Mix-up based on data valuation score

-Summary-

Table of Contents

① Theoretical parts

1. Effect of Mix-up via Taylor expansion (Prior works)
2. Linkage between Mix-up and data valuation score

② Observation parts

1. Hard mix-up pairing training + presence of frequently incorrect directions by class (based on forgetting score)
2. Data valuation of mix-up samples (based on GraNd, EL2N score)

③ Experiments parts

1. Hard mix-up with adjusting hard data portion
2. RegMixUp with λ scheduler (**best result**)

④ Future works

Theoretical parts (Effect of Mix-up via Taylor expansion)

- There are some papers dealing with theoretical analysis of mix-up technique :

[Brief summary & idea]

1. How Does Mixup Help With Robustness And Generalization [Zhang et al., ICLR 2021]
   ① **Showed regularization effect of mix-up using Taylor expansion** on mix-up loss.
   ② Given adversarial attack size, **demonstrated mix up loss is the upper bound of adversarial loss**.
2. On Mixup Regularization [Carratino et al., JMLR 2022] (Our focus)
   ① Showed Mix-up loss can be written as a perturbed ERM loss.
   ② **Showed regularization effect** of mix-up using Taylor expansion on various training case : Cross entropy loss / logistic regression loss / MSE loss
   ③ Suggested '**Approximated Mixup'** by dropping out the regularization term, which is an intermediate compromise of Mixup and ERM training in the view of regularization.

Theoretical parts (Effect of Mix-up via Taylor expansion)

- There are some papers dealing with theoretical analysis of mix-up technique :

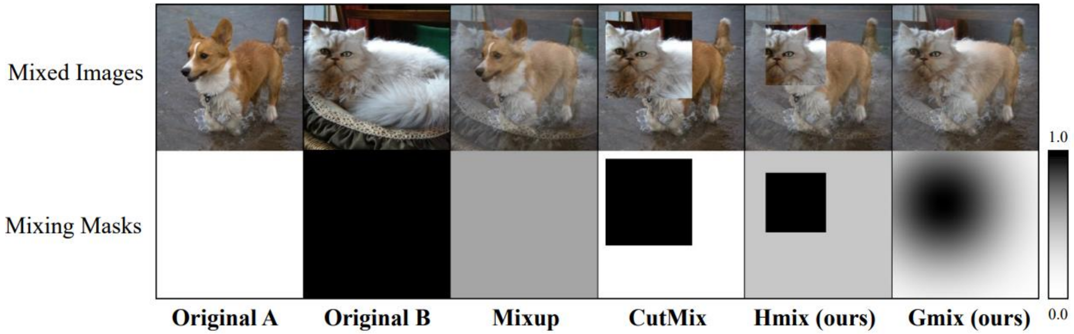
[Brief summary & idea]

3. A Unified Analysis of Mixed Sample Data Augmentation [C. Park et al., NeurIPS 2022]

① Based on [Zhang et al., ICLR 2021], suggest an unified framework of vision field mix-up (such as CutMix, naïve mixup), which **demonstrates generalized input gradient / hessian regularization effect of vision field mix-up.**

② Propose H-mix, G-mix that uses CutMix and naïve Mix-up simultaneously

1. H-mix : Use CutMix and naïve Mix-up simultaneously
2. G-mix : Use CutMix and naïve Mix-up simultaneously + smooth crop boundary with gaussian kernel density
3. ~1% test acc ↑ in CIFAR-100 compared to CutMix



Description of H-mix and G-mix

Theoretical parts (Linkage between Mix-up and data valuation score)

- Linkage between Mix-up and data valuation score (from 'On Mixup Regularization')

Mix-up loss : $\boldsymbol{\mathcal{E}^{Mixup}(f) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{i=1}^{n}\mathbb{E}_{\lambda}[l(\lambda y_i + (1-\lambda)y_j, f(\lambda x_i + (1-\lambda)x_j)]}$

Reformulation of above equation

Reformulated Mix-up loss : $\boldsymbol{\mathcal{E}^{Mixup}(f) = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{\theta,j}\left[l\left(\tilde{y}_i + \epsilon_{i,j}, f(\tilde{x}_i + \delta_{i,j})\right)\right]}$ **(Mixup loss as perturbed ERM)**

where $\delta_{i,j} \coloneqq \theta x_i + (1-\theta)x_j - \mathbb{E}_{\theta,j}[\theta x_i + (1-\theta)x_j]$ *(perturbation of mixed input)*

$\epsilon_{i,j} \coloneqq \theta y_i + (1-\theta)y_j - \mathbb{E}_{\theta,j}[\theta y_i + (1-\theta)y_j]$ *(perturbation of mixed label)*

$\tilde{x}_i \coloneqq \mathbb{E}_{\theta,j}[\theta x_i + (1-\theta)x_j]$ *(expected mix-up point of x_i)*

$\tilde{y}_i \coloneqq \mathbb{E}_{\theta,j}[\theta y_i + (1-\theta)y_j]$ *(expected mix-up point of x_j)*

Also, $\theta \sim Beta_{[\frac{1}{2},1]}(\alpha,\alpha)$ (Truncated beta distribution), $j \sim Unif([n])$

Multivariate Taylor expansion w.r.t $\epsilon_{i,j}, \delta_{i,j}$ + Plugging C.E loss

Theoretical parts (Linkage between Mix-up and data valuation score)

- Linkage between Mix-up and data valuation score (from 'On Mixup Regularization')

Approximated Mix-up C.E loss : $\boldsymbol{\mathcal{E}_Q^{Mixup}(f) = \frac{1}{n}\sum_{i=1}^{n} l^{CE}\left(\tilde{y}_i, f(\tilde{x}_i)\right) + R_1^{CE}(f) + R_2^{CE}(f) + R_3^{CE}(f)}$

where

$$R_1^{CE}(f) = \frac{1}{2n}\sum_{i=1}^{n} < \Sigma_{\tilde{x}\tilde{x}}^{(i)}, \left(\nabla f(\tilde{x}_i) - J^{(i)}\right)^T H\left(f(\tilde{x}_i)\right)\left(\nabla f(\tilde{x}_i) - J^{(i)}\right) >_F$$

$$R_2^{CE}(f) = \frac{1}{2n}\sum_{i=1}^{n} < \Sigma_{\tilde{x}\tilde{x}}^{(i)}, \left(S\left(f(\tilde{x}_i)\right) - \tilde{y}_i\right)^T \nabla^2 f(\tilde{x}_i) >_F$$

$$R_3^{CE}(f) = -\frac{1}{2n}\sum_{i=1}^{n} < \Sigma_{\tilde{x}\tilde{y}}^{(i)} \left(\Sigma_{\tilde{x}\tilde{x}}^{(i)}\right)^{-1} \Sigma_{\tilde{y}\tilde{x}}^{(i)}, H\left(f(\tilde{x}_i)\right) >_F$$

Notations :

1. $S(u) \coloneqq Softmax(u)$
2. $H(u) \coloneqq diag\left(S(u)\right) - S(u)S(u)^T$
3. $J^{(i)} \coloneqq H\left(f(\tilde{x}_i)\right)^{-1}\Sigma_{\tilde{y}\tilde{x}}^{(i)}\left(\Sigma_{\tilde{x}\tilde{x}}^{(i)}\right)^{-1}$

Note : $\Sigma_{\tilde{x}\tilde{x}}^{(i)} \coloneqq \mathbb{E}_{\theta,j}\left[\delta_i\delta_i^T\right]$, $\Sigma_{\tilde{x}\tilde{y}}^{(i)} \coloneqq \mathbb{E}_{\theta,j}\left[\delta_i\epsilon_i^T\right]$, $\Sigma_{\tilde{y}\tilde{y}}^{(i)} \coloneqq \mathbb{E}_{\theta,j}\left[\epsilon_i\epsilon_i^T\right]$

Remark :$\boldsymbol{R_1^{CE}(f)}$, $\boldsymbol{-R_3^{CE}(f)}$ is positive with high probability in practice. (not yet verified on $\boldsymbol{R_2^{CE}(f)}$…)
(can be negative; similar issue can happens for suggested regularization terms on [Zhang et al, 2020])

Theoretical parts (Linkage between Mix-up and data valuation score)

- *Assuming* $\Sigma^{(i)}_{\tilde{x}\tilde{x}}, \Sigma^{(i)}_{\tilde{x}\tilde{y}}$ *are similar in terms of* $\|\cdot\|_F$ *for every* $i \in [n]$ *:*

: Regularization target

: Regularization intensity

1. $\boldsymbol{R_1^{CE}(f)}$ **analysis** : focus on $< \Sigma^{(i)}_{\tilde{x}\tilde{x}}, \left(\nabla f(\tilde{x}_i) - J^{(i)}\right)^T H\left(f(\tilde{x}_i)\right)\left(\nabla f(\tilde{x}_i) - J^{(i)}\right) >_F$

$$| < \Sigma^{(i)}_{\tilde{x}\tilde{x}}, \left(\nabla f(\tilde{x}_i) - J^{(i)}\right)^T H\left(f(\tilde{x}_i)\right)\left(\nabla f(\tilde{x}_i) - J^{(i)}\right) >_F | \leq \left\|\Sigma^{(i)}_{\tilde{x}\tilde{x}}\right\|_F \left\|H\left(f(\tilde{x}_i)\right)\right\|_F \left\|\left(\nabla f(\tilde{x}_i) - J^{(i)}\right)\right\|_F$$

Target : Jacobian of logit→ weighted multivariate OLS / Intensity : Jacobian of Softmax

2. $\boldsymbol{R_2^{CE}(f)}$ **analysis** : focus on $< \Sigma^{(i)}_{\tilde{x}\tilde{x}}, \left(S\left(f(\tilde{x}_i)\right) - \tilde{y}_i\right)^T \nabla^2 f(\tilde{x}_i) >_F$

$$| < \Sigma^{(i)}_{\tilde{x}\tilde{x}}, \left(S\left(f(\tilde{x}_i)\right) - \tilde{y}_i\right)^T \nabla^2 f(\tilde{x}_i) >_F | \leq \left\|\Sigma^{(i)}_{\tilde{x}\tilde{x}}\right\|_F \left\|S\left(f(\tilde{x}_i)\right) - \tilde{y}_i\right\|_2 \left\|\nabla^2 f(\tilde{x}_i)\right\|_F$$

Target : Hessian of logit (tensor) / Intensity : EL2N score

3. $\boldsymbol{R_3^{CE}(f)}$ **analysis** : focus on $< \Sigma^{(i)}_{\tilde{x}\tilde{y}} \left(\Sigma^{(i)}_{\tilde{x}\tilde{x}}\right)^{-1} \Sigma^{(i)}_{\tilde{y}\tilde{x}}, H\left(f(\tilde{x}_i)\right) >_F$

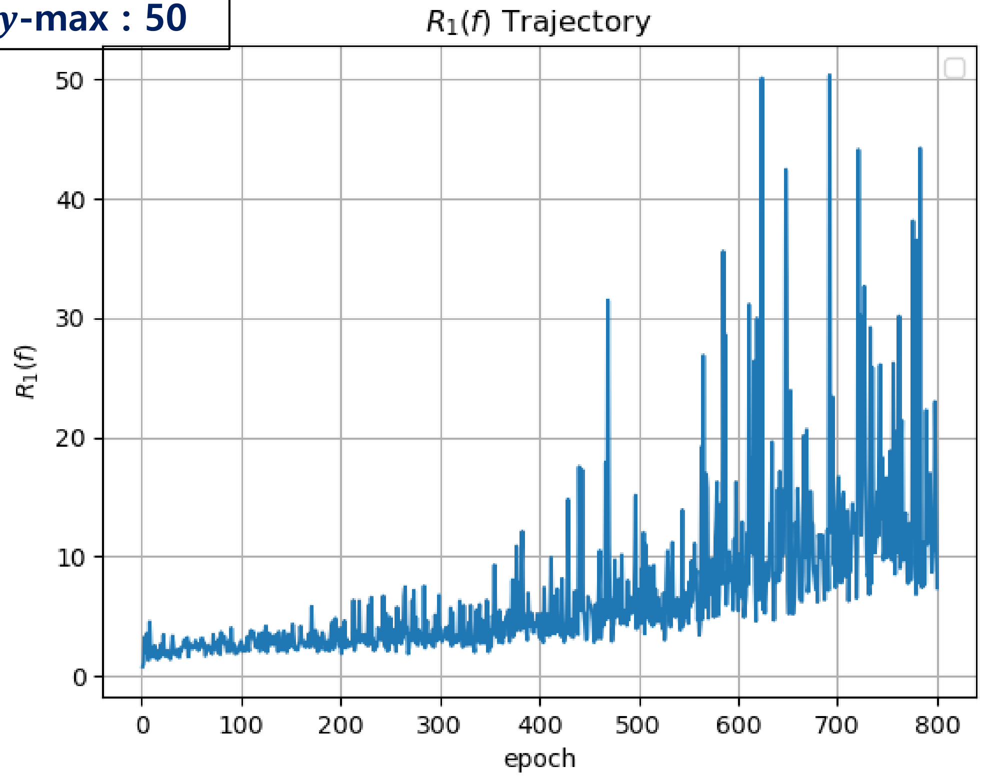
$$| < \Sigma^{(i)}_{\tilde{x}\tilde{y}} \left(\Sigma^{(i)}_{\tilde{x}\tilde{x}}\right)^{-1} \Sigma^{(i)}_{\tilde{y}\tilde{x}}, H\left(f(\tilde{x}_i)\right) >_F | \leq \left\|\Sigma^{(i)}_{\tilde{x}\tilde{y}} \left(\Sigma^{(i)}_{\tilde{x}\tilde{x}}\right)^{-1} \Sigma^{(i)}_{\tilde{y}\tilde{x}}\right\|_F \left\|H\left(f(\tilde{x}_i)\right)\right\|_F$$

