[Paper Review]

FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence

NeurIPS 2020

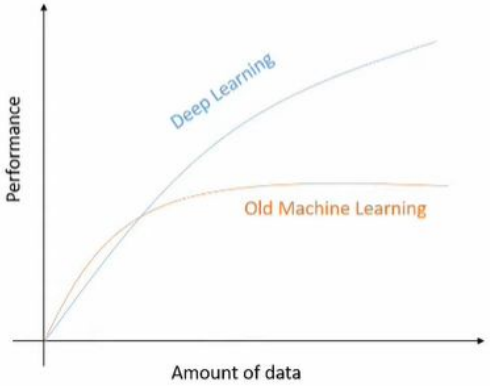
2023.03 김다혜

Index

Introduction

❖ To make better performance in Deep neural networks,

- 많은 수의 데이터 확보가 필연적이다.
  (충분히 많은 데이터가 있어야 현상을 설명할 수 있는 대표성을 띌 수 있기 때문)
- 따라서 훈련 시, "입력-출력"으로 구성된 labeled data가 다수 필요하다.



❖ Labeled data 수집 시 한계

- 표기해야 할 객체 수가 많아지면 많은 인력과 시간이 필요하고,
- 특수분야(ex, medical..)에서는 전문가가 직접 annotation 해야한다. (**cost 증가)**

➔ Semi-Supervised Learning

“ 많은 unlabeled data ” + “ 적은 labeled data ” 활용하여 모델을 학습하는 방식으로, **FixMatch**는 SSL 방법론 중 하나이다.

Introduction

❖ Semi-Supervised Learning

- 대부분의 SSL methods은 unlabeled data를 학습에 활용하기 위해,
  모델이 **unlabeled data를 input으로 넣었을 때 인공 label을 예측할 수 있도록** 훈련시킨다.

  ex) Pseudo-labeling : 모델이 예측한 class를 unlabeled data의 인공 레이블 생성
  ex) Consistency regularization : 'input' 또는 '모델의 함수'를 랜덤하게 변형한 다음, 이에 대한 예측값을 통해 인공 레이블 생성

❖ FixMatch

- FixMatch는 **Consistency Regularization 과 pseudo-labeling 을 모두 활용**하여 인공 label을 생성한다.
- 기존 SSL의 SOTA 방법을 결합하여 간단하면서도 정확한 결과를 보인다.

