ML & Mathematical Foundations Reference Sheet

Fictional Academy of Technology — Department of Computer Science & AI

Topics: Linear Alg | Calc/Prob | ML Foundations | Deep Learning

Vector Spaces & Norms

Vector Norms (p-norms):

$\|x\|_p = \left(\sum_{i=1}^d |x_i|^p\right)^{1/p} \quad (p \geq 1)$

$\|x\|_1 = \sum_{i=1}^d |x_i|$ (sparsity-inducing)

$\|x\|_2 = \sqrt{x^T x}$ (Euclidean, rotation invariant)

$\|x\|_\infty = \max_i |x_i|$ (Chebyshev norm)

Matrix Norms:

$\|A\|_F = \sqrt{\sum_{i,j} A_{ij}^2} = \sqrt{\text{tr}(A^T A)}$ (Frobenius)

$\|A\|_2 = \sigma_{\max}(A)$ (Spectral / Operator norm)

Angles & Projections:

$\cos\theta = \frac{x^T y}{\|x\|_2 \|y\|_2}$ (Cosine similarity)

$\text{proj}_y(x) = \frac{x^T y}{\|y\|_2^2} y$ (Orthogonal projection)

Fundamental Inequalities:

- Cauchy-Schwarz: $|x^T y| \leq \|x\|_2 \|y\|_2$
- Triangle Inequality: $\|x + y\| \leq \|x\| + \|y\|$
- Hölder's: $\|x^T y\| \leq \|x\|_p \|y\|_q$ where $\frac{1}{p} + \frac{1}{q} = 1$

Linear Systems & Projections

System of Equations: $Ax = b$

- Consistent iff $\text{rank}(A) = \text{rank}([A \mid b])$.
- If $A \in \mathbb{R}^{m \times n}$ has full rank:

1. $m > n$ (overdetermined): No exact solution. Least Squares: $x^* = (A^T A)^{-1} A^T b$
2. $m < n$ (underdetermined): Infinite solutions. Minimum Norm: $x^* = A^T (A A^T)^{-1} b$

Projections onto Subspaces:

- Projection matrix onto range(A): $P = A(A^T A)^{-1} A^T$
- Properties: $P^2 = P$ (idempotent), $P^T = P$ (symmetric), $\text{rank}(P) = \text{tr}(P)$.
- Orthogonal complement: $P_\perp = I - P$.

Matrix Decompositions

Eigenvalues & Eigenvectors: $Av = \lambda v,\ v \neq 0$.

- Characteristic Equation: $\det(A - \lambda I) = 0$.
- Trace & Determinant: $\text{tr}(A) = \sum_{i=1}^d \lambda_i$, $\det(A) = \prod_{i=1}^d \lambda_i$.
- Spectral Theorem: Symmetric matrix $A = A^T$ can be diagonalized by orthogonal matrix Q: $A = Q \Lambda Q^T$.

Singular Value Decomposition (SVD):

For any $A \in \mathbb{R}^{m \times n}$: $A = U \Sigma V^T$.

- $U \in \mathbb{R}^{m \times m}$: Left singular vectors (eigenvectors of AA^T).
- $V \in \mathbb{R}^{n \times n}$: Right singular vectors (eigenvectors of $A^T A$).
- $\Sigma \in \mathbb{R}^{m \times n}$: Diagonal singular values $\sigma_i = \sqrt{\lambda_i(A^T A)}$ $(\sigma_1 \geq \sigma_2 \geq \cdots \geq 0)$.
- Low-Rank Approximation (Eckart-Young): $A_k = \sum_{i=1}^k \sigma_i u_i v_i^T = \arg\min_{\text{rank}(B) \leq k} \|A - B\|_F^2$

Matrix Calculus Identities

For vector $x \in \mathbb{R}^d$, matrices A, B, and vectors a, b:

$\nabla_x (a^T x) = a$

$\nabla_x (x^T A y) = Ay$

$\nabla_x (x^T A x) = (A + A^T)x \quad (= 2Ax \text{ if } A = A^T)$

$\nabla_x \|Ax - b\|_2^2 = 2A^T (Ax - b)$

Traces:

$\nabla_A \text{tr}(AB) = B^T$

$\nabla_A \text{tr}(A^T B) = B$

$\nabla_A \text{tr}(ABA^T) = A(B + B^T)$

$\nabla_A \log\det(A) = A^{-T}$ (for $A \succ 0$)

$\nabla_A \det(A) = \det(A) A^{-T}$

Multivariate Chain Rule:

If $z = g(y)$ and $y = f(x)$, then:

$\nabla_x z = \left(\frac{\partial y}{\partial x}\right)^T \nabla_y z$

Taylor Series & Critical Points

Hessian Matrix: $H(x) \in \mathbb{R}^{d \times d}$ where $H_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}$

- Symmetric if f is continuously twice differentiable (C^2).

