Data Debugging is NP-hard for Classifiers Trained with SGD

Anonymous Author(s)
Affiliation
Address
email

Abstract

Data debugging is to find a subset of the training data such that the model obtained by retraining on the subset has a better accuracy. A bunch of heuristic approaches are proposed, however, none of them are guaranteed to solve this problem effectively. This leaves an open issue whether there exists an efficient algorithm to find the subset such that the model obtained by retraining on it has a better accuracy. To answer this open question and provide theoretical basis for further study on developing better algorithms for data debugging, we investigate the computational complexity of the problem named DEBUGGABLE. Given a machine learning model $\mathcal{M}$ obtained by training on dataset D and a test instance $(\mathbf{x}_{\text{test}}, y_{\text{test}})$ where $\mathcal{M}(\mathbf{x}_{\text{test}}) \neq y_{\text{test}}$, DEBUGGABLE is to determine whether there exists a subset D' of D such that the model $\mathcal{M}'$ obtained by retraining on D' satisfies $\mathcal{M}'(\mathbf{x}_{\text{test}}) = y_{\text{test}}$. To cover a wide range of commonly used models, we take SGD-trained linear classifier as the model and derive the following main results. (1) If the loss function and the dimension of the model are not fixed, DEBUGGABLE is NP-complete regardless of the training order in which all the training samples are processed during SGD. (2) For hinge-like loss functions, a comprehensive analysis on the computational complexity of DEBUGGABLE is provided; (3) If the loss function is a linear function, DEBUGGABLE can be solved in linear time, that is, data debugging can be solved easily in this case. These results not only highlight the limitations of current approaches but also offer new insights into data debugging.

1 Introduction

Given a machine learning model, data debugging is to find a subset of the training data such that the model will have a better accuracy if retrained on that subset [1]. Data debugging serves as a popular method of both data cleaning and machine learning interpretation. In the context of data cleaning, data debugging (*a.k.a.* training data debugging [2] or data cleansing [1]) can be used to improve the quality of the training data by removing the flaws leading to mispredictions [3–5]. When it comes to ML interpretation, data debugging locates the part of the training data responsible for unexpected predictions of an ML model. Therefore it is also studied as a training data-based (*a.k.a.* instance-based [6]) interpretation, which is crucial for helping system developers and ML practitioners to debug ML system by reporting the harmful part of training data [7].

To solve the data debugging problem, existing researches adopt a two-phase score-based heuristic approach [2]. In the first phase, a score representing the estimated impact on the model accuracy is assigned to each training sample in the training data. It is hoped that the harmful part of training data gets a lower score than the other part. In the second phase, training samples with lower scores are removed greedily and the model is retrained on the modified training data. The two phases are carried out iteratively until a well-trained model is obtained. Most of the related works focus on

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

developing algorithms to estimate the scores efficiently in the first phase [8–16], but rarely study the effectiveness of the entire two-phase approach.

Since it is computationally intractable to estimate the score for all possible subsets of the training data, it is often assumed that the score representing the impact of a subset is approximately equal to the sum of the scores of each individual training samples from the subset. However, Koh et. al. [10] showed this is not always the case. For a bunch of subsets sampled from the training data, they empirically studied the difference between the estimated impact and the actual impact of each subset by taking influence functions as the scoring method. The estimated impact is calculated by summing up the score by influence function of each training samples in the subset, and the actual impact is measured by the improvement of accuracy of the model retrained after removing the subset from training data. They found that the estimated impact tends to underestimate the actual impact. Removing a large number of training samples could result in a large deviation between estimated and actual impacts. Although an upper bound of the deviation under certain assumptions has been derived, it is still unknown whether the deviation can be reduced or eliminated efficiently.

The above deviation also poses challenges to the effectiveness of the entire approach. Suppose the influence function is adopted as the scoring method, the accuracy of the model is not guaranteed to improve due to the deviation reported in [10] if a large group of training samples are removed during each iteration. Moreover, there is no theoretical analysis for the effectiveness of the greedy approach in the second phase. Even if only one training sample is removed during each iteration of the two-phase approach, the accuracy of the model is still not guaranteed to be improved. The effectiveness of the entire two-phase approach is therefore not assured. This leaves the following open problem:

Problem 1.1. Is there an efficient algorithm to find the subset of the training data, such that the model obtained by retraining on it has a better accuracy?

The computational complexity results presented in this paper demonstrate that it is unlikely to solve the data debugging problem efficiently in polynomial time. To figure out its hardness, we study the problem DEBUGGABLE which is the decision version of data debugging when the test set consists of only one instance. Formally, DEBUGGABLE is defined as follows:

Problem 1.2 (DEBUGGABLE)**.** Given a classifier $\mathcal{M}$, its training data T, a test instance $(\mathbf{x}, y)$. Is there a $T' \subseteq T$, such that $\mathcal{M}$ predicts y on $\mathbf{x}$ if retrained on T'?

Basically, we prove that DEBUGGABLE is NP-complete, which means data debugging is unlikely to be solved in polynomial time. This result answers the open question mentioned above directly, this is, the large deviation of estimated impacts [10] cannot be reduced or eliminated efficiently. This is because if the impact of a subset of the training data could be accurately estimated as the sum of the impact of each training sample in the subset, data debugging can be solved in polynomial time, which is impossible unless P=NP.

Although DEBUGGABLE is generally intractable, we still hope to develop efficient algorithms tailored to specific cases. Thus it is necessary to figure out the root cause of the hardness for DEBUGGABLE. Previous research are always conducted based on the belief that the complexity of data debugging is due to the chosen model architecture is complicated. However, we show that at least for models trained by stochastic gradient descent (SGD), the hardness stems from the hyper-parameter configuration selected for the SGD training, which was not yet aware of by previous work. To cover a wide range of commonly used machine learning models, we take linear classifiers as the model and show that even for linear classifiers, DEBUGGABLE is NP-hard as long as they are trained by SGD. Moreover, we provided a comprehensive analysis on hyper-parameter configurations that affect the computational complexity of DEBUGGABLE, including the loss function, the model dimension and the training order. Training order, *a.k.a.* training data order [17] or order of training samples [18], refers to the order in which each training sample is considered during the SGD. Detailed complexity results are shown in Table 1.

Our contribution can be concluded as follows:

- We studied the computational complexity of data debugging and showed that data debugging is NP-hard for linear classifiers in the general setting for *all possible training orders*.
- We studied the complexity of DEBUGGABLE when the loss is fixed as the hinge-like function. For 2 or higher dimension, DEBUGGABLE is NP-complete when the training order

Table 1: Computational complexity of the data debugging problem

Loss Function	Dimension	Training Order	Complexity
Not Fixed	Not Fixed	-	NP-hard
Hinge-like	≥ 2	Adversarially Chosen	NP-hard
Hinge-like, $\beta < 0$	1	Adversarially Chosen	NP-hard
Hinge-like, $\beta \geq 0$	1	-	Linear Time
Linear	-	-	Linear Time

is adversarially chosen; For one-dimensional cases, DEBUGGABLE can be NP-hard when the interception $\beta < 0$, and is solvable in linear time when $\beta \geq 0$.

- We proved that DEBUGGABLE is solvable in linear time when the loss function is linear.

Moreover, we have a discussion on the implications of these complexity results for machine learning interpretability and data quality, as well as limitations of score-based greedy methods. Our results suggest the further study as follows. (1) It is better to characterize the training sample and find the criterion which can be used to decide the existence of efficient algorithms; (2) Designing algorithms with CSP-solver is a potential way to solve data debugging more efficiently than the brute-force one; (3) Developing random algorithms is a potential way to solve data debugging successfully with high probability.

1.1 Related Works

The solution of data debugging has applications in database query results reliability enhancement [2, 19], training data cleaning [1] and machine learning interpretation[9, 8, 10, 20, 21]. Existing works on data debugging mainly adopt a two-phase approach, which scores the training samples in the first phase and greedily deletes training samples with lower scores in the second phase. Most of the research focus on the first phase. There are mainly two ways of scoring adopted for data debugging in practice. Leave-one-out (LOO) retraining is a widely studied way, which evaluates the contribution of a training sample through the difference in the model's accuracy trained without that training sample. To avoid the cost of model retraining, Koh and Liang took influence functions as an approximation of LOO [8]. After that, various extensions and improvements of the influence function based method are proposed, such as Fisher kernel [9], influence function for group impacts [10], second-order approximations [11] and scalable influence functions [12]. Another way is Shapley-based scoring, where the impact of a training sample is measured by its average marginal contribution to all subsets of the training data [13]. Since Shapley-base scoring suffers from expensive computational cost [22], recent works focus on techniques that efficiently estimate the Shapley value, including Monte-Carlo sampling [13], group testing [14, 15] and using proxy models such as k-NN [16, 3]. However, those methods do not admit any theoretical guarantee on the effectiveness. This paper discusses the limitations of the above methods and suggests some future directions on data debugging.

2 Preliminaries and Problem Definition

Linear classifiers. Formally, a (binary) linear classifier is a function $\lambda_{\mathbf{w}} : \mathbb{R}^d \to \{-1, 1\}$, where d is called its *dimension* and $\mathbf{w} \in \mathbb{R}^d$ its parameter. Without loss of generality, the bias term of a linear classifier is set as zero in this paper. All vectors in this paper are assumed to be *column* vectors. For an input $\mathbf{x}$, the value of $\lambda_{\mathbf{w}}$ is defined as

$$\lambda_{\mathbf{w}}(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w}^\top \mathbf{x} \geq 0 \\ -1 & \text{otherwise.} \end{cases}$$

We denote the class of linear models as Λ.

Training data. A *training sample* is a pair $(\mathbf{x}, y)$ in which $\mathbf{x} \in \mathbb{R}^d$ is the input and $y \in \{-1, 1\}$ is the label of $\mathbf{x}$. The *training data* is a multiset of training samples. We employ $\mathbf{w} \xrightarrow{T} \mathbf{w}'$ to denote that the parameter $\mathbf{w}'$ is obtained by training the parameter $\mathbf{w}$ on the training data T, and employ $\mathbf{w} \xrightarrow{(\mathbf{x},y)} \mathbf{w}'$ to denote that $\mathbf{w}'$ is obtained by training $\mathbf{w}$ on the training sample $(\mathbf{x}, y)$.

Loss functions and learning rates. Binary linear classifiers typically use unary functions on $y\mathbf{w}^\top \mathbf{x}$ as their loss functions [23]. Therefore we only consider loss functions of the form $\mathcal{L} : y\mathbf{w}^\top\mathbf{x} \mapsto \mathbb{R}$ for the rest of the paper.

The *linear* loss is in the form of

$$\mathcal{L}_{\texttt{lin}}(y\mathbf{w}^\top\mathbf{x}) = -\alpha(y\mathbf{w}^\top\mathbf{x} + \beta).$$

The *hinge-like* loss function is defined as the following form

$$\mathcal{L}_{\texttt{hinge}}(y\mathbf{w}^\top\mathbf{x}) = \begin{cases} -\alpha(y\mathbf{w}^\top\mathbf{x} + \beta), & y\mathbf{w}^\top\mathbf{x} < \beta \\ 0, & \text{otherwise.} \end{cases}$$

We call β as the *interception* of $\mathcal{L}_{\texttt{hinge}}$. We represent the learning rate of a model using a vector $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_d)$, where $\eta_i \geq 0$ and each parameter w_i can be updated with the corresponding learning rate η_i.

Stochastic gradient descent. The stochastic gradient descent (SGD) method updates parameter $\mathbf{w}$ from its initial value $\mathbf{w}^{(0)}$ through several epochs. During each epoch, the SGD goes through the entire set of training samples in some training order through several iterations. The training order is defined as a sequence of training samples, in the form of $(\mathbf{x}_1, y_1)\ldots(\mathbf{x}_n, y_n)$. For $1 \leq i < j \leq n$, $(\mathbf{x}_i, y_i)$ is considered before $(\mathbf{x}_j, y_j)$ during the SGD. We use w_i to denote the i-th coordinate of $\mathbf{w}$. We also use $\mathbf{w}^{(e,k)}$ to denote the value of $\mathbf{w}$ at the end of k-th iteration of epoch e and use $\mathbf{w}^{(e)}$ to denote the value of $\mathbf{w}$ after the end of epoch e. Assuming $(\mathbf{x}, y)$ to be the training sample considered at iteration k, the stochastic gradient descent (SGD) method updates parameter w_i for each i by

$$w_i^{(e,k)} \leftarrow w_i^{(e,k-1)} - \eta_i \cdot \frac{\partial \mathcal{L}(y(\mathbf{w}^{(e,k-1)})^\top\mathbf{x})}{\partial w_i} \tag{1}$$

In other words, we have

$$\mathbf{w}^{(e,k)} \leftarrow \mathbf{w}^{(e,k-1)} - \boldsymbol{\eta} \otimes \nabla\mathcal{L}(y(\mathbf{w}^{(e,k-1)})^\top\mathbf{x})$$

where $\boldsymbol{\eta} \otimes \nabla\mathcal{L} = (\eta_1 \frac{\partial \mathcal{L}}{\partial w_1}, \ldots, \eta_d \frac{\partial \mathcal{L}}{\partial w_d})$ is the Hadamard product. We say a training sample $\mathbf{x}$ is *activated* at iteration k during epoch e if $\nabla\mathcal{L}(y(\mathbf{w}^{(e,k-1)})^\top\mathbf{x}) \neq 0$. The SGD terminates at the end of epoch e if $\|\mathbf{w}^{(e-1)} - \mathbf{w}^{(e)}\| < \varepsilon$ for threshold ε or e reached some predetermined value. We denote $\mathbf{w}^* = \mathbf{w}^{(e)}$. A linear classifier trained by SGD with the meta-parameters mentioned above is denoted as $\texttt{SGD}_\Lambda(\mathcal{L}, \boldsymbol{\eta}, \varepsilon, T) = \lambda_{\mathbf{w}^*}$. With a slight abuse of notation, we define $\texttt{SGD}_\Lambda(\mathcal{L}, \boldsymbol{\eta}, \varepsilon, T, \mathbf{x}) = \lambda_{\mathbf{w}^*}(\mathbf{x})$. We also use $\texttt{SGD}_\Lambda(T, \mathbf{x})$ to avoid cluttering when the context is clear.

