DMQA Open Seminar

Finding Optimal Augmentation

2022. 07. 08

Data Mining & Quality Analytics Lab.

발표자 : 황성진

Contents

발표자 소개

❖ 황성진

- 고려대학교 산업경영공학과
- Data Mining & Quality Analytics Lab (김성범 교수님)
- 석사 과정 (2021. 3~)

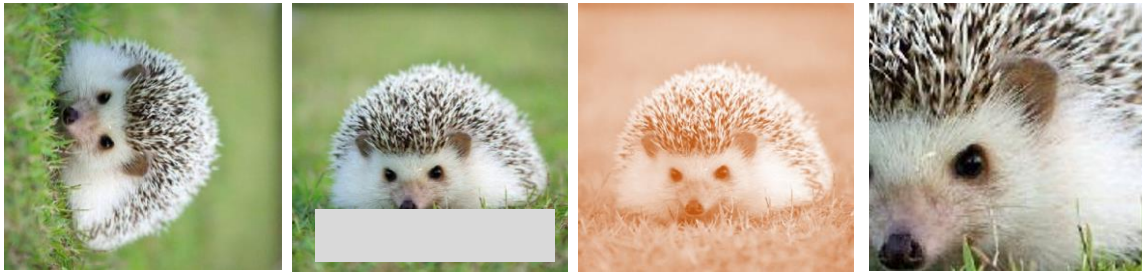
❖ 관심 연구분야

- Machine Learning and Deep Learning
- Active learning in seminconductor manufacturing
- Image Data Augmentation

Data Augmentation_Image

❖ 데이터 증강 : 원본 데이터에 인위적인 변화를 주어, 여러개의 새로운 데이터로 늘리는 방법

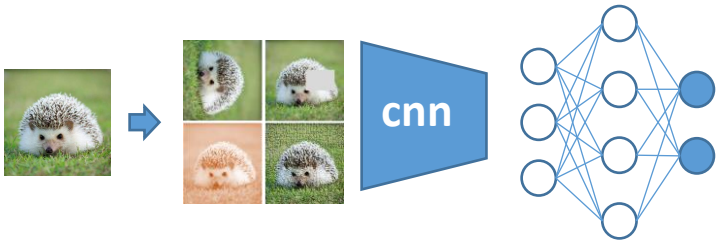
"고슴도치"



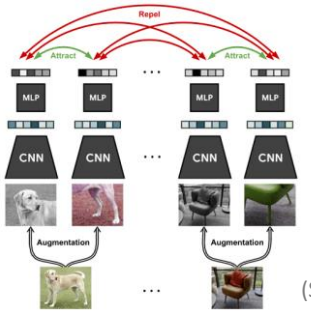
"고슴도치"

<활용 예시>

증강된 Train Data로 사용



Self-supervised learning 활용



"Contrastive learning"

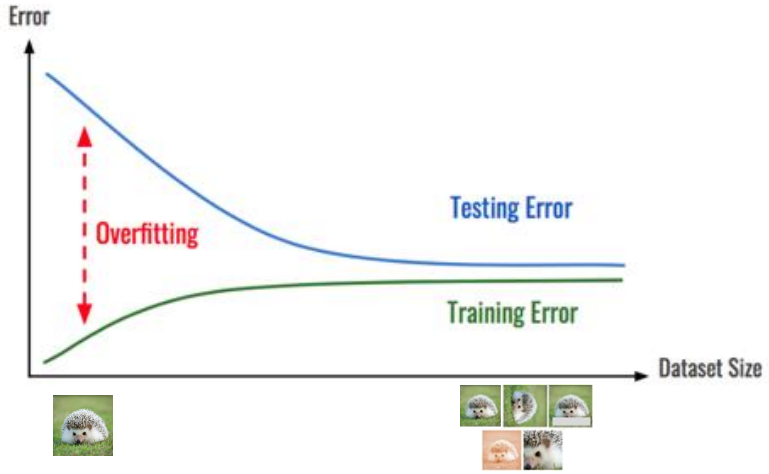
서로 다른 두 Augmentation이 같아지도록...

(SimCLR)

Data Augmentation

❖ Data Augmentation의 의미?

