# Training Theory: Stability, Calibration, Compression, and Distillation

Mathematics of Deep Learning — Chapter 14

Yin Yang

Foundation Books Project

# Roadmap

Training Loss vs Deployment Behavior

Stability as Sensitivity

Sharpness, Flatness, Local Robustness

Regularization Revisited

Calibration and Confidence

Compression as Approximation

Knowledge Distillation

Fine-Tuning, Forgetting, Drift

Training Diagnostics Decision Guide

Summary

# Training Loss vs Deployment Behavior

# After the Run Finished

Earlier chapters built losses, optimizers, and architectures. This chapter asks what happens *after* training.

- Running example: 3-class classifier with logits $g_\theta(x) \in \mathbb{R}^3$ and $p_\theta(y \mid x) = \mathrm{softmax}(g_\theta(x))$.
- Sample value: logits $(2, 1, 0) \Rightarrow$ probabilities $(0.665, 0.245, 0.090)$.
- Four classes of check: *generalization*, *stability*, *calibration*, *compression/distillation*.

A trained model still has to prove it behaves well outside the training loop.

# Four Risks, Four Roles

**Training, validation, test empirical risks**

$$\widehat{R}_\bullet(\theta) = \frac{1}{|S_\bullet|} \sum_{(x,y)\in S_\bullet} \ell(f_\theta(x), y), \quad \bullet \in \{\text{train, val, test}\}.$$

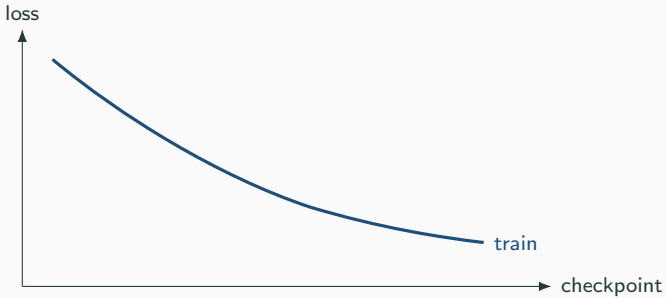
**Deployment risk**

$$R_{\text{deploy}}(\theta) = \mathbb{E}_{(X,Y)\sim \mathcal{D}_{\text{deploy}}}[\ell(f_\theta(X), Y)].$$

Validation generalization gap: $G_{\text{val}}(\theta) = \widehat{R}_{\text{val}}(\theta) - \widehat{R}_{\text{train}}(\theta)$.

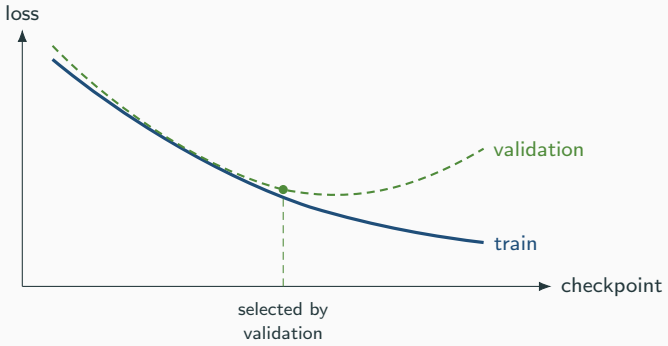
Validation is one held-out distribution; deployment may be another.