Problem definition. With the above definitions, DEBUGGABLE for SGD-trained linear classifiers can be formalized as follows:

154

DEBUGGABLE-LIN
Input: Training data T, loss function $\mathcal{L}$, initial parameter $\mathbf{w}^{(0)}$, learning rate $\boldsymbol{\eta}$, threshold ε and instance $(\mathbf{x}_{\text{test}}, y_{\text{test}})$.
Output: "Yes": if $\exists\Delta \subseteq T$ such that $\texttt{SGD}_\Lambda(\mathcal{L}, \boldsymbol{\eta}, \varepsilon, T \setminus \Delta, \mathbf{x}_{\text{test}}) = y_{\text{test}}$;
"No": otherwise.

We say $\texttt{SGD}_\Lambda(\mathcal{L}, \boldsymbol{\eta}, \varepsilon, T)$ is *debuggable* on $(\mathbf{x}_{\text{test}}, y_{\text{test}})$ if $(\mathcal{L}, \mathbf{w}^{(0)}, \boldsymbol{\eta}, \varepsilon, T, \mathbf{x}_{\text{test}}, y_{\text{test}})$ is a yes-instance of DEBUGGABLE-LIN, and not *debuggable* on $(\mathbf{x}_{\text{test}}, y_{\text{test}})$ otherwise.

3 Results for Unfixed Loss Functions

In this section, we prove the NP-hardness of DEBUGGABLE-LIN. Intuitively, DEBUGGABLE-LIN is to determine whether there exists a subset $T' \subseteq T$ where activated training samples within T' drive the parameter $\mathbf{w}$ toward the region defined by $y_{\text{test}}\mathbf{w}^\top\mathbf{x}_{\text{test}} > 0$. The activation of training samples depends on the complex interaction between the training data and the model.

Theorem 3.1. DEBUGGABLE-LIN is NP-hard for all training orders.

We only show the proof sketch and leave the details in the appendix.

Proof Sketch. We build a reduction from an NP-hard problem MONOTONE 1-IN-3 SAT [24]:

> MONOTONE 1-IN-3 SAT
> **Input:** A 3-CNF formula φ with no negation signs.
> **Output:**"Yes": if φ has a 1-in-3 assignment, under which each clause contains exactly one true literal;
> "No": otherwise.

For example, $\varphi_1 = (x_1 \vee x_2 \vee x_3) \wedge (x_2 \vee x_3 \vee x_4)$ is a yes-instance because $(x_1, x_2, x_3, x_4) =$ (T,F,F,T) is an 1-in-3 assignment; $\varphi_2 = (x_1 \vee x_2 \vee x_3) \wedge (x_2 \vee x_3 \vee x_4) \wedge (x_1 \vee x_2 \vee x_4) \wedge (x_1 \vee x_3 \vee x_4)$ is a no-instance.

Given a 3-CNF formula φ, our goal is to construct a configuration of the training process, such that the resulting model outputs the correct answer if and only if its training data T' encodes an 1-in-3 assignment ν of φ. This can be done by carefully designing the encoding so that for each $x_i \in \varphi$, $\nu(x_i) =$ TRUE if and only if $\mathbf{t}_{x_i} \in T'$. Finally, we can construct some T with $T \supseteq T' \cup \{\mathbf{t}_{x_i} | x_i \in \varphi\}$, such that some classifier trained on T is a yes-instance of DEBUGGABLE-LIN if and only if φ is a yes-instance of MONOTONE 1-IN-3 SAT, thereby finishing our proof.

The reduction. Suppose φ has m clauses and n variables, let $N = n + 2m + 1$. We set the dimension of the linear classifier to N.

<u>The input.</u> Each coordinate of the input is named as

$$\mathbf{x} = (x_{c_1}, \ldots, x_{c_m}, x_{x_1}, \ldots, x_{x_n}, x_{b_1}, \ldots, x_{b_m}, x_{\text{dummy}})^\top$$

We also use x_i to denote the i-th coordinate of $\mathbf{x}$.

<u>The parameters.</u> Each coordinate of the parameter is named as

$$\mathbf{w} = (w_{c_1}, \ldots, w_{c_m}, w_{x_1}, \ldots, w_{x_n}, w_{b_1}, \ldots, w_{b_m}, w_{\text{dummy}})^\top$$

We also use w_i to denote the i-th coordinate of $\mathbf{w}$. Each w_{x_j} represents the truth value of variable x_j, where 1 represents TRUE and -1 represents FALSE. Similarly, each w_{c_j} represents the truth value of clause c_j based on the value of its variables. w_{b_j} and $w_{\texttt{dummy}}$ are used for convenience of proof.

The initial value of the parameter is set to

$$\mathbf{w}^{(0)} = (\overbrace{\frac{1}{2}, \ldots, \frac{1}{2}}^{m}, \overbrace{-1, \ldots, -1}^{n}, \overbrace{-1, \ldots, -1}^{m}, 1)^\top$$

<u>Loss function.</u> We denote $U(x_0, \delta) := \{x | x_0 - \delta < x < x_0 + \delta\}$ as the δ-neighborhood of x_0 and define $U(\pm x_0, \delta) = U(x_0, \delta) \cup U(-x_0, \delta)$. We define the *local ramp function* as

$$r_{x_0,\delta}(x) = \begin{cases} 0 & , x \le x_0 - \delta; \\ x - x_0 + \delta & , x \in U(x_0, \delta); \\ 2\delta & , x \ge x_0 + \delta. \end{cases}$$

The loss function is defined as

$$\mathcal{L} = -\frac{12N}{5} r_{-5,0.01}(y\mathbf{w}^\top \mathbf{x}) - r_{-\frac{1}{2},0.26}(y\mathbf{w}^\top \mathbf{x}) - \frac{1}{1000N} \sum_{x_0 \in \{\pm 1, \pm 3\}} r_{x_0,0.01}(y\mathbf{w}^\top \mathbf{x}).$$

$\mathcal{L}$ is monotonically decreasing with derivatives

$$\frac{\partial \mathcal{L}}{\partial w_i} = \begin{cases} -\frac{12N}{5} \cdot y x_i & , y\mathbf{w}^\top \mathbf{x} \in U(-5, 0.01); \\ -y x_i & , y\mathbf{w}^\top \mathbf{x} \in U(-\frac{1}{2}, 0.26); \\ -\frac{1}{1000N} y x_i & , y\mathbf{w}^\top \mathbf{x} \in \bigcup_{x_0 \in \{\pm 1, \pm 3\}} U(x_0, 0.01); \\ 0 & , \text{otherwise}. \end{cases} \tag{2}$$

Table 2: Training data for $\texttt{var}(i)$

x_{x_i}	y
5	1

Table 3: Training data for $\texttt{clause}(i, i_1, i_2, i_3)$

x_{c_i}	$x_{x_{i_1}}$	$x_{x_{i_2}}$	$x_{x_{i_3}}$	x_{b_i}	y
1	1	1	1	$\frac{1}{2}$	1

Learning rate. The learning rate for SGD is set to be

$$\boldsymbol{\eta} = (\overbrace{5, \dots, 5}^{m}, \overbrace{\frac{1}{6N}, \dots, \frac{1}{6N}}^{n}, \overbrace{2000N, \dots, 2000N}^{m}, 1)^\top.$$

Training data. We define two gadgets, $\texttt{var}(i)$ and $\texttt{clause}(i, i_1, i_2, i_3)$, as illustrated in Table 2 and 3. All the unspecified coordinates are set to zero. We use T_0 to denote the training data. $\texttt{var}(i)$ is contained in T_0 if and only if $x_i \in \varphi$, and $\texttt{clause}(i, i_1, i_2, i_3)$ is contained in T_0 if and only if $c_i = (x_{i_1} \vee x_{i_2} \vee x_{i_3}) \in \varphi$.

Threshold and instance. The threshold ε can be any fixed value in $\mathbb{R}_+$. The instance is defined as $(\mathbf{x}_{\text{test}}, y_{\text{test}})$, where $y_{\text{test}} = 1$ and

$$\mathbf{x}_{\text{test}} = (\overbrace{1, \dots, 1}^{m}, \overbrace{0, \dots, 0}^{n+m}, \frac{-11m + 5}{2})^\top.$$

The following reduction works for all possible training orders. Intuitively, during the training process, each $\texttt{var}(i)$ in the training data will set w_{x_i} to around 1 (that is, mark x_i as TRUE) in the first epoch, and each $\texttt{clause}(i, i_1, i_2, i_3)$ will set w_{c_i} to near $\frac{11}{2}$ in the second epoch, if and only if exactly one of $w_{x_{i_1}}, w_{x_{i_2}}, w_{x_{i_3}}$ is near 1 and the others near -1 (that is, mark c_i as satisfied if exactly one of its literals is TRUE and the others FALSE). The training process terminates at the end of the second epoch. $\square$

4 Results for Fixed Loss Functions

We have proved the NP-hardness for DEBUGGABLE-LIN when the loss function is not fixed. In this section, we study the complexity when the loss function is fixed as linear and hinge-like functions. Assuming that SGD terminates after only one epoch with a fixed order, we will show that DEBUGGABLE-LIN is solvable in linear time for linear loss. For hinge-like loss functions, DEBUGGABLE-LIN can be solved in linear time only when the dimension $d = 1$ and the interception $\beta \geq 0$. For the rest cases, DEBUGGABLE-LIN becomes NP-hard.

4.1 The Easy Case

We start with the linear loss function $\mathcal{L} = -\alpha(y\mathbf{w}^\top\mathbf{x} + \beta)$, with which all the training data are activated and $\mathbf{w}^* = \mathbf{w}^*(T) = \mathbf{w}^{(0)} + \sum_{(\mathbf{x},y)\in T} \alpha y \boldsymbol{\eta} \otimes \mathbf{x}$. Since $y_{\text{test}} \in \{-1, 1\}$, DEBUGGABLE-LIN is equivalent to deciding whether

$$\max_{T' \subseteq T} \{y_{\text{test}}(\mathbf{w}^*(T'))^\top \mathbf{x}_{\text{test}}\} > 0.$$

A training sample $(\mathbf{x}, y)$ is "good" if $y_{\text{test}}(\alpha y \boldsymbol{\eta} \otimes \mathbf{x})^\top \mathbf{x}_{\text{test}} > 0$ and "bad" otherwise. The *good training-sample assessment* (GTA) algorithm, as shown in Algorithm 1, deals with this situation by greedily picking all "good" training samples.

Denoting T^* as the set of all good data in T, it follows that

$$\begin{aligned} y_{\text{test}}(\mathbf{w}^*(T^*))^\top \mathbf{x}_{\text{test}} &= y_{\text{test}}(\mathbf{w}^{(0)})^\top \mathbf{x}_{\text{test}} + \sum_{(\mathbf{x},y)\in T^*} y_{\text{test}}(\alpha y \boldsymbol{\eta} \otimes \mathbf{x})^\top \mathbf{x}_{\text{test}} \\ &\geq y_{\text{test}}(\mathbf{w}^{(0)})^\top \mathbf{x}_{\text{test}} + \sum_{(\mathbf{x},y)\in T'} y_{\text{test}}(\alpha y \boldsymbol{\eta} \otimes \mathbf{x})^\top \mathbf{x}_{\text{test}} \end{aligned}$$

for all $T' \subseteq T$. Hence $\max_{T' \subseteq T}\{y_{\text{test}}(\mathbf{w}^*(T'))^\top \mathbf{x}_{\text{test}}\} = y_{\text{test}}(\mathbf{w}^*(T^*))^\top \mathbf{x}_{\text{test}}$ and DEBUGGABLE-LIN can be solved by GTA in linear time. The following theorem is straightforward.

Theorem 4.1. DEBUGGABLE-LIN is linear time solvable for linear loss functions.

Algorithm 1: Good Training-sample Assessment (GTA)

Input: Training data T, loss function $\mathcal{L}$, initial parameter $\mathbf{w}^{(0)}$, learning rate $\boldsymbol{\eta}$, threshold ε and test instance $(\mathbf{x}_{\text{test}}, y_{\text{test}})$.

Output: TRUE, iff $\texttt{SGD}_\Lambda(\mathcal{L}, \boldsymbol{\eta}, \varepsilon, T)$ is debuggable on $(\mathbf{x}_{\text{test}} y_{\text{test}})$.

$\mathbf{w} \leftarrow \mathbf{w}^{(0)}$;
for $(\mathbf{x}, y) \in T$ **do**
  if $y_{test}(\alpha y \boldsymbol{\eta} \otimes \mathbf{x})^\top \mathbf{x}_{test} > 0$ **then**
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha y \boldsymbol{\eta} \otimes \mathbf{x}$;
  end
end
if $y_{test} \mathbf{w}^\top \mathbf{x}_{test} \geq 0$ **then**
  return TRUE;
end
return FALSE;

GTA is still effective for one-dimensional classifiers trained with hinge-like losses when $\beta \geq 0$.

Theorem 4.2. DEBUGGABLE-LIN is linear time solvable for hinge-like loss functions, when $d = 1$ and $\beta \geq 0$.

