Conformal Risk Minimization with Variance Reduction

Sima Noorani [1] **Orlando Romero** [1] **Nicolo Dal Fabbro** [1] **Hamed Hassani** [1] **George Pappas** [1]

Abstract

Conformal prediction (CP) is a distribution-free framework for achieving probabilistic guarantees on black-box models. CP is generally applied to a model post-training. Recent research efforts, on the other hand, have focused on optimizing CP efficiency *during training*. We formalize this concept as the problem of *conformal risk minimization* (CRM). In this direction, *conformal training* (`ConfTr`) by Stutz et al. (2022) is a CRM technique that seeks to minimize the expected prediction set size of a model by simulating CP in-between training updates. In this paper, we provide a novel analysis for the `ConfTr` gradient estimation method, revealing a strong source of sample inefficiency that introduces training instability and limits its practical use. To address this challenge, we propose *variance-reduced conformal training* (`VR-ConfTr`), a CRM method that carefully incorporates a novel variance reduction technique in the gradient estimation of the `ConfTr` objective function. Through extensive experiments on various benchmark datasets, we demonstrate that `VR-ConfTr` consistently achieves faster convergence and smaller prediction sets compared to baselines.

1. Introduction

Consider a classification task with input (features) $X \in \mathcal{X}$ and corresponding labels $Y \in \mathcal{Y} = \{1, \dots, K\}$. Let $\pi_\theta : \mathcal{X} \to \mathbb{R}^K$ be a parameterized predictor which, for every input x and label y, approximates the posterior probability $\pi(y|x) = \mathbb{P}(Y = y \,|\, X = x)$. Using π_θ, we can estimate the label corresponding to an input x as $\delta_\theta(x) = \arg\max_{y \in \mathcal{Y}} \pi_\theta(y|x)$. Usually, the performance of the predictor π_θ is assessed via the *accuracy*, which is the portion of testing samples whose predicted label matches the true label. While the accuracy is a key performance metric, in safety-critical applications it is crucial not only to predict accurately but also to quantify the uncertainty associated with a prediction. To address this, conformal prediction (CP) (Vovk et al., 2005; Shafer & Vovk, 2008; Angelopoulos et al., 2023), uses the (pre-trained) model π_θ to construct, for an input X, a prediction set $C(X) \subseteq \mathcal{Y}$ that contains the true label with high probability, satisfying the desired *coverage* guarantee. For example, the set $C(X)$ satisfies *marginal coverage* with miscoverage rate $\alpha \in (0, 1)$ if $\mathbb{P}(Y \in C(X)) \geq 1 - \alpha$.

One way to evaluate the usefulness of prediction sets is the *length efficiency* (Fontana et al., 2023), which represents a measure of their size. For instance, while it is possible to trivially guarantee any desired coverage by including the entire label space in $C(x)$, a prediction set constructed in this way is non-informative and useless. Thus, an efficient $C(x)$ is as small as possible while maintaining the desired coverage guarantee. Addressing the efficiency challenge by refining the CP set construction technique applied post-training, though effective, is *inherently constrained* by the performance of the pre-trained model π_θ. On the other hand, by integrating CP into the training process, recent research efforts (Dheur & Taieb, 2024; Cherian et al., 2024; Einbinder et al., 2022; Stutz et al., 2022; Bellotti, 2021) guide the training of a model π_θ via CP-induced metrics, optimizing the model parameters θ to improve its *inherent CP efficiency*. Here, we formalize this emerging optimization setting as the problem of *conformal risk minimization* (CRM). Among the existing CRM methods, the *conformal training* (`ConfTr`) approach first introduced by Stutz et al. (2022) is an intuitive technique - based on simulating CP during training to construct differentiable approximations of prediction sets - which is recently gaining momentum (Cherian et al., 2024; Yan et al., 2024; Wang et al., 2025). However, despite its potential, `ConfTr` suffers from training instability and struggles to converge (Stutz et al., 2022; Liu et al., 2024; Correia et al., 2024). In this paper, we focus on the following questions:

What is the source of training instability in `ConfTr`? How can we address this limitation?

We provide answers to both questions. First, we theoretically show that `ConfTr` is intrinsically *sample-inefficient*. Second, to address this limitation, we introduce *variance-*

[1]University of Pennsylvania. Correspondence to: <noorani@seas.upenn.edu>.

Published at ICML 2025 Workshop on Reliable and Responsible Foundation Models.

reduced conformal training VR-ConfTr, a provably sample efficient CRM algorithm that radically improves gradient estimation and enhances training stability.

1.1. Contributions

Our contributions can be summarized as follows:

Analysis of the ConfTr algorithm. Focusing on CRM for length efficiency optimization in the classification setting, we provide a novel analysis for the ConfTr (Stutz et al., 2022) method, which reveals a strong source of sample inefficiency in its gradient estimation technique. In particular, we show that the ConfTr gradient variance *is not reduced with the batch size*, and we show that this is related to the need for improved estimators of the quantile gradients.

A "plug-in" algorithm. We introduce the pipeline of *variance-reduced conformal training* (VR-ConfTr), our proposed algorithm to overcome this challenge, which (i) decouples the estimation of the population quantile and of its gradient, and (ii) leverages a "plug-in" step to incorporate improved estimates of quantiles' gradients in the training.

Novel variance reduction technique. Building on a fundamental result, which characterizes the gradient of the population quantile as a conditional expectation, we propose a novel estimator for quantiles' gradients, which can be seamlessly integrated into VR-ConfTr. We show that, under reasonable assumptions, this integration makes VR-ConfTr provably *sample-efficient*: unlike ConfTr, our approach effectively reduces the variance of the resulting estimated gradients with the training batch size.

Empirical validations. We extensively validate our method on various benchmark and real-world datasets, including MNIST, FMNIST, KMNIST, OrganAMNIST, and CIFAR10. Our results demonstrate that VR-ConfTr consistently and significantly improves the efficiency and stability of CRM for length efficiency optimization.

Broad applicability. Our approach and variance reduction technique can be integrated into any CRM method that requires quantile gradient estimation, at essentially no additional computational cost, extending its utility to a large class of CP frameworks and learning models.

1.2. Related Works

Conformal prediction (CP) is a distribution-free, principled framework that provides formal probabilistic guarantees for black-box models (Vovk et al., 2005; Shafer & Vovk, 2008; Angelopoulos et al., 2023), with exemplar applications in computer vision (Angelopoulos et al., 2020), large language models (Mohri & Hashimoto, 2024; Kumar et al., 2023) and path planning (Lindemann et al., 2023). To study the expected size of the prediction sets, (Lei, 2017; Lei & Wasserman, 2014; Vovk et al., 2016) provide asymptotic analysis in the context of statistical optimality, and (Dhillon et al., 2024) provide a finite-sample analysis under the split conformal prediction framework. To improve CP efficiency, many research efforts have focused on approaches that apply CP post-training to black-box models. In particular, recent algorithmic developments address improving length efficiency through better **conformity score** design (Romano et al., 2020; Yang & Kuchibhotla, 2024; Amoukou & Brunel, 2023; Deutschmann et al., 2024; Luo & Zhou, 2024), or on designing better **calibration procedures** (Kiyani et al., 2024; Bai et al., 2022; Yang & Kuchibhotla, 2021; Colombo & Vovk, 2020). These efforts do not fall under the CRM framework because they focus on learning low-dimensional hyper-parameters for pre-trained models as opposed to fully guiding the training of a θ-parameterized model $\pi_\theta(y|x)$.

Conformal risk minimization. There is a growing body of work (Einbinder et al., 2022; Cherian et al., 2024; Stutz et al., 2022; Bellotti, 2021; Yan et al., 2024) integrating ideas from conformal prediction in order to directly train a model for improved CP. Among these, ConfTr proposed by Stutz et al. (2022) has gained significant attention. This approach addresses length efficiency optimization by defining a loss function obtained by simulating conformal prediction during training. We will extensively describe and provide a novel analysis for this approach in the next section. Earlier work by Bellotti (2021) considered an approach analogous to ConfTr in that the authors simulate conformal prediction during training. However, the algorithm provided by Bellotti (2021) treats the quantile-threshold as fixed and not as a function of the model parameters. It has been extensively shown by Stutz et al. (2022) that the approach by Bellotti (2021) provides inferior performance with respect to ConfTr. Moreover, Yan et al. (2024) use a similar training pipeline to Stutz et al. (2022) in order to minimize the inefficiency of their proposed conformal predictor. Cherian et al. (2024) train a score function, rather than a point predictor, subject to conditional coverage constraints (Gibbs & Candes, 2021). Einbinder et al. (2022) utilizes conformal prediction insights in order to mitigate overconfidence in multi-class classifiers by minimizing a carefully designed loss function. Among these approaches, ConfTr is the method that has been applied the most. For example, Zhao et al. (2025) investigate the use of the ConfTr loss function in the context of neural network pruning. Additionally, Wang et al. (2025) have recently attempted to apply ConfTr to train Graph Neural Networks (GNNs).

2. Problem Formulation

Let us consider a (parameterized) model of logits $f_\theta : \mathcal{X} \to \mathbb{R}^K$ and let $\pi_\theta(x) = \mathrm{softmax}(f_\theta(x))$ denote the corresponding predicted probabilities. A central objective in

conformal prediction is to use a given black box model f_θ to construct a *set* predictor $C_\theta : \mathcal{X} \to 2^{\mathcal{Y}}$ in such a way that C_θ satisfies some form of probabilistic *coverage* guarantee. In particular, $C_\theta(X)$ satisfies *marginal* coverage if

$$\mathbb{P}\left(Y \in C_\theta(X)\right) \geq 1 - \alpha, \tag{1}$$

for a user-specified miscoverage rate $\alpha \in (0, 1)$.

One common approach to achieve marginal coverage is via a *thresholding* (THR) set predictor (Vovk et al., 2005), $C_\theta(x; \tau) = \{y \in \mathcal{Y} : E_\theta(x, y) \geq \tau\}$ for some well-chosen threshold $\tau \in \mathbb{R}$ and *conformity score* $E_\theta(x, y)$, which can be any heuristic notion of uncertainty regarding label y upon input x for the predictor $f_\theta(\cdot)$. Some choices for the conformity score include *(i)* the predicted probabilities $E_\theta(x, y) = \pi_\theta(y|x) = [\pi_\theta(x)]_y$, *(ii)* the logits $E_\theta(x, y) = [f_\theta(x)]_y$, and *(iii)* the predicted log-probabilities $E_\theta(x, y) = \log \pi_\theta(y|x)$.

If the distribution of $Z = (X, Y)$ were known, we could achieve marginal coverage by setting $\tau = \tau(\theta)$ as the α-quantile of the scalar random variable $E_\theta(Z)$, i.e. $\tau(\theta) = \inf\{t \in \mathbb{R} : \mathbb{P}\left(E_\theta(Z) \leq t\right) \geq \alpha\}$. However since Z is generally unknown in practice, we estimate $\tau(\theta)$ from samples $Z_1, \ldots, Z_n$ and use the empirical quantile $\hat{\tau}_n(\theta)$. If the data are *exchangeable*, i.e., their joint distribution is invariant to permutations, then choosing $\hat{\tau}_n(\theta)$ as the empirical α-quantile ensures that $C_\theta(x) := C_\theta(x, \hat{\tau}_n(\theta))$ satisfies the marginal coverage guarantee (1). Specifically,

$$\hat{\tau}_n(\theta) = E_{(\lceil \alpha n \rceil)}(\theta), \tag{2}$$

where $E_{(1)}(\theta) \leq \ldots \leq E_{(n)}(\theta)$ denote the order statistics for $E_\theta(Z_1), \ldots, E_\theta(Z_n)$.

2.1. Conformal Risk Minimization

As we outlined in the introduction, recent research efforts have attempted to combine training and CP into one, as opposed to using CP only as a post-training method. Here, we formalize this by borrowing terminology from statistical supervised learning and introducing the problem of *conformal risk minimization* (CRM). CRM can be understood as a framework for training a parameterized predictor that learns according to some CP efficiency metric. CRM can be formulated as follows:

$$\min_{\theta \in \Theta} \left\{ L(\theta) := \mathbb{E}\left[\ell(C_\theta(X), Y)\right] \right\} \tag{CRM}$$

for some *conformal loss* ℓ, where $C_\theta(x)$ is a *conformalized* predictor. This problem is closely related to the *conformal risk control* explored by Angelopoulos et al. (2022). More concretely, we will consider the threshold-based set predictor $C_\theta(x) := C_\theta(x; \tau(\theta)) = \{y \in \mathcal{Y} : E_\theta(x, y) \geq \tau(\theta)\}$.

One difficult aspect in solving (CRM) is its non-differentiability. We can remedy this by introducing a smoothed approximation of the problem. To do this, we can adopt measures similar to Stutz et al. (2022). More precisely, we can first rewrite $\ell(C, y)$ as $\tilde{\ell}(\mathbf{C}, y)$, where $\mathbf{C} \in \{0, 1\}^K$ is a vector of binary decision variables with $[\mathbf{C}]_k = 1_{k \in C}$. Then, assuming that $\ell(\mathbf{C}, y)$ is well defined for every $\mathbf{C} \in [0, 1]^K$, we can consider a smooth approximation of (CRM) by replacing $\ell(C_\theta(x; \tau(\theta)), y)$ with $\tilde{\ell}(\mathbf{C}_\theta(x; \tau(\theta)), y)$, where $[\mathbf{C}_\theta(x; \tau)]_k = 1_{E_\theta(x,y) - \tau \geq 0}$ is replaced with $[\mathbf{C}_\theta(x; \tau)]_k = \text{sigmoid}\left(\frac{E_\theta(x,y) - \tau}{T}\right)$ for some temperature parameter $T > 0$.

With this, we will focus on the following smoothed version of problem (CRM):

$$\min_{\theta \in \Theta} \left\{ L(\theta) := h(\mathbb{E}\left[\ell(\theta, \tau(\theta), X, Y)\right]) + R(\theta) \right\}, \tag{ConfTr-risk}$$

for some monotone function $h(\cdot)$, loss $\ell(\theta, \tau, x, y)$, and regularizer $R(\theta)$. Recall also that $\tau(\theta) = \inf\{t \in \mathbb{R} : \mathbb{P}\left(E_\theta(X, Y) \leq t\right) \geq \alpha\}$ for a given conformity score function $E_\theta(x, y)$.

Unlike the case of risk minimization problems, which can be understood as traditional stochastic optimization problems of the form $\theta \mapsto \mathbb{E}\left[\ell(\theta, X, Y)\right]$, the problem (ConfTr-risk) does not lend itself to a trivially unbiased estimator of its gradient with variance decaying as $\mathcal{O}(1/n)$ when given n i.i.d. samples from (X, Y). The reason, aside from the monotone transform $h(\cdot)$, lies in the presence of $\tau(\theta)$. Unlike the traditional stochastic optimization problem, $\frac{\partial}{\partial \theta}\ell(\theta, \tau(\theta), X, Y)$ cannot be evaluated from a single realization of (X, Y) since $\tau(\theta)$ and $\frac{\partial \tau}{\partial \theta}(\theta)$ are unknown due to the underlying distribution of (X, Y) being unknown. However, these quantities can be estimated from data.

The quality of any such estimator directly affects the estimation $\widehat{\frac{\partial L}{\partial \theta}}(\theta)$ of $\frac{\partial L}{\partial \theta}(\theta)$ and, consequently, the performance of any gradient-based optimization algorithm used to approximately solve (ConfTr-risk). Motivated by this, we aim to answer the following question: *can we design a gradient estimator for $L(\theta)$ that achieves arbitrarily small bias and (co)variance with sufficient samples?*

3. Analysis of `ConfTr` (Stutz et al., 2022)

Stutz et al. (2022) introduced *conformal training* (`ConfTr`), which we categorize as a CRM approach for length efficiency optimization. In particular, `ConfTr` focuses on reducing *inefficiency* of calibrated classifiers, quantified by the *target size* of predicted sets. This can be understood as the problem in (CRM) with $\ell(C, y) = \max(0, |C| - \kappa)$ for some *target size* κ (intended to discourage no predictions at all) and with a log transform h for numerical stability reasons. In this regard, it is worth noting that the earlier work of Sadinle et al. (2019) was the first to study the closely related problem of *least ambiguous* set-valued classifiers,

which corresponds to $l(C, y) = |C|$.

The underlying assumption, just as in any supervised learning task, is that the marginal distribution of (X, Y) is unknown but that instead we can collect some i.i.d. training data $\mathcal{D} = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$. With this, an issue presents itself in that, unlike a typical loss function, we cannot evaluate $\frac{\partial}{\partial\theta}[\ell(\theta, \tau(\theta), X_i, Y_i)]$ from realizations of X_i, Y_i alone, because $\tau(\theta) = \text{quantile}_\alpha(E_\theta(X, Y))$ and $\frac{\partial\tau}{\partial\theta}(\theta)$ are functions of the *distribution* of (X, Y) and not a mere transformation. To resolve this issue, Stutz et al. (2022) propose their `ConfTr` algorithm, which randomly splits a given batch B into two parts, which they refer to as *calibration* batch B_{cal} and *prediction* batch B_{pred}. With this, the authors advocate for employing any smooth (differentiable) quantile estimator algorithm for $\tau(\theta)$ using the calibration batch. Then, they propose using this estimator to compute a sampled approximation of (ConfTr-risk), replacing expectations by sample means constructed using the prediction batch. Let $\hat{L}(\theta)$ denote the end-to-end empirical approximation of $L(\theta)$. Once $\hat{L}(\theta)$ is constructed, the authors advocate for a (naive) risk minimization procedure where $\frac{\partial\hat{L}}{\partial\theta}(\theta)$ is computed and passed to an optimizer of choice.

