BIASED STOCHASTIC GRADIENT FOR THE LOG-SOFTMAX LOSS

Anonymous authors
Paper under double-blind review

ABSTRACT

The log-Softmax loss is widely used in machine learning, but computing its gradient can be expensive when the number of classes is large. Class sampling reduces this cost, but also raises the concern of bias: whether, after class sampling, the expectation of the stochastic gradient equals the full gradient. Such stochastic gradients have generally been suspected to be biased, but a formal proof has long been missing. Recently, Lin et al. (2025) made an important step toward resolving this question by proving that no estimator based on only a subset of class logits can estimate the full Softmax probability without bias. However, their result concerns Softmax estimation at the logit level only, rather than the gradients with respect to model parameters. Moreover, they only consider class sampling for a fixed training instance, whereas stochastic training involves both instance sampling and class sampling. These differences show that the unbiasedness question itself must be stated more carefully, including which stochastic gradient is considered and which sources of randomness are averaged over. Our first contribution is a proper definition on the unbiasedness of stochastic gradient for the log-Softmax loss by taking the model, data, parameter values, sampling procedures, and the randomness over expectation into account. With the proposed definition, our second contribution is to formally prove that, for the log-Softmax loss, no universal estimator can ensure unbiased stochastic gradient under class sampling. In addition, we extend our definition/proof to cover a related problem of minimizing the log-sum-exp form.

1 INTRODUCTION

The log-Softmax loss (also known as the cross-entropy loss) appears in numerous machine learning applications including multi-class classification, contrastive learning, etc. If we take multi-class classification as an example and follow the notation in Lin et al. (2025), the log-Softmax loss takes the following form,

$$-\log \mathrm{Softmax}(\hat{y}^+, \hat{y}_1^-, \dots, \hat{y}_n^-),$$

where $\hat{y}^+ \in \mathbb{R}$ is the logit of the target class, $\hat{y}_i^- \in \mathbb{R}$ for every $1 \le i \le n$ are the logits of all negative classes. This setting assumes $n+1$ total classes. In the log-Softmax loss, the softmax function gives the probability of a class by

$$\mathrm{Softmax}(\hat{y}^+, \hat{y}_1^-, \dots, \hat{y}_n^-) = \frac{e^{\hat{y}^+}}{e^{\hat{y}^+} + \sum_{i=1}^n e^{\hat{y}_i^-}}. \tag{1}$$

By summing over the losses over all training data, we get the total training loss for minimization.

A well-known issue in training a problem with the log-Softmax loss is the high computational cost in calculating the denominator in (1). When using large neural networks, computing all n logits can be extremely expensive. A widely used technique to reduce the computational cost is to sample a subset of k classes for loss calculation, and $k \ll n$ typically (e.g., Yi et al., 2019). An important scenario is the stochastic gradient descent method for minimizing the total training loss. In calculating the stochastic gradient, we only use k sampled classes to calculate the denominator of (1). In contrast, if we consider all $n+1$ classes in the calculation, then we say that we are using the full Softmax.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

For large-scale scenarios, stochastic gradient methods are the major optimization methods to be used. In this work, we consider stochastic gradients in a general sense: randomness may come from instance sampling, class sampling, or both, and the sampling procedure can be arbitrary. This is more general than the naive stochastic gradient method that uniformly samples only one instance at each step. To ensure the equivalence with training a model using the full gradient, which considers all data and classes, we often would like to have an unbiased stochastic gradient satisfying the following condition:

$$\mathbb{E}\left[\text{Stochastic Gradient}\right] = \text{Full Gradient}. \tag{2}$$

If this condition does not hold, the bias may be harmful to the optimization process and solutions.

People have long suspected that for the log-Softmax loss, (2) does not hold. However, this result was never formally given. We note that a formal proof is highly needed because works trying to address the issue of biased stochastic gradient implicitly assume this result. For example, in the works (Wei et al., 2026; Gladin et al., 2026) to minimize the log-Softmax loss and a related problem in the log-sum-exp form, they cite the bias as the reason for developing some reformulation of the optimization problem.

A recent breakthrough in studying (2) was by Lin et al. (2025). They claimed that they have proved the impossibility of (2). However, a careful check of their results reveals that they did not. Instead, what they have proved is the bias of the Softmax function, which is related to but different from the bias of the stochastic gradient. More details are in Section 2. Therefore, we see an essential need to fill this gap, which is the main subject of this work.

Initially we thought the proof was easy. Because the stochastic gradient is strongly related to the Softmax calculation, some ways to connect the results in Lin et al. (2025) should suffice to prove the impossibility of (2). But we quickly found out that the proof is more difficult than we thought. We can see that the gradient (or stochastic gradient) depends not only on the logits, which are outputs of a machine learning model, but also the model parameters. Moreover, what Lin et al. (2025) dealt with is the situation of one single data instance. In contrast, a stochastic gradient algorithm involves a random selection of instances in each step.

After many failed attempts, we learned that the root cause of our inability to do the proof lies in the lack of a proper definition of (2), which we can see is a rough instead of a rigorous mathematical statement. In other words, if we do not know what Eq. (2) means, there is no way that we can prove or disprove it. This situation sounds unbelievable as stochastic gradient methods have been well developed. What we would like to point out is that, though we have a definition of E[Stochastic gradient] under the standard settings of instance sampling, it seems we never discuss a proper definition for situations such as the sampled log Softmax, for which another dimension of sampling logits (or say classes) is conducted. To this end, we must first study a suitable definition for unbiased stochastic gradient using the log-Softmax loss. From that, we were eventually able to rigorously prove the impossibility of (2).

In addition, we extend our definition/proof to cover a related problem of minimizing the log-sum-exp form.

In summary, we make two main contributions in this work:

- We give a proper definition of unbiased stochastic gradients for the log-Softmax loss.
- Based on the proposed definition, we fill the gap left in Lin et al. (2025) and prove that there is no universally unbiased stochastic gradient estimator.

2 Related Work: What Lin et al. (2025) Have Done and the Remaining Gap

Lin et al. (2025) explicitly stated that the condition (2) is what to be checked. However, their result does not establish the impossibility of (2), because their theorem concerns only the logits,

$$\hat{\boldsymbol{y}} := [\hat{y}^+, \hat{y}_1^-, \ldots, \hat{y}_n^-]^\top,$$

without accounting for the model parameters, $\boldsymbol{\theta}$. Specifically, the gradient in (2) is

$$\frac{\partial \,\text{loss}}{\partial \boldsymbol{\theta}} = \frac{\partial \hat{\boldsymbol{y}}}{\partial \boldsymbol{\theta}} \frac{\partial \,\text{loss}}{\partial \hat{\boldsymbol{y}}}, \tag{3}$$

where the first term represents the derivatives of the logits with respect to the model parameters, and the second term represents the derivatives of the loss with respect to the logits.