- 성능 개선과 함께 Overfitting 문제를 해결할 수 있는 기법 중 하나.
- 원본 데이터 사이에 있는 Missing data point를 채워주는 기법 (Faster autoaugment, 2020)



과적합 되지 않도록 다양한 변형을 주어 늘어난 Dataset으로 학습

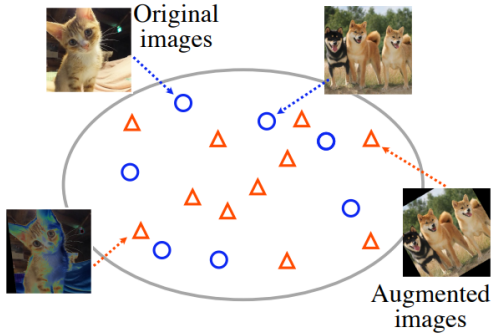


Figure 2. We regard data augmentation as a process that **fills missing data points of the original training data**; therefore, our objective is to minimize the distance between the distributions of augmented data and the original data using adversarial learning.

원본 데이터 사이에 있는 missing data point를 채워줌

데이터가 가진 "근본적인 특징을 잘 잡을 수 있도록" 다양한 형태를 학습시키는 것이 목적

Ref) https://hackernoon.com/memorizing-is-not-learning-6-tricks-to-prevent-overfitting-in-machine-learning-820b091dc42
Hataya, R., Zdenek, J., Yoshizoe, K., & Nakayama, H. (2020, August). Faster autoaugment: Learning augmentation strategies using backpropagation. In European Conference on Computer Vision (pp. 1-16). Springer, Cham.

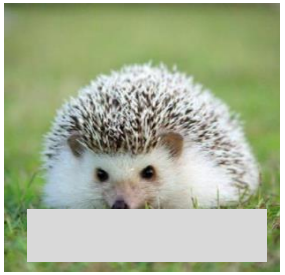
Data Augmentation_예시

❖ 목적 : 데이터의 근본적인 특징을 더 잘 잡아내기 위함!

"고슴도치"

잔디 위
세모 귀
까만 눈 좌우 2개
아래로 튀어나온 코
흑백이 섞인 가시
백색 털
위쪽 뿌연 배경
동그란 몸

왼쪽 잔디
세모 귀
까만 눈 상하 2개
왼쪽으로 튀어나온 코
흑백이 섞인 가시
백색 털
오른쪽 뿌연 배경
동그란 몸



~~잔디~~
회색 네모
세모 귀
까만 눈 좌우 2개
아래로 튀어나온 코
흑백이 섞인 가시
백색 털
위쪽 뿌연 배경
동그란 몸

세모 귀
눈동자 좌우 2개
아래로 튀어나온 코
붉은 가시
붉은 털
위쪽 뿌연 배경
동그란 몸

세모 귀
까만 눈
튀어나온 코
흑백이 섞인 가시
하얀 털

"고슴도치"

Ref) Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of big data*, *6*(1), 1-48.

Data Augmentation_예시

❖ 목적 : 데이터의 근본적인 특징을 더 잘 잡아내기 위함!

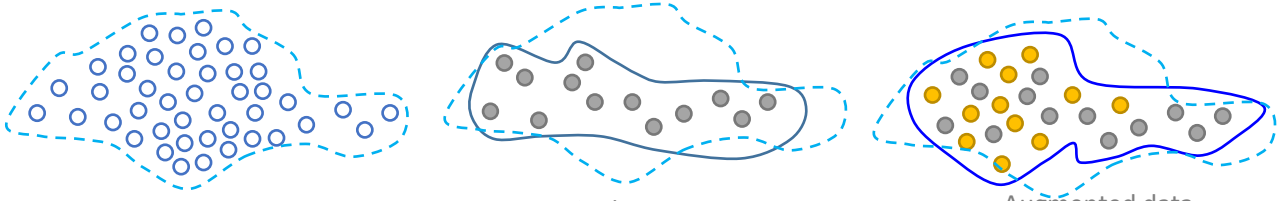
"고슴도치"

"고슴도치"

잔디 위
세모 귀
까만 눈 좌우 2개
아래로 **튀어나온 코**
흑백이 섞인 가시
백색 **털**
위쪽 뿌연 배경
동그란 몸

일부가 변형된 증강데이터에 의해

"고슴도치"의 특징을 더 확실하게 학습할 수 있음



Ref) Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of big data*, *6*(1), 1-48.