Proof. It suffices to prove that if $\exists T' \subseteq T$ such that $\texttt{SGD}_\Lambda(T', x_{\text{test}}) = y_{\text{test}}$, $\texttt{SGD}_\Lambda(T^*, x_{\text{test}}) = y_{\text{test}}$.

a) Suppose all the data in T^* are activated, we have

$$\begin{aligned}
y_{\text{test}} w^*(T^*) x_{\text{test}} &= y_{\text{test}} w^{(0)} x_{\text{test}} + \sum_{(x,y) \in T^*} y_{\text{test}} \alpha y \eta x x_{\text{test}} \\
&\geq y_{\text{test}} w^{(0)} x_{\text{test}} + \sum_{(x,y) \in T' \cap T^*} y_{\text{test}} \alpha y \eta x x_{\text{test}} + \sum_{(x,y) \in T' \setminus T^*} y_{\text{test}} \alpha y \eta x x_{\text{test}} \\
&= y_{\text{test}} w^*(T') x_{\text{test}} \geq 0
\end{aligned}$$

b) Suppose $(x, y) \in T^*$ is the first inactivated data during the training phase, and w is the current parameter, we have $ywx > \beta$. Since $\alpha\eta \cdot (xy) \cdot (x_{\text{test}} y_{\text{test}}) \geq 0$, we have $(x_{\text{test}} y_{\text{test}}) \cdot w \geq 0$. Let T'' be the set of training data appeared before (x, y), we have $y_{\text{test}} w^*(T^*) x_{\text{test}} \geq y_{\text{test}} w^*(T'') x_{\text{test}} \geq 0$. □

4.2 The Hard Case

The gradient of training data may not always be activated and could be affected by the training order. When the training order is adversarially chosen, the following theorem shows that DEBUGGABLE-LIN is NP-hard for all $d \geq 2$ and $\beta \in \mathbb{R}$.

Theorem 4.3. If the training order is adversarially chosen and $d \geq 2$, DEBUGGABLE-LIN is NP-hard for *each* hinge-like loss function at *every* constant learning rate.

Proof sketch. Since the result can be easily extended for all $d > 2$ by padding the other $d - 2$ dimensions with zeros, we only prove for the case of $d = 2$. We assume $\beta \geq -1$ and leave the $\beta < -1$ case to the appendix. To avoid cluttering, we further assume $\boldsymbol{\eta} = \mathbf{1}$ and $\alpha = 1$. The proof can be easily generalized by appropriately re-scaling the constructed vectors.

We build a reduction from the subset sum problem, which is well-known to be NP-hard:

SUBSET SUM
Input: A set of positive integer S, and a positive integer t.
Output: "Yes": if $\exists S' \subseteq S$ such that $\sum_{a \in S'} a = t$;
"No": otherwise.

Suppose $n = |S|$, $m = \max_{a \in S}\{a\}$, $\gamma = \max\{\beta, 1\}$ and $S = \{a_1, a_2, \dots, a_n\}$. We further assume $n > 1$. Let the training data be

$$T = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)\} \cup \{(\mathbf{x}_c, y_c), (\mathbf{x}_b, y_b), (\mathbf{x}_a, y_a)\}$$

where $\mathbf{x}_i y_i = (\frac{\sqrt{\gamma}}{n+1}, 3\sqrt{\gamma}a_i)$ for all $1 \le i \le n$, $\mathbf{x}_c y_c = ((18n^2m^2 - 2)\sqrt{\gamma}, -3t\sqrt{\gamma})$, $\mathbf{x}_b y_b = (\sqrt{\gamma}, -\sqrt{\gamma})$, $\mathbf{x}_a y_a = (\sqrt{\gamma}, \sqrt{\gamma})$. Let $\mathbf{w}^{(0)} = (-18n^2m^2\sqrt{\gamma}, 0)$. Let the test instance $(\mathbf{x}_{\text{test}}, y_{\text{test}})$ satisfy $\mathbf{x}_{\text{test}} y_{\text{test}} = (1, 0)$.

Let the training order be $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n), (\mathbf{x}_c, y_c), (\mathbf{x}_b, y_b), (\mathbf{x}_a, y_a)$.

For each $1 \le i < n$, suppose $\mathbf{w}^{(0)} \xrightarrow{T \cap \{(\mathbf{x}_i, y_i) | 1 \le j \le i\}} \mathbf{w}_i$, we have

$$\begin{aligned} y_{i+1}\mathbf{w}_i^\top \mathbf{x}_{i+1} &\le \frac{\sqrt{\gamma}}{n+1}(-18n^2m^2\sqrt{\gamma} + \frac{\sqrt{\gamma}i}{n+1}) + 3\sqrt{\gamma}a_{i+1}\sum_{j=1}^{i} 3\sqrt{\gamma}a_j \\ &\le \gamma\left(-\frac{n-1}{n+1} \cdot 9nm^2 + \frac{n}{(n+1)^2}\right) < -1 \le \beta \end{aligned}$$

This means all the $T \setminus \{(\mathbf{x}_c, y_c), (\mathbf{x}_b, y_b), (\mathbf{x}_a, y_a)\}$ can be activated. Thus the resulting parameter trained by $T \setminus \{(\mathbf{x}_c, y_c), (\mathbf{x}_b, y_b), (\mathbf{x}_a, y_a)\}$ is

$$\mathbf{w}_c = \mathbf{w}^{(0)} + \sum_{i=1}^{n} \mathbf{x}_i y_i = \left(-18n^2m^2\sqrt{\gamma} + \frac{\sqrt{\gamma}|T^*|}{n+1}, 3\sqrt{\gamma}\sum_{i=1}^{n} a_i\right).$$

It now suffices to prove that for all $S' \subseteq S$, $\sum_{a \in S'} a = t$ if and only if $\exists T' \subseteq T$ such that $\mathbf{w} : \mathbf{w}^{(0)} \xrightarrow{T'} \mathbf{w}$ satisfies $y_{\text{test}}\mathbf{w}^\top \mathbf{x}_{\text{test}} > 0$.

<u>If:</u> Suppose $\exists S' \subseteq S$ such that $\sum_{a \in S} a = t$, we prove that $\exists T' \subseteq T$ such that $y_{\text{test}}(\mathbf{w}^*)^\top \mathbf{x}_{\text{test}} > 0$ for $\mathbf{w}^*$ satisfying $\mathbf{w}^{(0)} \xrightarrow{T'} \mathbf{w}^*$.

Let $T^* = \{(\mathbf{x}_i, y_i) | a_i \in S'\}$, $T' = T^* \cup \{(\mathbf{x}_c, y_c), (\mathbf{x}_b, y_b), (\mathbf{x}_a, y_a)\}$. We have

$$\mathbf{w}_c = (-18n^2m^2\sqrt{\gamma} + \frac{\sqrt{\gamma}|T^*|}{n+1}, 3\sqrt{\gamma}\sum_{a_i \in S'} a_i) = (-18n^2m^2\sqrt{\gamma} + \frac{\sqrt{\gamma}|T^*|}{n+1}, 3\sqrt{\gamma}t).$$

And therefore $y_c\mathbf{w}_c^\top \mathbf{x}_c = \gamma\left((-18n^2m^2 + \frac{|T^*|}{n+1})(18n^2m^2 - 2) - 9t^2\right) < -1 \le \beta$, so

$$\mathbf{w}_c \xrightarrow{(\mathbf{x}_c, y_c)} \mathbf{w}_b = \mathbf{w}_c + \mathbf{x}_c y_c = (\sqrt{\gamma}(\frac{|T^*|}{n+1} - 2), 0).$$

Note that $y_b\mathbf{w}_b^\top \mathbf{x}_b = \gamma(\frac{|T^*|}{n+1} - 2) < -1 \le \beta$, we have

$$\mathbf{w}_b \xrightarrow{(\mathbf{x}_b, y_b)} \mathbf{w}_a = \mathbf{w}_b + \mathbf{x}_a y_a = (\sqrt{\gamma}(\frac{|T^*|}{n+1} - 1), -\sqrt{\gamma})$$

Note also that $y_a\mathbf{w}_a^\top \mathbf{x}_a = \gamma(\frac{|T^*|}{n+1} - 2) < -1 \le \beta$, we have

$$\mathbf{w}_a \xrightarrow{(\mathbf{x}_a, y_a)} \mathbf{w}^* = \mathbf{w}_a + \mathbf{x}_a y_a = (\frac{|T^*|\sqrt{\gamma}}{n+1}, 0)$$

Therefore, $y_{\text{test}}(\mathbf{w}^*)^\top \mathbf{x}_{\text{test}} = \frac{|T^*|\sqrt{\gamma}}{n+1} > 0$.

<u>Only if:</u> For each $T' \subseteq T$, let $T^* = T' \setminus \{(\mathbf{x}_c, y_c), (\mathbf{x}_b, y_b), (\mathbf{x}_a, y_a)\}$. If $y_{\text{test}}(\mathbf{w}^*)^\top \mathbf{x}_{\text{test}} > 0$ for $\mathbf{w}^*$ satisfying $\mathbf{w}^{(0)} \xrightarrow{T'} \mathbf{w}^*$, we prove that $\exists S' \subseteq S$ such that $\sum_{a \in S'} a = t$. We first show that for each $T' \subseteq T$, if $\mathbf{w}(\mathbf{w}^{(0)} \xrightarrow{T'} \mathbf{w})$ satisfying $y_{\text{test}}\mathbf{w}^\top \mathbf{x}_{\text{test}} > 0$, we have $\forall k \in \{a, b, c\}, (\mathbf{x}_k, y_k) \in T', y_k\mathbf{w}_k^\top \mathbf{x}_k < \gamma$, where $\mathbf{w}^{(0)} \xrightarrow{T^*} \mathbf{w}_c \xrightarrow{(\mathbf{x}_c, y_c)} \mathbf{w}_b \xrightarrow{(\mathbf{x}_b, y_b)} \mathbf{w}_a$. Otherwise, suppose $\exists k \in \{a, b, c\}$ such that $(\mathbf{x}_k, y_k) \notin T'$ or $y_k\mathbf{w}_k^\top \mathbf{x}_k \ge \gamma$, we have

$$y_{\text{test}}\mathbf{w}^\top \mathbf{x}_{\text{test}} \le \sqrt{\gamma}(\frac{|T^*|}{n+1} - 1) < 0$$

which contradicts to the fact that $y_{\text{test}}\mathbf{w}^\top\mathbf{x}_{\text{test}} \geq 0$.

Let $S' = \{a_i | (\mathbf{x}_i, y_i) \in T^*\}$ and $t' = \sum_{a \in S'} a_i$, it suffices to prove $t' = t$. Notice that

$$\mathbf{w}^{(0)} \xrightarrow{T^* \cap \{(\mathbf{x}_i, y_i) | 1 \leq j \leq i\}} \mathbf{w}_c = (\sqrt{\gamma}(-18n^2m^2 + \frac{|T^*|}{n+1}), 3\sqrt{\gamma}\sum_{a_i \in S'} a_i)$$
$$= (\sqrt{\gamma}(-18n^2m^2 + \frac{|T^*|}{n+1}), 3\sqrt{\gamma}t')$$

Hence $y_c\mathbf{w}_c^\top\mathbf{x}_c = \gamma(-18n^2m^2 + \frac{|T^*|}{n+1})(18n^2m^2 - 2) - 9\gamma tt' < -1 \leq \beta$, thus

$$\mathbf{w}_c \xrightarrow{(\mathbf{x}_c, y_c)} \mathbf{w}_b = \mathbf{w}_c + \mathbf{x}_c y_c = (\sqrt{\gamma}(\frac{|T^*|}{n+1} - 2), 3\sqrt{\gamma}(t' - t))$$

(1) If $t' \leq t - 1$, we have $y_b\mathbf{w}_b^\top\mathbf{x}_b = \gamma\left(\frac{|T^*|}{n+1} - 2 + 3(t - t')\right) > \gamma \geq \beta$, a contradiction.

(2) If $t' \geq t + 1$, we have $y_a\mathbf{w}_a^\top\mathbf{x}_a = \gamma\left(\frac{|T^*|}{n+1} - 2 + 3(t' - t)\right) > \gamma \geq \beta$, another contradiction.

Therefore $t' = t$, and this completes the proof. □

Moreover, DEBUGGABLE-LIN is NP-hard even when $d = 1$ and $\beta < 0$.

Theorem 4.4. If the training order is adversarially chosen and $d = 1$, DEBUGGABLE-LIN remains NP-hard for *each* hinge-like loss function with $\beta < 0$ at *every* constant learning rate.

Remarks. The training order in this section can be arbitrary as long as the last three training samples are $(\mathbf{x}_c, y_c), (\mathbf{x}_b, y_b), (\mathbf{x}_a, y_a)$, respectively. All the training samples are "good" since for each $(\mathbf{x}, y) \in T$ we have $\mathbf{x}^\top\mathbf{x}_{\text{test}}yy_{\text{test}} > 0$. This implies that DEBUGGABLE-LIN is NP-hard even if all the training data are "good" training samples, and exemplifies why the GTA algorithm fails for higher dimensions.

5 Discussion and Conclusion

In this paper, we provided a comprehensive analysis on the complexity of DEBUGGABLE. We focus on the linear classifier that is trained using SGD, as it is a key component in the majority of popular models.

Since DEBUGGABLE is a special case of data debugging, the above results proved the intractability of data debugging and therefore gives a negative answer to Problem 1.1 declared in the introduction. The complexity results also demonstrated that it is not accurate to estimate the impact of subset of training data by summing up the score of each training samples in the subset, *as long as the scores can be calculated in polynomial time*.

In Section 4, a training sample is said to be "good" if it can help the resulting model to predict correctly on the test instance. That is, it can increase $y_{\text{test}}(\mathbf{w}^*)^\top\mathbf{x}_{\text{test}}$. However, in our proof we showed that DEBUGGABLE remains NP-hard even if all training samples are "good". This suggests that the quality of a training sample does not depend only on some properties of itself but also on the interaction between the rest of the training data, which should be taken into consideration when developing data cleaning approaches.