Target : Jacobian of Softmax ***(related with Entropy of Softmax)***

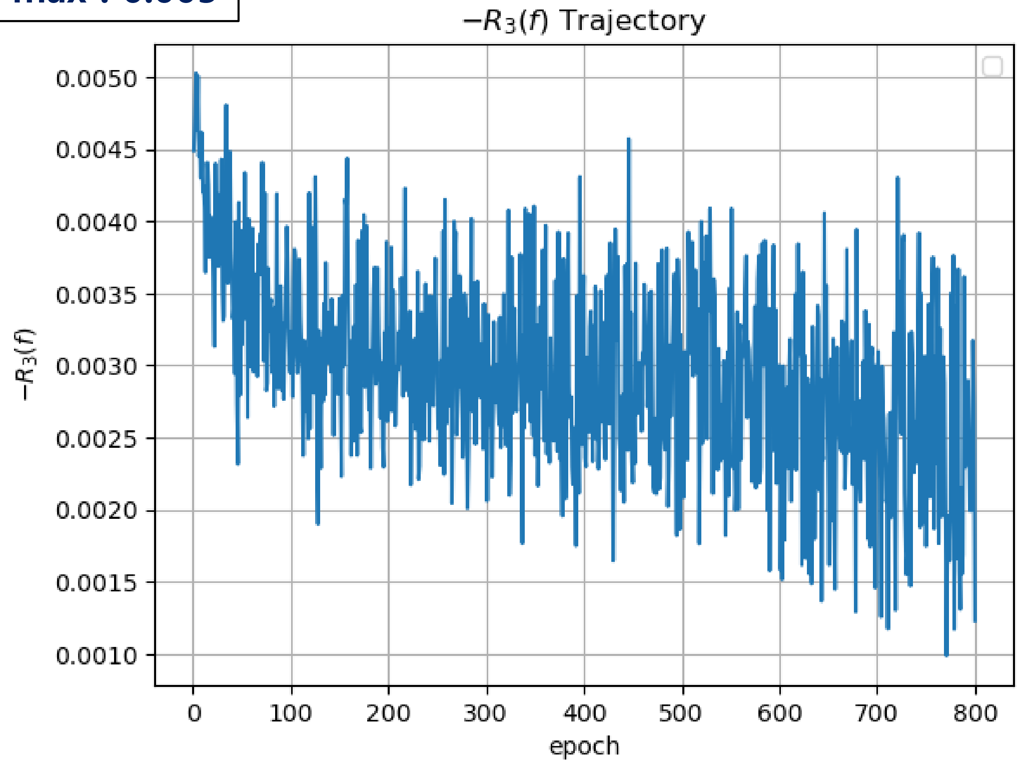
Theoretical parts (Linkage between Mix-up and data valuation score)

- What happen in $R_1^{CE}(f)$, $-R_3^{CE}(f)$ in practice ? (CIFAR-10, ResNet-18, α = 0.5)
- Note : $R_2^{CE}(f)$ is extremely hard to calculate due to tensor hessian (dim = $3072^2 \times 10$)
- Observation : $R_3(f)$ can be negligible compared to $R_1(f)$ which increases as epoch ↑

y-max : 50



y-max : 0.005

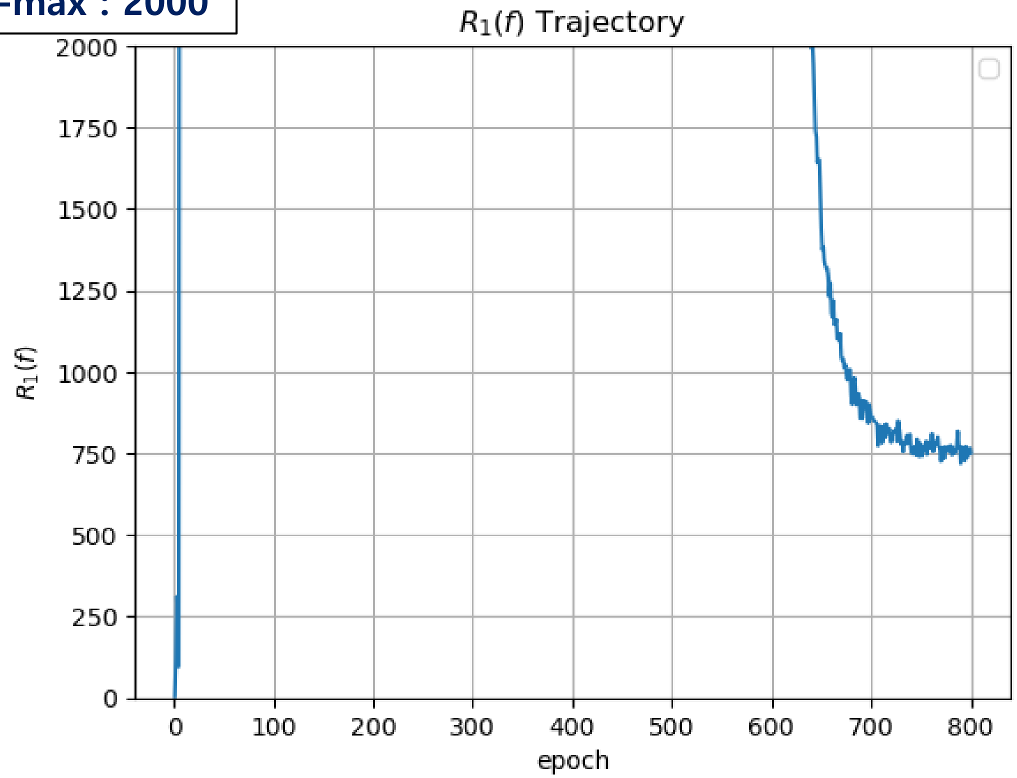


Left : $R_1(f)$ trajectory, Right : $-R_3(f)$ trajectory (values are averaged using randomly selected 100 samples

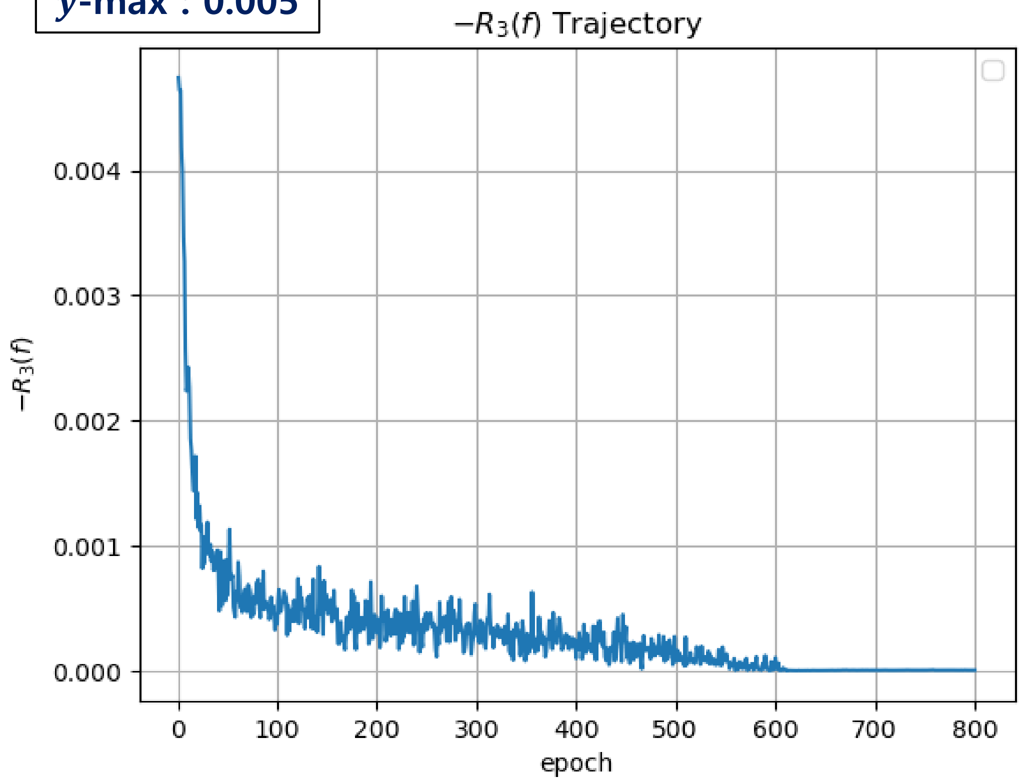
Theoretical parts (Linkage between Mix-up and data valuation score)

- What if we naively train the model (without mix-up)?
  - It cannot regularize the $R_1(f)$ well compared to mix-up. (but better for $-R_3(f)$)
  - This may demonstrates the regularization effect of mix-up (not sure ∵ randomness of samples)

y-max : 2000



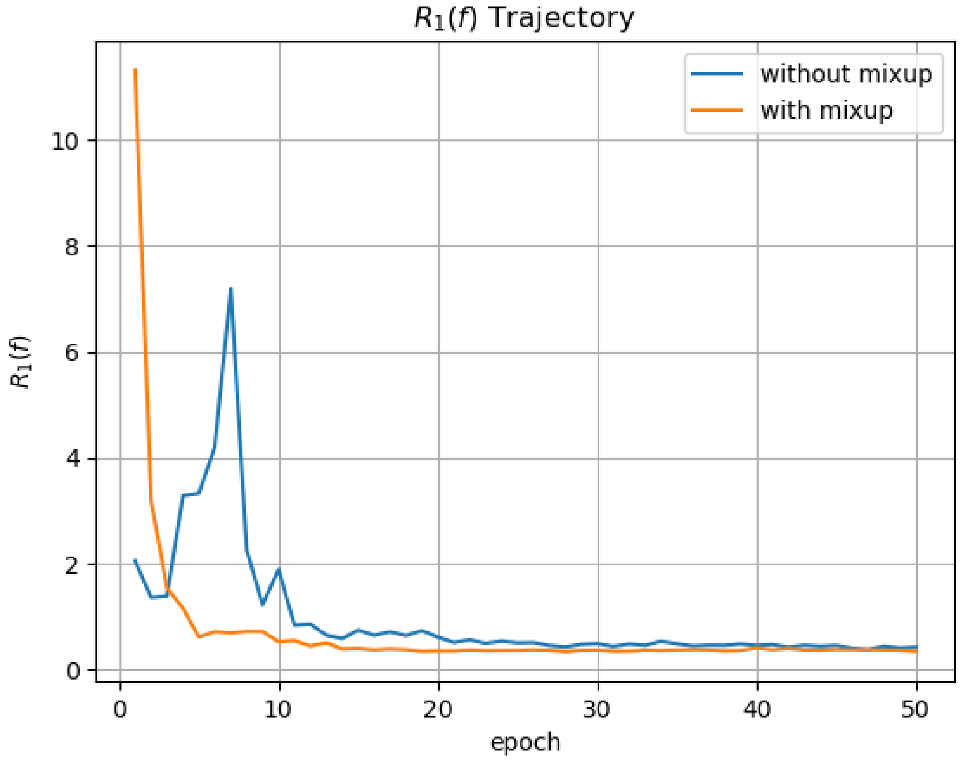
y-max : 0.005

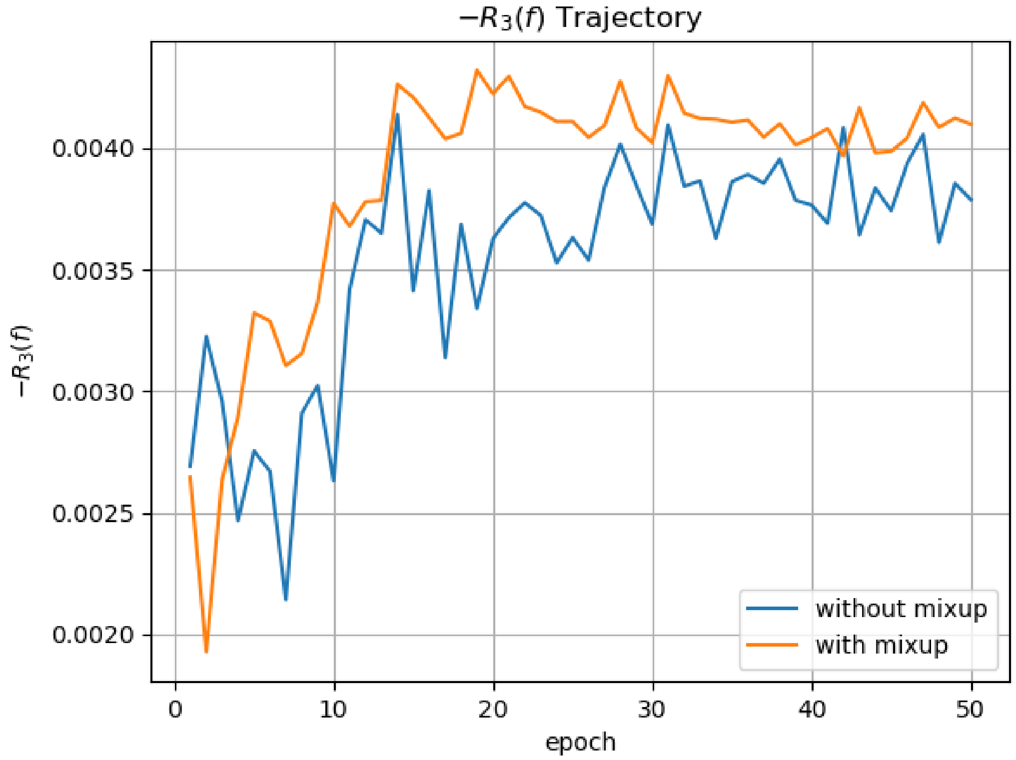


Left : $R_1(f)$ trajectory, Right : $-R_3(f)$ trajectory (values are averaged using randomly selected 100 samples

Theoretical parts (Linkage between Mix-up and data valuation score)

- What if we naively train the model (without mix-up) while imposing same samples?
  - Training without mix-up also ends up regularize $R_1(f)$, but slower than with mix-up.