Data Augmentation

❖ 주의 : 변형에 의해서 "Label"의 특징을 잃어서는 안됨

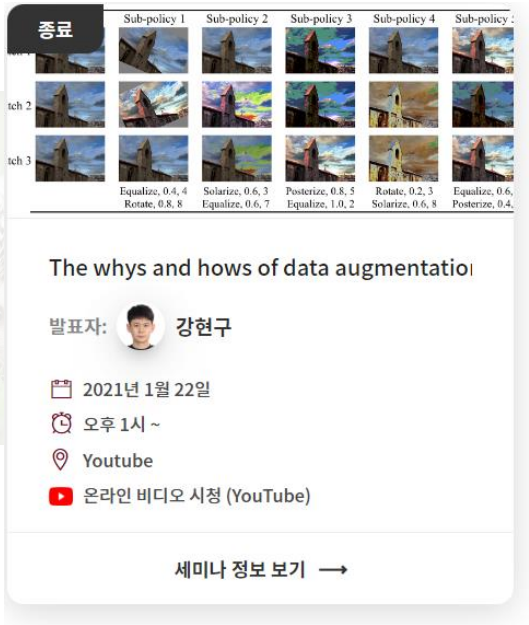



"O" "X"

어떤 변형을, 얼만큼해야 도움이 될까?

Data Augmentation

❖ 주의 : 변형에 의해서 "Label"의 특징을 잃어서는 안됨



[DMQA Open Seminar]

Data Augmentation

https://youtu.be/7fwxcCJsvPw

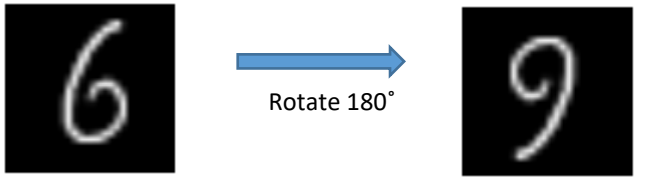
"X"

어떤 변형을, 얼만큼해야 도움이 될까?

Data Augmentation_Domain Knowledge

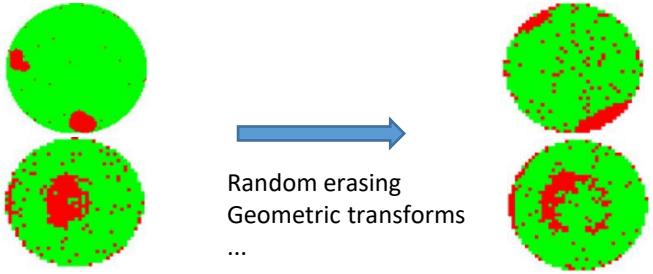
❖ **Augmentation의 적용은 데이터의 도메인에 따라 다름**

- Augmentation 종류 / 정도



"6?"

문자 : 회전 시 문제가 되는 class 존재



전문 Data: 엔지니어 등 분석자의 판단이 필요

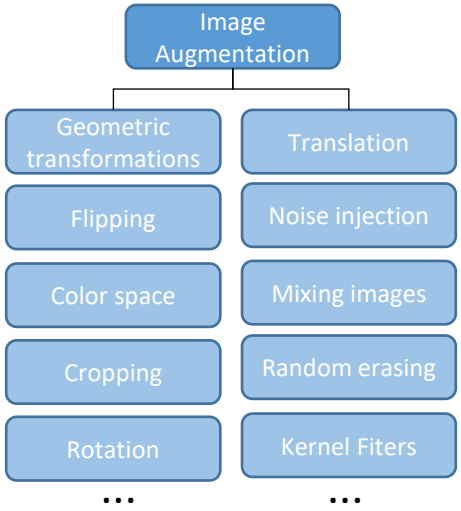
도메인에 알맞은 Augmentation을 적용하는 것은 사전 지식이 필요

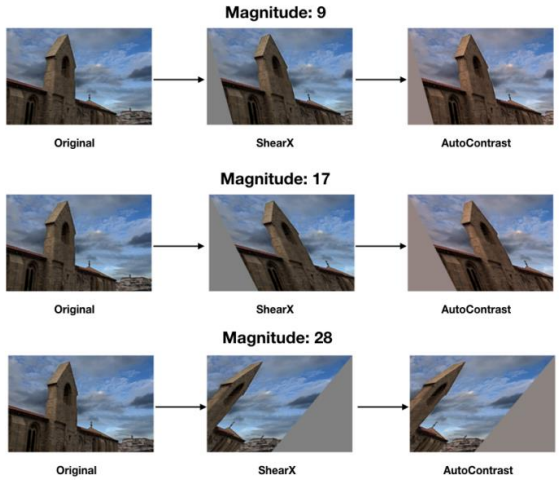
+

분석자의 주관적인 선택으로는 최적의 조건을 찾아내기 어려움

문제 정의

❖ **Optimal Augmentation** : 어떻게 최적의 Augmentation을 판단할 수 있을까?