Moreover, the NP-hardness of DEBUGGABLE implies that, it is in general intractable to figure out the causality between even the prediction of a linear classifier and its training data. This may be seem surprising since linear classifiers have long been considered "inherently interpretable". As warned in [25], *a method being "inherently interpretable" needs to be verified before it can be trusted*, the concept of interpretability must be *rigorously defined*, or at least its boundaries specified.

Our results suggests the following directions for future research. Firstly, characterizing the training sample may be helpful in designing efficient algorithms for data debugging; Secondly, designing algorithms using CSP-solver is a potential way to solve data debugging more efficiently than the brute-force algorithms; Finally, developing random algorithms is a potential way to solve data debugging successfully with high probability.

References

[1] Satoshi Hara, Atsushi Nitanda, and Takanori Maehara. *Data Cleansing for Models Trained with SGD*. Curran Associates Inc., Red Hook, NY, USA, 2019.

[2] Weiyuan Wu, Lampros Flokas, Eugene Wu, and Jiannan Wang. Complaint-driven training data debugging for query 2.0. pages 1317–1334, 06 2020. doi: 10.1145/3318464.3389696.

[3] Bojan Karlaš, David Dao, Matteo Interlandi, Bo Li, Sebastian Schelter, Wentao Wu, and Ce Zhang. Data debugging with shapley importance over end-to-end machine learning pipelines, 2022.

[4] Felix Neutatz, Binger Chen, Ziawasch Abedjan, and Eugene Wu. From cleaning before ml to cleaning for ml. *IEEE Data Eng. Bull.*, 44:24–41, 2021. URL `https://api.semanticscholar.org/CorpusID:237542697`.

[5] Peng Li, Xi Rao, Jennifer Blase, Yue Zhang, Xu Chu, and Ce Zhang. Cleanml: A study for evaluating the impact of data cleaning on ml classification tasks. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 13–24, 2021. doi: 10.1109/ICDE51399.2021.00009.

[6] Juhan Bae, Nathan Ng, Alston Lo, Marzyeh Ghassemi, and Roger Grosse. If influence functions are the answer, then what is the question? In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9781713871088.

[7] Romila Pradhan, Jiongli Zhu, Boris Glavic, and Babak Salimi. Interpretable data-based explanations for fairness debugging. In *Proceedings of the 2022 International Conference on Management of Data*, SIGMOD '22, page 247–261, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450392495. doi: 10.1145/3514221.3517886. URL `https://doi.org/10.1145/3514221.3517886`.

[8] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1885–1894. JMLR.org, 2017.

[9] Rajiv Khanna, Been Kim, Joydeep Ghosh, and Oluwasanmi Koyejo. Interpreting black box predictions using fisher kernels. In *International Conference on Artificial Intelligence and Statistics*, 2018. URL `https://api.semanticscholar.org/CorpusID:53085397`.

[10] Pang Wei Koh, Kai-Siang Ang, Hubert Hua Kian Teo, and Percy Liang. On the accuracy of influence functions for measuring group effects. In *Neural Information Processing Systems*, 2019. URL `https://api.semanticscholar.org/CorpusID:173188850`.

[11] Samyadeep Basu, Xuchen You, and Soheil Feizi. On second-order group influence functions for black-box predictions. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

[12] Han Guo, Nazneen Rajani, Peter Hase, Mohit Bansal, and Caiming Xiong. FastIF: Scalable influence functions for efficient model interpretation and debugging. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10333–10350, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.808. URL `https://aclanthology.org/2021.emnlp-main.808`.

[13] Amirata Ghorbani and James Y. Zou. Data shapley: Equitable valuation of data for machine learning. *ArXiv*, abs/1904.02868, 2019. URL `https://api.semanticscholar.org/CorpusID:102350503`.

[14] R. Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nicholas Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Xiaodong Song, and Costas J. Spanos. Towards efficient data valuation based on the shapley value. *ArXiv*, abs/1902.10275, 2019. URL `https://api.semanticscholar.org/CorpusID:67855573`.

[15] Ruoxi Jia, Fan Wu, Xuehui Sun, Jiacen Xu, David Dao, Bhavya Kailkhura, Ce Zhang, Bo Li, and Dawn Song. Scalability vs. utility: Do we have to sacrifice one for the other in data importance quantification? In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8235–8243, 2021. doi: 10.1109/CVPR46437.2021.00814.

[16] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nezihe Merve Gurel, Bo Li, Ce Zhang, Costas Spanos, and Dawn Song. Efficient task-specific data valuation for nearest neighbor algorithms. *Proc. VLDB Endow.*, 12(11):1610–1623, jul 2019. ISSN 2150-8097. doi: 10.14778/3342263.3342637. URL `https://doi.org/10.14778/3342263.3342637`.

[17] Jeremy Mange. Effect of training data order for machine learning. In *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 406–407, 2019. doi: 10.1109/CSCI49370.2019.00078.

[18] Ernie Chang, Hui-Syuan Yeh, and Vera Demberg. Does the order of training samples matter? improving neural data-to-text generation with curriculum learning. *ArXiv*, abs/2102.03554, 2021. URL `https://api.semanticscholar.org/CorpusID:231846815`.

[19] Yejia Liu, Weiyuan Wu, Lampros Flokas, Jiannan Wang, and Eugene Wu. Enabling sql-based training data debugging for federated learning. *Proceedings of the VLDB Endowment*, 15:388–400, 02 2022. doi: 10.14778/3494124.3494125.

[20] Marc-Etienne Brunet, Colleen Alkalay-Houlihan, Ashton Anderson, and Richard Zemel. Understanding the origins of bias in word embeddings, 2019.

[21] Hao Wang, Berk Ustun, and Flavio P. Calmon. Repairing without retraining: Avoiding disparate impact with counterfactual distributions, 2019.

[22] Xiaotie Deng and Christos H. Papadimitriou. On the complexity of cooperative solution concepts. *Math. Oper. Res.*, 19:257–266, 1994. URL `https://api.semanticscholar.org/CorpusID:12946448`.

[23] Qi Wang, Yue Ma, Kun Zhao, and Yingjie Tian. A comprehensive survey of loss functions in machine learning. *Annals of Data Science*, 9, 04 2022. doi: 10.1007/s40745-020-00253-5.

[24] Erik D. Demaine, William Gasarch, and Mohammad Hajiaghayi. *Computational Intractability: A Guide to Algorithmic Lower Bounds*. MIT Press, 2024.

[25] Alon Jacovi and Yoav Goldberg. Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness? In *Annual Meeting of the Association for Computational Linguistics*, 2020. URL `https://api.semanticscholar.org/CorpusID:215416110`.

[26] Victor Parque. Tackling the subset sum problem with fixed size using an integer representation scheme. In *2021 IEEE Congress on Evolutionary Computation (CEC)*, pages 1447–1453, 2021. doi: 10.1109/CEC45853.2021.9504889.

A Detailed Proofs for Section 3

Notations. Given some orderings $\{o_e\}$ of training data, where $o^e_{\mathtt{t}}$ as the order of $\mathtt{t}$ in epoch e. We use $w^{(e,l)}_{x_i}$ to denote the value of w_{x_i} after the l-th iteration in epoch e. We also denote $\mathbf{x}_{\mathtt{t}}$ and $y_{\mathtt{t}}$ as the feature and the label of training data $\mathtt{t}$, respectively. We denote $\mathtt{t}^{(e,l)}$ as the training sample being considered during epoch e, iteration l.

Lemma A.1. *Suppose $T \subseteq T_0$ is the training data and let $T^e_{l,r} = \{\mathtt{t}^{(e,l)}, \mathtt{t}^{(e,l+1)}, \ldots, \mathtt{t}^{(e,r)}\}$ be the set of consecutive training samples considered during epoch e from iteration l to r. For $1 \le l \le r \le |T|$, if $\mathtt{clause}(\gamma, i_1, i_2, i_3) \notin T^e_{l,r}$, then $w^{(e,l-1)}_{c_\gamma} = w^{(e,r)}_{c_\gamma}$.*

Proof. For each $\mathtt{t} \in T^e_{l,r}$, we have $(x_{\mathtt{t}})_{c_\gamma} = 0$. Therefore

$$\left|\left.\frac{\partial \mathcal{L}}{\partial c_\gamma}\right|_{\mathtt{t}}\right| \le \max\left\{\left|-\frac{12N}{5} y x_{c_\gamma}\right|, |-yx_{c_\gamma}|, \left|-\frac{1}{1000N} y x_{c_\gamma}\right|, 0\right\} = 0$$

Hence $\left.\frac{\partial \mathcal{L}}{\partial c_\gamma}\right|_{\mathtt{t}} = 0$, and

$$w^{(e,r)}_{c_\gamma} = w^{(e,l-1)}_{c_\gamma} - \eta_{c_\gamma} \sum_{\mathtt{t} \in T^e_{l,r}} \left.\frac{\partial \mathcal{L}}{\partial c_\gamma}\right|_{\mathtt{t}} = w^{(e,l-1)}_{c_\gamma}$$

Similarly, $(x_{\mathtt{t}})_{b_\gamma} = 0$, and

$$\left|\left.\frac{\partial \mathcal{L}}{\partial b_\gamma}\right|_{\mathtt{t}}\right| \le \max\left\{\left|-\frac{12N}{5} y x_{b_\gamma}\right|, |-yx_{b_\gamma}|, \left|-\frac{1}{1000N} y x_{b_\gamma}\right|, 0\right\} = 0$$

Hence $\left.\frac{\partial \mathcal{L}}{\partial b_\gamma}\right|_{\mathtt{t}} = 0$, and

$$w^{(e,r)}_{b_\gamma} = w^{(e,l-1)}_{b_\gamma} - \eta_{b_\gamma} \sum_{\mathtt{t} \in T^e_{l,r}} \left.\frac{\partial \mathcal{L}}{\partial b_\gamma}\right|_{\mathtt{t}} = w^{(e,l-1)}_{b_\gamma}$$

□

Lemma A.2. *Suppose $T \subseteq T_0$ is the training data and $T_l := \{\mathtt{t}^{(1,1)}, \ldots, \mathtt{t}^{(1,l)}\}$. $\forall 1 \le i \le n, 1 \le l \le |T|$, $w^{(1,l)}_{x_i} \in U(1, \frac{l+1}{6000N^2})$ if $\mathtt{var}(i) \in T_l$; Otherwise $w^{(1,l)}_{x_i} \in U(-1, \frac{l+1}{6000N^2})$.*

Proof. We prove this lemma by induction.

<u>Basic Case:</u> Note that for all $1 \le i \le n$, $w^{(0)}_{x_i} = -1$, and for all $1 \le \gamma \le m$, $w^{(0)}_{c_\gamma} = 1/2$, $w^{(0)}_{b_\gamma} = -1$. We denote $\mathtt{t} = \mathtt{t}^{(1,1)}$ to avoid cluttering. For any fixed i:

(1) If $\mathtt{t} = \mathtt{var}(i)$. We have $y_{\mathtt{t}}(\mathbf{w}^{(0)})^\top \mathbf{x}'_{\mathtt{t}} = 5w^{(0)}_{x_i} = -5$, hence

$$\left.\frac{\partial \mathcal{L}}{\partial w_{x_i}}\right|_{\mathtt{t}} = -\frac{12N}{5} y_{\mathtt{t}}(x_{\mathtt{t}})_i = -12N$$

and

$$w^{(1,1)}_{x_i} = w^{(0)}_{x_i} - \eta_{x_i} \left.\frac{\partial \mathcal{L}}{\partial w_{x_i}}\right|_{\mathtt{t}} = -1 - \frac{1}{6N}\left(-\frac{12N}{5}\right) = 1 \in U(1, \frac{2}{6000N^2})$$

(2) If $\mathtt{t} = \mathtt{clause}(\gamma, i, i', i'')$. We have

$$y_{\mathtt{t}}(\mathbf{w}^{(0)})^\top \mathbf{x}'_{\mathtt{t}} = w^{(0)}_{x_i} + w^{(0)}_{x_{i'}} + w^{(0)}_{x_{i''}} + w^{(0)}_{c_\gamma} + \frac{1}{2} w^{(0)}_{b_\gamma} = -3$$

hence

$$\left.\frac{\partial \mathcal{L}}{\partial w_{x_i}}\right|_{\mathtt{t}} = -\frac{1}{1000N} y_{\mathtt{t}}(x_{\mathtt{t}})_{x_i} = -\frac{1}{1000N}$$

and

$$w_{x_i}^{(1,1)} = w_{x_i}^{(0)} - \eta_{x_i} \left.\frac{\partial \mathcal{L}}{\partial w_{x_i}}\right|_{\mathbf{t}} = -1 - \frac{1}{6N}\left(-\frac{1}{1000N}\right)$$
$$= -1 + \frac{1}{6000N^2} \in U(-1, \frac{2}{6000N^2})$$

(3) Otherwise, w_{x_i} will not be updated. Therefore $w_{x_i}^{(1,1)} = w_{x_i}^{(0)} = -1 \in U(-1, \frac{2}{6000N^2})$.

Hence this lemma is true for $l = 1$.

