Improving SAM Requires Rethinking its Optimization Formulation

Wanyun Xie [1] **Fabian Latorre** [1] **Kimon Antonakopoulos** [1] **Thomas Pethick** [1] **Volkan Cevher** [1]

Abstract

This paper rethinks Sharpness-Aware Minimization (SAM), which is originally formulated as a zero-sum game where the weights of a network and a bounded perturbation try to minimize/maximize, respectively, the same differentiable loss. To fundamentally improve this design, we argue that SAM should instead be reformulated using the 0-1 loss. As a continuous relaxation, we follow the simple conventional approach where the minimizing (maximizing) player uses an upper bound (lower bound) surrogate to the 0-1 loss. This leads to a novel formulation of SAM as a bilevel optimization problem, dubbed as BiSAM. BiSAM with newly designed lower-bound surrogate loss indeed constructs stronger perturbation. Through numerical evidence, we show that BiSAM consistently results in improved performance when compared to the original SAM and variants, while enjoying similar computational complexity. Our code is available at https://github.com/LIONS-EPFL/BiSAM.

1. Introduction

The rise in popularity of deep neural networks has motivated the question of which optimization methods are better suited for their training. Recently, it has been found that *Sharpness-Aware Minimization (SAM)* (Foret et al., 2021) can greatly improve their generalization with almost negligible increase in computational complexity. SAM not only benefits supervised learning tasks from computer vision greatly (Foret et al., 2021; Dosovitskiy et al., 2020) but also improves the performance of large language models (Bahri et al., 2021; Zhong et al., 2022). Hence, it is natural to ask whether SAM can be improved further. Indeed, many works have been quick to present modifications of the original SAM algorithm, that improve its speed (Du et al., 2022) or performance (Kwon et al., 2021) in practice.

The motivation behind SAM is to find a parameter $w^\star$ in the so-called *loss landscape*, that achieves a low *loss value* while being *flat* i.e., the loss in its immediate neighborhood should not deviate meaningfully from the value attained at $w^\star$. To seek the flat minima, SAM algorithm derived by Foret et al. (2021) is formulated as a two-player zero-sum game, where two players respectively seek to minimize and maximize the same differential loss.

In supervised classification tasks, common differential losses like cross-entropy provide an upper bound to misclassification error, making it reasonable to expect that minimizing these losses will lead to a reduction in the 0-1 loss for the minimizer. However, can we do such a natural replacement in the SAM formulation? A key question is *"Does maximizing such surrogate loss lead to a maximum of the 0-1 loss?"* No, there is no guarantee. We make this precise with a counterexample in Section 2.2.

Therefore, we argue that even though the surrogate loss used in SAM formulation appears beneficial, we should recall that the goal in supervised classification is not to achieve a low value of the cross-entropy, rather, the goal is to enjoy a small *misclassification error rate* on the testing set. This raises the question:

If our goal is to achieve a better classifier, should we not apply the SAM formulation directly on the misclassification error i.e., the 0-1 loss?

Motivated by this actual goal, we argue that the original SAM with surrogate loss suffers from a fundamental flaw that leads to a weaker perturbation. Our analysis reveals that maximizing a surrogate loss like cross-entropy has no guarantee that misclassification errors will increase due to its upper-bound nature. To overcome this shortcoming, the minimizer and maximizer should have different objectives, specifically an upper bound of misclassification error for the minimizer while a lower bound for the maximizer. This guides us to propose a novel bilevel formulation of SAM, called BiSAM, with a new loss function for the maximizer. BiSAM fixes SAM's fundamental issue without any extra computational cost and importantly it can be incorporated

[1]Laboratory for Information and Inference Systems, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland. Correspondence to: Wanyun Xie <wanyun.xie@epfl.ch>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

in almost all existing variants of SAM.

We summarize **our contributions** as follows:

- We present a novel bilevel optimization formulation of SAM, where instead of solving a min-max zero-sum game between the model parameters w and the perturbation ϵ, each player has a different objective. This formulation appears naturally by applying SAM to the relevant performance metric in supervised learning: the misclassification error, i.e., the so-called 0-1 loss.
- We propose *BiSAM* (Algorithm 1), a scalable first-order optimization method to solve our proposed bilevel formulation of SAM. BiSAM is simple to implement and enjoys a similar computational complexity when compared to SAM.
- We present numerical evidence on CIFAR10/CIFAR100 showing that BiSAM consistently outperforms SAM across five models, and also see improvement on ImageNet-1K. BiSAM incorporating variants of SAM (ASAM and ESAM) also demonstrates enhancement. We additionally verify that our reformulation remains robust in fine-tuning both image and text classification tasks and noisy label tasks.

2. Preliminaries and Problem Setup

Notation. $f_w : \mathbb{R}^d \to \mathbb{R}^K$ corresponds to the logits (scores) that are output by a neural network with parameters w. For a given *loss function* ℓ, we denote the *training set loss* $L(w) = \frac{1}{n}\sum_{i=1}^{n} \ell(f_w(x_i), y_i)$. We denote the corresponding training set losses of cross entropy loss and 0-1 loss as L^{ce} and L^{01}, respectively. We denote as $\mathbb{I}\{A\}$ the indicator function of an event i.e., $\mathbb{I}\{A\} = 1$ if A is true or $\mathbb{I}\{A\} = 0$ if A is false.

2.1. Preliminaries: Surrogate loss

Considering a classification task, the goal is to obtain a classifier predicting the label y correctly from data x. Then, the problem is

$$\min_w L^{01}(w) = \min_w \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}\left\{\arg\max_{j=1,\dots,K} f_w(x)_j \neq y\right\} \quad (1)$$

The difficulty of solving Equation (1) is the fact that L^{01} is discontinuous, which hinders first-order optimization methods. Normally, it can be solved by replacing L^{01} by a surrogate loss. For minimization problems, a differential upper bound of misclassification error can be used, i.e.

$$L^{01}(w) \leq L(w) \quad (2)$$

Common surrogate losses are cross-entropy loss and hinge loss. It is important to note that this inequality Equation (2) guarantees that minimizing $L(w)$ provides a solution that decreases L^{01}, which is the real goal in supervised classification tasks.

2.2. SAM and its limitation

SAM (Foret et al., 2021) and its variants (Kwon et al., 2021; Du et al., 2022) search for a flat minima by solving the following minimax optimization problem:

$$\min_w \max_{\epsilon:\|\epsilon\|_2\leq\rho} L(w+\epsilon) \quad (3)$$

where ρ is a small constant controlling the radius of a neighborhood, and a surrogate upper-bound loss is usually used as the objective function following the minimization problem. SAM addresses this minimax problem by applying a two-step procedure at iteration t:

$$\begin{aligned} \epsilon_t &= \rho\frac{\nabla L(w_t)}{\|\nabla L(w_t)\|_2} \approx \arg\max_{\epsilon:\|\epsilon\|_2\leq\rho} L(w_t+\epsilon) \\ w_{t+1} &= w_t - \eta_t \nabla L(w_t+\epsilon_t) \end{aligned} \quad (4)$$

where the perturbation ϵ_t is obtained as the closed-form solution to a first-order approximation.

