Logit Gradient Correction: Deriving Score Centering from Logit Gradients

Xiaotian Han · *September 24, 2026*

This post derives SCORE CENTERING from removing a bias term in the logit gradient before it is propagated through the logit Jacobian via chain rule, and then derives the corresponding surrogate objective.[1]

Background: REINFORCE and the Logit Gradient

At a fixed prefix $\boldsymbol{x}$, a trainer with D parameters $\boldsymbol{\theta} \in \mathbb{R}^D$ produces logits $\boldsymbol{z}(\boldsymbol{x}, \boldsymbol{\theta}) \in \mathbb{R}^V$ over a vocabulary of size V. Its probability vector $\boldsymbol{p} = \mathrm{softmax}(\boldsymbol{z}) \in \mathbb{R}^V$ defines the trainer distribution P, with $P(a) = p_a$. A sampler at the same prefix generates token $a \sim Q$, with probabilities $\boldsymbol{q} \in \mathbb{R}^V$ and $Q(a) = q_a$. During backward, the recorded $\boldsymbol{q}$ and reward $R_a \in \mathbb{R}$ are held fixed. Scalars use plain letters, vectors bold lowercase, and matrices bold uppercase.

REINFORCE wants to increase the trainer's expected reward J_P under the trainer's own distribution P:

$$J_P = \mathbb{E}_{a\sim P}[R_a] = \sum_a p_a R_a.$$

Holding the reward function fixed, differentiate J_P with respect to $\boldsymbol{\theta}$ to obtain $\nabla_{\boldsymbol{\theta}} J_P \in \mathbb{R}^D$:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} J_P &= \sum_a R_a \nabla_{\boldsymbol{\theta}} p_a \\ &= \sum_a p_a R_a \nabla_{\boldsymbol{\theta}} \log p_a \\ &= \mathbb{E}_{a\sim P}[R_a \nabla_{\boldsymbol{\theta}} \log p_a].\end{aligned}$$

The term $R_a \nabla_{\boldsymbol{\theta}} \log p_a \in \mathbb{R}^D$ is REINFORCE's per-sample gradient contribution. With R_a held fixed, it equals the negative gradient with respect to $\boldsymbol{\theta}$ of the reward-weighted cross-entropy loss $-R_a \log p_a$.

We first differentiate $\log p_a$ with respect to the logits $\boldsymbol{z}$. Given that $\boldsymbol{p} = \mathrm{softmax}(\boldsymbol{z})$, we have[2]

$$\nabla_{\boldsymbol{z}} \log p_a = \boldsymbol{e}_a - \boldsymbol{p} \in \mathbb{R}^V.$$

Here, $\boldsymbol{e}_a \in \mathbb{R}^V$ is the sampled token's one-hot vector. The gradient $\boldsymbol{e}_a - \boldsymbol{p}$ contains the derivatives of $\log p_a$ with respect to the logits of all vocabulary tokens. The mean logits contribution on the right therefore determines the mean parameter contribution.

Log-Prob Gradients Have Zero/Nonzero Bias when On/off-Policy

We consider a scenario that $R_a = 1$ for every token, we have:

$$\mathbb{E}_{a\sim P}[R_a \nabla_{\boldsymbol{z}} \log p_a] = \mathbb{E}_{a\sim P}[\nabla_{\boldsymbol{z}} \log p_a] = \mathbb{E}_{a\sim P}[\boldsymbol{e}_a - \boldsymbol{p}] = \mathbb{E}_{a\sim P}[\boldsymbol{e}_a] - \boldsymbol{p} \in \mathbb{R}^V.$$

Thus the gradient of J_P with respect to the model parameters $\boldsymbol{\theta}$ will be

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} J_P &= \mathbb{E}_{a\sim P}[R_a \nabla_{\boldsymbol{\theta}} \log p_a] \\ &= \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}}\right)^\top \mathbb{E}_{a\sim P}[R_a \nabla_{\boldsymbol{z}} \log p_a] \\ &= \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}}\right)^\top (\mathbb{E}_{a\sim P}[\boldsymbol{e}_a] - \boldsymbol{p}) \in \mathbb{R}^{D\times V} \times \mathbb{R}^V = \mathbb{R}^D.\end{aligned}$$

[1] This method is introduced in Score Centering Stabilizes Off-Policy Reinforcement Learning, and the perspective presented here is inspired by this discussion.