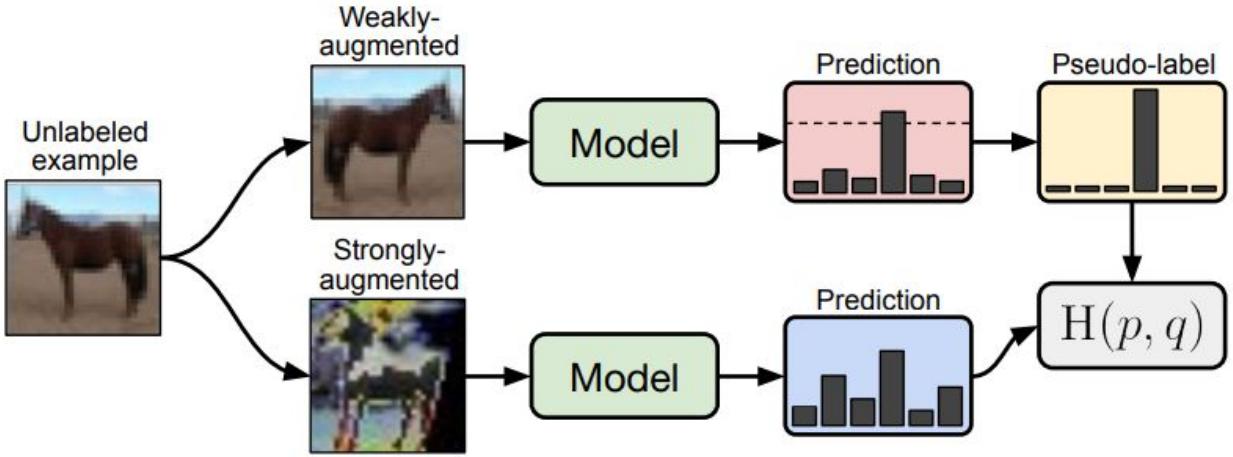
  Contribution

  ➔ FixMatch는 기존 methods 보다 **간단하고, 적은 파라미터를 사용**했음에도 **SSL benchmarks에서 SOTA 달성**

  ➔ 추가적인 Ablation study 통해 어떤 요소가 성공적인 method에 기여했는지 연구를 진행함

Introduction

❖ Diagram of FixMatch



FixMatch의 차별점은

1) **Consistency Regularization 과 pseudo-labeling 을 모두 활용**

2) Consistency Regularization 수행 시 **weak & strong augmentation을 각각 진행**했다는 점이다.

Background

❖ Consistency Regularization

- 하나의 이미지로부터 augmented된 이미지간의 간극을 줄이도록 학습시키는 방법
- 가정 : noise 추가해도 기존 이미지와 비슷한 예측 분포를 띈다.

loss function은 다음과 같다.

$$\sum_{b=1}^{\mu B} \| p_{\mathrm{m}}(y|\, \alpha(u_b)) - p_{\mathrm{m}}(y|\, \alpha(u_b)) \|_2^2$$

- $\mathcal{U} = \{u_b : b \in (1, \ldots, \mu B)\}$: a batch of μB unlabeled examples
- $p_{\mathrm{m}}(y \mid x)$: predicted class distribution
- $\alpha(\cdot)$: weak augmentation

두 예측 확률 분포를 동일하게 만들도록 (= **두 분포의 차이가 줄어들도록**) 학습

- Consistency Regularization은

  1) weakly augmentation 대신 adversarial transformation을 수행하거나 (ex. **Virtual Adversarial Training**),

  2) l2 loss 대신 cross entropy를 사용하거나 (ex. **Virtual Adversarial Training, UDA, ReMixMatch**) ,

  3) strong augmentation을 하는 등 (ex. **UDA, ReMixMatch**)

  여러 가지 방법으로 확장되어 왔다.

Background

❖ Pseudo-labeling

$$\frac{1}{\mu B}\sum_{b=1}^{\mu B}\mathbb{1}\left(\max(q_b) \geq \tau\right)\mathrm{H}(\hat{q}_b, q_b)$$

Max probability in pseudo label

Threshold

- $\mathrm{H}(\hat{q}_b, q_b)$: pseudo-label, 예측확률 간의 Cross Entropy loss

- **모델 자체에서 unlabeled data의 인공 레이블**을 얻는다 (= self-training)
- hard label(확률값이 가장 큰 class의 레이블)만을 사용하여, 이 확률값이 **threshold를 넘길 때만** pseudo-labeling 한다.

  => unlabeled data에 대해 low entropy, high confidence를 가짐 (entropy minimization)

FixMatch

- FixMatch의 Loss function은 다음과 같다.

$$\ell_s + \lambda_u \ell_u$$

(1) labeled data에 적용되는 Supervised loss 와, (2) unlabeled data에 적용되는 Unsupervised loss 로 구성된다.

❖ Supervised Loss ℓ_s

$$\ell_s = \frac{1}{B} \sum_{b=1}^{B} \mathrm{H}(p_b, p_{\mathrm{m}}(y \mid \alpha(x_b)))$$

- 배치사이즈가 B일때, 하나의 배치 내의 평균 Loss function
- Labeled data에 대해서 weakly augmentation을 진행하고, augmented data의 예측 확률과 정답 one-hot vector간의 Cross entropy를 구한다.
- 이 때 weakly augmentation에는 Random flip, Random translation 을 진행한다.