Multivariate Taylor Expansion:

$f(x) \approx f(x_0) + \nabla f(x_0)^T (x - x_0) + \frac{1}{2}(x - x_0)^T H(x_0)(x - x_0)$

Second-Order Optimality Conditions:

At critical point x^* where $\nabla f(x^*) = 0$:

- Local Min: $H(x^*) \succ 0$ (Positive Definite).
- Local Max: $H(x^*) \prec 0$ (Negative Definite).
- Saddle Point: $H(x^*)$ has mixed-sign eigenvalues.

Optimization Algorithms

Convexity:

- f is convex if $f(\lambda x + (1 - \lambda) y) \leq \lambda f(x) + (1 - \lambda) y$ for all $\lambda \in [0, 1]$.
- First-order: $f(y) \geq f(x) + \nabla f(x)^T (y - x)$.
- Second-order: $H(x) \succeq 0$ (PSD) $\forall x$.
- Jensen's Inequality: $f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)]$.

Gradient Descent (GD):

$x^{(t+1)} = x^{(t)} - \eta \nabla f(x^{(t)})$

- Linear convergence rate for strongly convex functions.

Newton-Raphson Method:

$x^{(t+1)} = x^{(t)} - [H(x^{(t)})]^{-1} \nabla f(x^{(t)})$

- Quadratic convergence near x^*. Expensive Hessian inverse ($O(d^3)$).

Constrained Optimization

Primal Problem: $\min_x f(x)$ s.t. $g_i(x) \leq 0\ (\forall i)$, $h_j(x) = 0\ (\forall j)$.

Lagrangian Function:

$\mathcal{L}(x, \lambda, \nu) = f(x) + \sum_{i=1}^m \lambda_i g_i(x) + \sum_{j=1}^p \nu_j h_j(x)$

Karush-Kuhn-Tucker (KKT) Conditions:

If x^* is a local optimum and strong duality holds, then $\exists \lambda^*, \nu^*$:

1. **Stationarity:** $\nabla_x f(x^*) + \sum_i \lambda_i^* \nabla g_i(x^*) + \sum_j \nu_j^* \nabla h_j(x^*) = 0$
2. **Primal Feasibility:** $g_i(x^*) \leq 0\ \forall i,\ h_j(x^*) = 0\ \forall j$
3. **Dual Feasibility:** $\lambda_i^* \geq 0\ \forall i$
4. **Complementary Slackness:** $\lambda_i^* g_i(x^*) = 0\ \forall i$

Duality: Dual $d^* = \max_{\lambda \geq 0, \nu} \min_x \mathcal{L}(x, \lambda, \nu)$.

- Weak Duality: $d^* \leq p^*$ (always holds).
- Strong Duality: $d^* = p^*$ (holds under Slater's condition).

Probability & Bayes' Theorem

Axioms: $0 \leq P(A) \leq 1$, $P(\Omega) = 1$, $P(\cup_i A_i) = \sum_i P(A_i)$ (if disjoint).

Bayes' Rule:

$P(A \mid B) = \frac{P(B|A)P(A)}{P(B)} = \frac{P(B|A)P(A)}{\sum_k P(B|A_k)P(A_k)}$

- $P(A)$: Prior; $P(B \mid A)$: Likelihood; $P(A \mid B)$: Posterior.

Independence: $P(A \cap B) = P(A)P(B) \iff P(A \mid B) = P(A)$.

Conditional Independence: $P(A, B \mid C) = P(A \mid C) P(B \mid C)$.

Chain Rule: $P(X_1, \ldots, X_d) = \prod_{i=1}^d P(X_i \mid X_1, \ldots, X_{i-1})$.