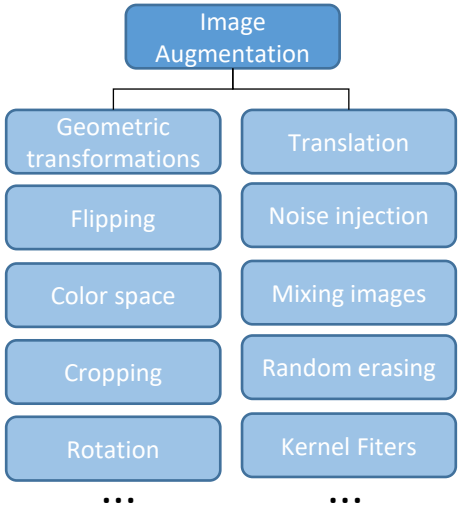


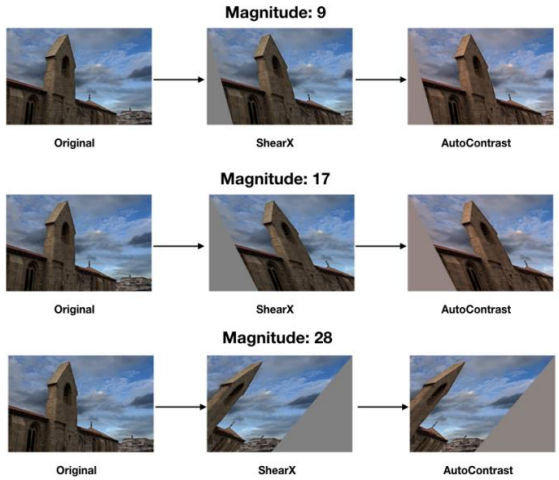
type, probability, magnitude → **Policy**

분석자 : 사전 지식 + 실험 결과를 통해 해석 & 선택

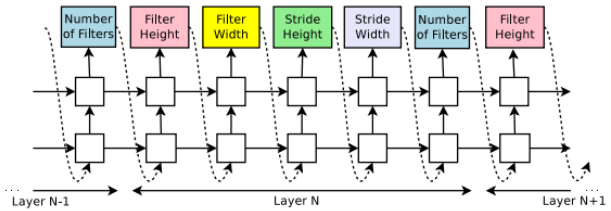
문제 정의

❖ **Optimal Augmentation** : 어떻게 최적의 Augmentation을 판단할 수 있을까?





