GenLabel : Mixup Relabeling Using Generative Models

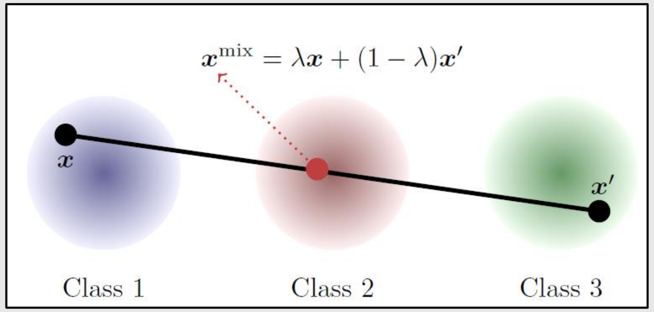
-Summary-

Introduction

Questions

- Mix-up sometimes degrades performance for some failure scenarios

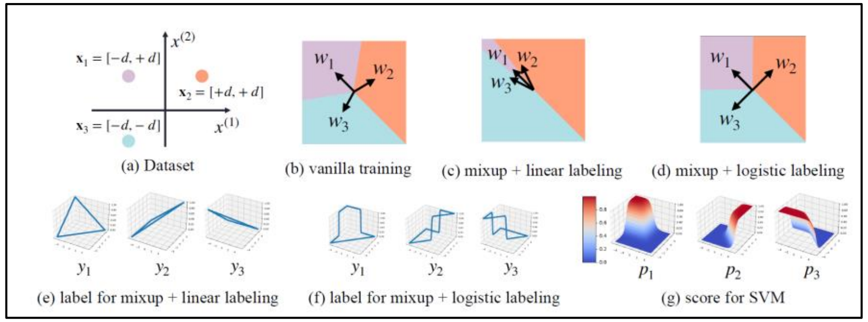
1. ***Manifold intrusion*** : mix-up sample from two classes intrudes manifold of a third class



Manifold intrusion

2. ***Naïve linear combination of two labels (linear labeling):***

   In some cases, linear combination $(\lambda y_1 + (1-\lambda) y_2)$ does not achieve largest (classification) margin, while logistic labeling $(\rho y_1 + (1-\rho) y_2$, where $\rho = \frac{1}{1+\exp(-\frac{2(\lambda - \frac{1}{2})}{\sigma^2})}$, $\sigma > 0$ does.



Sub-optimality of linear labeling
Recall : SVM achieves maximum margin by its algorithm

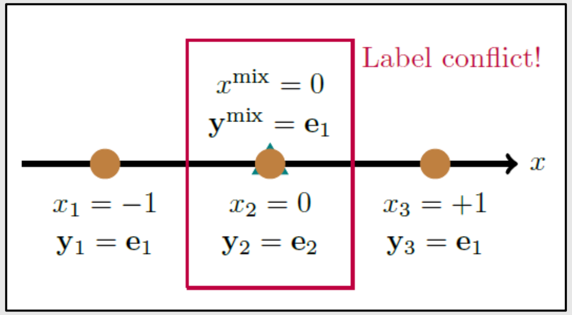
Introduction

Notation & Preliminaries

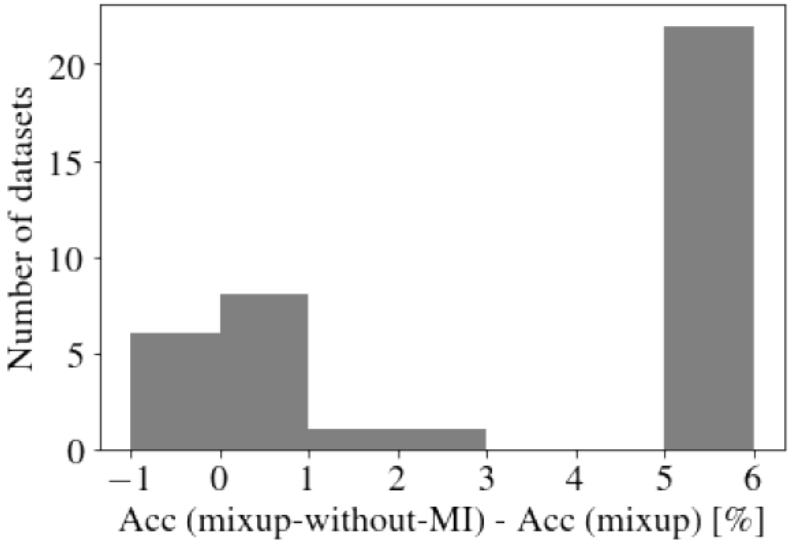
- Example : $z_i = (x_i, y_i)$, where $x_i \in \mathbb{R}^d$, $y_i \in [0,1]^k$ (one-hot encoding vector)
- Set of training data : $S = \{(x_i, y_i)\}_{i=1}^n$ (Define $X = \{x_i\}_{i=1}^n$)
- **Gaussian mixture model (GM) :**
  1. Assume sample x (in class c) $\sim N(\mu_c, \Sigma_c)$ (i.e : $p_c(x) = p(x|y = c) = N(\mu_c, \Sigma_c)$)
  2. Use estimator $\widehat{\mu_c}$ = sample mean in class c, $\widehat{\Sigma_c}$ = sample covariance in class c
  3. GM model for k classes : $p(x) = \sum_{i=1}^k \mathbb{P}(y = e_i)\, N(\mu_i, \Sigma_i)$
- **Kernel density estimator (KDE) with Gaussian kernel:**
  - Gaussian KDE of class c distribution : $p_c(x) = \frac{1}{n_c} \sum_{i=1}^{n_c} N(x_i, h^2\, \widehat{\Sigma_c})$

