Sub-quadratic AUC Optimization

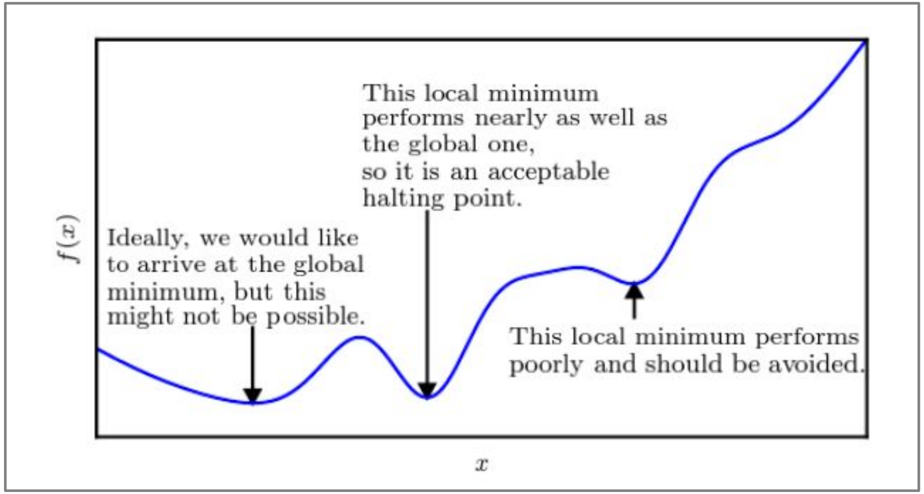
By Kyle Rust

What is an objective function?

- Most deep learning algorithms involve optimization of one kind or another
- We want to minimize a function *f(x)* by altering the input *x*
- This *f(x)* we want to minimize we call the **objective function**
- When we are focused on minimizing this *f(x)* we usually call it a loss function

How do we go about minimizing *f(x)*?

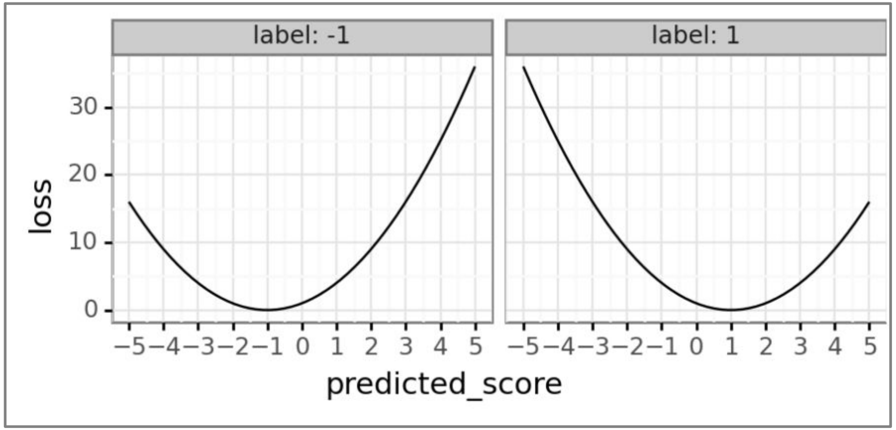
- Calculus! 💖
- We need to take the derivative of our loss function *f'(x)* which gives us the slope of *f(x)*
- In other words, how does *f(x)* change as we change *x*
- We use a technique called **gradient descent**, to move in the opposite direction of the gradient in order to move toward minimum



