AN ANALYSIS OF HINGE LOSS FUNCTIONS FOR LINEAR MULTI-CLASS CLASSIFICATION

Assignment 1, CS-587

Author
Nikolaos Barmparousis
csd4690
March 3, 2024

Contents

1 Introduction

This report outlines the implementation and evaluation of a linear classifier for the CIFAR-10 dataset, leveraging Mini-Batch Stochastic Gradient Descent (SGD). The dataset comprises vectorized images across ten categories, forming a collection of N pairs $(\mathbf{x}_i, y_i)$, where $\mathbf{x}_i$ represents the i-th image and $y_i \in \{0, 1, 2, \ldots, 9\}$ its class label. The classifier is modeled as $f(\mathbf{x}; \mathbf{W}, \mathbf{b}) = \mathbf{W}\mathbf{x} + \mathbf{b}$, with $\mathbf{W} \in \mathbb{R}^{10 \times D}$ and $\mathbf{b} \in \mathbb{R}^{10}$.

To optimize the classifier, we aim to minimize the regularized empirical loss function:

$$L(\mathbf{W}, \mathbf{b}) = \lambda R(\mathbf{W}) + \frac{1}{N} \sum_{i=1}^{N} \text{loss}(f(\mathbf{x}_i; \mathbf{W}, \mathbf{b}), y_i), \tag{1}$$

where λ denotes regularization strength, and $R(\mathbf{W})$ represents the regularization term, either L_1 or L_2.

The update step of our iterative algorithm is based on the following formulas:

$$\mathbf{W} \leftarrow \mathbf{W} - \gamma \left(\lambda \nabla_{\mathbf{W}} R(\mathbf{W}) + \frac{1}{N} \sum_{i=1}^{N} \nabla_{\mathbf{W}} \text{loss}(f(\mathbf{x}_i; \mathbf{W}, \mathbf{b}), y_i) \right), \tag{2}$$

$$\mathbf{b} \leftarrow \mathbf{b} - \gamma \left(\frac{1}{N} \sum_{i=1}^{N} \nabla_{\mathbf{b}} \text{loss}(f(\mathbf{x}_i; \mathbf{W}, \mathbf{b}), y_i) \right). \tag{3}$$

At the heart of our inquiry lies an exploration of three distinct implementations of the multi-class hinge loss function:

1. The **Sum Hinge Loss**, accumulating margin violations across all non-target classes.
2. The **Max-Max Hinge Loss**, prioritizing the rectification of the most significant margin violation.
3. The **Normalized Hinge Loss**, a novel concoction aiming to harmonize the sum and max-max strategies by dividing the sum of all violations by the maximum violation, thereby imbuing each term with relative significance.

We delve into the mathematical derivation of these loss functions, aiming to discern their impact on model performance through gradient computation and hyperparameter tuning. This endeavor seeks to enhance our understanding of loss function behavior in the context of linear classification, guiding the selection of an optimal approach for the CIFAR-10 challenge.

2 Applicable Hinge Loss Functions

Optimizing our linear classifier for the CIFAR-10 dataset crucially hinges on the loss function selection, with hinge loss variants standing out for their effectiveness in multi-class scenarios due to margin-enforcing properties. This section delves into three specific hinge loss implementations: Sum Hinge Loss, Max-Max Hinge Loss, and Normalized Hinge Loss. We dissect their mathematical expressions and explore their similarities and differences.

2.1 Sum Hinge Loss (Weston and Watkins [1])

Sum Hinge Loss accumulates margin violations for each incorrect class, formulated as:

$$L_i(s, y_i) = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + \Delta), \tag{4}$$

where s_j represents the score for an incorrect class, and s_{y_i} is the score for the correct class y_i. This approach ensures a clear margin from all incorrect classes, facilitating correct class identification.

2.2 Max-Max Hinge Loss (Crammer and Singer [2])

Contrarily, Max-Max Hinge Loss emphasizes the most significant margin violation:

$$L_i(s, y_i) = \max(0, \max_{j \neq y_i}(s_j) - s_{y_i} + \Delta). \tag{5}$$

This loss function prioritizes the correction of the most egregious classification error, focusing on the competitive incorrect class score.