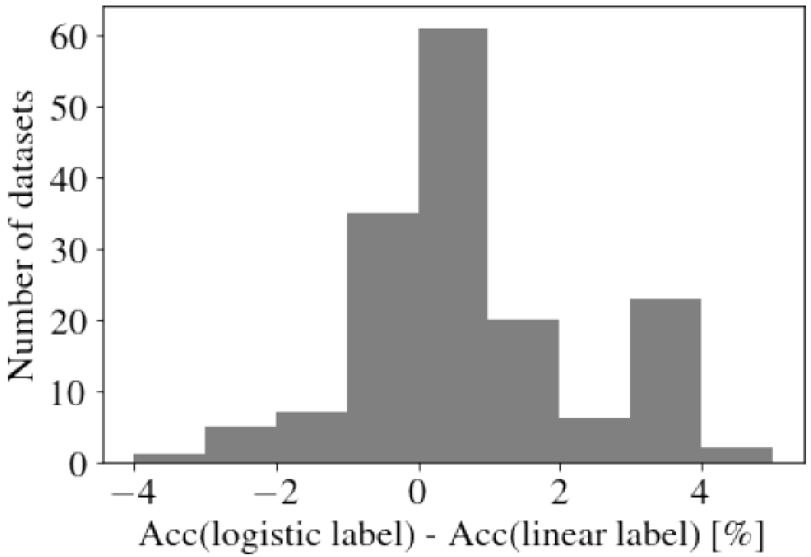
    where h = bandwidth, $\{x_i\}_{i=1}^{n_c}$ = set of samples in class c
- **Manifold intrusion removal (Guo et al., 2019):**
  - Not use $x^{mix} = \lambda x_1 + (1 - \lambda) x_2$ if the label of $x^{nn} := argmin_{x \in X}\, d(x, x^{mix})$ is different from y_1 and y_2

Manifold intrusion



Effect of Manifold intrusion removal / logistic labeling





Note

- Mix-up / Mix-up without-MI(Manifold intrusion removal) : 38 datasets in OpenML

  => 24 of 38 : positive accuracy difference

- Linear label / logistic label : 160 datasets in OpenML (after manifold intrusion removal)

  => 87 of 160 : positive accuracy difference

➢ Manifold intrusion is causing accuracy drop in various real datasets

➢ Linear labeling is sub-optimal for a large number of low-dimensional(<20) real datasets

GenLabel Algorithm

Idea for GenLabel

- Linear labeling y^{mix} does not offer good label for x^{mix} => requires some appropriate label for x^{mix} considering the given data.
- **Algorithm Step :**
  1. Estimate class-conditional data distribution $p_c(x)$ for each class c using GM or KDE
  2. Apply conventional mix-up data augmentation generating (x^{mix}, y^{mix})
  3. Relabel y^{mix} to $y^{gen} \coloneqq softmax(\log p_1(x^{mix}), \log p_2(x^{mix}), \dots, \log p_k(x^{mix}))$

     Note : $y^{gen} = \sum_{c=1}^{k} \frac{p_c(x^{mix})}{\sum_{c'=1}^{k} p_{c'}(x^{mix})} e_c$ (rephrase)

GenLabel Algorithm – Generative model learns in input feature

Algorithm 1 GenLabel

Input Dataset $S = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^n$, learning rate η, loss ratio γ
Output Trained discriminative model $f_\theta(\cdot)$

1: $\theta \leftarrow$ Random initial model parameter
2: $p_c(\boldsymbol{x}) \leftarrow$ Density estimated by generative model for input feature $\boldsymbol{x} \in X$, conditioned on class $c \in [k]$
3: **for** $(\boldsymbol{x}_i, \boldsymbol{y}_i), (\boldsymbol{x}_j, \boldsymbol{y}_j) \in S$ **do**
4: $\quad (\boldsymbol{x}^{\text{mix}}, \boldsymbol{y}^{\text{mix}}) = (\lambda \boldsymbol{x}_i + (1-\lambda)\boldsymbol{x}_j, \lambda \boldsymbol{y}_i + (1-\lambda)\boldsymbol{y}_j)$
5: $\quad \boldsymbol{y}^{\text{gen}} \leftarrow \sum_{c=1}^k \frac{p_c(\boldsymbol{x}^{\text{mix}})}{\sum_{c'=1}^k p_{c'}(\boldsymbol{x}^{\text{mix}})} \boldsymbol{e}_c$
6: $\quad \theta \leftarrow \theta - \eta \nabla_\theta \{\gamma \cdot \ell_{\text{CE}}(\boldsymbol{y}^{\text{gen}}, f_\theta(\boldsymbol{x}^{\text{mix}})) + (1-\gamma) \cdot \ell_{\text{CE}}(\boldsymbol{y}^{\text{mix}}, f_\theta(\boldsymbol{x}^{\text{mix}}))\}$
7: **end for**

GenLabel algorithm when Generative model learns the density ($p_C(x)$) in the input feature

Note :
Generative model can be imperfect estimate on the data distribution. So, y^{gen} may be incorrect for some samples.
↓
Solution : Use linear combination of y^{mix}, y^{gen} using fixed parameter $\gamma \in [0,1]$ (loss ratio)

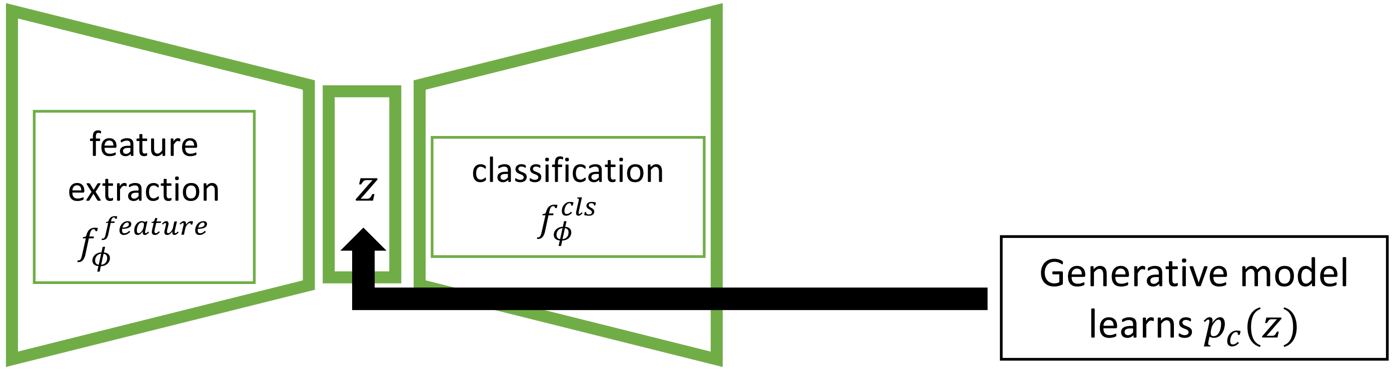
GenLabel Algorithm – Generative model learns in latent feature

Algorithm 2 *GenLabel* (using generative models for the latent feature)

Input Dataset $S = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{n}$, input feature set $X = \{\boldsymbol{x}_i\}_{i=1}^{n}$, learning rate η, loss ratio γ
Output Trained discriminative model f_θ