# Loss Curves: Train, Val, Deploy




**(1)** Training loss can keep decreasing...

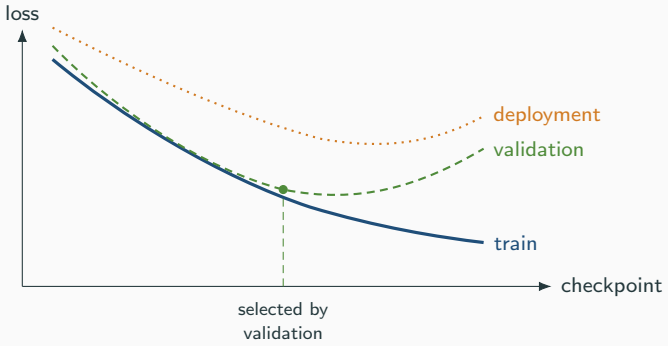
# Loss Curves: Train, Val, Deploy




**(1)** Training loss can keep decreasing...

**(2)** ...while validation has already stopped improving.

# Loss Curves: Train, Val, Deploy




**(1)** Training loss can keep decreasing. . .

**(2)** . . . while validation has already stopped improving.

**(3)** Deployment curve sits elsewhere: different distribution.

Train, val, deploy are different mathematical roles.

# Validation Concentration (Hoeffding)

Fix $\theta$ *before* validation, draw $n$ iid val examples with loss in $[0, B]$. Then

$$\mathbb{P}\left(\left|\widehat{R}_{\mathrm{val}}(\theta) - R_{\mathrm{val}}(\theta)\right| \geq \epsilon\right) \leq 2\exp\left(-\tfrac{2n\epsilon^2}{B^2}\right).$$

Equivalently, with probability $\geq 1 - \delta$,

$$\left|\widehat{R}_{\mathrm{val}}(\theta) - R_{\mathrm{val}}(\theta)\right| \leq B\sqrt{\tfrac{\log(2/\delta)}{2n}}.$$

- Requires $\theta$ fixed *before* inspecting val data.
- Repeated model selection on the same val set weakens the bound.
- Concentration around $R_{\mathrm{val}}$ does not control $R_{\mathrm{deploy}}$ if $\mathcal{D}_{\mathrm{deploy}} \neq \mathcal{D}_{\mathrm{val}}$.

The bound estimates one distribution, not every future use.

# Stability as Sensitivity

# Learning Algorithm $A$ Maps Data to Parameters

A learning algorithm $A$ takes a training set $S$ and returns fitted parameters $\theta_S = A(S)$.

Leave-one-out loss sensitivity at validation point $(x, y)$:

$$\Delta_i(x, y) = \left| \ell(f_{A(S)}(x), y) - \ell(f_{A(S^{\setminus i})}(x), y) \right|.$$

Example. Prediction $(0.66, 0.25, 0.09)$ becomes $(0.61, 0.28, 0.11)$ after removing one training example. Max change: 0.05.

- Compares two fitted models on the *same* val point.
- Asks how much removing one example moves the predictor.

Stability asks how much the fitted predictor changes after a small training change.

# Uniform Stability Bounds the Generalization Gap

**$\beta$-uniform stability**
For bounded loss, every $S$, removed index $i$, and checked $(x, y)$:

$$\left|\ell(f_{A(S)}(x), y) - \ell(f_{A(S^{\setminus i})}(x), y)\right| \leq \beta.$$

**Expected gap bound**

$$\left|\mathbb{E}_S[R(A(S)) - \widehat{R}_S(A(S))]\right| \leq \beta.$$

Stability levers (reduce $\beta$ in practice): smaller learning rate, early stopping, larger val sets, weight decay, less aggressive fine-tuning.

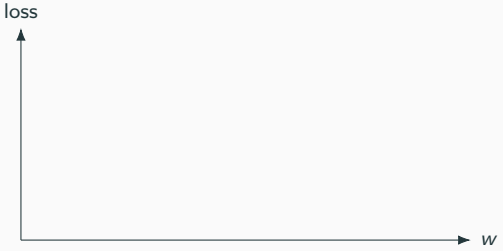
Many practical training choices are stability choices in disguise.

# Sharpness, Flatness, Local Robustness

# Sharpness Proxy and Two Minima

Local perturbation set: $B_\rho(\theta) = \{\theta + \delta : \|\delta\|_2 \le \rho\}$.