2.3 Normalized Hinge Loss

Normalized Hinge Loss seeks equilibrium between Sum and Max-Max strategies by normalizing the sum of violations with the maximum violation, thus:

$$L_i(s, y_i) = \frac{\sum_{j \neq y_i} \max(0, s_j - s_{y_i} + \Delta)}{\max(0, \max_{j \neq y_i}(s_j) - s_{y_i} + \Delta)}. \tag{6}$$

This proposed formulation balances comprehensive and focused margin enforcement strategies, assigning relative importance to each violation.

2.4 Similarities and Differences

Although all three hinge loss functions aim to enforce a margin between the correct class score and the incorrect ones, they employ varied strategies for addressing margin violations. The Sum and Normalized Hinge Losses consider all violations but differ in their weighting approach, with the latter providing a balanced perspective by considering the severity of violations. The Max-Max Hinge Loss, however, zeroes in on the most critical violation, offering a targeted yet potentially narrower scope for margin enforcement.

3 Gradient Derivation for Hinge Loss Functions

We now explore the derivation of gradients for the Sum, Max-Max, and Normalized Hinge Loss functions, key to optimizing our linear classifier on the CIFAR-10 dataset. We will derive the gradients for a single training sample and by using vectorized implementations, we can extend this to our mini-batch approach.

3.1 Gradient of the Sum Hinge Loss

The loss function, L_i, for a single data point i with the true class y_i, sums the violations when the predicted margin falls short of a specified threshold, Δ. It is formally defined as:

$$L_i = \sum_{j \neq y_i} \left[\max(0, \mathbf{w}_j^\top \mathbf{x}_i - \mathbf{w}_{y_i}^\top \mathbf{x}_i + \Delta) \right]. \tag{7}$$

The gradient of this loss with respect to the weights associated with the correct class, $\mathbf{w}_{y_i}$, considers the sum of the margin violations and is given by:

$$\nabla_{\mathbf{w}_{y_i}} L_i = -\left(\sum_{j \neq y_i} \mathbb{1}\left[\mathbf{w}_j^\top \mathbf{x}_i - \mathbf{w}_{y_i}^\top \mathbf{x}_i + \Delta > 0 \right] \right) \cdot \mathbf{x}_i, \quad \square$$

where $\mathbb{1}$ is the indicator function that is 1 if the condition inside is true, and 0 otherwise.

For weights corresponding to the incorrect classes, $\mathbf{w}_j$ where $j \neq y_i$, the gradient reflects the contribution of each class that violates the margin:

$$\nabla_{\mathbf{w}_j} L_i = \mathbb{1}\left[\mathbf{w}_j^\top \mathbf{x}_i - \mathbf{w}_{y_i}^\top \mathbf{x}_i + \Delta > 0 \right] \mathbf{x}_i. \quad \square$$

The same principle applies to the bias terms, b_{y_i} and b_j, for the true and incorrect classes, respectively. However, instead of scaling by the vectorized input $\mathbf{x}_i$, the update for the bias terms simply involves incrementing or decrementing by 1, corresponding to the presence or absence of a violation.

For the true class bias b_{y_i}, the gradient is:

$$\nabla_{b_{y_i}} L_i = -\left(\sum_{j \neq y_i} \mathbb{1}\left[\mathbf{w}_j^\top \mathbf{x}_i - \mathbf{w}_{y_i}^\top \mathbf{x}_i + \Delta > 0 \right] \right) \cdot 1, \quad \square$$

and for the incorrect class biases b_j where $j \neq y_i$, the gradient is:

$$\nabla_{b_j} L_i = \mathbb{1}\left[\mathbf{w}_j^\top \mathbf{x}_i - \mathbf{w}_{y_i}^\top \mathbf{x}_i + \Delta > 0 \right] \cdot 1. \quad \square$$

Both the formulas for the gradients as well as the explanation behind their derivation, were taken from Stanford's CS231n class notes [3].

3.2 Gradient of the Max-Max Hinge Loss

The hinge loss function for a single data point is defined as:

$$L_i = \max\left(0, \max_{j \neq y_i} \left(\mathbf{w}_j^\mathsf{T} \mathbf{x}_i\right) - \mathbf{w}_{y_i}^\mathsf{T} \mathbf{x}_i + \Delta\right)$$

To compute the gradient of the loss with respect to the weights, we define two functions:

$$\begin{aligned} f(z) &= \max(0, z) \\ g(\mathbf{w}) &= \max_{j \neq y_i} \left(\mathbf{w}_j^\mathsf{T} \mathbf{x}_i\right) - \mathbf{w}_{y_i}^\mathsf{T} \mathbf{x}_i + \Delta \end{aligned}$$