$\theta \leftarrow$ Random initial parameter for model $f_\theta = f_\theta^{\text{cls}} \circ f_\theta^{\text{feature}}$
$\phi \leftarrow$ Vanilla-trained parameter for model $f_\phi = f_\phi^{\text{cls}} \circ f_\phi^{\text{feature}}$
$p_c(\boldsymbol{z}) \leftarrow$ Density estimated by generative model for latent feature $\boldsymbol{z} \in f_\phi^{\text{feature}}(X)$, conditioned on class $c \in [k]$
for $(\boldsymbol{x}_i, \boldsymbol{y}_i), (\boldsymbol{x}_j, \boldsymbol{y}_j) \in S$ **do**
    $(\boldsymbol{x}^{\text{mix}}, \boldsymbol{y}^{\text{mix}}) \leftarrow (\lambda \boldsymbol{x}_i + (1-\lambda)\boldsymbol{x}_j, \lambda \boldsymbol{y}_i + (1-\lambda)\boldsymbol{y}_j)$
    $\boldsymbol{z}^{\text{mix}} \leftarrow f_\phi(\boldsymbol{x}^{\text{mix}})$
    $\boldsymbol{y}^{\text{gen}} \leftarrow \sum_{c=1}^{k} \frac{p_c(\boldsymbol{z}^{\text{mix}})}{\sum_{c'=1}^{k} p_{c'}(\boldsymbol{z}^{\text{mix}})} \boldsymbol{e}_c$
    $\theta \leftarrow \theta - \eta \nabla_\theta \{\gamma \cdot \ell_{\text{CE}}(\boldsymbol{y}^{\text{gen}}, f_\theta(\boldsymbol{x}^{\text{mix}})) + (1-\gamma) \cdot \ell_{\text{CE}}(\boldsymbol{y}^{\text{mix}}, f_\theta(\boldsymbol{x}^{\text{mix}}))\}$
end for

GenLabel algorithm when Generative model learns the density ($p_C(z)$) in the latent feature space



GenLabel Algorithm – Variant (used for image datasets)

Algorithm 3 GenLabel (learning generative/discriminative models at the same time)

Input Data D, mix function $\text{mix}(\cdot)$, learning rate η, loss ratio γ, memory ratio β, batch size B, max iteration T
Output Trained model $f_\theta = f_\theta^{\text{cls}} \circ f_\theta^{\text{feature}}$

$\theta \leftarrow$ Random initial model parameter, $\quad \pi \leftarrow$ Permutation of $[B]$
$(\boldsymbol{\mu}_c^{(0)}, \boldsymbol{\Sigma}_c^{(0)}) \leftarrow (\mathbf{0}, \boldsymbol{I}_d)$ for $c \in [k]$
for iteration $t = 1, 2, \cdots, T$ **do**
  $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{B} \leftarrow$ Randomly chosen batch samples in D
  for class $c \in [k]$ **do**
    $S_c \leftarrow \{i : \boldsymbol{y}_i = \boldsymbol{e}_c\}$
    $\boldsymbol{\mu}_c^{(t)} \leftarrow \frac{1}{|S_c|} \sum_{i \in S_c} f_\theta^{\text{feature}}(\boldsymbol{x}_i), \quad \boldsymbol{\mu}_c^{(t)} \leftarrow (1-\beta)\boldsymbol{\mu}_c^{(t)} + \beta \boldsymbol{\mu}_c^{(t-1)}$
    $\boldsymbol{\Sigma}_c^{(t)} \leftarrow \frac{1}{|S_c|} \sum_{i \in S_c} (f_\theta^{\text{feature}}(\boldsymbol{x}_i) - \boldsymbol{\mu}_c^{(t)})(f_\theta^{\text{feature}}(\boldsymbol{x}_i) - \boldsymbol{\mu}_c^{(t)})^T$
    $\boldsymbol{\Sigma}_c^{(t)} \leftarrow \frac{1}{d}\text{trace}(\boldsymbol{\Sigma}_c^{(t)})\boldsymbol{I}_d, \quad \boldsymbol{\Sigma}_c^{(t)} \leftarrow (1-\beta)\boldsymbol{\Sigma}_c^{(t)} + \beta\boldsymbol{\Sigma}_c^{(t-1)}$
  end for
  for sample index $i \in [B]$ **do**
    $(\boldsymbol{x}_i^{\text{mix}}, \boldsymbol{y}_i^{\text{mix}}) \leftarrow \text{mix}((\boldsymbol{x}_i, \boldsymbol{y}_i), (\boldsymbol{x}_{\pi(i)}, \boldsymbol{y}_{\pi(i)}))$
    $p_c \leftarrow \det(\boldsymbol{\Sigma}_c^{(t)})^{-1/2} \exp\{-(f_\theta^{\text{feature}}(\boldsymbol{x}_i^{\text{mix}}) - \boldsymbol{\mu}_c^{(t)})^T (\boldsymbol{\Sigma}_c^{(t)})^{-1} (f_\theta^{\text{feature}}(\boldsymbol{x}_i^{\text{mix}}) - \boldsymbol{\mu}_c^{(t)})\}$ for $c \in [k]$
    $c_1 \leftarrow \arg\min_{c \in [k]} p_c, \quad c_2 \leftarrow \arg\min_{c \in [k] \setminus \{c_1\}} p_c$
    $\boldsymbol{y}_i^{\text{gen}} \leftarrow \frac{p_{c_1}}{p_{c_1} + p_{c_2}} \boldsymbol{e}_{c_1} + \frac{p_{c_2}}{p_{c_1} + p_{c_2}} \boldsymbol{e}_{c_2}$
  end for
  $\theta \leftarrow \theta - \eta \sum_{i \in [B]} \nabla_\theta \{\gamma \cdot \ell_{\text{CE}}(\boldsymbol{y}_i^{\text{gen}}, f_\theta(\boldsymbol{x}_i^{\text{mix}})) + (1-\gamma) \cdot \ell_{\text{CE}}(\boldsymbol{y}_i^{\text{mix}}, f_\theta(\boldsymbol{x}_i^{\text{mix}}))\}$
end for