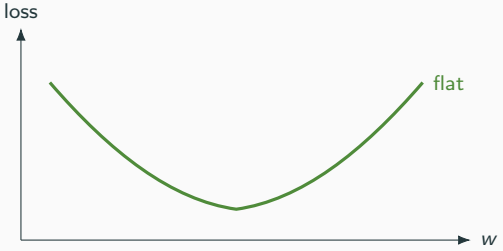
$$\text{sharp}_\rho(\theta) = \max_{\|\delta\|_2 \le \rho} \big(L(\theta + \delta) - L(\theta)\big).$$



# Sharpness Proxy and Two Minima

Local perturbation set: $B_\rho(\theta) = \{\theta + \delta : \|\delta\|_2 \le \rho\}$.

$$\text{sharp}_\rho(\theta) = \max_{\|\delta\|_2 \le \rho} \big(L(\theta + \delta) - L(\theta)\big).$$

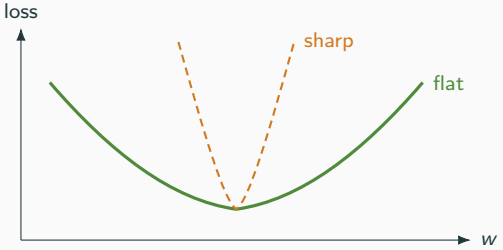



Flat $(w-1)^2$: $\delta = 0.1$ raises loss by 0.01.

# Sharpness Proxy and Two Minima

Local perturbation set: $B_\rho(\theta) = \{\theta + \delta : \|\delta\|_2 \le \rho\}$.

$$\text{sharp}_\rho(\theta) = \max_{\|\delta\|_2 \le \rho} \big(L(\theta + \delta) - L(\theta)\big).$$

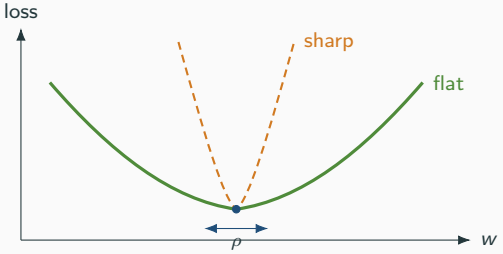



Flat $(w-1)^2$: $\delta = 0.1$ raises loss by 0.01.
Sharp $20(w-1)^2$: same $\delta$ raises loss by 0.20.

# Sharpness Proxy and Two Minima

Local perturbation set: $B_\rho(\theta) = \{\theta + \delta : \|\delta\|_2 \le \rho\}$.

$$\text{sharp}_\rho(\theta) = \max_{\|\delta\|_2 \le \rho} \big(L(\theta + \delta) - L(\theta)\big).$$




Flat $(w-1)^2$: $\delta = 0.1$ raises loss by 0.01.
Sharp $20(w-1)^2$: same $\delta$ raises loss by 0.20.

Same minimum value, different local sensitivity.

# Curvature, Hessian, and Caveats

Quadratic model near a stationary point:

$$L(\theta + \delta) \approx L(\theta) + \nabla L(\theta)^\top \delta + \tfrac{1}{2}\delta^\top H(\theta)\delta.$$

For 1D $L(w) = L(w_0) + a(w - w_0)^2$, $\max_{|\delta| \le \rho} \Delta L = a\rho^2$. Doubling $\rho$ (e.g. quantization) multiplies the worst-case rise by 4.

- Sharpness depends on parameterization and scale.
- Compare proxies only under same architecture, scale, and eval.
- Useful for explaining quantization / fine-tuning sensitivity.

Sharpness is a local diagnostic, not a complete generalization theory.

# Regularization Revisited

# Regularizers: Name What They Change

$$J(\theta) = \widehat{R}_{\text{train}}(\theta) + \lambda\,\Omega(\theta).$$

| Method | Object changed | Purpose |
|---|---|---|
| Weight decay | objective ($\lambda\|\theta\|_2^2$) | discourage large weights |
| Dropout | training-time computation | reduce co-adaptation |
| Label smoothing | target distribution | avoid one-hot certainty |
| Mixup | examples and labels | smooth behavior between examples |
| Early stopping | chosen checkpoint | stop before val degrades |

Label-smoothed target for $K = 3$, $\epsilon = 0.15$: $(1, 0, 0) \mapsto (0.90, 0.05, 0.05)$.

