ReMixMatch: Semi-Supervised Learning with Distribution Alignment and Augmentation Anchoring

School of Industrial and Management Engineering, Korea University

Jong Kook Heo

KOREA UNIVERSITY

Contents

Research Purpose

❖ ReMixMatch: Semi-Supervised Learning with Distribution Alignment and Augmentation Anchoring

- Google Research 에서 연구되었으며, 2022년 8월 27일 기준 494회 인용됨

REMIXMATCH: SEMI-SUPERVISED LEARNING WITH DISTRIBUTION ALIGNMENT AND AUGMENTATION ANCHORING

David Berthelot, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Han Zhang, Colin Raffel
Google Research
{dberth,ncarlini,cubuk,kurakin,zhanghan,craffel}@google.com

Kihyuk Sohn
Google Cloud AI
kihyuks@google.com

ABSTRACT

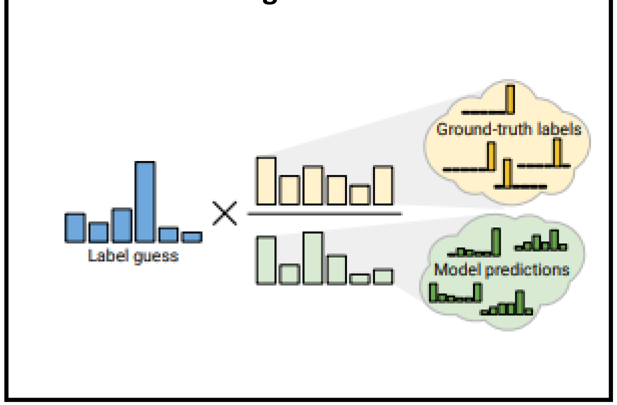
We improve the recently-proposed “MixMatch” semi-supervised learning algorithm by introducing two new techniques: distribution alignment and augmentation anchoring. *Distribution alignment* encourages the marginal distribution of predictions on unlabeled data to be close to the marginal distribution of ground-truth labels. *Augmentation anchoring* feeds multiple strongly augmented versions of an input into the model and encourages each output to be close to the prediction for a weakly-augmented version of the same input. To produce strong augmentations, we propose a variant of AutoAugment which learns the augmentation policy while the model is being trained. Our new algorithm, dubbed ReMixMatch, is significantly more data-efficient than prior work, requiring between $5\times$ and $16\times$ less data to reach the same accuracy. For example, on CIFAR-10 with 250 labeled examples we reach 93.73% accuracy (compared to MixMatch’s accuracy of 93.58% with $4{,}000$ examples) and a median accuracy of 84.92% with just **four** labels per class. We make our code and data open-source at https://github.com/google-research/remixmatch.

Research Purpose

❖ Summary

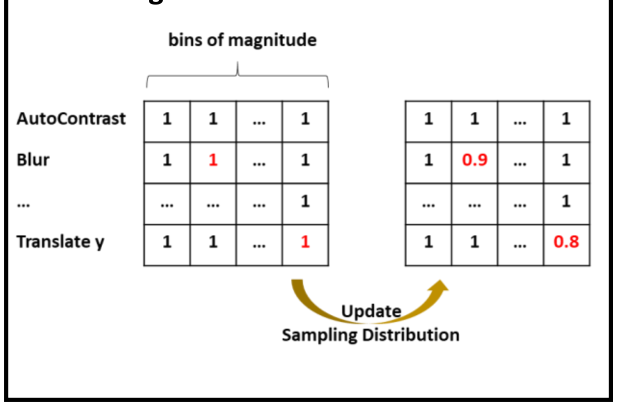
- MixMatch 에 **Distribution Alignment** 와 **Augmentation Anchoring** 요소를 추가로 적용
- CIFAR 10 Dataset 에서 MixMatch 보다 16배 높은 Data Efficiency 를 보여줌
  - ✓ 250 Labeled data 성능이 4000 Labeled data 로 학습한 MixMatch 와 동일
- AutoAugment 보다 시간/비용적으로 효율적인 CTAugment 를 제안
  - ✓ CTAugment 는 Supervision 없이 동적으로 Augmentation 의 분포를 dynamic 하게 조정