The analysis of Lin et al. (2025) concerns only the second term, $\partial \operatorname{loss} / \partial \hat{\boldsymbol{y}}$, in (3), which involves the following Softmax calculation:

$$\frac{\partial \operatorname{loss}}{\partial \hat{\boldsymbol{y}}} = \frac{\partial}{\partial \hat{\boldsymbol{y}}} \left(-\log \operatorname{Softmax}(\hat{y}^+, \hat{y}_1^-, \ldots, \hat{y}_n^-) \right),$$

$$\frac{\partial}{\partial \hat{y}^+} \left(\operatorname{loss}(\hat{y}^+, \hat{y}_1^-, \ldots, \hat{y}_n^-) \right) = -1 + \operatorname{Softmax}(\hat{y}^+, \hat{y}_1^-, \ldots, \hat{y}_n^-), \tag{4}$$

$$\frac{\partial}{\partial \hat{y}_i^-} \left(\operatorname{loss}(\hat{y}^+, \hat{y}_1^-, \ldots, \hat{y}_n^-) \right) = \operatorname{Softmax}(\hat{y}_i^-, \hat{y}_1^-, \ldots, \hat{y}_{i-1}^-, \hat{y}^+, \hat{y}_{i+1}^-, \ldots, \hat{y}_n^-), \forall i = 1, \ldots, n. \tag{5}$$

From (4) and (5), Lin et al. (2025) stated "both the loss and gradient computations inevitably rely on the softmax function." Then they showed that any approximation of the Softmax function with access to a subset of logits must be biased. Precisely, when $k < n$, they consider the following set of real-valued functions with $k + 1$ input variables:

$$\mathbb{S}_k = \{ s \mid s : \mathbb{R}^{k+1} \to \mathbb{R} \}.$$

The set covers all possible estimators operating on one positive logit and k sampled negative logits. Lin et al. (2025) proved that

$$\forall s \in \mathbb{S}_k, \ \exists \hat{y}^+, \hat{y}_1^-, \ldots, \hat{y}_n^- \in \mathbb{R}, \text{ s.t.}$$

$$\mathbb{E}_{i_1, \ldots, i_k}[s(\hat{y}^+, \hat{y}_{i_1}^-, \ldots, \hat{y}_{i_k}^-)] \neq \operatorname{Softmax}(\hat{y}^+, \hat{y}_1^-, \ldots, \hat{y}_n^-) = \frac{e^{\hat{y}^+}}{e^{\hat{y}^+} + \sum_{i=1}^n e^{\hat{y}_i^-}}, \tag{6}$$

where k classes $(i_1, \ldots, i_k)$ are randomly sampled from the n classes. Their results extend earlier works, such as Bengio & Senecal (2008); Blanc & Rendle (2018), that consider the following special $s(\cdot)$ function:

$$s(\hat{y}^+, \hat{y}_{i_1}^-, \ldots, \hat{y}_{i_k}^-) := \frac{e^{\hat{y}^+}}{e^{\hat{y}^+} + \sum_{j=1}^k w(i_j) e^{\hat{y}_{i_j}^-}},$$

where $w(i_j), \forall j$ are the weights. Clearly, they approximate the Softmax calculation by replacing the denominator in (1) with a weighted sum of k randomly sampled logits. Regarding the selection of $(i_1, \ldots, i_k)$, many past works do not uniformly select logits. Instead, they conduct hard negative mining to identify more informative classes, such as Dahiya et al. (2023); Gupta et al. (2024).

The discussion shows that, different from their claim, Lin et al. (2025) did not really prove the biased stochastic gradient for the log-Softmax loss. In the following sections, we fill the gap left by them to formally prove that (2) does not hold.

3 Problem Definition

To prove or disprove (2), at first glance we should leverage the existing proof in Lin et al. (2025). However, after several failed attempts, we learned the need to first give a proper definition of stochastic gradient and its unbiasedness. We have mentioned in (3) that stochastic gradient involves not only the Softmax function but also other information such as the model parameters. Further, while (6) considers only class sampling for a fixed instance, practical stochastic gradient methods also involve mini-batch sampling over training instances. Therefore, a study on the stochastic gradient of the log-Softmax loss should take both instance sampling and class sampling into account. As the task involving two types of sampling is complicated, we start in Section 3.1 with the situation involving only instance sampling.

3.1 Unbiasedness of Stochastic Gradient: Situation Only Involving Instance Sampling

To begin, we lay out the machine learning problem to be considered. Each training data point is a pair $(\boldsymbol{x}, \boldsymbol{y})$ so that $\boldsymbol{x}$ is the input and $\boldsymbol{y}$ is the target value (e.g., class label indicator in classification or a real-valued vector in regression).

We consider any machine learning model with the following output function

$$\hat{y}(\boldsymbol{\theta}; \boldsymbol{x}),$$

where $\boldsymbol{\theta}$ is the model parameter and $\boldsymbol{x}$ is any input. Our only requirement on the machine learning model is that $\hat{y}(\boldsymbol{\theta}; \boldsymbol{x})$ is differentiable with respect to $\boldsymbol{\theta}$.

Following the principle of empirical risk minimization, for any $(\boldsymbol{x}, \boldsymbol{y})$ pair of instance and target values, we define a loss function

$$\text{loss}(\boldsymbol{\theta}; \boldsymbol{x}, \boldsymbol{y})$$

to characterize the error between the predicted $\hat{y}(\boldsymbol{\theta}; \boldsymbol{x})$ and the true values $\boldsymbol{y}$. Thus for a given training set $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^l$ of l data points, the training objective is

$$\min_{\boldsymbol{\theta}} \frac{1}{l} \sum_{i=1}^{l} \text{loss}(\boldsymbol{\theta}; \boldsymbol{x}_i, \boldsymbol{y}_i). \tag{7}$$

Gradient descent is an effective method to minimize (7), but gradient calculation can be expensive if the number of data l is huge. Stochastic gradient descent is a well-developed technique to randomly choose a subset B of training instances for calculating an approximate gradient:

$$\frac{1}{|B|} \sum_{i \in B} \frac{\partial}{\partial \boldsymbol{\theta}} \text{loss}(\boldsymbol{\theta}; \boldsymbol{x}_i, \boldsymbol{y}_i). \tag{8}$$

It is well known that the idea is based on the fact that the expectation of stochastic gradient is the same as the full gradient. In many machine learning course materials we can find the following statement

$$\mathbb{E}\left[\frac{1}{|B|} \sum_{i \in B} \frac{\partial}{\partial \boldsymbol{\theta}} \text{loss}(\boldsymbol{\theta}; \boldsymbol{x}_i, \boldsymbol{y}_i)\right] = \frac{1}{l} \sum_{i=1}^{l} \frac{\partial}{\partial \boldsymbol{\theta}} \text{loss}(\boldsymbol{\theta}; \boldsymbol{x}_i, \boldsymbol{y}_i). \tag{9}$$

However, this statement is rough without all the mathematical details. While a rough statement is enough in most discussions, for more complicated situations such as the bias/unbiasedness of the log-Softmax loss to be investigated in this paper, we find that writing down a complete mathematical statement is essential.

We argue that (9) is a rough statement of the following definition.

Definition 3.1. Assume a scheme to sample a subset without replacement is given. The stochastic gradient is unbiased if the following statement holds.

$$\begin{aligned}
&\forall \hat{y}(\boldsymbol{\theta}; \boldsymbol{x}), \forall \text{ training set} \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^l, \\
&\forall \bar{B} \le l, \text{ sample a subset } B \subset \{1, \dots, l\} \text{ with } |B| = \bar{B}, \\
&\forall \bar{\boldsymbol{\theta}}, \qquad \mathbb{E}_B\left[\frac{1}{|B|} \sum_{i \in B} \frac{\partial}{\partial \boldsymbol{\theta}} \text{loss}(\boldsymbol{\theta}; \boldsymbol{x}_i, \boldsymbol{y}_i)\bigg|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}}\right] = \frac{1}{l} \sum_{i=1}^{l} \frac{\partial}{\partial \boldsymbol{\theta}} \text{loss}(\boldsymbol{\theta}; \boldsymbol{x}_i, \boldsymbol{y}_i)\bigg|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}}.
\end{aligned} \tag{10}$$

In the definition, we need $\boldsymbol{\theta}$ and $\bar{\boldsymbol{\theta}}$ because the former is a variable while the latter is a specific value.