3.1. Non-Diminishing Variance of `ConfTr`

In this subsection, we will show that the approach used in (Stutz et al., 2022) leads to an asymptotically unbiased gradient estimator, but its variance does not vanish.

We will use $E_\theta(x, y)$ and $E(\theta, x, y)$ interchangeably and often write (x, y) as z. To distinguish between partial and total differentiation, $\frac{\partial}{\partial\theta}\ell(\theta, \tau(\theta), x, y)$ denotes the Jacobian of $\theta \mapsto \ell(\theta, \tau(\theta), x, y)$ evaluated at θ, whereas $\frac{\partial\ell}{\partial\theta}(\theta, \tau(\theta), x, y)$ denotes the Jacobian of $\ell(\cdot, \tau(\theta), x, y)$ evaluated at θ. In particular, we have $\frac{\partial\ell}{\partial\theta}(\theta, \tau(\theta), x, y) = \frac{\partial}{\partial\theta'}\ell(\theta', \tau(\theta), x, y)\big|_{\theta'=\theta}$. Recall that $Z_i = (X_i, Y_i)$ are i.i.d. copies of $Z = (X, Y)$ and $E_{(1)}(\theta) \leq \ldots E_{(n)}(\theta)$ denote the order statistics for $E_\theta(Z_1), \ldots, E_\theta(Z_n)$. We make the following regularity assumptions:

Assumption 1. $E(\theta, x, y)$ is continuously differentiable and M-Lipschitz in θ.

Assumption 2. $E_\theta(Z)$ and $\frac{\partial E}{\partial\theta}(\theta, Z)$ each admit a continuous probability density function.

Stutz et al. (2022) consider *smooth* quantile estimators based on smooth sorting. However, in the next proposition we argue that the empirical quantile does not need to be smoothed.

Proposition 3.1. *$E_{(1)}(\theta), \ldots, E_{(n)}(\theta)$ are almost surely (a.s.) everywhere differentiable in θ. In particular, the empirical quantile $\hat{\tau}_n(\theta) = E_{(\lceil\alpha n\rceil)}(\theta)$ is a.s. everywhere differentiable.*

The proof of this result relies on noting that while the sort function is not differentiable everywhere, it is nonetheless differentiable almost everywhere. More precisely, the sort function is piecewise linear, with the non-differentiable points corresponding to ties in variables that are to be sorted. Due to the assumed continuous distribution of the scores $E_\theta(Z_1), \ldots, E_\theta(Z_n)$, it follows that there are almost surely (a.s.) no ties, and therefore the order statistics are indeed differentiable (a.s.). Subsequently, the empirical quantile $\hat{\tau}_n(\theta)$ is a.s. differentiable (everywhere in θ). Further, the smooth quantile $\hat{\tau}_n(\theta; \varepsilon)$ based on the smooth sorting method from Blondel et al. (2020) (which Stutz et al. (2022) re-implement in their own codebase) actually satisfies $\hat{\tau}_n(\theta; \varepsilon) = \hat{\tau}_n(\theta) = E_{(\lceil\alpha n\rceil)}(\theta)$ as long as $\varepsilon > 0$ is small enough, to the order of $\min_{i\neq j} |E_\theta(Z_i) - E_\theta(Z_j)|$, as can be seen from Lemma 3 in (Blondel et al., 2020).

We can now analyze the quality of the estimate $\widehat{\frac{\partial L}{\partial\theta}}(\theta)$ of $\frac{\partial L}{\partial\theta}(\theta)$ using `ConfTr`, assuming for simplicity that the "smooth quantile" exactly coincides with the empirical sample quantile. More precisely, Stutz et al. (2022) propose to use a batch of $2n$ i.i.d. samples, which get split in two parts of equal size n: one part for "calibration" and the other for "prediction." The calibration samples, as the name suggest, are used to estimate $\tau(\theta)$ as $\hat{\tau}_n(\theta)$ and nothing else, whereas the prediction samples are used to finish the estimation of $\frac{\partial L}{\partial\theta}(\theta)$ by merely replacing $\tau(\theta)$ with $\hat{\tau}_n(\theta)$ and replacing the expectations with sample means. This is formalized in Algorithm 1 with $\hat{\tau}_n(\theta) := E_{(\lceil\alpha n\rceil)}(\theta)$ and $\widehat{\frac{\partial\tau}{\partial\theta}}(\theta) := \frac{\partial\hat{\tau}_n}{\partial\theta}(\theta) = \frac{\partial}{\partial\theta}E_{(\lceil\alpha n\rceil)}(\theta)$.

To proceed with our analysis, let us first characterize the asymptotic behavior of $\frac{\partial\hat{\tau}_n}{\partial\theta}(\theta)$.

Proposition 3.2. *Let $\hat{\tau}_n(\theta) = E_{(\lceil\alpha n\rceil)}(\theta)$. Then,*

$$\frac{\partial\hat{\tau}_n}{\partial\theta}(\theta) \xrightarrow{\text{dist}} \frac{\partial E}{\partial\theta}(\theta, Z)\Big|_{E_\theta(Z)=\tau(\theta)} \tag{3}$$

as $n \to \infty$.

Since, $Z = (X, Y)$ will typically be a high-dimensional random vector (due to X), $\frac{\partial E}{\partial\theta}(\theta, Z)\Big|_{E_\theta(Z)=\tau(\theta)}$ will not be a constant. In particular, $\frac{\partial\hat{\tau}_n}{\partial\theta}(\theta)$ will fail to be a consistent estimator of $\frac{\partial\tau}{\partial\theta}(\theta)$. We can better characterize the behavior of $\frac{\partial\hat{\tau}_n}{\partial\theta}(\theta)$ by first noting the following helpful result.

Proposition 3.3 (Hong, 2009, Theorem 2)**.** *Suppose that X is absolutely continuous and $E_\theta(x, y)$ is continuously differentiable in θ and x. Then, for every $\theta \in \Theta$,*

$$\frac{\partial\tau}{\partial\theta}(\theta) = \mathbb{E}\left[\frac{\partial E}{\partial\theta}(\theta, X, Y)\,\middle|\, E_\theta(X, Y) = \tau(\theta)\right]. \tag{4}$$

Using this proposition and leveraging Assumption 1, we can establish that $\frac{\partial\hat{\tau}_n}{\partial\theta}(\theta)$ is asymptotically unbiased but its covariance matrix does not vanish as $n \to \infty$. The formal proof of this can be found in Corollary A.4 in the appendix.

From Proposition 3.1, we have that, almost surely, $\theta \mapsto \ell(\theta, \hat{\tau}_n(\theta), Z)$ is differentiable. Therefore, it follows from the chain rule that

$$\frac{\partial}{\partial\theta}[\ell(\theta,\hat{\tau}_n(\theta),Z)] = \frac{\partial\ell}{\partial\theta}(\theta,\hat{\tau}_n(\theta),x,y) + \frac{\partial\ell}{\partial\tau}(\theta,\hat{\tau}_n(\theta),Z)\frac{\partial\hat{\tau}_n}{\partial\theta}(\theta) \tag{5}$$

holds almost surely. By inspection, we can see that the covariance of $\frac{\partial\hat{\tau}_n}{\partial\theta}(\theta)$ will bottleneck the covariance of (5), and thus also the covariance of $\widehat{\frac{\partial L}{\partial\theta}}(\theta)$. We can formalize this in the following theorem.

Theorem 3.4. *Suppose that $\ell(\cdot,\cdot,,z)$ is continuously differentiable. For the* `ConfTr` *method, the estimator $\widehat{\frac{\partial L}{\partial\theta}}(\theta)$ converges weakly to a random vector that is not constant. If, in addition, $\ell(\cdot,\cdot,z)$ is M-Lipschitz and bounded by M, then $\widehat{\frac{\partial L}{\partial\theta}}(\theta)$ is asymptotically unbiased but its covariance matrix does not vanish as $n \to \infty$.*

The first part of the theorem follows from the continuous mapping theorem and by noting that $\hat{\tau}_n(\theta) \xrightarrow{\text{a.s.}} \tau(\theta)$ (Serfling, 1980). The second part follows by noting that all the terms in line 7 of Algorithm 1 are uniformly integrable.

The takeaways of the analysis in this section are that $\frac{\partial\hat{\tau}_n}{\partial\theta}(\theta)$ is a poor estimator of $\frac{\partial\tau}{\partial\theta}(\theta)$, *even though $\hat{\tau}_n(\theta)$ is an effective estimator of $\tau(\theta)$*. This in turn causes *the overall gradient's variance to be* $\Omega(1)$, because, as we show in Theorem 3.4, the bias and covariance of $\frac{\partial\hat{\tau}_n}{\partial\theta}(\theta)$ get inherited by $\widehat{\frac{\partial L}{\partial\theta}}(\theta)$. This motivates our proposed solution addressing the gradient estimation issue of `ConfTr`, which we present next.

4. Variance-Reduced Conformal Training

In order to surpass the shortcoming of `ConfTr` described in the previous section, let us first note that the gradient of the conformal risk (ConfTr-risk) can be written as

$$\frac{\partial L}{\partial\theta}(\theta) = h'\left(\mathbb{E}\big[\ell(\theta,\tau(\theta),Z)\big]\right) \times\left(\mathbb{E}\left[\frac{\partial\ell}{\partial\theta}(\theta,\tau(\theta),Z)\right]+\mathbb{E}\left[\frac{\partial\ell}{\partial\tau}(\theta,\tau(\theta),Z)\right]\frac{\partial\tau}{\partial\theta}(\theta)\right), \tag{6}$$

where h' denotes the derivative of h, $Z = (X, Y)$. Note that we dropped the regularizer for simplicity.

Discussion: Inspecting equation (6), we note that, in order to estimate the gradient $\frac{\partial L}{\partial\theta}(\theta)$, the estimator $\widehat{\frac{\partial\tau}{\partial\theta}}(\theta)$ of the population quantile gradient $\frac{\partial\tau}{\partial\theta}(\theta)$ *does not necessarily need to be the gradient of the sample quantile $\hat{\tau}(\theta)$*, which was the strategy adopted by the `ConfTr` method. As we show in section 3.1, this naive choice, which sets $\widehat{\frac{\partial\tau}{\partial\theta}}(\theta) = \frac{\partial\hat{\tau}}{\partial\theta}(\theta)$, results in a *highly sample-inefficient gradient estimator*. With this in mind, we will now leverage the structure of the population quantile gradient established in (4) to design a novel estimator for $\frac{\partial\tau}{\partial\theta}(\theta)$. Then, in section 4.2, we illustrate the pipeline of our proposed `VR-ConfTr` algorithm, showing how we can integrate the new estimator in computing the estimate $\widehat{\frac{\partial L}{\partial\theta}}$ of $\frac{\partial L}{\partial\theta}(\theta)$.

4.1. Quantile Gradient Estimation

We will now use the relationship established in Proposition 3.3 to design a novel estimator of the quantile gradient. The idea is as follows: let us denote

$$\eta_\varepsilon(\theta) := \mathbb{E}\left[\frac{\partial E}{\partial\theta}(\theta,X,Y)\,\middle|\,A_\varepsilon(\theta)\right], \quad \Sigma_\varepsilon(\theta) := \operatorname{cov}\left(\frac{\partial E}{\partial\theta}(\theta,X,Y)\,\middle|\,A_\varepsilon(\theta)\right), \tag{7}$$

for $\varepsilon > 0$, where $A_\varepsilon(\theta) := \{|E_\theta(X,Y) - \tau(\theta)| \le \varepsilon\}$. Note that the term $\eta_\varepsilon(\theta)$ in (7) is approximately equal to the population quantile gradient $\frac{\partial\tau}{\partial\theta}(\theta)$ if $\varepsilon \approx 0$. Based on this, assuming that we have a good estimate $\hat{\tau}(\theta)$ of the population quantile $\tau(\theta)$ available, we can estimate $\eta(\theta) = \frac{\partial\tau}{\partial\theta}(\theta)$, using $\eta_\varepsilon(\theta)$ via the following ε-threshold strategy:

$$\hat{\eta}(\theta) := \frac{1}{\sum_{i=1}^n 1_{\hat{A}_{\varepsilon,i}(\theta)}}\sum_{i=1}^n 1_{\hat{A}_{\varepsilon,i}(\theta)}\frac{\partial E}{\partial\theta}(\theta,X_i,Y_i), \tag{8}$$

using i.i.d. samples $(X_1,Y_1),\ldots,(X_n,Y_n)$, where $\hat{A}_{\varepsilon,i}(\theta) = \{|E_\theta(X_i,Y_i) - \hat{\tau}(\theta)| \le \varepsilon\}$. In other words, given a batch of n samples, the estimator $\hat{\eta}(\theta)$ in (8) is computed as the average of $m = \sum_{i=1}^n 1_{\hat{A}_{\varepsilon,i}(\theta)}$ conformity scores' gradients, corresponding to the samples whose conformity scores are ε-close to the estimated population quantile $\hat{\tau}(\theta)$.

Tuning ε with m-ranking. In practice, a good value of ε may depend on batch and parameter θ, and it could significantly change for each iteration of the optimization process. Furthermore, fine-tuning a time-varying value of a scalar quantity $\varepsilon > 0$ can be challenging, and we therefore need some other heuristic way to tune ε adaptively. One intuitive way to do this, that we adopt in our experiments, is to fix an integer m, sort the samples based on the distances $\{\big|E_\theta(X_i,Y_i) - \hat{\tau}(\theta)\big|\}_{i=1,\ldots,n}$, and then average the "top" m samples (smallest distances) to estimate the quantile gradient. This strategy is equivalent to selecting $\varepsilon = \inf\left\{\varepsilon' > 0 : \sum_{i=1}^n 1_{\hat{A}_{\varepsilon',i}(\theta)} \ge m\right\}$, and it is very practical since it only requires to fine-tune an integer m.

4.2. Proposed Algorithm: `VR-confTr`

Suppose that the variance-reduced estimator for $\frac{\partial\tau}{\partial\theta}(\theta)$ has been already designed. Then, the new estimate for $\tau(\theta)$ and

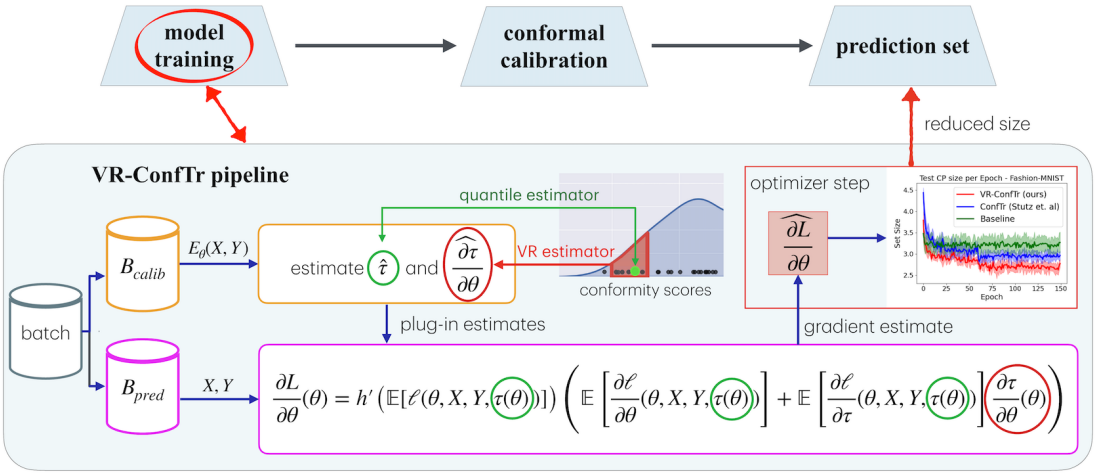

Figure 1. In this figure, we illustrate the `VR-ConfTr` pipeline and position it with respect to a typical CP procedure.

$\frac{\partial\tau}{\partial\theta}(\theta)$ *can be plugged into expression* (6) for the gradient of the conformal training risk function, before approximating the expectation by sample means, leading to the *plug-in* estimator for $\frac{\partial L}{\partial\theta}(\theta)$. Naturally, the plug-in gradient estimator is then passed through an optimizer in order to approximately solve (CRM). Our proposed pipeline, which we call *variance-reduced conformal training* (`VR-ConfTr`) algorithm, paired with our novel estimator in (8), constitutes our main contribution and proposed solution to improve the sample inefficiency of `ConfTr`. The critical step of constructing the plug-in estimator is summarized in Algorithm 1 and the entire pipeline is illustrated in Figure 1. Similar to Theorem 3.4, we can establish that the bounds on the bias and covariance of $\widehat{\frac{\partial\tau}{\partial\theta}}(\theta)$ get inherited by $\widehat{\frac{\partial L}{\partial\theta}}(\theta)$, assuming that $\hat{\tau}(\theta) \xrightarrow{\text{a.s.}} \tau(\theta)$.

4.3. Sample-Efficiency of `VR-ConfTr`

We now provide an analysis for the ε-estimator $\hat{\eta}(\theta)$ (8) of the population quantile gradient, establishing its bias-variance trade-off in the following theorem.

Theorem 4.1. *Fix θ and $\varepsilon > 0$. Let $\hat{\eta}(\theta)$ be the gradient estimator defined in* (8)*. Then, the bias and variance of the estimator can be characterized as follows:*

$$(i) \quad \mathbb{E}\left[\hat{\eta}(\theta)\right] = (1 - [q_\varepsilon(\theta)]^n)\eta_\varepsilon(\theta)$$

$$(ii) \quad \operatorname{cov}\left(\hat{\eta}(\theta)\right) \preceq \frac{2\Sigma_\varepsilon(\theta)}{p_\varepsilon(\theta)n} + [q_\varepsilon(\theta)]^n\, \eta_\varepsilon(\theta)\eta_\varepsilon^\mathsf{T}(\theta),$$

where $p_\varepsilon(\theta) = \mathbb{P}(A_{\varepsilon,}(\theta))$ and $q_\varepsilon(\theta) = 1 - p_\varepsilon(\theta)$.