Expectation, Variance & Distributions

Expectation: $\mathbb{E}[X] = \int x p(x) dx$. Linearity: $\mathbb{E}[aX + b] = a\mathbb{E}[X] + b$.

Variance: $\text{Var}(X) = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$.

- Scaling: $\text{Var}(aX + b) = a^2 \text{Var}(X)$.
- Sum: $\text{Var}(X + Y) = \text{Var}(X) + \text{Var}(Y) + 2\text{Cov}(X, Y)$.

Covariance: $\text{Cov}(X, Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])] = \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y]$.

- Covariance Matrix: $\Sigma \in \mathbb{R}^{d \times d}$, $\Sigma_{ij} = \text{Cov}(X_i, X_j)$.

Gaussian (Normal) Distribution:

- Univariate: $p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$.
- Multivariate: $p(x) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right)$.

Bernoulli: $P(X = x) = p^x (1 - p)^{1-x}$, $x \in \{0, 1\}$. Mean p, Var $p(1 - p)$.

Multinomial: $P(X_1 = x_1, \ldots, X_k = x_k) = \frac{n!}{\prod x_i!} \prod p_i^{x_i}$ where $\sum x_i = n$.

Information Theory Metrics

Entropy (Information): $H(X) = -\sum_x P(x) \log_2 P(x) = -\mathbb{E}[\log_2 P(X)]$.

Joint & Conditional Entropy:

$H(X, Y) = -\sum_{x,y} P(x, y) \log_2 P(x, y)$

$H(Y \mid X) = -\sum_{x,y} P(x, y) \log_2 P(y \mid x) = H(X, Y) - H(X)$.

Kullback-Leibler (KL) Divergence:

$D_{\text{KL}}(P \,\|\, Q) = \sum_x P(x) \log_2 \frac{P(x)}{Q(x)} = \mathbb{E}_{X \sim P}\left[\log_2 \frac{P(X)}{Q(X)}\right]$

- $D_{\text{KL}}(P \,\|\, Q) \geq 0$; $D_{\text{KL}}(P \,\|\, Q) = 0 \iff P = Q$. Asymmetric.

Cross-Entropy:

$H(P, Q) = -\mathbb{E}_{X \sim P}[\log_2 Q(X)] = H(P) + D_{\text{KL}}(P \,\|\, Q)$.

Mutual Information:

$I(X; Y) = H(X) - H(X \mid Y) = D_{\text{KL}}(P(x, y) \,\|\, P(x)P(y))$.

Linear Regression & Regularization

Let $X \in \mathbb{R}^{N \times d}$ design matrix, $y \in \mathbb{R}^N$ target vector.

Ordinary Least Squares (OLS):

$\min_\theta J(\theta) = \frac{1}{2N} \|X\theta - y\|_2^2$

- Normal Equations: $\nabla_\theta J(\theta) = \frac{1}{N} X^T (X\theta - y) = 0$.
- Closed-form solution: $\theta^*_{\text{OLS}} = (X^T X)^{-1} X^T y$.

Ridge Regression (L_2 Regularization):

$\min_\theta J(\theta) = \frac{1}{2}\|X\theta - y\|_2^2 + \frac{\lambda}{2}\|\theta\|_2^2$

- Closed-form solution: $\theta^*_{\text{Ridge}} = (X^T X + \lambda I)^{-1} X^T y$.

Lasso Regression (L_1 Regularization):

$\min_\theta J(\theta) = \frac{1}{2N}\|X\theta - y\|_2^2 + \lambda\|\theta\|_1$

- No closed-form solution. Promotes sparse weights (feature selection).

Logistic Regression

Sigmoid Function: $\sigma(z) = \frac{1}{1 + e^{-z}} \in (0, 1)$.

- Derivative: $\frac{d\sigma(z)}{dz} = \sigma(z)(1 - \sigma(z))$.

Model Formulation: $P(y = 1 \mid x; \theta) = \sigma(\theta^T x) = h_\theta(x)$.

Binary Cross-Entropy Loss:

$J(\theta) = -\frac{1}{N} \sum_{i=1}^N \left[y^{(i)} \log(h_\theta(x^{(i)})) + (1 - y^{(i)}) \log(1 - h$

Gradient Descent Update:

$\nabla_\theta J(\theta) = \frac{1}{N} X^T (h_\theta(X) - y)$

$\theta \leftarrow \theta - \eta \nabla_\theta J(\theta)$

- Multi-class: Softmax function $\text{Softmax}(z)_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$.