The above definition seems to indicate that we can trivially have a rigorous statement by adding "for all" on everything. However, we must think deeply to make some decisions:

- Our definition concerns "a stochastic gradient step" rather than the whole optimization process. An extended definition to cover all stochastic gradient updates is possible, but can be more complicated. For example, as discussed below, the sampling schemes in many practical implementations change across steps.
- We let the sampling scheme for selecting a batch B be pre-specified for several reasons:
  1. We should not have "for any sampling scheme" in the statement of the definition because typically some common ways (e.g., uniform sampling) are enough. That is, the statement may be too strong.
  2. Nor can we have "there exists a sampling scheme" in the definition because we do not want unbiasedness to hold only by using very strange sampling.

3. For non-uniform sampling such as hard-negative mining mentioned in Section 2, their selection corresponds to a sampling distribution at the beginning of each stochastic gradient update. Note that because they select negative classes based on current information, we can say that their sampling schemes change over the stochastic gradient updates.
4. Our setting is similar to Lin et al. (2025, Assumption 3.2), in which they consider one Softmax evaluation. And the sampling scheme is assumed to be given before the evaluation.

- Also note that we assume sampling without replacement. This setting follows most practical implementations.

As a simple illustration, in Appendix A.4, we show the well-known result that under Definition 3.1, the stochastic gradient is unbiased if the minibatch is sampled uniformly without replacement.

A benefit of giving all details in the definition is that it is easy to consider some variants. For example, the following definition is a relaxation of Definition 3.1:

Definition 3.2. Assume a scheme to sample a subset without replacement is given. The stochastic gradient is unbiased if the following statement holds.

$$\begin{aligned}
&\forall \hat{y}(\boldsymbol{\theta}; \boldsymbol{x}), \forall \text{ data distributions,} \\
&\qquad \text{let } \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{l} \text{ be a dataset of } l \text{ i.i.d. samples drawn from the data distribution,} \\
&\forall \bar{B} \le l, \text{ sample a minibatch without replacement } B \subset \{1, \dots, l\} \text{ with } |B| = \bar{B}, \\
&\forall \bar{\boldsymbol{\theta}}, \qquad \mathbb{E}_{\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{l}} \mathbb{E}_B \left[\frac{1}{|B|} \sum_{i \in B} \frac{\partial}{\partial \boldsymbol{\theta}} \operatorname{loss}(\boldsymbol{\theta}; \boldsymbol{x}_i, \boldsymbol{y}_i) \bigg|_{\boldsymbol{\theta} = \bar{\boldsymbol{\theta}}} \right] \\
&\qquad = \mathbb{E}_{\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{l}} \left[\left(\frac{1}{l} \sum_{i=1}^{l} \frac{\partial}{\partial \boldsymbol{\theta}} \operatorname{loss}(\boldsymbol{\theta}; \boldsymbol{x}_i, \boldsymbol{y}_i) \right) \bigg|_{\boldsymbol{\theta} = \bar{\boldsymbol{\theta}}} \right].
\end{aligned} \tag{11}$$

Definition 3.1 obviously implies Definition 3.2 as inside $\mathbb{E}_{\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{l}}[\cdots]$, the data points $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{l}$ are fixed values instead of random variables. Thus we can apply Definition 3.1 to have the equality in (11).

Definition 3.2 is useful when the training data themselves are viewed as random samples from an underlying data distribution, instead of one fixed training set. For example, Goodfellow et al. (2016, Section 8.1.3) explain that when a minibatch consists of samples independently drawn from the data-generating distribution, its gradient can be an unbiased estimator of the gradient of the generalization error. They further use online learning as a typical example, where new training examples continuously arrive from the data-generating distribution.

In this work, we mainly follow Definition 3.1 to consider a training set rather than a data distribution. However, our theoretical results hold for both situations; see more details in Appendix A.3.

3.2 Unbiasedness of Stochastic Gradient: Log-Softmax Loss

Results in Section 3.1, well established in the literature, consider instance sampling to construct the stochastic gradient. However, in using the log-Softmax loss, we have mentioned in Section 2 that class sampling is also applied. We now investigate what a suitable extension of Definition 3.1 should be. We follow Lin et al. (2025) to consider a multi-class setting of $n+1$ possible class labels. Thus, for any instance $\boldsymbol{x}$, its label

$$y \in \{1, \dots, n+1\}$$

indicates that $\boldsymbol{x}$ is associated with one of the $n+1$ classes. Therefore, our output function contains $n+1$ components:

$$\hat{y}(\boldsymbol{\theta}; \boldsymbol{x}) = \begin{bmatrix} \hat{y}_1(\boldsymbol{\theta}; \boldsymbol{x}) \\ \vdots \\ \hat{y}_{n+1}(\boldsymbol{\theta}; \boldsymbol{x}) \end{bmatrix} \in \mathbb{R}^{n+1}.$$

Our loss is now

$$\begin{aligned}
&\operatorname{loss}(\boldsymbol{\theta}; \boldsymbol{x}, y) \\
&= -\log \Big(\operatorname{Softmax} \big(\hat{y}_y(\boldsymbol{\theta}; \boldsymbol{x}), \hat{y}_1(\boldsymbol{\theta}; \boldsymbol{x}), \dots \hat{y}_{y-1}(\boldsymbol{\theta}; \boldsymbol{x}), \hat{y}_{y+1}(\boldsymbol{\theta}; \boldsymbol{x}), \dots \hat{y}_{n+1}(\boldsymbol{\theta}; \boldsymbol{x}) \big) \Big).
\end{aligned}$$

We consider the following abbreviation for the Softmax function

$$\mathrm{Softmax}(\hat{y}_y(\boldsymbol{\theta};\boldsymbol{x}), \hat{y}_{\mathrm{All}\backslash\{y\}}(\boldsymbol{\theta};\boldsymbol{x})).$$

We then check the full gradient. From chain rule,

$$\frac{\partial}{\partial \boldsymbol{\theta}} \mathrm{loss}(\boldsymbol{\theta};\boldsymbol{x},y) = \frac{\partial \hat{y}(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}} \frac{\partial \,\mathrm{loss}(\boldsymbol{\theta};\boldsymbol{x},y)}{\partial \hat{y}},$$

where

$$\frac{\partial \hat{y}(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}} = \underbrace{\left[\frac{\partial \hat{y}_1(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}}, \ldots, \frac{\partial \hat{y}_{n+1}(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}}\right]}_{n+1 \text{ columns}} \text{ and } \frac{\partial \,\mathrm{loss}(\boldsymbol{\theta};\boldsymbol{x},y)}{\partial \hat{y}} \in \mathbb{R}^{n+1}.$$

By the definition of the loss function, we have

$$\frac{\partial \,\mathrm{loss}(\boldsymbol{\theta};\boldsymbol{x},y)}{\partial \hat{y}_j} = \begin{cases} -1 + \mathrm{Softmax}(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}), \hat{y}_{\mathrm{All}\backslash\{j\}}(\boldsymbol{\theta};\boldsymbol{x})) & \text{if } j = y, \\ \mathrm{Softmax}(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}), \hat{y}_{\mathrm{All}\backslash\{j\}}(\boldsymbol{\theta};\boldsymbol{x})) & \text{otherwise.} \end{cases}$$

Therefore,

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\theta}} \mathrm{loss}(\boldsymbol{\theta};\boldsymbol{x},y) = (-1 + \mathrm{Softmax}(\hat{y}_y(\boldsymbol{\theta};\boldsymbol{x}), \hat{y}_{\mathrm{All}\backslash\{y\}}(\boldsymbol{\theta};\boldsymbol{x}))) \frac{\partial \hat{y}_y(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}} + \\ \sum_{j:j\neq y} \mathrm{Softmax}(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}), \hat{y}_{\mathrm{All}\backslash\{j\}}(\boldsymbol{\theta};\boldsymbol{x})) \frac{\partial \hat{y}_j(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}}. \end{aligned} \tag{12}$$