A regularizer should be named by what it changes.

# Implicit Bias, PAC-Bayes, NTK

**Implicit max-margin (informal)**
For separable logistic regression, GD drives loss toward zero and $w_t/\|w_t\|_2 \to w_{\mathrm{SVM}}/\|w_{\mathrm{SVM}}\|_2$.

**PAC-Bayes (informal)**
With probability $\geq 1-\delta$, posterior $Q$ over predictors:

$$\mathrm{kl}(\widehat{L}_S(Q) \,\|\, L(Q)) \leq \frac{D_{\mathrm{KL}}(Q\|P) + \log((n+1)/\delta)}{n}.$$

Benign overfitting: zero training error can still predict well in some regimes. NTK: a fixed-kernel reference for wide-network dynamics.

The optimizer chooses among many zero-error solutions; data geometry decides if interpolation is benign.

# Calibration and Confidence

# Confidence vs Correctness

**Top-class confidence**
$\widehat{y}(x) = \arg\max_k p_\theta(y = k \mid x)$, $C(x) = \max_k p_\theta(y = k \mid x)$.

Calibration (population): $\mathbb{P}(Z = 1 \mid C = c) = c$.

Among predictions with confidence $c$, fraction correct should be $c$.

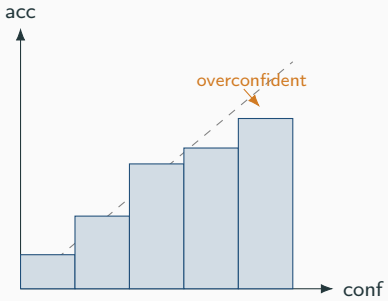
- Accuracy: did we pick the right class?
- Calibration: does $C(x)$ have the right frequency meaning?
- A model can be accurate and overconfident.

Calibration is about probability *meaning*, not just class accuracy.

# Expected Calibration Error and Reliability Diagram

Bin predictions $B_1, \ldots, B_M$ over $N$ examples:

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{N} \left| \mathrm{acc}(B_m) - \mathrm{conf}(B_m) \right|.$$




Caveats: ECE depends on bins, binning rule, and $N$. Empty bins are noisy.

Report ECE *with* its binning rule and sample size.

# Temperature Scaling Preserves Ranking

Post-training temperature $\tau > 0$:

$$\widehat{\tau} = \arg\min_{\tau>0} \; -\frac{1}{n}\sum_{i=1}^{n} \log \frac{\exp(z_{i,y_i}/\tau)}{\sum_k \exp(z_{i,k}/\tau)}.$$

Logits $(2, 1, 0)$:

| $\tau$ | probabilities | reading |
|---|---|---|
| 0.5 | $(0.867, 0.117, 0.016)$ | sharper |
| 1.0 | $(0.665, 0.245, 0.090)$ | original |
| 2.0 | $(0.506, 0.307, 0.186)$ | softer |

Order is preserved $\Rightarrow$ top-1 accuracy unchanged. Temperature scaling cannot repair misranked classes.

Overconfident on val $\Rightarrow$ raise $\tau$; underconfident $\Rightarrow$ lower $\tau$.

# Compression as Approximation

# Compression: Approximation with a Resource Budget

Replace $f_\theta$ by cheaper $\widetilde{f}$. Prediction-space error:

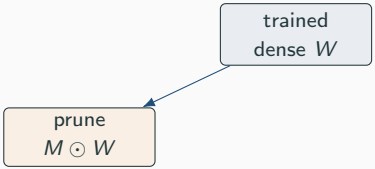
$$E_{\text{comp}} = \frac{1}{|S_{\text{val}}|} \sum_{x \in S_{\text{val}}} \|f_\theta(x) - \widetilde{f}(x)\|_1.$$

trained dense $W$

# Compression: Approximation with a Resource Budget

Replace $f_\theta$ by cheaper $\widetilde{f}$. Prediction-space error:

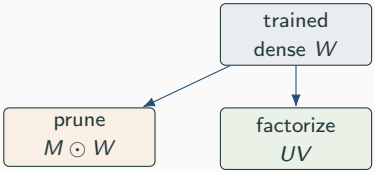
$$E_{\text{comp}} = \frac{1}{|S_{\text{val}}|} \sum_{x \in S_{\text{val}}} \|f_\theta(x) - \widetilde{f}(x)\|_1.$$



# Compression: Approximation with a Resource Budget

Replace $f_\theta$ by cheaper $\widetilde{f}$. Prediction-space error:

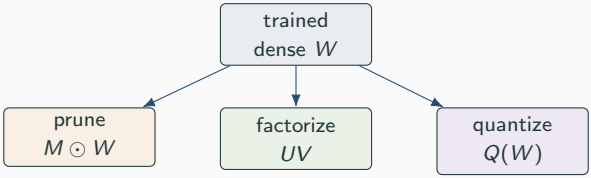
$$E_{\text{comp}} = \frac{1}{|S_{\text{val}}|} \sum_{x \in S_{\text{val}}} \|f_\theta(x) - \widetilde{f}(x)\|_1.$$



# Compression: Approximation with a Resource Budget

Replace $f_\theta$ by cheaper $\widetilde{f}$. Prediction-space error:

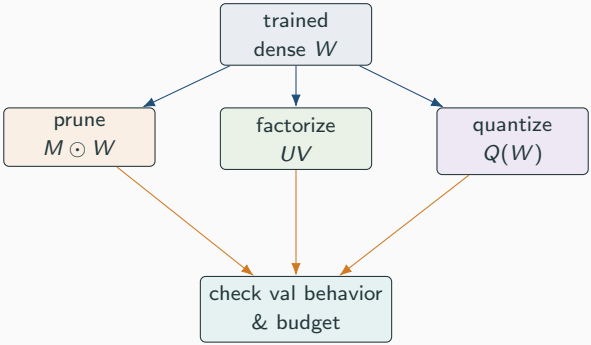
$$E_{\text{comp}} = \frac{1}{|S_{\text{val}}|} \sum_{x \in S_{\text{val}}} \|f_\theta(x) - \widetilde{f}(x)\|_1.$$



# Compression: Approximation with a Resource Budget

Replace $f_\theta$ by cheaper $\widetilde{f}$. Prediction-space error:

$$E_{\text{comp}} = \frac{1}{|S_{\text{val}}|} \sum_{x \in S_{\text{val}}} \|f_\theta(x) - \widetilde{f}(x)\|_1.$$




Compression error is rarely uniform across slices.

# Eckart–Young, Low-Rank, Quantization

**Eckart–Young–Mirsky**
For $W = U\Sigma V^\top$, the rank-$r$ truncated SVD $W_r$ minimizes $\|W - A\|_F$ (and spectral norm) over $\mathrm{rank}(A) \le r$.

Dense $m \times n$ uses $mn$ weights; rank-$r$ uses $r(m+n)$. Saves weights iff $r < mn/(m+n)$.

Example: $m = 1000$, $n = 500$, $r = 50$: $500\,000 \to 75\,000$ weights (15%).

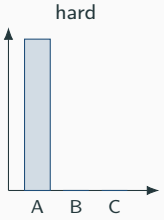
**Uniform scalar quantization**
$Q(w) = s \cdot \mathrm{round}(w/s)$, step $s > 0$.

Best-approximation for one matrix is not best-approximation for one task.

# Knowledge Distillation

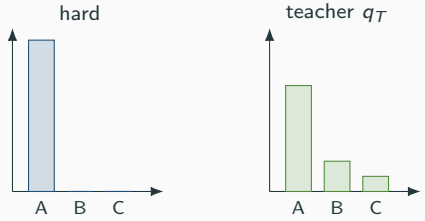
# Soft Targets Carry More Than the Winner

Hard label $e_A = (1, 0, 0)$ vs teacher $q_T = (0.70, 0.20, 0.10)$.