Although this approach is pervasive in practice, we argue that the surrogate loss for the inner maximization problem has a fundamental limit that will lead to **weaker perturbation**. To be specific, the maximizer in Equation (4) maximizes an *upper bound* on the classification error. This means that any $\epsilon^\star$ obtained by Equation (4) has no guarantee to increase the classification error. To make our argument more convincing, we give an example.

Example. This example is to illustrate explicitly that maximizing the upper bound of 0-1 loss leads to the *wrong* result. Consider two possible vectors of logits:

- "Option A": $(1/K+\delta, 1/K-\delta, \dots, 1/K)$
- "Option B": $(0.5-\delta, 0.5+\delta, 0, 0, \dots, 0)$

where δ is a small positive number, and assume the first class is the correct one, so we compute the cross-entropy with respect to the vector $(1, 0, \dots, 0)$.

For "Option A", the cross-entropy is $-\log(1/K+\delta)$, which tends to infinity as $K \to \infty$ and $\delta \to 0$. For "Option B", the cross-entropy is $-\log(0.5-\delta)$, which is a small constant number. Hence, an adversary that maximizes an upper bound like the cross-entropy, would always choose "Option A". However, note that "Option A" *never* leads to a maximizer of the 0-1 loss, since the predicted class is the correct one (zero loss). In contrast, "Option B" always achieves the maximum of the 0-1 loss (loss is equal to one), even if it has low (i.e., constant) cross-entropy. This illustrates why

maximizing an upper bound like cross-entropy provides a possibly weak weight perturbation.

The misalignment observed above suggests that we should take one step back and rethink SAM on L^{01} directly, as it directly reflects the accuracy of interest:

$$\min_{w} \max_{\epsilon:\|\epsilon\|_2\leq\rho} L^{01}(w+\epsilon) \tag{5}$$

Handling the discontinuous nature of L^{01} to fit the first-order optimization methods, we decouple Equation (5) into a bilevel formulation and treat the maximizer and the minimizer separately in Section 3.

3. Bilevel Sharpness-aware Minimization (BiSAM)

In order to obtain a differentiable objectives for the minimization and maximization players in the formulation Equation (5), our starting point is to decouple the problem as follows:

$$\begin{aligned} &\min_{w} L^{01}(w+\epsilon^\star), \\ &\text{subject to } \epsilon^\star \in \arg\max_{\epsilon:\|\epsilon\|_2\leq\rho} L^{01}(w+\epsilon) \end{aligned} \tag{6}$$

Up to this point, there has been no modification of the original objective Equation (5). We first note that for the minimization player w, we can minimize a differentiable upper bound (e.g., cross-entropy) instead of the 0-1 loss, leading to the formulation:

$$\begin{aligned} &\min_{w} L^{\text{ce}}(w+\epsilon^\star), \\ &\text{subject to } \epsilon^\star \in \arg\max_{\epsilon:\|\epsilon\|_2\leq\rho} L^{01}(w+\epsilon) \end{aligned} \tag{7}$$

Now, we only need to deal with replacing the 0-1 loss in the objective of the perturbation ϵ. Because this corresponds to a maximization problem, we need to derive a *lower bound*.

Recall the example. We maximize a lower bound instead of the cross-entropy in the example in Section 2.2. Consider a lower-bound loss like $\max_{j\in[K]} \tanh(z_j - z_y)$ where z is the vector of logits (see (9) regarding the construction). This loss for "option A" is 0 and for "option B" it is $\tanh(2\delta) > 0$. Thus, an adversary maximizing this loss would choose "option B", correctly leading to maximization of the 0-1 loss.

Lemma 3.1. *Let $\phi(x)$ be a lower bound of the 0-1 step function $\{x > 0\}$. For each $j \in [K]$, let $F_{w+\epsilon}(x_i, y_i)_j = f_{w+\epsilon}(x_i)_j - f_{w+\epsilon}(x_i)_{y_i}$ and let $\mu > 0$. It holds that*

$$\begin{aligned} L^{01}(w+\epsilon) \geq &\frac{1}{n}\sum_{i=1}^{n} \frac{1}{\mu} \log\left(\sum_{j=1}^{K} e^{\mu\phi(F_{w+\epsilon}(x_i,y_i)_j)}\right) \\ &- \frac{1}{\mu}\log(K) \end{aligned} \tag{8}$$

Remark 3.2. Choice $\phi(x) = \tanh(x)$ is a valid lower bound (see (14) for further discussion).

Proof. Note that for a training sample (x_i, y_i) we have misclassification error if and only if for some class $j \neq y_i$ the score assigned to class j is larger than the score assigned to y_i. Equivalently, $\arg\max_{j\in[K]} f_{w+\epsilon}(x_i)_j \neq y_i$ if and only if $\max_{j=1,\dots K} F_{w+\epsilon}(x_i, y_i)_j > 0$. Thus,

$$\begin{aligned} L^{01}(w+\epsilon) &= \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}\left\{\arg\max_{j\in[K]} f_{w+\epsilon}(x_i)_j \neq y_i\right\} \\ &= \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}\left\{\max_{j\in[K]} F_{w+\epsilon}(x_i,y_i)_j > 0\right\} \\ &= \frac{1}{n}\sum_{i=1}^{n} \max_{j\in[K]} \mathbb{I}\left\{F_{w+\epsilon}(x_i,y_i)_j > 0\right\} \\ &\geq \frac{1}{n}\sum_{i=1}^{n} \max_{j\in[K]} \phi\left(F_{w+\epsilon}(x_i,y_i)_j\right) \end{aligned} \tag{9}$$

Although this lower bound can be directly used, it is computationally consuming due to the need for K perturbation calculations per update. To get rid of the non-differentiable maximum operator over the set $[K] = \{1, \dots, K\}$, we use the well-known bounds of the log-sum-exp function:

$$\frac{1}{\mu}\log\left(\sum_{i=1}^{K} e^{\mu a_i}\right) \leq \max\{a_1,\dots,a_K\} + \frac{1}{\mu}\log(K) \tag{10}$$

Using Equation (10) in Equation (9) yields the desired bound. □

As a consequence of Lemma 3.1 we conclude that a valid approach for the maximization player is to solve the differentiable problem in the right-hand-side of the following inequality:

$$\begin{aligned} &\max_{\epsilon:\|\epsilon\|_2\leq\rho} L^{01}(w+\epsilon) + \frac{1}{\mu}\log(K) \\ &\geq \max_{\epsilon:\|\epsilon\|_2\leq\rho} \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\mu}\log\left(\sum_{j=1}^{K} e^{\mu\phi(F_{w+\epsilon}(x_i,y_i)_j)}\right) \\ &=: \max_{\epsilon:\|\epsilon\|_2\leq\rho} Q_{\phi,\mu}(w+\epsilon) \end{aligned} \tag{11}$$

Continuing from Equation (7), we finally arrive at a bilevel and fully differentiable formulation:

$$\begin{aligned} &\min_{w} L^{\text{ce}}(w+\epsilon^\star), \\ &\text{subject to } \epsilon^\star \in \arg\max_{\epsilon:\|\epsilon\|_2\leq\rho} Q_{\phi,\mu}(w+\epsilon) \end{aligned} \tag{12}$$

Note that both upper and lower objective functions in Equation (12) are aligned with the ultimate goal of the 0-1 loss.