Distribution Aligment



Augmentation Anchoring

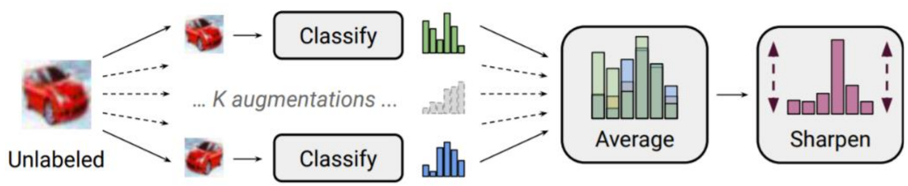
CTAugment



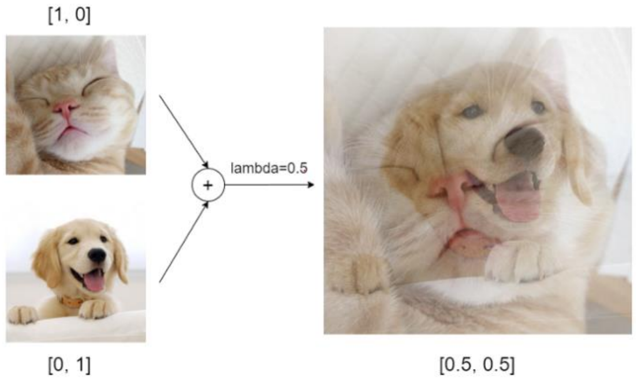
ReMixMatch

❖ Background - Mixmatch

- Semi-SL 의 대표적인 아이디어인 Consistency Regularization, Entropy Minimization, Regularization 을 전부 하나의 알고리즘으로 병합
- Consistency Regularization : perturbation 에도 예측 값이 바뀌지 않도록(**Averaging**)
- Entropy Minimization : 모델의 예측값의 분포가 high-confident 하도록(**Sharpening**)
- Regularization : 모델이 over-parameterize 되지 않도록
  - ✓ L2 weight decay vs **Mix-up**



Label Guessing on Unlabeled data



Mix-up with labeled data

https://medium.com/@wolframalphav1.0/easy-way-to-improve-image-classifier-performance-part-1-mixup-augmentation-with-codes-33288db92de5

ReMixMatch

❖ Distribution Alignment

- Semi SL 의 목적은 unlabeled data 를 통해 모델의 성능을 향상 시키는 것
- 가장 일반적인 방법은 unlabeled data 의 input 과 output 의 mutual information 을 증가 시키는 것
- **Fariness** + **Entropy Minimization**

$$\mathcal{I}(y;x) = \iint p(y,x) \log \frac{p(y,x)}{p(y)p(x)} \, \mathrm{d}y \, \mathrm{d}x$$
$$= \boxed{\mathcal{H}(\mathbb{E}_x[p_{\mathrm{model}}(y|x;\theta)])} - \boxed{\mathbb{E}_x[\mathcal{H}(p_{\mathrm{model}}(y|x;\theta))]}$$

예측 분포의 전체 평균의 Entropy

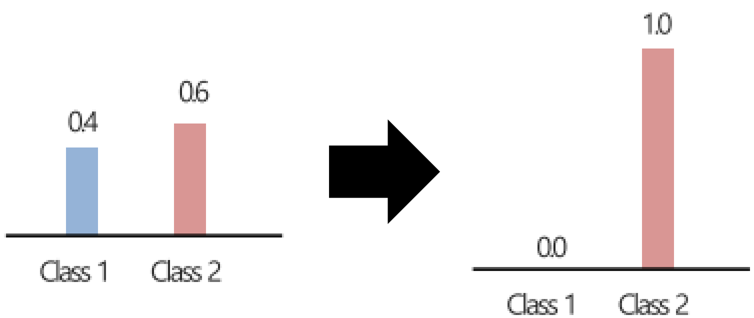
개별 예측 분포의 Entropy 의 기댓값

DMQA

ReMixMatch

❖ Distribution Alignment

- Semi SL 의 목적은 unlabeled data 를 통해 모델의 성능을 향상 시키는 것
- 가장 일반적인 방법은 unlabeled data 의 input 과 output 의 mutual information 을 증가 시키는 것
- Fariness + **Entropy Minimization**
  - ✓ Mixmatch 에서 Temperature Sharpening 으로 이미 해결하였음