# Soft Targets Carry More Than the Winner

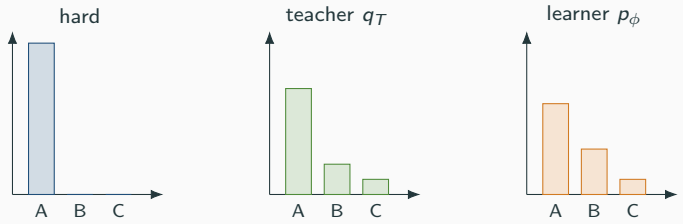
Hard label $e_A = (1, 0, 0)$ vs teacher $q_T = (0.70, 0.20, 0.10)$.




**(2)** Teacher: $B$ more plausible than $C$ — extra signal.

# Soft Targets Carry More Than the Winner

Hard label $e_A = (1, 0, 0)$ vs teacher $q_T = (0.70, 0.20, 0.10)$.




**(2)** Teacher: $B$ more plausible than $C$ — extra signal.

**(3)** Learner trained to match the teacher's full distribution.

A soft teacher target says more than the winning class.

# Distillation KL and Combined Objective

$$L_{\text{distill}}(\phi; x) = D_{\text{KL}}(q_T(\cdot \mid x) \,\|\, p_\phi(\cdot \mid x)) = \sum_k q_T(k \mid x) \log \frac{q_T(k \mid x)}{p_\phi(k \mid x)}.$$

Gibbs: $D_{\text{KL}} \geq 0$ with equality iff $p_\phi = q_T$.

**Combined objective**

$$L_{\text{learn}} = (1 - \alpha)\, \text{CE}(e_y, p_\phi) + \alpha D_{\text{KL}}(q_T \,\|\, p_\phi).$$

Learner $p = (0.60, 0.30, 0.10)$, hard label $A$: $-\log(0.60) \approx 0.511$. Teacher loss also checks $B$ and $C$. Temperature on the teacher exposes non-winner similarities.

Tune $\alpha$: how much teacher behavior vs. how much hard labels.

# Fine-Tuning, Forgetting, Drift

# Parameter Drift and Retained-Slice Regression

**Parameter drift**
$D_{\text{param}} = \|\theta_1 - \theta_0\|_2$.

**Retained-slice regression**

$$\Delta_{\text{retain}} = \widehat{R}_{S_{\text{retain}}}(\theta_1) - \widehat{R}_{S_{\text{retain}}}(\theta_0).$$

| Slice | before | after | reading |
|---|---|---|---|
| Target task accuracy | 71% | 84% | improved |
| Retained source acc. | 88% | 79% | forgetting |
| Calibration ECE | 0.04 | 0.11 | worse confidence |

Improving a target task can damage retained behavior and calibration.

# Anchor Regularizer: Penalize Drift

$$J(\theta) = \widehat{R}_{\text{target}}(\theta) + \lambda \, \|\theta - \theta_0\|_2^2, \qquad \nabla\Omega = 2\lambda(\theta - \theta_0).$$

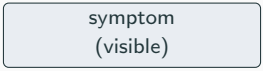
- Weight decay pulls $\theta$ toward 0.
- Anchor penalty pulls $\theta$ toward the *pretrained* $\theta_0$.
- Related to continual-learning methods that reduce catastrophic forgetting.

Fine-tuning reports should record: starting checkpoint, trainable layers, data size, seeds, validation slices, retained-behavior checks.

Name the retained slice and metric *before* declaring fine-tuning helped.