[2] Same to $\nabla_{\boldsymbol{z}} \mathrm{CE}(\boldsymbol{e}_a, \mathrm{softmax}(\boldsymbol{z})) = \boldsymbol{p} - \boldsymbol{e}_a \in \mathbb{R}^V$.

The matrix $\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}} \in \mathbb{R}^{V \times D}$ is the trainer's logit Jacobian: it describes how the logits change with the model parameters. Its transpose maps a V-dimensional logits gradient to a D-dimensional parameter gradient, applying the chain rule through backpropagation. At a fixed prefix and parameter value, this matrix is the same for every possible sampled token, so it can be moved outside the expectation. Score Centering supplies a corrected logits gradient to this same backward pass.

If the tokens are generated by the trainer distribution P, the expectation $\mathbb{E}_{a\sim P}[\boldsymbol{e}_a]$ equals $\boldsymbol{p}$,

$$\nabla_{\boldsymbol{\theta}} J_P = \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}}\right)^\top (\mathbb{E}_{a\sim P}[\boldsymbol{e}_a] - \boldsymbol{p}) = \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}}\right)^\top (\boldsymbol{p} - \boldsymbol{p}) = \boldsymbol{0}.$$

the logits gradient is unbiased and has zero mean when the reward is constant.

Logit Gradient Correction

If tokens instead come from the sampler distribution Q, then $\mathbb{E}_{a\sim Q}[\boldsymbol{e}_a] = \boldsymbol{q}$. Still assuming $R_a = 1$, the mean trainer log-probability gradient becomes:

$$\begin{aligned}\mathbb{E}_{a\sim Q}[\nabla_{\boldsymbol{\theta}} \log p_a] &= \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}}\right)^\top (\mathbb{E}_{a\sim Q}[\boldsymbol{e}_a] - \boldsymbol{p}) \\ &= \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}}\right)^\top (\boldsymbol{q} - \boldsymbol{p}) \in \mathbb{R}^D.\end{aligned}$$

Here Q specifies where tokens come from; the differentiated log-probability still belongs to P. This expression is the mean sampled contribution, not the gradient of the sampler's expected reward.

Since every reward is 1, the trainer objective $J_P = 1$ has zero gradient. We want the corrected contributions to have zero mean even when tokens come from Q. Shift each one-hot contribution $\boldsymbol{e}_a$ by $\boldsymbol{p} - \boldsymbol{q}$, giving $\mathbb{E}_{a\sim Q}[\boldsymbol{e}_a - \boldsymbol{q} + \boldsymbol{p}] = \boldsymbol{q} - \boldsymbol{q} + \boldsymbol{p} = \boldsymbol{p}$.

Let J denote the per-sample surrogate objective that we will construct in the next section. We require its expected parameter gradient to equal the corrected mean contribution. With Q held fixed during backward, this requirement is:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} \mathbb{E}_{a\sim Q}[J] &= \mathbb{E}_{a\sim Q}[\nabla_{\boldsymbol{\theta}} J] \\ &= \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}}\right)^\top (\mathbb{E}_{a\sim Q}[\boldsymbol{e}_a - \boldsymbol{q} + \boldsymbol{p}] - \boldsymbol{p}) \\ &= \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}}\right)^\top (\mathbb{E}_{a\sim Q}[\boldsymbol{e}_a - \boldsymbol{q} + \boldsymbol{p} - \boldsymbol{p}]) \\ &= \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}}\right)^\top (\mathbb{E}_{a\sim Q}[\boldsymbol{e}_a - \boldsymbol{q}]) \\ &= \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}}\right)^\top (\boldsymbol{q} - \boldsymbol{q}) \\ &= \boldsymbol{0}.\end{aligned}$$

The essential of this correction is to replace each token's logits gradient contribution $\boldsymbol{e}_a - \boldsymbol{p}$ with $\boldsymbol{e}_a - \boldsymbol{q}$. Averaging under the sampler distribution Q then gives $\sum_a q_a(\boldsymbol{e}_a - \boldsymbol{q}) = \boldsymbol{0}$, restoring zero mean for constant rewards.