$$-\boxed{\mathbb{E}_x[\mathcal{H}(p_{\text{model}}(y|x;\theta))]}$$

개별 예측 분포의 Entropy 의 기댓값

Unlabeled data 의 개별 예측 값은 high-confident 해야한다

DMQA

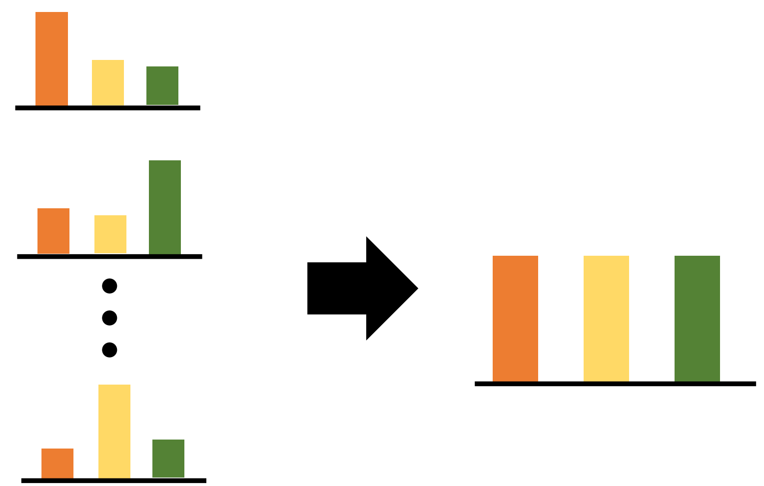
ReMixMatch

❖ Distribution Alignment

- Semi SL 의 목적은 unlabeled data 를 통해 모델의 성능을 향상 시키는 것
- 가장 일반적인 방법은 unlabeled data 의 input 과 output 의 mutual information 을 증가 시키는 것
- **Fariness** + Entropy Minimization
  - ✓ 이거도 반영해보자!

$$\boxed{\mathcal{H}(\mathbb{E}_x[p_{\mathrm{model}}(y|x;\theta)])}$$

예측 분포의 전체 평균의 Entropy



Unlabeled data 의 Prediction의 전체 평균은 Uniform 해야한다

ReMixMatch

❖ Distribution Alignment

- Semi SL 의 목적은 unlabeled data 를 통해 모델의 성능을 향상 시키는 것
- 가장 일반적인 방법은 unlabeled data 의 input 과 output 의 mutual information 을 증가 시키는 것
- **Fariness** + **Entropy Minimization**

$$\mathcal{I}(y;x) = \iint p(y,x)\log\frac{p(y,x)}{p(y)p(x)}\,\mathrm{d}y\,\mathrm{d}x$$

$$= \boxed{\mathcal{H}(\mathbb{E}_x[p_{\text{model}}(y|x;\theta)])} - \boxed{\mathbb{E}_x[\mathcal{H}(p_{\text{model}}(y|x;\theta))]}$$

원래 클래스 분포는 균일한데, **개별 예측 값의 분포는 불균일 하도록**

ReMixMatch

❖ Distribution Alignment

- Semi SL 의 목적은 unlabeled data 를 통해 모델의 성능을 향상 시키는 것
- 가장 일반적인 방법은 unlabeled data 의 input 과 output 의 mutual information 을 증가 시키는 것
- **Fariness** + **Entropy Minimization**

$$\mathcal{I}(y;x) = \iint p(y,x)\log\frac{p(y,x)}{p(y)p(x)}\,\mathrm{d}y\,\mathrm{d}x$$

$$= \boxed{\mathcal{H}(\mathbb{E}_x[p_{\text{model}}(y|x;\theta)])} - \boxed{\mathbb{E}_x[\mathcal{H}(p_{\text{model}}(y|x;\theta))]}$$

클래스 분포가 균일하지 않다면??