Algorithm 1 Bilevel SAM (BiSAM)

Input: Initialization $w_0 \in \mathbb{R}^d$, iterations T, batch size b, step sizes $\{\eta_t\}_{t=0}^{T-1}$, neighborhood size $\rho > 0$, $\mu > 0$, lower bound ϕ.

for $t = 0$ ***to*** $T - 1$ **do**
- Sample minibatch $\mathcal{B} = \{(x_1, y_1), \ldots, (x_b, y_b)\}$.
- Compute the (stochastic) gradient of the perturbation loss $Q_{\phi,\mu}(w_t)$ defined in Equation (11)
- Compute perturbation $\epsilon_t = \rho \frac{\nabla_w Q(w)}{\|\nabla_w Q(w)\|}$.
- Compute gradient $g_t = \nabla_w L_{\mathcal{B}}(w_t + \epsilon_t)$.
- Update weights $w_{t+1} = w_t - \eta_t g_t$.

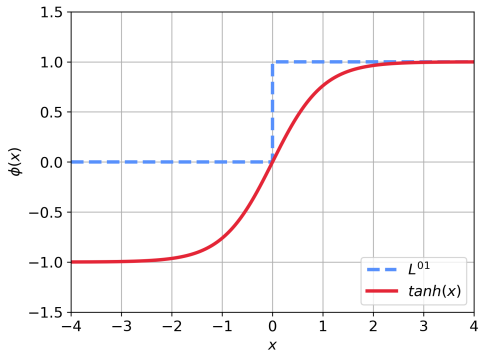
Specifically, the bound in the lower level problem is tight up to a small additive logarithmic term of $\frac{1}{\mu}\log(K)$ and it holds everywhere.

The above setting forces us to take a slight detour to the general framework of bilevel optimization and its solution concepts. In particular, the nested nature of the problem makes its solution to be notoriously difficult. Therefore, the success of the up-to-date iterative methods relies on a set of quite restrictive assumptions, which do not apply in the complex environment of neural networks (we refer the reader to Ghadimi and Wang (2018); Tarzanagh and Balzano (2022) for more details). In particular, an important feature that needs to be satisfied is that the so-called inner problem should be strongly convex; which here is clearly not the case. Therefore, in order to devise a fast algorithm for the problem in the right-hand-side of (12), some particular modifications should be made. More precisely, we follow the same approach in the original SAM algorithm (Foret et al., 2021), and do a first-order Taylor expansion of $Q_{\phi,\mu}(w + \epsilon)$ with respect to ϵ around 0. We obtain:

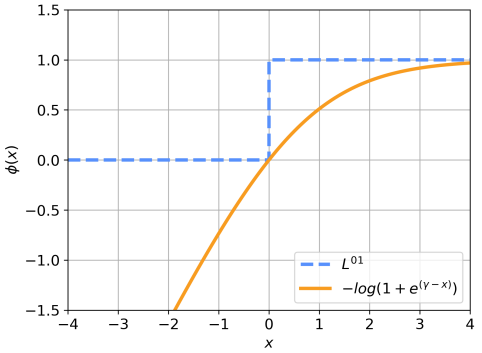
$$\begin{aligned} \epsilon^\star &\in \arg\max_{\epsilon:\|\epsilon\|_2 \leq \rho} Q_{\phi,\mu}(w + \epsilon) \\ &\approx \arg\max_{\epsilon:\|\epsilon\|_2 \leq \rho} Q_{\phi,\mu}(w) + \epsilon^\top \nabla_w Q(w) \\ &= \arg\max_{\epsilon:\|\epsilon\|_2 \leq \rho} \epsilon^\top \nabla_w Q(w) = \rho \frac{\nabla_w Q(w)}{\|\nabla_w Q(w)\|_2} \end{aligned} \tag{13}$$

As the function $Q_{\phi,\mu}$ involves a sum over the whole dataset, this makes the computation of the full gradient in Equation (13) too costly. For scalability, we use stochastic gradients defined on a mini-batch in practice. Our proposed algorithm to solve SAM in the bilevel optimization paradigm (BiSAM), finally takes shape as shown in Algorithm 1.

On the choice of lower bound ϕ. The function ϕ plays a crucial role in the objective $Q_{\phi,\mu}$ that defines the perturbation Equation (13). Although in theory we can use any lower bound for the 0-1 step function $\{x > 0\}$, the choice can affect the performance of the optimization algorithm. As is always the case in Deep Learning, one should be on the look for possible sources of *vanishing/exploding gradients* (Hochreiter et al., 2001).



(a) $\phi(x) = \tanh(\alpha x)$



(b) $\phi(x) = -\log(1 + e^{(\gamma - x)}) + 1$

Figure 1. Plot of suggested lower bounds.

As shown in Figure 1, the function

$$\phi(x) = \tanh(\alpha x) \tag{14}$$

seems to be a good lower bound of the 0-1 step function. However, at all points far from zero, the gradient quickly vanishes, which might harm performance. We suggest considering the alternative:

$$\phi(x) = -\log(1 + e^{(\gamma - x)}) + 1 \tag{15}$$

where $\gamma = \log(e - 1)$, also shown in Figure 1, as it only suffers from a vanishing gradient on large positive values. However, note that having a vanishing gradient in such a region is not really an issue: the objective of the perturbation ϵ is to move points towards the lower side of the plot in Figure 1, where misclassification happens. Hence, if a point stays there due to the vanishing gradient problem, it means it will remain misclassified. In contrast, having vanishing gradients on the top side of the plot in Figure 1 might mean that the optimization algorithm is unable to move points that

are correctly classified towards the misclassification region, therefore the adversary would fail.

Both Equation (14) and Equation (15) used in $Q_{\phi,\mu}(w)$ serve as valid lower bounds in implementation. Figure 2 shows the number of misclassified samples under perturbation that the model predicts correctly while turning wrong after adding the weight perturbation. Note that its detailed setting is in Section 4.1. It indicates that SAM indeed exhibits weaker perturbation compared to our proposed BiSAM.

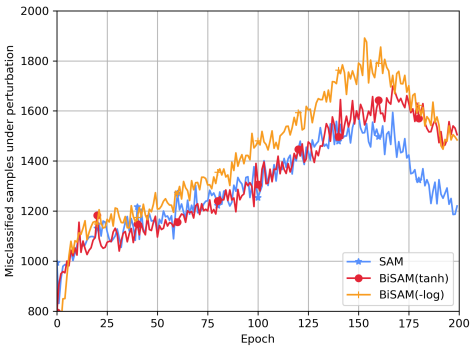


Figure 2. Number of misclassified samples under perturbation of Resnet56 on CIFAR-10.

4. Experiments

In this section, we verify the benefit of BiSAM across a variety of models, datasets and tasks.

4.1. Image classification

We follow the experimental setup of Kwon et al. (2021). We use the CIFAR-10 and CIFAR-100 datasets (Krizhevsky et al., 2009), both consisting of 50 000 training images of size 32×32, with 10 and 100 classes, respectively. For data augmentation we apply the commonly used random cropping after padding with 4 pixels, horizontal flipping, and normalization using the statistics of the training distribution at both train and test time. We train multiple variants of VGG (Simonyan and Zisserman, 2014), ResNet (He et al., 2016), DenseNet (Huang et al., 2017) and WideResNet (Zagoruyko and Komodakis, 2016) (see Tables 1 and 2 for details) using cross entropy loss. All experiments are conducted on an NVIDIA A100 GPU.