The loss function can then be expressed as a composition of these two functions:

$$L_i(\mathbf{w}) = f(g(\mathbf{w}))$$

Using the chain rule, we can obtain the gradient of L_i with respect to $\mathbf{w}$ by calculating the derivative of the outer function $f(z)$ and multiplying it by the derivative of the inner function $g(\mathbf{w})$. For the outer function $f(z)$, the derivative with respect to its input z is:

$$\frac{\partial f}{\partial z} = \begin{cases} 1 & \text{if } z > 0, \\ 0 & \text{otherwise.} \end{cases}$$

The inner function $g(\mathbf{w})$ with respect to the weights has two cases:

1. For the correct class weights $\mathbf{w}_{y_i}$, the gradient is:

$$\frac{\partial g}{\partial \mathbf{w}_{y_i}} = -\mathbf{x}_i$$

2. For the incorrect class weights $\mathbf{w}_j$, where $j \neq y_i$:

   First, we denote j^* as the class with the largest violation, satisfying:

$$\mathbf{w}_{j^*}^\mathsf{T} \mathbf{x}_i > \mathbf{w}_k^\mathsf{T} \mathbf{x}_i, \qquad \forall k \in \{0, \ldots, 9\} \setminus \{y_i, j^*\}. \tag{8}$$

$$\frac{\partial g}{\partial \mathbf{w}_j} = \begin{cases} \mathbf{x}_i, & \text{if } j = j^*, \\ 0, & \text{otherwise.} \end{cases}$$

Combining these derivatives using the chain rule, the gradient of the loss $L_i(\mathbf{w})$ can be expressed as follows:

- For $\mathbf{w}_{y_i}$:

$$\nabla_{\mathbf{w}_{y_i}} L_i = \frac{\partial f}{\partial z} \cdot \frac{\partial g}{\partial \mathbf{w}_{y_i}} = \begin{cases} -\mathbf{x}_i & \text{if } \mathbf{w}_{j^*}^\mathsf{T} \mathbf{x}_i - \mathbf{w}_{y_i}^\mathsf{T} \mathbf{x}_i + \Delta > 0, \\ \mathbf{0} & \text{otherwise.} \end{cases} \quad \square$$

- For $\mathbf{w}_j$, $j \neq y_i$:

$$\nabla_{\mathbf{w}_j} L_i = \frac{\partial f}{\partial z} \cdot \frac{\partial g}{\partial \mathbf{w}_j} = \begin{cases} \mathbf{x}_i & \text{if } j = j^* \wedge \mathbf{w}_{j^*}^\mathsf{T} \mathbf{x}_i - \mathbf{w}_{y_i}^\mathsf{T} \mathbf{x}_i + \Delta > 0, \\ \mathbf{0} & \text{otherwise.} \end{cases} \quad \square$$

The gradients for the bias terms are computed analogously to those of the weights:

- For the true class bias b_{y_i}, we have:

$$\nabla_{b_{y_i}} L_i = \begin{cases} -1 & \text{if } \mathbf{w}_{j^*}^\mathsf{T} \mathbf{x}_i - \mathbf{w}_{y_i}^\mathsf{T} \mathbf{x}_i + \Delta > 0, \\ 0 & \text{otherwise.} \end{cases} \quad \square$$

- For the bias term corresponding to any other class b_j :

$$\nabla_{b_j} L_i = \begin{cases} 1 & \text{if } j = j^* \wedge \mathbf{w}_{j^*}^\mathsf{T} \mathbf{x}_i - \mathbf{w}_{y_i}^\mathsf{T} \mathbf{x}_i + \Delta > 0, \\ 0 & \text{otherwise.} \end{cases} \quad \square$$

3.3 Gradient of the Normalized Hinge Loss

The gradient of the Normalized Hinge Loss is derived using the product rule from calculus, as the loss itself is a product of the Sum Hinge Loss and the reciprocal of the Max-Max Hinge Loss. It is expressed as follows:

$$L_i(s, y_i) = L_{\text{sum}}(s, y_i) \cdot \left(\frac{1}{L_{\max}(s, y_i)} \right), \tag{9}$$

where L_{sum} represents the Sum Hinge Loss, and $L_{\max}$ represents the Max-Max Hinge Loss. Utilizing the product rule, the gradient with respect to the weights $\mathbf{w}$ is given by:

$$\nabla_{\mathbf{w}} L_i = \nabla_{\mathbf{w}} L_{\text{sum}} \cdot \left(\frac{1}{L_{\max}} \right) - L_{\text{sum}} \cdot \left(\frac{\nabla_{\mathbf{w}} L_{\max}}{L_{\max}^2} \right). \tag{10}$$