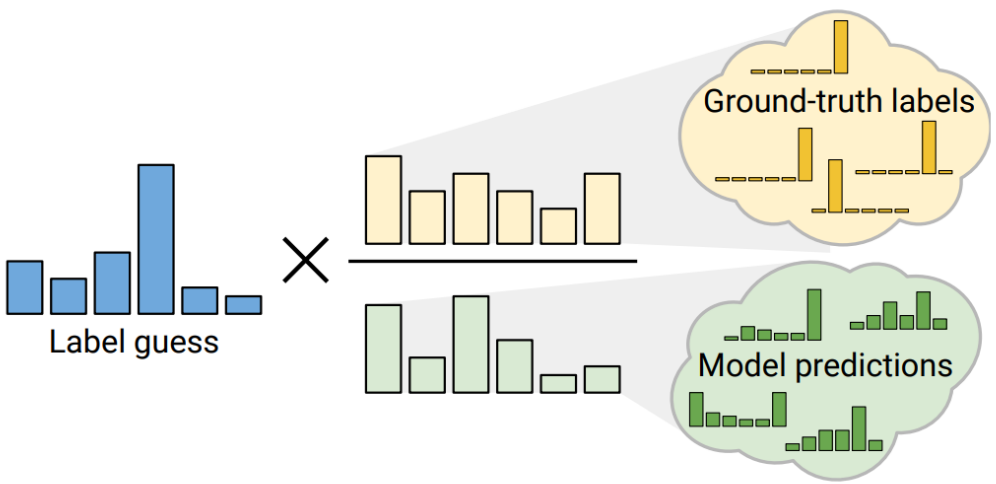
원래 클래스 분포는 균일한데, 개별 예측 값의 분포는 불균일 하도록

Unlabeled Prediction 이 Uniform 이 아니라 Labeled Data 의 클래스 비율을 따라가도록 하자

ReMixMatch

❖ Distribution Alignment

- Labeled Data 의 Class Ratio 를 따라가도록 Loss Function 설계 가능
- 하지만 추가적인 Loss 나 hyper-parameter 없이 간단하게 병합할 수 없을까?
- 아래 식을 통해 Unlabeled data 의 예측 값을 보정
  - ✓ 개인적인 생각으로는 Importance Sampling 과 유사한 듯



$$\tilde{q} = Normalize\ (q)$$
$$= q \times \frac{p(y)}{\tilde{p}(y)}$$

$q = p_{model}(y|u;\theta)$

$p(y)$: Labeled data의 label 분포의 average

$\tilde{p}(y)$: Unlabeled data의 예측분포(q)의 moving average

ReMixMatch

❖ Augmentation Anchoring

- Mixmatch 에서는 Fip 과 Crop 등의 Weak Augmentation 만 적용
- 관련 연구(Xie et al, 2019) 왈,
  - ✓ **"Stronger forms of Augmentation can significantly improve the performance of consistency regularization!!"**

- **Weak Augmentation 대신 AutoAugment 를 적용해서 Consistency Regularization 향상을 해보자!!**

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. Unsupervised data augmentation for consistency training. arXiv preprint arXiv:1904.12848, 2019

ReMixMatch

❖ Augmentation Anchoring

- Mixmatch 에서는 Fip 과 Crop 등의 Weak Augmentation 만 적용
- 관련 연구(Xie et al, 2019) 왈,
  - ✓ "Stronger forms of Augmentation can significantly improve the performance of consistency regularization!!"
- Weak Augmentation 대신 AutoAugment 를 적용해서 Consistency Regularization 향상을 해보자!!