Goodfellow et al., 2016

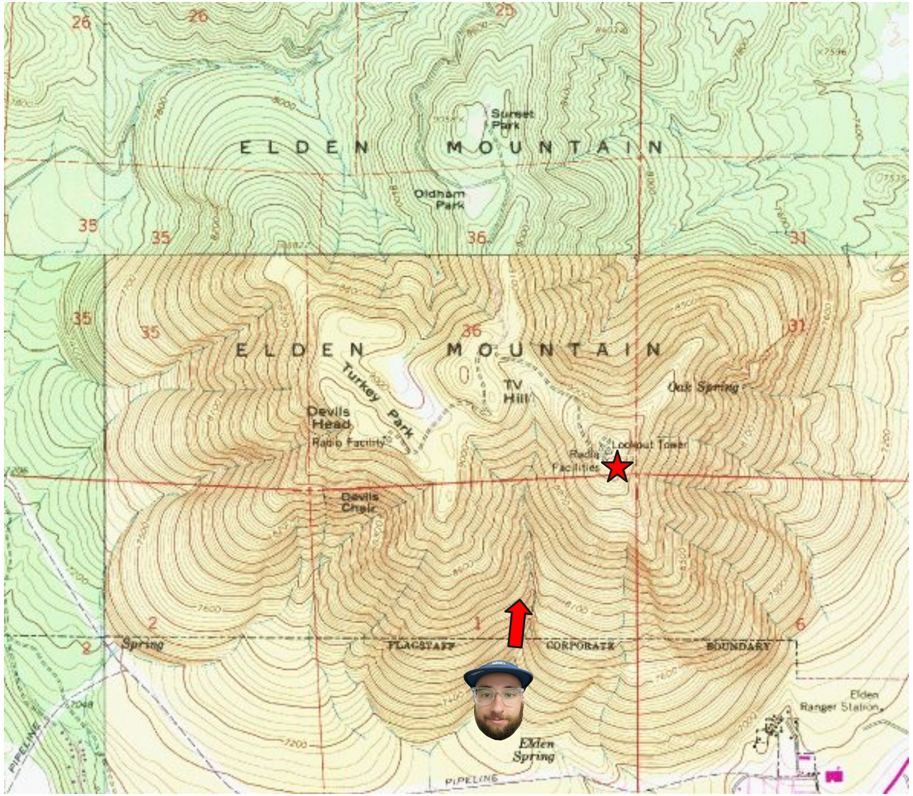
What is a convex surrogate?

- If a function is **non-convex**, that means it's wavy
    - It has some 'valleys', or **local minima**, that aren't as deep as the **global minimum**
- Thus, we optimize a **surrogate loss function** which acts as a proxy, but has advantages
- For example, the square loss provides a convex and differentiable function to minimize via gradient descent