Because (12) involves calculating all $n+1$ logits, to reduce the cost, we randomly select a subset

$$J \subset \{1, \ldots, n+1\}\backslash\{y\} \text{ with } |J| = k. \tag{13}$$

Our hope is that by considering only logits in

$$J \cup \{y\},$$

we can well approximate the full gradient in (12). To this end, if we follow (6) in Lin et al. (2025), it seems that we need a magic function $s : \mathbb{R}^{k+1} \to \mathbb{R}$ so that

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\theta}} \mathrm{loss}(\boldsymbol{\theta};\boldsymbol{x},y) \approx (-1 + s(\hat{y}_y(\boldsymbol{\theta};\boldsymbol{x}), \hat{y}_J(\boldsymbol{\theta};\boldsymbol{x}))) \frac{\partial \hat{y}_y(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}} + \\ \sum_{j:j\in J} s(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}), \hat{y}_{J_j}(\boldsymbol{\theta};\boldsymbol{x})) \frac{\partial \hat{y}_j(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}}, \end{aligned} \tag{14}$$

where

$$J_j = J \cup \{y\}\backslash\{j\}. \tag{15}$$

In the $s(\cdot)$ function, we assume that the second part of the input, i.e., $\hat{y}_J(\boldsymbol{\theta};\boldsymbol{x})$ or $\hat{y}_{J_j}(\boldsymbol{\theta};\boldsymbol{x})$ in (14), is arranged to have class indices in ascending order.

In (14), we may understand that

$$s(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}), \hat{y}_{J_j}(\boldsymbol{\theta};\boldsymbol{x})) \text{ aims to approximate } \mathrm{Softmax}(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}), \hat{y}_{\mathrm{All}\backslash\{j\}}(\boldsymbol{\theta};\boldsymbol{x})).$$

However, there exists a discrepancy between $\sum_{j:j\in J}$ in (14) and $\sum_{j:j\neq y}$ in (12). The former involves $|J|$ terms, but the latter involves n terms. In Appendix A.1, we justify the use of (14), present some alternatives, and show that these variants do not affect our proof. Such concerns were not touched by past works, such as Bengio & Senecal (2008); Blanc & Rendle (2018); Lin et al. (2025), as they solely focused on the Softmax function. In other words, they did not fully check details of the stochastic gradient.

Finally, we can have the stochastic gradient in our training algorithm. We still select a batch B of training instances. For each instance $\boldsymbol{x}_i$ selected in the batch, according to y_i, we select a corresponding set J^i as in (13). Then we can define the following stochastic gradient at any model parameter $\bar{\boldsymbol{\theta}}$:

$$\frac{1}{|B|} \sum_{i\in B} \text{Eq. (14) with } \boldsymbol{x}_i, y_i, J^i \Big|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}}.$$

We then formally define what we mean by stochastic gradient being unbiased.

Definition 3.3. Assume we are given the following information.

- Number of sampled classes $k \geq 1$.
- A scheme to sample a subset of instances without replacement.
- A scheme to sample a subset of classes without replacement.

The stochastic gradient using the log-Softmax loss is unbiased if the following statement holds.

$$
\begin{aligned}
&\exists s : \mathbb{R}^{k+1} \to \mathbb{R}, \forall \text{ training set} \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{l} \text{ with } n+1 \text{ classes, and } n > k, \forall \hat{y}(\boldsymbol{\theta}; \boldsymbol{x}) \in \mathbb{R}^{n+1}, \\
&\forall \bar{B} \leq l, \text{ sample a minibatch without replacement } B \subset \{1, \ldots, l\} \text{ with } |B| = \bar{B}, \text{ and} \\
&\qquad \text{for each } i \in B, \text{ sample a subset without replacement } J^i \subseteq \{1, \ldots, n+1\} \setminus \{y_i\}, \\
&\forall \bar{\boldsymbol{\theta}}, \qquad \mathbb{E}_B \mathbb{E}_{J^i, i \in B} \left[\frac{1}{|B|} \sum_{i \in B} \text{Eq. (14) with } \boldsymbol{x}_i, y_i, J^i \middle|_{\boldsymbol{\theta} = \bar{\boldsymbol{\theta}}} \right] \\
&\qquad = \left(\frac{1}{l} \sum_{i=1}^{l} \frac{\partial}{\partial \boldsymbol{\theta}} \operatorname{loss}(\boldsymbol{\theta}; \boldsymbol{x}_i, y_i) \right) \middle|_{\boldsymbol{\theta} = \bar{\boldsymbol{\theta}}}.
\end{aligned}
\tag{16}
$$

We assume k is given for defining the $s(\cdot)$ function. Then, $n > k$ is any value indicating the number of classes. Also, the number of instances l is arbitrary. While numerous works have mentioned the bias of stochastic gradient under instance and class sampling for the log-Softmax loss, ours may be the first to give a formal and complete definition. Note that without a suitable definition, there is no way to prove our main result that stochastic gradient is biased.

In this definition, we consider the same approximation function $s(\cdot)$ for every instance. We can extend this setting to, for example,

$$
\begin{aligned}
&\text{split } B \text{ into } B_1, \ldots, B_{10}, \\
&\forall i \in B_1, \text{ use } S_1(\cdot) \text{ on } k_1 \text{ sampled logits}, \\
&\vdots \\
&\forall i \in B_{10}, \text{ use } S_{10}(\cdot) \text{ on } k_{10} \text{ sampled logits}.
\end{aligned}
$$

We do not consider such extensions for simplicity.

Our aim is to prove the opposite of Definition 3.3 in the following theorem:

Theorem 3.4. *Consider the same assumptions in Definition 3.3. For the log-Softmax loss, the stochastic gradient is biased because the following statement holds.*

$$
\begin{aligned}
&\forall s : \mathbb{R}^{k+1} \to \mathbb{R}, \exists \textit{ a training set} \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{l} \textit{ with } n+1 \textit{ classes, and } n > k, \exists \hat{y}(\boldsymbol{\theta}; \boldsymbol{x}) \in \mathbb{R}^{n+1}, \\
&\exists \bar{B} \leq l, \textit{sample a minibatch without replacement } B \subset \{1, \ldots, l\} \textit{ with } |B| = \bar{B}, \textit{ and} \\
&\qquad \textit{for each } i \in B, \textit{ sample a subset without replacement } J^i \subseteq \{1, \ldots, n+1\} \setminus \{y_i\}, \\
&\exists \bar{\boldsymbol{\theta}}, \qquad \mathbb{E}_B \mathbb{E}_{J^i, i \in B} \left[\frac{1}{|B|} \sum_{i \in B} \textit{Eq. (14) with } \boldsymbol{x}_i, y_i, J^i \middle|_{\boldsymbol{\theta} = \bar{\boldsymbol{\theta}}} \right] \\
&\qquad \neq \left(\frac{1}{l} \sum_{i=1}^{l} \frac{\partial}{\partial \boldsymbol{\theta}} \operatorname{loss}(\boldsymbol{\theta}; \boldsymbol{x}_i, y_i) \right) \middle|_{\boldsymbol{\theta} = \bar{\boldsymbol{\theta}}}.
\end{aligned}
$$

We give the proof of Theorem 3.4 in Section 4.