실패

학습 시 수렴하지 않음

Stronger Augmentation 으로 인한 개별 예측 값이 서로 너무 다름

→①**평균 값이 유의미한 타겟이 아님**

너무 오래 걸림

강화학습을 통해 Sub-policy를 학습 해야 함

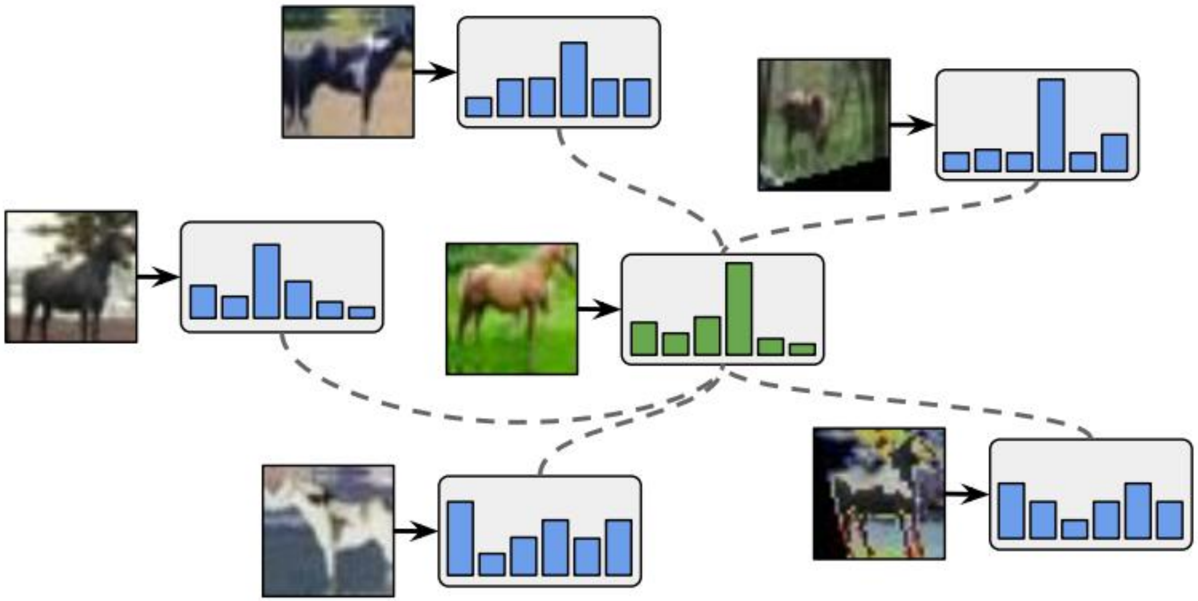
→②**Sub-policy의 학습 시간이 오래걸림**

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. Unsupervised data augmentation for consistency training. arXiv preprint arXiv:1904.12848, 2019

ReMixMatch

❖ Augmentation Anchoring

- ① Stronger Augmentation 이 유의미한 Target 을 따라가야함
- **Weak Augmentation 의 Guessed Label 을 Target 으로 사용하자**



DMQA

ReMixMatch

❖ Augmentation Anchoring

- ② 추가적인 학습이 필요 없는 Augmentation Policy 가 필요함
- CT Augment 를 제안 → 학습 과정에서 각 Augmentation 의 강도에 따른 확률을 동적으로 변형
- **Step 1. 각 Augmentation 별로 magnitude 에 대한 bin 을 설정(초기 값 1)**

bins of magnitude

AutoContrast	1	1	…	1
Blur	1	1	…	1
…	…	…	…	1
Translate y	1	1	…	1

Transformation	Description	Parameter	Range
Autocontrast	Maximizes the image contrast by setting the darkest (lightest) pixel to black (white), and then blends with the original image with blending ratio λ.	λ	[0, 1]
Blur			
Brightness	Adjusts the brightness of the image. $B = 0$ returns a black image, $B = 1$ returns the original image.	B	[0, 1]
Color	Adjusts the color balance of the image like in a TV. $C = 0$ returns a black & white image, $C = 1$ returns the original image.	C	[0, 1]
Contrast	Controls the contrast of the image. A $C = 0$ returns a gray image, $C = 1$ returns the original image.	C	[0, 1]
Cutout	Sets a random square patch of side-length ($L\times$image width) pixels to gray.	L	[0, 0.5]
Equalize	Equalizes the image histogram, and then blends with the original image with blending ratio λ.	λ	[0, 1]
Invert	Inverts the pixels of the image, and then blends with the original image with blending ratio λ.	λ	[0, 1]
Identity	Returns the original image.		
Posterize	Reduces each pixel to B bits.	B	[1, 8]
Rescale	Takes a center crop that is of side-length ($L\times$image width), and rescales to the original image size using method M.	L	[0.5, 1.0]
		M	see caption
Rotate	Rotates the image by θ degrees.	θ	[-45, 45]
Sharpness	Adjusts the sharpness of the image, where $S = 0$ returns a blurred image, and $S = 1$ returns the original image.	S	[0, 1]
Shear_x	Shears the image along the horizontal axis with rate R.	R	[-0.3, 0.3]
Shear_y	Shears the image along the vertical axis with rate R.	R	[-0.3, 0.3]
Smooth	Adjusts the smoothness of the image, where $S = 0$ returns a maximally smooth image, and $S = 1$ returns the original image.	S	[0, 1]
Solarize	Inverts all pixels above a threshold value of T.	T	[0, 1]
Translate_x	Translates the image horizontally by ($\lambda\times$image width) pixels.	λ	[-0.3, 0.3]
Translate_y	Translates the image vertically by ($\lambda\times$image width) pixels.	λ	[-0.3, 0.3]