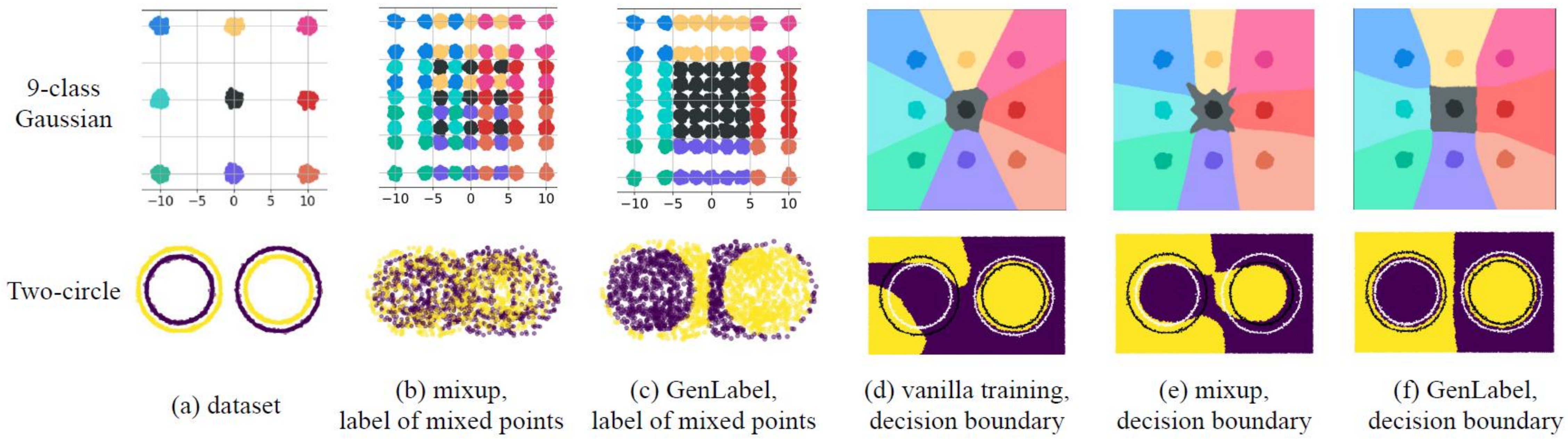
GM parameter update

Use top-2 classes only

Note:
Σ_c is approximated as a multiple of identity matrix.

- Update GM model parameters μ_c, Σ_c at each batch training
- Use only top-2 classes c_1, c_2 satisfying $p_{c_1} \geq p_{c_2} \geq p_c$ for $c \in [k] - \{c_1, c_2\}$

GenLabel Effect – Manifold intrusion / Margin reduction



Effect : Manifold intrusion / margin reduction

- Perform experiment using 9-class gaussian / Two-circle datasets
- (b), (c) represents original top 1-label of mixed points for y^{mix} (b) and y^{gen} (c)

  (Note : top 1-label : $y^{top-1} = \; argmax_{c\in[k]}\, y_c$ where $y = [y_1, \ldots, y_k]$

- GenLabel algorithm helps to reduce manifold intrusions and guide the classifier to have a larger margin compared to vanilla / mix-up training

GenLabel Analysis – Margin reduction

What is the relationship between GenLabel and logistic labeling? (for gaussian data)

Proposition 1

Consider a binary classification problem when the class-conditional data distribution is $(x|y=0) \sim N(0, \sigma^2)$ and $(x|y=1) \sim N(1, \sigma^2)$. Let $x^{mix} = \lambda$ be the mixed point generated by mix-up. For small $\sigma > 0$, the label of x^{mix} for mix-up and GenLabel are

$$y^{mix} = \lambda, \qquad y^{gen} = \frac{1}{1 + \exp\left(-\frac{(\lambda - \frac{1}{2})}{\sigma^2}\right)}$$

Note

- When GM is used as generative model, then GenLabel is the same as softmax labeling

$$y^{gen} = softmax\left(-\frac{\left(x^{mix} - \mu_1\right)^2}{2\sigma_1^2}, \dots, -\frac{\left(x^{mix} - \mu_k\right)^2}{2\sigma_k^2}\right)$$

GenLabel Analysis – Adversarial robustness

GenLabel still preserves adversarial robustness?

Analysis setting

- d-dimensional gaussian dataset : $(x|y=0) \sim N(-e_1, \frac{\Sigma}{\sigma_1^2})$ and $(x|y=1) \sim N\left(e_1, \frac{\Sigma}{\sigma_2^2}\right)$

  where $\Sigma_{ij} = 1$ for $i = j$ and $\Sigma_{ij} = \tau$ for $i \neq j$ (for $\tau \in (-1,1)$)
- Consider the loss function : $l\big(\theta, (x,y)\big) = h\big(f_\theta(x)\big) - y f_\theta(x)$

  where $h(w) = \log(1 + \exp(w))$ [logistic regression]
- $L_n^{gen}(\theta, S) = \frac{1}{n^2} \sum_{i,j=1}^{n} \mathbb{E}_\lambda[l\left(\theta, z_{ij}^{gen}\right)]$ where $z_{ij}^{gen} = (x_{ij}^{mix}, y_{ij}^{gen})$
- $L_n^{adv}(\theta, S) = \frac{1}{n} \sum_{i=1}^{n} \max_{\|\delta_i\|_2 \leq \epsilon\sqrt{d}} l(\theta, (x_i + \delta_i, y_i))$
- Assumptions required for proof

  1. $\tau \notin \{-\frac{1}{d-1}, -\frac{1}{d-2}\}$
  2. $\sigma_2 = c\sigma_1$ with $2 - \sqrt{3} < c < 2 + \sqrt{3}$

GenLabel Analysis – Adversarial robustness

Lemma 1

The second order Taylor approximation of the GenLabel loss under analysis setting is given by

$$\tilde{L}_n^{gen}(\theta, S) = L_n^{std}(\theta, S) + R_1^{gen}(\theta, S) + R_2^{gen}(\theta, S) + R_3^{gen}(\theta, S)$$

where

$$R_1^{gen}(\theta, S) = \frac{1}{n}\sum_{i=1}^{n} A_{\sigma_1,c,\tau,d}^{i}\left(h'\left(f_\theta(x_i)\right) - y_i\right)\nabla f_\theta(x_i)^T \mathbb{E}_{r_x\sim D_x}[r_x - x_i]$$

$$R_2^{gen}(\theta, S) = \frac{1}{2n}\sum_{i=1}^{n} B_{\sigma_1,c,\tau,d}^{i}\, h''^{(f_\theta(x_i))}\nabla f_\theta(x_i)^T \mathbb{E}_{r_x\sim D_x}[(r_x - x_i)(r_x - x_i)^T]\nabla f_\theta(x_i)$$

$$R_3^{gen}(\theta, S) = \frac{1}{2n}\sum_{i=1}^{n} B_{\sigma_1,c,\tau,d}^{i}\,(h'\left(f_\theta(x_i)\right) - y_i)\, \mathbb{E}_{r_x\sim D_x}[(r_x - x_i)^T\nabla^2 f_\theta(x_i)(r_x - x_i)]$$