FixMatch

❖ Unsupervised Loss ℓ_u

$$\ell_u = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \mathbb{1}(\max(q_b) \geq \tau)\, \mathrm{H}(\hat{q}_b, p_{\mathrm{m}}(y \mid \mathcal{A}(u_b)))$$

- 배치사이즈가 B일때, Unlabeled data에 대한 배치 내 Loss function

  1) Pseudo-label을 얻기 위해 Unlabeled image에 대해 weakly augmentation 진행한다.

  2) $\hat{q}_b = \arg\max(q_b)$ 가 threshold를 넘을 때, 이를 pseudo-label로 사용, (q_b = weakly augmented data에 대한 output 확률값)

     (threshold 넘지 못하면 해당 loss값= 0, 반영x)

  3) Pseudo-label과, strongly augmented data 예측확률 간의 Cross Entropy를 구한다.

- 기존 Consistency Regularization와 다른점
  - **Pseudo-label은 weakly-augmented image을 기반으로 생성**된다.
  - **Strongly-augmented image에 대한 loss 값이 계산**된다.

FixMatch

❖ FixMatch Algorithm

Algorithm 1 FixMatch algorithm.

Input: Labeled batch $\mathcal{X} = \{(x_b, p_b) : b \in (1, \dots, B)\}$, unlabeled batch $\mathcal{U} = \{u_b : b \in (1, \dots, \mu B)\}$, confidence threshold τ, unlabeled data ratio μ, unlabeled loss weight λ_u.

$\ell_s = \frac{1}{B} \sum_{b=1}^{B} \mathrm{H}(p_b, \alpha(x_b))$ {*Cross-entropy loss for labeled data*}

for $b = 1$ **to** μB **do**

  $q_b = p_{\mathrm{m}}(y \mid \alpha(u_b); \theta)$ {*Compute prediction after applying weak data augmentation of* u_b}

end for

$\ell_u = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \mathbb{1}\{\max(q_b) > \tau\} \mathrm{H}(\arg\max(q_b), p_{\mathrm{m}}(y \mid \mathcal{A}(u_b))$ {*Cross-entropy loss with pseudo-label and confidence for unlabeled data*}

return $\ell_s + \lambda_u \ell_u$

➔ 각 batch에서

1) labeled data 에 대한 Cross Entropy 구하고,

2) unlabeled data의 weakly-augmented image의 예측 확률을 계산하여 pseudo-label 생성한 다음,

3) pseudo-label과 strongly-augmented image에 대한 Cross Entropy를 구하여 최종 loss 값을 반환

FixMatch

❖ FixMatch Loss Function

$$\ell_s + \lambda_u \ell_u$$

λ = Unsupervised loss 의 영향력을 조절하는 고정된 가중치 파라미터

- 일반적으로, 학습 초기에는 unlabeled가 이상치 발생 확률이 높기 때문에 점진적으로 weight를 증가해야 한다.

- **FixMatch 에서 λ 를 따로 조정할 필요 없는 이유 ?**

  ➔ Unsupervised loss에 있는 threshold 값이 그 역할을 대신한다.

  - 학습 초기에는 모델이 불안정하기 때문에 unlabeled는 대부분 max(qb)값이 threshold를 넘지 못한다.

    (= 학습 초기에는 unlabeled를 pseudo-label 로 많이 활용하지 못함)

  - 학습 할수록 정확도가 높아져 threshold 넘는 unlabeled data가 많아지기 때문에 unlabeled 활용 비율이 높아진다.

    (자동적으로 비율이 조정된다.)

FixMatch

❖ Augmentation in FixMatch

Weak augmentation

- Random horizontal flip
- Random translate (vertically 12.5% and horizontally 50%)