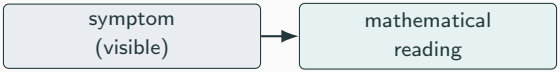
# Training Diagnostics Decision Guide

# Symptom → Measurement, Step by Step



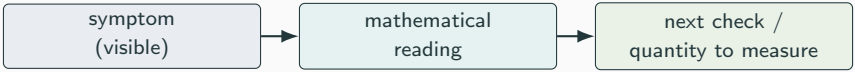

**(1)** Start from the visible symptom.

# Symptom → Measurement, Step by Step



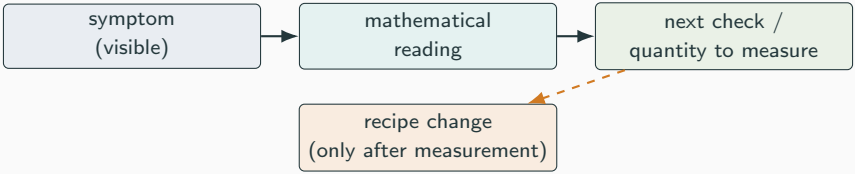

**(1)** Start from the visible symptom.
**(2)** Translate to a mathematical quantity.

# Symptom → Measurement, Step by Step




**(1)** Start from the visible symptom.
**(2)** Translate to a mathematical quantity.
**(3)** Pick the measurement that decides between causes.

# Symptom → Measurement, Step by Step




**(1)** Start from the visible symptom.
**(2)** Translate to a mathematical quantity.
**(3)** Pick the measurement that decides between causes.
**(4)** Only then change the training recipe.

Name the measured quantity before changing the recipe.

# The Diagnostic Table

| Symptom | Math reading | Next check |
|---|---|---|
| Train falls, val rises | positive gen. gap | splits, capacity, reg., early stop |
| High acc, high ECE | over/underconfident | reliability table, temperature |
| Seeds vary widely | training is unstable | report spread; reduce LR |
| Avg acc kept, rare class drops | uneven approx. error | per-slice metrics |
| Top-1 matches, conf. poor | probabilities not transferred | tune $\alpha$, calibrate |
| Target up, retained down | drift / forgetting | retained slice, anchor |

A model card should record: split, checkpoint, train+val loss, slice metrics, calibration summary, model size, compression / distillation objective.

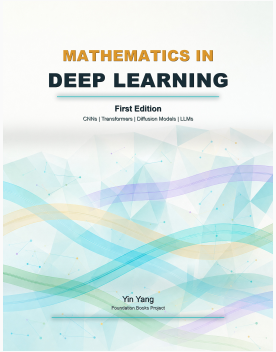
The measurement identifies the problem being solved.

# Summary

# Chapter in One Slide

- **Deployment $\neq$ validation:** four risks, four roles; Hoeffding only handles fixed $\theta$ on one distribution.
- **Stability:** how much the fitted predictor moves when data or randomness moves; bounds the expected gen. gap.
- **Sharpness:** local sensitivity to parameter perturbations; depends on scale, useful as a relative diagnostic.
- **Regularization:** name the object changed — objective, target, data, computation, or checkpoint.
- **Calibration:** probability meaning, measured with ECE and adjusted with temperature; ranking unchanged.
- **Compression & distillation:** approximate the model under a budget; transfer the full distribution, not just the top class.
- **Fine-tuning:** target improvement does not imply retained improvement; anchor and slice metrics make drift visible.

Next: scaling laws, evaluation under shift, and decision-quality metrics for deployed models.




# Mathematics in Deep Learning

Chapter overview slides for the book by Yin Yang.

| | |
|---|---|
| Book page | `foundation-books.github.io/mathematics-in-deep-learning` |
| Code | `github.com/foundation-books/mathematics-in-deep-learning-code` |
| Paperback | `amazon.com/dp/B0H1ZZHDT2` |
| Hardcover | `amazon.com/dp/B0H1ZL11MJ` |
| Kindle | `amazon.com/dp/B0GX32M8SP` |
| License | CC BY-NC-ND 4.0 |

These free overview slides may be shared non-commercially with attribution and without modifications. Full explanations, exercises, and companion-code context are in the book and linked repository.