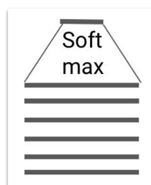
Robust Bi-Tempered Logistic Loss Based on Bregman Divergences

Ehsan Amid, Manfred Warmuth, Rohan Anil & Tomer Koren
Google Brain

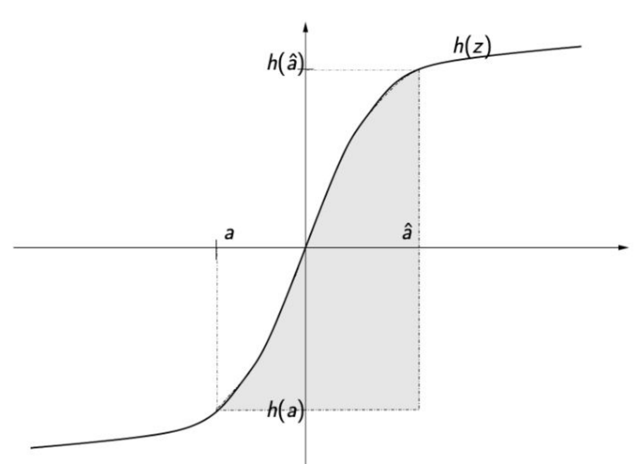
Logistic Loss as Matching Loss

- The most commonly used loss for classification in NN
  - Always convex in the activations/weights of the last layer
  - However, anyway non-convex in weights of lower layers
  - We replace last layer by non-convex loss that makes NN robust to outliers



Softmax $\hat{y}_i = \dfrac{\exp(\hat{a}_i)}{\sum_{j=1}^k \exp(\hat{a}_j)}$

Matching loss



Logistic loss = area under sigmoid

$$\Delta_{\log}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \sum_i y_i \log \frac{y_i}{\hat{y}_i}$$

In general [HKW95]

$$\Delta_h(\hat{\boldsymbol{a}}, \boldsymbol{a}) = \int_{\boldsymbol{a}}^{\hat{\boldsymbol{a}}} (h(\mathbf{z}) - h(\boldsymbol{a}))\, d\,\mathbf{z} = \Delta_{h^{-1}}(\underbrace{h(\boldsymbol{a})}_{\boldsymbol{y}}, \underbrace{h(\hat{\boldsymbol{a}})}_{\hat{\boldsymbol{y}}})$$

Simpliciy of the Matching Loss

- $$\Delta_h(\hat{\boldsymbol{a}}, \boldsymbol{a}) = \int_{\boldsymbol{a}}^{\hat{\boldsymbol{a}}} (h(\mathbf{z}) - h(\boldsymbol{a}))\, d\,\mathbf{z}$$
- **Convex for any increasing transfer function**

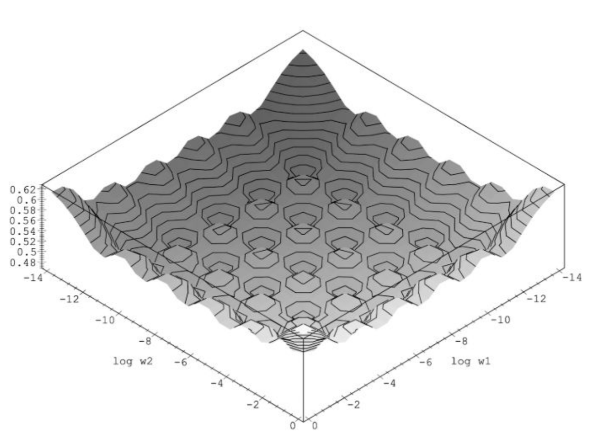
$$\frac{\partial}{\partial \hat{\boldsymbol{a}}} \Delta_h(\hat{\boldsymbol{a}}, \boldsymbol{a}) = h(\hat{\boldsymbol{a}}) - h(\boldsymbol{a}) = \underbrace{\hat{\boldsymbol{y}} - \boldsymbol{y}}_{\text{delta rule}}$$

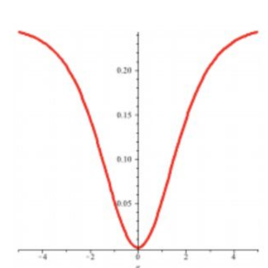
- Examples
  - h = id: $\Delta_{id}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = 1/2 \sum_i (y_i - \hat{y}_i)^2$
  - h = softmax: $\Delta_{\log}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \sum_i y_i \log \frac{y_i}{\hat{y}_i}$ (logistic loss)