Strong augmentation

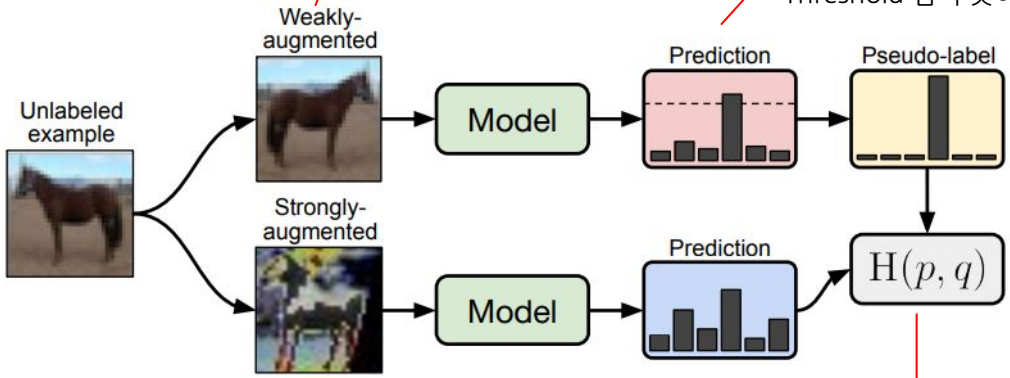
- Cutout
  - 이미지의 일부를 0으로 채움
- RandAugment
  - 변형 정도는 pre-defined range로 고정됨 (UDA에서도 같은 방법 사용)
- CTAugment
  - 각각의 data에 적용되는 변형 정도 달라진다.
  - 학습 과정에서 magnitude bin의 weights 조정을 통해 정도를 조절함

FixMatch

Weak augmentation

- Random horizontal flip
- Random translate (vertically and horizontally)

Threshold 넘으면 -> Pseudo-label 생성
Threshold 넘지 못하면 -> unsupervised loss = 0



$$\ell_u = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \mathbb{1}(\max(q_b) \geq \tau)\, \mathrm{H}(\hat{q}_b, p_{\mathrm{m}}(y \mid \mathcal{A}(u_b)))$$

Strong augmentation

- RandAugment (RA)
- CTAugment (CTA)
- CutOut

FixMatch

❖ FixMatch vs Previous work (UDA, ReMixMatch)

Algorithm	Artificial label augmentation	Prediction augmentation	Artificial label post-processing	Notes
TS / Π-Model	Weak	Weak	None	
Temporal Ensembling	Weak	Weak	None	Uses model from earlier in training
Mean Teacher	Weak	Weak	None	Uses an EMA of parameters
Virtual Adversarial Training	None	Adversarial	None	
UDA	Weak	Strong	Sharpening	Ignores low-confidence artificial labels
MixMatch	Weak	Weak	Sharpening	Averages multiple artificial labels
ReMixMatch	Weak	Strong	Sharpening	Sums losses for multiple predictions
FixMatch	Weak	Strong	Pseudo-labeling	

- UDA와 ReMixMatch는 pseudo-labeling을 사용하지 않고, sharpening을 수행
  => FixMatch의 threshold 기반 pseudo labeling은 샤프닝과 유사한 효과를 갖는다.
- - ReMixMatch는 unlabeled 데이터 loss의 가중치를 반영하지 않는다.
  => FixMatch의 pseudo-labeling에 사용된 threshold는 이와 유사한 효과를 가지고 있다.

➔ FixMatch는 threshold를 통해 UDA 및 ReMixMatch의 두 가지 기술을 결합하여 간단하면서도 효과적인 방법을 제시한다.

Experiments, Ablation Study

- Experiment settings

- 각 클래스당 4n, 25n, 400n개의 labeled samples 사용
- Dataset - Model

  CIFAR-10, CIFAR-100, SVHN -> Wide ResNet-28-2, **STL-10** -> Wide ResNet-37-2, **ImageNet** -> ResNet-50
- ImageNet을 제외하고, 데이터셋의 크기가 작기 때문에 실험을 통해 얻은 최적의 하이퍼 파라미터값 고정해서 사용했다.
- $\lambda u = 1$, $\eta = 0.03$, $\beta = 0.9$, $\tau = 0.95$, $\mu = 7$, $B = 64$, $K = 220$