Considering the cases of y_i, j, and j^*:

1. For $\mathbf{w}_j$, with $j \in \{0, \ldots, 9\} \setminus \{y_i, j^*\}$, the gradient is given by:

$$\nabla_{\mathbf{w}_j} L_i = \begin{cases} \frac{\mathbf{x}_i}{s_{j^*} - s_{y_i} + \Delta} \cdot \mathbb{1}\left(s_j - s_{y_i} + \Delta > 0\right), & \text{if } s_{j^*} - s_{y_i} + \Delta > 0, \\ 0, & \text{otherwise.} \end{cases} \quad \square$$

   We observe that this is the gradient of $\mathbf{w}_j$,$j \neq y_i$, for the case of the simple Sum-Max Loss Function, scaled by the biggest violation.

2. For $\mathbf{w}_{y_i}$:

   Let us first denote the term:

$$C = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + \Delta) - \sum_{j \neq y_i} \mathbb{1}\left(s_j - s_{y_i} + \Delta > 0\right) \cdot (s_{j^*} - s_{y_i} + \Delta) \tag{11}$$

as the difference between the sum of all positive violations for the incorrect classes $j \neq y_i$ and the product of the number of these violations by the maximum violation.

With C defined, the gradient with respect to the weights corresponding to the correct class y_i, can be expressed more cleanly as follows:

$$\nabla_{\mathbf{w}_{y_i}} L_i = \begin{cases} \frac{\mathbf{x}_i}{(s_{j^*} - s_{y_i} + \Delta)^2} \cdot C, & \text{if } s_{j^*} - s_{y_i} + \Delta > 0, \\ 0, & \text{otherwise.} \end{cases} \quad \square$$

It can be shown, that C is always non-positive, meaning that the gradient step will always be non-positive. As in the previous case, the gradient w.r.t y_i, is simply the gradient of y_i of the Max-Max Loss Function, scaled by a factor of $\mathrm{C}\frac{}{(s_{j^*} - s_{y_i} + \Delta)^2.}$

It should be noted, that a problem arises, when the max violation is the only violation, since, the gradient step of y_i will be 0 in that case. This is something that must be taken into consideration, if we are to refine this proposed Loss Function more.

Before presenting the gradient formula for $\mathbf{w}_{j^*}$, let us denote the ratio of the sum of all positive violations for the incorrect classes $j \neq y_i$ to the maximum violation for class j^* by R, where:

$$R = \frac{\sum_{j \neq y_i} \max(0, s_j - s_{y_i} + \Delta)}{s_{j^*} - s_{y_i} + \Delta} \tag{12}$$

Now, we can use R in the gradient formula for $\mathbf{w}_{j^*}$ as follows:

$$\nabla_{\mathbf{w}_{j^*}} L_i = \begin{cases} \frac{\mathbf{x}_i}{s_{j^*} - s_{y_i} + \Delta} \cdot (1 - R), & \text{if } s_{j^*} - s_{y_i} + \Delta > 0, \\ 0, & \text{otherwise.} \end{cases} \quad \square$$

We observe that this is very similar to the gradient of the other $\mathbf{w}_j$ classes, but this time subtracted by a ratio R.

This is rather problematic, since we are updating the maximum violation by a factor that is definitely smaller then the factor that will be used to update the non-max violations. We should also note that, since the sum of all violations contains the max violation itself, $R \geq 1$, thus x_i will be scaled by a non-positive value.

Even though the gradient checking of this derivation was somewhat satisfactory, see Table 1, intuitively this update term can be considered troublesome. As seen in the Results Section 5, the performance of the proposed Loss Function using this gradient term, is significantly worse compared to the other 2 already established Loss Functions.

Thus, we present a new gradient term for $\mathbf{w}_j$, which while on principle may sound logical, is derived from pure intuition, and is not based on any mathematical analysis.

