

# Towards a Deep Adversarial Classifier that Exactly Optimizes the 0-1 Loss

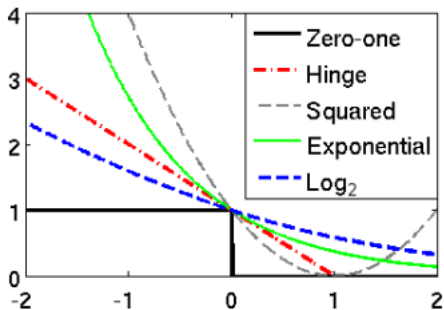
CS 270 Final Project

Nikhil Mishra    Mostafa Rohaninejad

May 1, 2017

# Motivation

- ▶  $K$ -way classification problem
- ▶ Performance criterion we care about is accuracy (0-1 loss).
- ▶ Gradient of 0-1 loss is not conducive to optimization
- ▶ So we end up optimizing a smooth, convex, upper bound:
  - ▶ L2, cross-entropy, hinge losses ...



- ▶ More generally: incur cost  $C(i, j) \geq 0$  for predicting class  $i$  when true class is  $j$ .

# Our Project

- ▶ We will summarize a recently-proposed technique, based on duality in zero-sum games, that allows exact optimization of the 0-1 loss (and cost-sensitive variants)
  - ▶ Introduced by: Asif, Kaiser, et al. "Adversarial Cost-Sensitive Classification." UAI. 2015.
  - ▶ More related papers from Brian Ziebart's group at UIC
- ▶ Then we offer a novel extension that allows their idea to scale to larger feature spaces (e.g. images).

# Preliminaries

- ▶ Cost-matrix  $C$ :
  - ▶ Where  $C_{i,i} = 0, C_{i,j \neq i} > 0$
  - ▶ 0-1 loss corresponds to  $C_{i,j \neq i} = 1, \forall i, j$
- ▶ Input points drawn from some data distribution  $P(X)$
- ▶ Given some  $x \sim P(X)$ , consider the true distribution over classes  $P(Y|X = x)$ :
  - ▶ Abbreviate  $P(Y|X = x)$  as  $y$ .
  - ▶ If we predict distribution  $\hat{y}$ , then the expected loss is  $\hat{y}^T Cy$ .
- ▶ Want to minimize:  $\mathbb{E}_{P(X)P(Y|X)}[\hat{y}^T Cy]$ 
  - ▶ Abbreviate to:  $\mathbb{E}[\hat{y}^T Cy]$

# A minimax game

- ▶ Key idea: instead of optimizing an upper bound on the loss with respect to the training data, let's optimize the true loss on a pessimistic perturbation of the training data.
- ▶ Classifier predicts the distribution  $\hat{y}$ , then adversary chooses the evaluation distribution  $\check{y}$  to be used in place  $y$ .
- ▶ Yielding a two-player zero-sum game:

$$\min_{\hat{y} \in \Delta} \max_{\check{y} \in \Delta} \hat{y}^T C \check{y}$$

where  $\Delta$  is the probability simplex.

- ▶ Too easy for the adversary! (set all the labels to be random)

# A constrained minimax game

- ▶ Suppose we have a feature function  $\phi(x, y_i) \rightarrow \mathbb{R}^D$ .
- ▶ Add constraint that  $\mathbb{E}_{y_i \sim \check{y}}[\phi(x, y_i)] = \mathbb{E}_{y_i \sim y}[\phi(x, y_i)]$ 
  - ▶ Moment of features should match between evaluation distribution  $\check{y}$  and true distribution  $y$ .
  - ▶ Constrains the adversary to be close to the true distribution.
  - ▶ Let  $\Xi$  be feasible set for  $\check{y}$  given this new constraint.
- ▶ New constrained game:

$$\min_{\hat{y} \in \Delta} \max_{\check{y} \in \Delta \cap \Xi} \hat{y}^T C \check{y}$$