- Error Rates in CIFAR-10 & CIFAR-100 & SVHN & STL-10

	CIFAR-10			CIFAR-100			SVHN			STL-10
Method	40 labels	250 labels	4000 labels	400 labels	2500 labels	10000 labels	40 labels	250 labels	1000 labels	1000 labels
Π-Model	-	54.26±3.97	14.01±0.38	-	57.25±0.48	37.88±0.11	-	18.96±1.92	7.54±0.36	26.23±0.82
Pseudo-Labeling	-	49.78±0.43	16.09±0.28	-	57.38±0.46	36.21±0.19	-	20.21±1.09	9.94±0.61	27.99±0.83
Mean Teacher	-	32.32±2.30	9.19±0.19	-	53.91±0.57	35.83±0.24	-	3.57±0.11	3.42±0.07	21.43±2.39
MixMatch	47.54±11.50	11.05±0.86	6.42±0.10	67.61±1.32	39.94±0.37	28.31±0.33	42.55±14.53	3.98±0.23	3.50±0.28	10.41±0.61
UDA	29.05±5.93	8.82±1.08	4.88±0.18	59.28±0.88	33.13±0.22	24.50±0.25	52.63±20.51	5.69±2.76	**2.46**±0.24	7.66±0.56
ReMixMatch	**19.10**±9.64	**5.44**±0.05	4.72±0.13	**44.28**±2.06	**27.43**±0.31	**23.03**±0.56	**3.34**±0.20	**2.92**±0.48	2.65±0.08	**5.23**±0.45
FixMatch (RA)	**13.81**±3.37	**5.07**±0.65	**4.26**±0.05	48.85±1.75	28.29±0.11	**22.60**±0.12	**3.96**±2.17	**2.48**±0.38	**2.28**±0.11	7.98±1.50
FixMatch (CTA)	**11.39**±3.35	**5.07**±0.33	**4.31**±0.15	49.95±3.01	28.64±0.24	23.18±0.11	7.65±7.65	**2.64**±0.64	**2.36**±0.19	**5.17**±0.63

실험 결과, CIFAR-100을 제외하고 모두 기존 연구보다 나은 성능, SOTA 달성

Experiments, Ablation Study

Results

- CIFAR-100을 제외하고 모두 기존 연구보다 나은 성능을 보인다
- CIFAR-100에서 ReMixMatch가 나은 성능 보인 이유를 분석하기 위해,

  ReMixMatch에 쓰인 **Distribution Alignment를 Fixmatch에 적용하는 추가 실험**을 진행했다.
- Distribution Alignment (DA) 방식을 추가한 결과,

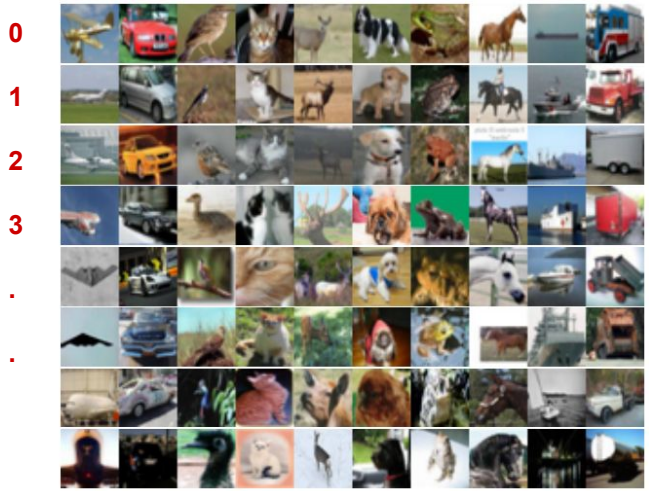
  CIFAR-100에서도 40.14%로 ReMixMatch(44.28%) 보다 좋은 성능 달성 (with 400 labels)
- CTAugment 적용 결과와 RandAugment 적용 결과가 비슷하지만, 4 labels 일 때는 차이 존재한다.