Induction Step: Suppose the lemma is true for $l < |T|$. We prove that this lemma remains true for $l+1$. We denote $\mathbf{t} = \mathbf{t}^{(1,l+1)}$ to avoid cluttering. This makes sense since $l+1 \le |T|$ and thus $\mathbf{t} \in T$. For any fixed i:

(1) If $\mathbf{t} =$`var(`i`)`, then `var(`i`)`$\notin T_l$ because there are at most one `var(`i`)` in T for each i. Therefore $w_{x_i}^{(1,l)} \in U(-1, \frac{l+1}{6000N^2})$. We have $y_{\mathbf{t}}(\mathbf{w}^{(1,l)})^\top \mathbf{x}'_{\mathbf{t}} = 5w_{x_i}^{(1,l)} \in U(-5, 0.01)$, and $\left.\frac{\partial \mathcal{L}}{\partial w_{x_i}}\right|_{\mathbf{t}} = -\frac{12N}{5} y_{\mathbf{t}}(x_{\mathbf{t}})_i = -12N$. Hence

$$w_{x_i}^{(1,l+1)} = w_{x_i}^{(1,l)} - \eta_{x_i} \left.\frac{\partial \mathcal{L}}{\partial w_{x_i}}\right|_{\mathbf{t}} = w_{x_i}^{(1,l)} - \frac{1}{6N}\left(-\frac{12N}{5}\right)$$
$$= w_{x_i}^{(1,l)} + 2 \in U(1, \frac{l+2}{6000N^2})$$

(2) If $\mathbf{t} =$`clause(`γ, i, i', i''`)`. In this case, `clause(`$\gamma, \cdot, \cdot, \cdot$`)`$\notin T_{1,l}^1$ and by Lemma A.1 we have $w_{c_\gamma}^{(1,l)} = w_{c_\gamma}^{(0)}, w_{b_\gamma}^{(1,l)} = w_{b_\gamma}^{(0)}$. From the induction hypothesis we have

$$w_{x_i}^{(1,l)}, w_{x_{i'}}^{(1,l)}, w_{x_{i''}}^{(1,l)} \in U(\pm 1, \frac{l+1}{6000N^2})$$

and thus

$$y_{\mathbf{t}}(\mathbf{w}^{(1,l)})^\top \mathbf{x}'_{\mathbf{t}} = w_{x_i}^{(1,l)} + w_{x_{i'}}^{(1,l)} + w_{x_{i''}}^{(1,l)} + w_{c_\gamma}^{(1,l)} + \frac{1}{2} w_{b_\gamma}^{(1,l)}$$
$$= w_{x_i}^{(1,l)} + w_{x_{i'}}^{(1,l)} + w_{x_{i''}}^{(1,l)}$$
$$\in \bigcup_{x_0 \in \{\pm 1, \pm 3\}} U(x_0, \frac{3(l+1)}{6000N^2}) \subseteq \bigcup_{x_0 \in \{\pm 1, \pm 3\}} U(x_0, 0.01)$$

We have $\left.\frac{\partial \mathcal{L}}{\partial w_{x_i}}\right|_{\mathbf{t}} = -\frac{1}{1000N}$ and $w_{x_i}^{(1,l+1)} = w_{x_i}^{(1,l)} - \eta_{x_i} \left.\frac{\partial \mathcal{L}}{\partial w_{x_i}}\right|_{\mathbf{t}} = w_{x_i}^{(1,l)} + \frac{1}{6000N^2}$. Consider the following cases:

- If `var`$(i)\in T_l$, then `var`$(i)\in T_{l+1}$ and $w_{x_i}^{(1,l)} \in U(1, \frac{l+1}{6000N^2})$. Therefore $w_{x_i}^{(1,l+1)} \in U(1, \frac{l+2}{6000N^2})$.
- If `var`$(i)\notin T_l$, then `var`$(i)\notin T_{l+1}$ and $w_{x_i}^{(1,l)} \in U(-1, \frac{l+1}{6000N^2})$. Therefore $w_{x_i}^{(1,l+1)} \in U(-1, \frac{l+2}{6000N^2})$.

(3) Otherwise, w_{x_i} will not be updated, and $w_{x_i}^{(1,l+1)} = w_{x_i}^{(1,l)}$. If `var`$(i)\in T_l$ then `var`$(i)\in T_{l+1}$ and $w_{x_i}^{(1,l+1)} \in U(1, \frac{l+2}{6000N^2})$; Otherwise `var`$(i)\notin T_{l+1}$ and $w_{x_i}^{(1,l+1)} \in U(-1, \frac{l+2}{6000N^2})$.

Hence if the lemma is true for $l < |T|$, it is also true for $l+1$. Therefore, the lemma is true for all $1 \le l \le |T|$. □

Corollary A.1. Suppose $T \subseteq T_0$ is the training data. $\forall 1 \le i \le n, 1 \le l \le |T|$, if `var(`$i$`)`$\in T$, then $w_{x_i}^{(1)} \in U(1, \frac{1}{6000N})$. Otherwise $w_{x_i}^{(1)} \in U(-1, \frac{1}{6000N})$.

Proof. Note that $w_{x_i}^{(1)} = w_{x_i}^{(1,|T|)}$ and $N = 2m + n + 1$. By Lemma A.2, if $\texttt{var}(i) \in T$ we have

$$w_{x_i}^{(1,|T|)} \in U(1, \frac{|T|+1}{6000N^2}) \subseteq U(1, \frac{m+n+1}{6000N^2}) \subseteq U(1, \frac{1}{6000N})$$

If $\texttt{var}(i) \notin T$, we have

$$w_{x_i}^{(1,|T|)} \in U(-1, \frac{|T|+1}{6000N^2}) \subseteq U(-1, \frac{m+n+1}{6000N^2}) \subseteq U(-1, \frac{1}{6000N})$$

$\square$

Lemma A.3. Suppose $T \subseteq T_0$ is the training data. $\forall 1 \leq \gamma \leq m$, if $\exists 1 \leq i_1, i_2, i_3 \leq n$ such that $\texttt{clause}(\gamma, i_1, i_2, i_3) \in T$, then $w_{b_\gamma}^{(1)} = 0, w_{c_\gamma}^{(1)} = \frac{1}{2} + \frac{1}{200N}$; Otherwise, $w_{b_\gamma}^{(1)} = -1, w_{c_\gamma}^{(1)} = \frac{1}{2}$.

Proof. (1) If such $\mathbf{t}_\gamma = \texttt{clause}(\gamma, i_1, i_2, i_3)$ exists in T, by Lemma A.2 we have

$$w_{x_{i_1}}^{(1,o^1_{\mathbf{t}_\gamma})} + w_{x_{i_2}}^{(1,o^1_{\mathbf{t}_\gamma})} + w_{x_{i_3}}^{(1,o^1_{\mathbf{t}_\gamma})} \in \bigcup_{x_0 \in \{\pm 1, \pm 3\}} U(x_0, \frac{3(o^1_{\mathbf{t}_\gamma}+1)}{6000N^2}) \subseteq \bigcup_{x_0 \in \{\pm 1, \pm 3\}} U(x_0, 0.01)$$

By Lemma A.1 we have $w_{c_\gamma}^{(1,o^1_{\mathbf{t}_\gamma}-1)} = w_{c_\gamma}^{(0)}$ and $w_{b_\gamma}^{(1,o^1_{\mathbf{t}_\gamma}-1)} = w_{b_\gamma}^{(0)}$ because $\texttt{clause}(\gamma, \cdot, \cdot, \cdot) \notin T^1_{1, o_{\mathbf{t}_\gamma}-1}$. Hence

$$\begin{aligned} y_{\mathbf{t}_\gamma}(\mathbf{w}^{(1,o^1_{\mathbf{t}_\gamma}-1)})^\top \mathbf{x}'_{\mathbf{t}_\gamma} &= w_{x_{i_1}}^{(1,o^1_{\mathbf{t}_\gamma})} + w_{x_{i_2}}^{(1,o^1_{\mathbf{t}_\gamma})} + w_{x_{i_3}}^{(1,o^1_{\mathbf{t}_\gamma})} + w_{c_\gamma}^{(1,o^1_{\mathbf{t}_\gamma}-1)} + \frac{1}{2} w_{b_\gamma}^{(1,o^1_{\mathbf{t}_\gamma}-1)} \\ &= w_{x_{i_1}}^{(1,o^1_{\mathbf{t}_\gamma})} + w_{x_{i_2}}^{(1,o^1_{\mathbf{t}_\gamma})} + w_{x_{i_3}}^{(1,o^1_{\mathbf{t}_\gamma})} + w_{c_\gamma}^{(1,o^1_{\mathbf{t}_\gamma}-1)} \\ &\in \bigcup_{x_0 \in \{\pm 1, \pm 3\}} U(x_0, 0.01) \end{aligned}$$

We have $\frac{\partial \mathcal{L}}{\partial w_{c_\gamma}}\Big|_{\mathbf{t}_\gamma} = -\frac{1}{1000N}$, and

$$w_{c_\gamma}^{(1,o^1_{\mathbf{t}_\gamma})} = w_{c_\gamma}^{(1,o^1_{\mathbf{t}_\gamma}-1)} - \eta_{c_\gamma} \frac{\partial \mathcal{L}}{\partial w_{c_\gamma}}\Big|_{\mathbf{t}_\gamma} = \frac{1}{2} + 5 \times \frac{1}{1000N} = \frac{1}{2} + \frac{1}{200N}$$

Similarly, $\frac{\partial \mathcal{L}}{\partial w_{b_\gamma}}\Big|_{\mathbf{t}_\gamma} = -\frac{1}{2000N}$ and

$$w_{b_\gamma}^{(1,o^1_{\mathbf{t}_\gamma})} = w_{b_\gamma}^{(1,o^1_{\mathbf{t}_\gamma}-1)} - \eta_{b_\gamma} \frac{\partial \mathcal{L}}{\partial w_{c_\gamma}}\Big|_{\mathbf{t}_\gamma} = -1 - 2000N \times (-\frac{1}{2000N}) = 0$$

Note also that $\texttt{clause}(\gamma, \cdot, \cdot, \cdot) \notin T^1_{o_{\mathbf{t}_\gamma}, |T|}$, by Lemma A.1 we have $w_{c_\gamma}^{(1)} = w_{c_\gamma}^{(1,|T|)} = w_{c_\gamma}^{(1,o^1_{\mathbf{t}_\gamma})} = \frac{1}{2} + \frac{1}{200N}$ and $w_{b_\gamma}^{(1)} = w_{b_\gamma}^{(1,|T|)} = w_{b_\gamma}^{(1,o^1_{\mathbf{t}_\gamma})} = 0$.

(2) If such $\mathbf{t}_\gamma = \texttt{clause}(\gamma, i_1, i_2, i_3)$ does not exist in T, by Lemma A.1 we have $w_{c_\gamma}^{(1)} = w_{c_\gamma}^{(0)} = \frac{1}{2}$ and $w_{b_\gamma}^{(1)} = w_{b_\gamma}^{(0)} = -1$. $\square$

Lemma A.4. Suppose $T \subseteq T_0$ and C_l be the number of $\texttt{clause}()$ in $T^2_{1,l}$. $\forall 1 \leq i \leq n, 1 \leq l \leq |T|$, $w_{x_i}^{(2,l)} \in U(1, \frac{C_l + 1/2}{6N})$ if $\texttt{var}(i) \in T$; Otherwise $w_{x_i}^{(2,l)} \in U(-1, \frac{C_l+1/2}{6N})$.

Proof. Similar to the proof of A.2, we prove this lemma by induction.

<u>Basic Case:</u> Note that for all $1 \leq i \leq n$, $w_{x_i}^{(1)} = U(\pm 1, \frac{1}{6000N})$, and for all $1 \leq \gamma \leq m, w_{c_\gamma}^{(1)} \in \{\frac{1}{2}, \frac{1}{2} + \frac{1}{200N}\}, w_{b_\gamma}^{(1)} \in \{-1, 0\}$. We denote $\mathbf{t} = \mathbf{t}^{(2,1)}$ to avoid cluttering. For any fixed i:

(1) If $\mathbf{t} = \texttt{var}(i)$, $C_1 = 0$. By Corollary A.1, $w_{x_i}^{(1)} = U(1, \frac{1}{6000N})$. We have

$$y_{\mathbf{t}}(\mathbf{w}^{(1)})^\top \mathbf{x}'_{\mathbf{t}} = 5 w_{x_i}^{(1)} \in U(5, \frac{1}{1200N})$$

hence $\frac{\partial \mathcal{L}}{\partial w_{x_i}}\Big|_{\mathbf{t}} = 0$, and

$$w_{x_i}^{(2,1)} = w_{x_i}^{(1)} \in U(1, \frac{1}{6N}) = U(1, \frac{C_l + 1/2}{6N})$$

(2) If $\mathbf{t} =$ `clause(`γ, i, i', i''`)`, $C_1 = 1$. By Lemma A.3, we have $w_{c_\gamma}^{(1)} = \frac{1}{2} + \frac{1}{200N}$ and $w_{b_\gamma}^{(1)} = 0$. Therefore,

$$\begin{aligned} y_{\mathbf{t}}(\mathbf{w}^{(1)})^\top \mathbf{x}'_{\mathbf{t}} &= w_{x_i}^{(1)} + w_{x_{i'}}^{(1)} + w_{x_{i''}}^{(1)} + w_{c_\gamma}^{(1)} + \frac{1}{2} w_{b_\gamma}^{(1)} \\ &= w_{x_i}^{(1)} + w_{x_{i'}}^{(1)} + w_{x_{i''}}^{(1)} + \frac{1}{2} - \frac{1}{200N} \\ &\in \bigcup_{x_0 \in \{\frac{1}{2} \pm 1, \frac{1}{2} \pm 3\}} U(x_0, 0.01) \end{aligned}$$

hence $\frac{\partial \mathcal{L}}{\partial w_{x_i}}\Big|_{\mathbf{t}} \in \{0, -yx_{x_i}\} = \{-1, 0\}$, and $\eta_{x_i} \frac{\partial \mathcal{L}}{\partial w_{x_i}}\Big|_{\mathbf{t}} \in \{-\frac{1}{6N}, 0\}$.