Two variants of BiSAM are compared against two baselines.

SGD: Standard training using stochastic gradient descent (SGD) (see details below)

SAM: The original Sharpness-Aware Minimization (SAM) algorithm from Foret et al. (2021)

BiSAM (tanh): Algorithm 1 using (14) as the lower bound

BiSAM (-log): Algorithm 1 using (15) as the lower bound

The models are trained using stochastic gradient descent (SGD) with a momentum of 0.9 and a weight decay of 5×10^{-4}. We used a batch size of 128, and a cosine learning rate schedule that starts at 0.1. The number of epochs is set to 200 for SAM and BiSAM while SGD are given 400 epochs. This is done in order to provide a computational fair comparison as (Bi)SAM uses twice as many gradient computation. Label smoothing with a factor 0.1 is employed for all method. For the SAM and BiSAM hyperparameter ρ we use a value of 0.05. We fix $\mu = 10$ and $\alpha = 0.1$ for BiSAM (tanh) and $\mu = 1$ for BiSAM (-log) throughout *all* experiments on both CIFAR-10 and CIFAR-100 datasets as a result of a grid search over $\{0.01, 0.1, 1, 10, 100\}$ for α and over $\{0.1, 0.5, 1, 2, 4\}$ for μ using the validation dataset on CIFAR-10 with Resnet-56.

The training data is randomly partitioned into a training set and validation set consisting of 90% and 10%, respectively. We deviate from Foret et al. (2021); Kwon et al. (2021) by using the *validation* set to select the model on which we report the *test* accuracy in order to avoid overfitting on the test set. We report the test accuracy of the model with the highest validation accuracy across the training with mean and standard deviations computed over 6 independent executions. The results can be found in Tables 1 and 2. Compared to the accuracy increase from SGD to SAM, the

Table 1. **Test accuracies on CIFAR-10.** BiSAM (-log) has strictly better performance than SAM across *all* models. We include BiSAM (tanh) for completeness which sometimes performs better than BiSAM (-log).

Model	SGD	SAM	BiSAM (-log)	BiSAM (tanh)
DenseNet-121	$96.14_{\pm 0.09}$	$96.52_{\pm 0.10}$	$\mathbf{96.61}_{\pm 0.17}$	$96.63_{\pm 0.21}$
Resnet-56	$94.01_{\pm 0.26}$	$94.09_{\pm 0.26}$	$\mathbf{94.28}_{\pm 0.31}$	$94.87_{\pm 0.34}$
VGG19-BN	$94.76_{\pm 0.10}$	$95.09_{\pm 0.12}$	$\mathbf{95.22}_{\pm 0.13}$	$95.01_{\pm 0.06}$
WRN-28-2	$95.71_{\pm 0.19}$	$96.00_{\pm 0.10}$	$\mathbf{96.02}_{\pm 0.12}$	$95.99_{\pm 0.09}$
WRN-28-10	$96.77_{\pm 0.21}$	$97.18_{\pm 0.04}$	$\mathbf{97.26}_{\pm 0.10}$	$97.17_{\pm 0.05}$
Average	$95.48_{\pm 0.08}$	$95.78_{\pm 0.06}$	$\mathbf{95.88}_{\pm 0.08}$	$95.93_{\pm 0.08}$

Table 2. **Test accuracies on CIFAR-100.** BiSAM (-log) consistently improves over SAM across all models. We include BiSAM (tanh) for completeness which sometimes performs better than BiSAM (-log).

Model	SGD	SAM	BiSAM (-log)	BiSAM (tanh)
DenseNet-121	$81.31_{\pm 0.38}$	$82.31_{\pm 0.15}$	$\mathbf{82.49}_{\pm 0.14}$	$82.88_{\pm 0.42}$
Resnet-56	$73.98_{\pm 0.16}$	$74.38_{\pm 0.37}$	$\mathbf{74.67}_{\pm 0.15}$	$74.54_{\pm 0.35}$
VGG19-BN	$74.90_{\pm 0.30}$	$74.94_{\pm 0.12}$	$\mathbf{75.25}_{\pm 0.24}$	$75.12_{\pm 0.34}$
WRN-28-2	$77.95_{\pm 0.14}$	$78.09_{\pm 0.13}$	$\mathbf{78.21}_{\pm 0.23}$	$78.07_{\pm 0.13}$
WRN-28-10	$81.50_{\pm 0.48}$	$82.89_{\pm 0.47}$	$\mathbf{83.27}_{\pm 0.26}$	$83.35_{\pm 0.25}$
Average	$77.93_{\pm 0.14}$	$78.52_{\pm 0.13}$	$\mathbf{78.78}_{\pm 0.09}$	$78.79_{\pm 0.14}$

average improvement from BiSAM to SAM reaches 33.3% on CIFAR-10 and 44.1% on CIFAR-100.

For evaluations at a larger scale, we compare the performance of SAM and BiSAM on ImageNet-1K (Russakovsky et al., 2015). We apply each method with $\rho = 0.05$ for both SAM and BiSAM. We use training epochs 90, peak learning rate 0.2, and batch size 512. We employ mSAM (Foret et al., 2021; Behdin et al., 2023) with micro batch size $m = 128$ to accelerate training and improve performance. We set $\mu = 5$ for BiSAM (-log) and $\mu = 20$ and $\alpha = 0.1$ for BiSAM (tanh). Other parameters are the same as CIFAR-10 and CIFAR-100. We run 3 independent experiments for each method and results are shown in Table 3. Note that we do not reproduce experiments of SGD on ImageNet-1K due to computational restriction but it well-documented that SAM and its variants have better performance than SGD (Foret et al., 2021; Kwon et al., 2021; Du et al., 2022).

Table 3. Test accuracies on ImageNet-1K.

	SAM	BiSAM (-log)	BiSAM (tanh)
Top1	$75.83_{\pm 0.16}$	$\mathbf{75.96}_{\pm 0.15}$	$76.02_{\pm 0.08}$
Top5	$92.47_{\pm 0.02}$	$\mathbf{92.49}_{\pm 0.10}$	$92.40_{\pm 0.13}$

We find that BiSAM (-log) *consistently* outperforms SGD and SAM across all models on both CIFAR-10 and CIFAR-100, and it also outperforms SAM on ImageNet-1K. In most cases, BiSAM (tanh) has better or almost same performance than SAM. Average accuracies across 5 models of both BiSAM (-log) and BiSAM (tanh) outperform SAM and the result is statistically significant as shown by the small standard deviation when aggregated over all model types. This improvement is achieved *without* modifying the original experimental setup and the hyperparameter involved. Specifically, we use the same $\rho = 0.05$ for BiSAM which has originally been tuned for SAM. The consistent improvement using BiSAM, despite the favorable setting for SAM, shows the benefit of our reformulation based on the 0-1 loss. Note that the generalization improvement provided by BiSAM comes at essentially *no computational overhead* (see Appendix B for detailed discussion).