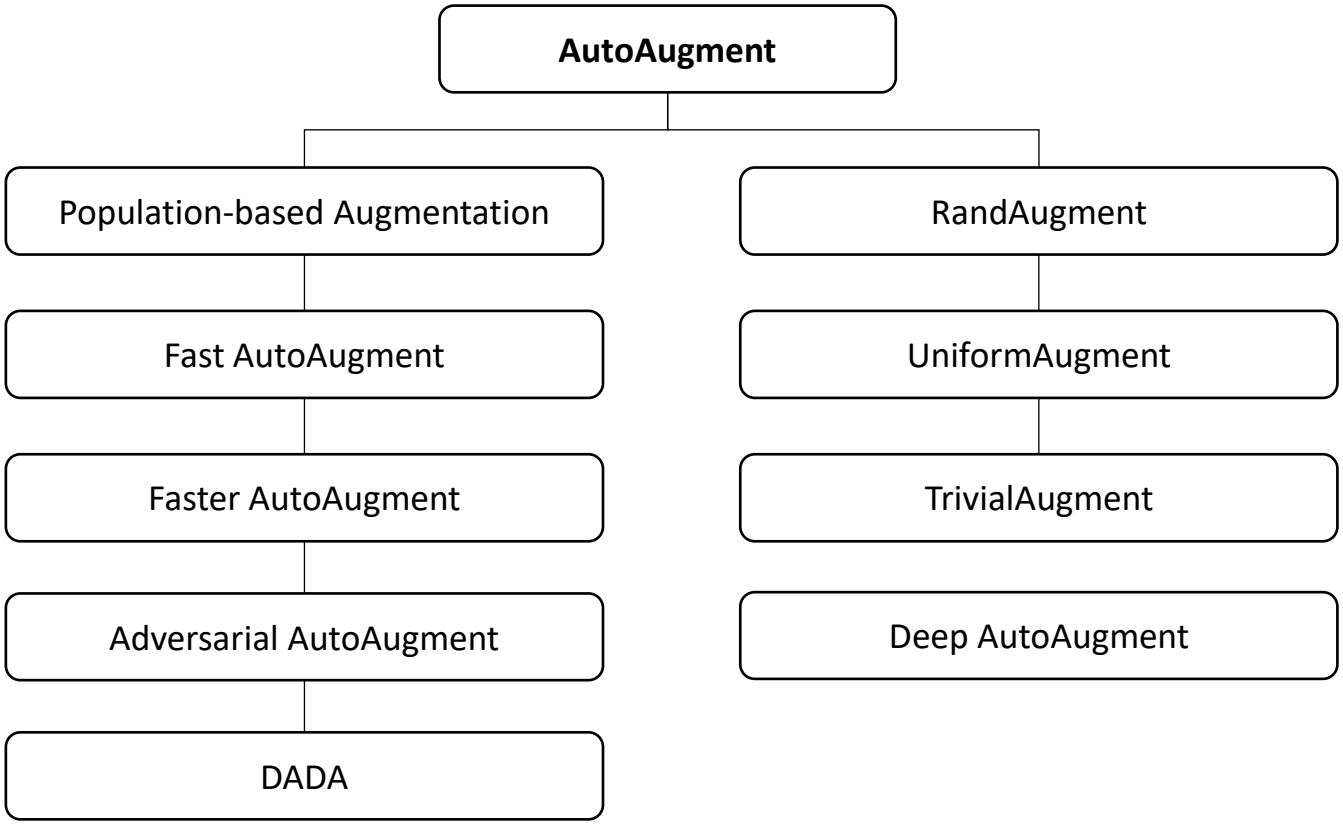
method, probability, magnitude = Policy



모델을 통해 결정하자!
Auto Augmentation

주제

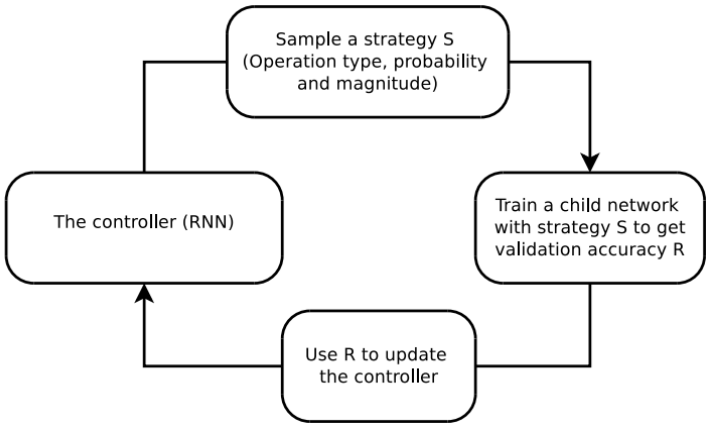
❖ **Finding Optimal by Auto Augmentation** : 모델을 통해 찾아내자!



AutoAugment _Cubuk et al.(2018)

❖ **Augmentation + AutoML**

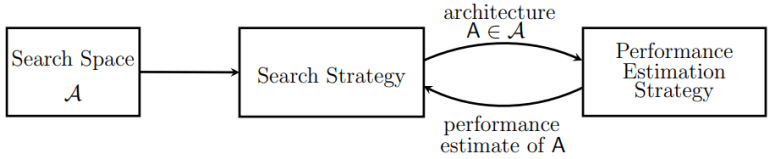
- AutoML을 활용하여, 최적의 Augmentation policy를 찾음

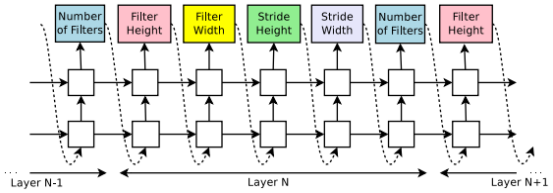


<Overview of framework of AutoAugment search method>

Neural Architecture Search(NAS)

Neural network을 구성할 때,
모델을 통해 최적의 구조를 찾는 방법론

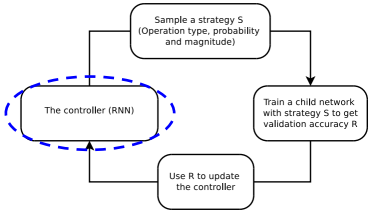




Neural architecture search with reinforcement learning(Zoph & Le, 2017)

Ref) Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2019). Autoaugment: Learning augmentation strategies from data. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 113-123).
Elsken, T., Metzen, J. H., & Hutter, F. (2019). Neural architecture search: A survey. The Journal of Machine Learning Research, 20(1), 1997-2017.