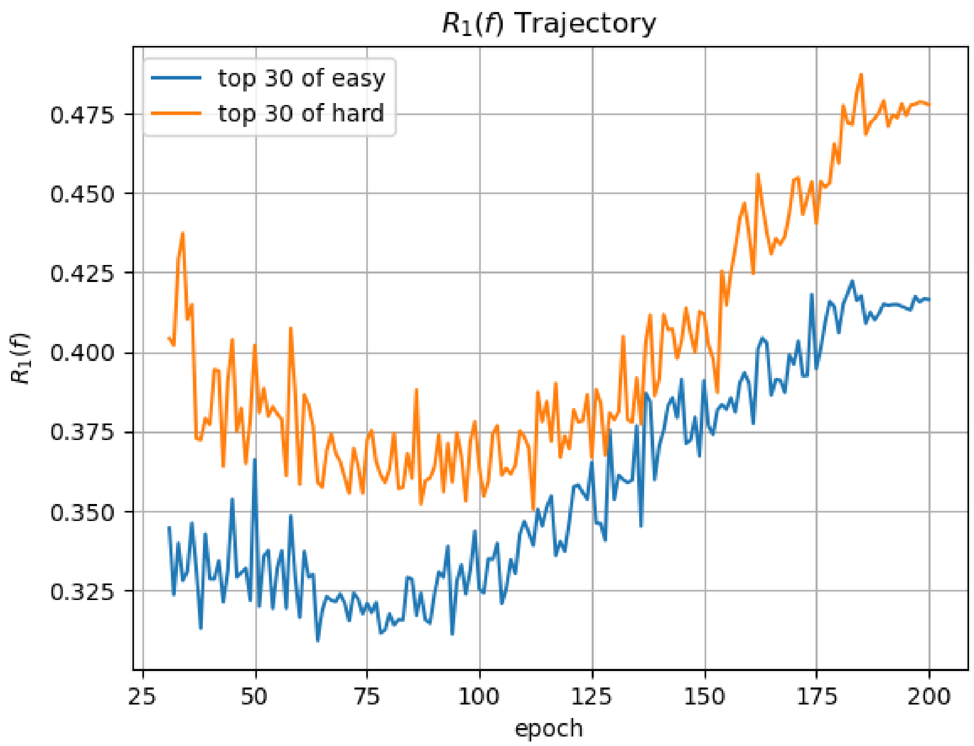


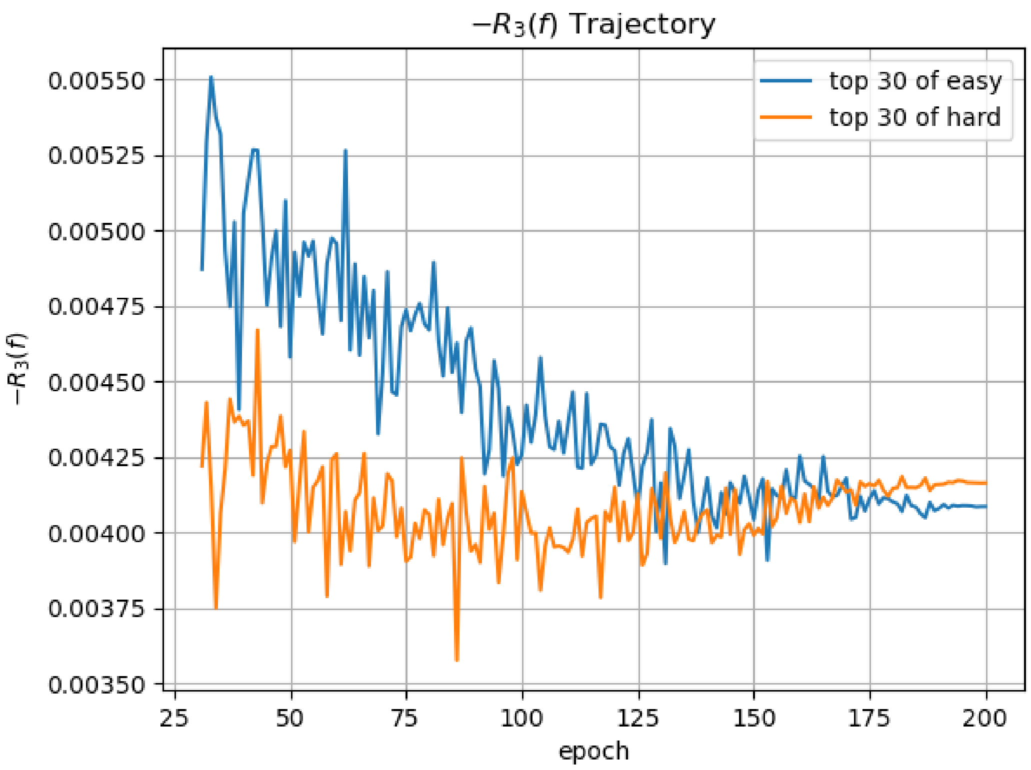


Left : $R_1(f)$ trajectory, Right : $-R_3(f)$ trajectory (values are averaged using randomly selected 30 samples

Theoretical parts (Linkage between Mix-up and data valuation score)

- Then, $R_1(f)$ is becomes larger when data x_i becomes harder?
  - Hard examples x_i gets larger $R_1(f)$ values compared to easy examples x_j.
  - **This implies mix-up loss can affected slightly larger by hard samples rather easy one.**
    - Hence, it can be beneficial to train mix-up samples generated by hard samples.





Theoretical parts (Linkage between Mix-up and data valuation score)

- Summary of observations :

1. Mix-up regularization is related with **regularization of the Jacobian of logit / Hessian of logit** whose **intensity varies on data valuation score of** $\boldsymbol{\widetilde{x}_i}$ (expected mix-up point of x_i)

2. Regularization may not indicate 'reducing' the values instead **'attenuating' the increase.**

Plausible Conclusion :

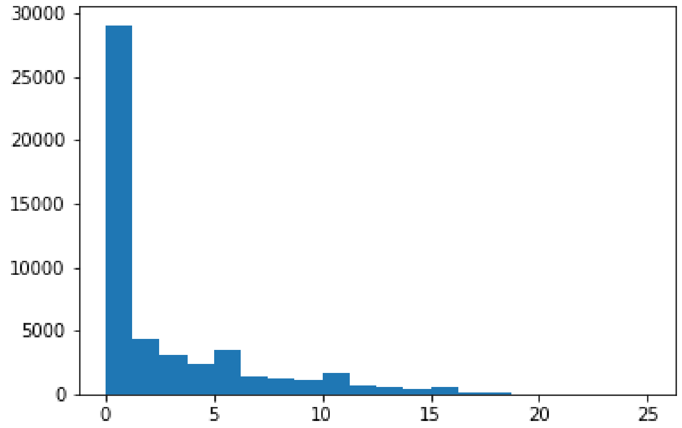
∴ **Data valuation score of $\boldsymbol{x_i}$ may contributes to the regularization effect of mix-up.**

Observation parts (Presence of frequently incorrect directions by class)

- Plausible surmise : There is a tendency that Images in a particular class are usually incorrect in some other certain class.

- For example (CIFAR-10) : Automobile will be confused with trucks.
  Cats will be confused with dogs.

- One idea : How about mixing up between samples and their corresponding samples which are in 'certain' classes that former samples usually get wrong (based on forgetting score)



Histogram of the number of forgetting events
(x-axis : # of forgetting events, y-axis : Frequency)

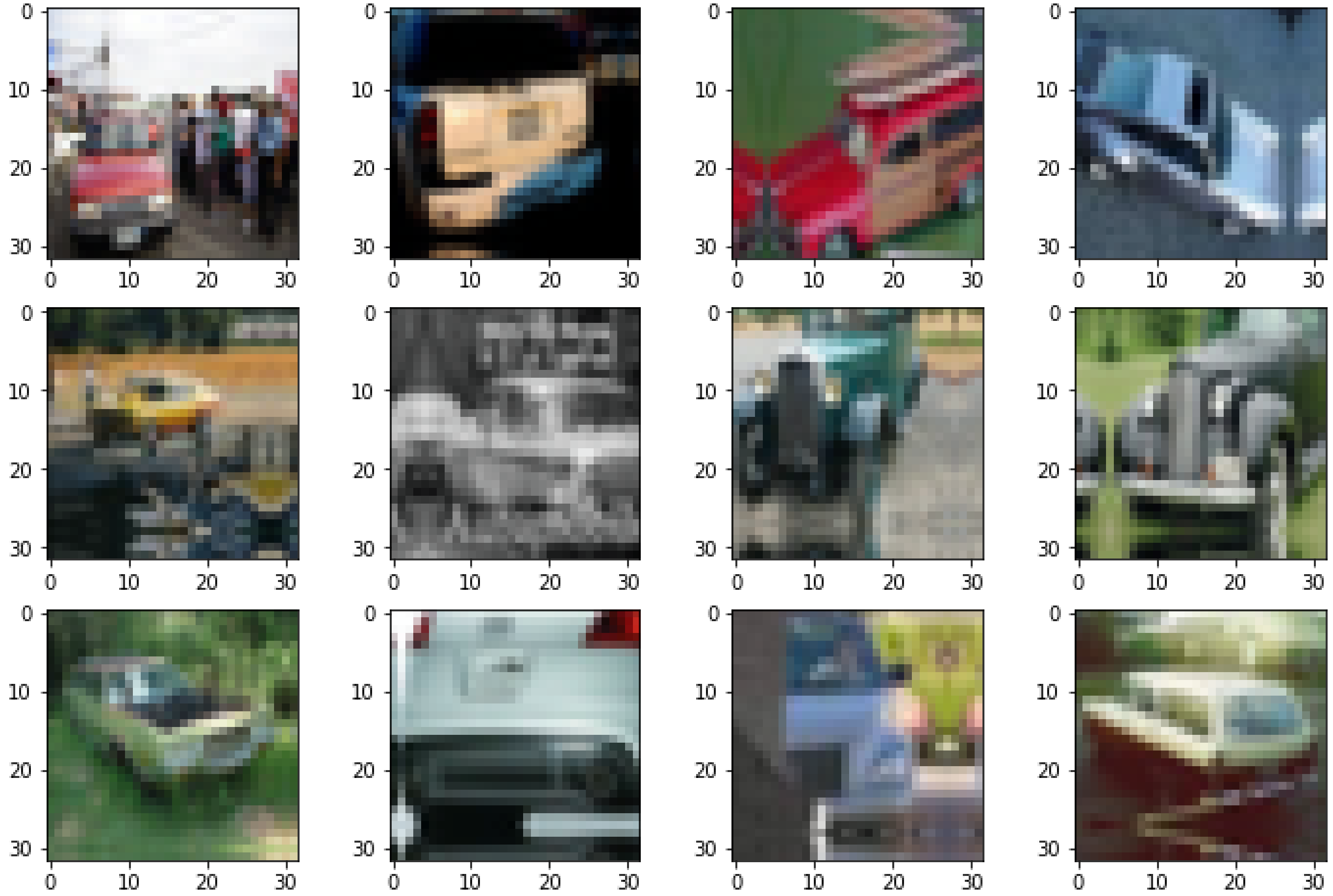
Experiment environment :
CIFAR-10 / ResNet-18 / AdamW with CosineAnnealing / 200 epoch

Observation parts (Presence of frequently incorrect directions by class)

Criterion for selection : top 10 of # of forgetting events on each class

automobile (class index = 1)

classification : [airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck]



truck (class index = 9)

```
(1,1)th stat : [0, 0, 0, 0, 0, 0, 0, 0, 1, 18]
(1,2)th stat : [1, 0, 0, 1, 0, 0, 0, 0, 2, 15]
(1,3)th stat : [0, 0, 0, 0, 0, 0, 1, 0, 0, 17]
(1,4)th stat : [13, 0, 0, 0, 0, 0, 0, 0, 3, 1]
(2,1)th stat : [8, 0, 0, 0, 0, 0, 4, 0, 0, 5]
(2,2)th stat : [3, 0, 0, 1, 1, 0, 0, 0, 9, 3]
(2,3)th stat : [0, 0, 0, 0, 0, 0, 1, 0, 0, 15]
(2,4)th stat : [0, 0, 0, 0, 0, 0, 0, 1, 0, 14]
(3,1)th stat : [0, 0, 0, 0, 0, 0, 0, 0, 15, 0]
(3,2)th stat : [6, 0, 0, 1, 0, 0, 0, 0, 1, 7]
(3,3)th stat : [0, 0, 2, 0, 1, 3, 0, 0, 1, 8]
(3,4)th stat : [2, 0, 0, 0, 0, 0, 0, 0, 12, 1]
```

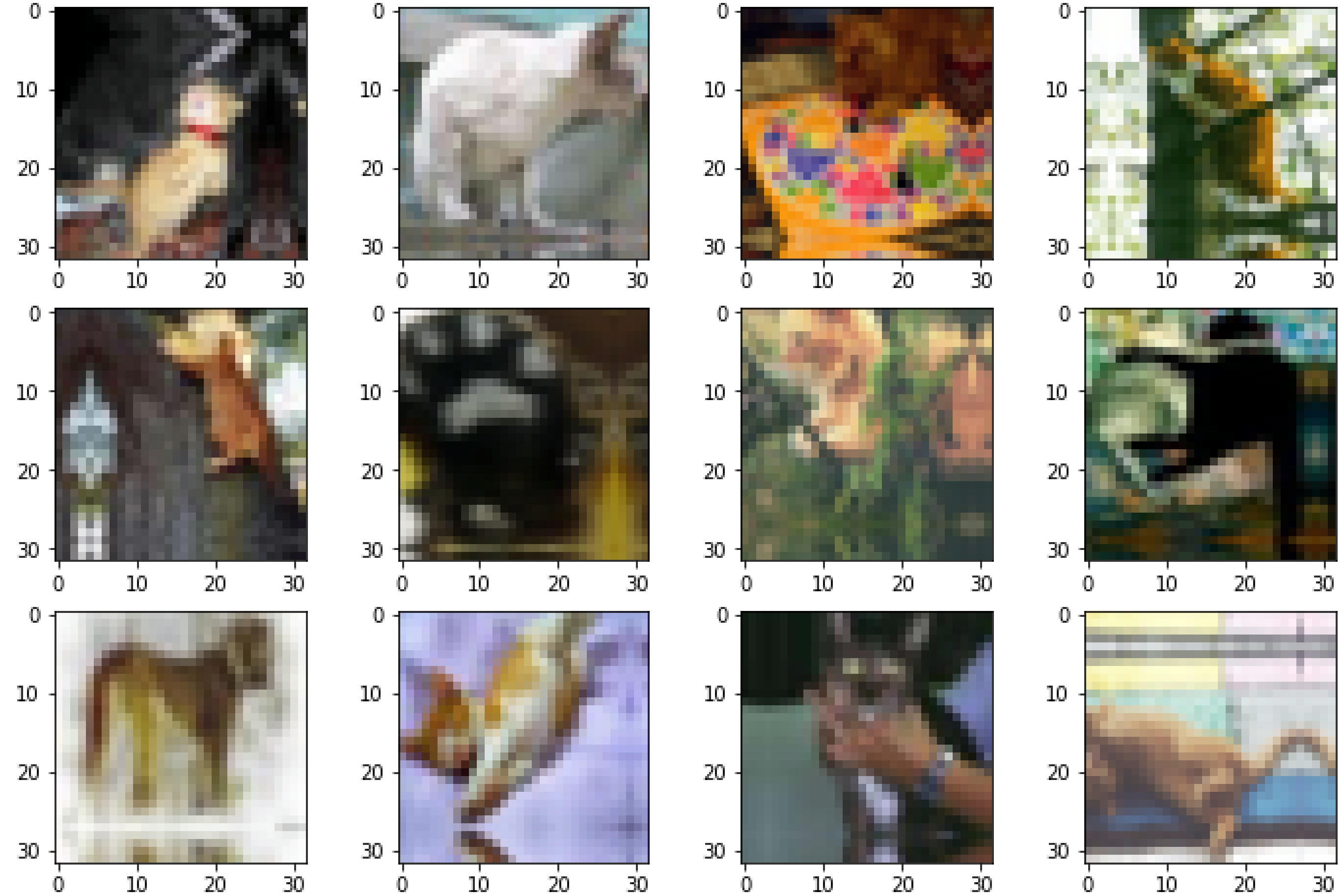
Forgetting statistics
(represent # of events w.r.t each classes)