Canonical Mismatch Case

Sigmoid with square loss

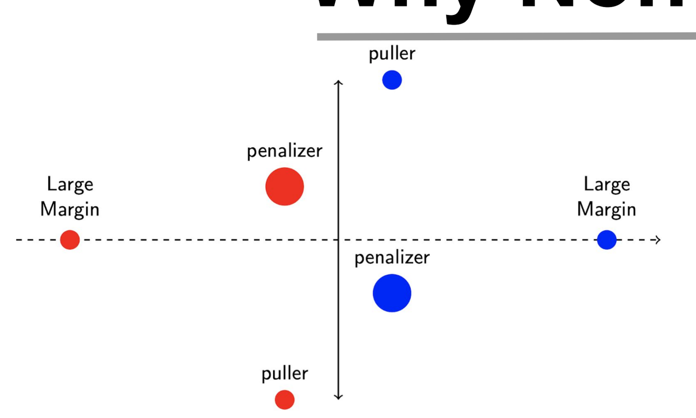
$$\frac{\partial}{\partial \hat{a}} (\sigma(\hat{a}) - y)^2 = (\sigma(\hat{a}) - y) \underbrace{\sigma'(\hat{a})}_{\sigma(\hat{a})(1-\sigma(\hat{a}))}$$

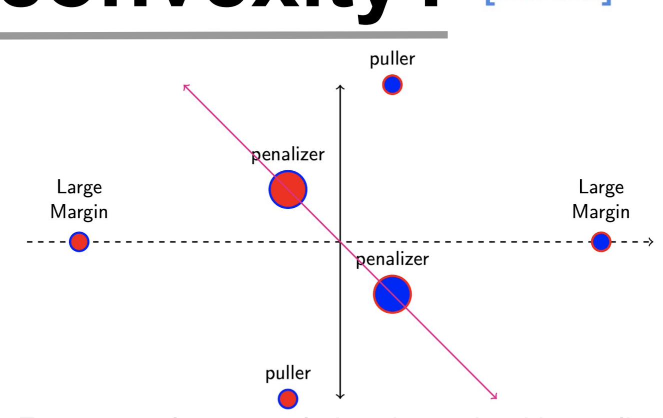




Can lead to exponent. many minimas [AHW95]

Why Non-convexity? [LS08]





For convex losses, solution determined by outliers

Two Drawback of Logistic Loss

1. Convex losses are not robust to noise
   - Convex losses increase unboundedly
   - A single bad example can dominate the cumulative loss
   - Extreme examples can cause large gradients
   - Non-convex (bending down) losses have been shown to perform significantly better
2. Softmax probabilities have exponentially decaying tail
   - The margin becomes small for mislabeled examples near the boundary
   - Heavy-tailed alternatives yield better margins and improved results [DV,10]

Logistic loss = rel. entr. + softmax

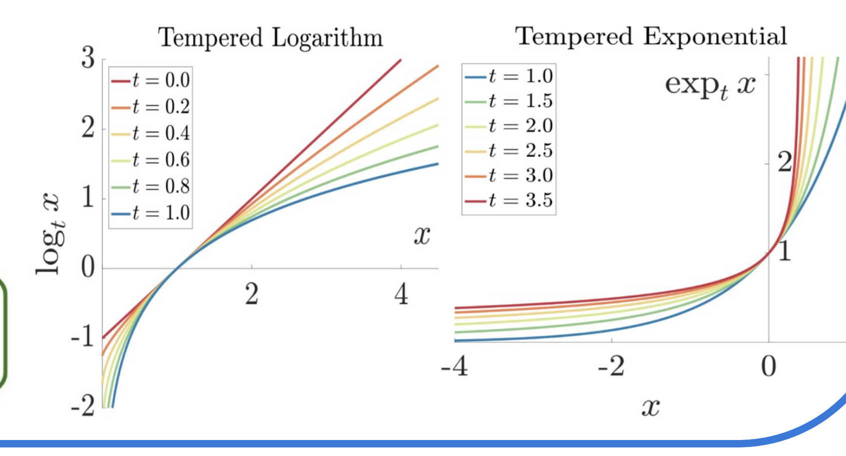
- Introduce temperatures into links, i.e. $\log_{t_1}$ and $\exp_{t_2}$
- When the 2 temperatures are **equal**, we again obtain a **convex** loss
- By increasing the temperature in the exponential, loss becomes non-convex
- Tuning the two temperatures will be crucial

$$\log_t(x) := \frac{1}{1-t}(x^{1-t} - 1)$$

Bounded by $-1/(1-t)$ at 0 for $0 \leqslant t < 1$

$$\exp_t(x) := [1 + (1-t)\,x]_+^{1/(1-t)}$$

Heavier tail for $x < 0$ for $t > 1$



1. Replacing the Relative Entropy Divergence

Tempered relative entropy divergence ($0 \leqslant t_1 < 1$):

$$\sum_{i=1}^k \left(y_i (\log_{t_1} y_i - \log_{t_1} \hat{y}_i) - \tfrac{1}{2-t_1}(y_i^{2-t_1} - \hat{y}_i^{2-t_1})\right) \overset{\text{if } y \text{ one-hot}}{=} -\log_{t_1} \hat{y}_c - \tfrac{1}{2-t_1}\left(1 - \sum_{i=1}^k \hat{y}_i^{2-t_1}\right)$$

bounded by $1/(1-t_1)$

2. Replacing the Softmax Probabilities

$\boldsymbol{z}$: vector of inputs to softmax layer
$\boldsymbol{w}_i$: trainable weight vector for class i
$\boldsymbol{y}$: target vector