And, $A_{\sigma_1,c,\tau,d}^{i}$ and $B_{\sigma_1,c,\tau,d}^{i}$ are two constants

GenLabel Analysis – Adversarial robustness

Theorem 1

Consider the logistic regression setting having $f_\theta(x) = \theta^T x$ where $\theta \in \Theta = \left\{\theta \in \mathbb{R}^d : (2y_i - 1) f_\theta(x_i) \geq 0 \text{ for all } i = 1,2,\ldots,n\right\}$. suppose there exists a constant $c_X > 0$ such that $\|x_i\|_2 > c_x$ for all $i \in \{1,2..n\}$. Then, for large σ_1, we have

$$\tilde{L}_n^{mix}(\theta, S) > \tilde{L}_n^{gen}(\theta, S) \geq \frac{1}{n}\sum_{i=1}^{n} \tilde{l}_{adv}\left(\delta_{gen}, (x_i, y_i)\right)$$

Where $\delta_{gen} = R \cdot c_x A^i_{\sigma_1,c,\tau,d}$ with $R = \min_{i\in\{1,\ldots,n\}} |\cos(\theta, x_i)|$ and $A^i_{\sigma_1,c,\tau,d}$ is constant

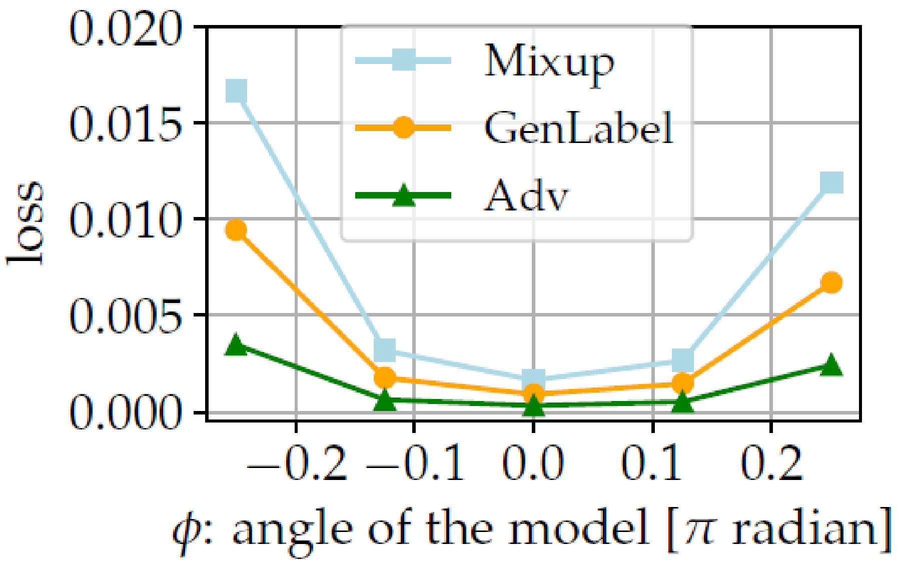
Recall (second order taylor approximation of $L_n^{adv}(\theta, S)$, Zhang et al., 2021) [under logistic regression setting]

1. $\tilde{l}_{adv}\big(\eta, (x, y)\big) = l\big(\theta, (x, y)\big) + \eta\left|g\left(x^T\theta\right) - y\right| \cdot \|\theta\|_2 + \frac{\eta^2}{2} g(x^T\theta)(1 - g\left(x^T\theta\right)) \cdot \|\theta\|_2^2$
   where $g(s) = \frac{e^s}{1+e^s}$ is logistic function
2. The second order Taylor approximation of $L_n^{adv}(\theta, S)$ is $\frac{1}{n}\sum_{i=1}^{n} \tilde{l}_{adv}\left(\epsilon\sqrt{d}, (x_i, y_i)\right)$, where $x \in \mathbb{R}^d$ and $y \in \{0,1\}$

GenLabel Analysis – Adversarial robustness

Experiment

- Dataset $S = \{(x_i^+, +1), (x_i^-, -1)\}_{i=1}^{20}$, where $x_i^+ \sim N([+1,0], \frac{1}{100} I_2)$, $x_i^- \sim N([-1,0], \frac{1}{100} I_2)$
- Obviously, θ = (10,0) achieves minimum loss (or $\phi = 0$, when $\theta = (10cos\phi, 10sin\phi)$
- Perform logistic regression and scan each losses for $\phi \in \left[-\frac{\pi}{4}, \frac{\pi}{4}\right]$, adversarial attack size = δ_{gen}
- Result : GenLabel loss is strictly smaller than mix-up loss -> coincide with Thm1



GenLabel Analysis – Adversarial robustness

Theorem 2

Consider the FC ReLU network setting having $f_\theta(x) = \beta^T \sigma\left(W_{N-1} \cdots \left(W_2 \sigma(W_1 X)\right)\right)$, where σ : ReLU function, W_i : parameter matrix, β : parameter vector, and $\theta = (W_1, \dots W_{N-1}, \beta) \in \Theta = \left\{\theta \in \mathbb{R}^d : (2y_i - 1) f_\theta(x_i) \geq 0 \text{ for all } i = 1,2, \dots, n\right\}$. Suppose there exists a constant $c_X > 0$ such that $\|x_i\|_2 > c_x$ for all $i \in \{1,2..n\}$. Then, for large σ_1, we have

$$\tilde{L}_n^{mix}(\theta, S) > \tilde{L}_n^{gen}(\theta, S) \geq \frac{1}{n} \sum_{i=1}^{n} \tilde{l}_{adv}\left(\delta_{gen}, (x_i, y_i)\right)$$