Observation parts (Presence of frequently incorrect directions by class)

Criterion for selection : top 10 of # of forgetting events on each class

cat (class index= 3)

classification : [airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck]

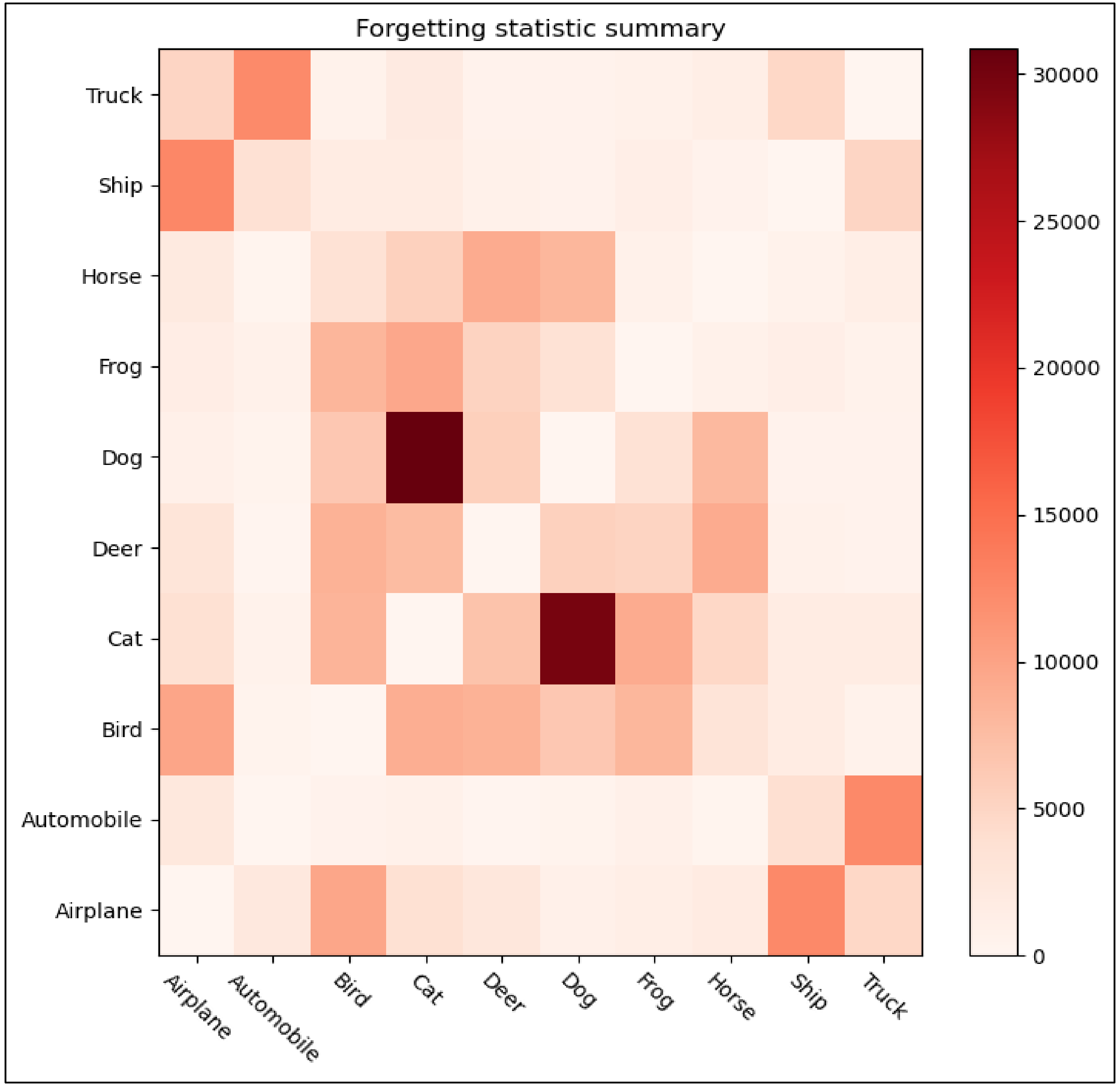


dog (class index= 5)

```
(1,1)th stat : [0, 0, 8, 0, 3, 5, 5, 1, 0, 0]
(1,2)th stat : [0, 0, 4, 0, 0, 14, 0, 4, 0, 0]
(1,3)th stat : [0, 4, 1, 0, 1, 9, 1, 1, 5, 0]
(1,4)th stat : [0, 0, 3, 0, 3, 0, 14, 0, 0, 1]
(2,1)th stat : [0, 0, 1, 0, 13, 0, 7, 0, 0, 0]
(2,2)th stat : [0, 1, 3, 0, 2, 12, 0, 3, 0, 0]
(2,3)th stat : [0, 0, 3, 0, 0, 2, 16, 0, 0, 0]
(2,4)th stat : [0, 1, 8, 0, 4, 0, 6, 1, 0, 0]
(3,1)th stat : [0, 0, 0, 0, 2, 18, 0, 0, 0, 0]
(3,2)th stat : [13, 0, 2, 0, 0, 4, 0, 0, 0, 0]
(3,3)th stat : [2, 0, 7, 0, 2, 8, 0, 0, 0, 0]
(3,4)th stat : [0, 0, 9, 0, 5, 2, 0, 3, 0, 0]
```

Forgetting statistics
(represent # of events w.r.t each classes)

Observation parts (Presence of frequently incorrect directions by class)



Note (Examples to read) :
[Reading direction : y-axis → x-axis]

1. **Truck tends to be confusing with Automobile (2,10)**
2. **Dog tends to be confusing with Cat (4,6)**

Property :

1. **It has symmetric shape. (seems to be obvious)**
2. **There is a tendency that Images in a particular class are usually incorrect in some other certain class.**

Forgetting statistics summary
(represent # of events w.r.t each classes)

Observation parts (Presence of frequently incorrect directions by class)

- Forgetting score based Mix-up
  - Basic idea : Exploit the tendency to be misclassified as a specific classes for each training classes.

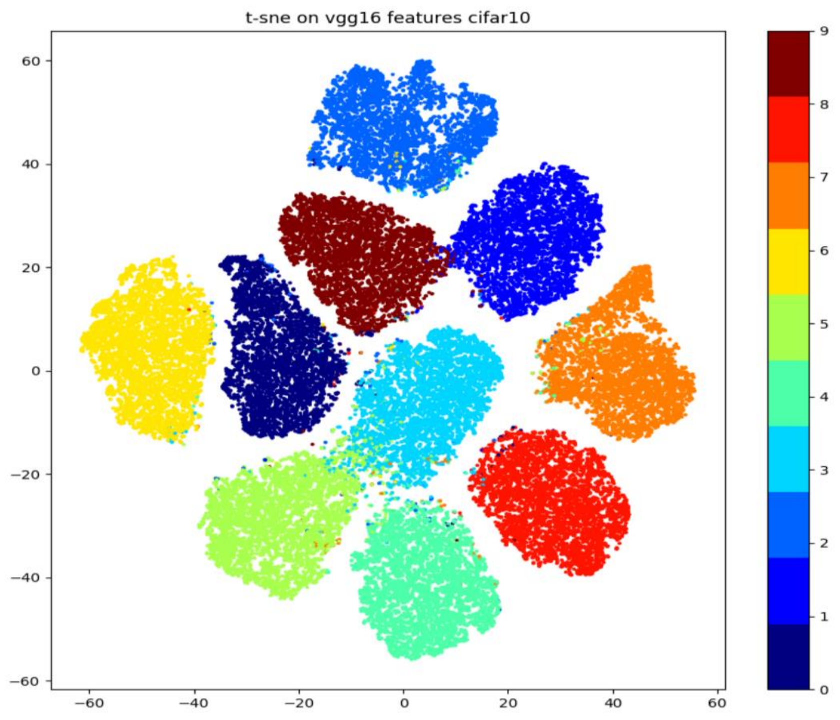
  - Overall (brief) strategy : (motivation will be discussed in later slides)

  : For each batch samples, select corresponding hard samples based on forgetting statistics. [**Hard mix-up pairing**]

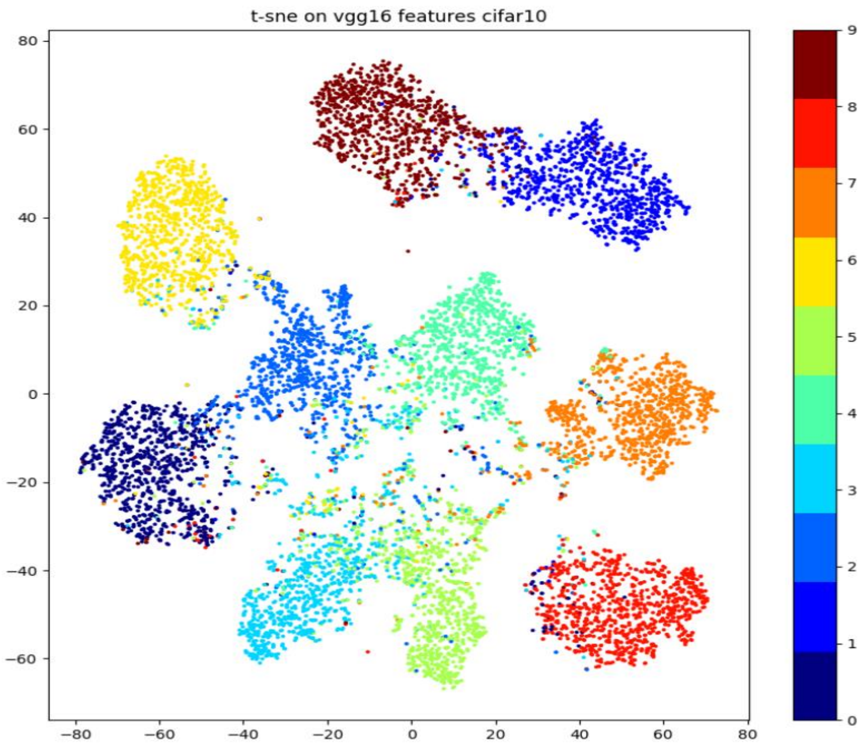
  Expected effect: Avoid manifold intrusion and make smooth decision boundary only around decision boundary region.

Observation parts (Presence of frequently incorrect directions by class)

- Hard mix-up pairing :
  - Problem : Random mix-up pairing is highly likely to induce manifold intrusion. (even in the mix-up on the penultimate layer)



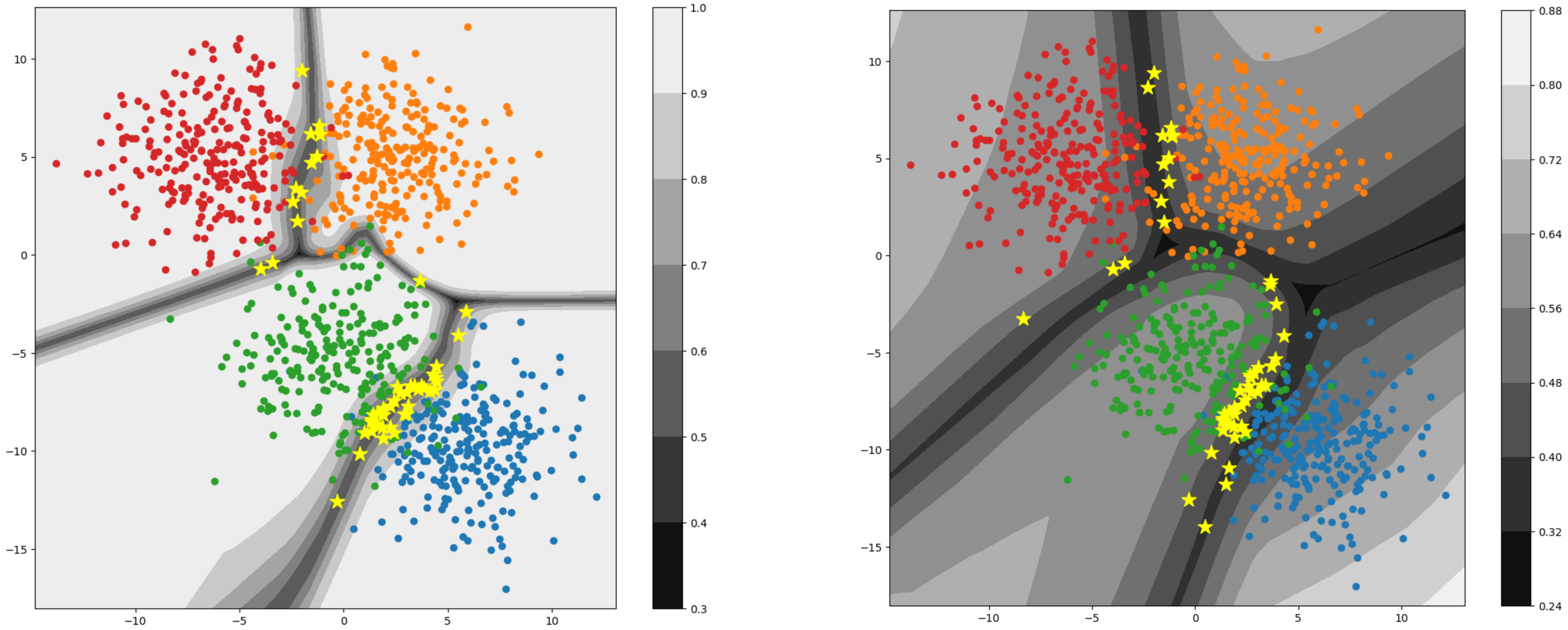
t-SNE of penultimate layer of VGG16 on CIFAR-10 train set



t-SNE of penultimate layer of VGG16 on CIFAR-10 test set

Observation parts (Presence of frequently incorrect directions by class)