ReMixMatch

❖ Augmentation Anchoring

- ② 추가적인 학습이 필요 없는 Augmentation Policy 가 필요함
- CT Augment 를 제안 → 학습 과정에서 각 Augmentation 의 강도에 따른 확률을 동적으로 변형
- **Step 2. 임의의 Augmentation 두 개를 선별하고, Augmentation 별로 bin 의 가중치를 따라 magnitude 를 추출**

bins of magnitude

AutoContrast	1	1	…	1
Blur	1	1	…	1
…	…	…	…	1
Translate y	1	1	…	1

Blur : 0.05

Translate y : -0.25

Sample two augmentations and magnitudes

DMQA

ReMixMatch

❖ Augmentation Anchoring

- ② 추가적인 학습이 필요 없는 Augmentation Policy 가 필요함
- CT Augment 를 제안 → 학습 과정에서 각 Augmentation 의 강도에 따른 확률을 동적으로 변형
- **Step 3. Labeled Data 에 대해 해당 Augmentation 에 대한 예측 확률 값 계산, Label 과 비교**

Blur : 0.05
Translate y : -0.25

Class1	Class2	Class3
0.6	0.3	0.1
0.2	0.7	0.1
…	…	…
0.1	0.3	0.6

Label
1
2
..
3

$$w = 1 - \frac{1}{2L}\Sigma|p_{model}(y|x;\theta) - p|$$

DMQA

ReMixMatch

❖ Augmentation Anchoring

- ②추가적인 학습이 필요 없는 Augmentation Policy 가 필요함
- CT Augment 를 제안 → 학습 과정에서 각 Augmentation 의 강도에 따른 확률을 동적으로 변형
- **Step 4. 샘플된 Augmentation 의 magnitude bin 에 대한** m_i**에** w **를 반영 & 정규화**

bins of magnitude

AutoContrast	1	1	…	1
Blur	1	1	…	1
…	…	…	…	1
Translate y	1	1	…	1

Normalize(	1	1	…	1	**)**
Normalize(	0.8	1	…	1	**)**
Normalize(	…	…	…	1	**)**
Normalize(	1	0.8	…	1	**)**

$$m_i \leftarrow 0.99 m_i + 0.01 w$$

DMQA

ReMixMatch

❖ Augmentation Anchoring

- ② 추가적인 학습이 필요 없는 Augmentation Policy 가 필요함
- CT Augment 를 제안 → 학습 과정에서 각 Augmentation 의 강도에 따른 확률을 동적으로 변형
- **m_i 가 0.8 이하인 경우 0으로 가중치를 설정하여 예측 값의 Confidence 가 낮은 Augmentation은 사용하지 않음**
  - ✓ Strong Augmentation 으로 인한 개별 예측 값의 불안정성을 해결

ReMixMatch

❖ Putting it all together

- $\boldsymbol{X'}$: Strong Augmented Labeled Data + Mix-up with Unlabeled Data(Strong + Weak)
- $\boldsymbol{U'}$: Unlabeled Data(Strong + Weak) + Mix-up with Strong Augmented Labeled Data
- $\boldsymbol{U'_1}$: First Strong Augmented Unlabeled Data with **No Mix-up**

Algorithm 1 ReMixMatch algorithm for producing a collection of processed labeled examples and processed unlabeled examples with label guesses (cf. Berthelot et al. (2019) Algorithm 1.)