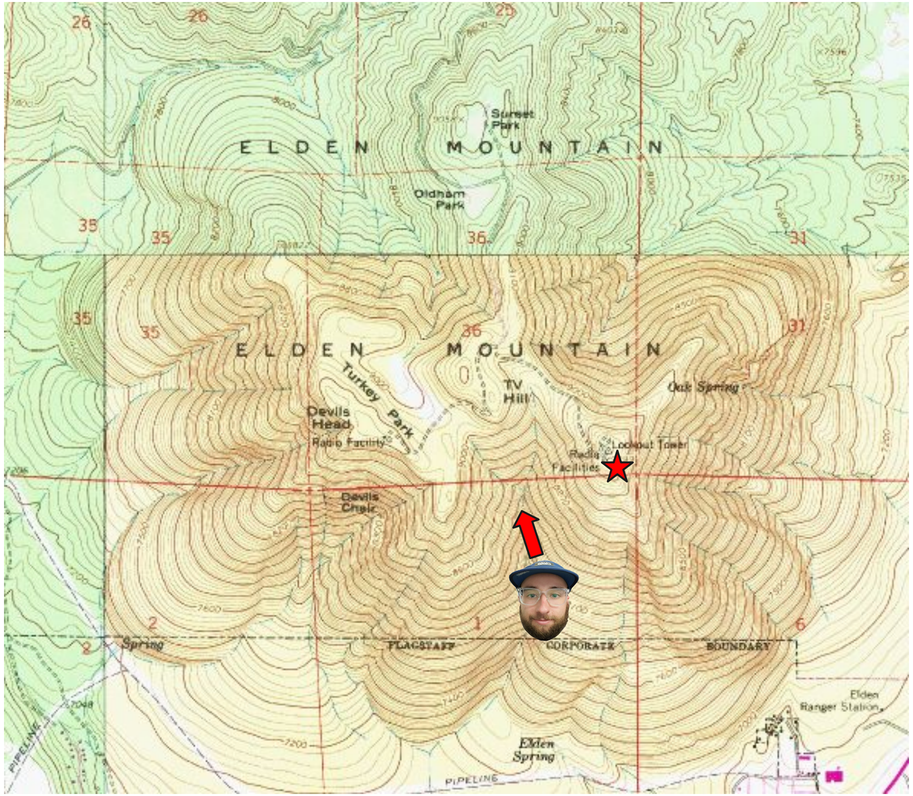


Hocking, 2022

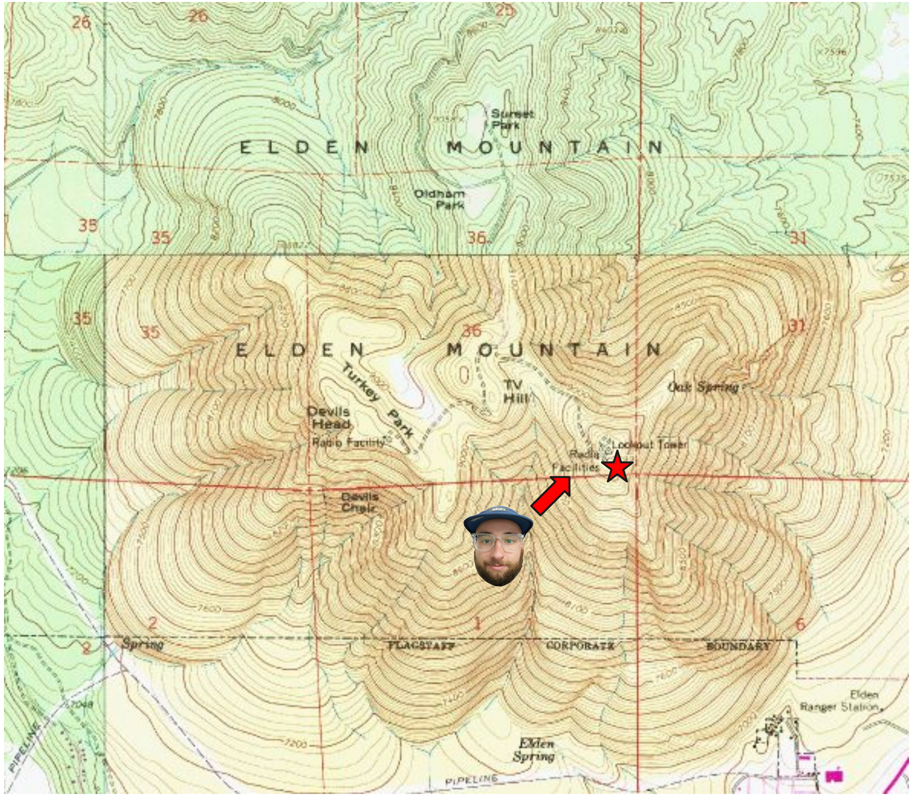
Gradient Example



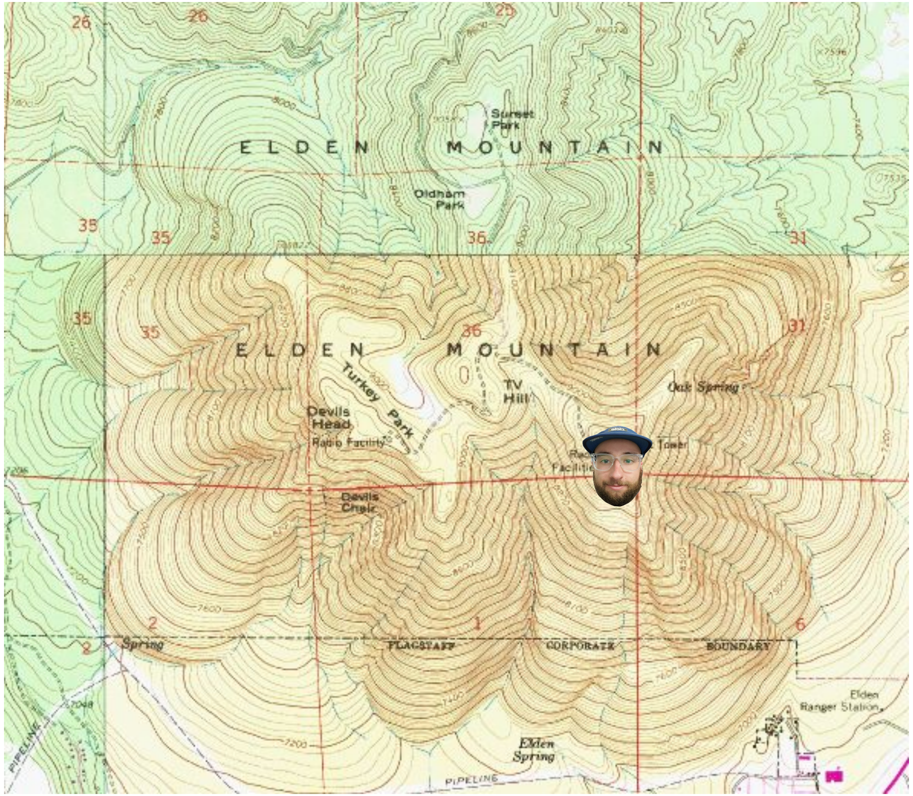
Gradient Example



Gradient Example

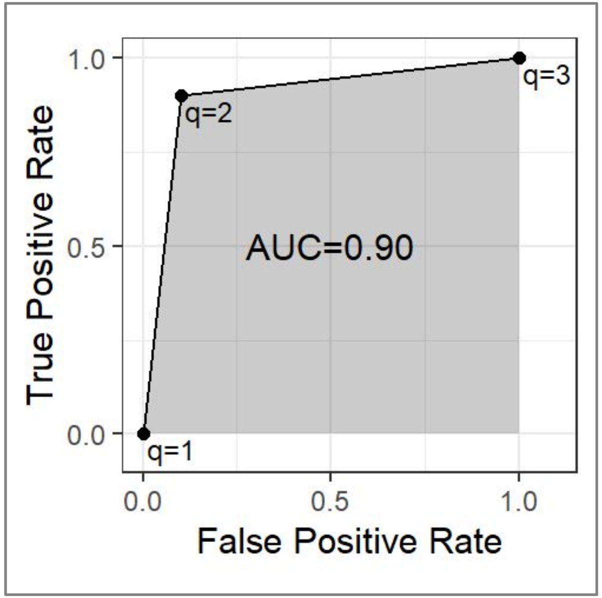


Gradient Example



Area Under the ROC curve

- As we've discussed at **Receiver Operating Characteristic** curve plots the FPR vs the TPR
- Donald Bamber was the first to prove that maximizing the ROC-AUC and minimizing the Mann-Whitney test statistic
- This opens up a research area for algorithms based on loss functions that are convex surrogates and sum over pairs of positive and negative labels



Hillman and Hocking, 2021

LIBAUC

- Instead of minimizing cross entropy loss, they instead chose to maximize this AUC value
- Benefits to this approach
  - AUC is a core machine learning metric so aiming to maximize it lead to model performance improvements
  - AUC is more adept at handling highly unbalanced data sets because it aims to rank the score of any positive data higher than any negative data
- This approach is more challenging however, as AUC is very sensitive to model changes
- Foremost challenge is finding a surrogate loss for the AUC score
- Naive method is pairwise surrogate loss which is **too slow!**