From Corrected Logit Gradient to a Surrogate Objective

We now construct the per-sample surrogate J. For constant reward $R_a = 1$, the required relation is:

$$\nabla_{\boldsymbol{\theta}}\mathbb{E}_{a\sim Q}[J] = \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}}\right)^{\top} \mathbb{E}_{a\sim Q}[\boldsymbol{e}_a - \boldsymbol{q}].$$

For varying rewards, we weight each corrected contribution by R_a. The required gradient of the expected surrogate is

$$\nabla_{\boldsymbol{\theta}}\mathbb{E}_{a\sim Q}[J] = \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}}\right)^{\top} \mathbb{E}_{a\sim Q}[R_a(\boldsymbol{e}_a - \boldsymbol{q})].$$

First, we seek a scalar surrogate objective J in terms of the logits $\boldsymbol{z}$, and whose gradient with respect to the logits $\boldsymbol{z}$ is:

$$\nabla_{\boldsymbol{z}} J = R_a(\boldsymbol{e}_a - \boldsymbol{q}) \in \mathbb{R}^V.$$

The required gradient $R_a(\boldsymbol{e}_a - \boldsymbol{q})$ is constant with respect to $\boldsymbol{z}$. An intuitive antiderivative is a linear function whose gradient equals its coefficient vector:

$$\begin{aligned} J &= R_a(\boldsymbol{e}_a - \boldsymbol{q})^{\top}\boldsymbol{z} \\ &= R_a\left(\boldsymbol{e}_a^{\top}\boldsymbol{z} - \sum_v \mathrm{sg}[q_v] z_v\right) \\ &= R_a\left(z_a - \sum_v \mathrm{sg}[q_v] z_v\right). \end{aligned}$$

The last line uses $\boldsymbol{e}_a^{\top}\boldsymbol{z} = z_a$. The sum $\sum_v \mathrm{sg}[q_v] z_v$ is the sampler-weighted average of the trainer's logits over the entire vocabulary. The operator sg holds the probabilities fixed during differentiation while preserving gradients through the logits.

Second, we express the surrogate objective J in terms of log-probabilities. Softmax gives $\log p_v = z_v - \log\left(\sum_j \exp(z_j)\right)$ for every token v. Using $\sum_v \mathrm{sg}[q_v] = 1$, we obtain:

$$\begin{aligned} & z_a - \sum_v \mathrm{sg}[q_v] z_v \\ &= \left(z_a - \log\left(\sum_j \exp(z_j)\right)\right) - \left[\sum_v \mathrm{sg}[q_v] z_v - \log\left(\sum_j \exp(z_j)\right)\right] \\ &= \left(z_a - \log\left(\sum_j \exp(z_j)\right)\right) - \left[\sum_v \mathrm{sg}[q_v] z_v - \underbrace{\left(\sum_v \mathrm{sg}[q_v]\right)}_{=1} \log\left(\sum_j \exp(z_j)\right)\right] \\ &= \left(z_a - \log\left(\sum_j \exp(z_j)\right)\right) - \sum_v \mathrm{sg}[q_v]\left(z_v - \log\left(\sum_j \exp(z_j)\right)\right) \\ &= \log p_a - \sum_v \mathrm{sg}[q_v] \log p_v. \end{aligned}$$

We have the surrogate objective in terms of log-probability:

$$\begin{aligned} J &= R_a\left[\log p_a - \sum_v \mathrm{sg}[q_v] \log p_v\right] \\ &= R_a\left[\log p_a - \mathbb{E}_{v\sim Q}[\log p_v]\right]. \end{aligned}$$

The surrogate loss replaces the per-sample REINFORCE objective $R_a \log p_a$ with R_a times the difference between the sampled token's trainer log-probability and the Q-weighted average of the trainer's log-probabilities over the vocabulary.

Averaging the per-sample surrogate over tokens drawn from Q gives:

$$\mathbb{E}_{a\sim Q}[J] = \mathbb{E}_{a\sim Q}\left[R_a\left[\log p_a - \mathbb{E}_{v\sim Q}[\log p_v]\right]\right].$$