The main takeaway of result (i) is that $\hat{\eta}(\theta)$ is an *asymptotically unbiased* estimator of $\eta_\varepsilon(\theta)$, but not $\eta(\theta)$. However, by definition we also have $\eta_\varepsilon(\theta) \approx \eta(\theta)$ for $\varepsilon \approx 0$. The second result (ii), instead, shows that *variance reduction* is obtained by the proposed estimator, when compared to the naive estimator $\frac{\partial\hat{\tau}}{\partial\theta}(\theta)$. For large n, the variance reduction is proportional to $p_\varepsilon(\theta)n$, which is equal to the (expected) proportion of samples that are used in the estimator. More precisely, the variance of the estimator is $\mathcal{O}\left(\frac{1}{p_\varepsilon(\theta)n}\right)$ as $\varepsilon \to 0$ or $n \to \infty$. A key takeaway of (i) and (ii) is the explicit characterization of the *bias-variance trade-off* as a function of the threshold $\varepsilon > 0$ and of the batch size n: for a given batch size n, a larger ε increases the expected amount of samples used by the estimator, reducing its variance. However, larger ε also increases the bias of the estimator

Algorithm 1 Gradient estimator of `VR-ConfTr`

Require:
batch $B = \{(X_1, Y_1), \ldots, (X_{2n}, Y_{2n})\}$ of i.i.d. samples from (X, Y),
score function $E(\theta, x, y) : \Theta \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$,
conformal loss $\ell(\theta, x, y, \tau) : \Theta \times \mathcal{X} \times \mathcal{Y} \times \mathbb{R} \to \mathbb{R}$,
monotone transformation $\mathcal{F} : \mathbb{R} \to \mathbb{R}$,
estimator $\hat{\tau}(\cdot)$ for $\tau(\theta) = Q_\alpha(E_\theta(X, Y))$,
estimator $\widehat{\frac{\partial\tau}{\partial\theta}}(\cdot)$ for $\frac{\partial\tau}{\partial\theta}(\theta)$.

Ensure: output an estimate $\widehat{\frac{\partial L}{\partial\theta}}$ of the gradient $\frac{\partial L}{\partial\theta}(\theta)$ of the conformal training risk (ConfTr-risk)

1: partition B into $\{B_{\text{cal}}, B_{\text{pred}}\}$, with $|B_{\text{cal}}| = |B_{\text{pred}}| = n$.
2: $\hat{\tau} \leftarrow \hat{\tau}(B_{\text{cal}})$ // estimate $\tau(\theta)$ using B_{cal}
3: $\widehat{\frac{\partial\tau}{\partial\theta}} \leftarrow \widehat{\frac{\partial\tau}{\partial\theta}}(B_{\text{cal}})$ // estimate $\frac{\partial\tau}{\partial\theta}(\theta)$ using B_{cal}
4: $\hat{\ell} \leftarrow \frac{1}{|B_{\text{pred}}|}\sum_{(x,y)\in B_{\text{pred}}} \ell(\theta, x, y, \hat{\tau})$
5: $\widehat{\frac{\partial\ell}{\partial\theta}} \leftarrow \frac{1}{|B_{\text{pred}}|}\sum_{(x,y)\in B_{\text{pred}}} \frac{\partial\ell}{\partial\theta}(\theta, x, y, \hat{\tau})$
6: $\widehat{\frac{\partial\ell}{\partial\tau}} \leftarrow \frac{1}{|B_{\text{pred}}|}\sum_{(x,y)\in B_{\text{pred}}} \frac{\partial\ell}{\partial\tau}(\theta, x, y, \hat{\tau})$
7: $\widehat{\frac{\partial L}{\partial\theta}} \leftarrow h'(\hat{\ell})\left(\widehat{\frac{\partial\ell}{\partial\theta}} + \widehat{\frac{\partial\ell}{\partial\tau}}\,\widehat{\frac{\partial\tau}{\partial\theta}}\right) + \frac{\partial R}{\partial\theta}(\theta)$ // "**plug-in**"

Dataset	Model	Algorithm	Accuracy (Avg ± Std)	Size (Avg ± Std) (%)
MNIST	Linear	Baseline	0.887 ± 0.004	4.122 ± 0.127 (+12%)
		ConfTr (Stutz et al., 2022)	0.842 ± 0.141	3.990 ± 0.730 (+8%)
		VR-ConfTr **(ours)**	0.886 ± 0.071	$\mathbf{3.688 \pm 0.350}$
Fashion-MNIST	MLP	Baseline	0.845 ± 0.002	3.218 ± 0.048 (+15%)
		ConfTr (Stutz et al., 2022)	0.799 ± 0.065	3.048 ± 0.201 (+9%)
		VR-ConfTr **(ours)**	0.839 ± 0.043	$\mathbf{2.795 \pm 0.154}$
Kuzushiji-MNIST	MLP	Baseline	0.872 ± 0.046	4.982 ± 0.530 (+6%)
		ConfTr (Stutz et al., 2022)	0.783 ± 0.125	4.762 ± 0.226 (+2%)
		VR-ConfTr **(ours)**	0.835 ± 0.098	$\mathbf{4.657 \pm 0.680}$
OrganA-MNIST	ResNet-18	Baseline	0.552 ± 0.017	4.823 ± 0.748 (+2%)
		ConfTr (Stutz et al., 2022)	0.526 ± 0.047	6.362 ± 0.857 (+33%)
		VR-ConfTr **(ours)**	0.547 ± 0.021	$\mathbf{4.776 \pm 1.178}$
CIFAR-10	ResNet-20	Baseline	0.743 ± 0.003	2.881 ± 0.038 (+12%)
		ConfTr (Stutz et al., 2022)	0.733 ± 0.051	2.806 ± 0.389 (+10%)
		VR-ConfTr **(ours)**	0.742 ± 0.023	$\mathbf{2.508 \pm 0.039}$

Table 1. Summary of evaluation results. For `VR-ConfTr`, we show in percentage the average set size (**Size (Avg ± Std) (%)**) improvement against `ConfTr` by (Stutz et al., 2022). The third column presents the average accuracy and its standard deviation (**Accuracy (Avg ± Std)**). The confidence level α used for conformal prediction is fixed at 0.01 for all datasets except CIFAR-10, where it is set to 0.1.

towards the unconditional expectation $\mathbb{E}\left[\frac{\partial E}{\partial \theta}(\theta, X, Y)\right]$.

5. Experiments

We evaluate `VR-ConfTr` against (i) a baseline model trained with cross-entropy loss (`Baseline`), and (ii) `ConfTr`. We perform experiments across benchmark datasets - MNIST (Deng, 2012), Fashion-MNIST (Xiao et al., 2017a), Kuzushiji-MNIST (Clanuwat et al., 2018), CIFAR10 (Krizhevsky, 2009), and a healthcare dataset comprising abdominal computed tomography scans, OrganAMNIST (Yang et al., 2021). One of the main performance metrics that we consider is the *length-efficiency* of the conformal prediction sets produced by applying a standard CP procedure to the trained model. Other relevant metrics are the convergence speed of the algorithm, as well as the final accuracy of the model. To tune the `VR-ConfTr` ε-estimator for the quantile gradient in (8), we employ the m-ranking method 4.1, and investigate multiple choices for m, detailed in Appendix C. We provide extensive details about the training settings, the adopted model architectures and hyper-parameters in Appendix D. Next, we summarize results from evaluating the trained model, reporting average *accuracy* and CP *length efficiency* over multiple runs. In Section 5.2 we compare `Vr-ConfTr` and `ConfTr` illustrating curves of relevant evaluation metrics, highlighting the improved training performance and variance reduction.

5.1. Summary of Evaluation Results

Table 1 presents the CP set size resulting from CP procedure applied post-training, and the accuracy of the trained model for each dataset.

The metrics in Table 1 are averaged over 5-10 training trials, with details on trial variations in Appendix D. For a fair comparison, we used the same model architecture for all methods (`ConfTr`, `VR-ConfTr`, `Baseline`) and identical hyper-parameters for `ConfTr` and `VR-ConfTr`. For post-training CP, we use the standard `THR` method with the corresponding α. Average set sizes are reported over 10 calibration-test splits. The main takeaway from Table 1 is that `VR-ConfTr` improves over all considered metrics compared to `ConfTr`. `VR-ConfTr` consistently achieves smaller prediction set sizes than `ConfTr` and `Baseline`. Important to note that our focus is not to optimize `ConfTr` but to demonstrate that `VR-ConfTr` improves performance and stability regardless of `ConfTr`'s performance or hyper-parameters. As noted by Stutz et al. (2022), `Baseline` sometimes achieves slightly higher accuracy than `ConfTr` and `VR-ConfTr`, though `VR-ConfTr` consistently outperforms `ConfTr`. However, the focus of conformal training is to improve CP efficiency by reducing prediction set sizes while maintaining comparable accuracy to non-CRM methods, not surpassing `Baseline` in accuracy.

5.2. On the Training Performance of `VR-ConfTr`

Here, we focus on the training performance of `VR-ConfTr`, with special attention to the speed in minimizing the confor-

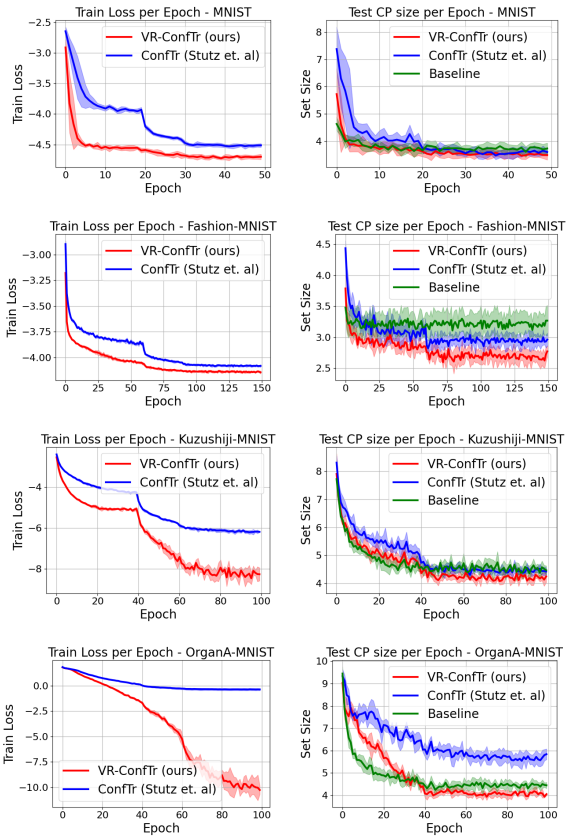

Figure 2. Learning curves for MNIST, Fashion-MNIST, Kuzushiji-MNIST, and OrganAMNIST. Each row shows training loss (left) and test CP set sizes (right) for the corresponding dataset, evaluated using the `THR` conformal predictor.

mal training loss described in section 2, and in minimizing the CP set sizes on test data. The results illustrate the evolution of the different metrics across epochs, validate the beneficial effect of the variance reduction technique and the superior performance of `VR-ConfTr` when compared to the competing `ConfTr` by Stutz et al. (2022).

In Figure 2, we show the training performance for four datasets (MNIST, FMNIST, KMNIST and OrganAMNIST) illustrating two key metrics: (i) the evolution of the conformal training loss defined in section 2 and (ii) the test CP size across epochs. In all the plots, we see that `VR-ConfTr` reaches smaller values of the loss and in significantly fewer epochs as compared to `ConfTr`. In the case of MNIST, for example, `VR-ConfTr` reaches a lower value of the loss in 10 times fewer epochs as compared to `ConfTr`. Similarly, for FMNIST `VR-ConfTr` achieves a smaller size in one third of epochs compared to `ConfTr`. For both Kuzushiji-MNIST and OrganA-MNIST, we notice that not only `VR-ConfTr` is faster, but it also gets to significantly smaller values of the loss. For the more challenging OrganA-MNIST dataset, this difference appears even more accentuated, not only in the training loss but also in the test CP set sizes. Notice that for all the three methods (`VR-ConfTr`, `ConfTr` and `Baseline`) we performed hyper-parameter tuning. Notably, in the case of the OrganA-MNIST dataset, we were not able to obtain an improvement with `ConfTr` in the final set size with respect to `Baseline`, which stresses the need for a method with improved gradient estimation, as the one we propose in this paper.

Fine-tuning Cifar-10: Figure 3 presents the training performance for CIFAR-10, where we fine-tune the last linear layer of a pretrained ResNet20 trained with cross-entropy loss. Unlike the other datasets, where we train all model parameters using the `ConfTr` or `VR-ConfTr` loss, we train CIFAR-10 fine-tuning the final layer only. Fine-tuning with `ConfTr` has gained traction due to its reduced computational overhead (Yan et al., 2024). Despite modifying only the final layer, `Vr-ConfTr` consistently yields smaller CP sets compared to `ConfTr`, while also achieving higher accuracy within the first epoch.

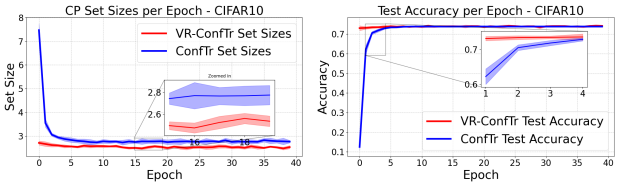

Figure 3. Learning curves for CIFAR-10 illustrating the fine-tuning process of a linear layer on a pretrained ResNet20 model using `ConfTr` and `VR-ConfTr`. Test CP set sizes are evaluated using the THR conformal predictor, consistent with the other datasets.

6. Concluding Remarks and Future Directions

We theoretically justified the sample inefficiency in the `ConfTr` method proposed by Stutz et al. (2022), which is a *conformal risk minimization* (CRM) method for length efficiency optimization. We have shown that the source of sample inefficiency lies in the estimation of the gradient of the population quantile. To address this issue, we introduced a novel technique that improves the gradient estimation of the population quantile of the conformity scores by provably reducing its variance. We show that, by incorporating this estimation technique in our proposed `VR-ConfTr` algorithm, the training becomes more stable and the post-training conformal predictor is often more efficient as well. Our work also opens up possibilities for future research in the area of CRM. Indeed, further methods for quantile gradient estimation could be developed and readily integrated with our "plug-in" algorithm, for which we can expect improved training performance.

Impact statement

In this paper, we propose `VR-ConfTr`, a novel algorithmic framework for Conformal Prediction that improves the efficiency and stability of training. This work has immediate applications in areas requiring reliable uncertainty quantification, such as healthcare, autonomous systems, and natural language processing. We do not anticipate any negative societal impact from this research.

References

Amoukou, S. I. and Brunel, N. J. Adaptive conformal prediction by reweighting nonconformity score, 2023.

Angelopoulos, A., Bates, S., Malik, J., and Jordan, M. I. Uncertainty sets for image classifiers using conformal prediction. *arXiv preprint arXiv:2009.14193*, 2020.

Angelopoulos, A. N., Bates, S., Fisch, A., Lei, L., and Schuster, T. Conformal risk control. *arXiv preprint arXiv:2208.02814*, 2022.

Angelopoulos, A. N., Bates, S., et al. Conformal prediction: A gentle introduction. *Foundations and Trends® in Machine Learning*, 16(4):494–591, 2023.

Bai, Y., Mei, S., Wang, H., Zhou, Y., and Xiong, C. Efficient and differentiable conformal prediction with general function classes, 2022.

Bellotti, A. Optimized conformal classification using gradient descent approximation. *arXiv preprint arXiv:2105.11255*, 2021.

Blondel, M., Teboul, O., Berthet, Q., and Djolonga, J. Fast differentiable sorting and ranking. In *International Conference on Machine Learning*, pp. 950–959. PMLR, 2020.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., and Wanderman-Milne, S. JAX: Autograd and XLA, 2018. URL http://github.com/google/jax.

Cherian, J. J., Gibbs, I., and Candès, E. J. Large language model validity via enhanced conformal prediction methods. *arXiv preprint arXiv:2406.09714*, 2024.

Clanuwat, T., Bober-Irizar, M., Kitamoto, A., Lamb, A., Yamamoto, K., and Ha, D. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*, 2018.

Colombo, N. and Vovk, V. Training conformal predictors, 2020.

Correia, A. H. C., Massoli, F. V., Louizos, C., and Behboodi, A. An information theoretic perspective on conformal prediction, 2024. URL https://arxiv.org/abs/2405.02140.

Defazio, A., Bach, F., and Lacoste-Julien, S. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/937964195d6fb3a55cd7cc578165f058-Paper.pdf.

Deng, L. The MNIST database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

Deutschmann, N., Rigotti, M., and Martinez, M. R. Adaptive conformal regression with split-jackknife+ scores. *Transactions on Machine Learning Research*, 2024.

Dheur, V. and Taieb, S. B. Probabilistic calibration by design for neural network regression. In *International Conference on Artificial Intelligence and Statistics*, pp. 3133–3141. PMLR, 2024.

Dhillon, G. S., Deligiannidis, G., and Rainforth, T. On the expected size of conformal prediction sets, 2024. URL https://arxiv.org/abs/2306.07254.

Einbinder, B.-S., Romano, Y., Sesia, M., and Zhou, Y. Training uncertainty-aware classifiers with conformalized deep learning, 2022.