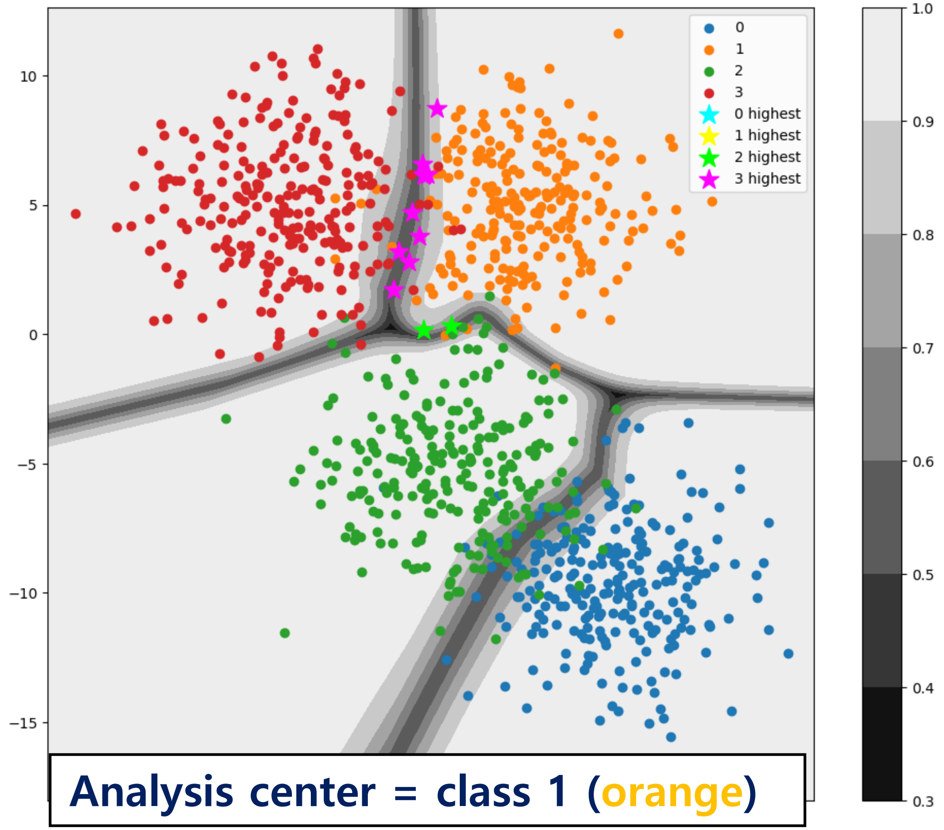
- Hard mix-up pairing :
  - Observation : forgetting score is effective on detecting hard samples. (samples located near decision boundary)

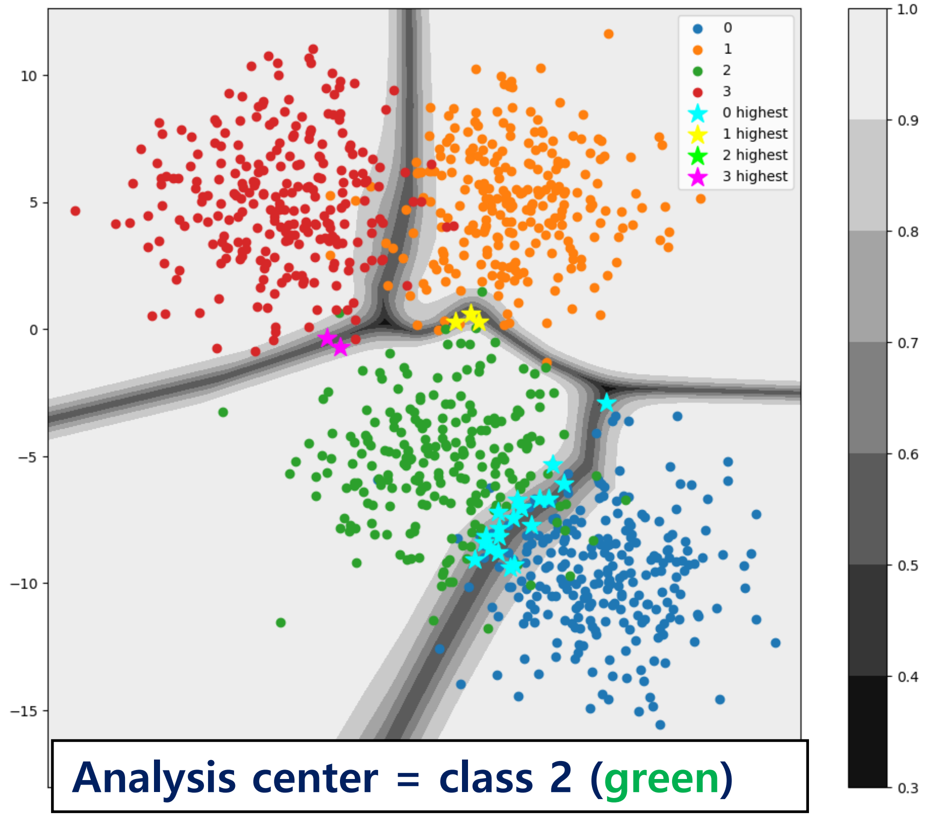


Decision boundary of simple NN on blob dataset (★ : top-50 hard samples) [Left : naïve training / Right : mix-up training, $\alpha = 1$]

Observation parts (Presence of frequently incorrect directions by class)

- Hard mix-up pairing :
  - Hard samples tend to be outside the data cluster toward the other class direction.
    (not only low-dimensional dataset, but also high-dimensional dataset such as CIFAR-10)





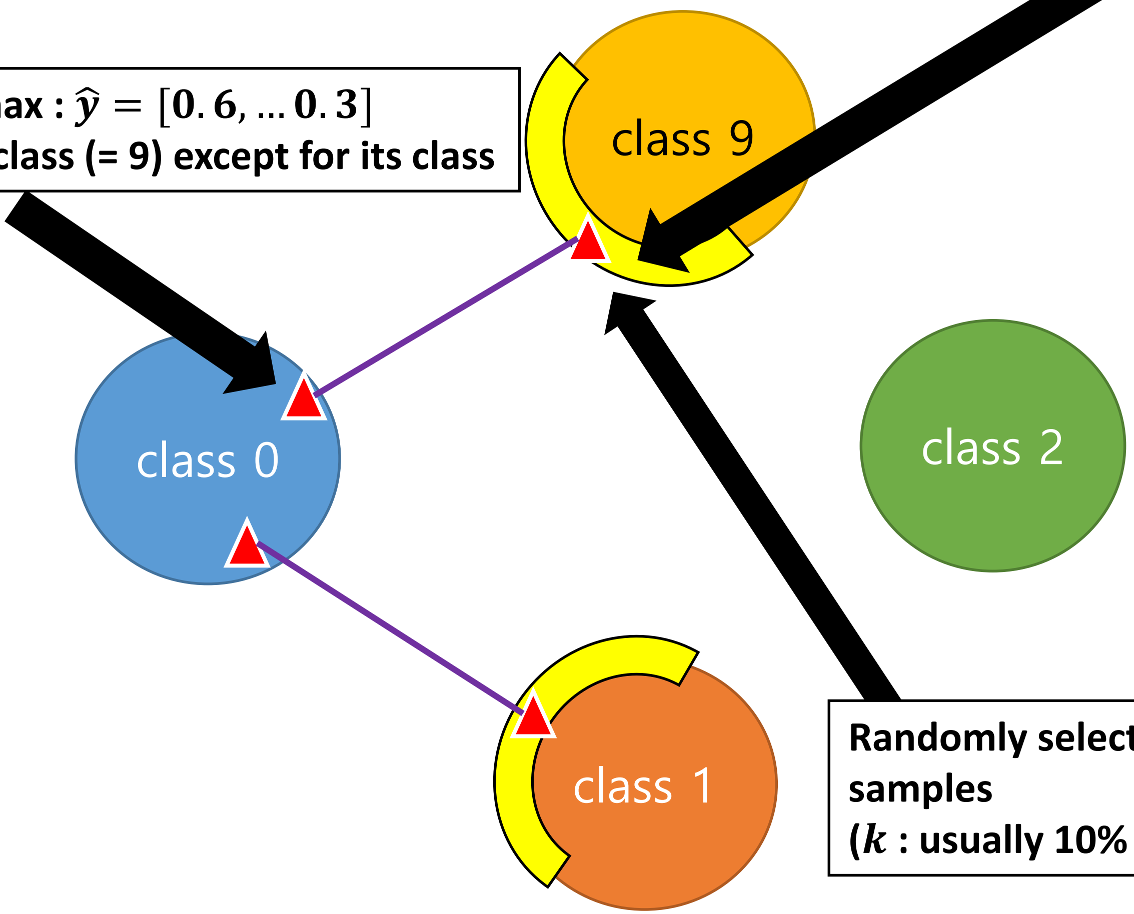
Top-20 hard (≥3) examples (for each class) with respect to highest error rate towards each nearby classes [Left : class 1 / Right : class 2]

Observation parts (Presence of frequently incorrect directions by class)

- Hard mix-up pairing :

Choose top k hard samples whose highest forgetting statistic belongs to class 0

previous softmax : $\hat{y} = [0.6, \ldots 0.3]$
→ pick top1 softmax class (= 9) except for its class



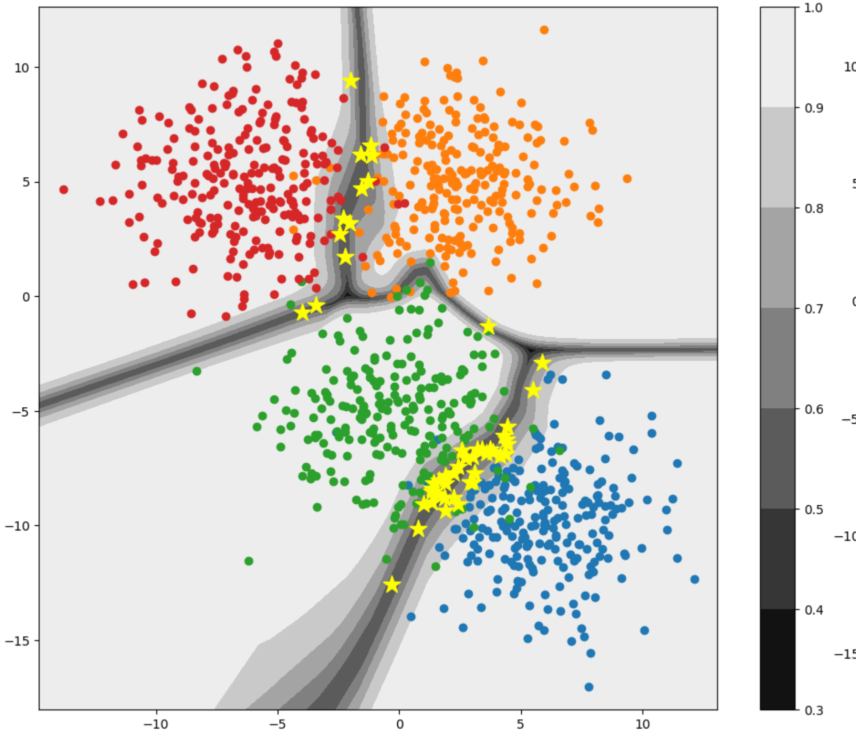
Randomly select hard sample in class 9 among top k hard samples
(k : usually 10% of # of class samples by trial and error)

Hard mix-up pairing using forgetting statistics (Here, class 9 sample is chosen)

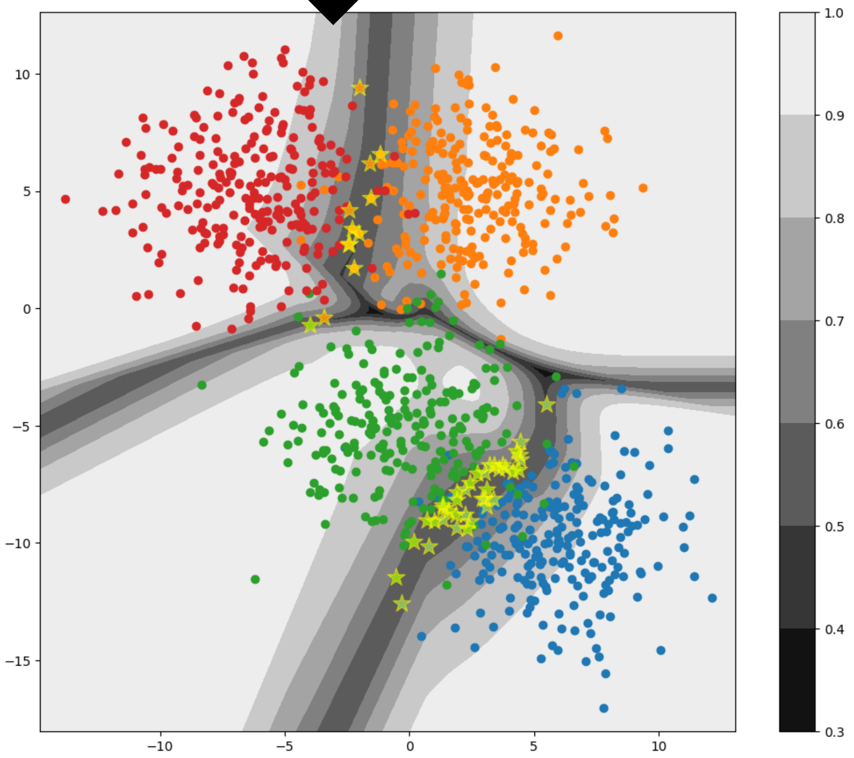
Observation parts (Presence of frequently incorrect directions by class)