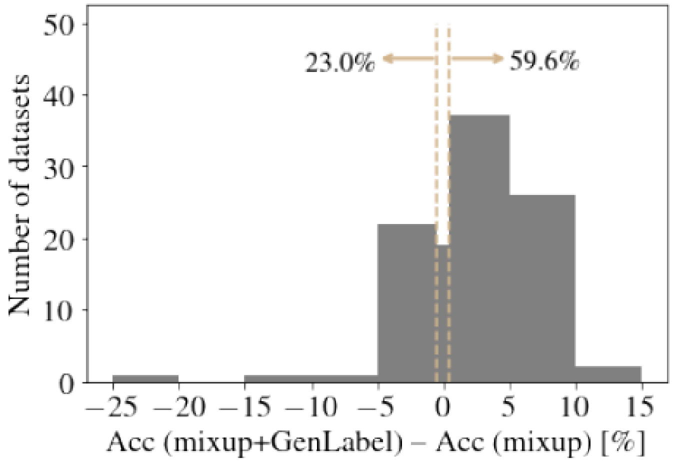
Where $\delta_{gen} = R \cdot c_x A^i_{\sigma_1, c, \tau, d}$ with $R = \min_{i \in \{1, \dots, n\}} |\cos(\nabla f_\theta(x_i), x_i)|$ and $A^i_{\sigma_1, c, \tau, d}$ is constant

Recall (second order taylor approximation of $L_n^{adv}(\theta, S)$, Zhang et al., 2021) [under FC ReLU network setting]

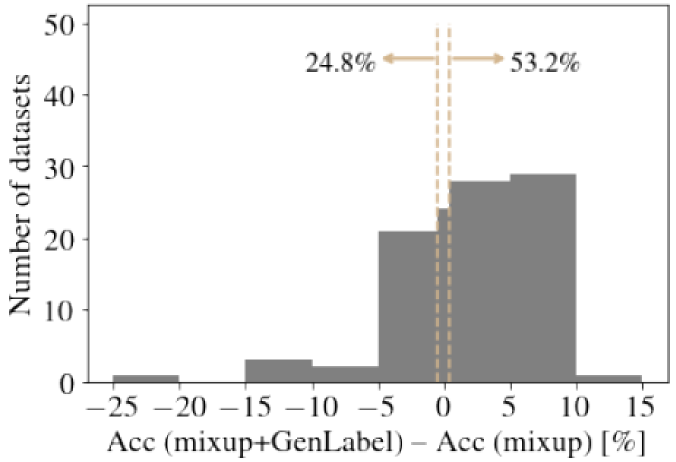
- $\tilde{l}_{adv}(\delta, (x, y)) = l(\theta, (x, y)) + \delta \left| g(x^T \theta) - y \right| \cdot \|\nabla f_\theta(x)\|_2 + \frac{\delta^2 d}{2} |h''(f_\theta(x))| \cdot \|\nabla f_\theta(x)\|_2^2$
  where $g(s) = \frac{e^s}{1+e^s}$ is logistic function

Experimental results – Generalization performance

accuracy(GenLabel(CV)) – accuracy(mix-up)



(a) $\alpha = 1.0$



(b) $\alpha = 2.0$

Note : α is mix-up hyper parameter (i.e : $\lambda \sim Beta(\alpha, \alpha)$)

Dataset : 109 OpenML dataset
Algorithm : logistic regression

Statistics of the accuracy of GenLabel and baselines

Mixup+GenLabel (CV) versus	**Vanilla**	**Adamixup**	**Mixup**	**Mixup + exclude MI**	**Generative Classifier (GM)**
Higher ($> 0.5\%$)	**37.6%**	**46.8%**	**59.6%**	**56.9%**	**44.0%**
On-par (within 0.5%**)**	31.2%	25.7%	17.4%	16.5%	23.9%
Lower ($< -0.5\%$)	31.2%	27.5%	23.0%	26.6%	32.1%

Experimental results – Generalization performance (logistic)

Accuracy on selected OpenML datasets

Methods \ OpenML Dataset ID	36	61	721	778	817	830	855	869
Generative classifier (GM)	89.75±0.00	95.56±0.00	78.33±0.00	89.47±0.00	60.00±0.00	78.67±0.00	65.33±0.00	73.33±0.00
Mixup	88.98±0.78	88.00±1.09	79.33±0.82	95.00±1.53	60.00±0.00	76.27±1.31	66.00±1.74	71.73±3.79
Mixup + Excluding MI	89.21±0.52	93.33±0.00	79.67±0.67	95.00±1.53	61.33±2.67	78.13±1.36	66.40±1.37	72.00±3.55
Mixup + GenLabel (GM)	92.21±0.58	96.00±1.67	**81.00**±1.33	**97.11**±0.98	64.00±5.33	**86.13**±1.36	66.40±2.88	**76.27**±3.17
Mixup + GenLabel (KDE)	**92.64**±0.26	96.00±0.89	79.67±1.25	96.05±0.83	**66.67**±5.96	77.33±4.84	**67.60**±0.90	74.53±3.99
Mixup + GenLabel (CV)	92.55±0.15	**96.44**±1.09	80.33±1.63	96.05±1.86	65.33±4.99	84.53±1.81	67.33±2.76	73.87±2.13

Methods \ OpenML Dataset ID	885	907	915	925	938	1006	40710	40981
Generative classifier (GM)	95.00±0.00	47.50±0.00	42.11±0.00	90.72±0.00	92.31±0.00	77.78±0.00	69.23±0.00	73.91±0.00
Mixup	94.50±1.00	44.67±3.14	46.11±3.08	92.99±1.37	90.77±5.76	80.00±0.00	68.13±0.70	74.78±0.56
Mixup + Excluding MI	94.50±1.00	44.00±4.06	47.79±3.37	93.20±1.40	89.23±6.15	77.33±4.31	68.57±1.12	74.69±0.84
Mixup + GenLabel (GM)	**97.00**±1.00	47.83±3.56	46.74±7.37	93.61±0.77	**98.46**±3.08	81.33±1.09	69.67±0.54	75.07±1.08
Mixup + GenLabel (KDE)	96.50±1.22	45.67±4.39	46.11±7.57	93.61±0.77	**98.46**±3.08	80.00±3.44	69.45±0.44	**76.43**±0.36
Mixup + GenLabel (CV)	**97.00**±1.00	**48.83**±4.46	**48.42**±3.82	**94.23**±0.82	95.38±6.15	**81.78**±0.89	**70.11**±0.82	76.23±0.71

Robust accuracy under FGSM attack on openML dataset

Methods \ OpenML Dataset ID	3	223	312	313	346	463	753	834
Vanilla	45.61±14.11	11.12±4.33	21.78±9.27	12.23±2.34	42.92±12.29	71.60±13.57	28.35±6.60	14.78±3.73
Mixup	43.36±14.10	11.16±5.22	23.40±9.82	14.22±3.73	40.83±11.90	68.80±12.04	29.42±6.03	15.20±3.91
Mixup + GenLabel (GM)	**51.87**±5.45	**13.99**±6.65	**36.61**±14.07	**18.39**±2.57	**52.92**±12.79	**84.46**±2.62	**38.23**±6.29	**21.94**±3.69