By Corollary A.1, if `var(`i`)`$\in T$, we have

$$w_{x_i}^{(2,1)} = w_{x_i}^{(1)} - \eta_{x_i} \frac{\partial \mathcal{L}}{\partial w_{x_i}}\Big|_{\mathbf{t}} \in U(1, \frac{3/2}{6N}) = U(1, \frac{C_l + 1/2}{6N})$$

If `var(`i`)`$\notin T$, we have

$$w_{x_i}^{(2,1)} = w_{x_i}^{(1)} - \eta_{x_i} \frac{\partial \mathcal{L}}{\partial w_{x_i}}\Big|_{\mathbf{t}} \in U(-1, \frac{3/2}{6N}) = U(-1, \frac{C_l + 1/2}{6N})$$

(3) Otherwise, w_{x_i} will not be updated and $C_1 \leq 1$. Therefore if `var(`i`)`$\in T$,

$$w_{x_i}^{(2,1)} = w_{x_i}^{(1)} \in U(1, \frac{3/2}{6N}) \subseteq U(1, \frac{C_l + 1/2}{6N})$$

If `var(`i`)`$\notin T$,

$$w_{x_i}^{(2,1)} = w_{x_i}^{(1)} \in U(-1, \frac{3/2}{6N}) \subseteq U(-1, \frac{C_l + 1/2}{6N})$$

Hence this lemma is true for $l = 1$.

<u>Induction Step:</u> Suppose the lemma is true for $l < |T|$. We prove that this lemma remains true for $l + 1$. We denote $\mathbf{t} = \mathbf{t}^{(2,l+1)}$ to avoid cluttering. This makes sense since $l + 1 \leq |T|$ and thus $\mathbf{t} \in T$. For any fixed i:

(1) If $\mathbf{t} =$ `var(`i`)`, $C_{l+1} = C_l$. By Corollary A.1, $w_{x_i}^{(2,l)} \in U(1, \frac{C_l+1/2}{6N})$.

We have $y_{\mathbf{t}}(\mathbf{w}^{(2,l)})^\top \mathbf{x}'_{\mathbf{t}} = 5w_{x_i}^{(2,l)} \in U(5, 1/6)$ and $\frac{\partial \mathcal{L}}{\partial w_{x_i}}\Big|_{\mathbf{t}} = 0$.Hence $w_{x_i}^{(2,l+1)} = w_{x_i}^{(2,l)} \in U(1, \frac{C_{l+1}+1/2}{6N})$.

(2) If $\mathbf{t} =$ `clause(`γ, i, i', i''`)`, $C_{l+1} = C_l + 1$. In this case, `clause(`$\gamma, \cdot, \cdot, \cdot$`)`$\notin T_{1,l}^2$ and by Lemma A.1 and Lemma A.3 we have $w_{c_\gamma}^{(2,l)} = w_{c_\gamma}^{(1)} = \frac{1}{2} + \frac{1}{200N}, w_{b_\gamma}^{(2,l)} = w_{b_\gamma}^{(1)} = 0$. From the induction hypothesis we have $w_{x_i}^{(2,l)}, w_{x_{i'}}^{(2,l)}, w_{x_{i''}}^{(2,l)} \in U(\pm 1, \frac{C_l+1/2}{6N})$. Noting that

$$\frac{C_l + 1/2}{6N} \leq \frac{m + 1/2}{6N} = \frac{m + 1/2}{(n + 2(m + 1/2))} \leq \frac{1}{12}$$

we have

$$\begin{aligned} y_{\mathbf{t}}(\mathbf{w}^{(2,l)})^\top \mathbf{x}'_{\mathbf{t}} &= w_{x_i}^{(2,l)} + w_{x_{i'}}^{(2,l)} + w_{x_{i''}}^{(2,l)} + w_{c_\gamma}^{(2,l)} + \frac{1}{2} w_{b_\gamma}^{(2,l)} \\ &= w_{x_i}^{(2,l)} + w_{x_{i'}}^{(2,l)} + w_{x_{i''}}^{(2,l)} + \frac{1}{2} + \frac{1}{200N} \\ &\in \bigcup_{x_0 \in \{\frac{1}{2} \pm 1, \frac{1}{2} \pm 3\}} U\left(x_0, \frac{3(C_l + 1/2)}{6N} + \frac{1}{200N}\right) \\ &\subseteq \bigcup_{x_0 \in \{\frac{1}{2} \pm 1, \frac{1}{2} \pm 3\}} U(x_0, 0.26) \end{aligned}$$

And thus $\frac{\partial \mathcal{L}}{\partial w_{x_i}}\Big|_{\mathbf{t}} \in \{0, -yx_{x_i}\} = \{-1, 0\}$, and $\eta_{x_i} \frac{\partial \mathcal{L}}{\partial w_{x_i}}\Big|_{\mathbf{t}} \in \{-\frac{1}{6N}, 0\}$.

By Corollary A.1, if $\texttt{var}(i) \in T$, $w_{x_i}^{(2,l+1)} = w_{x_i}^{(l)} - \eta_{x_i} \frac{\partial \mathcal{L}}{\partial w_{x_i}}\Big|_{\mathbf{t}} \in U(1, \frac{C_l+3/2}{6N}) = U(1, \frac{C_{l+1}+1/2}{6N})$; if $\texttt{var}(i) \notin T$, $w_{x_i}^{(2,l+1)} = w_{x_i}^{(l)} - \eta_{x_i} \frac{\partial \mathcal{L}}{\partial w_{x_i}}\Big|_{\mathbf{t}} \in U(-1, \frac{C_l+3/2}{6N}) = U(-1, \frac{C_{l+1}+1/2}{6N})$.

(3) Otherwise, w_{x_i} will not be updated. We have $C_{l+1} \leq C_l + 1$ $w_{x_i}^{(2,l+1)} = w_{x_i}^{(2,l)}$. If $\texttt{var}(i) \in T$ then $w_{x_i}^{(2,l+1)} \in U(1, \frac{C_{l+1}+1/2}{6N})$; If $\texttt{var}(i) \notin T$ then $w_{x_i}^{(2,l+1)} \in U(-1, \frac{C_{l+1}+1/2}{6N})$.

Hence if the lemma is true for $l < |T|$, it is also true for $l+1$. Therefore, the lemma is true for all $1 \leq l \leq |T|$. $\square$

Corollary A.2. Suppose $T \subseteq T_0$ is the training data. $\forall 1 \leq i \leq n$, if $\texttt{var}(i) \in T$, then $w_{x_i}^{(2)} \in U(1, 0.1)$. Otherwise $w_{x_i}^{(2)} \in U(-1, 0.1)$.

Proof. Note that $w_{x_i}^{(2)} = w_{x_i}^{(2,|T|)}$ and $C_{|T|} \leq m$. By Lemma A.4, if $\texttt{var}(i) \in T$ we have

$$w_{x_i}^{(2,|T|)} \in U(1, \frac{C_{|T|}+1/2}{6N}) \subseteq U(1, \frac{m+1/2}{6N}) \subseteq U(1, \frac{1}{12}) \subseteq U(1, 0.1)$$

If $\texttt{var}(i) \notin T$, we have

$$w_{x_i}^{(1,|T|)} \in U(-1, \frac{C_{|T|}+1/2}{6N}) \subseteq U(-1, \frac{m+1/2}{6N}) \subseteq U(-1, \frac{1}{12}) \subseteq U(-1, 0.1)$$

$\square$

Lemma A.5. Suppose $T \subseteq T_0$ is the training data. $\forall 1 \leq i \leq m$, if $\exists 1 \leq i_1, i_2, i_3 \leq n$ such that $\texttt{clause}(i, i_1, i_2, i_3) \in T$, then

1. $w_{b_j}^{(2)} = 1000N$;
2. $w_{c_j}^{(2)} = \frac{11}{2} + \frac{1}{200N}$ if exactly one of $\texttt{var}(i_1)$, $\texttt{var}(i_2)$, $\texttt{var}(i_3)$ is in T. Otherwise $w_{c_j}^{(2)} = \frac{1}{2} + \frac{1}{200N}$.

Otherwise, $w_{b_i}^{(2)} = -1, w_{c_i}^{(2)} = \frac{1}{2}$.

Proof. (1) If such $\mathbf{t}_\gamma = \texttt{clause}(\gamma, i_1, i_2, i_3)$ exists in T, by Lemma A.4 we have

$$w_{x_{i_1}}^{(2,o^1_{\mathbf{t}_\gamma})}, w_{x_{i_2}}^{(2,o^1_{\mathbf{t}_\gamma})}, w_{x_{i_3}}^{(2,o^1_{\mathbf{t}_\gamma})} \in U(\pm 1, \frac{m+1/2}{6N}) \subseteq U(\pm 1, \frac{1}{12N})$$

By Lemma A.1 we have $w_{c_\gamma}^{(2,o^1_{\mathbf{t}_\gamma}-1)} = w_{c_\gamma}^{(1)} = \frac{1}{2} + \frac{1}{200N}$ and $w_{b_\gamma}^{(2,o^1_{\mathbf{t}_\gamma}-1)} = w_{b_\gamma}^{(1)} = 0$ because $\texttt{clause}(\gamma, \cdot, \cdot, \cdot) \notin T^1_{1, o_{\mathbf{t}_\gamma}-1}$. Consider the following two cases:

(a) If exactly one of $\texttt{var}(i_1)$, $\texttt{var}(i_2)$, $\texttt{var}(i_3)$ is in T, by Corollary A.2 we have

$$\begin{aligned} y_{\mathbf{t}_\gamma}(\mathbf{w}^{(2,o^1_{\mathbf{t}_\gamma}-1)})^\top \mathbf{x}'_{\mathbf{t}_\gamma} &= w_{x_{i_1}}^{(2,o^1_{\mathbf{t}_\gamma}-1)} + w_{x_{i_2}}^{(2,o^1_{\mathbf{t}_\gamma}-1)} + w_{x_{i_3}}^{(2,o^1_{\mathbf{t}_\gamma}-1)} + w_{c_\gamma}^{(2,o^1_{\mathbf{t}_\gamma}-1)} + \frac{1}{2} w_{b_\gamma}^{(2,o^1_{\mathbf{t}_\gamma}-1)} \\ &= w_{x_{i_1}}^{(2,o^1_{\mathbf{t}_\gamma}-1)} + w_{x_{i_2}}^{(2,o^1_{\mathbf{t}_\gamma}-1)} + w_{x_{i_3}}^{(2,o^1_{\mathbf{t}_\gamma}-1)} + \frac{1}{2} + \frac{1}{200N} \\ &\in U(-\frac{1}{2}, \frac{3}{12N} + \frac{1}{200N}) \subseteq U(-\frac{1}{2}, 0.26) \end{aligned}$$

Hence $\frac{\partial \mathcal{L}}{\partial w_{c_\gamma}}\Big|_{\mathbf{t}_\gamma} = -1$, and

$$w_{c_\gamma}^{(2,o^1_{\mathbf{t}_\gamma})} = w_{c_\gamma}^{(2,o^1_{\mathbf{t}_\gamma}-1)} - \eta_{c_\gamma} \frac{\partial \mathcal{L}}{\partial w_{c_\gamma}}\Big|_{\mathbf{t}_\gamma} = \frac{1}{2} + \frac{1}{200N} + 5 = \frac{11}{2} + \frac{1}{200N}$$

Similarly,

$$w_{b_\gamma}^{(2,o^1_{\mathbf{t}_\gamma})} = w_{b_\gamma}^{(2,o^1_{\mathbf{t}_\gamma}-1)} - \eta_{b_\gamma} \left.\frac{\partial \mathcal{L}}{\partial w_{b_\gamma}}\right|_{\mathbf{t}_\gamma} = 1000N$$

Note also that $\texttt{clause}(\gamma,\cdot,\cdot,\cdot) \notin T^1_{o_{\mathbf{t}_\gamma},|T|}$, by Lemma A.1 we have $w_{c_\gamma}^{(2)} = w_{c_\gamma}^{(2,|T|)} = w_{c_\gamma}^{(2,o^1_{\mathbf{t}_\gamma})} = \frac{11}{2} - \frac{1}{200N}$ and $w_{b_\gamma}^{(2)} = w_{b_\gamma}^{(2,|T|)} = w_{b_\gamma}^{(2,o^1_{\mathbf{t}_\gamma})} = 1000N$.

(b) Otherwise, we have

$$\begin{aligned}
y_{\mathbf{t}_\gamma}(\mathbf{w}^{(2,o^1_{\mathbf{t}_\gamma}-1)})^\top \mathbf{x}'_{\mathbf{t}_\gamma} &= w_{x_{i_1}}^{(2,o^1_{\mathbf{t}_\gamma}-1)} + w_{x_{i_2}}^{(2,o^1_{\mathbf{t}_\gamma}-1)} + w_{x_{i_3}}^{(2,o^1_{\mathbf{t}_\gamma}-1)} + w_{c_\gamma}^{(2,o^1_{\mathbf{t}_\gamma}-1)} + \frac{1}{2} w_{b_\gamma}^{(2,o^1_{\mathbf{t}_\gamma}-1)} \\
&= w_{x_{i_1}}^{(2,o^1_{\mathbf{t}_\gamma}-1)} + w_{x_{i_2}}^{(2,o^1_{\mathbf{t}_\gamma}-1)} + w_{x_{i_3}}^{(2,o^1_{\mathbf{t}_\gamma}-1)} + \frac{1}{2} + \frac{1}{200N} \\
&\in \bigcup_{x_0 \in \{-\frac{7}{2},\frac{1}{2},\frac{5}{2}\}} U(x_0, \frac{3}{12N} + \frac{1}{200N}) \subseteq \bigcup_{x_0 \in \{-\frac{7}{2},\frac{1}{2},\frac{5}{2}\}} U(x_0, 0.26)
\end{aligned}$$

Hence $\left.\frac{\partial \mathcal{L}}{\partial w_{c_\gamma}}\right|_{\mathbf{t}_\gamma} = \left.\frac{\partial \mathcal{L}}{\partial w_{b_\gamma}}\right|_{\mathbf{t}_\gamma} = 0$, so $w_{c_\gamma}^{(2,o^1_{\mathbf{t}_\gamma})} = w_{c_\gamma}^{(2,o^1_{\mathbf{t}_\gamma}-1)} = \frac{1}{2} + \frac{1}{200N}, w_{b_\gamma}^{(2,o^1_{\mathbf{t}_\gamma})} = w_{b_\gamma}^{(2,o^1_{\mathbf{t}_\gamma}-1)} = 0$.