Fontana, M., Zeni, G., and Vantini, S. Conformal prediction: a unified review of theory and new challenges. *Bernoulli*, 29(1):1–23, 2023.

Gibbs, I. and Candes, E. Adaptive conformal inference under distribution shift. *Advances in Neural Information Processing Systems*, 34:1660–1672, 2021.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Herman, E. and Strang, G. Calculus: Volume 3. *openstax*, 2018.

Hong, L. J. Estimating quantile sensitivities. *Operations research*, 57(1):118–130, 2009.

Johnson, R. and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. In Burges, C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. (eds.), *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper_files/paper/2013/file/

ac1dd209cbcc5e5d1c6e28598e8cbbe8-Paper.pdf.

Kiyani, S., Pappas, G., and Hassani, H. Length optimization in conformal prediction. *arXiv preprint arXiv:2406.18814*, 2024.

Krizhevsky, A. Learning multiple layers of features from tiny images. pp. 32–33, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Kumar, B., Lu, C., Gupta, G., Palepu, A., Bellamy, D., Raskar, R., and Beam, A. Conformal prediction with large language models for multi-choice question answering. *arXiv preprint arXiv:2305.18404*, 2023.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lei, J. Cross-validation with confidence, 2017. URL https://arxiv.org/abs/1703.07904.

Lei, J. and Wasserman, L. Distribution-free prediction bands for non-parametric regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76 (1):71–96, 2014. doi: 10.1111/rssb.12021.

Lindemann, L., Cleaveland, M., Shim, G., and Pappas, G. J. Safe planning in dynamic environments using conformal prediction. *IEEE Robotics and Automation Letters*, 2023.

Liu, K., Zeng, H., Huang, J., Zhuang, H., Vong, C.-M., and Wei, H. C-adapter: Adapting deep classifiers for efficient conformal prediction sets, 2024. URL https://arxiv.org/abs/2410.09408.

Luo, R. and Zhou, Z. Weighted aggregation of conformity scores for classification, 2024.

Mohri, C. and Hashimoto, T. Language models with conformal factuality guarantees. *arXiv preprint arXiv:2402.10978*, 2024.

Nair, V. and Hinton, G. E. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

Nguyen, L. M., Liu, J., Scheinberg, K., and Takáč, M. Sarah: A novel method for machine learning problems using stochastic recursive gradient, 2017. URL https://arxiv.org/abs/1703.00102.

Romano, Y., Sesia, M., and Candes, E. Classification with valid and adaptive coverage, 2020.

Sadinle, M., Lei, J., and Wasserman, L. Least ambiguous set-valued classifiers with bounded error levels. *Journal of the American Statistical Association*, 114(525):223–234, 2019.

Serfling, R. J. *Approximation Theorems of Mathematical Statistics*. John Wiley & Sons, New York, 1980. doi: 10.1002/9780470316481. URL http://dx.doi.org/10.1002/9780470316481.

Shafer, G. and Vovk, V. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9(3), 2008.

Shalev-Shwartz, S. and Zhang, T. Stochastic dual coordinate ascent methods for regularized loss minimization, 2013. URL https://arxiv.org/abs/1209.1873.

Stutz, D., Dvijotham, K. D., Cemgil, A. T., and Doucet, A. Learning optimal conformal classifiers. In *International Conference on Learning Representations*, 2022.

Sutskever, I., Martens, J., Dahl, G., and Hinton, G. Importance of initialization and momentum in deep learning. In *International conference on machine learning*, pp. 1139–1147. PMLR, 2013.

Vovk, V., Gammerman, A., and Shafer, G. *Algorithmic learning in a random world*, volume 29. Springer, 2005.

Vovk, V., Nouretdinov, I., Fedorova, V., Petej, I., and Gammerman, A. Criteria of efficiency for conformal prediction, 2016. URL https://arxiv.org/abs/1603.04416.

Wang, T., Zhou, Z., and Luo, R. Enhancing trustworthiness of graph neural networks with rank-based conformal training, 2025. URL https://arxiv.org/abs/2501.02767.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017a.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017b.

Yan, G., Romano, Y., and Weng, T.-W. Provably robust conformal prediction with improved efficiency. *arXiv preprint arXiv:2404.19651*, 2024.

Yang, J., Shi, R., Wei, D., Liu, Z., Zhao, L., Ke, B., Zhao, J., Fu, H., Xu, Y., and Heng, P.-A. Medmnist v2: A large-scale lightweight benchmark for 2d and 3d biomedical image classification. *arXiv preprint arXiv:2110.14795*, 2021.

Yang, Y. and Kuchibhotla, A. K. Finite-sample efficient conformal prediction. *arXiv preprint arXiv:2104.13871*, 2021.

Yang, Y. and Kuchibhotla, A. K. Selection and aggregation of conformal prediction sets, 2024.

Zhao, X., Farjudian, A., and Bellotti, A. Pruning convolutional neural networks for inductive conformal prediction. *Neurocomputing*, 611:128704, 2025. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2024.128704. URL https://www.sciencedirect.com/science/article/pii/S0925231224014759.

A. Proofs for Section 3 and 4

In this appendix, we provide the proofs of all the theoretical results presented in section 3.

Proposition A.1. ***(Proposition 3.1)*** *$E_{(1)}(\theta), \ldots, E_{(n)}(\theta)$ are almost surely (a.s.) everywhere differentiable in θ. In particular, the empirical quantile $\hat{\tau}_n(\theta) = E_{(\lceil \alpha n \rceil)}(\theta)$ is a.s. everywhere differentiable.*

Proof. We will formally show that the ordered statistics $E_{(j)}(\theta)$, with $j = 1, ..., n$, are differentiable for any θ with probability 1. We first recall some notation. Let $E_{(1)}(\theta) \leq \ldots \leq E_{(n)}(\theta)$ denote the order statistics corresponding to the scalar random variables $E(\theta, X_1, Y_1), \ldots, E(\theta, X_n, Y_n)$.
Let us also denote by $\omega(\theta) : [n] \to [n]$ the permutation of indices $[n] := \{1, \ldots, n\}$ that correspond to the order statistics, i.e., $\omega(\theta) = (\omega_1(\theta), \ldots, \omega_n(\theta))$, and $(E_{(1)}(\theta), \ldots E_{(n)}(\theta)) = (E(\theta, X_{\omega_1(\theta)}, Y_{\omega_1(\theta)}), \ldots, E(\theta, X_{\omega_n(\theta)}, Y_{\omega_n(\theta)}))$. Now define the set A_n as follows:

$$A_n = \{(E_1, ..., E_n) : E_i = E_j \text{ for some } i \neq j\}. \tag{9}$$

Now note that, by definition, the conformity score function $E(\theta, X, Y)$ is continuous and differentiable in θ. Now fix some $\bar{\theta}$. Consider the event in which the ordered statistics are such that $E_{(1)}(\bar{\theta}) < \ldots < E_{(n)}(\bar{\theta})$, hence

$$(E(\bar{\theta}, X_{\omega_1(\bar{\theta})}, Y_{\omega_1(\bar{\theta})}), \ldots, E(\bar{\theta}, X_{\omega_n(\bar{\theta})}, Y_{\omega_n(\bar{\theta})})) = (E_{(1)}(\bar{\theta}), \ldots, E_{(n)}(\bar{\theta})) \notin A_n, \tag{10}$$

which means that $\omega(\bar{\theta})$ is the unique ordered statistics permutation for $\{E(\theta, X_i, Y_i)\}_{i=1}^n$ and note that this happens *almost surely* (with probability 1), because $E(\theta, X, Y)$ is an absolutely continuous random variable. The key step is now to note that by continuity of $E(\theta, X_i, Y_i)$ in θ, there exists $\delta > 0$ such that, for $\theta \in \{\theta' : \|\theta' - \bar{\theta}\| \leq \delta\}$, we have $\omega(\theta) = \omega(\bar{\theta})$, which means that, if $\|\theta - \bar{\theta}\| \leq \delta$,

$$\begin{aligned}(E_{(1)}(\theta), \ldots, E_{(n)}(\theta)) &= (E(\theta, X_{\omega_1(\theta)}, Y_{\omega_1(\theta)}), \ldots, E(\theta, X_{\omega_n(\theta)}, Y_{\omega_n(\theta)})) \\ &= (E(\theta, X_{\omega_1(\bar{\theta})}, Y_{\omega_1(\bar{\theta})}), \ldots, E(\theta, X_{\omega_n(\bar{\theta})}, Y_{\omega_n(\bar{\theta})}))\end{aligned} \tag{11}$$

At this point, let $j \in \{1, ..., n\}$, and let us denote $E_{(j)}(\theta) = E(\theta, X_{\omega_j(\theta)}, Y_{\omega_j(\theta)}) = E(\theta, \omega_j(\theta))$, and, for any $\theta \in \{\theta' : \|\theta' - \bar{\theta}\| \leq \delta\}$ the derivative of $E_{(j)}(\theta)$ is

$$\frac{\partial}{\partial \theta} E_{(j)}(\theta) = \frac{\partial}{\partial \theta} E(\theta, \omega_j(\theta)) = \frac{\partial}{\partial \theta} E(\theta, \omega_j(\bar{\theta})) = \frac{\partial E}{\partial \theta}(\theta, \omega_j(\bar{\theta})), \tag{12}$$

which is true because, as we show in (11) above, for $\theta \in \{\theta' : \|\theta' - \bar{\theta}\| \leq \delta\}$, the function $\omega_j(\theta)$ is a constant equal to $\omega_j(\bar{\theta})$. Note that, as we do in the main paper, we here denote by $\frac{\partial E}{\partial \theta}(\theta, \omega_j(\bar{\theta}))$ the partial derivative with respect to θ. Note that, given that the choice of $\bar{\theta}$ is arbitrary, we have shown that the function $\theta \mapsto E(\theta, X_{\omega_j(\theta)}, Y_{\omega_j(\theta)})$ is indeed differentiable with probability 1 for all $j = 1, \ldots, n$.
To be absolutely convinced that (11) is true, note that we can show it by continuity of $\theta \mapsto E(\theta, X, Y)$, as follows: let's fix $\bar{\theta}$ and let us denote again $E_{(j)}(\theta) = E(\theta, X_{\omega_j(\theta)}, Y_{\omega_j(\theta)}) = E(\theta, \omega_j(\theta))$. We want to show that there exists $\delta > 0$ such that $\omega(\bar{\theta}) = \omega(\theta)$ for any $\theta \in \{\theta' : \|\theta' - \bar{\theta}\| \leq \delta\}$. To do so, it is sufficient to show that, for any $\theta \in \{\theta' : \|\theta' - \bar{\theta}\| \leq \delta\}$,

$$E(\theta, \omega_{i+1}(\bar{\theta})) > E(\theta, \omega_i(\bar{\theta})), \text{ for } i = 1, ..., n-1. \tag{13}$$

Let us define

$$\varepsilon = \min_{i=1,\ldots,n-1} \{E(\bar{\theta}, \omega_{i+1}(\bar{\theta})) - E(\bar{\theta}, \omega_i(\bar{\theta}))\}. \tag{14}$$

From continuity of $\theta \mapsto E(\theta, X, Y)$, there exists $\delta > 0$ such that if $\theta \in \{\theta' : \|\theta' - \bar{\theta}\| \leq \delta\}$, we have

$$|E(\theta, \omega_i(\bar{\theta})) - E(\bar{\theta}, \omega_i(\bar{\theta}))| < \frac{\epsilon}{2}. \tag{15}$$

Note that, from (14) and (15), we have for all $i = 1, \ldots, n$,

$$E(\bar{\theta}, \omega_{i+1}(\bar{\theta})) \geq E(\bar{\theta}, \omega_i(\bar{\theta})) + \epsilon, \tag{16}$$

and

$$E(\theta, \omega_i(\bar{\theta})) > E(\bar{\theta}, \omega_i(\bar{\theta})) - \frac{\epsilon}{2}. \tag{17}$$

Hence, note that, starting from this last inequality, and then using (16)

$$\begin{aligned} E(\theta, \omega_{i+1}(\bar{\theta})) &> E(\bar{\theta}, \omega_{i+1}(\bar{\theta})) - \frac{\epsilon}{2} \\ &\geq E(\bar{\theta}, \omega_i(\bar{\theta})) + \frac{\epsilon}{2}. \end{aligned} \tag{18}$$

Now, we can use again (15) (continuity) to show that

$$E(\bar{\theta}, \omega_i(\bar{\theta})) > E(\theta, \omega_i(\bar{\theta})) - \frac{\epsilon}{2}, \tag{19}$$

and thus observe that, for all $i = 1, \ldots, n$,

$$E(\theta, \omega_{i+1}(\bar{\theta})) > E(\bar{\theta}, \omega_i(\bar{\theta})) + \frac{\epsilon}{2} > E(\theta, \omega_i(\bar{\theta})), \tag{20}$$

from which we can confirm that, for $\theta \in \{\theta' : \|\theta' - \bar{\theta}\| \leq \delta\}$, $\omega(\theta) = \omega(\bar{\theta})$, and we can conclude. □

Lemma A.2. *Given any continuous, bounded function h, the function $g(t, \theta) = \mathbb{E}\left[h(\frac{\partial E}{\partial \theta}(\theta, Z) \mid E_\theta(Z) = t\right]$ is continuous in t.*

Proof. Given an absolutely continuous random vector V, we will denote its probability density function (PDF) as $f_V(v)$. Let $\mathcal{E}_\theta$ and $\mathcal{V}_\theta$ denote the support of $E_\theta(Z)$ and $\frac{\partial E}{\partial \theta}(\theta, Z)$ respectively, i.e. the set of points where the PDF is strictly positive. Suppose that $t \in \mathcal{E}_\theta$. Then,

$$\begin{aligned} g(t, \theta) &= \mathbb{E}\left[h\left(\frac{\partial E}{\partial \theta}(\theta, Z)\right) \mid E_\theta(Z) = t\right] \\ &= \int_{\mathcal{V}_\theta} f_{\frac{\partial E}{\partial \theta}(\theta, Z) \mid E_\theta(Z) = t}(v)\, h\left(\frac{\partial E}{\partial \theta}(\theta, v)\right) \mathrm{d}v \\ &= \int_{\mathcal{V}_\theta} \frac{f_{\frac{\partial E}{\partial \theta}(\theta, Z), E_\theta(Z)}(v, t)}{f_{E_\theta(Z)}(t)} h\left(\frac{\partial E}{\partial \theta}(\theta, v)\right) \mathrm{d}v. \end{aligned}$$

Since $f_{\frac{\partial E}{\partial \theta}(\theta, Z), E_\theta(Z)}(v, t)$ is continuous in t (from Assumption 2), then so is $f_{E_\theta(Z)}(t)$. Subsequently, their ratio is continuous, and the result follows.

□

Proposition A.3. ***(Proposition 3.2)*** *Let $\hat{\tau}_n(\theta) = E_{(\lceil \alpha n \rceil)}(\theta)$. Then,*

$$\frac{\partial \hat{\tau}_n}{\partial \theta}(\theta) \xrightarrow{\text{dist}} \frac{\partial E}{\partial \theta}(\theta, Z)\,\big|_{E_\theta(Z) = \tau(\theta)} \tag{21}$$

as $n \to \infty$.

Proof. Let $Z_i = (X_i, Y_i)$ for $i \in \{1, \ldots, n\}$ and $Z = (X, Y)$. Let h be a bounded, continuous function. Then,

$$\begin{aligned} &\mathbb{E}\left[h\left(\frac{\partial \hat{\tau}_n}{\partial \theta}(\theta)\right)\right] = \mathbb{E}\left[h\left(\frac{\partial E}{\partial \theta}(\theta, Z_{\omega_{\lceil \alpha n \rceil}(\theta)})\right)\right] \\ &= \mathbb{E}_{t \sim \text{dist}(\hat{\tau}_n(\theta))}\left[\mathbb{E}\left[h\left(\frac{\partial E}{\partial \theta}(\theta, Z_{\omega_{\lceil \alpha n \rceil}(\theta)})\right) \mid \hat{\tau}_n(\theta) = t\right]\right]. \end{aligned}$$

Now, focusing on the inner expectation, we can see that $\hat{\tau}_n(\theta) = t$ occurs almost surely (assuming no ties in the scores $E_\theta(Z_1), \ldots, E_\theta(Z_n)$) if and only if exactly one of the scores is equal to t and exactly $\lceil \alpha n \rceil - 1$ of the remaining being strictly smaller than t. Due to exchangeability of the scores, every ordering is equally likely. Therefore,

$$\begin{aligned} &\mathbb{E}\left[h\left(\frac{\partial E}{\partial \theta}(\theta, Z_{\omega_{\lceil \alpha n \rceil}(\theta)})\right) \mid \hat{\tau}_n(\theta) = t\right] \\ &= \mathbb{E}\Big[h\left(\frac{\partial E}{\partial \theta}(\theta, Z_1)\right) \mid E_\theta(Z_1) = t, \\ &\qquad \sum_{i=2}^{n} 1_{E_\theta(Z_i) < t} = \lceil \alpha n \rceil - 1\Big]. \end{aligned}$$

Now, due to the assumed independence of $Z_1, \ldots, Z_n$, we can drop $\{\sum_{i=2}^{n} 1_{E(\theta,Z_i)<t} = \lceil \alpha n \rceil - 1\}$ from the conditional, because $h\left(\frac{\partial E}{\partial \theta}(\theta, Z_1)\right)$ does not depend on $Z_2, \ldots, Z_n$. Subsequently,

$$\begin{aligned}
&\mathbb{E}\left[h\left(\frac{\partial \hat{\tau}_n}{\partial \theta}(\theta)\right)\right] \\
&= \mathbb{E}_{t\sim \mathrm{dist}(\hat{\tau}_n(\theta))}\left[\mathbb{E}\left[h\left(\frac{\partial E}{\partial \theta}(\theta, Z_1)\right) \mid E_\theta(Z_1) = t\right]\right] \\
&= \mathbb{E}_{t\sim \mathrm{dist}(\hat{\tau}_n(\theta))}\left[\mathbb{E}\left[h\left(\frac{\partial E}{\partial \theta}(\theta, Z)\right) \mid E_\theta(Z) = t\right]\right] \\
&= \mathbb{E}_{t\sim \mathrm{dist}(\hat{\tau}_n(\theta))}[g(t,\theta)],
\end{aligned}$$

where $g(t,\theta) := \mathbb{E}\left[h\left(\frac{\partial E}{\partial \theta}(\theta, Z)\right) \mid E_\theta(Z) = t\right]$. Therefore,

$$\mathbb{E}\left[h\left(\frac{\partial \hat{\tau}_n}{\partial \theta}(\theta)\right)\right] = \mathbb{E}\left[g(\hat{\tau}_n(\theta), \theta)\right].$$

Noting that $\hat{\tau}_n(\theta) \to \tau(\theta)$ in distribution, it follows from Lemma A.2 and by the continuous mapping theorem that $g(\hat{\tau}_n(\theta), \theta) \xrightarrow{\text{dist}} g(\tau(\theta), \theta)$. We can rewrite this as

$$\begin{aligned}
&\lim_{n\to\infty} \mathbb{E}\left[h\left(\frac{\partial E}{\partial \theta}(\theta, Z_{\omega_{\lceil \alpha n \rceil}})\right)\right] \\
&= \mathbb{E}\left[h\left(\frac{\partial E}{\partial \theta}(\theta, Z)\right) \mid E_\theta(Z) = \tau(\theta)\right].
\end{aligned}$$

Since h was arbitrary, it follows from the Portmanteau lemma that

$$\frac{\partial E}{\partial \theta}(\theta, Z_{\omega_{\lceil \alpha n \rceil}}) \xrightarrow{\text{dist}} \frac{\partial E}{\partial \theta}(\theta, Z)\big|_{E_\theta(Z)=\tau(\theta)} \tag{22}$$

as $n \to \infty$.