Methods \ OpenML Dataset ID	952	954	978	987	988	1022	1045	1059
Vanilla	30.85±3.12	71.40±4.86	28.89±12.62	68.04±18.17	50.35±12.41	35.85±12.20	57.89±10.42	68.05±21.90
Mixup	31.38±3.12	69.32±5.02	34.89±11.32	67.82±14.56	53.54±10.91	41.25±12.11	59.88±14.38	67.18±25.48
Mixup + GenLabel (GM)	**42.19**±7.61	**85.16**±7.55	**43.74**±15.93	**83.21**±1.77	**63.76**±9.89	**56.61**±15.00	**66.22**±16.60	**74.64**±20.14

Experimental results – Generalization performance (FC ReLU)

Accuracy on selected OpenML datasets

Methods \ OpenML Dataset ID	719	770	774	804	818	862	900	906
Vanilla	71.62±5.55	65.58±11.38	59.34±5.16	81.44±8.45	87.56±19.75	81.51±6.06	61.00±1.62	53.74±2.21
Mixup	70.89±5.47	65.26±11.11	59.80±5.04	80.05±10.22	88.17±15.68	80.40±5.51	60.99±2.05	53.74±2.51
Mixup+Excluding MI	71.62±5.55	64.15±9.62	59.50±7.09	80.05±10.22	88.49±15.56	79.21±8.03	60.74±2.31	53.50±2.31
Generative classifier (GM)	67.90±6.00	51.69±10.93	49.22±6.03	71.59±10.21	82.66±18.41	67.62±10.23	57.50±6.31	48.26±3.02
Mixup+GenLabel (GM)	**73.10**±7.66	**66.69**±11.06	**59.95**±5.28	**81.57**±9.57	**89.15**±17.40	**82.70**±7.56	**61.24**±2.49	**54.21**±4.75

Methods \ OpenML Dataset ID	908	949	956	1011	1014	1045	1055	1075
Vanilla	54.00±1.70	85.69±0.46	68.90±2.52	96.14±3.46	80.55±0.25	94.53±1.96	78.77±4.36	92.35±2.23
Mixup	55.00±1.95	85.69±0.46	69.88±4.01	96.14±3.46	80.55±0.25	94.53±1.96	78.77±4.36	92.35±2.23
Mixup+Excluding MI	54.50±2.34	85.69±0.46	69.88±4.01	**96.43**±3.74	80.55±0.25	94.53±1.96	78.77±4.36	92.35±2.23
Generative classifier (GM)	47.99±3.86	65.59±15.61	67.97±2.03	95.53±2.72	48.43±4.80	94.53±1.96	40.75±9.62	90.83±2.73
Mixup+GenLabel (GM)	**55.75**±1.48	**87.14**±3.66	**70.81**±3.75	**96.43**±3.74	**80.80**±0.61	**95.19**±1.57	**79.81**±4.01	**93.11**±2.41

Robust accuracy under FGSM attack on openML dataset

Methods \ OpenML Dataset ID	312	715	718	723	797	806	837	866
Vanilla	54.22±14.88	42.70±3.62	28.40±2.04	39.90±3.54	32.79±3.31	32.69±3.51	30.30±2.42	41.10±2.07
Mixup	66.23±11.75	44.10±3.27	40.29±3.27	41.39±3.45	38.70±3.24	35.99±2.78	30.60±1.92	45.80±1.67
Mixup+GenLabel (GM)	**82.09**±0.08	**54.00**±0.95	**54.70**±0.89	**52.10**±0.89	**55.10**±0.73	**53.39**±2.44	**49.99**±1.98	**58.00**±0.32

Methods \ OpenML Dataset ID	871	909	917	1038	1043	1130	1138	1166
Vanilla	32.48±4.20	39.26±3.32	37.90±3.51	16.49±1.85	57.65±2.54	47.71±8.86	53.97±4.77	33.92±5.39
Mixup	32.07±2.69	39.22±5.52	41.20±3.71	14.45±2.25	61.95±1.82	49.20±9.45	62.00±4.31	46.21±5.58
Mixup+GenLabel (GM)	**41.42**±0.73	**50.50**±0.61	**51.50**±2.30	**26.04**±2.87	**74.97**±0.16	**85.57**±2.01	**88.70**±2.48	**74.62**±3.09

Experimental results – Generalization/robustness for image data

Generalization and robustness performances on real image datasets

Methods	MNIST		CIFAR-10		CIFAR-100		TinyImageNet-200	
	Robust	Clean	Robust	Clean	Robust	Clean	Robust	Clean
Vanilla	48.17 ± 13.1	99.34 ± 0.03	16.89 ± 0.98	94.57 ± 0.25	17.19 ± 0.20	74.48 ± 0.28	13.19 ± 0.19	58.13 ± 0.09
AdaMixup	-	99.32 ± 0.05	-	95.45 ± 0.13	-	-	-	-
Mixup	55.44 ± 1.80	99.27 ± 0.03	11.65 ± 1.96	95.68 ± 0.06	18.44 ± 0.45	77.65 ± 0.30	14.91 ± 0.48	59.46 ± 0.30
Mixup+GenLabel	56.54 ± 1.03	99.36 ± 0.06	14.32 ± 1.23	**96.09** ± 0.01	**19.58** ± 0.71	78.04 ± 0.21	**15.34** ± 0.30	59.78 ± 0.09
Manifold mixup	55.56 ± 1.53	99.32 ± 0.04	18.14 ± 1.88	94.78 ± 0.49	19.25 ± 0.61	78.61 ± 0.17	14.78 ± 0.28	59.87 ± 0.63
Manifold mixup+GenLabel	**56.62** ± 1.31	**99.37** ± 0.07	**18.91** ± 1.26	95.10 ± 0.10	19.28 ± 1.04	**78.99** ± 0.54	15.19 ± 0.22	**60.02** ± 0.25

Robust accuracy was analyzed under AutoAttack (Croce, 2020)

GenMix

Is the linear combination of two examples are good way for data augmentation?