- Low-dimensional experiment (dataset : 2-dimensional blob dataset, model = simple NN)
  : Visualization of decision boundary

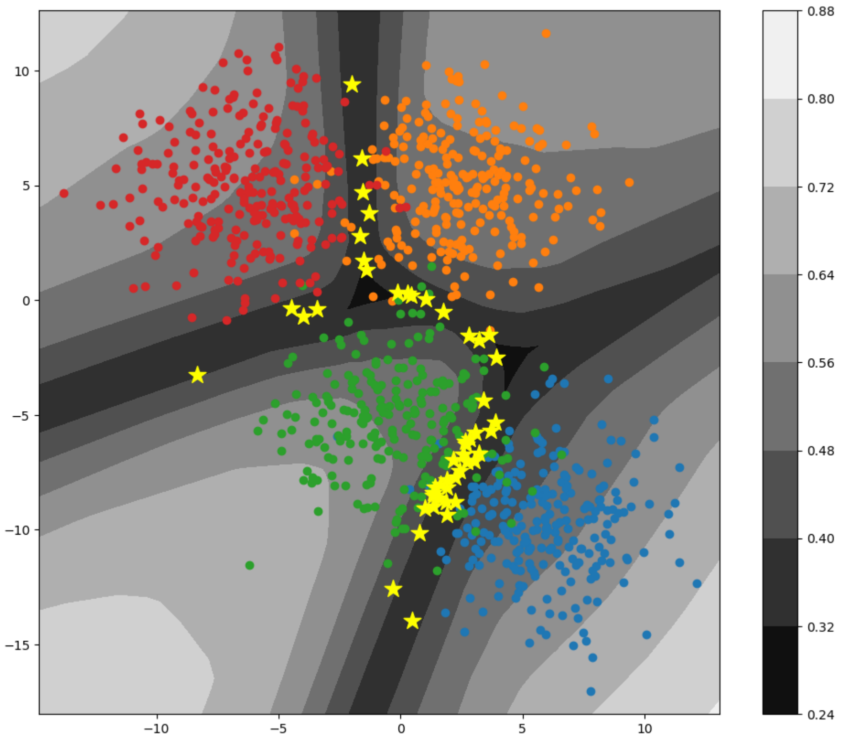
Demonstration of hard mix-up pairing :
Mix-up happened only near decision boundary



Original training



Hard mix-up pairing training ($\alpha = 0.2$)



mix-up training ($\alpha = 0.2$)

Note : On CIFAR-10, It showed poor performance : 95.03% (penultimate hard mix-up pairing training, α = 0.3, 600 epoch)
(∵ It turns out that their mix-up partner becomes very restricted during training → reduce sample diversity on mix-up)

Observation parts (Data valuation of mix-up samples)

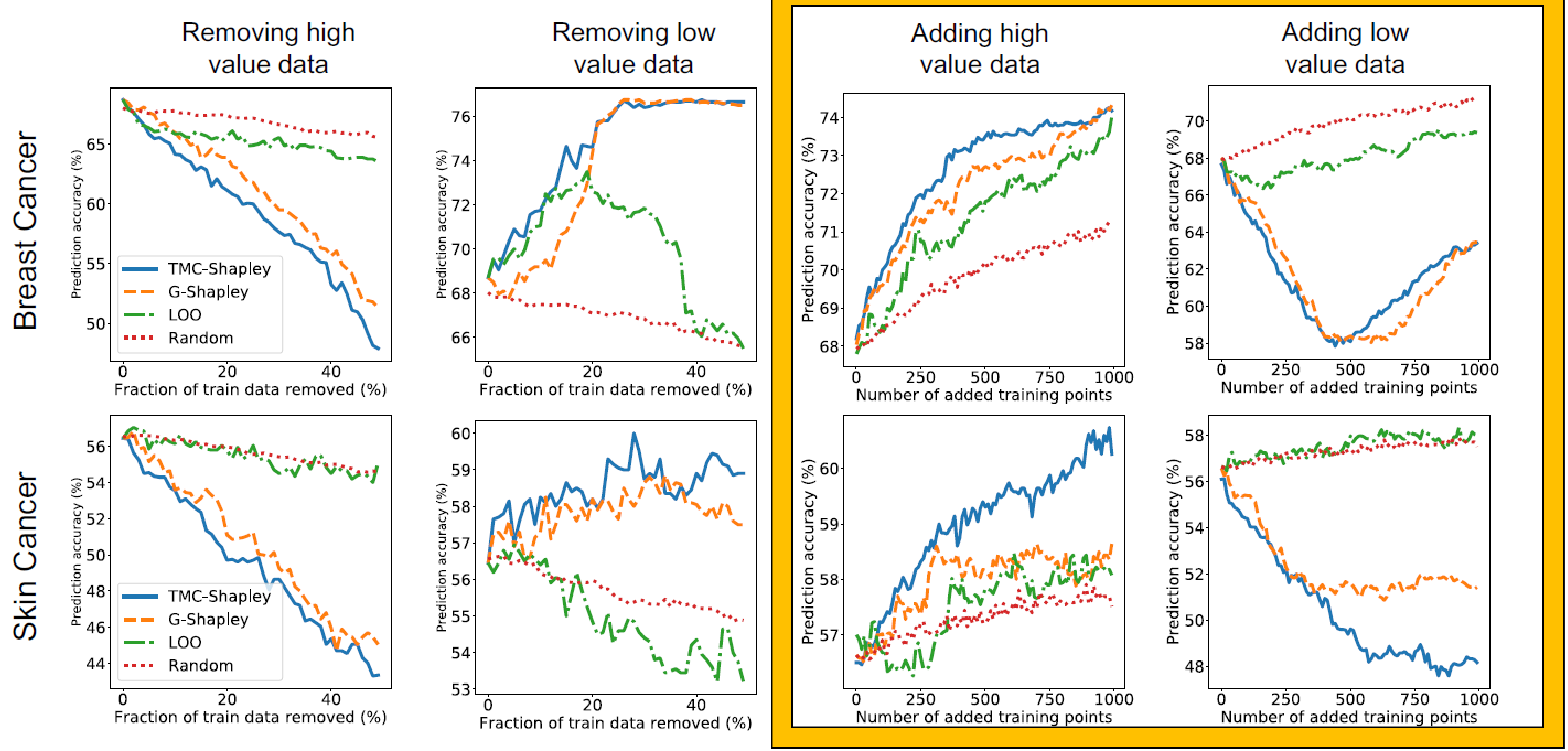
- Then, how about limiting the mix-up partner pool into top η % of overall hard samples ?

- According to [M.paul et al., 2021], it turned out that learning via hard examples (based on EL2N, GraNd score) usually have high **'training error barrier'** and shows high value of **'NTK velocity'**. (similar logic holds also for easy samples)

- Further, if EL2N / GraNd score correlate well with Shapley value [Z. Jiang et al., 2020], **It is beneficial for model to improve test accuracy by learning hard samples.**

- Experiment environment : CIFAR-10 / ResNet-18
  - Original data ranking score: EL2N
  - Mix-up data ranking score : EL2N / GraNd
  - Learned training data = $50{,}000 \times 6\ (\text{original}) + 500 \times 10^2 \times 6\ (\text{mixup}) = 300{,}000$ samples (To mimic following experiment environment, we adopt RegMixUp setting)

Observation parts (Data valuation of mix-up samples)

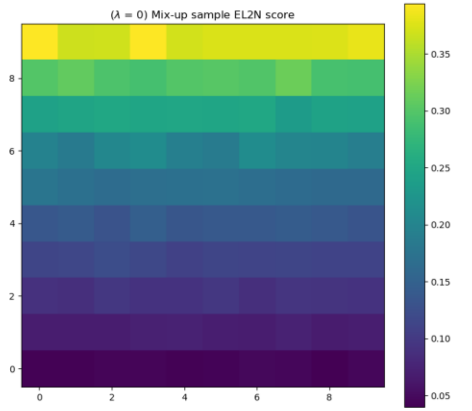


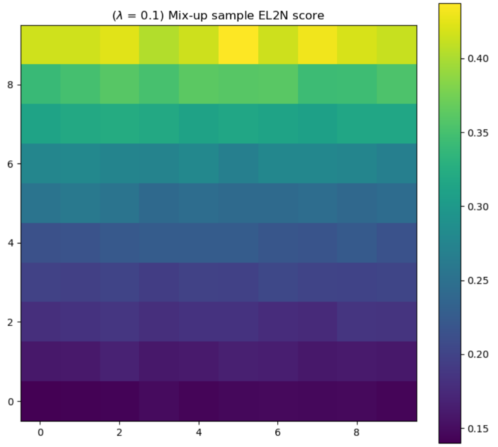
Adding hard examples on each data set can be more beneficial than adding easy value data (based on Shapley value)

Observation parts (Data valuation of mix-up samples)

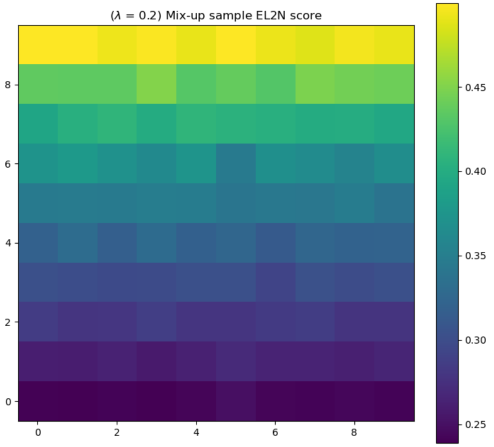
- When Mix-up samples' score are analyzed by **EL2N score.**
- EL2N score computation epoch : 20 / model average # = 10

$\lambda = 0$

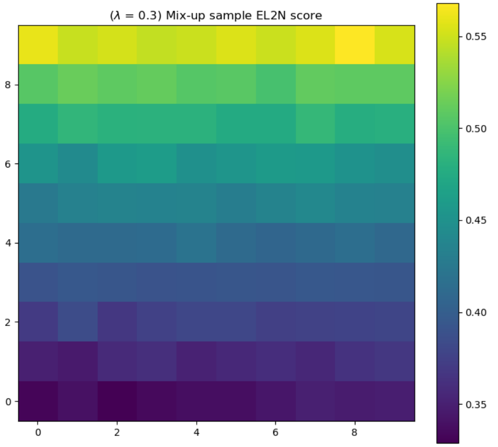




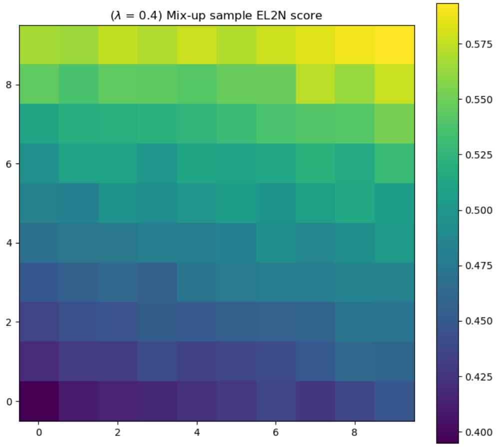
$\lambda = 0.1$



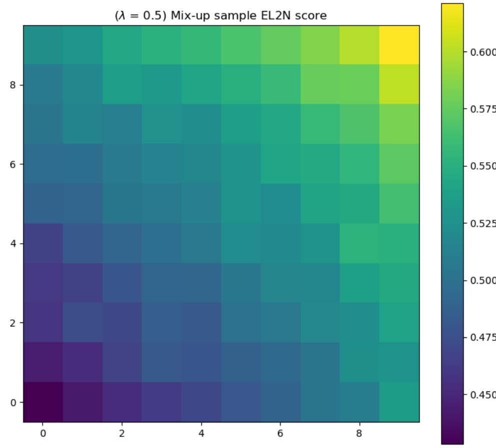
$\lambda = 0.2$



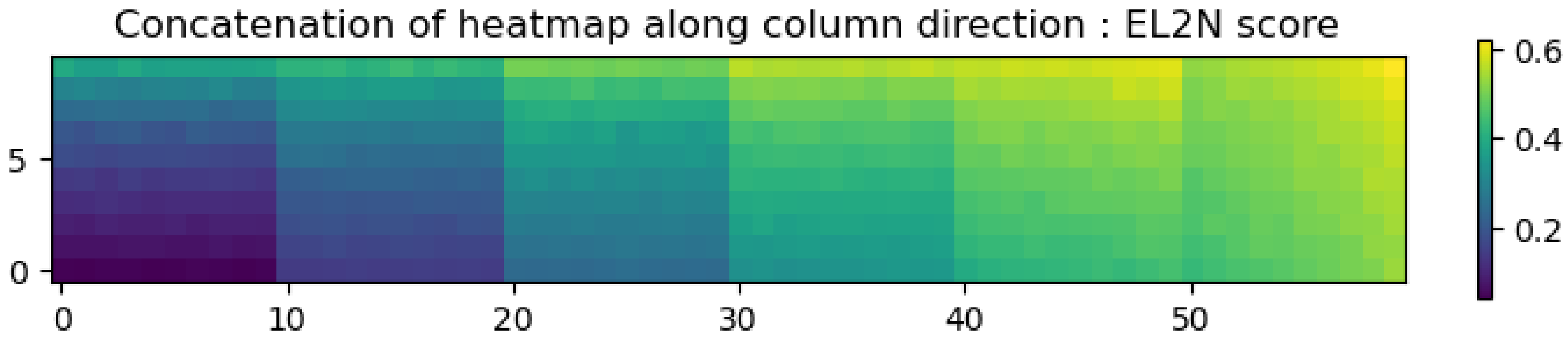
$\lambda = 0.3$



$\lambda = 0.4$



$\lambda = 0.5$



Combined Heatmap along column direction

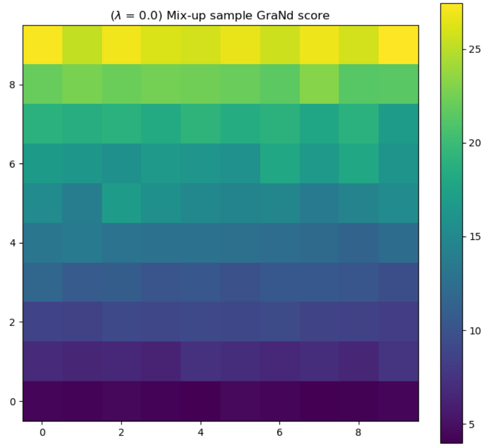
Note :