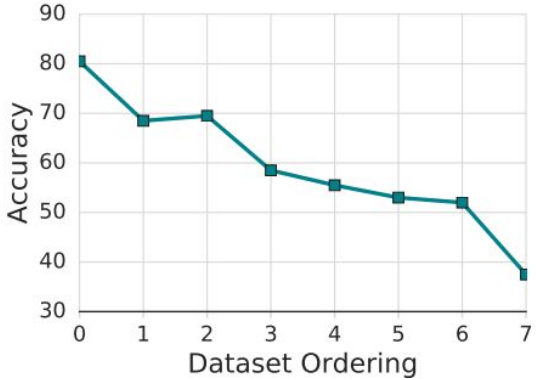
  ➔ **label 수 적을 때 5 fold 에서의 결과의 편차가 큰 것**을 보여준다. (4 labels 일 때 분산 : 3.35%, 25 labels 분산 : 0.33%)

Experiments, Ablation Study

❖ Barely Supervised Learning

- FixMatch 성능의 한계치를 확인해보기 위해, **Barely Supervised Learning**을 수행하였다.
  (CIFAR-10을 사용하여 class 당 하나씩 총 10개의 data로 실험)
- 랜덤하게 고른 4개의 training set으로 실험한 결과, set 별로 accuracy 편차가 큰 것을 확인 (48.58% ~ 85.32%)
- Prototypical 점수가 높은 데이터셋 (대표성을 띄는 dataset) 일수록 높은 accuracy 보임

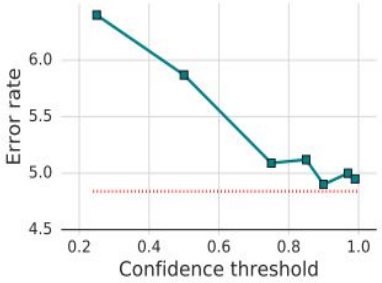




Experiments, Ablation Study

❖ Sharpening and Thresholding

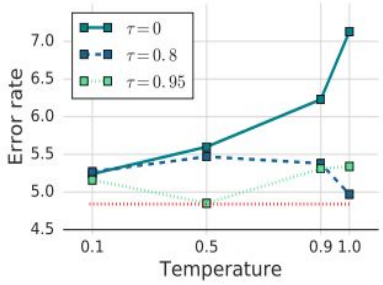
- **T** 는 softmax 값 구할 시 **확률값 scaling하는 역할** 하는 파라미터이다. Pseudo-labeling에서는 T=0으로 둠
- **UDA, MixMatch, RemixMatch**에서 사용한 sharpening 방식은 pseudo-labeling의 soft version이다. (T > 0)



(a)

(a) T=0 (pseudo-labeling) 적용시 threshold에 따른 error rate 확인해보면 0.95일 때 가장 낮은 것을 확인할 수 있다.

➔ T값에 변화를 줬을 때 (Sharpening 방식 적용 시), 위에서 구한 error보다 나은 결과 낼 수 있는가?



(b)

(b) 실험 결과, **위에서 구한 결과(빨간 점선)보다 크게 나은 성능을 달성하지는 못한 것 확인**

➔ T라는 새로운 하이퍼파라미터를 쓰는 것 보다 T=0으로 고정해서 Pseudo-labeling 하는 것이 더 효과적이다. **(FixMatch의 Contribution)**

Experiments, Ablation Study

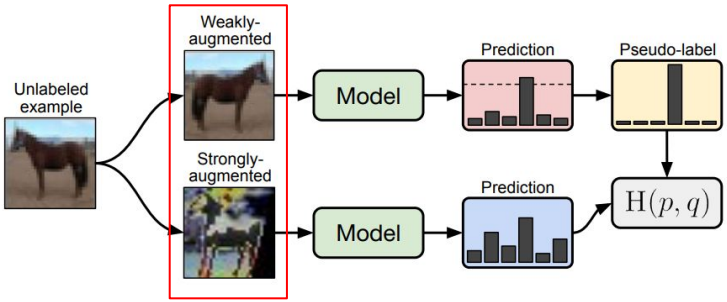
❖ Augmentation Strategy

추가적으로, Augmentation Strategy에 대한 ablation study를 진행했다.