□

Corollary A.4. *Under the same conditions as Proposition 3.2, we have*

$$\mathbb{E}\left[\frac{\partial \hat{\tau}_n}{\partial \theta}(\theta)\right] \to \mathbb{E}\left[\frac{\partial E}{\partial \theta}(\theta, Z) \,\middle|\, E_\theta(Z) = \tau(\theta)\right] \tag{23}$$

and

$$\mathrm{cov}\left(\frac{\partial \hat{\tau}_n}{\partial \theta}(\theta)\right) \to \mathrm{cov}\left(\frac{\partial E}{\partial \theta}(\theta, Z) \,\middle|\, E_\theta(Z) = \tau(\theta)\right) \tag{24}$$

as $n \to \infty$.

Proof. For the corollary, simply note that we can apply the Portmanteau lemma again, choosing some continuous, bounded h such that $h(\theta) = \theta_l$ and $h(\theta) = \theta_l\theta_p$ (with l, p indexing components of θ) over $\{\|\theta\| \leq M\}$. This way, noting that $\|\frac{\partial E}{\partial \theta}(\theta, z)\| \leq M$, we can see that

$$\mathbb{E}\left[\frac{\partial \hat{\tau}_n}{\partial \theta_l}(\theta)\right] \to \mathbb{E}\left[\frac{\partial E}{\partial \theta_l}(\theta, Z) \,\middle|\, E_\theta(Z) = \tau(\theta)\right]$$

and

$$\begin{aligned}
&\mathbb{E}\left[\frac{\partial \hat{\tau}_n}{\partial \theta_l}(\theta)\frac{\partial \hat{\tau}_n}{\partial \theta_p}(\theta)\right] \\
&\to \mathbb{E}\left[\frac{\partial E}{\partial \theta_l}(\theta, Z)\frac{\partial E}{\partial \theta_p}(\theta, Z) \,\middle|\, E_\theta(Z) = \tau(\theta)\right].
\end{aligned}$$

Gathering all the indices l, p completes the proof. □

Proof of Theorem 4.1

We start with two preliminary results that we will use in the proof.

Some preliminaries. First, we recall a well-known result. Let $k \sim \text{Binomial}(n, p)$ be a random variable sampled from a Binomial distribution with n trials and with probability p of success. The following holds:

$$\mathbb{E}\left[\frac{1}{1+k}\right] = \frac{(1-(1-p)^{n+1})}{(n+1)p}. \tag{25}$$

Note that this follows from the following simple steps:

$$\begin{aligned}
\mathbb{E}\left[\frac{1}{1+k}\right] &= \sum_{k=0}^{n} \frac{1}{1+k} \cdot \binom{n}{k} p^k (1-p)^{n-k} \\
&= \frac{1}{p(n+1)} \sum_{k=0}^{n} \binom{n+1}{k+1} p^{k+1} (1-p)^{n-k} \\
&= \frac{1}{p(n+1)} \sum_{j=1}^{n+1} \binom{n+1}{j} p^{j} (1-p)^{n+1-j} \\
&= \frac{\left(1-(1-p)^{n+1}\right)}{p(n+1)}.
\end{aligned}$$

Next, we state another well-known identity. Let us consider the following recursion:

$$a_{n+1} = \rho\, a_n + b,$$

where $\rho > 0$. Simply unrolling the recursion, we can obtain

$$a_n = \rho^n a_0 + b\left(\frac{1-\rho^n}{1-\rho}\right). \tag{26}$$

Proof of the Theorem.

Before we proceed, we introduce some notation. For simplicity, since $\hat{\tau}(\theta)$ converges *almost surely* to $\tau(\theta)$, we assume $\hat{\tau}(\theta) = \tau(\theta)$ in the analysis. Let us then denote

$$\begin{aligned}
G_i(\theta) &= \frac{\partial E}{\partial \theta}(\theta, X_i, Y_i), \\
A_{\varepsilon,i}(\theta) &= \{\varepsilon \le E(\theta, X_i, Y_i) - \tau(\theta) \le \varepsilon\}, \\
R_{\varepsilon,n}(\theta) &= \sum_{i=1}^{n} \chi_{A_{\varepsilon,i}(\theta)}, \\
S_{\epsilon,n}(\theta) &= \{i \in [n] : \chi_{A_{\epsilon,i}(\theta)} = 1\},
\end{aligned} \tag{27}$$

where $\chi_{A_{\varepsilon,i}(\theta)}$ is an indicator function for the event $A_{\varepsilon,i}(\theta)$, i.e.,

$$\chi_{A_{\varepsilon,i}(\theta)} = \begin{cases} 1, & \text{if } |E_\theta(X_i, Y_i) - \tau(\theta)| \le \varepsilon \\ 0, & \text{if } |E_\theta(X_i, Y_i) - \tau(\theta)| > \varepsilon \end{cases}. \tag{28}$$

We are now ready to analyze the estimator $\hat{\eta}_{\varepsilon,n}(\theta)$ for $\eta_\varepsilon(\theta)$:

$$\hat{\eta}_{\varepsilon,n}(\theta) = \begin{cases} \frac{1}{R_{\varepsilon,n}(\theta)} \sum_{i=1}^{n} \chi_{A_{\varepsilon,i}(\theta)} G_i(\theta), & \text{if } R_{\varepsilon,n}(\theta) > 0 \\ 0 & \text{if } R_{\varepsilon,n}(\theta) = 0 \end{cases}, \tag{29}$$

where ε and n are denoted explicitly to remove ambiguity.

Equipped with the basic results established earlier in this subsection, we can proceed first with proving assertion *(i)*. Note that, by definition (29), and because $\{X_i, Y_i\}_{i=1}^n$ are sampled independently, we have

$$\mathbb{E}\left[\frac{1}{|S_{\epsilon,n}(\theta)|}\sum_{i\in S_{\epsilon,n}(\theta)} G_i | \bigcap_{i\in S_{\epsilon,n}(\theta)} A_{\epsilon,i}(\theta)\right] = \frac{1}{|S_{\epsilon,n}(\theta)|}\sum_{i\in S_{\epsilon,n}(\theta)} \mathbb{E}\left[\chi_{A_{\epsilon,i}(\theta)} G_i(\theta) | A_{\epsilon,i}(\theta)\right] = \eta_\epsilon. \tag{30}$$

Also note that $S_{\epsilon,n}(\theta) = \emptyset$ is equivalent to $R_{\epsilon,n}(\theta) = 0$, and that

$$\mathbb{P}\left(S_{\epsilon,n}(\theta) = \emptyset\right) = \mathbb{P}\left(R_{\epsilon,n}(\theta) = 0\right) = q_\epsilon(\theta)^n, \tag{31}$$

with $q_\epsilon(\theta) = 1 - p_\epsilon(\theta)$ and $p_\epsilon(\theta) = \mathbb{P}(A_{\varepsilon,i}(\theta))$. For simplicity, we denote $p = p_\varepsilon(\theta)$ and $q = 1 - p$ for the remainder of the proof. Hence, we can get (i) as follows:

$$\mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta)\right] = q^n \mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta) | R_{\epsilon,n}(\theta) = 0\right] + (1 - q^n)\eta_\epsilon, \tag{32}$$

where we used the fact that $\sum_{S\neq\emptyset} \mathbb{P}\left(S_{\epsilon,n}(\theta) = S\right) = 1 - \mathbb{P}\left(S_{\epsilon,n}(\theta) = \emptyset\right) = 1 - q^n$.

Now, we prove (ii). We start by analyzing $\mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta)\hat{\eta}_{\epsilon,n}(\theta)^\top\right]$:

$$\begin{aligned}
\mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta)\hat{\eta}_{\epsilon,n}(\theta)^\top\right] &= \sum_{S\subseteq[n]} \mathbb{P}\left(S_{\epsilon,n}(\theta) = S\right) \mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta)\hat{\eta}_{\epsilon,n}(\theta)^\top | S_{\epsilon,n}(\theta) = S\right] \\
&= \mathbb{P}\left(S_{\epsilon,n}(\theta) = \emptyset\right) \mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta)\hat{\eta}_{\epsilon,n}(\theta)^\top | S_{\epsilon,n}(\theta) = S_{\epsilon,n}(\theta)\right] \\
&+ \sum_{S\neq\emptyset} \mathbb{P}\left(S_{\epsilon,n}(\theta) = S\right) \mathbb{E}\left[\left(\frac{1}{|S|}\sum_{i\in S} G_i(\theta)\right)\left(\frac{1}{|S|}\sum_{i\in S} G_i(\theta)\right)^\top | \bigcap_{i\in S} A_{\epsilon,i}(\theta)\right] \\
&= \mathbb{P}\left(S_{\epsilon,n}(\theta) = \emptyset\right) \mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta)\hat{\eta}_{\epsilon,n}(\theta)^\top | R_{\epsilon,n}(\theta) = 0\right] \\
&+ \sum_{S\neq\emptyset} \mathbb{P}\left(S_{\epsilon,n}(\theta) = S\right) \frac{1}{|S|^2} \sum_{i\in S}\sum_{j\in S} \mathbb{E}\left[G_i(\theta) G_j(\theta)^\top | A_{\epsilon,i}(\theta), A_{\epsilon,j}(\theta)\right].
\end{aligned} \tag{33}$$

Now note that

$$\begin{aligned}
\mathbb{E}\left[G_i(\theta)G_j(\theta)^\top | A_{\epsilon,i}(\theta), A_{\epsilon,j}(\theta)\right] &= \delta_{i,j}\mathbb{E}\left[G_i(\theta)G_i(\theta)^\top | A_{\epsilon,i}(\theta)\right] \\
&+ (1 - \delta_{ij})\mathbb{E}\left[G_i(\theta) | A_{\epsilon,i}(\theta)\right] \mathbb{E}\left[G_j(\theta)^\top | A_{\epsilon,j}(\theta)\right] \\
&= \mathbb{E}\left[G_i(\theta) | A_{\epsilon,i}(\theta)\right] \mathbb{E}\left[G_j(\theta)^\top | A_{\epsilon,i}(\theta)\right] \\
&+ \delta_{ij}\left(\mathbb{E}\left[G_i(\theta)G_i(\theta)^\top | A_{\epsilon,i}(\theta)\right] - \mathbb{E}\left[G_i(\theta) | A_{\epsilon,i}(\theta)\right] \mathbb{E}\left[G_i(\theta)^\top | A_{\epsilon,i}(\theta)\right]\right) \\
&= \eta_\epsilon\eta_\epsilon^\top + \delta_{ij}\Sigma_\epsilon,
\end{aligned} \tag{34}$$

where $\delta_{i,j} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases}$. We used the fact that $\{X_i, Y_i\}_{i=1}^n$ are sampled i.i.d. and the definitions in (27). Now, we can proceed as follows:

$$\begin{aligned}
\mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta)\hat{\eta}_{\epsilon,n}(\theta)^\top\right] &= q^n\mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta)\hat{\eta}_{\epsilon,n}(\theta)^\top | R_{\epsilon,n}(\theta) = 0\right] \\
&+ \sum_{S\neq\emptyset} \frac{\mathbb{P}\left(S_{\epsilon,n}(\theta) = S\right)}{|S|}\left(|S|\eta_\epsilon\eta_\epsilon^\top + \Sigma_\epsilon\right) \\
&= q^n\mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta)\hat{\eta}_{\epsilon,n}(\theta)^\top | R_{\epsilon,n}(\theta) = 0\right] \\
&+ (1 - q^n)\eta_\epsilon\eta_\epsilon^\top + f_n\Sigma_\epsilon,
\end{aligned} \tag{35}$$

where we write

$$f_n = \sum_{S\neq\emptyset} \frac{\mathbb{P}\left(S_{\epsilon,n}(\theta) = S\right)}{|S|}. \tag{36}$$

Now, we will show that

$$f_n \leq \frac{2 - p}{pn}. \tag{37}$$

First, let us define the following function

$$f(k) = \begin{cases} 0, & \text{if } k = 0 \\ \frac{1}{k} & \text{if } k \geq 1 \end{cases}, \tag{38}$$

and note that

$$f_n = \mathbb{E}\left[f(|S_{\epsilon,n}(\theta)|)\right] = \mathbb{E}\left[f(R_{\epsilon,n}(\theta))\right]. \tag{39}$$

Now note that

$$\begin{aligned} f_{n+1} &= \mathbb{E}\left[f(R_{\epsilon,n+1}(\theta))\right] \\ &= \mathbb{P}\left(A_{\epsilon,i}(\theta)^c\right)\mathbb{E}\left[f(R_{\epsilon,n}(\theta))\right] + \mathbb{P}\left(A_{\epsilon,i}(\theta)\right)\mathbb{E}\left[f(1+R_{\epsilon,n}(\theta))\right] \\ &= q\mathbb{E}\left[f(R_{\epsilon,n}(\theta))\right] + p\mathbb{E}\left[f(1+R_{\epsilon,n}(\theta))\right] \\ &= qf_n + p\mathbb{E}\left[\frac{1}{1+R_{\epsilon,n}(\theta)}\right] \\ &= qf_n + \frac{1-q^{n+1}}{n+1}, \end{aligned} \tag{40}$$

where, in the last equation, we used the fact shown in the preliminaries (see (25)):

$$\mathbb{E}\left[\frac{1}{1+R_{\epsilon,n}(\theta)}\right] = \frac{1-q^{n+1}}{p(n+1)}. \tag{41}$$

Now let $a_n = nf_n$. We can write

$$(n+1)f_{n+1} = (n+1)\left(qf_n + \frac{1-q^n}{n+1}\right), \tag{42}$$

from which we obtain the following recursion:

$$\begin{aligned} a_{n+1} &= qnf_n + qf_n + (1-q^{n+1}) \\ &= qa_n + qf_n + (1-q^{n+1}) \\ &\leq qa_n + 1 + q, \end{aligned} \tag{43}$$

where we used the fact that $f_n \leq 1$ and that $1 - q^n \leq 1$. With this recursion, we can now use the result illustrated in the preliminaries in (26) and get, using $q = 1 - p$,

$$a_n \leq q^n a_0 + \frac{1-q^n}{1-q}(1+q) \leq \frac{2-p}{p}. \tag{44}$$

From the above inequality, we can conclude that

$$0 \leq f_n = \frac{a_n}{n} \leq \frac{2-p}{pn}. \tag{45}$$

Plugging this last result in (35), we can get

$$\mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta)\hat{\eta}_{\epsilon,n}(\theta)^\top\right] \preceq q^n\mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta)\hat{\eta}_{\epsilon,n}(\theta)^\top | R_{\epsilon,n}(\theta) = 0\right] + (1-q^n)\eta_\epsilon\eta_\epsilon^\top + \frac{2-p}{pn}\Sigma_\epsilon. \tag{46}$$

We are now in the position to write and bound $\mathrm{cov}(\hat{\eta}_{\epsilon,n}(\theta))$:

$$\begin{aligned}
\mathrm{cov}\left(\hat{\eta}_{\epsilon,n}(\theta)\right) &= \mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta)\hat{\eta}_{\epsilon,n}(\theta)^\top\right] - \left[\hat{\eta}_{\epsilon,n}(\theta)\right]\left[\hat{\eta}_{\epsilon,n}(\theta)^\top\right] \\
&\preceq q^n \mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta)\hat{\eta}_{\epsilon,n}(\theta)^\top | R_{\epsilon,n}(\theta) = 0\right] + (1-q^n)\eta_\epsilon\eta_\epsilon^\top + \frac{2-p}{pn}\Sigma_\epsilon \\
&- \left(q^n \mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta) | R_{\epsilon,n}(\theta) = 0\right] + (1-q^n)\eta_\epsilon\right) \\
&\cdot \left(q^n \mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta) | R_{\epsilon,n}(\theta) = 0\right] + (1-q^n)\eta_\epsilon\right)^\top \\
&= \frac{2-p}{pn}\Sigma_\epsilon + (1-q^n)\eta_\epsilon\eta_\epsilon^\top - (1-q^n)^2\eta_\epsilon\eta_\epsilon^\top \\
&= \frac{2-p}{pn}\Sigma_\epsilon + (1-q^n)(1-(1-q^n))\eta_\epsilon\eta_\epsilon^\top \\
&= \frac{2-p}{pn}\Sigma_\epsilon + (1-q^n)q^n\eta_\epsilon\eta_\epsilon^\top \\
&\preceq \frac{2-p}{pn}\Sigma_\epsilon + q^n\eta_\epsilon\eta_\epsilon^\top,
\end{aligned} \tag{47}$$

where we used (i), the fact that $1-q^n \leq 1$ and the fact that $\mathbb{E}\left[\hat{\eta}_{\epsilon,n}(\theta) | R_{\epsilon,n}(\theta) = 0\right] = 0$, which follows by (29).