# Duality Tricks

Let  $\Phi_x$  be a  $D \times K$  matrix, where the  $i$ -th column is  $\phi(x, y_i)$ .  
Then we can rewrite the constrained minimax problem as:

$$\min_{\hat{y} \in \Delta} \max_{\check{y} \in \Delta \cap \Xi} \hat{y}^T C \check{y} \quad (1)$$

$$= \max_{\check{y} \in \Delta \cap \Xi} \min_{\hat{y} \in \Delta} \hat{y}^T C \check{y} \quad (2)$$

$$= \max_{\check{y} \in \Delta} \min_{\nu} \left[ \min_{\hat{y} \in \Delta} \hat{y}^T C \check{y} + \nu^T (\Phi_x \check{y} - \Phi_x y) \right] \quad (3)$$

$$= \min_{\nu} \left[ \max_{\check{y} \in \Delta} \min_{\hat{y} \in \Delta} \left( \hat{y}^T C \check{y} + \nu^T (\Phi_x \check{y} - \Phi_x y) \right) \right] \quad (4)$$

(2) strong duality in two-player zero-sum games

(3) formed the Lagrangian for adversary (primal variable  $\check{y}$ )

(4) strong duality holds for Lagrangian (true distribution  $y$  is strictly feasible)

# Algebra Tricks

$$\begin{aligned} & \min_{\nu} \left[ \max_{\check{y} \in \Delta} \min_{\hat{y} \in \Delta} \left( \hat{y}^T C \check{y} + \nu^T (\Phi_x \check{y} - \Phi_x y) \right) \right] \\ &= \min_{\nu} \left[ \max_{\check{y} \in \Delta} \min_{\hat{y} \in \Delta} \hat{y}^T \left( C + 1 \cdot \nu^T (\Phi_x - \Phi_x y \cdot 1) \right) \check{y} \right] \\ &= \min_{\nu} \left[ \max_{\check{y} \in \Delta} \min_{\hat{y} \in \Delta} \hat{y}^T (\tilde{C}_{x,\nu}) \check{y} \right] \end{aligned}$$

- ▶ Augmented cost-matrix  $\tilde{C}_{x,\nu} = C + 1 \cdot \nu^T (\Phi_x - \Phi_x y \cdot 1)$
- ▶ Loss function:  $J(\nu) = \mathbb{E}_{x \sim P(X)} \left[ \max_{\check{y} \in \Delta} \min_{\hat{y} \in \Delta} \hat{y}^T (\tilde{C}_{x,\nu}) \check{y} \right]$
- ▶ Optimize the expected value of a minimax game wrt Lagrange multiplier  $\nu$ .



# Deriving a Subgradient

- ▶ Loss function:  $J(\nu) = \mathbb{E}_{P(X)} \left[ \min_{\hat{y} \in \Delta} \max_{\check{y} \in \Delta} \hat{y}^T (\tilde{C}_{x,\nu}) \check{y} \right]$
- ▶ Convex in  $\nu$  (max wrt  $\check{y}$  of affine functions of  $\nu$ ).
- ▶ Not differentiable, but we can find a subgradient:

$$\begin{aligned} & \partial_\nu \mathbb{E}_{P(X)} \left[ \min_{\hat{y} \in \Delta} \max_{\check{y} \in \Delta} \hat{y}^T (\tilde{C}_\nu) \check{y} \right] \\ &= \mathbb{E}_{P(X)} \left[ \partial_\nu \min_{\hat{y} \in \Delta} \max_{\check{y} \in \Delta} \hat{y}^T (\tilde{C}_\nu) \check{y} \right] \\ &\ni \mathbb{E}_{P(X)} \left[ \frac{\partial}{\partial \nu} \left( (\hat{y}^*)^T (C_\nu) \check{y}^* \right) \right] \\ &= \Phi_x(\check{y}^* - y) \\ &= \mathbb{E}_{y_i \sim \check{y}^*} [\phi(x, y_i)] - \mathbb{E}_{y_i \sim y} [\phi(x, y_i)] \end{aligned}$$