Note also that $\texttt{clause}(\gamma,\cdot,\cdot,\cdot) \notin T^1_{o_{\mathbf{t}_\gamma},|T|}$, by Lemma A.1 we have $w_{c_\gamma}^{(2)} = w_{c_\gamma}^{(2,|T|)} = w_{c_\gamma}^{(2,o^1_{\mathbf{t}_\gamma})} = \frac{1}{2} + \frac{1}{200N}$ and $w_{b_\gamma}^{(2)} = w_{b_\gamma}^{(2,|T|)} = w_{b_\gamma}^{(2,o^1_{\mathbf{t}_\gamma})} = 0$.

(2) If such $\mathbf{t}_\gamma = \texttt{clause}(\gamma, i_1, i_2, i_3)$ does not exist in T, by Lemma A.1 and Lemma A.3 we have $w_{c_\gamma}^{(2)} = w_{c_\gamma}^{(1)} = \frac{1}{2}$ and $w_{b_\gamma}^{(2)} = w_{b_\gamma}^{(1)} = -1$. □

Moreover, $\mathbf{w}$ reaches its fixpoint at the end of the second epoch and will no longer be updated.

Lemma A.6. $\mathbf{w}^{(2)} = \mathbf{w}^{(3)}$.

Proof. Suppose $\mathbf{w}^{(2)} \neq \mathbf{w}^{(3)}$, then there exists $1 \leq i \leq N$ such that $w_i^{(2)} \neq w_i^{(3)}$, and there are some training sample $\mathbf{t}$ in the training data such that $\left.\frac{\partial \mathcal{L}}{\partial w_i^{(2)}}\right|_{\mathbf{t}} \neq 0$. Let $\mathbf{t} = (\mathbf{x_t}, y_\mathbf{t})$ and $\mathbb{I} = U(-5, 0.01) \cup U(-\frac{1}{2}, 0.26) \cup \left(\bigcup_{x_0 \in \{\pm 1, \pm 3\}} U(x_0, 0.01)\right)$. By (2) we have $y_\mathbf{t}(\mathbf{w}^{(2)})^\top \mathbf{x_t}' \in \mathbb{I}$. At least one of the following is true:

1. $\exists 1 \leq i \leq n, \mathbf{t} = \texttt{var}(i)$. According to lemma A.2, $y_\mathbf{t}(\mathbf{w}^{(2)})^\top \mathbf{x_t}' = y w_{x_i}^{(2)} x_i \in U(5, 0.5) \subseteq \mathbb{R} \setminus \mathbb{I}$, contradicting to $y_\mathbf{t}(\mathbf{w}^{(2)})^\top \mathbf{x_t}' \in \mathbb{I}$.
2. $\exists 1 \leq i \leq m$ and $1 \leq i_1, i_2, i_3 \leq n$, such that $\mathbf{t} = \texttt{clause}(i, i_1, i_2, i_3)$. According to lemma A.5, we have

$$\begin{aligned}
y_\mathbf{t}(\mathbf{w}^{(2)})^\top \mathbf{x_t}' &= w_{b_i}^{(2)} + w_{c_i}^{(2)} + w_{x_{i_1}}^{(2)} + w_{x_{i_2}}^{(2)} + w_{x_{i_3}}^{(2)} \\
&\geq 1000N + \frac{1}{2} + \frac{1}{200N} + 3 \times (-1 - 0.1) \\
&\geq 1000 - 3.3 \geq 996
\end{aligned}$$

We have $y_\mathbf{t}(w^{(2)})^\top \mathbf{x_t}' \notin \mathbb{I}$, another contradiction.

Therefore $\mathbf{w}^{(2)} = \mathbf{w}^{(3)}$, $\mathbf{w}$ reaches its fixpoint at the end of the second epoch. In other words, $\mathbf{w}^* = \mathbf{w}^{(2)}$. □

We are now ready to give a rigorous proof of theorem 3.1.

Proof of theorem 3.1. It only suffices to prove the correctness of the reduction in section 3.

<u>If.</u> Suppose $\varphi \in$MONOTONE 1-IN-3 SAT, then there is a truth assignment $\nu(\cdot)$ that assigns exactly one variable in each clause of φ is true. Let $\Delta = \{\texttt{var}(i) | \nu(x_i) = \text{FALSE}\}$. Let $\mathbf{w}'$ be the parameter of $\texttt{SGD}_\Lambda(T_0 \setminus \Delta)$. By Lemma A.5, $(w')_{c_\gamma} = \frac{11}{2} + \frac{1}{200N}$ for all $1 \leq \gamma \leq m$, hence

$$(\mathbf{w}')^\top \mathbf{x}_{\text{test}} = \sum_{\gamma=1}^{m} w'_{c_\gamma} \geq \frac{11m}{2} + \frac{-11m+5}{2} = \frac{5}{2} > 0$$

and $\lambda_{\mathbf{w}'}(\mathbf{x}_{\text{test}}) = 1$, thus $\texttt{SGD}_\Lambda(T_0)$ is thus debuggable.

<u>Only if.</u> Suppose $\texttt{SGD}_\Lambda(T_0)$ is debuggable, there will be a Δ such that $\texttt{SGD}_\Lambda(T_0, \mathbf{x}_{\text{test}}) = y_{\text{test}}$. We denote $\mathbf{w}'$ as the parameter trained by $\texttt{SGD}$ on $T_0 \setminus \Delta$. We have $\lambda_{\mathbf{w}'}(\mathbf{x}_{\text{test}}) = 1$ and $(\mathbf{w}')^\top \mathbf{x}_{\text{test}} \geq 0$. By Lemma A.5, $w'_{c_\gamma} = \{\frac{1}{2} + \frac{1}{200N}, \frac{11}{2} + \frac{1}{200N}\}$. Suppose $w_{c^*} = \frac{1}{2} + \frac{1}{200N}$, then

$$\begin{aligned}
(\mathbf{w}')^\top \mathbf{x}_{\text{test}} &= w_{c^*} + \sum_{c_\gamma \neq c^*} w_{c_\gamma} \\
&\leq \frac{11}{2}(m-1) + \frac{1}{2} + \frac{m}{200N} - \frac{11m}{2} + \frac{5}{2} \\
&= -\frac{5}{2} + \frac{m}{200N} \\
&\leq -\frac{5}{2} + \frac{1}{200} = -2.495 < 0
\end{aligned}$$

leading to a contradiction.

As a consequence, $w'_{c_\gamma} = \frac{11}{2} + \frac{1}{200N}$ for all $1 \leq \gamma \leq m$. By Lemma A.5, exactly one of $\texttt{var}(i_1)$,$\texttt{var}(i_2)$,$\texttt{var}(i_3)$ is in $T_0 \setminus \Delta$ for each $c_\gamma = (x_{i_1} \vee x_{i_2} \vee x_{i_3})$. Consider a truth assignment ν that maps every x_i to FALSE where $\texttt{var}(i) \in \Delta$, and maps the rest to TRUE. Then ν assigns exactly one variable true in each $c_\gamma = (x_{i_1} \vee x_{i_2} \vee x_{i_3})$ if and only if exactly one of $\texttt{var}(i_1)$,$\texttt{var}(i_2)$,$\texttt{var}(i_3)$ is in $T_0 \setminus \Delta$. Hence ν is a truth assignment that assigns true to exactly one variable in each clause of φ, and thus φ is a yes-instance of MONOTONE 1-IN-3 SAT. $\square$

B Detailed Proofs for Section 4

B.1 Proof of Theorem 4.4

Proof. We build a reduction from the SUBSET SUM problem with a fixed size, which is NP-hard as a particular case of the class of knapsack problems [26]. Formally, it is defined as:

> SUBSET SUM with a fixed size
> **Input:** A set of positive integer S, and two positive integers t, k.
> **Output:** "Yes": if $\exists S' \subseteq S$ of size k such that $\sum_{a \in S'} a = t$;
> "No": otherwise.

The ordered training data T is constructed as

$$T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \cup \{(x_a, y_a)\}$$

where $x_i y_i = \frac{2}{3} + \frac{a_i}{3\sum_{a \in S} a}$ for all $1 \leq i \leq n$ and $x_a y_a = 1 + \frac{1}{6\sum_{a \in S} a}$. Let $\eta = 1, \alpha = 1, \beta = -1$, $w^{(0)} = -1 - \frac{2}{3}k - \frac{t}{3\sum_{a \in S} a}$ and let the test instance $(x_{\text{test}}, y_{\text{test}})$ satisfy $x_{\text{test}} y_{\text{test}} = 1$. It now suffices to prove that $\exists S' \subseteq S$ such that $|S'| = k$ and $\sum_{a \in S'} a = t$ if and only if $\exists T' \subseteq T$ such that $w : w^{(0)} \xrightarrow{T'} w$ satisfies $y_{\text{test}} w x_{\text{test}} > 0$.

<u>If:</u> Suppose $\exists S' \subseteq S$ such that $|S'| = k$ and $\sum_{a \in S} a = t$. Let $T^* = \{(x_i, y_i) | a_i \in S'\}$, we prove that $y_{\text{test}} w^* x_{\text{test}} > 0$ for w^* satisfying $w^{(0)} \xrightarrow{T' = T^* \cup \{(x_a, y_a)\}} w^*$.

Since

$$w^{(0)} + \sum_{a_i \in S'} x_i y_i = -1 - \frac{2}{3}k - \frac{t}{3\sum_{a\in S} a} + \sum_{a_i \in S'} \left(\frac{2}{3} + \frac{a_i}{3\sum_{a\in S} a}\right)$$
$$= -1 - \frac{2}{3}k - \frac{t}{3\sum_{a\in S} a} + \sum_{a_i \in S'} \frac{2}{3} + \frac{\sum_{a\in S'} a}{3\sum_{a\in S} a} = -1$$

and $\forall 1 \le i \le n, x_i y_i > \frac{2}{3}$, for each $1 \le i < n$, suppose $w^{(0)} \xrightarrow{T^* \cap \{(x_j,y_j)|1\le j\le i\}} w_i$, we have

$$w_i x_{i+1} y_{i+1} < \left(w^{(0)} + \sum_{a_j \in S'} x_j y_j - \frac{2}{3}\right) \cdot \frac{2}{3} < -\frac{10}{9} < \beta.$$

That is, each training sample in T^* is activated. Then for $w^{(0)} \xrightarrow{T^*} w_a$, we have $w_a = -1$. Then, since $y_a w_a x_a = -(1 + \frac{1}{6\sum_{a\in S} a}) < \beta$ and $w_a \xrightarrow{(x_a,y_a)} w^*$ we have $w^* = w_a + x_a y_a = \frac{1}{6\sum_{a\in S} a}$. Therefore, $y_{\text{test}} w^* x_{\text{test}} = \frac{1}{6\sum_{a\in S} a} > 0$.

Only if: For each $T' \subseteq T$, let $T^* = T' \setminus \{(x_a, y_a)\}$ and $c(T^*)$ be the set of training samples in T^* that are activated. If $y_{\text{test}} w^* x_{\text{test}} \ge 0$ for w^* satisfying $w^{(0)} \xrightarrow{T'} w^*$, we prove that the set $S' = \{a_i | (x_i, y_i) \in c(T^*)\}$ satisfies $|S'| = k$ and $\sum_{a\in S'} a = t$.

We first show that $y_{\text{test}} w_a x_{\text{test}} < 0$ for $w^{(0)} \xrightarrow{c(T^*)} w_a$. Otherwise, suppose $y_{\text{test}} w_a x_{\text{test}} \ge 0$ we have $w_a \ge 0$. Let (x, y) be the last training sample of $c(T')$, since $\frac{2}{3} < xy \le 1$, we have $w' \ge w_a - xy \ge -1$ for $w' \xrightarrow{(x,y)} w_a$. Thus $yw'x \ge \beta$, which contradicts to the definition of $c(T^*)$.

We next show that $|S'| = k$. Suppose $|S'| \le k - 1$, we have

$$w_a = w^{(0)} + \sum_{(x_i,y_i)\in c(T^*)} x_i y_i = -1 - \frac{2}{3}k - \frac{t}{3\sum_{a\in S} a} + \sum_{a_i \in S'} \frac{2}{3} + \frac{\sum_{a\in S'} a}{3\sum_{a\in S} a}$$
$$< -1 - \frac{2}{3}k + \frac{2}{3}(k-1) + \frac{1}{3} = -\frac{4}{3}$$

Thus $w^* \le w_a + x_a y_a < -\frac{4}{3} + (1 + \frac{1}{6\sum_{a\in S} a}) < 0$ and then $y_{\text{test}} w^* x_{\text{test}} < 0$, which contradicts to the fact that $y_{\text{test}} w^* x_{\text{test}} \ge 0$. Therefore $|S'| \ge k$.