Support Vector Machines (SVM)

Goal: Maximize margin $\frac{2}{\|w\|_2}$ subject to correct classification.

Primal Soft-Margin Formulation:

$\min_{w,b,\xi} \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^N \xi_i$ s.t. $y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i,\ \xi_i \geq 0$

- ξ_i: Slack variables; C: Penalty hyperparameter.

Dual Formulation (QP Problem):

$\max_\alpha \sum_{i=1}^N \alpha_i - \frac{1}{2}\sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y^{(i)} y^{(j)} \langle x^{(i)}, x^{(j)} \rangle$

s.t. $0 \leq \alpha_i \leq C\ \forall i$, and $\sum_{i=1}^N \alpha_i y^{(i)} = 0$.

- Weights: $w = \sum_{i=1}^N \alpha_i y^{(i)} x^{(i)}$. Support Vectors have $\alpha_i > 0$.

Kernel Trick:

Replace inner product $\langle x^{(i)}, x^{(j)} \rangle$ with kernel $K(x^{(i)}, x^{(j)}) = \langle \phi(x^{(i)}), \phi(x^{(j)}) \rangle$.

- Linear: $K(x, z) = x^T z$
- Polynomial: $K(x, z) = (x^T z + c)^d$
- Gaussian RBF: $K(x, z) = \exp(-\gamma \|x - z\|_2^2), \gamma > 0$.

Bias-Variance & Evaluation Metrics

Bias-Variance Decomposition (MSE):

If $y = f(x) + \epsilon$ where $\mathbb{E}[\epsilon] = 0$, $\text{Var}(\epsilon) = \sigma^2$:

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

- $\text{Bias}(\hat{f}(x)) = \mathbb{E}[\hat{f}(x)] - f(x)$
- $\text{Var}(\hat{f}(x)) = \mathbb{E}[(\hat{f}(x) - \mathbb{E}[\hat{f}(x)])^2]$
- σ^2: Irreducible error.

Classification Metrics:

- Precision = $\frac{\text{TP}}{\text{TP+FP}}$ (Positive Predictive Value)
- Recall (Sensitivity) = $\frac{\text{TP}}{\text{TP+FN}}$ (True Positive Rate)
- F_1-Score = $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2\text{TP}}{2\text{TP+FP+FN}}$
- False Positive Rate (FPR) = $\frac{\text{FP}}{\text{FP+TN}}$.

Principal Component Analysis (PCA)

Goal: Project data $X \in \mathbb{R}^{N \times d}$ onto a k-dimensional subspace ($k < d$) preserving maximum variance.

1. Center the data: $x_i \leftarrow x_i - \mu$ where $\mu = \frac{1}{N}\sum_i x_i$.
2. Compute Covariance Matrix: $\Sigma = \frac{1}{N} X^T X \in \mathbb{R}^{d \times d}$.
3. Solve Eigenvalue Problem: $\Sigma w_j = \lambda_j w_j$.

- Eigenvector w_j corresponding to top eigenvalues represent directions of maximum variance.
- Total variance fraction explained by k components: $\sum_{j=1}^k \lambda_j / \sum_{j=1}^d \lambda_j$.

4. Project data: $Z = XW_k \in \mathbb{R}^{N \times k}$ where $W_k = [w_1, \ldots, w_k]$.

Decision Trees & Ensemble Methods

Node Impurity Measures (split selection):

- Entropy: $H(D) = -\sum_{c=1}^C p_c \log_2 p_c$.
- Gini Impurity: $G(D) = 1 - \sum_{c=1}^C p_c^2$.

Information Gain (IG):

$$IG(D, A) = \text{Impurity}(D) - \sum_{v \in \text{Values}(A)} \frac{|D_v|}{|D|} \text{Impurity}(D_v).$$

Ensemble Techniques:

- **Bagging:** Train models independently on bootstrap samples. Reduces variance (e.g., Random Forest: also selects random feature subsets at splits).
- **Boosting:** Train models sequentially. Reduces bias.