Input: Batch of labeled examples and their one-hot labels $\mathcal{X} = \{(x_b, p_b) : b \in (1, \ldots, B)\}$, batch of unlabeled examples $\mathcal{U} = \{u_b : b \in (1, \ldots, B)\}$, sharpening temperature T, number of augmentations K, Beta distribution parameter α for MixUp.
for $b = 1$ **to** B **do**
  $\hat{x}_b = \text{StrongAugment}(x_b)$ *// Apply strong data augmentation to x_b*
  $\hat{u}_{b,k} = \text{StrongAugment}(u_b); k \in \{1, \ldots, K\}$ *// Apply strong data augmentation K times to u_b*
  $\tilde{u}_b = \text{WeakAugment}(u_b)$ *// Apply weak data augmentation to u_b*
  $q_b = p_{\text{model}}(y \mid \tilde{u}_b; \theta)$ *// Compute prediction for weak augmentation of u_b*
  $q_b = \text{Normalize}(q_b \times p(y)/\tilde{p}(y))$ *// Apply distribution alignment*
  $q_b = \text{Normalize}(q_b^{1/T})$ *// Apply temperature sharpening to label guess*
end for
$\hat{\mathcal{X}} = ((\hat{x}_b, p_b); b \in (1, \ldots, B))$ *// Augmented labeled examples and their labels*
$\hat{\mathcal{U}}_1 = ((\hat{u}_{b,1}, q_b); b \in (1, \ldots, B))$ *// First strongly augmented unlabeled example and guessed label*
$\hat{\mathcal{U}} = ((\hat{u}_{b,k}, q_b); b \in (1, \ldots, B), k \in (1, \ldots, K))$ *// All strongly augmented unlabeled examples*
$\hat{\mathcal{U}} = \hat{\mathcal{U}} \cup ((\tilde{u}_b, q_b); b \in (1, \ldots, B))$ *// Add weakly augmented unlabeled examples*
$\mathcal{W} = \text{Shuffle}(\text{Concat}(\hat{\mathcal{X}}, \hat{\mathcal{U}}))$ *// Combine and shuffle labeled and unlabeled data*
$\mathcal{X}' = (\text{MixUp}(\hat{\mathcal{X}}_i, \mathcal{W}_i); i \in (1, \ldots, |\hat{\mathcal{X}}|))$ *// Apply MixUp to labeled data and entries from $\mathcal{W}$*
$\mathcal{U}' = (\text{MixUp}(\hat{\mathcal{U}}_i, \mathcal{W}_{i+|\hat{\mathcal{X}}|}); i \in (1, \ldots, |\hat{\mathcal{U}}|))$ *// Apply MixUp to unlabeled data and the rest of $\mathcal{W}$*
return $\mathcal{X}', \mathcal{U}', \hat{\mathcal{U}}_1$

Augmentation Anchoring & Distribution Alignment

ReMixMatch

❖ Putting it all together

- $\boldsymbol{X'}$: Strong Augmented Labeled Data + Mix-up with Unlabeled Data(Strong + Weak)
- $\boldsymbol{U'}$: Unlabeled Data(Strong + Weak) + Mix-up with Strong Augmented Labeled Data
- $\boldsymbol{U'_1}$: First Strong Augmented Unlabeled Data with **No Mix-up**
- 최종 Loss Function은 아래와 같음
  - ✓ 3번과 4번 항을 추가하였을 때 성능이 더욱 증가
  - ✓ 4번은 Self SL 의 Pretext-task 인 Rotation 을 결합

$$\sum_{x,p\in\mathcal{X}'} \mathrm{H}(p, p_{\text{model}}(y|x;\theta)) + \lambda_{\mathcal{U}} \sum_{u,q\in\mathcal{U}'} \mathrm{H}(q, p_{\text{model}}(y|u;\theta))$$
$$+\lambda_{\hat{\mathcal{U}}_1} \sum_{u,q\in\hat{\mathcal{U}}_1} \mathrm{H}(q, p_{\text{model}}(y|u;\theta)) + \lambda_r \sum_{u\in\hat{\mathcal{U}}_1} \mathrm{H}(r, p_{\text{model}}(r|\,\mathrm{Rotate}(u,r);\theta))$$

Results

❖ CIFAR 10 & SVHN

- CIFAR 10 : 250 개 label 을 사용하였을 때 4000개 Label 을 쓴 MixMatch와 동일한 성능을 나타냄

	CIFAR-10			SVHN		
Method	250 labels	1000 labels	4000 labels	250 labels	1000 labels	4000 labels
VAT	36.03±2.82	18.64±0.40	11.05±0.31	8.41±1.01	5.98±0.21	4.20±0.15
Mean Teacher	47.32±4.71	17.32±4.00	10.36±0.25	6.45±2.43	3.75±0.10	3.39±0.11
MixMatch	11.08±0.87	7.75±0.32	6.24±0.06	3.78±0.26	3.27±0.31	2.89±0.06
ReMixMatch	6.27±0.34	5.73±0.16	5.14±0.04	3.10±0.50	2.83±0.30	2.42±0.09
UDA, reported*	8.76±0.90	5.87±0.13	5.29±0.25	2.76±0.17	2.55±0.09	2.47±0.15