# Training Algorithm

---

**Algorithm 1** Training the Classifier

---

- 1: **Input:** Cost matrix  $C$ , dataset  $\mathcal{D} = \{x, y\}$ , feature function  $\phi$
  - 2: **Output:** optimal Lagrange multiplier  $\nu^*$
  - 3: Initialize  $\nu \rightarrow 0$
  - 4: **while**  $\nu$  not converged **do**
  - 5:   Sample  $(x, y) \sim \mathcal{D}$
  - 6:   Compute  $\tilde{C}_{x,\nu}$  and solve minimax game for  $\tilde{y}^*$
  - 7:   Compute subgradient  $\nabla_\nu = \Phi_x(\tilde{y}^* - y)$
  - 8:   Update  $\nu$  using any gradient-descent-like algorithm
  - 9: **end while**
  - 10: **return**  $\nu$
-

# Inference Algorithm

---

## Algorithm 2 Making Predictions

---

- 1: **Input:** Cost matrix  $C$ , optimal multiplier  $\nu^*$ , feature function  $\phi$ , query point  $x$
- 2: **Output:** Predicted label distribution  $\hat{y}^*$
- 3: Take  $y = 0$  and construct  $\tilde{C}_\nu$  using  $\nu^*$
- 4: Solve the following LP:

$$\begin{aligned} \min_{v \in \mathbb{R}, \hat{y} \in \mathbb{R}^K} \quad & v \\ \text{s.t.} \quad & \hat{y}^T \tilde{C}_{x, \nu} \leq v, \hat{y} \in \Delta \end{aligned}$$

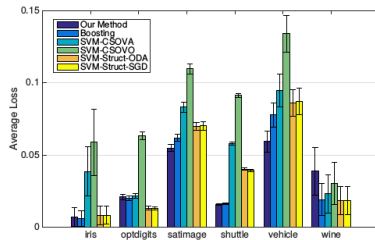
- 5: **return** LP solution  $\hat{y}^*$
-

# Results

## Datasets

Name	Classes	Attributes	Training	Testing
Iris	3	4	120	30
Optical Digits	10	64	3823	1797
Satellite Image	6	36	4435	2000
Shuttle	7	9	43500	14500
Vehicle	4	18	658	188
Wine	3	4	142	36
Breast Tissue	6	9	85	21
Ecoli	8	7	269	67
Glass	6	9	171	43
Image Segment	7	19	210	2100
Libras	15	90	288	72
Pen Digits	10	16	7494	3498
Vertebral	3	6	248	62

## Results



# Our Extension

- ▶ Where does  $\phi$  come from?
  - ▶ Original paper considers low-dimensional  $x$  and  $\phi$  is hand-specified
  - ▶ They just take a quadratic transformation:

$$\phi(x, y_i) = [\text{flatten}(xx^T) \cdot \mathbf{1}_{\{y_i = 1\}}, \dots, \text{flatten}(xx^T) \cdot \mathbf{1}_{\{y_i = K\}}]$$

- ▶ Not scalable: if  $x \in \mathbb{R}^d$ , then  $\Phi_x$  has dimensions  $d^2 K \times K$ .
- ▶ May not be using the best  $\phi$  for a particular problem

# Learned Features

- ▶ Suppose  $\phi$  is a differentiable function parametrized by  $\theta$ .
- ▶ Augmented cost-matrix:  $\tilde{C}_{x,\nu} \rightarrow C_{x,\nu,\theta}$
- ▶ Want to choose  $\phi$  that yields best  $\hat{y}$ :
  - ▶  $\nu$  forces adversary to satisfy constraint on expectation of  $\phi$
  - ▶  $\theta$  controls the expressiveness and usefulness of  $\phi$ .
  - ▶  $\nu$  and  $\theta$  are on the same team.