We recommend using **BiSAM (-log)** as it generally achieves better or comparable test accuracies to BiSAM (tanh). Therefore, we choose BiSAM (-log) as representative while BiSAM (tanh) serves as reference in all tables.

4.2. Incorporation with variants of SAM

Since we just reformulate the perturbation loss of the original SAM, existing variants of SAM can be incorporated within BiSAM. The mSAM variant has been combined with BiSAM in experiments on ImageNet-1K. Moreover, we incorporate BiSAM with both Adaptive SAM (Kwon et al., 2021) and Efficient SAM (Du et al., 2022).

Adaptive BiSAM. We combine BiSAM with Adaptive Sharpness in ASAM (Kwon et al., 2021) which proposes a normalization operator to realize adaptive sharpness. The Adaptive BiSAM (A-BiSAM) algorithm is specified in detail in Appendix A.1 and results on CIFAR-10 are shown in Table 4. A-BiSAM (-log) *consistently* outperforms ASAM across *all* models on CIFAR-10 except for one on DenseNet-121 where the accuracy is the same.

Table 4. Test accuracies of A-(Bi)SAM.

Model	ASAM	A-BiSAM (-log)
DenseNet-121	$\mathbf{96.79}_{\pm 0.14}$	$\mathbf{96.79}_{\pm 0.13}$
Resnet-56	$94.86_{\pm 0.18}$	$\mathbf{95.09}_{\pm 0.09}$
VGG19-BN	$95.10_{\pm 0.09}$	$\mathbf{95.14}_{\pm 0.14}$
WRN-28-2	$96.22_{\pm 0.10}$	$\mathbf{96.28}_{\pm 0.14}$
WRN-28-10	$97.37_{\pm 0.07}$	$\mathbf{97.42}_{\pm 0.09}$
Average	$96.07_{\pm 0.05}$	$\mathbf{96.14}_{\pm 0.05}$

Efficient BiSAM. BiSAM is also compatible with the two ideas constituting ESAM (Du et al., 2022), *Stochastic Weight Perturbation* and *Sharpness-sensitive Data Selection*. A detailed description of the combined algorithm, denoted E-BiSAM, is described in Appendix A.2 and results on CIFAR-10 are shown in Table 5. E-BiSAM (-log) improves the performance of ESAM across *all* models on CIFAR-10.

Table 5. Test accuracies of E-(Bi)SAM.

Model	ESAM	E-BiSAM (-log)
DenseNet-121	$96.30_{\pm 0.22}$	$\mathbf{96.35}_{\pm 0.12}$
Resnet-56	$94.21_{\pm 0.38}$	$\mathbf{94.60}_{\pm 0.24}$
VGG19-BN	$94.16_{\pm 0.09}$	$\mathbf{94.43}_{\pm 0.14}$
WRN-28-2	$95.95_{\pm 0.08}$	$\mathbf{96.00}_{\pm 0.04}$
WRN-28-10	$97.17_{\pm 0.09}$	$\mathbf{97.18}_{\pm 0.05}$
Average	$95.56_{\pm 0.09}$	$\mathbf{95.71}_{\pm 0.06}$

4.3. Fine-tuning on image classification

We conduct experiments of transfer learning on ViT architectures. In particular, we use pretrained ViT-B/16 checkpoint from Visual Transformers (Wu et al., 2020) and finetune the model on Oxford-flowers (Nilsback and Zisserman, 2008) and Oxford-IITPets (Parkhi et al., 2012) datasets. We use AdamW as base optimizer with no weight decay under a linear learning rate schedule and gradient clipping with global norm 1. We set peak learning rate to $1e-4$ and batch size to 512, and run 500 steps with a warmup step of 100. Note that for Flowers dataset, we choose $\mu = 4$ for BiSAM(-log) and $\mu = 20$ for BiSAM(tanh); and for Pets dataset, set $\mu = 6$ for BiSAM(-log) and $\mu = 20$ for BiSAM(tanh). The results in the table indicate that BiSAM benefits transfer learning.

Table 6. Test accuracies for image fine-tuning.

	SAM	BiSAM (-log)	BiSAM (tanh)
Flowers	$98.79_{\pm 0.07}$	$\mathbf{98.93}_{\pm 0.15}$	$98.87_{\pm 0.08}$
Pets	$93.66_{\pm 0.48}$	$\mathbf{94.15}_{\pm 0.24}$	$93.81_{\pm 0.45}$

4.4. NLP Fine-tuning

To check if BiSAM can benefit the natural language processing (NLP) domain, we show empirical text classification results in this section. In particular, we use BERT-base model and finetune it on the GLUE datasets (Wang et al., 2018). Note that we do not include STS-B because it is not a classification task, instead it is a regression task. We use AdamW as base optimizer under a linear learning rate schedule and gradient clipping with global norm 1. We set the peak learning rate to $2e-5$ and batch size to 32, and run 3 epochs with an exception for MRPC and WNLI which are tiny and where we used 5 epochs. We use BiSAM (-log) with the same hyperparameter as on CIFAR datasets in this experiment. Note that we set $\rho = 0.05$ for all datasets except for CoLA with $\rho = 0.01$ and RTE with $\rho = 0.005$. We report the results computed over 10 independent executions in the Table 7, which demonstrates that BiSAM also benefits in NLP domain.

4.5. Noisy labels task

We test on a task outside the i.i.d. setting that the method was designed for. Following Foret et al. (2021) we consider label noise, where a fraction of the labels in the training set are corrupted to another label sampled uniformly at random. Apart from the label perturbation, the experimental setup is otherwise the same as in Section 4.1, except for adjusting $\rho = 0.01$ for SAM and BiSAM when noise rate is 80%, as the original $\rho = 0.05$ causes failure for both methods. We find that BiSAM enjoys similar robustness to label noise as SAM despite not being specifically designed for the setting.

Across 4 tasks, 6 datasets, 9 models, and 2 incorporations with SAM variants, our experiments validate BiSAM's improvement broadly. A *consistent improvement* is observed in all experiments, which also implies a widespread cumulative impact. In addition, it is crucial to emphasize that BiSAM has the same computational complexity as SAM, which has been validated in the implementation. The detailed discussion can be found in Appendix B.

5. Related Work

The min-max zero-sum optimization template has been used in recent years in multiple applications beyond SAM (Foret et al., 2021) e.g., in Adversarial Training (Madry et al., 2018; Latorre et al., 2023) or Generative Adversarial Networks (GANs) (Goodfellow et al., 2014).

In particular, the SAM formulation as a two-player game interacting via addition, has precedence in Robust Bayesian Optimization (Bogunovic et al., 2018), where it is called ϵ-perturbation stability. Even though our formulation starts as a zero-sum game (5), a tractable reformulation (12) requires leveraging the bilevel optimization approach (Bard, 2013).

The bilevel paradigm has already seen applications in Ma-

Table 7. Experimental results of BERT-base finetuned on GLUE.

