimal classifier
$q_{\theta(x)} := P\{y = 1 \mid x\} = \frac{p(x)}{p(x) + p_{\theta(x)}}$
To detect fake samples, $y = 1$ for real samples, $y = 0$ for fake samples

Logistic likelihood
$\theta \min \to \ell^{*(\theta)} := E_{\tilde{p}_\theta}\left[y \ln q_{\theta(x)} + (1 - y) \ln\left(1 - q_{\theta(x)}\right)\right]$

Jensen-Shannon as effective objective
$\ell^* = E_{\tilde{p}_\theta}\left[y \ln q_{\theta(x)} + (1 - y) \ln\left(1 - q_{\theta(x)}\right)\right]$
$= -\frac{1}{2}H(p) - \frac{1}{2}H(p_\theta) + H\left(\frac{1}{2}(p + p_\theta)\right) - \ln 2$
$= \mathrm{JS}(p, p_\theta) - \ln 2.$

Discriminator model
$q_\varphi : x \mapsto [0; 1], \varphi \in \Phi$

Objective bounds
$\ell^{*(\theta)} \geq \sup_{\varphi\in\Phi} \ell(\theta, \varphi)$
$\ell(\theta, \varphi) := E_{\tilde{p}_\theta}\left[y \ln q_{\varphi(x)} + (1 - y) \ln\left(1 - q_{\varphi(x)}\right)\right]$

Saddle point optimization
$\theta^* := \mathrm{argmin}_{\theta\in\Theta}\left(\sup_{\varphi\in\Phi} \ell(\theta, \varphi)\right)$
φ: Generator, θ: Discriminator

Alternating gradient descent/ascent
$\theta_{t+1} = \theta_t - \eta\nabla_\theta \ell(\theta_t, \varphi_t)$
$\varphi_{t+1} = \varphi_t + \eta\nabla_\varphi \ell(\theta_{t+1}, \varphi_t)$

Extra-gradient steps
$\theta_{t+1} = \theta_t - \eta\nabla_\theta \ell(\theta_{t+0.5}, \varphi_t)$
with $\theta_{t+0.5} := \theta_t - \eta\nabla_\theta \ell(\theta_t, \varphi_t)$
$\varphi_{t+1} = \varphi_t + \eta\nabla_\varphi \ell(\theta_t, \varphi_{t+0.5})$
with $\varphi_{t+0.5} := \varphi_t + \eta\nabla_\varphi \ell(\theta_t, \varphi_t)$

Deconvolutional DNN
Upside-down ConvNet for image generation

1.10.3 Denoising Diffusion

Markov chains
$x_{0:t-1} \perp x_{t+1:\infty} \mid x_t$ (forall t)
$p(x_t \mid x_{t-1}) = p(x_1 \mid x_0)$ (forall t),
$p(x_{s:t}) = p(x_t)\mathrm{prod}_{\tau=s+1}^{t} p(x_{\tau-1} \mid x_\tau)$
$p(x_{s:t}) = p(x_s)\mathrm{prod}_{\tau=s+1}^{t} p(x_\tau \mid x_{\tau-1})$,
$\pi(x_{t+1}) = \int \pi(x_t)p(x_{t+1} \mid x_t)\,\mathrm{d}x_t$

Denoising diffusion
Forward (noise generation)
$\pi^* = \nu_0 \mapsto \nu_1 \mapsto ... \mapsto \nu_{T-1} \mapsto \nu_T = \pi$
Backward (denoising)
$\pi = \mu_T^\theta \mapsto \mu_{T-1}^\theta \mapsto ... \mapsto \mu_1^\theta \mapsto \mu_0^\theta \approx \pi^*$

Gaussian example
$\pi \approx N(0, I)$,
$x_t \mid x_{t-1} \sim \mathrm{N}\left(\sqrt{1 - \beta_t}x_{t-1}, \beta_t I\right)$
Forward SDE
$dx_t = -\frac{1}{2}\beta_t x_t\,\mathrm{d}t + \sqrt{\beta_t}\,\mathrm{d}\omega_t$
Backward SDE
$dx_t = \left[-\frac{1}{2}\beta_t x_t - \beta_t \nabla_{x_t} \log q_{t(x_t)}\right]\mathrm{d}t +$
$\sqrt{\beta_t}\,\mathrm{d}|(\omega)_t$
score · wiener process