The revised gradient term for $\mathbf{w}_{j^*}$, considering an adjustable penalization factor, is given by:

$$\nabla_{\mathbf{w}_{j^*}} L_i = \begin{cases} \frac{\mathbf{x}_i}{s_{j^*} - s_{y_i} + \Delta} \cdot (1 \pm (1 - R)), & \text{if } s_{j^*} - s_{y_i} + \Delta > 0, \\ 0, & \text{otherwise.} \end{cases} \quad \square$$

In this formulation:

- Choosing the positive sign in $(1 + (1 - R))$ results in a lesser penalization for the class with the maximum violation, since for $R \geq 1$, it follows that $1 - R \leq 0$ and thus $1 + (1 - R) \leq 1$.
- Conversely, opting for the negative sign in $(1 - (1 - R))$ leads to a greater penalization for the maximum violation class, because $R \geq 1$ implies $R - 1 \geq 0$, hence $1 - (1 - R) \geq 1$.

This heuristic approach adjusts the gradient step for $\mathbf{w}_{j^*}$ to either alleviate or intensify the penalization, which is not grounded in mathematical proof but is inspired by practical observations and intuition aimed at enhancing the classifier's performance.

3.4 Gradient of the Regularizations

Our model employs L1 and L2 regularization terms to penalize the magnitude of the weights, applying only to the weights and not to the bias terms. The gradients for these regularization terms are as follows:

For L1 regularization (Lasso):

$$R(\mathbf{W}) = \sum_{i,j} |\mathbf{W}_{i,j}|, \tag{13}$$

with the gradient given by the signum function:

$$\nabla_{\mathbf{W}} R(\mathbf{W}) = \text{sign}(\mathbf{W}). \tag{14}$$

For L2 regularization (Ridge):

$$R(\mathbf{W}) = \sum_{i,j} \mathbf{W}_{i,j}^2, \tag{15}$$

where the gradient is:

$$\nabla_{\mathbf{W}} R(\mathbf{W}) = 2\mathbf{W}. \tag{16}$$

3.5 Numerically Gradient Checking Derivations

Gradient checking [4] is a procedure used to verify the correctness of the analytical gradients by comparing them with numerical approximations. We performed gradient checking on our three original loss functions as well as on the two revised formulations for the Custom Hinge Loss, across a development set consisting of 500 images. The findings are summarized in Table 1, which includes the relative gradient error and the general loss values for each loss function.

Loss Type	**Relative Gradient Error**	**Loss Value**
Sum Hinge Loss	5.3149×10^{-4}	9.1141
Max-Max Hinge Loss	8.9221×10^{-4}	1.4856
Custom Hinge Loss	1.4778×10^{-3}	6.0906
Custom Hinge Loss +(1 - R)	1.1578×10^{-1}	6.1680
Custom Hinge Loss -(1 - R)	9.1163×10^{-1}	5.8481

Table 1: Numerical Gradient Checking Results

4 Setup

In this assignment, we conducted experiments on the CIFAR-10 dataset, which consists of 60,000 32x32 color images in 10 different classes. A linear classifier was trained using three distinct loss functions—Sum Hinge Loss, Max-Max Hinge Loss, and Custom Hinge Loss—to recognize and classify these images accurately.

To optimize our models, we performed hyperparameter search across all three loss functions. We considered a combination of eight different settings, as shown in Table 2.

Learning Rate	**Regularization Strength**	**Regularization Type**	**Batch Size**
1×10^{-8}	1×10^{4}	L1	50
1×10^{-7}	3×10^{4}	L2	100
3×10^{-7}	5×10^{4}	L1	200
3×10^{-7}	1×10^{4}	L2	400
5×10^{-7}	8×10^{4}	L1	100
8×10^{-7}	1×10^{5}	L2	200
1×10^{-6}	5×10^{4}	L1	200
1×10^{-5}	5×10^{5}	L2	400

Table 2: Hyperparameter Combinations

5 Results

In this section, we compare performance outcomes for established Sum-Max and Max-Max loss functions on the CIFAR-10 dataset, along with three versions of the Normalized-Sum Hinge Loss. We highlight their best hyperparameter configurations and examine how each variant affects classifier performance. Additionally, we include visualizations of the learned weights for each loss function, shedding light on the classifier's feature discrimination.

5.1 Comparing the Established Loss Functions

A visual comparison of the best-performing configurations for both, Sum/Max of all violations, loss functions is provided in Figure 1. Notably, both loss functions exhibit competitive performance, with Configuration 4 emerging as the best setting for each. Configuration 4 corresponds to a learning rate of 3×10^{-7}, a regularization strength of 1×10^{4}, employing L2 regularization, and a batch size of 400. Although the results are closely matched, the Sum-Max loss demonstrates a slight edge over the Max-Max loss. The Sum-Max function scored a 0.366 on the unseen test set, while the Max-Max function scored 0.314.