Softmax:

$$\hat{y}_i = \frac{\exp(\hat{a}_i)}{\sum_{j=1}^k \exp(\hat{a}_j)} = \exp\left(\hat{a}_i - \log \sum_{j=1}^k \exp(\hat{a}_j)\right), \text{ for linear activation } \hat{a}_i = \boldsymbol{w}_i \cdot \boldsymbol{z} \text{ for class } i$$

Tempered Softmax ($t_2 > 1$):

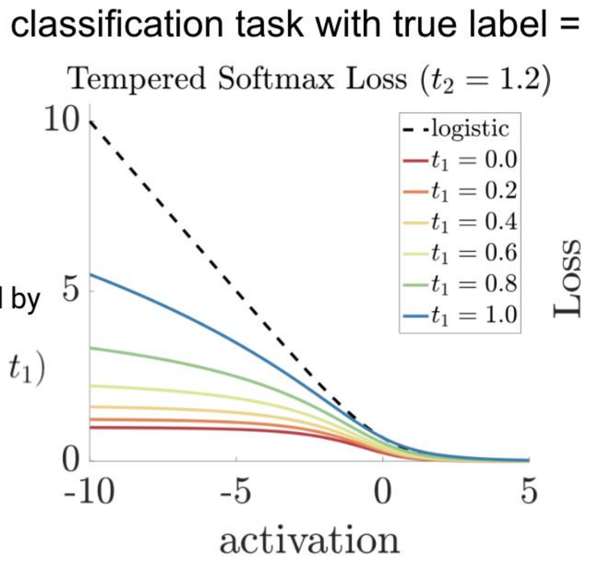
$$\hat{y}_i = \exp_{t_2}\left(\hat{a}_i - \lambda_{t_2}(\hat{\boldsymbol{a}})\right), \quad \text{where } \lambda_{t_2}(\hat{\boldsymbol{a}}) \in \mathbb{R} \text{ is s.t. } \sum_{j=1}^k \exp_{t_2}\left(\hat{a}_j - \lambda_{t_2}(\hat{\boldsymbol{a}})\right) = 1$$

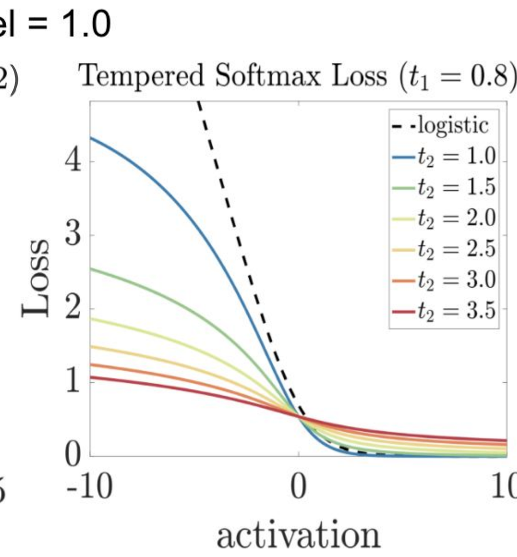
Tail-heavy for t2 > 1!

Examples of the Bi-Tempered Logistic Loss (y = 1.0)