Following the modification from Definition 3.1 to Definition 3.2, we can consider a data distribution instead of a data set in Definition 3.3. Theorem 3.4 can be modified to Corollary A.1 with details in Appendix A.3.

3.3 Relation to Existing Definitions of Stochastic-Gradient Bias

In this section, through the difference between Definitions 3.1 and 3.3, we connect to some past works that have also discussed the bias of stochastic gradient.

In Definition 3.1, the only randomness comes from instance sampling, so we have

$$\mathbb{E}_B[\cdots].$$

Definition 3.3 involves additional randomness from class sampling, leading to the expectation:

$$\mathbb{E}_B\mathbb{E}_{J^i,\, i\in B}[\cdots]. \tag{17}$$

They differ in ways to approximate the full gradient. Thus, we can consider the following more general form:

$$\mathbb{E}_{\text{Randomness of Stochastic Gradient}}\left[\cdots\right]. \tag{18}$$

Previous works have defined the difference between (18) and the full gradient as the bias. For example, Bottou et al. (2018) define

$$\text{Bias} := \mathbb{E}_\xi\left[\text{Stochastic Gradient}\right] - \text{Full Gradient}, \tag{19}$$

where ξ is a random variable representing the randomness of the stochastic gradient at the current iteration.

While (19) is a general definition, similar to (2), the statement is insufficient to check the existence of an unbiased estimator. We need clear specifications as in Definition 3.3 on the model $\hat{\boldsymbol{y}}(\boldsymbol{\theta};\boldsymbol{x})$, the dataset, the parameter values, the instance/class sampling, and the corresponding expectations. Our definition provides a useful basis for future studies of similar bias and impossibility questions.

4 Proof of Theorem 3.4

We prove the result by contradiction. If the theorem is wrong, then its opposite statement is correct. That is, (16) holds. We will prove that this leads to a contradiction to the result in Lin et al. (2025).

4.1 Considering Only One Training Instance

In (16), we have

$$\text{for all training sets } \{(\boldsymbol{x}_i, y_i)\}_{i=1}^l.$$

The statement of course holds for one single training instance. After incorporating (14), we have

$$\begin{aligned}
&\exists s : \mathbb{R}^{k+1} \to \mathbb{R}, \forall \hat{y}(\boldsymbol{\theta};\boldsymbol{x}) \in \mathbb{R}^{n+1}, \forall \text{ data point } (\bar{\boldsymbol{x}}, \bar{y}),\\
&\qquad \text{let } J \text{ be a subset randomly drawn from } \{1,\ldots,n+1\}\backslash\{\bar{y}\},\\
&\forall \bar{\boldsymbol{\theta}},\ \ \mathbb{E}_J\Big[\ \text{Eq. (14) with } \bar{\boldsymbol{x}}, \bar{y}, J|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}}\ \Big] = \left(\frac{\partial}{\partial \boldsymbol{\theta}}\,\text{loss}(\boldsymbol{\theta};\bar{\boldsymbol{x}},\bar{y})\right)\Bigg|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}}.
\end{aligned} \tag{20}$$

Here we distinguish the data point $\bar{\boldsymbol{x}}$ from the general input $\boldsymbol{x}$ of $\hat{y}(\cdot)$.

Our idea of using (20) is that if this single-instance statement already leads to a contradiction, then we do not need to worry about the more complicated situation of multiple instances.

4.2 Connection to the Proof in Lin et al. (2025)

Our strategy is to choose some special settings among those "for all" items. First we choose

$$\hat{y}(\theta;\boldsymbol{x}) = \begin{bmatrix} \theta + x_1 \\ x_2 \\ \vdots \\ x_{n+1} \end{bmatrix} \text{ with } \theta \in \mathbb{R} \text{ and } \boldsymbol{x} \in \mathbb{R}^{n+1}. \tag{21}$$

Clearly,

$$\frac{\partial \hat{y}(\theta;\boldsymbol{x})}{\partial \theta} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{22}$$

In (20), we can consider only training instances with class label $\bar{y} = 1$. Further, we let $\bar{\theta} = 0$. From (14) and (22), our stochastic gradient is

$$-1 + s(\hat{y}_1(0; \bar{\boldsymbol{x}}), \hat{y}_J(0; \bar{\boldsymbol{x}})). \tag{23}$$

Then we can rewrite (20) as

$$\begin{aligned} &\exists s : \mathbb{R}^{k+1} \to \mathbb{R}, \hat{y}(\theta; \boldsymbol{x}) \in \mathbb{R}^{n+1} \text{ as in } (21), \forall \bar{\boldsymbol{x}}, \\ &\quad \text{let } J \text{ be a subset randomly drawn from } \{2, \ldots, n+1\}, \\ &\quad \mathbb{E}_J \left[-1 + s(\hat{y}_1(0; \bar{\boldsymbol{x}}), \hat{y}_J(0; \bar{\boldsymbol{x}}))\right] = -1 + \mathrm{Softmax}(\hat{y}(0; \bar{\boldsymbol{x}})). \end{aligned} \tag{24}$$

From (21), we have

$$\hat{y}_1(0; \bar{\boldsymbol{x}}) = \bar{x}_1, \hat{y}_J(0; \bar{\boldsymbol{x}}) = \bar{\boldsymbol{x}}_J, \hat{y}(0; \bar{\boldsymbol{x}}) = \bar{\boldsymbol{x}}. \tag{25}$$

In the end, we get

$$\begin{aligned} &\exists s : \mathbb{R}^{k+1} \to \mathbb{R}, \forall \bar{\boldsymbol{x}} \in \mathbb{R}^{n+1}, \\ &\quad \text{let } J \text{ be a subset randomly drawn from } \{2, \ldots, n+1\}, \\ &\quad \mathbb{E}_J \left[s(\bar{x}_1, \bar{\boldsymbol{x}}_J)\right] = \mathrm{Softmax}(\bar{\boldsymbol{x}}), \end{aligned} \tag{26}$$

which is the opposite statement of Corollary 3.3 in Lin et al. (2025). Thus, we get a contradiction.

5 Minimizing Log-Sum-Exp

Log-sum-exp problems are in a very close form to the log-Softmax loss. The main difference is that the log-Softmax loss involves labels for data classification, but log-sum-exp does not. In Appendix A.2, we point out that this difference leads to different ways of obtaining stochastic gradient. For log-Softmax, we may always keep the logit corresponding to the positive label of each instance, while sampling the rest; see (12) and (14). We show that the proof in Section 4 is simpler because of taking this property. Because the log-sum-exp form does not involve labels, we must develop more sophisticated proof techniques. Finally, in Appendix A.2, we prove the same impossibility result.

6 Conclusions

In this work, we study whether class sampling can provide an unbiased stochastic gradient for the log-Softmax loss. Such an estimator has long been believed to be impossible, but a formal proof has remained missing. A recent important work (Lin et al., 2025) made substantial progress by proving that the full Softmax probability cannot be unbiasedly estimated from only sampled class logits. However, a careful examination reveals that this result does not establish the impossibility of an unbiased stochastic gradient with respect to model parameters.

While trying to bridge this gap, we identified an important reason why establishing such an impossibility result has been difficult: the underlying unbiasedness problem has not been rigorously defined. Our first contribution is therefore a proper definition of this unbiasedness problem. The definition explicitly accounts for the model, parameter values, data, instance sampling, class sampling, and the corresponding expectations, making clear what it means for stochastic gradient to be unbiased.

Our second contribution is to prove the corresponding impossibility result under this definition. We show that no universal estimator can provide an unbiased stochastic gradient under class sampling, thereby closing the gap between previous Softmax-estimation results (Bengio & Senecal, 2008; Blanc & Rendle, 2018; Lin et al., 2025) and stochastic-gradient unbiasedness. Beyond this result, our definition provides a useful basis for future studies of stochastic-gradient bias under more complicated sampling settings.