5.2 Comparing the Norm-Sum Loss Functions

In our experiments, three variants of the Norm-Sum loss function were evaluated: the original Norm-Sum, the Norm-Sum with the term $(1 + (1 - R))$, and the Norm-Sum with the term $(1 - (1 - R))$.

The experimental results revealed that both variations of the original Norm-Sum hinge loss function significantly outperformed the initial formulation. Furthermore, when compared against each other, the two revised loss functions exhibited highly competitive per-

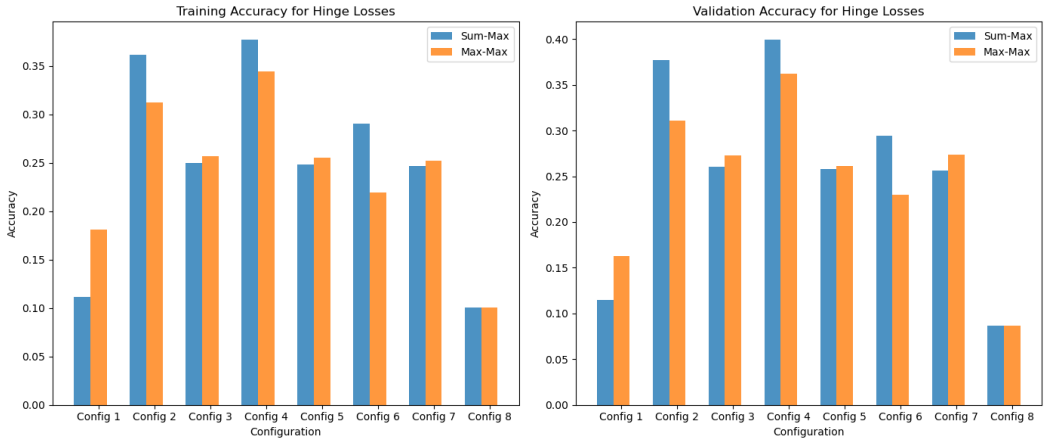


Figure 1: Training and validation accuracy comparison between Sum-Max and Max-Max hinge losses.

formance. Notably, Configuration 4 proved to be the most effective across the 2 variations of the Norm-Sum Functions.

The bar plots in Figure 2 display the training and validation accuracy for all three Norm-Sum loss function variations, illustrating the aforementioned performance trends. The $+(1-R)$ loss function had a score of 0.368 on the test set, while the $-(1-R)$ had the highest score of all the loss functions tested, at 0.376. Though, it should be noted that a more in depth analysis is required, in order to give weight to the results.

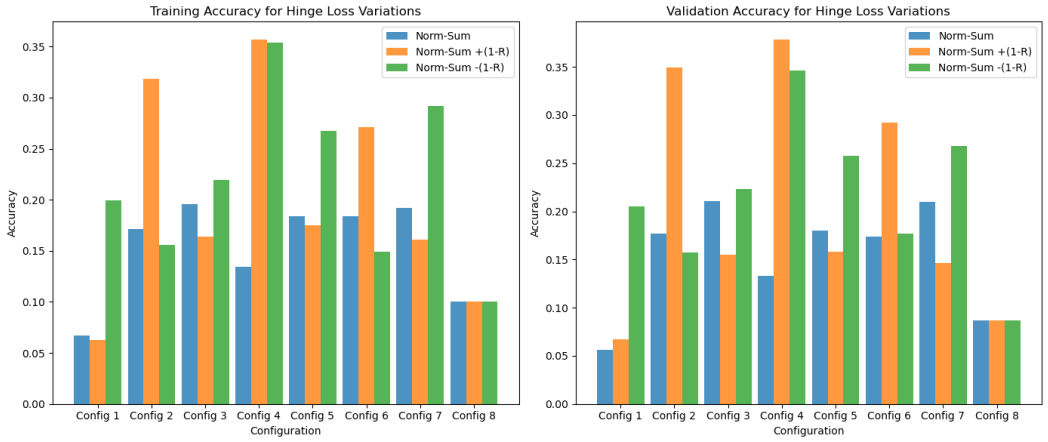
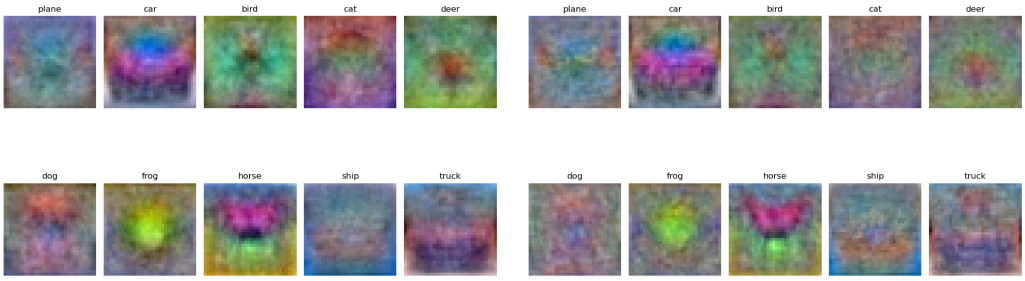