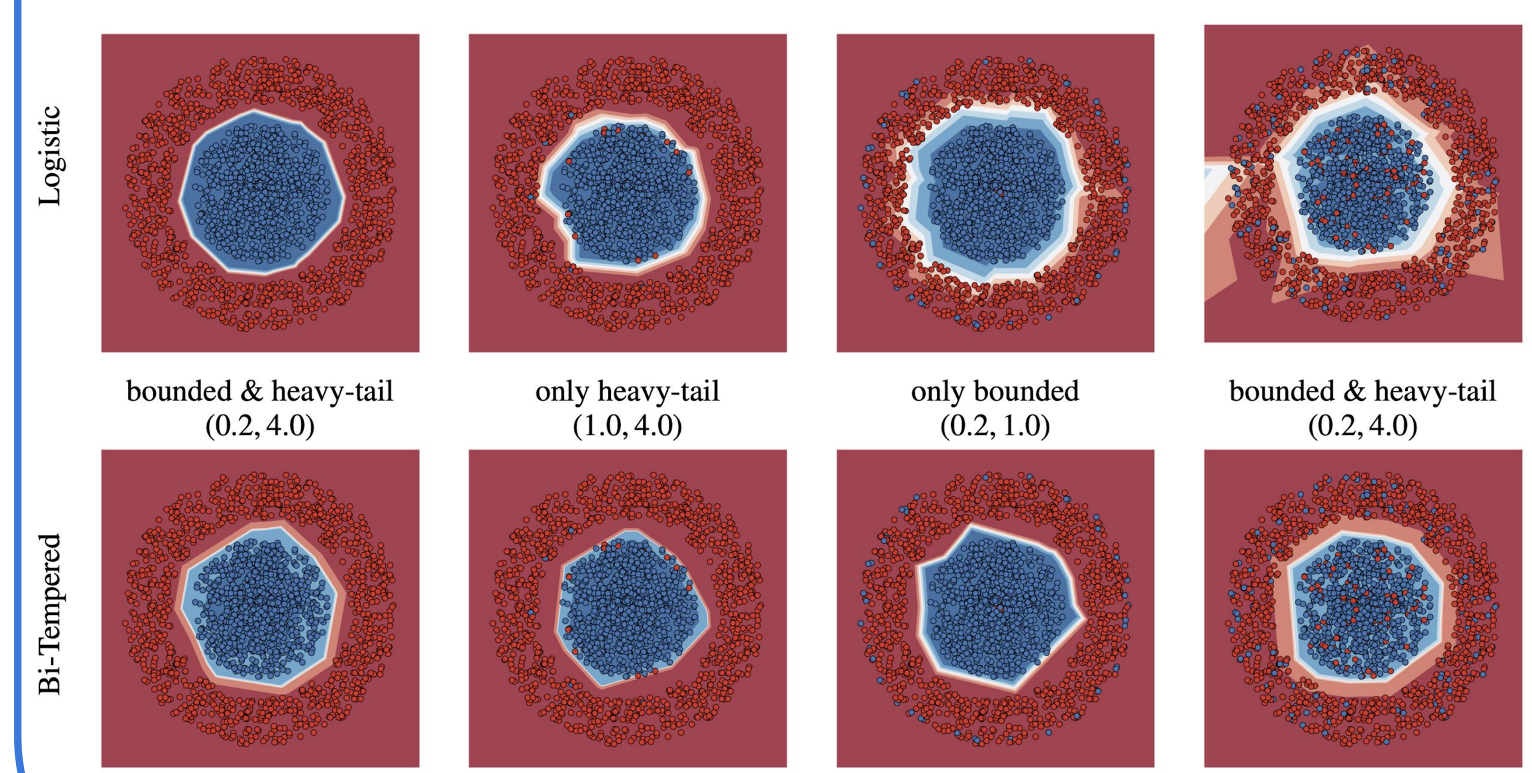
A binary classification task with true label = 1.0

Bounded by $1/(1-t_1)$





A Toy Example



Two-layer feed-forward network (10 and 5 neurons) with ReLU activations

Experiments

Synthetic label Noise:

Dataset	Loss	Label Noise Level 0.0	0.1	0.2	0.3	0.4	0.5
MNIST	Logistic	**99.40**	98.96	98.70	98.50	97.64	96.13
	Bi-Tempered (0.5, 4.0)	99.24	**99.13**	**99.02**	**98.62**	**98.56**	**97.69**
CIFAR-100	Logistic	74.03	69.94	66.39	63.00	53.17	52.96
	Bi-Tempered (0.8, 1.2)	**75.30**	**73.30**	**70.69**	**67.45**	**62.55**	**57.80**

Table 1: Top-1 accuracy on a clean test set for MNIST and CIFAR-100 datasets where a fraction of the training labels are corrupted.

Large-scale Experiments on ImageNet2012:

Model	Logistic	Bi-tempered (0.9,1.05)
Resnet18	71.333 ± 0.069	**71.618** ± 0.163
Resnet50	76.332 ± 0.105	**76.748** ± 0.164

Table 2: Top-1 accuracy on ImageNet-2012 with Resnet-18 and 50 architectures.

Code and References

Open source TF implementation available at **Google research Github**:

https://github.com/google-research/google-research/tree/master/bitempered_loss

Replace one line:

~~Softmax_cross_entropy_with_logits(activations,labels)~~

```
bi_tempered_logistic_loss(activation,labels,t1,t2)
```

[HKW95] David P. Helmbold, Jyrki Kivinen and Manfred K. Warmuth. Worst-case loss bounds for single neurons. NIPS '95, pp. 309–315.

[KW01] Jyrki Kivinen and Manfred K. Warmuth. Relative loss bounds for multidimensional regression problems. Journal of Machine Learning, Vol. 45(3), pp. 301-329.

[LS08] Philip M Long and Rocco A Servedio. Random classification noise defeats all convex potential boosters. ICML'08. pp. 608–615.