B. Useful Facts and Derivations

In this appendix, we provide, for completeness, some useful facts and explicit derivations of properties that we use in the paper. In particular, we explicitly derive equation (5) using the generalize chain rule (GCR).

B.1. Explicit derivation of equation (5)

Please note that equation (5) follows from taking the derivative of a function of multiple variables and the chain rule. This is also called the *generalized chain rule* in some textbooks (Herman & Strang, 2018)(see Theorem 4.10). In the paper, when writing

$$\frac{\partial}{\partial\theta}\ell(\theta, \hat{\tau}(\theta), X, Y), \tag{48}$$

we mean the *total* derivative of the function $\theta \mapsto l(\theta, \hat{\tau}(\theta), X, Y)$, evaluated at a dummy θ. On the other hand, when writing

$$\frac{\partial\ell}{\partial\theta}(\theta, \hat{\tau}(\theta), x, y), \tag{49}$$

we mean the *partial* derivative of $\ell(\theta, q, x, y)$ with respect to θ, evaluated at $(\theta, q, x, y) = (\theta, \hat{\tau}(\theta), X, Y)$. The difference is that, in the partial derivative, $\hat{\tau}(\theta)$ is treated as a constant, whereas for the total derivative we do not treat $\hat{\tau}(\theta)$ as a constant. Now, the generalized chain rule (in vector form) can be written as follows: let $u(\theta) \in \mathbb{R}^n$ and $v(\theta) \in \mathbb{R}^m$ be two differentiable functions of θ, and $f(u, v)$ a differentiable function of two vector variables u and v. Then

$$\frac{\partial}{\partial\theta}f(u(\theta), v(\theta)) = \left(\frac{\partial u}{\partial\theta}(\theta)\right)^\top \frac{\partial f}{\partial u}(u(\theta), v(\theta)) + \left(\frac{\partial v}{\partial\theta}(\theta)\right)^\top \frac{\partial f}{\partial v}(u(\theta), v(\theta)), \tag{50}$$

where $\frac{\partial u}{\partial\theta}(\theta)$ is the Jacobian of $u(\theta)$, i.e., the matrix with $\frac{\partial u_i}{\partial\theta_j}(\theta)$ in the i-th row and j-th column (equivalently, $\frac{\partial v}{\partial\theta}(\theta)$ is the Jacobian of $v(\theta)$). Note that in the case of $\ell(\theta, \hat{\tau}(\theta), x, y)$, x and y do not depend on θ so we can focus on ℓ as a function of the two functions $u(\theta) = \theta$ and $v(\theta) = \hat{\tau}(\theta)$. Replacing these $u(\theta)$ and $v(\theta)$ in equation (50), and replacing $f(u(\theta), v(\theta))$ with $\ell(\theta, \hat{\tau}(\theta), x, y)$ we see that then

$$\frac{\partial}{\partial\theta}\ell(\theta, \hat{\tau}(\theta), x, y) = \frac{\partial\ell}{\partial\theta}(\theta, \hat{\tau}(\theta), x, y) + \frac{\partial\ell}{\partial\hat{\tau}}(\theta, \hat{\tau}(\theta), x, y)\frac{\partial\hat{\tau}}{\partial\theta}(\theta), \tag{51}$$

which is precisely equation (5) in the main paper, where we used the fact that $\left(\frac{\partial\theta}{\partial\theta}\right) = I_d$, where I_d is a $d \times d$ identity matrix, with d the dimension of θ.

Given that usually in textbooks the generalized chain rule (GCR) is only shown for scalar multi-variable functions, we

now report the derivation of equation (5) using the scalar GCR as reported and proved in the statement of Theorem 4.10 in (Herman & Strang, 2018). Hence, we will now provide the derivation of (5) at a more granular level. Consider a differentiable function ℓ of k variables, $\ell : \mathbb{R}^k \to \mathbb{R}$. Now let $f_1, ..., f_k$ be differentiable functions, with $f_i : \mathbb{R}^d \to \mathbb{R}$, for $i = 1, ..., k$ and some $d \geq 1$. Then, denoting a vector $[t_1, ..., t_d] \in \mathbb{R}^d$ and $w = \ell(f_1(t_1, ..., t_d), ..., f_k(t_1, ..., t_d))$ we have (GCR):

$$\frac{\partial w}{\partial t_j} = \sum_{i=1}^{k} \frac{\partial w}{\partial f_i} \frac{\partial f_i}{\partial t_j}. \tag{52}$$

Now note that in the case of our paper, we have $w = \ell(\theta, \hat{\tau}(\theta), x, y)$. Note that x and y have no dependency on parameters in θ and hence their derivatives will be zero. We can then focus on θ and $\hat{\tau}(\theta)$. For convenience, note that we can write $\theta = [\theta_1, ..., \theta_d]$. Now note that the gradient of w is

$$\frac{\partial}{\partial \theta}[w] = \left[\frac{\partial w}{\partial \theta_1}, ..., \frac{\partial w}{\partial \theta_d}\right]^\top. \tag{53}$$

Now note that, for some $j \in \{1, ..., d\}$, using the chain rule (52) above,

$$\begin{aligned} \frac{\partial w}{\partial \theta_j} &= \sum_{i=1}^{d} \frac{\partial w}{\partial \theta_i} \frac{\partial \theta_i}{\partial \theta_j} + \frac{\partial \ell}{\partial \hat{\tau}}(\theta, \hat{\tau}(\theta), x, y) \frac{\partial \hat{\tau}}{\partial \theta_j}(\theta) \\ &\quad + \frac{\partial w}{\partial x} \frac{\partial x}{\partial \theta_j} + \frac{\partial w}{\partial y} \frac{\partial y}{\partial \theta_j} \\ &= \frac{\partial \ell}{\partial \theta_j}(\theta, \hat{\tau}(\theta), x, y) + \frac{\partial \ell}{\partial \hat{\tau}}(\theta, \hat{\tau}(\theta), x, y) \frac{\partial \hat{\tau}}{\partial \theta_j}(\theta), \end{aligned} \tag{54}$$

where we used the fact that $\frac{\partial \theta_i}{\partial \theta_j} = 0$ if $i \neq j$ and $\frac{\partial \theta_i}{\partial \theta_i} = 1$. We also explicitly used the fact that $\frac{\partial x}{\partial \theta_j} = 0$ and $\frac{\partial y}{\partial \theta_j} = 0$ because the samples do not depend on the parameter θ. Stacking together $\frac{\partial w}{\partial \theta_j}$ we can see that we obtain precisely equation (5) of the paper:

$$\begin{aligned} \frac{\partial}{\partial \theta}[w] &= \frac{\partial}{\partial \theta}[\ell(\theta, \hat{\tau}(\theta), X, Y)] \\ &= \frac{\partial \ell}{\partial \theta}(\theta, \hat{\tau}(\theta), X, Y) + \frac{\partial \ell}{\partial \hat{\tau}}(\theta, \hat{\tau}(\theta), X, Y) \frac{\partial \hat{\tau}}{\partial \theta}(\theta). \end{aligned} \tag{55}$$

C. Additional experiments

Here, we provide additional experimental results to complement the findings in the main paper.

C.1. GMM

As a warm-up, we validate the results of Theorem 4.1 on a synthetic Gaussian Mixture Model (GMM) dataset. To tune the ε-estimator in (8), we employ the m-ranking method described in section 4.1. The results, as shown in Figure 4, illustrate that our estimator (`VR-ConfTr`) reduces variance effectively, while the naive one (`ConfTr`) is sample inefficient.

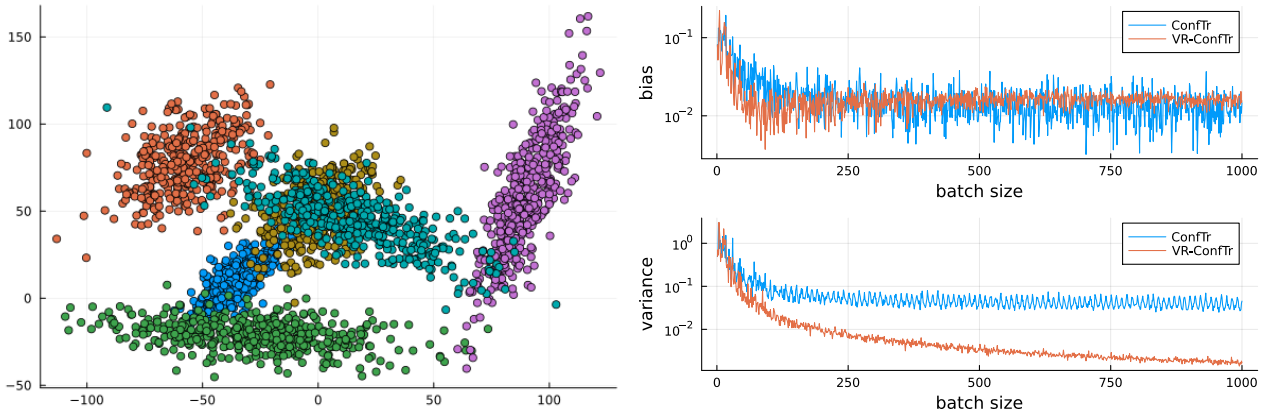

Figure 4. Sample batch from the GMM distribution (left); bias and variance for the quantile gradient estimates, comparing `ConfTr` and `VR-ConfTr` on the GMM dataset (right).

C.2. Additional Training Curves

We first present additional training curves, specifically the test loss and accuracy per epoch for MNIST, Fashion-MNIST, KMNIST, and OrganAMNIST. These plots highlights the performance throughout the training process, providing further insights into convergence behavior and generalization performance. It can be seen that the test loss exhibits a pattern similar to the training loss in 2. In terms of accuracy, `VR-ConfTr` achieves higher accuracy than `ConfTr`.

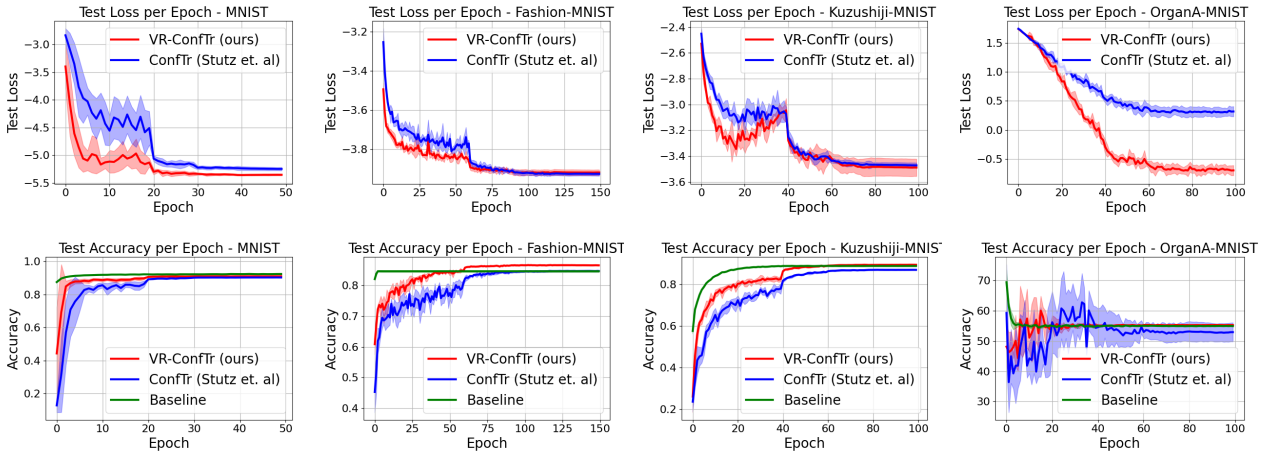

Figure 5. Learning curves for MNIST, Fashion-MNIST, Kuzushiji-MNIST, and OrganAMNIST. For each dataset, we show the test loss on the first row and tets accuracy on the bottom row at the end of each epoch.

C.2.1. Variance of the gradients over the course of training

In this section, we present visualization for the variance of the estimated quantile gradients during training for our proposed method `Vr-ConfTr`, compared to `ConfTr` in figure 6. We conduct this experiment on the MNIST dataset, using the m-ranking estimator with `Vr-ConfTr`, and evaluate performance across different batch sizes. This analysis aims to empirically substantiate our claim that `Vr-ConfTr` reduces variance of the estimated quantile gradients over the epochs, leading to more stable gradient updates and improved final performance. Furthermore, we demonstrate that with an appropriate choice of the hyperparameter m for the m-ranking estimator, `Vr-ConfTr` not only reduces variance but also shows improvements in terms of the bias of the estimated quantile gradients during training. In order to compute the variance and bias for the estimated quantile gradient $\widehat{\frac{\partial \tau}{\partial \theta}}$, we estimate the population quantile $\tau(\theta)$ and its gradient $\frac{\partial \tau}{\partial \theta}$ at each model update utilizing the full training, calibration, and test datasets.

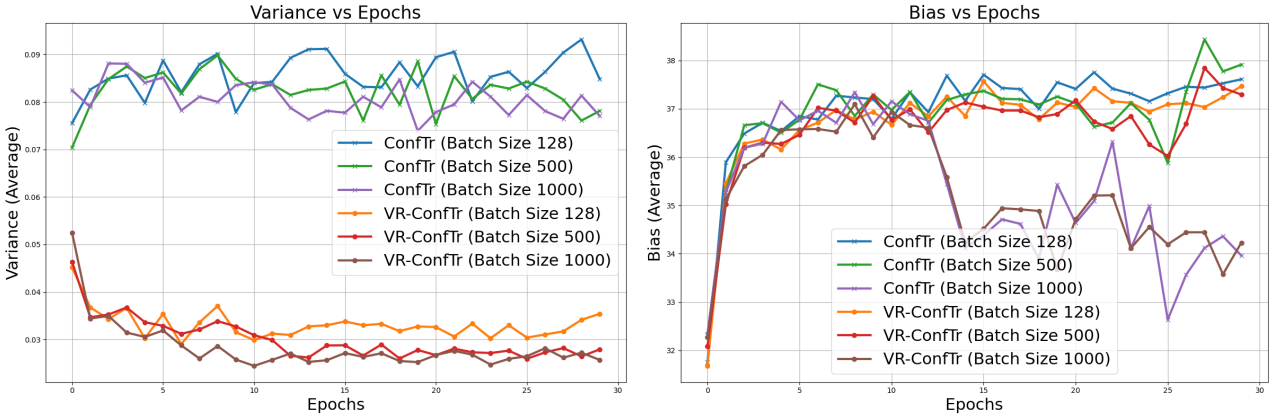

Figure 6. Variance and bias of the estimated quantile gradients during training for `ConfTr` and `Vr-ConfTr`, evaluated on the MNIST dataset across different batch sizes. The left figure shows the variance of the gradients over epochs. The right panel illustrates the bias of the estimated gradients, demonstrating that `Vr-ConfTr` maintains low bias while effectively reducing variance.

C.3. Ablation study for m and ε

C.3.1. ε-threshold tuning ablation study

This study evaluates the bias and variance of the $\widehat{\frac{\partial \tau}{\partial \theta}}$ using the ε-threshold estimator and tuning the threshold ε, with `VR-ConfTr` for the GMM dataset depicted in figure 4. Figure 7 shows how varying ε impacts the estimator's performance, highlighting the trade-offs between bias and variance of $\widehat{\frac{\partial \tau}{\partial \theta}}$ as ε changes.