ELBO bound
$x_t = \sqrt{1 - \beta_t}x_{t-1} + \sqrt{\beta_t}\varepsilon_t, \varepsilon_t \sim \mathrm{N}(0, I)$
$\ln p_{\theta(x_0)} = \ln \int q(x_{1:T} \mid x_0)\left(\frac{p_{\theta(x_{0:T})}}{q(x_{1:T} \mid x_0)}\right)\mathrm{d}x_{1:T}$
$\geq E\left[\ln\left(\frac{p_{\theta(x_{0:T})}}{q(x_{1:T} \mid x_0)}\right) \mid x_0\right]$
$= \sum_{t=0}^{T} L_t$
$L_t := \mathrm{cases}\Big(E\left[\ln p_{\theta(x_0 \mid x_1)}\right], t =$
$0; -D(q(x_T \mid x_0) \parallel \pi), t =$
$T; -D\left(q(x_{t-1} \mid x_t, x_0) \parallel p_{\theta(x_{t-1} \mid x_t)}\right), \mathrm{else}\Big)$

Backward model assumption
$x_{t-1} \mid x_t \sim \mathrm{N}(m(x_t, t), \Sigma(x_t, t))$

Entropy bounds
$H(x_t) \geq H(x_{t-1}) \Rightarrow$
$H(x_t \mid x_{t-1}) \geq H(x_{t-1} \mid x_t)$

Noise schedules
$|(\alpha)_t = \text{prod}_{\tau=1}^{t}(1-\beta_\tau)$, $|(\beta)_t = 1 - |(\alpha)_t$
$x_t \approx N\left(\sqrt{|(\alpha)_t} x_0, |(\beta)_t I\right)$ $t \to \infty \to N(0, I)$

Forward trajectory target
$x_{t-1} \mid x_t, x_0 = N\left(m(x_t, x_0, t), \tilde{\beta}_t I\right)$
$m(x_t, x_0, t) =$
$\left(\frac{\sqrt{|(\alpha)_{t-1}}\beta_t}{1-|(\alpha)_t}\right)x_0 + \frac{(1-|(\alpha)_{t-1})\sqrt{1-\beta_t}}{1-|(\alpha)_t}x_t$
with $\tilde{\beta}_t = \frac{1-|(\alpha)_{t-1}}{1-|(\alpha)_t}\beta_t$

Fixed isotropic covariance
$\Sigma(x_t, t) = \sigma_t^2 I$, where $\sigma_t^2 \in \left\{\beta_t, \tilde{\beta}_t\right\}$

Simplified ELBO
$L_t = -\frac{\left\|m(x_t, x_0, t) - m_{\theta(x_t, t)}\right\|^2}{2\sigma_t^2} + \text{const.}$

Reparameterization
$x_t = \sqrt{|(\alpha)_t} x_0 + \sqrt{1-|(\alpha)_t}\varepsilon \Rightarrow x_0 =$
$\frac{1}{\sqrt{|(\alpha)_t}}x_t - \frac{\sqrt{1-|(\alpha)_t}}{\sqrt{|(\alpha)_t}}\varepsilon$
$m(x_t, x_0, t) = \frac{1}{\sqrt{\alpha_t}}\left[x_{t(x_0,\varepsilon)} - \frac{\beta_t}{\sqrt{1-|(\alpha)_t}}\varepsilon\right]$
with $\varepsilon \sim \mathrm{N}(0, I)$

Expected squared error
$E_{q[L_t \mid x_0]} = E_{\varepsilon}\left[\rho_t\left\|\varepsilon - \varepsilon_{\theta\left(\sqrt{|(\alpha)_t}x_0 + \sqrt{1-|(\alpha)_t}\varepsilon, t\right)}\right\|^2 \mid x_0\right]$
with $\rho_t = \frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-|(\alpha)_t)}$

Final simplified criterion
$h(\theta)(x) =$
$\frac{1}{T}\sum_{t=1}^{T} E\left[\left\|\varepsilon - \varepsilon_{\theta\left(\sqrt{|(\alpha)_t}x + \sqrt{1-|(\alpha)_t}\varepsilon, t\right)}\right\|^2\right]$

1.11 Ethics

1.11.1 Adversarial Examples

Adversarial perturbation
$f(x+\nu) \neq f(x)$ s.t. $\|\nu\|_p \leq \varepsilon$

p-norm definitions
$\|x\|_p = \left(\sum_i |x_i|^p\right)^{\frac{1}{p}}$
$\|x\|_\infty = \max_i |x_i|$, $\|x\|_0 = |\{i : x_i \neq 0\}|$

Optimal perturbation (linear binary classification)
$\nu \propto \text{sign}(f_1(x) - f_2(x))(w_2 - w_1)$
for $f_i = w_i^\top x + b_i$

Optimal perturbation (multiclass)
$\nu = \text{argmin}_{i>1}\frac{f_1(x) - f_{i(x)}}{\|w_1 - w_i\|_2^2}(w_i - w_1)$