AutoAugment _Cubuk et al.(2018)



❖ AutoAugment

- Augmentation의 적용을 method, probability, magnitude 3가지로 정의(예: [Invert, 0.9, 4])

16 methods

Operation Name	Description	Range of magnitudes
ShearX(Y)	Shear the image along the horizontal (vertical) axis with rate *magnitude*.	[-0.3,0.3]
TranslateX(Y)	Translate the image in the horizontal (vertical) direction by *magnitude* number of pixels.	[-150,150]
Rotate	Rotate the image *magnitude* degrees.	[-30,30]
AutoContrast	Maximize the the image contrast, by making the darkest pixel black and lightest pixel white.	
Invert	Invert the pixels of the image.	
Equalize	Equalize the image histogram.	
Solarize	Invert all pixels above a threshold value of *magnitude*.	[0,256]
Posterize	Reduce the number of bits for each pixel to *magnitude* bits.	[4,8]
Contrast	Control the contrast of the image. A *magnitude*=0 gives a gray image, whereas *magnitude*=1 gives the original image.	[0.1,1.9]
Color	Adjust the color balance of the image, in a manner similar to the controls on a colour TV set. A *magnitude*=0 gives a black & white image, whereas *magnitude*=1 gives the original image.	[0.1,1.9]
Brightness	Adjust the brightness of the image. A *magnitude*=0 gives a black image, whereas *magnitude*=1 gives the original image.	[0.1,1.9]
Sharpness	Adjust the sharpness of the image. A *magnitude*=0 gives a blurred image, whereas *magnitude*=1 gives the original image.	[0.1,1.9]
Cutout [12, 69]	Set a random square patch of side-length *magnitude* pixels to gray.	[0,60]
Sample Pairing [24, 68]	Linearly add the image with another image (selected at random from the same mini-batch) with weight *magnitude*, without changing the label.	[0, 0.4]

<Operation(augmentation) type>

0.1~1.0 (**10** values)

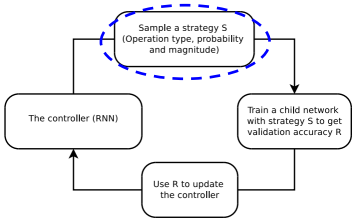
<The probability of applying the operation>

0~10 (**11** values)

<the magnitude of the operation>

Ref) Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2019). Autoaugment: Learning augmentation strategies from data. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 113-123).

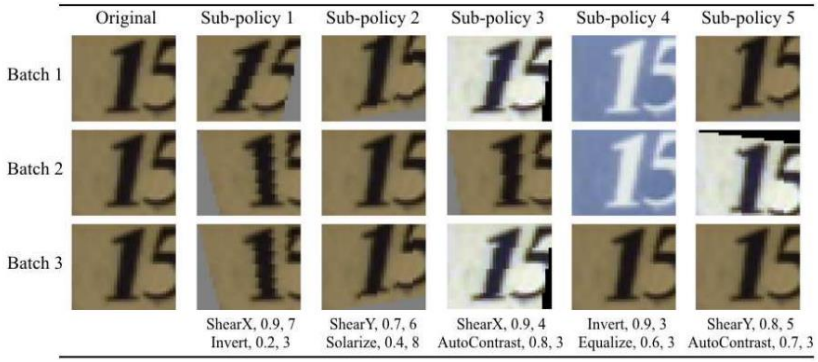
AutoAugment(AA) _Cubuk et al.(2018)