Proposed Factoring Trick

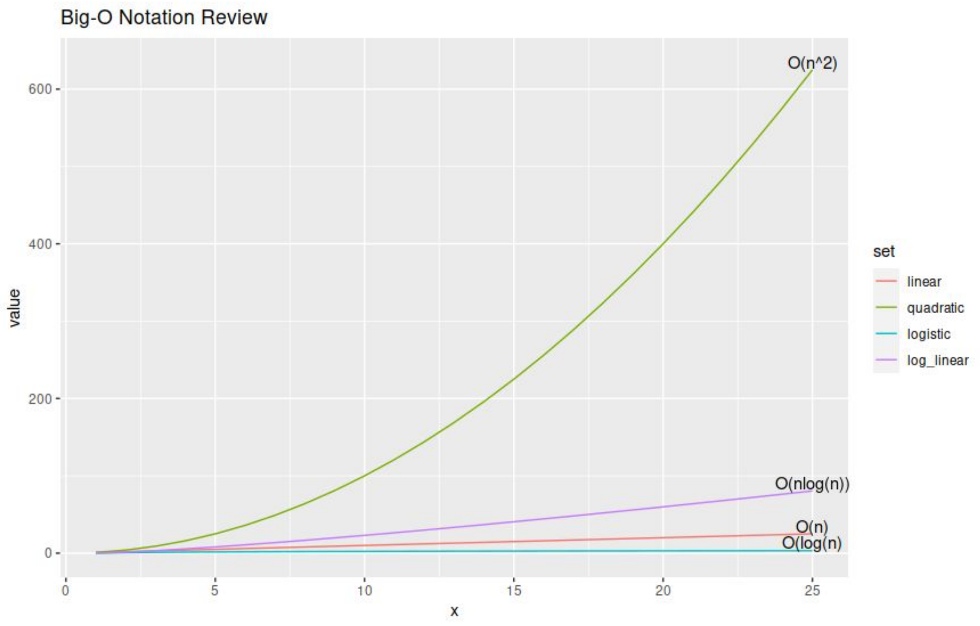
$m =$ *the margin parameter* $\quad$ *k subscript -> k ϵ I* $^-$

$l(z) = (m - z)^2$

$$
\begin{aligned}
\sum_{j \in \mathcal{I}^+} \ell(\hat{y}_j - \hat{y}_k) &= \sum_{j \in \mathcal{I}^+} (m - \hat{y}_j + \hat{y}_k)^2 \\
&= \sum_{j \in \mathcal{I}^+} (m - \hat{y}_j)^2 + 2(m - \hat{y}_j)\hat{y}_k + \hat{y}_k^2 \\
&= a^+ \hat{y}_k^2 + b^+ \hat{y}_k + c^+.
\end{aligned}
$$

Big-O Notation Review

- Big O notation is used to describe the asymptotic behavior of a function as the input goes to infinity
- Described in the “O” is the dominating term
- Rule of thumb: n describes the number of observations that need to be looped over



Naive Methods

```
Square Loss
for i in positive examples:
      for j in negative examples:
            Z = predictions[i] - predictions[k]
            Loss += (margin - Z)^2
 Return Loss


 O(n^2)
```

```
Squared Hinge Loss
for i in positive_labels:
       for j in negative_labels:
           Z = predictions[i] - predictions[j]
           loss_clipped = margin - Z
           if loss_clipped > 0:
               loss += loss_clipped^2
   return loss
O(n^2)
```

Functional Representations

Functional Square Loss

```
for i in positive_labels:
        Z = margin - predictions[i]
        A += 1
        B += -2*Z
        C += Z^2
for j in negative_labels:
        Loss += A*predictions[k] + B*predictions[k] + C
return Loss
```