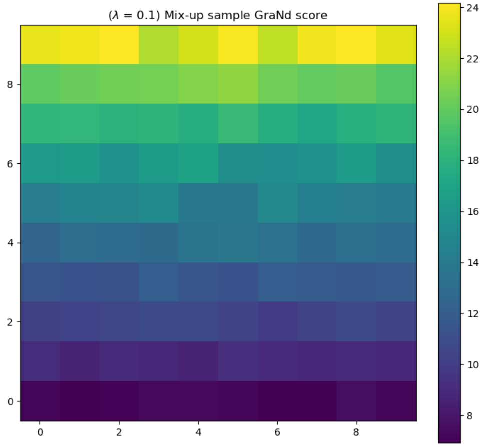
1. x-axis : top $10 \times x$ (%)~$10 \times (x+1)$% easiest sample where λ is applied
2. y-axis : top $10 \times x$ (%)~$10 \times (x+1)$% easiest sample where $1 - \lambda$ is applied

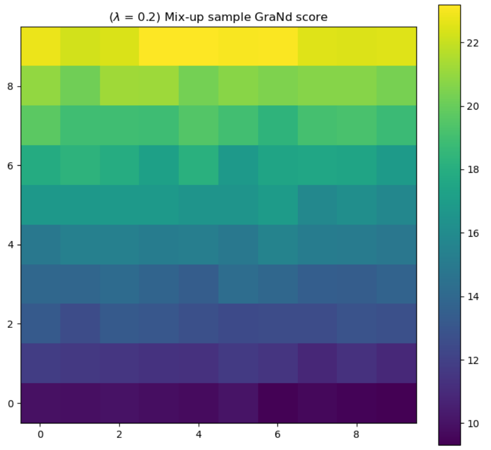
Observation parts (Data valuation of mix-up samples)

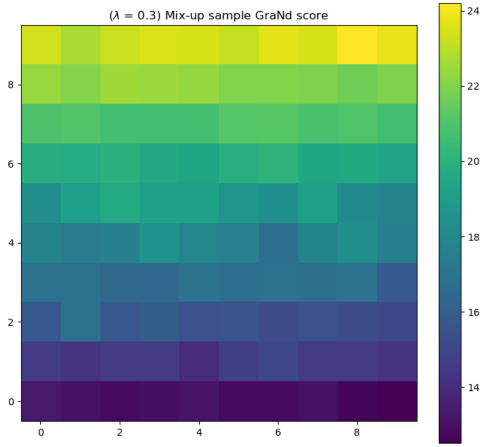
- When Mix-up samples' score are analyzed by **GraNd score.**
- GraNd score computation epoch : 5 / model average # = 10

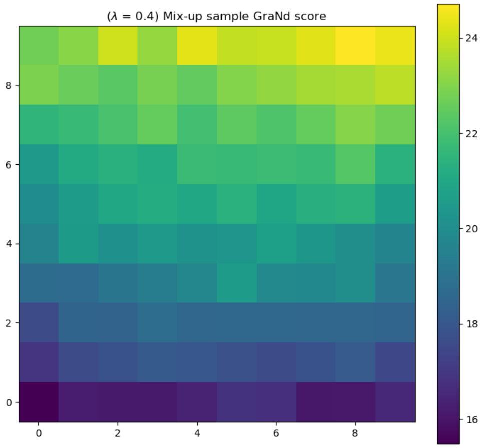
$\lambda = 0$

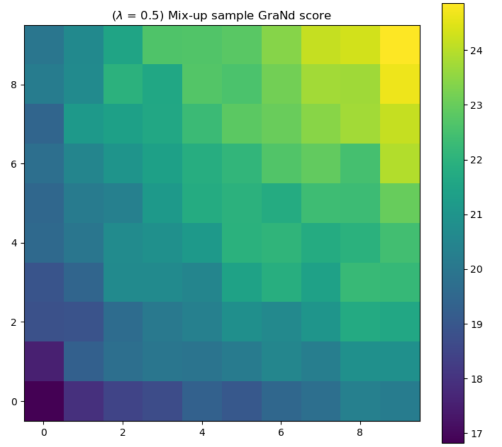




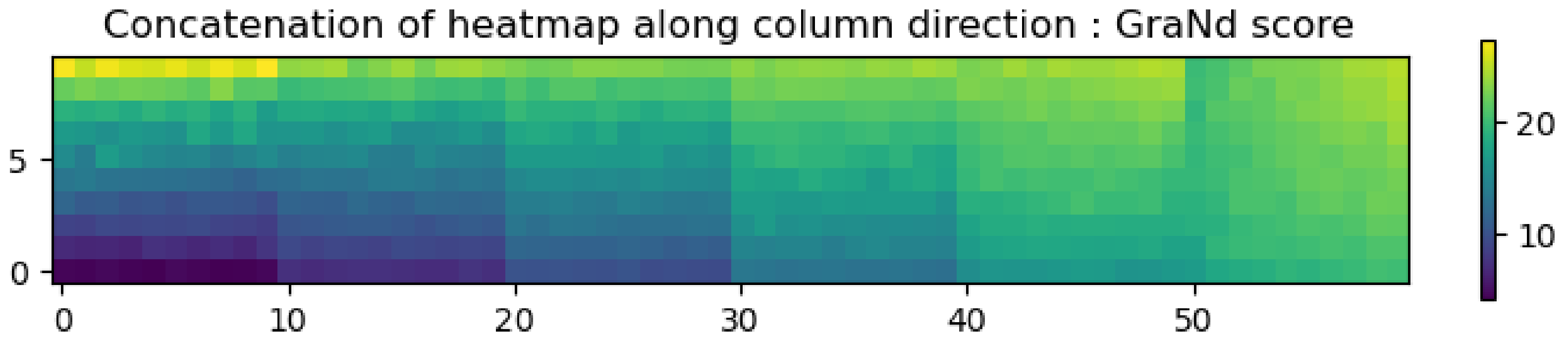








$\lambda = 0.1$ $\lambda = 0.2$ $\lambda = 0.3$ $\lambda = 0.4$ $\lambda = 0.5$



Combined Heatmap along column direction

Note :

1. x-axis : top $10 \times x$ (%)$\sim 10 \times (x+1)$% easiest sample where λ is applied
2. y-axis : top $10 \times x$ (%)$\sim 10 \times (x+1)$% easiest sample where $1-\lambda$ is applied

Observation parts (Data valuation of mix-up samples)

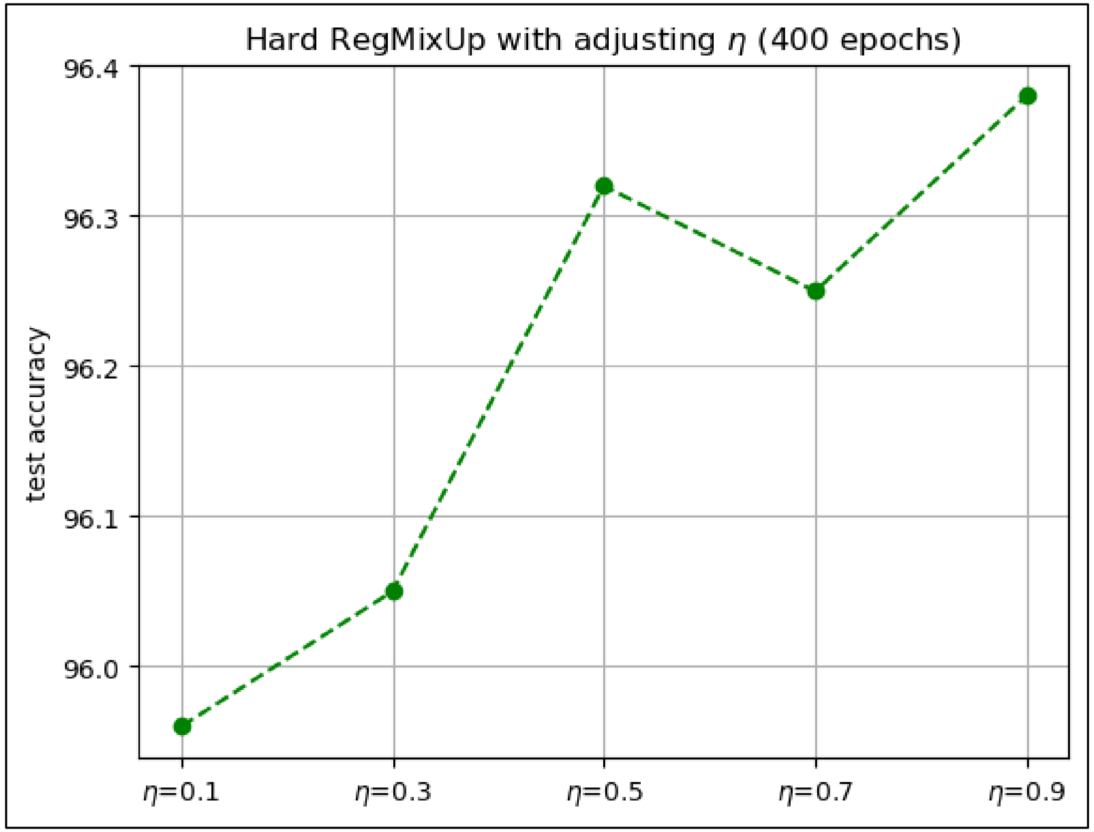
- Our intuition [Hard sample $\times$ Hard sample mix-up would result in Hard mix-up sample] seems to be clear.

- Since EL2N score is derived from GraNd score with logit gradient orthogonality assumption, **It may not represent good data valuation scores for 'mix-up samples'**

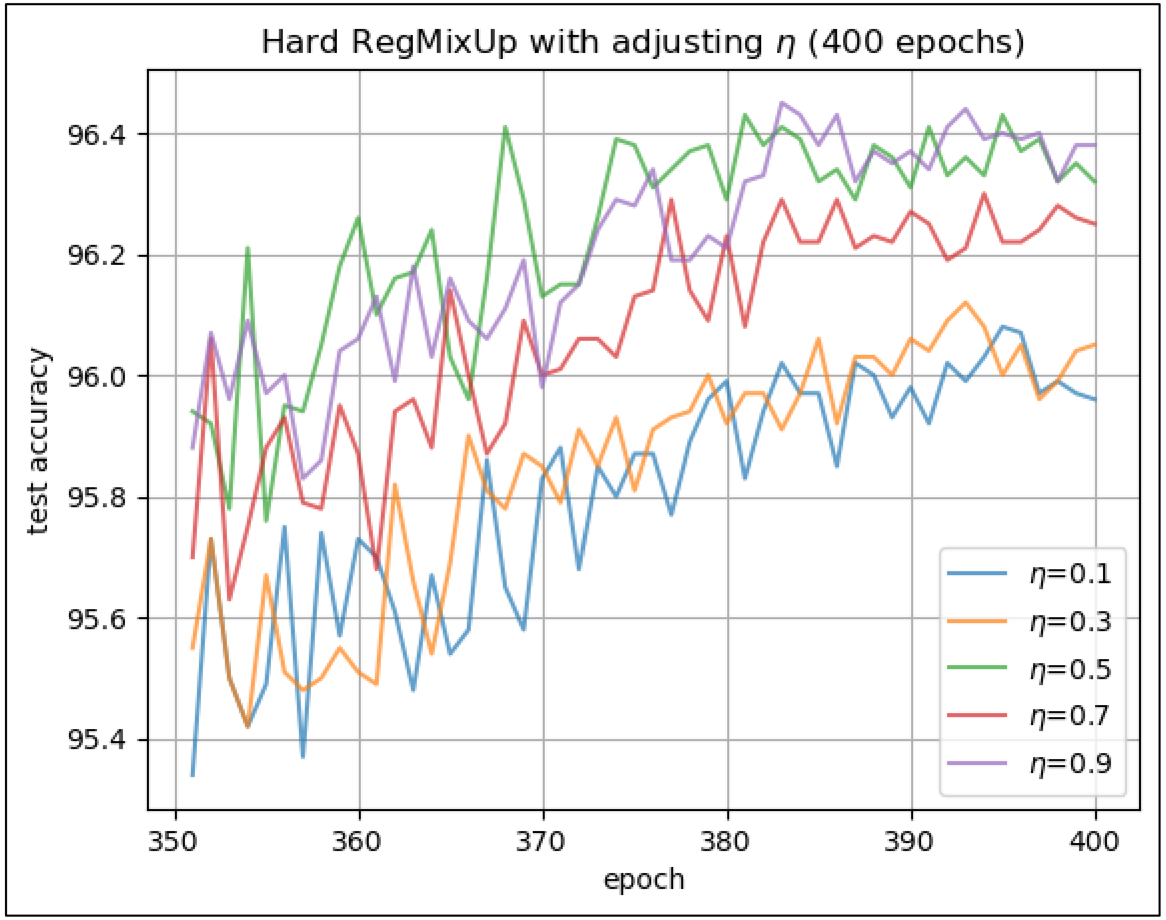
- Note that we can use only limited number of mix-up samples during mix-up training.
  - Hence, It may be better to use hard mix-up samples rather easy mix-up samples.
  - Again, how about limiting the mix-up partner pool to top $100 \times \eta$ % of overall hard samples ?