DeepFool iterative optimization
Iterate: $\text{argmin}_{\Delta\nu}\|\Delta\nu\|_2$ s.t.
$(\nabla f_1(x) - \nabla f_2(x))^\top \Delta\nu < f_1(x) - f_2(x)$

Robust training
$\ell(f(x), y) \to \max_{\nu:\|\nu\|_p \leq \varepsilon} \ell(f(x+\nu), y)$

Projected gradient ascent (p = 2)
$\nu_{t+1} = \varepsilon\Pi[\nu_t + \alpha\nabla_x \ell(f(x+\nu_t), y)]$
$\Pi[z] \coloneqq \frac{z}{\|z\|_2}$

Projected gradient ascent (p = oo)
$\nu_{t+1} = \varepsilon\Pi[\nu_t + \alpha\ \text{sign}(\nabla_x \ell(f(x+\nu_t), y))]$
$\Pi[z] \coloneqq \frac{z}{\|z\|_\infty}$

Fast Gradient Sign Method (FGSM)
$\nu = \varepsilon\ \text{sign}(\nabla_x \ell(f(x), y))$

2 Computer Vision

2.1 The Digital Image & Sensors

Charge Coupled Device (CCD)

Photons

- **Blooming**: Oversaturated photosites cause vertical channels to “flood” (bright vertical line)

Image Noise

Additive Gaussian noise:

Color camera concepts:

1. Prism (split light, 3 sensors, needs good alignment, good color separation)

2.2 Image Segmentation

Pixel-wise classification problem, to group pixels in an image that share common properties.
Segmentation of I: Find $R_1, \dots, R_n$ such that $I = \bigcup_{i=1}^{N} R_i$ with $R_i \cap R_j = \emptyset \quad \forall i \neq j$.

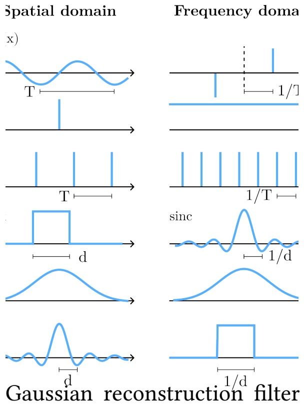
Thresholding Segment image into 2 classes. $B(x, y) = 1$ if $I(x, y) \geq T$ else 0, finding T with trial and error, compare results with ground truth.

Important Kernels

Laplacian	Prewitt_x	Low-pass/ Mean / Box	High-pass
Gaussian	Sobel_x	Diff_x	Diff_y
$\frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$	$\mathrm{mat}(-1, 0, 1; -2, 0, 2; -1, 0, 1)$	$\mathrm{mat}([-1], 1)$	$\mathrm{mat}([-1], 1)^\top$

Dirac delta: $\delta(x) = \mathrm{cases}(0 \text{ if } x \neq 0, \text{undefined else})$ with $\int_{-\infty}^{\infty} \delta(x) \, \mathrm{d}x = 1$. $\mathcal{F}[\delta(x - x_0)](u) = e^{-i2\pi u x_0}$. $\delta(u) = \int_{\mathbb{R}} e^{-i2\pi x u} \, \mathrm{d}x$.
Sampling f at points x_n: $f_s(x) = \sum_n f(x_n)\delta(x - x_n)$.

Property	$\boldsymbol{f(x)}$	$\boldsymbol{F(u)}$
Linearity	$\alpha f_1(x) + \beta f_2(x)$	$\alpha F_1(u) + \beta F_2(u)$
Duality	$F(x)$	$f(-u)$



Simple procedure of sampling and reconstructing a 2D signal: Sample Signal, FT, Cut out Magnitude Spectrum by multiplication with box filter, FT^{-1}. **Some reconstruction filters**: Nearest neighbor, Bilinear , Gaussian reconstruction filter (equiv. to convolving sampled signal w/ Gaussian kernel. $\mathrm{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$

Image restoration

Image degradation is applying kernel h to some image I. The inverse $\tilde{h}$ should compensate: $I \xrightarrow{h(x)} J \xrightarrow{\tilde{h}(x)} I$.
Determine with $\mathcal{F}[\tilde{h}](u, v) \cdot \mathcal{F}[h](u, v) = 1$. Or $\tilde{h} = \mathcal{F}^{-1}\left[\frac{1}{\mathcal{F}[h]}\right]$
Cancellation of frequencies & noise amplification $\rightarrow$ Regularize using $\tilde{\mathcal{F}}[\tilde{h}](u, v) = \mathcal{F}[h] / (|\mathcal{F}[h]|^2 + \varepsilon)$.

Motion blur: $h(x, y) = \frac{1}{2l}[\theta(x + l) - \theta(x - l)]\delta(y)$