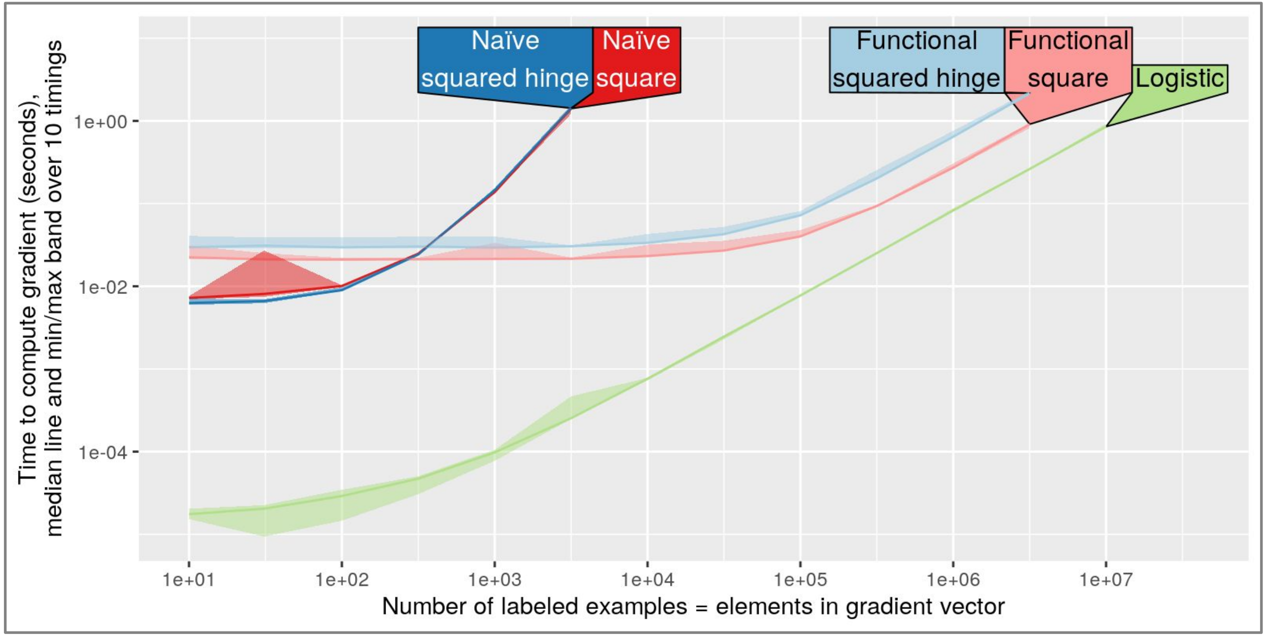
O(n)

Functional Squared Hinge Loss

```
Sorted_predictions = argsort(predictions)
for i in range(num_observations):
        Pred_value  = predictions[sorted_predictions[i]]
        If labels[sorted_predictions[i]] == 1:
                Z = margin - pred_value
                A += 1
                B += -2*Z
                C += Z^2
        Else
                Loss += A*pred_value^2 + B*pred_value + C
return Loss
```

O(nlog(n))