Figure 2: Training and validation accuracy comparison for the three Norm-Sum Hinge Loss variations.

5.3 Visualizing the Learned Weights

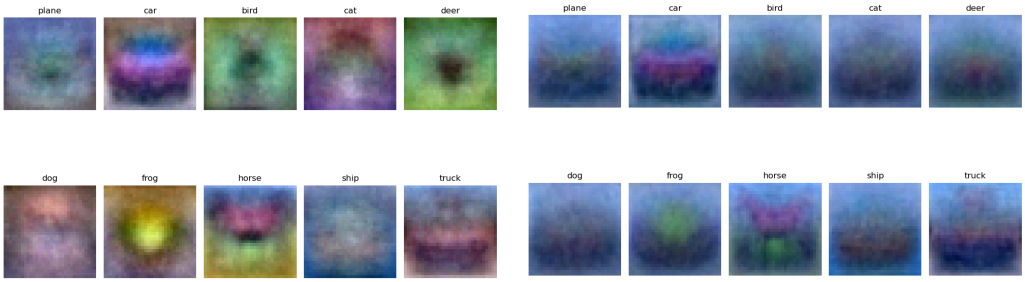
The visual representation of the learned weight vectors offers valuable insight into the features that the classifiers are using to make predictions. Presented here are the visualizations of the weight vectors from classifiers trained with different loss functions: Sum-Max and Max-Max in figure 3, the two variations of Norm-Sum in figure 4 and finally the original

Norm-Sum in figure 5. Each image reflects the template or pattern that the classifier has learned to associate with a particular class.



(a) Weights for the Sum-Max hinge loss (b) Weights of Max-Max hinge loss

Figure 3: Classifier's learned weights for the 2 established loss functions



(a) Weights for the Norm-Sum +(1 - R) hinge loss (b) Weights for the Norm-Sum -(1 - R) hinge loss

Figure 4: Classifier's learned weights for the 2 variations of the Norm-Sum loss function

Figure 5: Learned weights visualization for the original Norm-Sum hinge loss.

These visualizations are not only indicative of the differences in learning behavior induced by each loss function but also may shed light on the underlying biases towards certain features within the CIFAR-10 dataset. It is interesting to note the distinct patterns and color schemes that each set of weights embodies, reflecting the diverse approaches to capturing discriminative features for classification.

6 Summary

In this report, we addressed the classification challenge on the CIFAR-10 dataset using a linear classifier enhanced with various hinge loss functions. We examined two established loss functions, Sum and Max of all violations, and proposed a new variant, the Normalized Sum of Violations loss function, alongside two adaptations to improve its performance metrics.

While the new loss functions show promise, their theoretical underpinnings and practical efficacy warrant further exploration. A deeper analysis, encompassing both mathematical scrutiny and diverse practical scenarios, will be crucial to fully understand and harness the potential of these innovative approaches to loss functions in machine learning tasks.

References

[1] J. Weston and C. Watkins, "Support vector machines for multi-class pattern recognition," *Proc of the 7th European Sympo-sium On Artificial Neural Networks*, pp. 219–224, 01 1999.

[2] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," *J. Mach. Learn. Res.*, vol. 2, p. 265–292, mar 2002.

[3] "Cs231n: Convolutional neural networks for visual recognition." [Online]. Available: https://cs231n.github.io/optimization-1/#analytic

[4] "Numerically gradient checking." [Online]. Available: http://ufldl.stanford.edu/tutorial/supervised/DebuggingGradientChecking/