- 앞선 실험 결과에서는 RandAugment를 했을 때와 CTAugment를 했을 때 비슷한 성능을 보였다.
- 추가적으로 Cutout을 진행한 결과, RA / CTA를 CutOut을 함께 썼을 때 더 낮은 error rate을 보이는 것을 확인했다.

Ablation	Error
FixMatch	**4.84**
Only Cutout	6.15
No Cutout	6.15

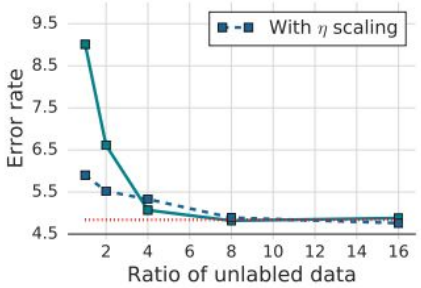
The effect of CutOut



- 위 그림에서 둘 다 strong augmentation을 적용했을 때, **모델이 예측을 잘 수행하지 못하고 발산**한다.
- 둘 다 weak augmentation을 적용했을 때, **모델이 불안정하고, 12% accuracy로 낮은 성능**을 보인다.

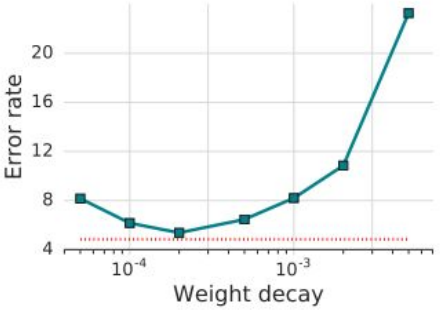
Experiments, Ablation Study

❖ Unlabeled data 비율, weight decay 조정



(a)

- 더 많은 unlabeled 사용할수록 error rate 감소한다.
- 비율이 7,8 넘어가면 감소폭이 크게 줄어든다.



(b)

- Weight decay:
  학습된 모델의 복잡도를 줄이기 위해서 학습 중 weight가 너무 큰 값을 갖지 않도록 Loss function에 Weight가 커질 경우에 대한 패널티
- low-label 환경에서는 큰 영향을 주므로 튜닝이 필요하다.

Experiments, Ablation Study

❖ Optimizer and Learning rate

Optimizer	Hyperparameters			Error
SGD	$\eta = 0.03$	$\beta = 0.90$	Nesterov	**4.84**
SGD	$\eta = 0.03$	$\beta = 0.999$	Nesterov	84.33
SGD	$\eta = 0.03$	$\beta = 0.99$	Nesterov	21.97
SGD	$\eta = 0.03$	$\beta = 0.50$	Nesterov	5.79
SGD	$\eta = 0.03$	$\beta = 0.25$	Nesterov	6.42
SGD	$\eta = 0.03$	$\beta = 0$	Nesterov	6.76
SGD	$\eta = 0.05$	$\beta = 0$	Nesterov	6.06
SGD	$\eta = 0.10$	$\beta = 0$	Nesterov	5.27
SGD	$\eta = 0.20$	$\beta = 0$	Nesterov	5.19
SGD	$\eta = 0.50$	$\beta = 0$	Nesterov	5.74
SGD	$\eta = 0.03$	$\beta = 0.90$		**4.86**
Adam	$\eta = 0.002$	$\beta_1 = 0.9$	$\beta_2 = 0.00$	29.42
Adam	$\eta = 0.002$	$\beta_1 = 0.9$	$\beta_2 = 0.90$	14.42
Adam	$\eta = 0.002$	$\beta_1 = 0.9$	$\beta_2 = 0.99$	15.44
Adam	$\eta = 0.002$	$\beta_1 = 0.9$	$\beta_2 = 0.999$	13.93
Adam	$\eta = 0.0008$	$\beta_1 = 0.9$	$\beta_2 = 0.999$	7.35
Adam	$\eta = 0.0006$	$\beta_1 = 0.9$	$\beta_2 = 0.999$	6.12
Adam	$\eta = 0.0005$	$\beta_1 = 0.9$	$\beta_2 = 0.999$	5.95
Adam	$\eta = 0.0004$	$\beta_1 = 0.9$	$\beta_2 = 0.999$	5.44
Adam	$\eta = 0.0003$	$\beta_1 = 0.9$	$\beta_2 = 0.999$	5.37
Adam	$\eta = 0.0002$	$\beta_1 = 0.9$	$\beta_2 = 0.999$	5.57
Adam	$\eta = 0.0001$	$\beta_1 = 0.9$	$\beta_2 = 0.999$	7.90