AdaBoost: Update weights: $w_i \leftarrow w_i \exp(-\alpha_t y_i h_t(x_i))$.

Classifier weight: $\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$.

Gradient Boosting: Fits weak learner $h_t(x)$ to negative gradient of loss $L(y, f(x))$:

$$r_{it} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x)=f_{t-1}(x)}$$

Clustering & Expectation-Maximization

K-Means Clustering:

Objective: Minimize inertia

$J = \sum_{i=1}^N \sum_{k=1}^K z_{ik} \|x_i - \mu_k\|_2^2$ s.t. $z_{ik} \in \{0, 1\}, \sum_k z_{ik} = 1$.

1. E-Step: Assign $z_{ik} = 1$ if $k = \arg\min_j \|x_i - \mu_j\|_2^2$, else 0.
2. M-Step: Update centroids $\mu_k = \frac{\sum_i z_{ik} x_i}{\sum_i z_{ik}}$.

Gaussian Mixture Models (GMM):

$p(x) = \sum_{k=1}^K \pi_k \mathcal{N}(x \mid \mu_k, \Sigma_k)$ where $\sum_k \pi_k = 1, \pi_k \geq 0$.

Solve using Expectation-Maximization (EM):

- **E-Step (Evaluate Responsibilities):**

$$\gamma_{ik} = \frac{\pi_k \mathcal{N}(x_i \mid \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}$$

- **M-Step (Maximize Likelihood):**

Define $N_k = \sum_{i=1}^N \gamma_{ik}$.

$\mu_k^{\text{new}} = \frac{1}{N_k}\sum_{i=1}^N \gamma_{ik} x_i \qquad \pi_k^{\text{new}} = \frac{N_k}{N}$

$\Sigma_k^{\text{new}} = \frac{1}{N_k}\sum_{i=1}^N \gamma_{ik}(x_i - \mu_k^{\text{new}})(x_i - \mu_k^{\text{new}})^T$.

Neural Network Layers & Activations

Layer Formulation:

$z^{(l)} = W^{(l)} a^{(l-1)} + b^{(l)} \quad a^{(l)} = g\left(z^{(l)}\right)$

- $W^{(l)} \in \mathbb{R}^{n_l \times n_{l-1}}$: weight matrix; $b^{(l)} \in \mathbb{R}^{n_l}$: bias vector.

Activation Functions $g(z)$:

- **Sigmoid:** $\sigma(z) = \frac{1}{1+e^{-z}}$. $\sigma'(z) = \sigma(z)(1 - \sigma(z))$.
- **Tanh:** $\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$. $\tanh'(z) = 1 - \tanh^2(z)$.
- **ReLU:** $g(z) = \max(0, z)$. $g'(z) = \mathbf{1}(z > 0)$.
- **Leaky ReLU:** $g(z) = \max(\alpha z, z)$, $\alpha \approx 0.01$.
- **GELU (Gaussian Error Linear Unit):**

$$g(z) = z\Phi(z) \approx 0.5z\left(1 + \tanh\left(\sqrt{2/\pi}(z + 0.044715z^3)\right)\right).$$

Backpropagation & Computational Graphs

Let L be the scalar loss. We compute gradients using the Chain Rule.

Forward Pass:

$z^{(l)} = W^{(l)} a^{(l-1)} + b^{(l)} \quad a^{(l)} = g\left(z^{(l)}\right)$

Backward Pass (Error Signal $\delta^{(l)} = \nabla_{z^{(l)}} L$):

- For output layer L: $\delta^{(L)} = \nabla_{a^{(L)}} L \odot g'\left(z^{(L)}\right)$.
- For hidden layers $l < L$:

$\delta^{(l)} = \left((W^{(l+1)})^T \delta^{(l+1)}\right) \odot g'\left(z^{(l)}\right)$

Parameter Gradients:

$\frac{\partial L}{\partial W^{(l)}} = \delta^{(l)}\left(a^{(l-1)}\right)^T \qquad \frac{\partial L}{\partial b^{(l)}} = \delta^{(l)}$

Optimization Algorithms in Deep Learning

Let $g_t = \nabla_\theta \mathcal{L}_t(\theta)$ be the mini-batch gradient at step t.

Stochastic Gradient Descent (SGD): $\theta_{t+1} = \theta_t - \eta g_t$.