❖ AutoAugment

- Controller(RNN)는 5개의 sub-policy를 가진 Augmentation 조합을 생성

Controller →



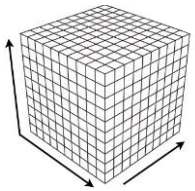
1 Policy =

5개의 Sub-policy

- 연속된 2개 type
- probability
- magnitude

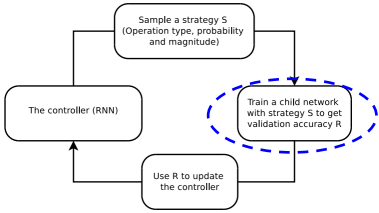
Search Space = $(16 \times 10 \times 11)^{2\times5} \approx 2.9 \times 10^{32}$

(2 type, 5 sub-policy; type, probability, magnitude)



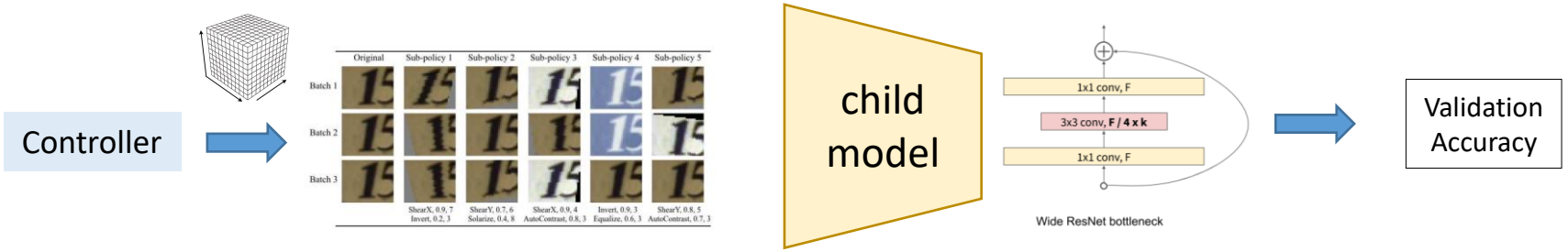
Ref) Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2019). Autoaugment: Learning augmentation strategies from data. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 113-123).

AutoAugment(AA) _Cubuk et al.(2018)



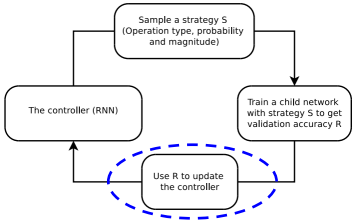
❖ AutoAugment

- 생성된 Augmentation을 child model에 학습시켜서, Validation Accuracy를 산출



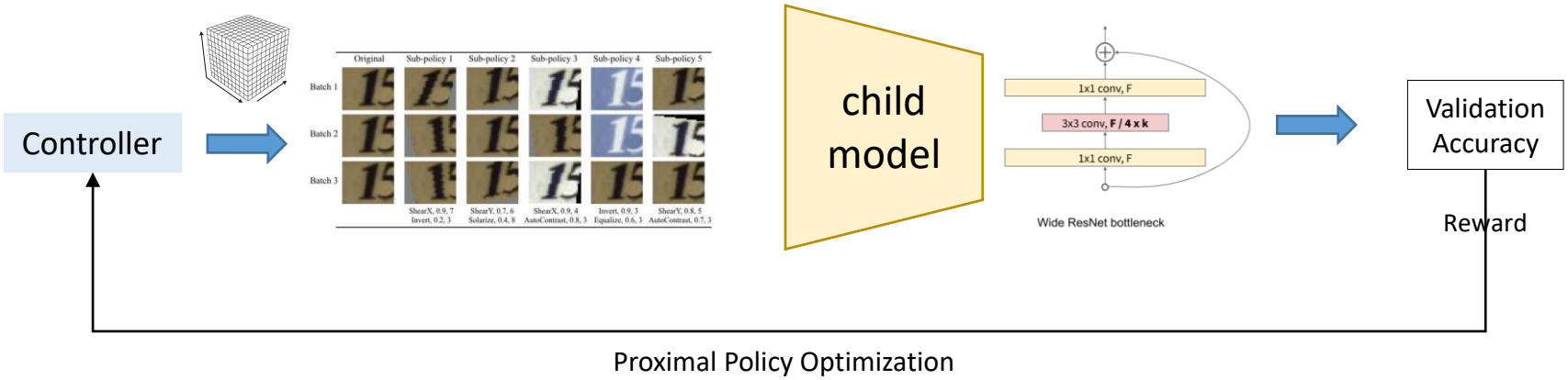
Ref) Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2019). Autoaugment: Learning augmentation strategies from data. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 113-123).