- Rotation Loss 의 hyper-parameter(λ_r) 를 0.5에서 2로 증가시킬 경우 CIFAR10 은 클래스 당 4 개(총 40개), SVHN 도 40개의 Label 만 가지고 학습해도 준수한 성능을 보임
  - ✓ Few-shot Learning 이 가능하다고 주장

DMQA

Results

❖ Ablation Study

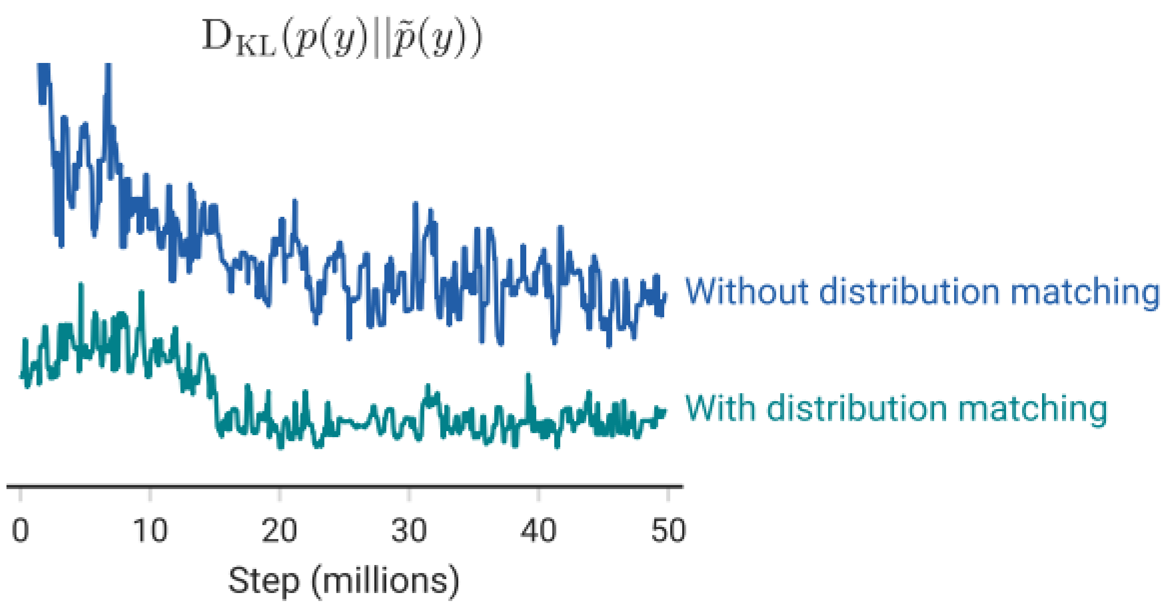
- K : Strong Augmentation 개수(Default=8)
- 결론 : 제안한 요소들 다 넣어야 좋다

Ablation	Error Rate	Ablation	Error Rate
ReMixMatch	5.94	No rotation loss	6.08
With K=1	7.32	No pre-mixup loss	6.66
With K=2	6.74	No dist. alignment	7.28
With K=4	6.21	L2 unlabeled loss	17.28
With K=16	5.93	No strong aug.	12.51
MixMatch	11.08	No weak aug.	29.36

Results

❖ KL Divergence between Labeled Data and Unlabeled Data

- 추가적인 목적 함수 없이, Distribution Alignment 로 보정만 해줘도 Labeled Data 의 분포를 따라감



Conclusion

❖ Data Efficiency 가 높다??

- Labeled Data 개수가 적어도 된다는 측면에서는 동의
- 하지만 실제로 데이터 당 8번의 Strong Augmentation 을 추가해야하기 때문에 Cost Efficiency 가 높다고 생각하지 않음

❖ CTAugment

- 추가적인 학습 없이 Augmentation Magnitude 의 샘플 확률을 조정한 것이 참신했음

❖ Too much

- 너무 복잡한 느낌이 들기도 함
- 이후 나온 Fixmatch 가 더 간단하고 성능도 잘나와서 Fixmatch 를 더 많이 쓰는 듯

Thank You