AI USE STATEMENT

We used large language models to assist with proofreading and identifying potential errors in the manuscript, including the proofs, and to suggest improvements to the clarity. All suggestions were reviewed by the authors, and all revisions were made by the authors. Tasks involving experiments or datasets are not applicable to this purely theoretical work. We take full responsibility for the final content of this paper.

ETHICS STATEMENT

This paper presents theoretical research and does not involve human participants, experiments, or the collection or use of data. We do not foresee any specific ethical concerns arising directly from the theoretical results presented in this work.

REPRODUCIBILITY STATEMENT

This paper presents exclusively theoretical results and does not involve experiments or datasets. To facilitate verification of our results, we state all the settings explicitly and provide complete proofs in the main text and appendices.

REFERENCES

Yoshua Bengio and Jean-Sébastien Senecal. Adaptive importance sampling to accelerate training of a neural probabilistic language model. *IEEE Transactions on Neural Networks*, 19(4):713–722, 2008.

Guy Blanc and Steffen Rendle. Adaptive sampled softmax with kernel based sampling. In *International Conference on Machine Learning*, pp. 590–599, 2018.

Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.

Kunal Dahiya, Nilesh Gupta, Deepak Saini, Akshay Soni, Yajun Wang, Kushal Dave, Jian Jiao, Gururaj K, Prasenjit Dey, Amit Singh, Deepesh Hada, Vidit Jain, Bhawna Paliwal, Anshul Mittal, Sonu Mehta, Ramachandran Ramjee, Sumeet Agarwal, Purushottam Kar, and Manik Varma. NGAME: negative mining-aware mini-batching for extreme classification. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 258–266, 2023.

Egor Gladin, Alexey Kroshnin, Jia-Jie Zhu, and Pavel Dvurechensky. Improved stochastic optimization of logsumexp. In *Proceedings of the Forty-Third International Conference on Machine Learning (ICML)*, 2026.

Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. The MIT Press, 2016.

Nilesh Gupta, Devvrit, Ankit Singh Rawat, Srinadh Bhojanapalli, Prateek Jain, and Inderjit S. Dhillon. Dual-encoders for extreme multi-label classification. In *Proceedings of the Twelfth International Conference on Learning Representations (ICLR)*, 2024.

Li-Chung Lin, Yaxu Liu, and Chih-Jen Lin. Sampled estimators for softmax must be biased. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, 2025. URL `https://www.csie.ntu.edu.tw/~cjlin/papers/softmax_biased/softmax_bias.pdf`.

Xiyuan Wei, Linli Zhou, Bokun Wang, Chih-Jen Lin, and Tianbao Yang. A geometry-aware efficient algorithm for compositional entropic risk minimization. In *Proceedings of the Forty-Third International Conference on Machine Learning (ICML)*, 2026.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *Proceedings of the Ninth International Conference on Learning Representations (ICLR)*, 2021.

Xinyang Yi, Ji Yang, Lichan Hong, Derek Zhiyuan Cheng, Lukasz Heldt, Aditee Kumthekar, Zhe Zhao, Li Wei, and Ed Chi. Sampling-bias-corrected neural modeling for large corpus item recommendations. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pp. 269–277, 2019.

A Appendix

A.1 Discussion on Using (14) as Stochastic Gradient

From the definition of the Softmax function,

$$1 - \text{Softmax}(\hat{y}_y, \hat{y}_{\text{All}\backslash\{y\}}) = \sum_{j:j\neq y} \text{Softmax}(\hat{y}_j, \hat{y}_{\text{All}\backslash\{j\}}). \tag{27}$$

Thus, (12) is equivalent to

$$\sum_{j:j\neq y} \text{Softmax}(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}), \hat{y}_{\text{All}\backslash\{j\}}(\boldsymbol{\theta};\boldsymbol{x})) \left(\frac{\partial \hat{y}_j(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}} - \frac{\partial \hat{y}_y(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}}\right). \tag{28}$$

It is reasonable to approximate $\text{Softmax}(\cdot)$ by $s(\cdot)$ and have the following stochastic gradient:

$$\sum_{j:j\in J} s(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}), \hat{y}_{J\cup\{y\}\backslash\{j\}}(\boldsymbol{\theta};\boldsymbol{x})) \left(\frac{\partial \hat{y}_j(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}} - \frac{\partial \hat{y}_y(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}}\right). \tag{29}$$

If our $s(\cdot)$ function satisfies

$$\sum_{j:j\in J\cup\{y\}} s(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}), \hat{y}_{J\cup\{y\}\backslash\{j\}}(\boldsymbol{\theta};\boldsymbol{x})) = 1, \tag{30}$$

then (29) can be written back as (14). Our derivation therefore justifies the use of (14) as stochastic gradient. Indeed, many hard-negative mining works (Gupta et al., 2024; Xiong et al., 2021) consider (14). Specifically, they mine a set $J \subset \{1,\dots,n+1\}\backslash\{y\}$ and have

$$s(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}), \hat{y}_{J\cup\{y\}\backslash\{j\}}(\boldsymbol{\theta};\boldsymbol{x})) = \frac{e^{\hat{y}_j}}{e^{\hat{y}_j} + \sum_{j':j'\in J\cup\{y\}\backslash\{j\}} e^{\hat{y}_{j'}}}, \forall j \in J\cup\{y\}.$$

Clearly, (30) holds.

In (29), we conduct approximations in two places

$$\sum_{j:j\neq y} \to \sum_{j:j\in J} \text{ and } \text{Softmax}(\cdot) \to s(\cdot).$$

One may wonder if doing two approximations together is reasonable. One reason is to maintain that the $s(\cdot)$ function always has $k+1$ input variables. Now we also see that the condition (30) helps. From (27) and (28), the full gradient is almost a convex combination if we do not consider $\text{Softmax}(\hat{y}_y, \hat{y}_{\text{All}\backslash\{y\}})$. With (30), we have the same property for the stochastic gradient. In (29), we may implicitly impose larger weights on the selected terms in J to compensate the reduction on the number of terms in the summation.

On the other hand, if our considered $s(\cdot)$ does not satisfy (30), we may slightly modify (14). We begin by explaining the discrepancy between $\sum_{j:j\in J}$ and $\sum_{j:j\neq y}$ in the following way. First, we can re-write (12) as

$$\begin{aligned}&\frac{\partial}{\partial \boldsymbol{\theta}}\text{loss}(\boldsymbol{\theta};\boldsymbol{x},y)\\ =&-\frac{\partial \hat{y}_y(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}} + \sum_{j=1}^{n+1} \text{Softmax}(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}), \hat{y}_{\text{All}\backslash\{j\}}(\boldsymbol{\theta};\boldsymbol{x}))\frac{\partial \hat{y}_j(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}}.\end{aligned}$$

Clearly, in (14), we do not randomly sample from all $n+1$ terms. Instead, we always keep the term with $j=y$, while sampling the remaining n terms. With this explanation, we consider the following alternatives of (14):

1. We do not keep the $j=y$ term, and sample from all $n+1$ terms. Under this setting, we replace (13) with the following selection of J:

$$J \subset \{1,\dots,n+1\} \text{ with } |J| = k+1. \tag{31}$$

Next, we use

$$\sum_{j:j\in J} s(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}),\hat{y}_{J_j}(\boldsymbol{\theta};\boldsymbol{x}))\frac{\partial \hat{y}_j(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}} \text{ with } J_j = J\backslash\{j\} \tag{32}$$

to approximate

$$\sum_{j=1}^{n+1} \mathrm{Softmax}(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}),\hat{y}_{\mathrm{All}\backslash\{j\}}(\boldsymbol{\theta};\boldsymbol{x}))\frac{\partial \hat{y}_j(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}}.$$

Note that in this approximation, $s(\cdot)$ implicitly involves a scaling factor to address the use of $|J|$ versus $n+1$ terms in two summations.