SGD with Momentum:

$v_t = \beta v_{t-1} + \eta g_t \qquad \theta_{t+1} = \theta_t - v_t$.

RMSprop (Adaptive Learning Rate):

$v_t = \beta v_{t-1} + (1 - \beta) g_t^2 \qquad \theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v_t} + \epsilon} g_t$.

Adam (Adaptive Moment Estimation):

$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$ (1st moment)

$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$ (2nd moment)

$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \qquad \hat{v}_t = \frac{v_t}{1 - \beta_2^t}$ (bias correction)

$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t$

- Defaults: $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$.

Normalization & Regularization

Batch Normalization (BN):

Normalizes activations across the mini-batch dimension B.

- Mean: $\mu_\mathcal{B} = \frac{1}{m}\sum_{i=1}^m x_i$ Variance: $\sigma_\mathcal{B}^2 = \frac{1}{m}\sum_{i=1}^m (x_i - \mu_\mathcal{B})^2$.
- Normalize: $\hat{x}_i = \frac{x_i - \mu_\mathcal{B}}{\sqrt{\sigma_\mathcal{B}^2 + \epsilon}}$ Scale & Shift: $y_i = \gamma \hat{x}_i + \beta$.
- During testing: use running averages of mean and variance.

Layer Normalization (LN):

Normalizes activations across the feature dimension H for each sample.

$\mu_L = \frac{1}{H}\sum_{j=1}^H x_j \qquad \sigma_L^2 = \frac{1}{H}\sum_{j=1}^H (x_j - \mu_L)^2$

- Crucial for Transformers and sequence models.

Dropout:

- Training: zero out activations with probability p.
- Inverted Dropout: scale active neurons by $\frac{1}{1-p}$ during training to keep identity map at test time: $\hat{h} = \frac{h \odot m}{1-p}$ where $m_j \sim \text{Bernoulli}(1 - p)$.

Convolutional Neural Networks (CNNs)

Spatial Dimensions of Output:

Given input size W, kernel size K, padding P, stride S:

$$W_{\text{out}} = \left\lfloor \frac{W - K + 2P}{S} \right\rfloor + 1$$

Receptive Field (RF) Calculation:

- Output size step (jump): $J_l = J_{l-1} \times S_l$, with $J_0 = 1$.
- Receptive field: $RF_l = RF_{l-1} + (K_l - 1) \times J_{l-1}$, with $RF_0 = 1$.

Parameter Counting:

For Conv layer with C_{in} input channels, C_{out} output channels, and kernel size $K_h \times K_w$:

Params = $C_{\text{out}} \times (C_{\text{in}} \times K_h \times K_w + 1)$ (if bias is true).

Transformers & Attention Mechanisms

Scaled Dot-Product Attention:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V$$

- $Q \in \mathbb{R}^{T \times d_k}$ (queries), $K \in \mathbb{R}^{T' \times d_k}$ (keys), $V \in \mathbb{R}^{T' \times d_v}$ (values).
- Scaling factor $\frac{1}{\sqrt{d_k}}$ prevents softmax saturation.

Multi-Head Attention (MHA):

$\text{MHA}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h) W^O$

where $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$.

- Projections: $W_i^Q, W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$, $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$.

Transformer Block Architecture:

$x^{(1)} = \text{LayerNorm}(x + \text{MultiHeadAttention}(x, x, x))$

$x^{(2)} = \text{LayerNorm}(x^{(1)} + \text{FeedForward}(x^{(1)}))$

where $\text{FeedForward}(u) = \text{GELU}(uW_1 + b_1)W_2 + b_2$.

Activation Quick Reference

Name	Formula $g(z)$	Derivative $g'(z)$	Ran
Sigmoid	$\frac{1}{1+e^{-z}}$	$g(z)(1-g(z))$	$(0,$
Tanh	$\frac{e^z - e^{-z}}{e^z + e^{-z}}$	$1 - g(z)^2$	$(-1,$
ReLU	$\max(0, z)$	$\mathbf{1}(z > 0)$	$[0, \infty$
L-ReLU	$\max(\alpha z, z)$	$\mathbf{1}(z > 0) + \alpha\mathbf{1}(z \leq 0)$	$(-\infty,$