Experiments parts (Hard mix-up with adjusting hard data portion)

- In RegMixUp, limiting the pool of mix up samples to top $100 \times \eta$ % of overall hard samples will degrade the performance as η gets lower.



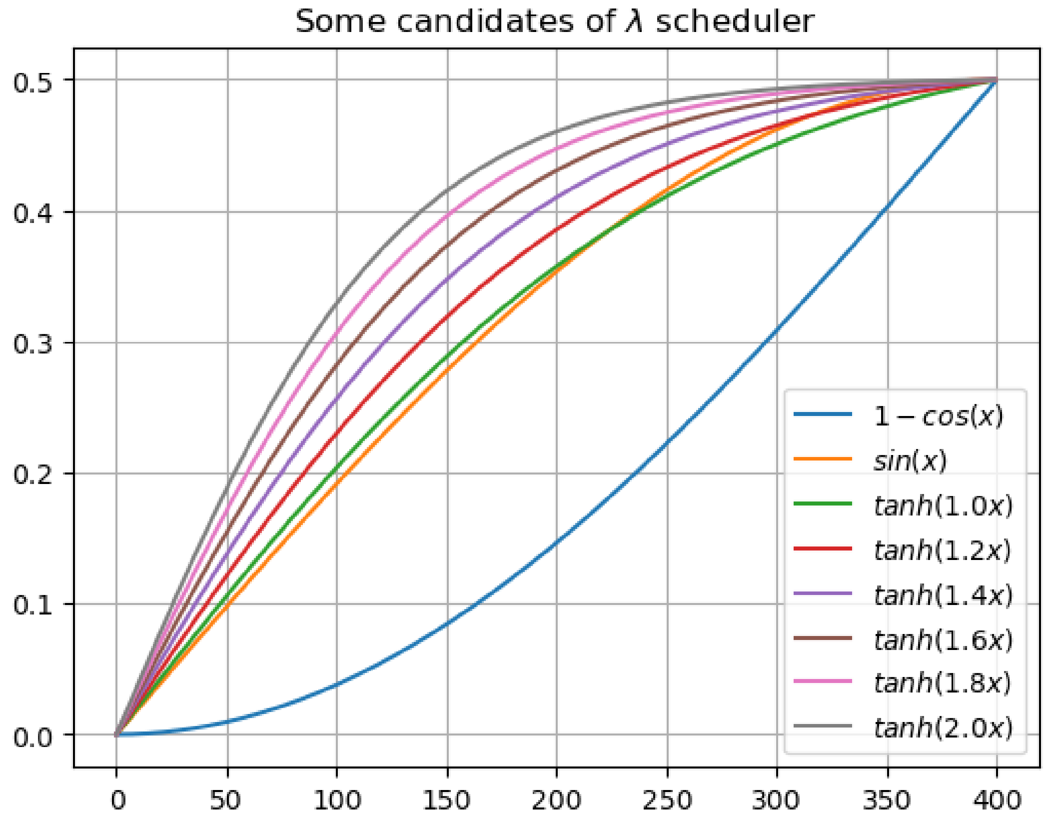
Test accuracies for several η values



Test accuracy trajectories for several η values

Experiments parts (RegMixUp with λ scheduler)

- How about learning mix-up samples in easy → hard samples?
  - [X. Zhou et al., 2021] suggest that learning easy samples first can significantly improve test accuracy.
  - Using this idea, we'll make an λ scheduler as follows :

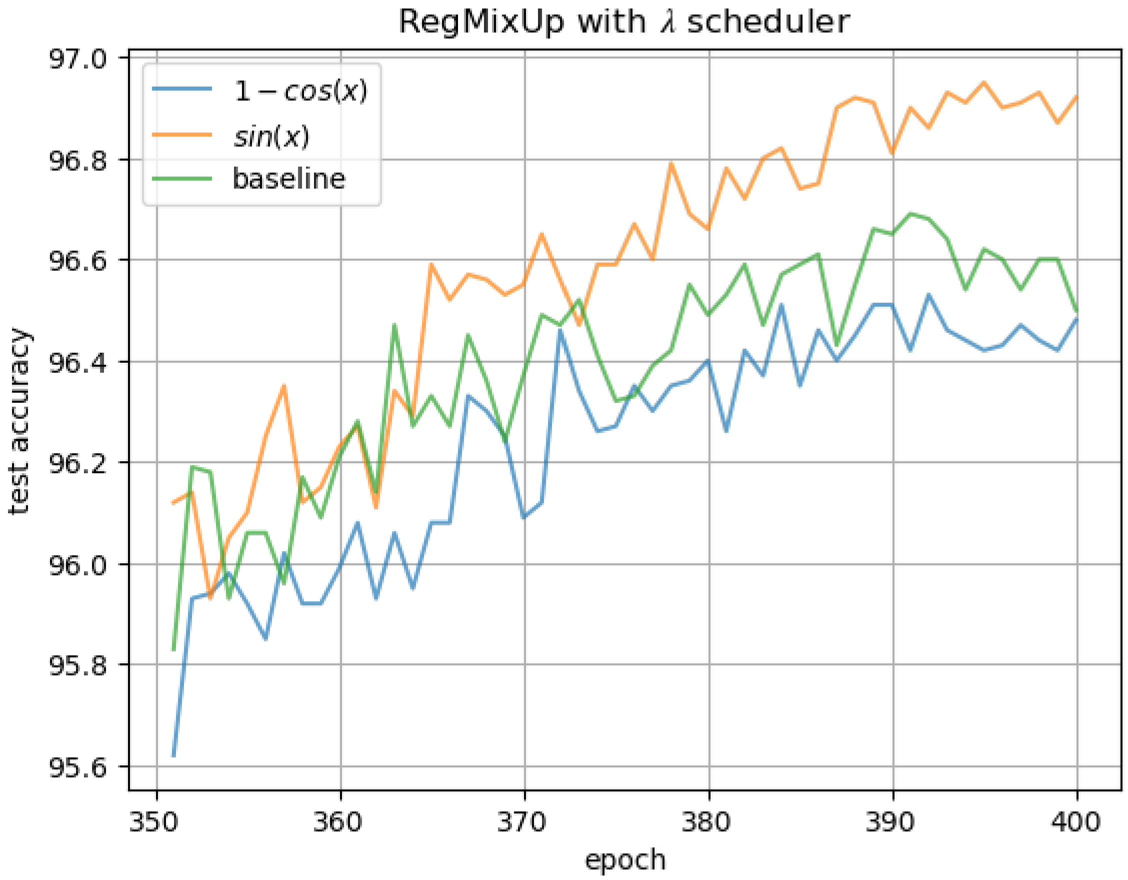


Note :

- **To exploit hard mix-up samples while keeping easy samples, We can use $tanh$ functions with scaler ζ.**
- **As the ζ value get higher, $tanh$ - λ scheduler can enforce hard mix-up sampling at the end tail of training.**
- **$tanh$ - λ scheduler is appropriately rescaled to have $\lambda = 0.5$ at the final epoch.**

Experiments parts (RegMixUp with λ scheduler)

- How about learning mix-up samples in easy → hard samples?
  - When we use $\sin(x)$ - λ scheduler, it improves test accuracy ~0.4% compared to baseline (naïve RegMixUp with $\alpha = 20$), which is comparable to SOTA non-vision mix.



Training method (800 epoch)	Test Accuracy
Original training	95.50
Mix-up	96.62
Manifold mix-up	96.71
RegMixUp (400 epoch)	96.60
MetaMixUP (PreActRN-18)	96.88
CutMix (Vision only)	96.68
PuzzleMix (Vision only)	97.10
AutoMix (Vision only)	97.34
RegMixUp w/ $sine$ λ scheduler (400 epoch)	**96.95**

Probable Future Works

① Can we generalize below approximate mix-up loss into multi-class version?

- If it is possible, R_1, R_2, R_3 can be relate

This analysis is restricted to only binary classification

(In fact, recent papers demonstrated validity of this loss only based on binary classification)

Lemma 3.1 [On Zhang et al., 2021]

Consider the loss function $l(\theta,(x,y)) = h(f_\theta(x)) - y f_\theta(x)$, where h, f are twice differentiable for all $\theta \in \Theta$.

Let us denote $\tilde{D}_\lambda = \frac{\alpha}{\alpha+\beta} Beta(\alpha+1,\beta) + \frac{\beta}{\alpha+\beta} Beta(\beta+1,\alpha)$, D_X = empirical distribution of $S = \{(x_i,y_i)\}_{i=1}^n$

Then, the following holds :

$$L_n^{mix}(\theta,S) = L_n^{std}(\theta,S) + \sum_{i=1}^{3} R_i(\theta,S) + \mathbb{E}_{\lambda\sim\tilde{D}_\lambda}[(1-\lambda)^2\varphi(1-\lambda)]$$

where $\lim_{\lambda\to 0}\varphi(\lambda) = 0$, and

$$R_1(\theta,S) = \frac{\mathbb{E}_{\lambda\sim\tilde{D}_\lambda}[1-\lambda]}{n}\sum_{i=1}^{n}(h'(f_\theta(x_i)) - y_i)\,\nabla f_\theta(x_i)^T\,\mathbb{E}_{r_x\sim D_x}[r_x - x_i]$$

$$R_2(\theta,S) = \frac{\mathbb{E}_{\lambda\sim\tilde{D}_\lambda}[(1-\lambda)^2]}{2n}\sum_{i=1}^{n} h''(f_\theta(x_i))\,\nabla f_\theta(x_i)^T\,\mathbb{E}_{r_x\sim D_x}[(r_x - x_i)\,(r_x - x_i)^T]\,\nabla f_\theta(x_i)$$

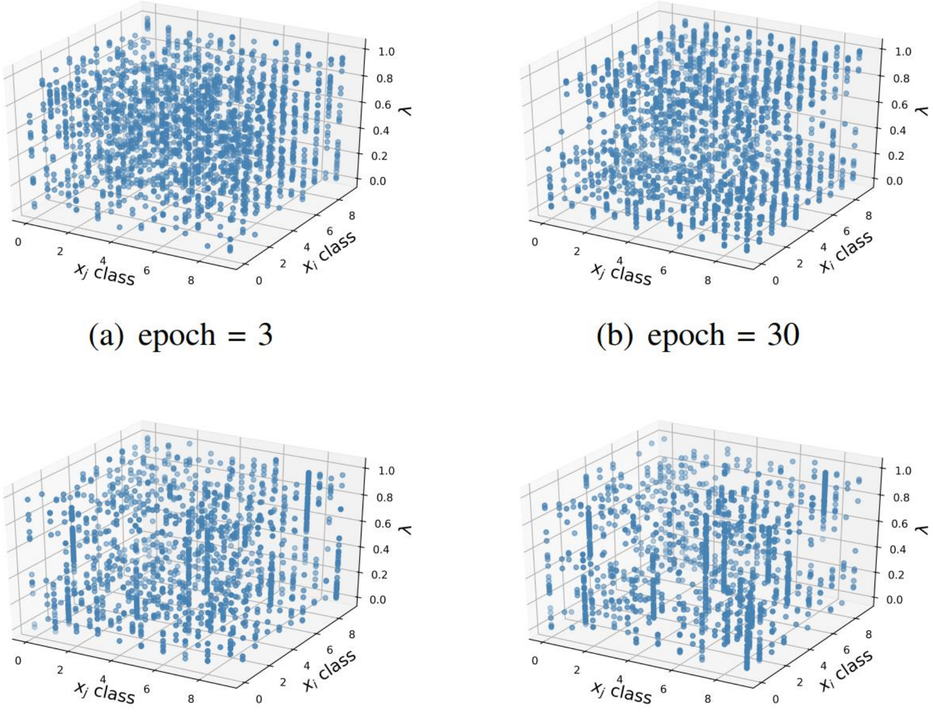
$$R_3(\theta,S) = \frac{\mathbb{E}_{\lambda\sim\tilde{D}_\lambda}[(1-\lambda^2)]}{2n}\sum_{i=1}^{n}(h'(f_\theta(x_i)) - y_i)\,\mathbb{E}_{r_x\sim D_x}[(r_x - x_i)\nabla^2 f_\theta(x_i)(r_x - x_i)^T]$$

Probable Future Works

② **What is the good mix-up samples in terms of model's generalization performance?**

1. How to cleverly choose good mix-up samples that helps model to improve?
   - 1st idea : we can analyze the **common-property of mix-up samples which becomes 'failure cause $\mathcal{C}$'** that is cause for **'failure case' $\mathcal{F}$** in test dataset. [R. Tanno et al., 2022]
   - 2nd idea : Analyze mix-up samples with Shapley value (Intractable) or Influence function (Tractable…?)

2. According to **MetaMixUp** [Z. Mai et al., 2020] it seems that **there is certain good λ range when we mix up two samples from class i and class j**



(a) epoch = 3
(b) epoch = 30
(c) epoch = 90
(d) epoch = 120

λ selection frequency learned by MetaMixUp

Probable Future Works

③ **How to make reasonably good λ scheduler?**

- From some experiments, it turned out that α scheduler shows poor performance compared to λ scheduler. Plus, **It seems that which sample is learned first has a significant effect on performance**.

- Is it possible to construct good λ scheduler that can be comparable to SOTA mix-up (including vision field mix-up)?

④ **Re-verification of Forgetting mix-up**

- We only have experimented hard-pairing mix-up based on forgetting mix-up. However, we can go over it without limiting mix-up sample pool
- **Idea**: Based on forgetting statistic summary, make a mix-up partner using below sampling code. (EX : **[Airplane]** p= [0.0000, 0.0743, 0.3148, 0.1221, 0.0839, 0.0310, 0.0458, 0.0595, 0.4032, 0.1552])

```
Pseudo code: np.random.choice(10, # of samples, p = forgetting statistics[class] / np.amax(forgetting statistics))
```