Method	GLUE	CoLA	SST-2	MRPC		QQP		MNLI	QNLI	RTE	WNLI
		Mcc.	*Acc.*	*Acc.*	*F1.*	*Acc.*	*F1.*	*Acc.*	*Acc.*	*Acc.*	*Acc.*
AdamW	73.21	56.63	91.64	85.64	89.94	90.18	86.81	82.59	89.78	62.38	26.41
-w SAM	75.60	58.78	92.29	86.49	90.51	90.62	87.56	83.96	90.36	60.65	41.20
-w BiSAM	**77.04**	**58.94**	**92.54**	**86.91**	**90.72**	**90.70**	**87.68**	**84.00**	**90.53**	**62.74**	**49.53**

Table 8. Test accuracies of ResNet-32 models trained on CIFAR-10 with label noise.

Noise rate	SGD	SAM	BiSAM (-log)	BiSAM (tanh)
0%	$94.76_{\pm 0.14}$	$94.95_{\pm 0.13}$	$\mathbf{94.98}_{\pm 0.17}$	$95.01_{\pm 0.08}$
20%	$88.65_{\pm 1.75}$	$92.57_{\pm 0.24}$	$\mathbf{92.59}_{\pm 0.11}$	$92.35_{\pm 0.29}$
40%	$84.24_{\pm 0.25}$	$\mathbf{89.03}_{\pm 0.09}$	$88.71_{\pm 0.23}$	$88.86_{\pm 0.18}$
60%	$76.29_{\pm 0.25}$	$82.77_{\pm 0.29}$	$\mathbf{82.91}_{\pm 0.46}$	$82.87_{\pm 0.71}$
80%	$44.44_{\pm 1.20}$	$44.68_{\pm 4.01}$	$\mathbf{50.00}_{\pm 1.96}$	$48.57_{\pm 0.64}$

chine Learning, with regard to hyperparameter optimization (Domke, 2012; Lorraine et al., 2020; Mackay et al., 2019; Franceschi et al., 2018), meta-learning (Franceschi et al., 2018; Rajeswaran et al., 2019), data denoising by importance learning (Ren et al., 2018), neural architecture search (Liu et al., 2018), training data poisoning (Mei and Zhu, 2015; Muñoz-González et al., 2017; Huang et al., 2020), and adversarial training (Robey et al., 2023). Our formulation is the first bilevel formulation in the context of SAM.

ESAM (Du et al., 2022) introduces two tricks, *Stochastic Weight Perturbation (SWP)* and *Sharpness-sensitive Data Selection (SDS)* that subset random variables of the optimization problem, or a subset of the elements in the mini-batch drawn in a given iteration. Neither modification is related to the optimization objective of SAM. Thus, analogous ideas can be used inside our bilevel approach as shown in Algorithm 3. This is useful, as ESAM can reduce the computational complexity of SAM while retaining its performance. We can see a similar result when combined with BiSAM.

In ASAM (Kwon et al., 2021), a notion of *Adaptive Sharpness* is introduced, whereby the constraint set of the perturbation ϵ in (3) is modified to depend on the parameter w. This particular choice yields a definition of sharpness that is invariant under transformations that do not change the value of the loss. The arguments in favor of adaptive sharpness hold for arbitrary losses, and hence, adaptivity can also be incorporated within the bilevel formulation of BiSAM as shown in Algorithm 2. Experimental results in Table 4 demonstrate that this incorporation improves performance.

A relationship between the inner-max objective in SAM and a Bayesian variational formulation was revealed by Möllenhoff and Khan (2022). Based on this result, they proposed *Bayesian SAM (bSAM)*, a modification of SAM that can obtain uncertainty estimates. While such results require a continuity condition on the loss c.f. Möllenhoff and Khan (2022, Theorem 1.), their arguments could be applied to any sufficiently tight continuous approximation of the 0-1 loss. Therefore, a similar relationship between our formulation of SAM and the Bayesian perspective could be derived, enabling uncertainty estimates for BiSAM.

In GSAM (Zhuang et al., 2021), propose to minimize the so-called *surrogate gap* $\max_{\epsilon:\|\epsilon\|\leq\rho} L_S(w+\epsilon) - L_S(w)$ and the perturbed loss $\max_{\epsilon:\|\epsilon\|\leq\rho} L_S(w+\epsilon)$ simultaneously, which leads to a modified SAM update. In Liu et al. (2022), it is proposed to add a random initialization before the optimization step that defines the perturbation. In Ni et al. (2022), it is suggested that using the top-k elements of the mini-batch to compute the stochastic gradients is a good alternative to improve the speed of SAM. To different degrees, such SAM variants have analogous versions in our framework.

6. Conclusions and Future Work

In this work, we proposed a novel formulation of SAM by utilizing the 0-1 loss for classification tasks. By reformulating SAM as a bilevel optimization problem, we aimed to maximize the lower bound of the 0-1 loss through perturbation. We proposed BiSAM, a scalable first-order optimization method, to effectively solve this bilevel optimization problem. Through experiments, BiSAM outperformed SAM in training, fine-tuning, and noisy label tasks meanwhile maintaining a similar computational complexity. In addition, incorporating variants of SAM (*e.g.,* ASAM, ESAM) in BiSAM can improve its performance or efficiency further.

Given the promising results in classification tasks, exploring BiSAM's applicability in other domains such as text generation could broaden its scope. In addition, we claim that the generalization bound, a motivation of SAM, remains valid for 0-1 loss as presented in (5). However, relating the solution of this to the solution of BiSAM shown in (12) is still an open problem, which we leave for future work. We discuss this in detail in Appendix C. Overall, the insights gained from this work offer new directions and opportunities for advancing the field of sharpness-aware optimization.

Acknowledgements

We thank the reviewers for their constructive feedback. Thanks to Fanghui Liu for the helpful discussion. This work was supported by the Swiss National Science Foundation (SNSF) under grant number 200021_205011. This work was supported by Google. This work was supported by Hasler Foundation Program: Hasler Responsible AI (project number 21043).

Impact Statement

This paper identifies a fundamental limitation in the vanilla SAM formulation, highlighting the suboptimality of replacing the 0-1 loss directly with a surrogate upper-bound loss like cross-entropy in the minimax formulation. We introduce BiSAM, an algorithm that addresses this issue using a newly defined lower-bound loss for the maximizer, demonstrating consistent empirical improvements. Our method is compatible with all SAM variants, and this also inspires a reconsideration of other minimax formulations. Additionally, we release our code publicly.

References

Dara Bahri, Hossein Mobahi, and Yi Tay. Sharpness-aware minimization improves language model generalization. In *Annual Meeting of the Association for Computational Linguistics*, 2021.

Jonathan F Bard. *Practical bilevel optimization: algorithms and applications*, volume 30. 2013.

Kayhan Behdin, Qingquan Song, Aman Gupta, Ayan Acharya, David Durfee, Borja Ocejo, Sathiya Keerthi, and Rahul Mazumder. msam: Micro-batch-averaged sharpness-aware minimization. *arXiv preprint arXiv:2302.09693*, 2023.

Ilija Bogunovic, Jonathan Scarlett, Stefanie Jegelka, and Volkan Cevher. Adversarially robust optimization with gaussian processes. *Advances in Neural Information Processing Systems (NeurIPS)*, 31, 2018.

Ben Bolte. Optimized log-sum-exp pytorch function. https://ben.bolte.cc/logsumexp, 2020.