Suppose $|S'| \ge k + 1$, we have

$$w_a = w^{(0)} + \sum_{(x_i,y_i)\in c(T^*)} x_i y_i \ge -1 - \frac{2}{3}k - \frac{1}{3} + \frac{2}{3}(k+1) = -\frac{2}{3}$$

Then $y_a w_a x_a \ge (-\frac{2}{3}) \cdot (1 + \frac{1}{6\sum_{a\in S} a}) \ge -\frac{7}{9} \ge \beta$, that is, (x_a, y_a) is not activated and $w^* = w_a$. Then since $y_{\text{test}} w_a x_{\text{test}} < 0$, we have $y_{\text{test}} w^* x_{\text{test}} = y_{\text{test}} w_a x_{\text{test}} < 0$, which contradicts to the fact that $y_{\text{test}} w^* x_{\text{test}} \ge 0$. Therefore $|S'| = k$.

It remains to prove that $\sum_{a\in S'} a = t$. Otherwise, suppose $\sum_{a\in S'} a \le t - 1$, we have

$$w_a = w^{(0)} + \sum_{(x_i,y_i)\in c(T^*)} x_i y_i \le -1 - \frac{2}{3}k - \frac{t}{3\sum_{a\in S} a} + \frac{2}{3}k + \frac{t-1}{3\sum_{a\in S} a}$$
$$= -1 - \frac{1}{3\sum_{a\in S} a}$$

Thus $y_{\text{test}} w^* x_{\text{test}} \le y_{\text{test}}(w_a + x_a y_a) x_{\text{test}} \le -\frac{1}{6\sum_{a\in S} a} < 0$, which contradicts to the fact that $y_{\text{test}} w^* x_{\text{test}} \ge 0$. Therefore $\sum_{a\in S'} a \ge t$.

Suppose $\sum_{a\in S'} a \ge t + 1$ we have

$$w_a = w^{(0)} + \sum_{(x_i,y_i)\in c(T^*)} x_i y_i \ge -1 - \frac{2}{3}k - \frac{t}{3\sum_{a\in S} a} + \frac{2}{3}k + \frac{t+1}{3\sum_{a\in S} a}$$
$$= -1 + \frac{1}{3\sum_{a\in S} a}$$

Thus

$$y_a w_a x_a \geq (-1 + \frac{1}{3\sum_{a\in S} a}) \cdot (1 + \frac{1}{6\sum_{a\in S} a})$$
$$\geq -1 + \frac{1}{6\sum_{a\in S} a} + \frac{1}{18(\sum_{a\in S} a)^2} \geq \beta.$$

That is, (x_a, y_a) is not activated and $w^* = w_a$. Then since $y_{\text{test}} w_a x_{\text{test}} < 0$, we have $y_{\text{test}} w^* x_{\text{test}} = y_{\text{test}} w_a x_{\text{test}} < 0$, which contradicts to the fact that $y_{\text{test}} w^* x_{\text{test}} \geq 0$. Therefore $\sum_{a\in S'} a = t$. $\square$

B.2 Proof of Theorem 4.3 for $\beta < -1$

Proof. To avoid cluttering, we still assume $\boldsymbol{\eta} = \mathbf{1}$ and $\alpha = 1$. The proof can be generalized by appropriately re-scaling the constructed vectors.

Let $M = -\beta(n+2) + 9\beta n m^2(n+1) + 3$. Suppose $n = |S| > 1$, $m = \max_{a\in S}\{a\}$ and $S = \{a_1, a_2, \ldots, a_n\}$. We further assume $n > 1$. Let the ordered set of training samples be

$$T = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\} \cup \{(\mathbf{x}_c, y_c), (\mathbf{x}_b, y_b), (\mathbf{x}_a, y_a)\}$$

where $\mathbf{x}_i y_i = (\frac{1}{n+1}, -3\beta a_i)$ for all $1 \leq i \leq n, \mathbf{x}_c y_c = (M + \frac{3}{2}\beta - 1, \beta(3t - \frac{1}{2})), \mathbf{x}_b y_b = (1, -1), \mathbf{x}_a y_a = (-\frac{3}{2}\beta, -\frac{3}{2}\beta)$. Let $\mathbf{w}^{(0)} = (-M, 0)$. Let the test instance $(\mathbf{x}_{\text{test}}, y_{\text{test}})$ satisfy $\mathbf{x}_{\text{test}} y_{\text{test}} = (1, 0)$.

For each $1 \leq i < n$, suppose $\mathbf{w}^{(0)} \xrightarrow{T\cap\{(\mathbf{x}_i, y_i)|1\leq j\leq i\}} \mathbf{w}_i$, we have

$$y_{i+1}\mathbf{w}_i^\top \mathbf{x}_{i+1} \leq -M \cdot \frac{1}{n+1} + \frac{i}{(n+1)^2} + 9\beta^2 a_{i+1} \sum_{j=1}^{i} a_j$$
$$\leq -M \cdot \frac{1}{n+1} + \frac{n}{(n+1)^2} + 9\beta^2 n m^2 < \beta$$

This means all the $(\mathbf{x}_i, y_i) \in T \setminus \{(\mathbf{x}_c, y_c), (\mathbf{x}_b, y_b), (\mathbf{x}_a, y_a)\}$ can be activated and thus the resulting parameter trained by $T \setminus \{(\mathbf{x}_c, y_c), (\mathbf{x}_b, y_b), (\mathbf{x}_a, y_a)\}$ is

$$\mathbf{w}_c = \mathbf{w}^{(0)} + \sum_{i=1}^{n} \mathbf{x}_i y_i = \left(-M + \frac{|T^*|}{n+1}, -3\beta \sum_{i=1}^{n} a_i\right)$$

It now suffices to prove that for all $S' \subseteq S$, $\sum_{a\in S'} a = t$ if and only if $\exists T' \subseteq T$ such that $\mathbf{w} : \mathbf{w}^{(0)} \xrightarrow{T'} \mathbf{w}$ such that $y_{\text{test}} \mathbf{w}^\top \mathbf{x}_{\text{test}} > 0$.

<u>If:</u> Suppose $\exists S' \subseteq S$ such that $\sum_{a\in S} a = t$, we prove that $\exists T' \subseteq T$ such that $y_{\text{test}}(\mathbf{w}^*)^\top \mathbf{x}_{\text{test}} > 0$ for $\mathbf{w}^*$ satisfying $\mathbf{w}^{(0)} \xrightarrow{T^*} \mathbf{w}^*$.

Let $T^* = \{(\mathbf{x}_i, y_i) | a_i \in S'\}, T' = T^* \cup \{(\mathbf{x}_c, y_c), (\mathbf{x}_b, y_b), (\mathbf{x}_a, y_a)\}$. We have

$$\mathbf{w}_c = (-M + \frac{|T^*|}{n+1}, -3\beta \sum_{a_i \in S'} a_i) = (-M + \frac{|T^*|}{n+1}, -3\beta t)$$

And $y_c \mathbf{w}_c^\top \mathbf{x}_c = (-M + \frac{|T^*|}{n+1})(M + \frac{3}{2}\beta - 1) - 3t\beta^2(3t - \frac{1}{2}) < \beta$, so

$$\mathbf{w}_c \xrightarrow{(\mathbf{x}_c, y_c)} \mathbf{w}_b = \mathbf{w}_c + \mathbf{x}_c y_c = (\frac{|T^*|}{n+1} + \frac{3}{2}\beta - 1, -\frac{1}{2}\beta)$$

Note that $\beta < -1$, we have $y_b \mathbf{w}_b^\top \mathbf{x}_b = \frac{|T^*|}{n+1} + 2\beta < (\beta + \frac{|T^*|}{n+1}) + \beta < \beta$, and

$$\mathbf{w}_b \xrightarrow{(\mathbf{x}_b, y_b)} \mathbf{w}_a = \mathbf{w}_b + \mathbf{x}_a y_a = (\frac{|T^*|}{n+1} + \frac{3}{2}\beta, -\frac{1}{2}\beta - 1)$$

Note also that $y_a \mathbf{w}_a^\top \mathbf{x}_a = \frac{3}{2}(-\beta)(\frac{|T^*|}{n+1} - 1 + \beta) < \beta$, we have

$$\mathbf{w}_a \xrightarrow{(\mathbf{x}_a, y_a)} \mathbf{w}^* = \mathbf{w}_a + \mathbf{x}_a y_a = (\frac{|T^*|}{n+1}, -2\beta - 1)$$

Therefore, $y_{\text{test}}(\mathbf{w}^*)^\top \mathbf{x}_{\text{test}} = \frac{|T^*|}{n+1} \geq 0$.

<u>Only if:</u> For each $T' \subseteq T$, let $T^* = T' \setminus \{(\mathbf{x}_c, y_c), (\mathbf{x}_b, y_b), (\mathbf{x}_a, y_a)\}$, if $y_{\text{test}}(\mathbf{w}^*)^\top \mathbf{x}_{\text{test}}$ for $\mathbf{w}^*$ satisfying $\mathbf{w}^{(0)} \xrightarrow{T'} \mathbf{w}^*$, we prove that $\exists S' \subseteq S$ such that $\sum_{a \in S'} a = t$. We first show that for each $T' \subseteq T$, if $\mathbf{w}(\mathbf{w}^{(0)} \xrightarrow{T'} \mathbf{w})$ satisfying $y_{\text{test}}\mathbf{w}^\top \mathbf{x}_{\text{test}} \geq 0$, we have $\forall k \in \{a, b, c\}, (\mathbf{x}_k, y_k) \in T', y_k \mathbf{w}_k^\top \mathbf{x}_k < \beta$, where $\mathbf{w}^{(0)} \xrightarrow{T^*} \mathbf{w}_c \xrightarrow{(\mathbf{x}_c, y_c)} \mathbf{w}_b \xrightarrow{(\mathbf{x}_b, y_b)} \mathbf{w}_a$. Otherwise, suppose $\exists k \in \{a, b, c\}$ such that $(\mathbf{x}_k, y_k) \notin T'$ or $y_k \mathbf{w}_k^\top \mathbf{x}_k \geq \beta$, we have

$$\begin{aligned} y_{\text{test}}\mathbf{w}^\top \mathbf{x}_{\text{test}} &\leq -M + \frac{|T^*|}{n+1} + M + \frac{3}{2}\beta - 1 + 1 - \frac{3}{2}\beta - \min\left\{1, M + \frac{3}{2}\beta - 1, -\frac{3}{2}\beta\right\} \\ &= \frac{|T^*|}{n+1} - 1 < 0 \end{aligned}$$

which contradicts to the fact that $y_{\text{test}}\mathbf{w}^\top \mathbf{x}_{\text{test}} \geq 0$.

Let $S' = \{a_i | (\mathbf{x}_i, y_i) \in T^*\}$ and $t' = \sum_{a \in S'} a_i$, it suffices to prove $t' = t$. Notice that

$$\begin{aligned} \mathbf{w}^{(0)} \xrightarrow{T^*} \mathbf{w}_c &= (-M + \frac{|T^*|}{n+1}, -3\beta \sum_{a_i \in S'} a_i) \\ &= (-M + \frac{|T^*|}{n+1}, -3\beta t') \end{aligned}$$

Hence $y_c \mathbf{w}_c^\top \mathbf{x}_c = (-M + \frac{|T^*|}{n+1})(M + \frac{3}{2}\beta - 1) - 3t'\beta^2(3t - \frac{1}{2}) < \beta$, thus

$$\mathbf{w}_c \xrightarrow{(\mathbf{x}_c, y_c)} \mathbf{w}_b = \mathbf{w}_c + \mathbf{x}_c y_c = (\frac{|T^*|}{n+1} + \frac{3}{2}\beta - 1, -3\beta(t' - t) - \frac{1}{2}\beta)$$

(1) If $t' \leq t - 1$, we have

$$\begin{aligned} y_b \mathbf{w}_b^\top \mathbf{x}_b &= \frac{|T^*|}{n+1} - 1 + 2\beta + 3\beta(t' - t) \\ &\geq \frac{|T^*|}{n+1} - (1 + \beta) > 0 > \beta \end{aligned}$$

a contradiction. Hence $\mathbf{w}_a = \mathbf{w}_b \xrightarrow{(\mathbf{x}_b, y_b)} \mathbf{w}_a = (\frac{|T^*|}{n+1} + \frac{3}{2}\beta, -3\beta(t' - t) - \frac{1}{2}\beta - 1)$.

(2) If $t' \geq t + 1$, we have

$$\begin{aligned} y_a \mathbf{w}_a^\top \mathbf{x}_a &= -\frac{3\beta}{2}\left(\frac{|T^*|}{n+1} - 1 + \beta - 3\beta(t' - t)\right) \\ &\geq -\frac{3\beta}{2}\left(\frac{|T^*|}{n+1} - 1 - 2\beta\right) \\ &> -\frac{3\beta}{2}\left(\frac{|T^*|}{n+1} + 1\right) > 0 > \beta \end{aligned}$$

another contradiction. Therefore $t' = t$, and this completes the proof.

$\square$

C Limitations

It is important to emphasize that the complexity results in section 4 requires the training order to be adversarially chosen. The complexity of DEBUGGABLE for randomly chosen training order is unclear and needs to be figured out in the future research.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main results are discussed in section 3 and section 4.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See section C in the appendix.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The proof of theorem 3.1 is available in section A; The proof of theorem 4.1 and theorem 4.2 are available in section 4; The proof of theorem 4.3 is available in section 4 and section B; The proof of theorem 4.4 is available in section B.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [NA]

Justification: This paper does not include experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [NA]

Justification: This paper does not include experiments requiring code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [NA]

Justification: This paper does not include experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [NA]

Justification: This paper does not include experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [NA]

   Justification: This paper does not include experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: They authors have made sure that the research conducted in the paper conform with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: The impacts are discussed in section 5

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This paper only provides theoretical results and poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [NA]

Justification: This paper does not use existing assets.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: This paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.