Timing Proof of Concept



Hocking, 2021

Linear Model Function

$$f(x) = \beta + w^T x$$

Pred Score Intercept Weights Inputs

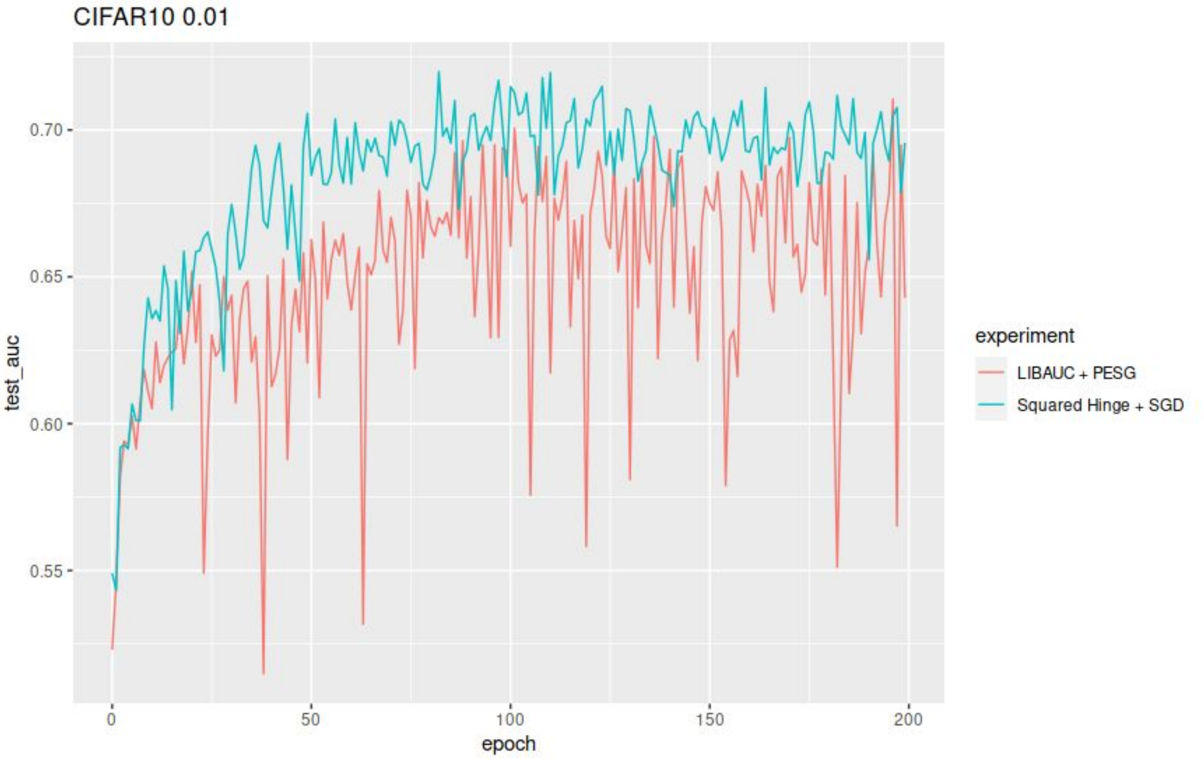
GOAL: Achieve overfitting on the test set in less epochs or achieve a higher AUC value in the same number of epochs

Overfitting: ‘U’ shape in test loss   Epoch: Number of times iterating over full dataset