Justin Domke. Generic methods for optimization-based modeling. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2020.

Jiawei Du, Hanshu Yan, Jiashi Feng, Joey Tianyi Zhou, Liangli Zhen, Rick Siow Mong Goh, and Vincent Tan. Efficient sharpness-aware minimization for improved training of neural networks. In *International Conference on Learning Representations (ICLR)*, 2022.

Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In Gal Elidan, Kristian Kersting, and Alexander Ihler, editors, *Uncertainty in Artificial Intelligence Conference*, 2017.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations (ICLR)*, 2021.

Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *International Conference on Machine Learning (ICML)*, 2018.

Saeed Ghadimi and Mengdi Wang. Approximation methods for bilevel programming. *arXiv preprint arXiv:1802.02246*, 2018.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in Neural Information Processing Systems (NeurIPS)*, 27, 2014.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, et al. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

W Ronny Huang, Jonas Geiping, Liam Fowl, Gavin Taylor, and Tom Goldstein. Metapoison: Practical general-purpose clean-label data poisoning. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 2020.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Jungmin Kwon, Jeongseop Kim, Hyunseo Park, and In Kwon Choi. ASAM: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *International Conference on Machine Learning (ICML)*, 2021.

Fabian Latorre, Igor Krawczuk, Leello Tadesse Dadi, Thomas Michaelsen Pethick, and Volkan Cevher. Finding actual descent directions for adversarial training. In *International Conference on Learning Representations (ICLR)*, 2023.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In *International Conference on Learning Representations (ICLR)*, 2018.

Yong Liu, Siqi Mai, Minhao Cheng, Xiangning Chen, Cho-Jui Hsieh, and Yang You. Random sharpness-aware minimization. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Jonathan Lorraine, Paul Vicol, and David Duvenaud. Optimizing millions of hyperparameters by implicit differentiation. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020.

Matthew Mackay, Paul Vicol, Jonathan Lorraine, David Duvenaud, and Roger Grosse. Self-tuning networks: Bilevel optimization of hyperparameters using structured best-response functions. In *International Conference on Learning Representations (ICLR)*, 2019.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018.

Shike Mei and Xiaojin Zhu. Using machine teaching to identify optimal training-set attacks on machine learners. In *AAAI Conference on Artificial Intelligence*, volume 29, 2015.

Thomas Möllenhoff and Mohammad Emtiyaz Khan. SAM as an optimal relaxation of Bayes. In *International Conference on Learning Representations (ICLR)*, 2022.

Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C. Lupu, and Fabio Roli. Towards poisoning of deep learning algorithms with back-gradient optimization. *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017.

Renkun Ni, Ping-yeh Chiang, Jonas Geiping, Micah Goldblum, Andrew Gordon Wilson, and Tom Goldstein. K-sam: Sharpness-aware minimization at the seed of SGD. *arXiv preprint arXiv:2210.12864*, 2022.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, 2008.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. Meta-learning with implicit gradients. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019.

Mengye Ren, Wenyuan Zeng, Binh Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning (ICML)*, 2018.

Alexander Robey, Fabian Latorre, George J Pappas, Hamed Hassani, and Volkan Cevher. Adversarial training should be cast as a non-zero-sum game. *arXiv preprint arXiv:2306.11035*, 2023.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115, 2015.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Davoud Ataee Tarzanagh and Laura Balzano. Online bilevel optimization: Regret analysis of online alternating gradient methods. *arXiv preprint arXiv:2207.02829*, 2022.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Zhicheng Yan, Masayoshi Tomizuka, Joseph Gonzalez, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision. *arXiv preprint arXiv:2006.03677*, 2020.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Qihuang Zhong, Liang Ding, Li Shen, Peng Mi, Juhua Liu, Bo Du, and Dacheng Tao. Improving sharpness-aware minimization with fisher mask for better generalization on language models. *arXiv preprint arXiv:2210.05497*, 2022.

Juntang Zhuang, Boqing Gong, Liangzhe Yuan, Yin Cui, Hartwig Adam, Nicha C Dvornek, James s Duncan, Ting Liu, et al. Surrogate gap minimization improves sharpness-aware training. In *International Conference on Learning Representations (ICLR)*, 2021.

A. Complete experiments

We provide complete algorithms and experimental results here as complementary for Section 4.2.

A.1. Adaptive BiSAM

As we introduced in Section 5, some existing variants of SAM can be incorporated within BiSAM. To demonstrate this, we combine BiSAM with Adaptive Sharpness in ASAM (Kwon et al., 2021). Kwon et al. (2021) propose that the fixed radius of SAM's neighborhoods has a weak correlation with the generalization gap. Therefore, ASAM proposes a normalization operator to realize adaptive sharpness. Following the element-wise operator in Kwon et al. (2021) defined by

$$T_w = \text{diag}(|w_1|, \dots, |w_b|), \text{ where } w = [w_1, \dots, w_b]. \tag{16}$$

We construct Adaptive BiSAM (A-BiSAM) in Algorithm 2.

Algorithm 2 Adaptive BiSAM (A-BiSAM)

Input: Initialization $w_0 \in \mathbb{R}^d$, iterations T, batch size b, step sizes $\{\eta_t\}_{t=0}^{T-1}$, neighborhood size $\rho > 0$, lower bound ϕ.

for $t = 0$ ***to*** $T-1$ **do**
- Sample minibatch $\mathcal{B} = \{(x_1, y_1), \dots, (x_b, y_b)\}$.
- Compute the (stochastic) gradient of the perturbation loss $Q_{\phi,\mu}(w_t)$ defined in Equation (11)
- Compute perturbation $\epsilon_t = \rho \frac{T_w^2 \nabla_w Q(w)}{\|T_w \nabla_w Q(w)\|}$.
- Compute gradient $g_t = \nabla_w L_{\mathcal{B}}(w_t + \epsilon_t)$.
- Update weights $w_{t+1} = w_t - \eta_t g_t$.

To compare A-BiSAM with ASAM, we use the same experimental setting as in Section 4.1 except for the choice of ρ. For both ASAM and A-BiSAM, we use $\rho = 2$ as a result of a grid search over $\{0.1, 0.5, 1, 2, 3\}$ using the validation dataset on CIFAR-10 with Resnet-56. (Note that we do not use $\rho = 0.5$ as in Kwon et al. (2021) because the results of ASAM with $\rho = 0.5$ cannot outperform SAM in our experiments.) We also have two variants of A-BiSAM compared against ASAM:

ASAM: The original Adaptive Sharpness-Aware Minimization (ASAM) algorithm from Kwon et al. (2021)

A-BiSAM (tanh): Algorithm 2 using (14) as lower bound

A-BiSAM (-log): Algorithm 2 using (15) as lower bound

We report the test accuracy of the model with the highest validation accuracy across the training with mean and standard deviations computed over 5 independent executions. The results can be found in Table 9.

Table 9. Test accuracies of A-(Bi)SAM on CIFAR-10 dataset.