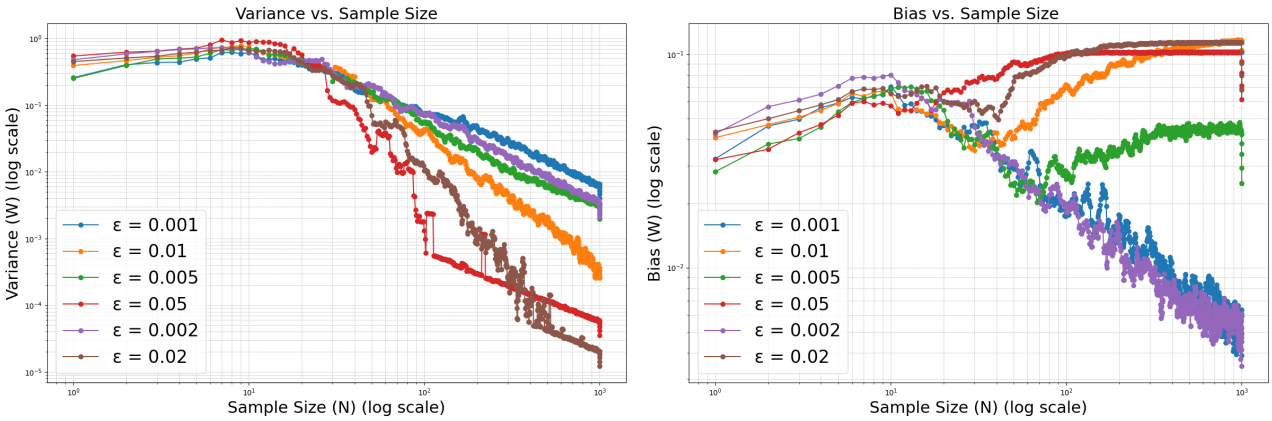

Figure 7. Bias and variance for the quantile gradient estimates tuning the ε threshold in `VR-ConfTr` on the GMM dataset. The left panel shows the variance, and the right panel shows the bias for different ε values.

C.3.2. ε-THRESHOLD TUNING WITH m-RANKING ABLATION STUDY

We evaluate the bias and variance of $\widehat{\frac{\partial \tau}{\partial \theta}}$ this time using the m-ranking strategy to fine-tune ε with `VR-ConfTr` for the GMM dataset. Figure 8 shows how varying m impacts the estimator's performance, highlighting the trade-offs between bias and variance of $\widehat{\frac{\partial \tau}{\partial \theta}}$ as m changes. Here m explicitly depends on the desired miscoverage rate α and the sample size n.

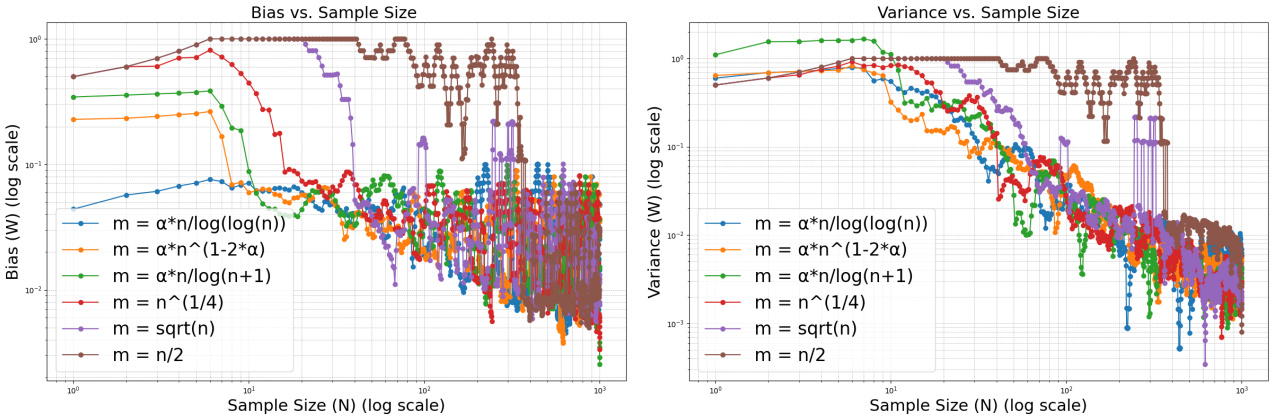

Figure 8. Bias and variance for the quantile gradient estimate using the m-ranking to adaptively tune ε with `VR-ConfTr` on the GMM dataset. The left panel shows the bias, and the right panel shows the variance for different m values

C.3.3. ON TUNING THE ϵ-THRESHOLD WITH m-RANKING

In practice, using the ε-estimator, when training the models, we noticed that a "good" value of ε *varies significantly* across iterations. Note that a good value of the threshold ε not only depends on the specific batch B_{cal} at a given iteration, but also on the model parameters θ at that iteration. Hence, hyper-parameter tuning with the ε-threshold estimator requires some heuristic to adapt the threshold to specific iterations. In this sense, the m-ranking estimator is a natural heuristic for a batch and parameter-dependent choice of the threshold ε. We noticed indeed that performing hyper-parameter tuning of the m-ranking estimator we were able to provide a good value of m to be used *across all iterations*, which from the point of view of hyper-parameter tuning is a great advantage.
To empirically illustrate this connection and validate the importance of dynamically tuning the ε-threshold estimator, figure 9 presents the adaptive choice of the ε-threshold estimator on the Fashion-MNIST dataset when using m-ranking tuning strategy with $m = 6$.

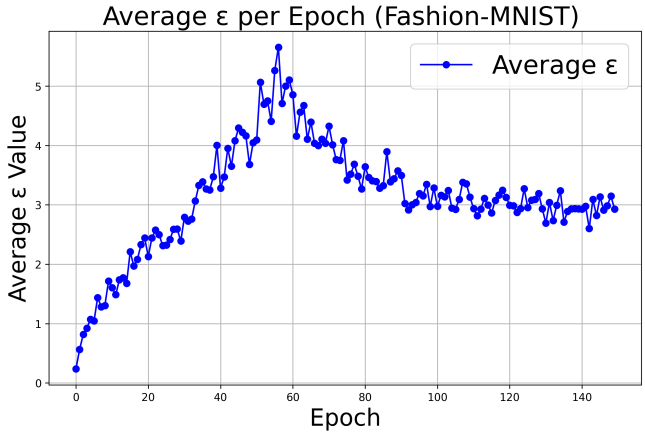

Figure 9. Adaptive tuning of the ε-threshold estimator on Fashion-MNIST using the m-ranking strategy. The variability in ε underscores the necessity of dynamic adjustment in threshold-based approaches.

C.4. Class-conditional coverage and set size

We evaluated the trained models in terms of class-conditional coverage and set size, using the same CP-procedure applied post-training with the standard `THR` method and $\alpha = 0.01$. Figure 18 displays the class-conditional coverage and set sizes for each dataset. The results show the effectiveness of `Vr-ConfTr` in achieving reliable class-conditional coverage while outperforming `ConfTr` in terms of producing smaller class-conditional prediction set sizes. The results are taken as the average over all the training and testing trials.

C.5. Tuning `VR-ConfTr`: Number of Points for Gradient Estimation (`m`)

In `VR-ConfTr`, the number of points (m) used in the m-ranking strategy plays a crucial role in the bias-variance trade-off. Consistent with the theory, increasing m (which translates to increasing the threshold ε) reduces the variance but potentially increases the bias of the gradient estimate. We conduct a grid search over the values $[4, 6, 8, 10, 16, 20]$ for m and report the results of tuning m for MNIST and Fashion MNIST. We select the value of m that experimentally provides the best trade-off between bias and variance. **MNIST Results:** We show in Fig 19, plots corresponding to the loss on the training loss, test loss, and the test accuracy per epoch. The results illustrate a consistent reduction in the variance of the gradient estimates as m increases. However, once m deviates from its best value, the bias of the gradient estimates increases, which results in higher values of the training loss as well as increase in the size of the prediction sets.

Fashion-MNIST: Similarly, tuning m on Fashion-MNIST shows that a value of $m = 6$ provides the best results, as depicted in Fig 20

C.6. Alternative Architecture

In this section, we compare the performance of `VR-ConfTr` on Kushuniji-MNIST using a simpler linear model architecture, different than the MLP used in the main paper. The results further reinforce that regardless of the model architecture, the trends observed in terms of convergence speed and prediction set efficiency remain consistent across datasets and architectures. Table 2 shows the average accuracy and set sizes for the two different model architectures trained on K-MNIST.

Dataset	Model Name	Accuracy (Avg ± Std)	Set Size (Avg ± Std)
K-MNIST (Linear)	Baseline	0.695 ± 0.007	6.799 ± 0.117
	ConfTr	0.582 ± 0.047	6.646 ± 0.226
	VR-ConfTr	0.612 ± 0.033	6.488 ± 0.148
K-MNIST (MLP)	Baseline	0.872 ± 0.046	4.982 ± 0.530
	ConfTr	0.783 ± 0.125	4.762 ± 0.226
	VR-ConfTr	0.835 ± 0.098	4.657 ± 0.680

Table 2. Evaluation results of the KMNIST dataset trained with different model architectures. Columns present average accuracy and set size with their standard deviations (**Avg ± Std**).

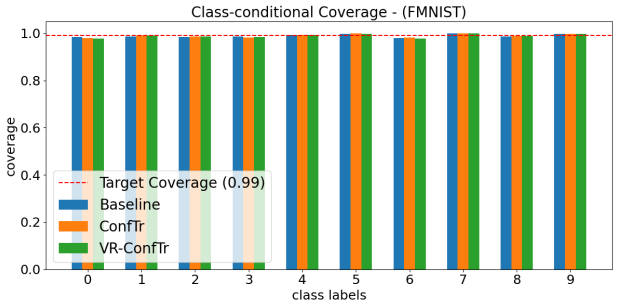

Figure 10. Class-Conditional Coverage (Fashion-MNIST)

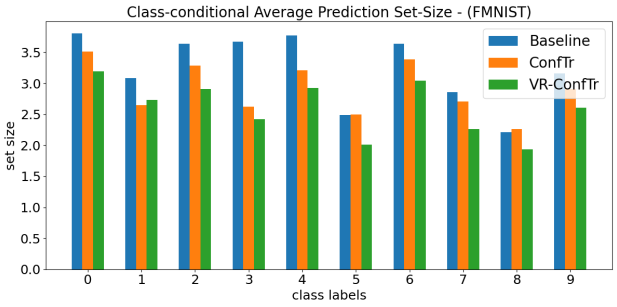

Figure 11. Class-Conditional Set Sizes (Fashion-MNIST)

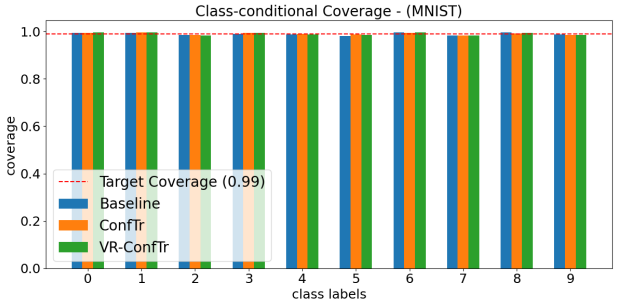

Figure 12. Class-Conditional Coverage (MNIST)

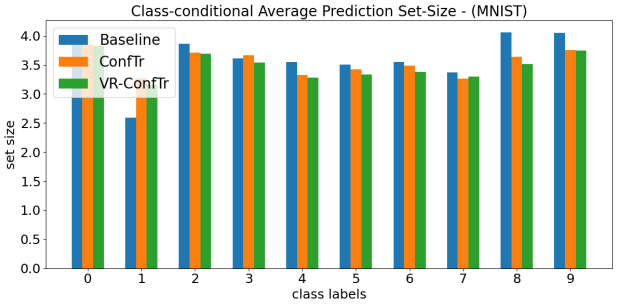

Figure 13. Class-Conditional Set Sizes (MNIST)

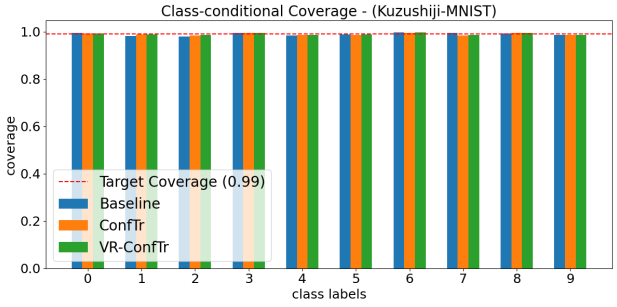

Figure 14. Class-Conditional Coverage (Kuzushiji-MNIST)

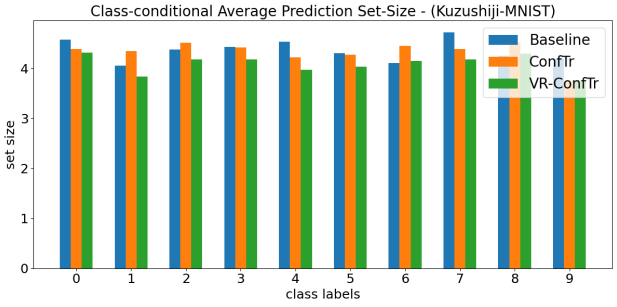

Figure 15. Class-Conditional Set Sizes (Kuzushiji-MNIST)

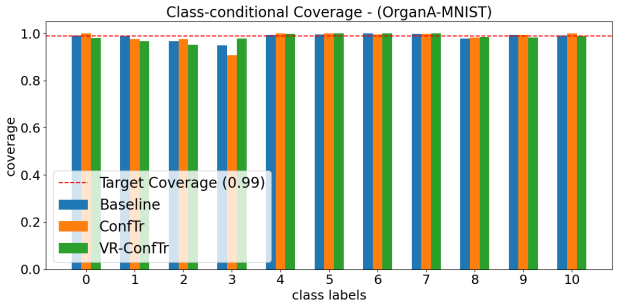

Figure 16. Class-Conditional Coverage (OrganA-MNIST)

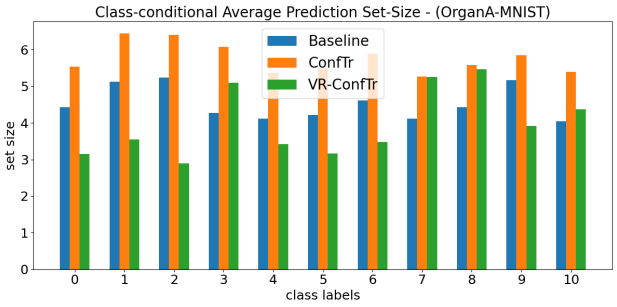

Figure 17. Class-Conditional Set Sizes (OrganA-MNIST)

Figure 18. Class-conditional coverage rates and average prediction set sizes for each dataset, reported over 10 randomized test trials. For each dataset, the left plot shows the class-conditional coverage rates with the target coverage level of $1 - \alpha = 0.99$ indicated by the horizontal red dashed line. The right plot shows the class-conditional average prediction set sizes.

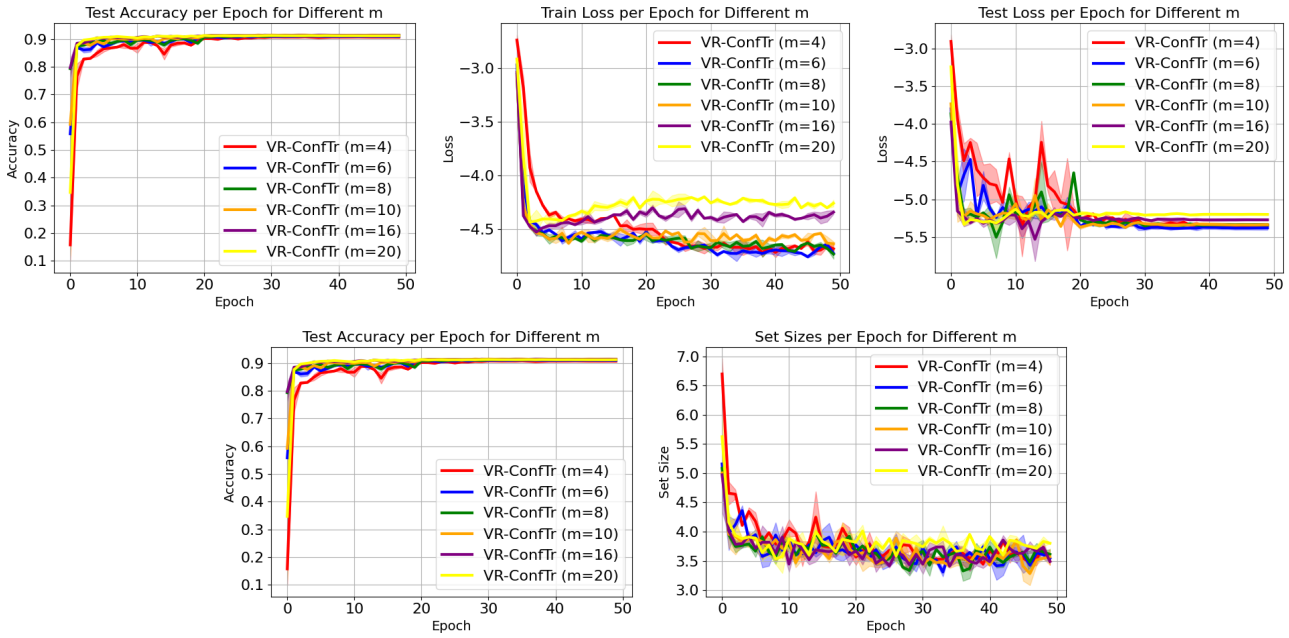

Figure 19. Training curves for different values of m on MNIST

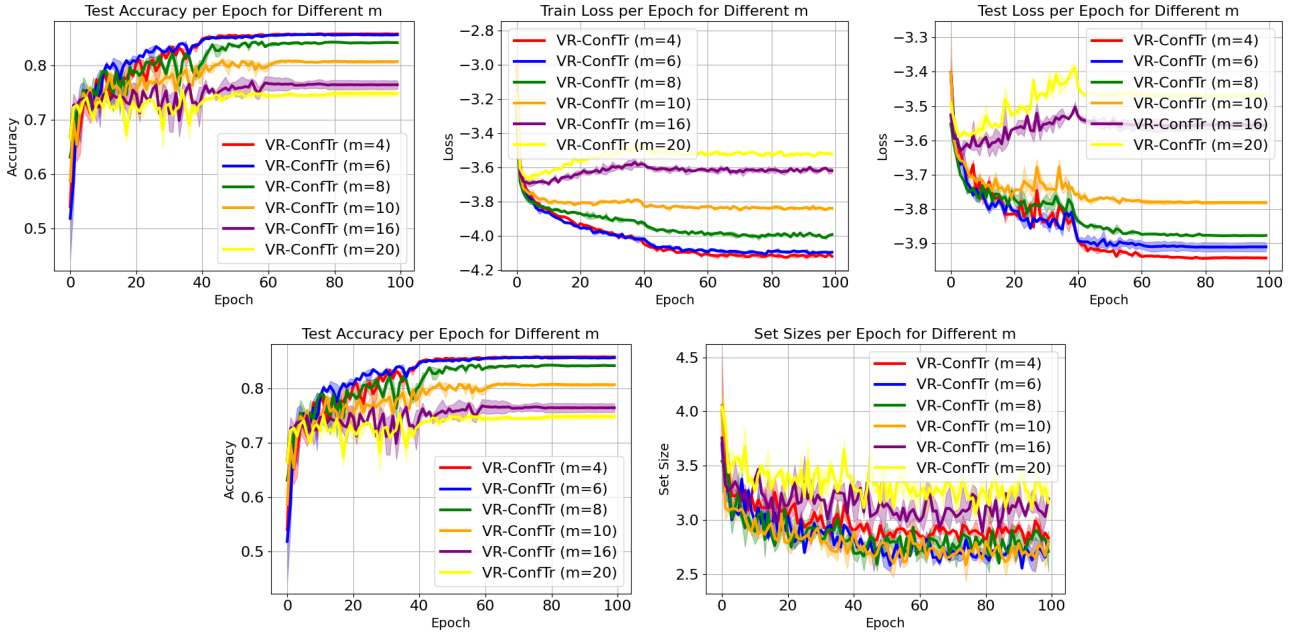

Figure 20. Learning curves for different values of m on Fashion-MNIST

D. Experimental Details

In this section we describe the experimental setup, including model architectures, dataset configurations, training protocol, testing procedure, and the corresponding hyper-parameters. The focus of the experiments is on evaluating the CP set sizes during training, convergence speed, and accuracy while ensuring a fair comparison between `ConfTr` and our proposed `VR-ConfTr`.