2. If we insist on having the $j = y$ term, we can consider two functions $s^+(\cdot)$ and $s^-(\cdot)$ such that

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{\theta}}\mathrm{loss}(\boldsymbol{\theta};\boldsymbol{x},y) \approx (-1 + s^+(\hat{y}_y(\boldsymbol{\theta};\boldsymbol{x}),\hat{y}_J(\boldsymbol{\theta};\boldsymbol{x})))\frac{\partial \hat{y}_y(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}} + \\ \sum_{j:j\in J} s^-(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}),\hat{y}_{J_j}(\boldsymbol{\theta};\boldsymbol{x}))\frac{\partial \hat{y}_j(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}}.\end{aligned} \tag{33}$$

Both remedies are reasonable, but for this paper, we simply consider (14) for the following reasons. First, we keep the $j = y$ term for consistency with the discussion in Lin et al. (2025). Second, our proof can be extended to these settings. See the details in Sections A.1.1 and A.1.2.

A.1.1 Proof for the Case with (31)

The stochastic gradient is now

$$-\frac{\partial \hat{y}_y(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}} + \sum_{j:j\in J} s(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x}),\hat{y}_{J_j}(\boldsymbol{\theta};\boldsymbol{x}))\frac{\partial \hat{y}_j(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}}.$$

If we do not consider the term $-\partial \hat{y}_y(\boldsymbol{\theta};\bar{\boldsymbol{x}})/\partial\boldsymbol{\theta}$, the stochastic gradient and the full gradient are in the same form as the situation of log-sum-exp problems in Section A.2. The same proof can be used to show the impossibility of having $s(\cdot)$.

A.1.2 Proof for the Case with (33)

If we consider (33) as our stochastic gradient, the proof in Section 4 remains the same. The reason is that in our proof it is sufficient to check the first term of (14). Thus the same proof establishes the impossibility of $s^+(\cdot)$ in (33). Because our requirement is the existence of both $s^+(\cdot)$ and $s^-(\cdot)$ in (33), this result is sufficient for the proof.

A.2 Extension to Log-Sum-Exp

Many applications, like the entropy-regularized optimal transport and distributionally robust optimization with KL divergence mentioned in Gladin et al. (2026), involve the following log-sum-exp loss:

$$\mathrm{loss}(\boldsymbol{\theta};\boldsymbol{x}) = \log \sum_{j=1}^{n+1} \exp(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x})). \tag{34}$$

Unlike multi-class classification, here we do not have labels. Then, we have

$$\frac{\partial \,\mathrm{loss}(\boldsymbol{\theta};\boldsymbol{x})}{\partial \hat{y}_j} = \mathrm{Softmax}(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x});\hat{y}_{\mathrm{All}\backslash\{j\}}(\boldsymbol{\theta};\boldsymbol{x})),$$

and thus

$$\frac{\partial \,\mathrm{loss}(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}} = \sum_{j=1}^{n+1} \mathrm{Softmax}(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x});\hat{y}_{\mathrm{All}\backslash\{j\}}(\boldsymbol{\theta};\boldsymbol{x}))\frac{\partial \hat{y}_j(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}}.$$

We select $J \subset \{1,\ldots,n+1\}$ with $|J| = k+1$, and have the following stochastic gradient:

$$\sum_{j\in J} s(\hat{y}_j(\boldsymbol{\theta};\boldsymbol{x});\hat{y}_{J_j}(\boldsymbol{\theta};\boldsymbol{x}))\frac{\partial \hat{y}_j(\boldsymbol{\theta};\boldsymbol{x})}{\partial \boldsymbol{\theta}}, \tag{35}$$

where $J_j = J \backslash \{j\}$. We then extend the proof in Section 4 to prove the impossibility of $s(\cdot)$. Interestingly, Section 4 takes the fact that the log-Softmax loss involves labels into account, so the proof is simpler. Here we must develop some new proof techniques.

To begin, we replace the equality in (20) with

$$\mathbb{E}_J\Big[\ (35) \text{ with } \bar{\boldsymbol{x}}, J|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}}\ \Big] = \left.\frac{\partial \operatorname{loss}(\boldsymbol{\theta}; \bar{\boldsymbol{x}})}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}}.$$

Then, by the settings in (20) and (21), and with $\theta = 0$, the stochastic gradient is

$$\mathbb{E}_J\Big[s(\hat{y}_j(0; \bar{\boldsymbol{x}}), \hat{y}_{J_j}(0; \bar{\boldsymbol{x}}))\mathbb{I}\left[j = 1\right]\Big], \tag{36}$$

where $\mathbb{I}\left[\cdot\right]$ is an indicator function. In Section 4, we get rid of the expectation by considering instances with label $\bar{y} = 1$; see (22) and (23). Now we no longer have labels, so we must compute the expectation.

With $\hat{y}_j(0; \bar{\boldsymbol{x}}) = \bar{x}_j$ and $\hat{y}_{J_j}(0; \bar{\boldsymbol{x}}) = \bar{\boldsymbol{x}}_{J_j}$, we can rewrite (36) as

$$\mathbb{E}_J\left[s(\bar{x}_1, \bar{\boldsymbol{x}}_{J_1})\mathbb{I}\left[1 \in J\right]\right]. \tag{37}$$

We calculate the expectation in (37) as

$$\sum_J \Pr(J, 1 \in J) s(\bar{x}_1, \bar{\boldsymbol{x}}_{J \backslash \{1\}}). \tag{38}$$

Define

$$\Delta = \sum_J \Pr(J, 1 \in J).$$

We have

$$\begin{aligned}(38) =\ & \Delta \sum_J \frac{\Pr(J, 1 \in J)}{\Delta} s(\bar{x}_1, \bar{\boldsymbol{x}}_{J \backslash \{1\}}) \\ =\ & \Delta \mathbb{E}_{\bar{J}}\left[s(\bar{x}_1, \bar{\boldsymbol{x}}_{\bar{J}})\right],\end{aligned}$$

where $\bar{J}$ is a subset randomly drawn from $\{2, \ldots, n+1\}$. What we have done is to create a conditional distribution of J by considering only the situation of $1 \in J$.

Because our distribution on sampling J is pre-defined, Δ is a constant. In the end, (26) becomes

$$\begin{aligned}&\exists s : \mathbb{R}^{k+1} \to \mathbb{R}, \forall \bar{\boldsymbol{x}} \in \mathbb{R}^{n+1}, \\ &\quad \text{let } \bar{J} \text{ be a subset randomly drawn from } \{2, \ldots, n+1\}, \\ &\quad \Delta \mathbb{E}_{\bar{J}}\left[s(\bar{x}_1, \bar{\boldsymbol{x}}_{\bar{J}})\right] = \operatorname{Softmax}(\bar{\boldsymbol{x}}).\end{aligned}$$

Clearly, if we consider $\bar{s}(\cdot) = \Delta s(\cdot)$, then

$$\mathbb{E}_{\bar{J}}\left[\bar{s}(\bar{x}_1, \bar{\boldsymbol{x}}_{\bar{J}})\right] = \operatorname{Softmax}(\bar{\boldsymbol{x}}),$$

leading to a contradiction.