$$\min_{\nu} \left[ \max_{\check{y} \in \Delta} \min_{\hat{y} \in \Delta} \hat{y}^T (\tilde{C}_{\nu}) \check{y} \right] \rightarrow \min_{\nu, \theta} \left[ \max_{\check{y} \in \Delta} \min_{\hat{y} \in \Delta} \hat{y}^T (\tilde{C}_{\nu, \theta}) \check{y} \right]$$

- ▶ Loss function is now  $J(\nu, \theta)$ , jointly minimize over  $\nu, \theta$
- ▶ Can compute a subgradient with respect to  $\Phi_x$ :
$$\partial_{\Phi} J = \nu(\check{y} - y)^T$$
- ▶ Can compute  $\frac{\partial \Phi}{\partial \theta}$  since  $f$  is differentiable.
- ▶ Using the chain rule:  $\partial_{\theta} J = \partial_{\Phi} J \cdot \frac{\partial \Phi}{\partial \theta}$
- ▶ If  $f$  is a neural network, then start from  $\partial_{\Phi} J = \nu(\check{y} - y)^T$  and just do backpropagation!

# Our Training Algorithm

---

## Algorithm 3 Training the Deep Classifier

---

- 1: **Input:** Cost matrix  $C$ , dataset  $\mathcal{D} = \{x, y\}$ , feature function  $\phi$  parametrized by  $\theta$
  - 2: **Output:** optimal Lagrange multiplier  $\nu^*$
  - 3: Initialize  $\nu \rightarrow 0$ ,  $\theta \rightarrow$  randomly
  - 4: **while**  $\nu, \theta$  not converged **do**
  - 5:   Sample  $(x, y) \sim \mathcal{D}$
  - 6:   Compute  $\tilde{C}_\nu$  and solve minimax game for  $\check{y}^*$
  - 7:   Compute subgradient  $\nabla_\nu = \Phi_x(\check{y}^* - y)$
  - 8:   Compute  $\nabla_\theta$  using backprop starting from  $\nu(\check{y}^* - y)^T$
  - 9:   Update  $\nu, \theta$  using  $\nabla_\nu, \nabla_\theta$  with Adam algorithm
  - 10: **end while**
  - 11: **return**  $\nu, \theta$
-

# Results

- ▶ Tried our method on CIFAR-10 dataset: 60k  $32 \times 32$  color images from 10 classes.
- ▶ Used a 6-layer convolutional neural network for  $\phi$ .

Method	Training Accuracy	Test Accuracy
Cross-Entropy Loss	<b>0.9932</b>	0.8061
Our Method	0.9745	<b>0.8386</b>

Table: CIFAR-10 Results

- ▶ Our method yields sensible results and overfits less.
- ▶ We actually optimized the true evaluation criterion!
- ▶ Note: state-of-the-art for this dataset gets around 96% test accuracy, but the networks are  $> 20$  layers and use many fancy tricks. Would have been nice to try one of those networks, but they require significant compute to train.



# Conclusion

- ▶ Contributions of the original paper:
  - ▶ Instead of optimizing a convex upper-bound on the loss wrt training data, optimize the true evaluation criterion subject to an adversarial perturbation of the training data.
  - ▶ Constrain adversary so that feature expectations are the same.
  - ▶ Training by (sub)gradient descent on Lagrange multiplier.
  - ▶ Inference by solving an LP.
- ▶ Our contributions:
  - ▶ Extend their method to simultaneously learn feature function and optimal Lagrange multiplier.
  - ▶ Our method allows learning optimal features and scales to high-dimensional input spaces.
  - ▶ Showed signs of life on an image classification task that would have been intractable for the original method.
- ▶ GitHub repo with implementation of both methods:  
<https://github.com/nikhilmishra000/adversarial/>