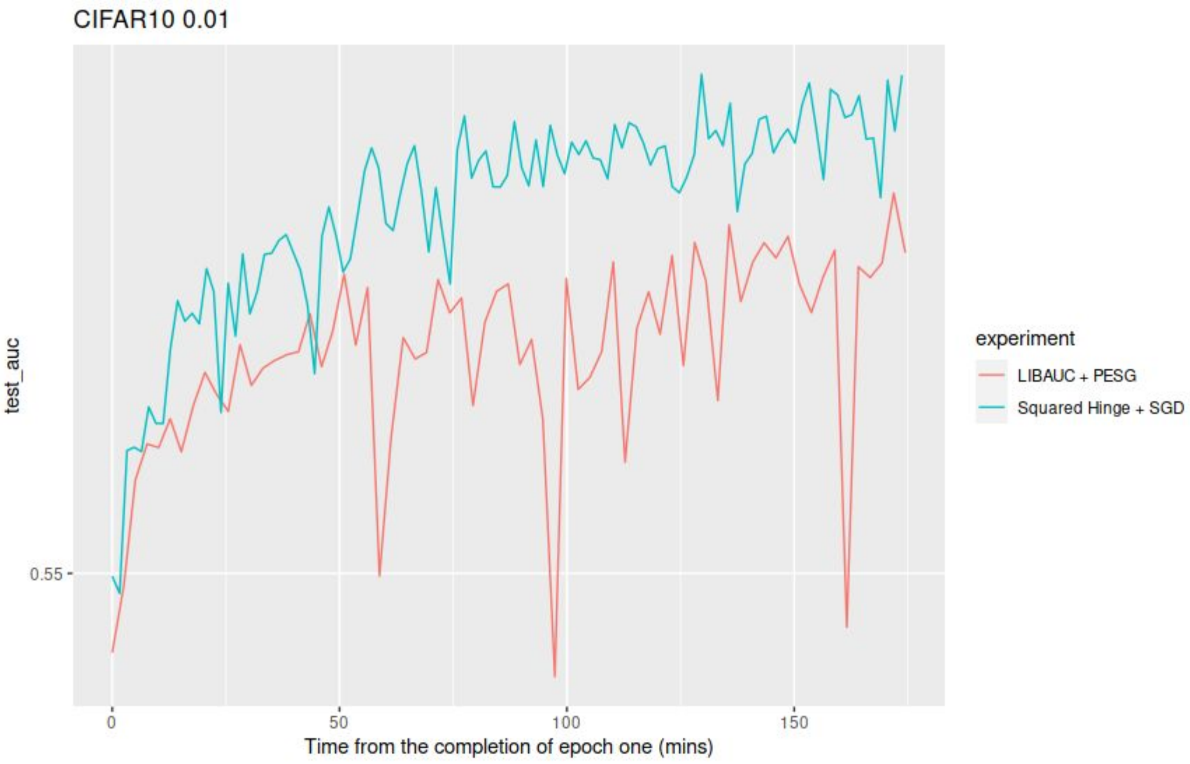
Overview of Training Process

1. Load standard machine learning data set CIFAR10, STL10, Cat & Dog etc.
2. Set parameters
    a. Step size, batch size, and imbalance ratio
3. Loop over the entire train set *z* times
    a. Compute a predicted labels vector
    b. Compute the loss between that vector and the true train labels
    c. Take a step in the opposite direction of the gradient (for minimization)
    d. Back propagate and update the weight values of your neural network
4. Loop over the entire test set *z* times
    a. Use weights at every epoch to compute the predicted class label on the test features
    b. Compute the loss between that predicted vector and the test labels

Test AUC Comparison



Test AUC Comparison



Future Work

- Continue to investigate benefit of our functional representation over LIBAUC
- Expand our investigation to other loss functions such as linear hinge
- Investigate empirical and theoretical properties of algorithms that could use our functional loss representation such as Stochastic Average Gradient[Roux et al., 2012]

References

Backcountry mapping evolved. (n.d.). https://caltopo.com/map.html

Bamber, D. (1975). The area above the ordinal dominance graph and the area below the receiver operating characteristic graph. Journal of mathematical psychology, 12 (4), 387–415.

Goodfellow, I., Bengio, Y. & Courville, A. (2016). Deep learning [http://www.deeplearningbook.org]. MIT Press.

Hillman, J. & Hocking, T. D. (2021). Optimizing roc curves with a sort-based surrogate loss function for binary classification and changepoint detection. https://doi.org/10.48550/ARXIV.2107.01285

Hocking, T. D. (2021). https://github.com/tdhock/min-aup-paper

Hocking, T. D. (2022). Linear models with early stopping regularization [https://github.com/tdhock/cs570-spring-2022/blob/master/slides/04-linear-models.pdf].

Roux, N. L., Schmidt, M. & Bach, F. (2012). A stochastic gradient method with an exponential convergence rate for finite training sets. https://doi.org/10.48550/ARXIV.1202.6258

Yuan, Z., Yan, Y., Sonka, M. & Yang, T. (2020). Robust Deep AUC Maximization: A New Surrogate Loss and Empirical Studies on Medical Image Classification [Preprint arXiv:2012.03173].