Earlier we implicitly assumed $\Delta > 0$. We consider the special case of $\Delta = 0$. In this situation, $\Pr(J, 1 \in J) = 0, \forall J$. That is, in sampling $\{1, \ldots, n+1\}$, we never get the index 1. In this situation, (37) and (38) imply

$$\operatorname{Softmax}(\bar{\boldsymbol{x}}) = 0, \forall \bar{\boldsymbol{x}} \in \mathbb{R}^{n+1},$$

which clearly causes a contradiction.

A.3 Extending Definition 3.3 to Consider Data Distributions

Recall that Definition 3.2 is a modification of Definition 3.1 by considering data distributions instead of data sets. We can apply the same modification to Definition 3.3. Because the change is straightforward, we omit details. Instead, we directly discuss the biased stochastic gradient under the modified definition in the following corollary, which extends Theorem 3.4 to the data-distribution setting.

Corollary A.1. *Given that number of sampled classes $k \geq 1$ and schemes for instance and class sampling without replacement, the following statement holds.*

$$
\begin{aligned}
&\forall s : \mathbb{R}^{k+1} \to \mathbb{R}, \exists \textit{ a data distribution},\\
&\quad \textit{let } \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{l} \textit{ be a dataset of } l \textit{ i.i.d. samples}\\
&\quad\quad \textit{drawn from the data distribution with } n+1 \textit{ classes, and } n > k,\\
&\exists \hat{y}(\boldsymbol{\theta}; \boldsymbol{x}) \in \mathbb{R}^{n+1},\\
&\exists \bar{B} \leq l, \textit{ let } B \textit{ with } |B| = \bar{B} \textit{ be a subset randomly drawn from } \{1, \ldots, l\},\\
&\quad \textit{let } J^i \textit{ be a subset randomly drawn from } \{1, \ldots, n+1\} \backslash \{y_i\},\\
&\exists \bar{\boldsymbol{\theta}}, \qquad \mathbb{E}_{\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{l}} \mathbb{E}_B \mathbb{E}_{J^i, i \in B} \left[\frac{1}{|B|} \sum_{i \in B} \textit{Eq. (14) with } \boldsymbol{x}_i, y_i, J^i \Big|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}} \right]\\
&\qquad \neq \mathbb{E}_{\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{l}} \left[\left(\frac{1}{l} \sum_{i=1}^{l} \frac{\partial}{\partial \boldsymbol{\theta}} \operatorname{loss}(\boldsymbol{\theta}; \boldsymbol{x}_i, y_i) \right) \Big|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}} \right].
\end{aligned}
\tag{39}
$$

Proof. Assume $\{(\bar{\boldsymbol{x}}_i, \bar{y}_i)\}_{i=1}^{l}$ is a training set used to prove the statement in Theorem 3.4. Let us consider a discrete uniform distribution with these l points as possible outcomes. We argue that

$$
\begin{aligned}
&\mathbb{E}_{\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{l}} \mathbb{E}_B \mathbb{E}_{J^i, i \in B} \left[\frac{1}{|B|} \sum_{i : i \in B} \text{Eq.}(14) \text{ with } \boldsymbol{x}_i, y_i, J^i \right]\\
&= \mathbb{E}_B \mathbb{E}_{J^i, i \in B} \left[\frac{1}{|B|} \sum_{i : i \in B} \text{Eq.}(14) \text{ with } \bar{\boldsymbol{x}}_i, \bar{y}_i, J^i \right],
\end{aligned}
\tag{40}
$$

and

$$
\mathbb{E}_{\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{l}} \left[\frac{1}{l} \sum_{i=1}^{l} \frac{\partial}{\partial \boldsymbol{\theta}} \operatorname{loss}(\boldsymbol{\theta}; \boldsymbol{x}_i, y_i) \Big|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}} \right] = \frac{1}{l} \sum_{i=1}^{l} \frac{\partial}{\partial \boldsymbol{\theta}} \operatorname{loss}(\boldsymbol{\theta}; \bar{\boldsymbol{x}}_i, \bar{y}_i) \Big|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}}. \tag{41}
$$

Let us explain (41) in detail and (40) follows by the same derivation. We can see that $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{l}$ are i.i.d. random variables under the discrete uniform distribution mentioned above. Thus, we can simplify the left-hand side of (41) by using one random variable $(\boldsymbol{x}, y)$:

$$
\mathbb{E}_{(\boldsymbol{x}, y)} \left[\frac{1}{l} \sum_{i=1}^{l} \frac{\partial}{\partial \boldsymbol{\theta}} \operatorname{loss}(\boldsymbol{\theta}; \boldsymbol{x}, y) \Big|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}} \right],
$$

which is the same as the right-hand side because $(\boldsymbol{x}, y)$ is uniformly distributed over the l given points $\{(\bar{\boldsymbol{x}}_i, \bar{y}_i)\}_{i=1}^{l}$.

With (40) and (41), we can apply Theorem 3.4 to prove the inequality in (39). □

A.4 Unbiasedness under Uniform Minibatch Sampling

As an illustration, we show that under uniform minibatch sampling without replacement in Definition 3.1, stochastic gradient is unbiased.

For a fixed training set, since B is sampled uniformly from all subsets of size $\bar{B}$, each training instance in the training set is included in B with probability

$$
\Pr(i \in B) \equiv \frac{\bar{B}}{l}.
$$

Therefore, for a fixed parameter value $\bar{\boldsymbol{\theta}}$,

$$
\mathbb{E}_B \left[\frac{1}{\bar{B}} \sum_{i \in B} \frac{\partial}{\partial \boldsymbol{\theta}} \operatorname{loss}(\boldsymbol{\theta}; \boldsymbol{x}_i, \boldsymbol{y}_i) \Big|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}} \right] = \frac{1}{\bar{B}} \sum_{i=1}^{l} \mathbb{E}_B \left[\mathbb{I}\left[i \in B \right] \right] \frac{\partial}{\partial \boldsymbol{\theta}} \operatorname{loss}(\boldsymbol{\theta}; \boldsymbol{x}_i, \boldsymbol{y}_i) \Big|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}},
$$

where the indicator function $\mathbb{I}[i \in B]$ equals 1 if $i \in B$ and 0 otherwise. Therefore, its expectation is the probability that i is included in B:

$$\mathbb{E}_B[\mathbb{I}[i \in B]] = \Pr(i \in B).$$

Then, it follows that

$$\begin{aligned}
\mathbb{E}_B\left[\frac{1}{\bar{B}}\sum_{i\in B}\frac{\partial}{\partial\boldsymbol{\theta}}\operatorname{loss}(\boldsymbol{\theta};\boldsymbol{x}_i,\boldsymbol{y}_i)\Big|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}}\right] &= \frac{1}{\bar{B}}\sum_{i=1}^{l}\Pr(i\in B)\ \frac{\partial}{\partial\boldsymbol{\theta}}\operatorname{loss}(\boldsymbol{\theta};\boldsymbol{x}_i,\boldsymbol{y}_i)\Big|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}} \\
&= \frac{1}{\bar{B}}\sum_{i=1}^{l}\frac{\bar{B}}{l}\ \frac{\partial}{\partial\boldsymbol{\theta}}\operatorname{loss}(\boldsymbol{\theta};\boldsymbol{x}_i,\boldsymbol{y}_i)\Big|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}} \\
&= \frac{1}{l}\sum_{i=1}^{l}\frac{\partial}{\partial\boldsymbol{\theta}}\operatorname{loss}(\boldsymbol{\theta};\boldsymbol{x}_i,\boldsymbol{y}_i)\Big|_{\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}}.
\end{aligned}$$

Hence, the minibatch gradient is an unbiased estimator of the full gradient.