Decay Schedule	Error
Cosine (FixMatch)	4.84
Linear Decay (end 0.01)	4.95
Linear Decay (end 0.02)	5.55
No Decay	5.70

- Adam보다 SGD가 더 좋은 성능 보임
- learning rate 작을 수록 좋은 성능
- cosine learning rate decay 사용했지만, Linear 써도 좋은 성능 달성

➔ 이전 연구에서는 이에 대한 연구가 자세히 이루어지지 않았지만,
Optimizer와 파라미터 조정이 성능에 큰 영향을 주는 것을 확인할 수 있다.

Experiments, Ablation Study

❖ Extensions of FixMatch

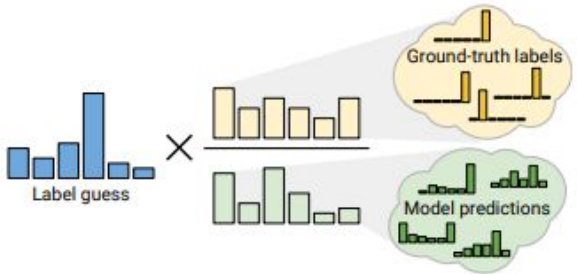
Fixmatch는 알고리즘이 단순하기 때문에 다른 methods와 결합하여 확장 해볼 수 있다.

RemixMatch의 Distribution Alignment을 추가하여 비교 실험을 진행해보았다.

** Distribution Alignment

: Unlabeled data의 예측분포를 Labeled Data의 실제 분포에 맞추어 조정한다.

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



➔ prediction(y)이 unlabeled input(x)과 최대한 의존적인 분포를 나타내기 위해, Mutual Information을 최대화하는 과정을 수행한다.

Results

- Fixmatch에 Distribution Alignment를 추가한 결과, 40 labeled data에서 error rate가 11.38% 에서 9.47%로 감소하였다.
- 또한 400 labeled data에서 40.14%의 error rate를 보였고, 이는 **ReMixMatch가 달성한 44.28%보다 우수함**

Conclusion

- 기존의 SSL methods는 복잡한 알고리즘과 많은 하이퍼파라미터 튜닝의 어려움으로 많은 비용이 들었다.
- Fixmatch는 **Consistency Regularization 과 pseudo-labeling 을 모두 활용하고, threshold를 활용**하여

  간단하면서도 좋은 성능을 달성
- 또한 많은 실험을 통해 성능에 기여한 요소들을 확인하였다. (weight decay, optimizer..)
- 실험에서 볼 수 있듯이, FixMatch의 간단한 알고리즘 덕분에 다른 모델과 결합하여 더 발전된 방법도 연구해 볼 수 있다.
  - 실제로, FixMatch이후 발전된 후속 연구도 확인해 볼 수 있었음

- Selfmatch: Combining contrastive self-supervision and consistency for semi-supervised learning. (2021)

  : self-supervised 방식(SimCLR)으로 학습한 encoder 로부터 FixMatch 구조의 예측 모델 초기 파라미터를 구하는 방법 활용

- Simmatch: Semi-supervised learning with similarity matching (2022)

  : Weak/ Strong Augmented 사이 유사도 분포 차이를 최소화하며 Fixmatch 적용