D.1. Dataset Configurations

We consider the benchmark datasets MNIST (LeCun et al., 1998), Fashion-MNIST (Xiao et al., 2017b), Kuzushiji-MNIST (Clanuwat et al., 2018) and OrganAMNIST (Yang et al., 2021) and CIFAR-10 (Krizhevsky, 2009). MNIST is a dataset of handwritten digits with 10 classes, and Fashion-MNIST consists of 10 fashion product categories. Kuzushiji-MNIST extends the MNIST paradigm by incorporating 10 classes of cursive Japanese characters. OrganAMNIST, derived from medical images, contains 11 classes of abdominal organ slices. CIFAR-10 is a dataset of natural images with 10 object categories. The training, calibration, and testing splits for each dataset are summarized in Table 3. MNIST and Fashion-MNIST are provided by the torchvision library, while Kuzushiji-MNIST and OrganAMNIST, and CIFAR-10 are available from their respective repositories. For MNIST, Fashion-MNIST, Kuzushiji-MNIST, and CIFAR-10 10% of the training set is reserved as calibration data. For OrganAMNIST, the validation set is used as the calibration data. During evaluation, we combine the calibration and test data and perform evaluations over 10 random splits of the combined dataset into calibration/test partitions. Model parameters are learned exclusively on the training data, while calibration and test data are used to evaluate the model as a black-box at the end of each epoch. The transformations applied to the dataset are as follows: for MNIST, Fashion-MNIST, and Kuzushiji-MNIST, images are normalized to have zero mean and unit variance, using a mean of 0.5 and a standard deviation of 0.5. For OrganAMNIST, images undergo random horizontal flips, random rotations of up to 15 degrees, and are normalized similarly. for CIFAR-10, we use random resizing, horizontal flips, and normalization as data augmentations.

Dataset	Classes	Image Size	Training Set	Calibration Set	Test Set
MNIST	10	28×28	55,000	5,000	10,000
Fashion-MNIST	10	28×28	55,000	5,000	10,000
OrganMNIST	11	28×28	34,561	6,491	17,778
Kuzushiji-MNIST	10	28×28	55,000	5,000	10,000
CIFAR-10	10	32×32	45,000	5,000	10,000

Table 3. Dataset Splits

D.2. Model Architectures

In our experiments, we implemented all models using JAX (Bradbury et al., 2018). We utilize a range of architectures including linear models, multi-layer perceptrons (MLPs), and modified ResNet architectures tailored for specific datasets. For the **MNIST** dataset, we employ a simple linear model, which consists of a single dense layer. The input images, reshaped from 28×28 into a flattened vector of size 784, are passed through a fully connected layer mapping the inputs directly to the 10 output classes. For **Fashion-MNIST**, we use a multi-layer perceptron (MLP), with two hidden layers. We use 64 units per hidden layer, with ReLU activations (Nair & Hinton, 2010) , followed by a dense layer for the 10 output classes. For **Kuzushiji-MNIST**, we utilize a similar MLP architecture. The model contains two hidden layers with 256 and 128 units, respectively. The input data is flattened and passed through these fully connected layers with ReLU activations. For **OrganAMNIST**, we used a residual network, inspired by the ResNet architecture from (He et al., 2016) , with modifications. The model consists of an initial convolutional layer followed by four stages of residual blocks, each with two layers. Each residual block uses 3×3 convolutions with ReLU activations. The number of output channels doubles after each state $(64, 128, 256, 512)$. Global average pooling is applied before the final fully connected layer, which maps the pooled feature representations to the 11 output classes. For **CIFAR-10**, we use a ResNet20 architecture, a lightweight version of ResNet (He et al., 2016) with 20 layers. Particularly, we use a pretrained ResNet20 model trained with cross-entropy loss. The last linear layer of the model is reinitialized and then fine-tuned using the `ConfTr` and `VR-ConfTr` algorithms. We do not attempt to optimize the model architectures in order to solve the datasets with high accuracy. Instead, we focus on the conformal prediction results, and ensure that the architecture used across different algorithms are identical for a fair comparison.

D.3. Training Details

Similar to (Stutz et al., 2022), we trained all models using Stochastic Gradient Descent (SGD) with Nesterov momentum (Sutskever et al., 2013). The learning rate follows a multi-step schedule where the initial learning rate was decreased by a factor of 0.1 after 2/5, 3/5, and 4/5 of the total number of epochs. The models were trained using cross-entropy-loss for `Baseline` training, and for `ConfTr` and `VR-ConfTr` based on the size-loss as described by (Stutz et al., 2022). During training, for MNIST, FMNIST, KMNIST, and OrganAMNIST we set the conformal prediction threshold parameter $\alpha = 0.01$. For fine-tuning CIFAR-10, we use $\alpha = 0.1$, and a weight decay term of 0.0005 for the optimizer. To ensure statistical robustness, we conducted multiple randomized training trials for each dataset, using a different random seed for each trial. Specifically, we performed 10 training trials for MNIST and 5 training trials each for FMNIST, KMNIST, OrganAMNIST, and CIFAR-10. The corresponding learning curves, i.e the training loss, testing loss, accuracy and CP set sizes evaluated on the test data at the end of every epoch, were averaged over these randomized trials to provide a smooth and general view of the model's performance.

The key hyper-parameters used for training are listed in Table 4. These hyper-parameters include **size weight** which scales the loss term associated with the size of the CP sets during training, **alpha** α corresponding to the miscoverage rate, **batch size** for SGD, **learning rate** for the optimizer, and the number of **epochs** for which the model is trained for. For the baseline method, we primarily optimized the model using Stochastic Gradient Descent (SGD) with Nesterov momentum. In cases where it led to improved performance, we employed the Adam optimizer instead. For `ConfTr`, the hyperparameters were obtained using a grid search over the following values: for batch size in $\{50, 100, 250, 300, 500\}$, learning rate in $\{0.005, 0.01, 0.05, 0.1, 0.5\}$, training epochs in $\{10, 50, 100, 150\}$, temperature in $\{0.1, 0.5, 1\}$, target set size in $\{0, 1\}$, size weight in $\{0.01, 0.05, 0.1, 0.5\}$, and the m-rank in $\{4, 6, 8, 10, 16, 20\}$. For `VR-ConfTr` the best reported hyperparameter for ConfTr were used. For the baseline we obtained the learning rate via grid search over $\{0.001, 0.005, 0.01, 0.05, 0.1, 0.5\}$ and the batch size via grid search over $\{32, 64, 128, 256\}$.

Hyper-parameter	**MNIST**	**Fashion-MNIST**	**Kuzushiji-MNIST**	**OrganA-MNIST**	**CIFAR-10**
Batch Size	500	500	500	500	500
Training Epochs	50	150	100	100	50
Learning Rate	0.05	0.01	0.01	0.01	0.01
Optimizer	SGD	SGD	SGD	SGD	SGD
Temperature	0.5	0.1	0.1	0.5	1
Target Set Size	1	0	1	1	0
Regularizer Weight	0.0005	0.0005	0.0005	0.0005	0.0005
Size Weight	0.01	0.01	0.01	0.1	0.05
α	0.01	0.01	0.01	0.01	0.1
m-rank	6	6	4	4	6

Table 4. Training and evaluation hyper-parameters for each dataset.

D.4. Evaluation Details

The evaluation of our models was conducted in two stages: (1) computing the test accuracy for each model after training, and (2) evaluating the conformal prediction (CP) set sizes and coverage over multiple test and calibration splits. **Test Accuracy:** For each dataset, the test accuracy of the trained models was evaluated on the test data, and the results were averaged over the randomized training trials. **CP set sizes** We first combine the holdout calibration and test data. We then randomly split this combined data into calibration and test portions, repeating the process 10 times. For each split, we apply the CP `THR` algorithm with α consistent with the value during training, and compute the CP set sizes on the test portion. The results are averaged across the 10 random splits. The cardinality of each split is consistent with the dataset configurations outlined in Table 3. This procedure is performed for each trained model, and the final reported results are averaged across both the training trials and testing splits.

D.5. Differences from ConfTr reports

We report the performance of `Conftr` with a batch size of 100 for Fashion-MNIST, as originally reported by (Stutz et al., 2022), selected for optimal performance. While a batch size of 500 yields smaller set sizes, it results in a slight (1%) decrease in accuracy. For completeness, we include the results for both configurations.

Model	Batch Size	Accuracy (Avg ± Std)	Set Size (Avg ± Std)
ConfTr	100	0.809 ± 0.051	3.125 ± 0.197
ConfTr	500	0.799 ± 0.065	3.048 ± 0.201
VR-ConfTr	500	0.839 ± 0.043	2.795 ± 0.154

Table 5. Final evaluation results for Fashion-MNIST, showing average accuracy and set size with their standard deviations (**Avg ± Std**).

Retrieving exact reported set sizes as (Stutz et al., 2022): Our experimental results and trends align with those reported in (Stutz et al., 2022). However, the smaller set sizes for `Conftr` on MNIST and FMNIST in their paper are likely due to their use of different architectures. Despite this, the overall trends— `Conftr` outperforming `Baseline`, and `VR-Conftr` outperforming `Conftr`—remain consistent regardless of the model. Our focus is on a fair comparison across algorithms by using the same architecture, rather than reproducing the exact figures or architectures from (Stutz et al., 2022).

E. On the Computational Complexity of `VR-ConfTr`.

We will now discuss the computational complexity of `VR-ConfTr` when compared to `ConfTr`. We will argue that the computational complexity of the two algorithms is essentially the same. We start by breaking down the computational cost of `ConfTr` and then illustrate the difference with `VR-ConfTr`.

Per-step computational complexity of `ConfTr`. Given a batch and partition $B = \{B_{\text{cal}}, B_{\text{pred}}\}$, with $|B_{\text{cal}}| = |B_{\text{pred}}| = n$, the first step of `ConfTr` is to compute a sample α quantile $\hat{\tau}(\theta)$ based on the calibration batch $B_{\text{cal}} = \{X_i^{\text{cal}}, Y_i^{\text{cal}}\}_{i=1}^n$, which requires the computation of the calibration batch conformity scores $\{E_\theta(X_i^{\text{cal}}, Y_i^{\text{cal}})\}_{i=1}^n$ and of their α-quantile. At this point, the computation of the `ConfTr` gradient is performed computing the gradient of the loss

$$\frac{1}{|B_{\text{pred}}|} \sum_{(x,y)\in B_{\text{pred}}} \ell(\theta, \hat{\tau}(\theta), x, y). \tag{56}$$

Note that for each sample (x, y), computing the `ConfTr` gradient implies computing the following (equation (5) in the main paper):

$$\frac{\partial}{\partial\theta}\left[\ell(\theta, \hat{\tau}(\theta), x, y)\right] = \frac{\partial\ell}{\partial\theta}(\theta, \hat{\tau}(\theta), x, y) + \frac{\partial\ell}{\partial\hat{\tau}}(\theta, \hat{\tau}(\theta), x, y) \cdot \frac{\partial\hat{\tau}}{\partial\theta}(\theta) \tag{57}$$

Note that computing this gradient requires computing (i) the gradients $\frac{\partial\ell}{\partial\theta}(\theta, \hat{\tau}(\theta), x, y)$ and $\frac{\partial\ell}{\partial\tau}(\theta, \hat{\tau}(\theta), x, y)$ for all samples $(x, y) \in B_{\text{cal}}$, and (ii) the gradient $\frac{\partial\hat{\tau}}{\partial\theta}(\theta)$. The difference in terms of computational complexity between `ConfTr` and our proposed `VR-ConfTr` lies in the computation of estimates of $\frac{\partial\tau}{\partial\theta}(\theta)$, which in `ConfTr` is done via computing the gradient of $\hat{\tau}(\theta)$, while in our algorithm is done plugging an improved estimate $\widehat{\frac{\partial\tau}{\partial\theta}}(\theta)$. We describe the computational difference between these two approaches in the next paragraph.

Per-step computational complexity of `VR-ConfTr`. Note that in our proposed algorithm `VR-ConfTr`, given a batch B defined as above, we consider the same per-step loss function of `ConfTr` of equation (56). However, instead of computing directly the gradient of (56), we compute separately an estimate $\widehat{\frac{\partial\tau}{\partial\theta}}(\theta)$ of $\frac{\partial\tau}{\partial\theta}(\theta)$ using our novel estimation technique and then plug this estimate in equation (57) in place of $\frac{\partial\hat{\tau}}{\partial\theta}(\theta)$. In the proposed estimator, computing $\widehat{\frac{\partial\tau}{\partial\theta}}(\theta)$ equals computing gradients $\{\frac{\partial E}{\partial\theta}(\theta, x, y)\}_{(x,y)\in\bar{B}}$, where $\bar{B}$ is the set containing the m samples whose conformity scores fall within ϵ distance from the sample quantile $\hat{\tau}(\theta)$. Note that, computationally, our algorithm requires computing $\frac{\partial\ell}{\partial\theta}(\theta, \hat{\tau}(\theta), x, y)$ and $\frac{\partial\ell}{\partial\tau}(\theta, \hat{\tau}(\theta), x, y)$, which is the same as `ConfTr`, while we do not need to compute the gradient $\frac{\partial\hat{\tau}}{\partial\theta}(\theta)$. Instead, we replace the computation of the gradient of $\hat{\tau}(\theta)$ with the computation of an average of m gradients of conformity scores. In conclusion, the main computational difference between `ConfTr` and `VR-ConfTr` is in the computation of the estimate of $\frac{\partial\tau}{\partial\theta}(\theta)$, which for both of the techniques boils down to computing and averaging a certain set of conformity scores. This is why we can safely conclude that the computational complexity of the two algorithms is essentially the same.

F. Extended Related Works

Variance Reduction There is a large body of work on variance reduction techniques for stochastic optimization problems in standard machine learning setting ((Johnson & Zhang, 2013; Defazio et al., 2014; Shalev-Shwartz & Zhang, 2013; Nguyen et al., 2017)). In particular, SVRG ((Johnson & Zhang, 2013)) is based on the idea of increasing the number of samples used in the estimate of the gradient to reduce variance, which is done by periodically computing a full gradient using the full dataset to perform the updates. Another well known approach SAGA ((Defazio et al., 2014)), maintains a memory of past gradients to again increase the number of samples used at each iteration to reduce the variance of the updates. For both SVRG and SAGA, they achieve variance reduction by increasing the number of samples used at each iteration, increasing the computational effort, and increasing memory usage. In contrast, the conformal training `ConfTr` objective presents a variance reduction challenge that is not seen in standard objectives. `ConfTr` (Stutz et al., 2022) simulates conformal prediction in the loop, which in turn requires embedding a quantile estimation into the training loss. This means that each gradient update involves differentiating through an empirical quantile of a sampled distribution. The key issue that arises is that the variance of the quantile-based stochastic gradient does not diminish with larger sample sizes. In other words, unlike an ordinary SGD where averaging more samples yields a lower variance gradient estimate, increasing the batch size in `ConfTr` does not proportionally reduce the noise in the gradient. Our proposed `VR-ConfTr` is based on plugging an improved estimate of a specific quantity - the population quantile gradient $\frac{\partial \tau}{\partial \theta}(\theta)$- in the conformal training cost function gradient, without requiring to increase the batch size, nor the memory, and it has essentially the same computation cost of the existing `ConfTr` algorithm.