Intuition - GenMix

- How about making new mixing data points using generative models?
- Make new mixing data x^{mix} that satisfy $p_i\left(x^{mix}\right): p_j(x^{mix})$ = $(1-\lambda): \lambda$ for arbitrary defined $\lambda \in [0,1]$
- Suggested two types of generative model : GM / GAN
- Goal : find virtual data x^{mix} which satisfy (for small margin ϵ)

$$\left|\frac{p_j\left(x^{mix}\right)}{p_i(x^{mix}) + p_j(x^{mix})} - \lambda\right| \leq \epsilon$$

- Perform manifold intrusion removal : remain x^{mix} only when $\min\left\{p_i\left(x^{mix}\right), p_j\left(x^{mix}\right)\right\} \geq p_l(x^{mix})$ for all $l \in [k] - \{i, j\}$
- Train model with data set $D \cup D_{mixup}$

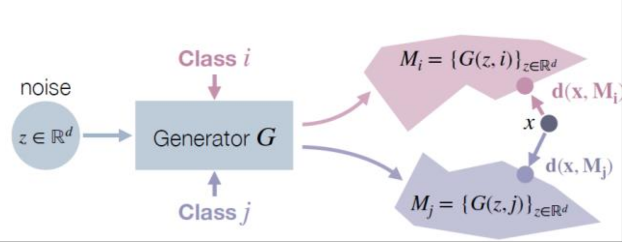
GenMix-Algorithm

Algorithm 4 GenMix

Input Training data D, Number of augmented data n_{aug}, likelihood-ratio margin $\varepsilon > 0$, mixing coefficient $\lambda \in [0, 1]$
Output Trained model $f(\cdot)$, Augmented data D_{mixup}

  $p_c \leftarrow$ Data distribution of class c learned by generative model
  $D_{\text{mixup}} \leftarrow \{\}$
  for classes $i \in [k]$ and $j \in [k] \backslash \{i\}$ **do**
    $n \leftarrow 0$
    while $n < n_{\text{aug}}$ **do**
      Find point $\boldsymbol{x}$ satisfying $\left|\frac{p_j(\boldsymbol{x})}{p_i(\boldsymbol{x})+p_j(\boldsymbol{x})} - \lambda\right| \leq \varepsilon$
      $p_\ell \leftarrow p_\ell(\boldsymbol{x})$ for $\ell \in [k]$
      if $\min\{p_i, p_j\} \geq p_\ell \quad \forall \ell \in [k] \backslash \{i, j\}$ **then**
        $D_{\text{mixup}} \leftarrow D_{\text{mixup}} \cup \{(\boldsymbol{x}, \frac{p_i}{p_i+p_j}\boldsymbol{e}_i + \frac{p_j}{p_i+p_j}\boldsymbol{e}_j)\}$
        $n \leftarrow n + 1$
      end if
    end while
  end for
  $f \leftarrow$ model training with $D \cup D_{\text{mixup}}$

GAN case (use CGAN)



← How to find this?

GM case:

Equivalent to solve $\left|\log\frac{p_j(x)}{p_i(x)+p_j(x)}\right| \cong \lambda$

(Has closed form solution applying ***quadratic discriminant analysis***)

GAN case : (assume spherical gaussian noise model, and use Conditional GAN)

1. $p_c(x) \cong \exp(-d(x, M_c))$ where M_C is generated manifold of class c
2. Equivalent to solve $\min_x \left(d(x, M_j) - d(x, M_i) - \log\left(\frac{1}{\lambda} - 1\right)\right)^2$
3. Above problem can be solved using Gradient Descent

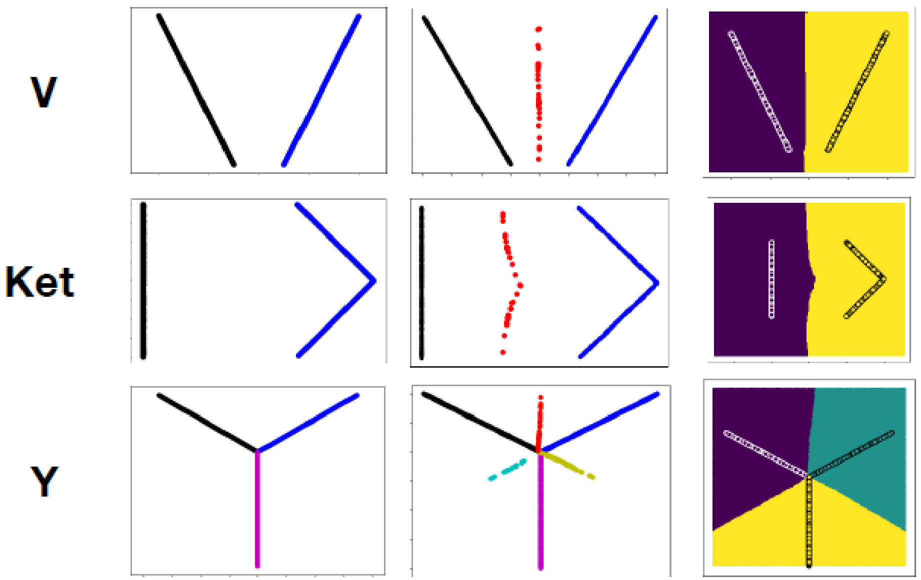
$$\min_\delta \left|d(x+\delta, M_j) - d(x+\delta, M_i) - \log(\frac{1}{\lambda} - 1)\right|^2 : d(x+\delta, m_c^*) \gg d(x, x+\delta)$$

$for\ c \in \{i, j\}$, where $m_c^* = argmin_{m \in M_c} d(x, m)$ and x is a random initial point.

Note: $d(x, M_c)$ is measured by inverting generator of the GAN

GenMix-Results

Result of GenMix+GAN on V, Ket, Y datasets



(a) Training data (b) Augmented data (c) Decision Boundary

Classification error comparison with other methods

Schemes / Datasets	Circle (2D)	Circle (3D)	MNIST 7/9 (n_{train}=500)
Vanilla Training	8.60 ± 4.84	1.40 ± 0.54	2.72 ± 0.20
Mixup	7.98 ± 2.94	5.22 ± 1.99	2.32 ± 0.40
Manifold-mixup	7.34 ± 1.43	0.94 ± 0.75	3.88 ± 0.53
GenMix+GAN	**4.90** ± 0.12	**0.22** ± 0.06	**2.13** ± 0.12

(a) : mid point generated by GenMix+GAN
(b), (c) : decision boundary for GenMix+GAN/ vanilla mix-up

Comparison between generative-model based mix-up(GenMix +GAN) and vanilla mix-up