Model	ASAM	A-BiSAM (-log)	A-BiSAM (tanh)
DenseNet-121	$96.79_{\pm 0.14}$	$96.79_{\pm 0.13}$	$96.76_{\pm 0.06}$
Resnet-56	$94.86_{\pm 0.18}$	$95.09_{\pm 0.09}$	$94.86_{\pm 0.12}$
VGG19-BN	$95.10_{\pm 0.09}$	$95.14_{\pm 0.14}$	$95.19_{\pm 0.15}$
WRN-28-2	$96.22_{\pm 0.10}$	$96.28_{\pm 0.14}$	$96.29_{\pm 0.18}$
WRN-28-10	$97.37_{\pm 0.07}$	$97.42_{\pm 0.09}$	$97.34_{\pm 0.11}$

A.2. Efficient BiSAM

A-BiSAM above mainly improves the performance of BiSAM while some variants of SAM can enhance the efficiency like Efficient SAM (ESAM) (Du et al., 2022). As we introduced in Section 5, ESAM proposes two tricks, *Stochastic Weight Perturbation (SWP)* and *Sharpness-sensitive Data Selection (SDS)*, which can also be used in BiSAM. When these two tricks are combined with BiSAM we refer to it as Efficient BiSAM (E-BiSAM) in Algorithm 3.

Algorithm 3 Efficient BiSAM (E-BiSAM)

Input: Initialization $w_0 \in \mathbb{R}^d$, iterations T, batch size b, step sizes $\{\eta_t\}_{t=0}^{T-1}$, neighborhood size $\rho > 0$, $\mu > 0$, lower bound ϕ, SWP hyperparameter β, SDS hyperparameter γ.

for $t = 0$ ***to*** $T - 1$ **do**
  Sample minibatch $\mathcal{B} = \{(x_1, y_1), \ldots, (x_b, y_b)\}$.
  for $i = 0$ ***to*** $d - 1$ **do**
    if $w_t[i]$ *is chosen by probability* β **then**
      $\epsilon_t[i] = \frac{\rho}{1-\beta} \nabla_{w[i]} Q(w_t)$
    else
      $\epsilon_t[i] = 0$
  Compute the perturbation loss $Q(w_t + \epsilon_t)$ and construct $\mathcal{B}^+$ with selection ratio γ:

$$\mathcal{B}^+ = \{(x_i, y_i) \in \mathcal{B} : Q(w_t + \epsilon_t) - Q(w_t) > a\}, \text{ where } a \text{ controls } \gamma = \frac{|\mathcal{B}^+|}{|\mathcal{B}|}.$$

  Compute gradient $g_t = \nabla_w L_{\mathcal{B}^+}(w_t + \epsilon_t)$.
  Update weights $w_{t+1} = w_t - \eta_t g_t$.

To compare E-BiSAM with ESAM, we use the same experimental setting as in Section 4.1. For hyperparameter β and γ for SWP and SDS respectively, we choose 0.5 for both which is same as Du et al. (2022). We compare two variants of E-BiSAM against ESAM:

ESAM: The original Efficient Sharpness-Aware Minimization (ESAM) algorithm from Du et al. (2022)

E-BiSAM (tanh): Algorithm 3 using (14) as lower bound

E-BiSAM (-log): Algorithm 3 using (15) as lower bound

We report the test accuracy of the model with the highest validation accuracy across the training with mean and standard deviations computed over 5 independent executions. The results can be found in Table 10.

We observe that E-BiSAM (-log) outperformances ESAM across *all* models and E-BiSAM (tanh) has same or better performance except for on WRN-18-10. As a result, E-BiSAM combined with SWP and SDS improves the efficiency of BiSAM meanwhile keeping good performance.

Table 10. Test accuracies of E-(Bi)SAM on CIFAR-10 dataset.

Model	ESAM	E-BiSAM (-log)	E-BiSAM (tanh)
DenseNet-121	$96.30_{\pm 0.22}$	$96.35_{\pm 0.12}$	$96.32_{\pm 0.11}$
Resnet-56	$94.21_{\pm 0.38}$	$94.60_{\pm 0.24}$	$94.32_{\pm 0.34}$
VGG19-BN	$94.16_{\pm 0.09}$	$94.43_{\pm 0.14}$	$94.31_{\pm 0.12}$
WRN-28-2	$95.95_{\pm 0.08}$	$96.00_{\pm 0.04}$	$95.95_{\pm 0.09}$
WRN-28-10	$97.17_{\pm 0.09}$	$97.18_{\pm 0.05}$	$97.14_{\pm 0.07}$

B. Computational complexity

We claim that BiSAM has the same computational complexity as SAM. This can be seen from the fact that the only change in BiSAM is the loss function used for the ascent step. By visual inspection of such loss function, its forward pass has the same complexity as that of vanilla SAM: we use the same logits but change the final loss function. Hence, the complexity should remain the same. We report timings of each epoch on CIFAR10 in our experiments. Note that time of training on CIFAR10 and CIFAR100 are roughly same.

The relatively small computational overhead (10% in the best cases) is most likely due to cross entropy being heavily optimized in Pytorch. There is no apparent reason why logsumexp should be slower so we expect that the gap can be made

Table 11. Time of each epoch.

Model	SAM (cross-entropy)	BiSAM (logsumexp)
DenseNet-121	58s	64s
Resnet-56	23s	30s
VGG19-BN	10s	16s
WRN-28-2	19s	21s
WRN-28-10	65s	71s

to effectively disappear if logsumexp is given similar attention. In fact, it has been pointed out before that logsumexp in particular is not well-optimized in Pytorch (Bolte, 2020).

To provide further evidence that the computation overhead can be removed, we time the forward/backward of the ascent loss in both Pytorch and TensorFlow with batch size=128 and number of class=100 for 10k repetitions. We find that in tensorflow BiSAM would even enjoy a speedup over SAM.

Table 12. Compare Pytorch with TensorFlow.

Model	SAM (cross-entropy)	BiSAM (logsumexp)
Pytorch	2.40s	3.96s
Tensorflow	3.25s	2.34s

C. Discussion

In this section we highlight that the generalization bounds provided in Foret et al. (2021) also holds when the loss is the (discontinuous) 0-1 loss. We restate for convinience the theorem, which uses the PAC-Bayesian generalization bound of Dziugaite and Roy (2017).

Theorem C.1. *(Foret et al., 2021, (stated informally)) For any $\rho > 0$, with high probability over training set S generated from distribution $\mathcal{D}$:*

$$L_{\mathcal{D}}(w) \leq \max_{\epsilon:\|\epsilon\|_2\leq\rho} L_S(w+\epsilon) + h\left(\|w\|_2^2/\rho^2\right) \tag{17}$$

where $h : \mathbb{R}_+ \to \mathbb{R}_+$ is a strictly increasing function (under some technical conditions on $L_{\mathcal{D}}(w)$).

Remark C.2. Due to no specific differential assumptions in the proof of Theorem C.1, the bound also applies to the 0-1 loss, i.e.,

$$L_{\mathcal{D}}^{01}(w) \leq \max_{\epsilon:\|\epsilon\|_2\leq\rho} L_S^{01}(w+\epsilon) + h\left(\|w\|_2^2/\rho^2\right). \tag{18}$$

This generalization bound provides motivation for solving the minimax problem over the 0-1 loss as given in (5). It remains open to relate the solution of the bilevel optimization relaxation (12) to a solution of (5).