S.NO. 3 CHEATSHEET

Choose an output head and loss

Work from the target's support and uncertainty, then use a stable implementation of its negative log-likelihood.

Target semantics → predictive distribution → parameterization → stable NLL.

1 · A practical model-selection map

Target	Distribution	Head / NLL
real value	Gaussian	μ (and $\sigma > 0$) / MSE or Gaussian NLL
robust real	Laplace	μ (and $b > 0$) / absolute-error NLL
heavy-tail real	Student-t	$\mu, s > 0, \nu > 0$ / Student-t NLL
binary label	Bernoulli	one logit / BCE with logits
one of K	Categorical	K logits / cross-entropy
count	Poisson	log-rate / Poisson NLL

A loss can be optimized even when its likelihood story is inappropriate. The story is what tells you whether its behaviour matches the data.

2 · Parameterize constraints safely

Let the network emit unconstrained raw values.

$p \in (0, 1)$	$p = \sigma(z)$
$\boldsymbol{p} \in \Delta^{K-1}$	$\boldsymbol{p} = \mathrm{softmax}(\boldsymbol{z})$
$\sigma, \lambda, s > 0$	$\mathrm{softplus}(a) + \varepsilon$ or model a log-scale
$\nu > 0$	$\nu = \mathrm{softplus}(u) + \varepsilon$; often impose a safer lower bound

Do not clamp learned probabilities as the primary model. Use a transformation whose range already obeys the distribution's constraints.

3 · Use logits, not rounded probabilities

Stable binary loss for logit z:

$$\ell(z, y) = \max(z, 0) - zy + \log(1 + \exp(-|z|)).$$

Stable multiclass loss:

$$\ell(\boldsymbol{z}, c) = -z_c + \mathrm{logsumexp}(\boldsymbol{z}).$$

These fused forms avoid explicitly computing $\log(0)$ after sigmoid or softmax saturation.

4 · Heteroscedastic regression

If uncertainty changes with input, predict both μ_i and $\sigma_i > 0$:

$$\ell_i = \tfrac{1}{2}\left[\tfrac{(y_i - \mu_i)^2}{\sigma_i^2} + \log \sigma_i^2\right] + C.$$

The first term standardizes the residual; the second prevents the model from making every σ_i arbitrarily large.

Validate: interval coverage and sharpness, not just point-error MSE.

5 · Robust regression is a modelling choice

For residual $r = y - \mu$:

Gaussian	$\partial \frac{\ell}{\partial r} \propto r$; influence grows with $\lvert r \rvert$
Laplace	$\partial \frac{\ell}{\partial r} \propto \mathrm{sign}(r)$; constant magnitude
Student-t	influence eventually decreases for very large $\lvert r \rvert$

Inspect residuals. A heavy-tailed likelihood is appropriate when unusual observations are plausible, not as a substitute for fixing corrupt data.

6 · Library semantics to verify

`BCEWithLogitsLoss`	raw logits + float targets in $[0, 1]$
`CrossEntropyLoss`	raw logits + integer class indices; no prior softmax
`GaussianNLLLoss`	mean + variance, not standard deviation
`PoissonNLLLoss`	check whether input is log-rate and whether constants are included
`Distribution.log_prob`	inspect event and batch shapes before reducing

Names describe formulas, not tensor conventions. Read the API contract and test a hand-computed example.

The output activation is part of the probabilistic model, not a cosmetic layer after the network.

S.NO. 3 CHEATSHEET

Debug a likelihood-based training pipeline

Check mathematics, tensors and numerical behaviour in that order; a decreasing loss alone is not enough.

1 · First prove one example by hand

Pick one tiny prediction-target pair and compute the expected NLL.

binary $y = 1, p = 0.8$	$-\log 0.8 \approx 0.223$
3-class $c = 2, p_c = 0.7$	$-\log 0.7 \approx 0.357$
Gaussian $r = 2, \sigma = 1$	$\frac{r^2}{2\sigma^2} = 2$ plus constants

Match the library's unreduced output to the hand value before training a model.

2 · Shape audit

- ✓ Batch axis and class / event axes are named.
- ✓ Broadcasting is intentional, not accidental.
- ✓ One categorical target is an index, not a length-K float unless soft labels are intended.
- ✓ Per-pixel or per-token masks remove padded observations before reduction.
- ✓ Start with `reduction="none"` and inspect every contribution.

3 · Reduction and weighting

Decide what one training example contributes:

$$\mathcal{L} = \frac{1}{n} \sum_i w_i \ell_i, \quad w_i \geq 0.$$

If an example contains many pixels or tokens, choose whether to average within each example or across all observations. The two objectives weight long examples differently.

Log the numerator and denominator separately when using masks, class weights or variable-length sequences.

4 · Numerical checks

- ✓ Logits, NLL and gradients are finite on the first batch.
- ✓ Positive scales are bounded away from zero.
- ✓ No duplicated sigmoid / softmax before a fused loss.
- ✓ Extreme logits are tested, not only values near zero.

5 · Behavioural sanity tests

Test	Expected result
perfect direction	loss falls as true-class logit increases or residual shrinks
label permutation	performance collapses toward chance
constant baseline	trained model beats mean / class-frequency predictor
duplicate batch	mean loss unchanged; summed loss doubles
one outlier	Gaussian reacts more strongly than Laplace / Student-t
zero probability	NLL tends to $+\infty$ for the observed event

6 · Diagnose with the right residual

Regression: plot residuals against predictions and inputs; check spread and tails.

Classification: compare confidence with empirical accuracy; inspect per-class NLL.

Counts: compare variance with the mean; strong overdispersion can invalidate Poisson.

A well-optimized misspecified likelihood can still give poor uncertainty and misleading gradients.

7 · Debugging order

1. target meaning and support
2. distribution family and parameter constraints
3. one-example formula and tensor shapes
4. reduction, masking and weighting
5. numerical stability and gradients
6. validation metrics and residual diagnostics

If the hand calculation, unreduced tensor and library result disagree, stop there: the training curve cannot rescue a broken objective.