AutoAugment(AA) _Cubuk et al.(2018)



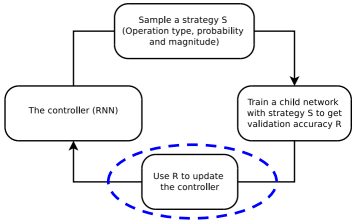
❖ AutoAugment

- Validation Accuracy를 raward로 사용하여 Controller를 최적화



Ref) Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2019). Autoaugment: Learning augmentation strategies from data. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 113-123).

AutoAugment(AA) _Cubuk et al.(2018)



❖ AutoAugment

- 상위 5개 Policy로부터 25개의 Sub-policy를 모아 하나의 최종 policy로 결정

	Operation 1	Operation 2
Sub-policy 0	(Invert,0.1,7)	(Contrast,0.2,6)
Sub-policy 1	(Rotate,0.7,2)	(TranslateX,0.3,9)
Sub-policy 2	(Sharpness,0.8,1)	(Sharpness,0.9,3)
Sub-policy 3	(ShearY,0.5,8)	(TranslateY,0.7,9)
Sub-policy 4	(AutoContrast,0.5,8)	(Equalize,0.9,2)
Sub-policy 5	(ShearY,0.2,7)	(Posterize,0.3,7)
Sub-policy 6	(Color,0.4,3)	(Brightness,0.6,7)
Sub-policy 7	(Sharpness,0.3,9)	(Brightness,0.7,9)
Sub-policy 8	(Equalize,0.6,5)	(Equalize,0.5,1)
Sub-policy 9	(Contrast,0.6,7)	(Sharpness,0.6,5)
Sub-policy 10	(Color,0.7,7)	(TranslateX,0.5,8)
Sub-policy 11	(Equalize,0.3,7)	(AutoContrast,0.4,8)
Sub-policy 12	(TranslateY,0.4,3)	(Sharpness,0.2,6)
Sub-policy 13	(Brightness,0.9,6)	(Color,0.2,8)
Sub-policy 14	(Solarize,0.5,2)	(Invert,0.0,3)
Sub-policy 15	(Equalize,0.2,0)	(AutoContrast,0.6,0)
Sub-policy 16	(Equalize,0.2,8)	(Equalize,0.6,4)
Sub-policy 17	(Color,0.9,9)	(Equalize,0.6,6)
Sub-policy 18	(AutoContrast,0.8,4)	(Solarize,0.2,8)
Sub-policy 19	(Brightness,0.1,3)	(Color,0.7,0)
Sub-policy 20	(Solarize,0.4,5)	(AutoContrast,0.9,3)
Sub-policy 21	(TranslateY,0.9,9)	(TranslateY,0.7,9)
Sub-policy 22	(AutoContrast,0.9,2)	(Solarize,0.8,3)
Sub-policy 23	(Equalize,0.8,8)	(Invert,0.1,3)
Sub-policy 24	(TranslateY,0.7,9)	(AutoContrast,0.9,1)

Table 7. AutoAugment policy found on reduced CIFAR-10.

CIFAR-10 final policy

	Operation 1	Operation 2
Sub-policy 0	(ShearX,0.9,4)	(Invert,0.2,3)
Sub-policy 1	(ShearY,0.9,8)	(Invert,0.7,5)
Sub-policy 2	(Equalize,0.6,5)	(Solarize,0.6,6)
Sub-policy 3	(Invert,0.9,3)	(Equalize,0.6,3)
Sub-policy 4	(Equalize,0.6,1)	(Rotate,0.9,3)
Sub-policy 5	(ShearX,0.9,4)	(AutoContrast,0.8,3)
Sub-policy 6	(ShearY,0.9,8)	(Invert,0.4,5)
Sub-policy 7	(ShearY,0.9,5)	(Solarize,0.2,6)
Sub-policy 8	(Invert,0.9,6)	(AutoContrast,0.8,1)
Sub-policy 9	(Equalize,0.6,3)	(Rotate,0.9,3)
Sub-policy 10	(ShearX,0.9,4)	(Solarize,0.3,3)
Sub-policy 11	(ShearY,0.8,8)	(Invert,0.7,4)
Sub-policy 12	(Equalize,0.9,5)	(TranslateY,0.6,6)
Sub-policy 13	(Invert,0.9,4)	(Equalize,0.6,7)
Sub-policy 14	(Contrast,0.3,3)	(Rotate,0.8,4)
Sub-policy 15	(Invert,0.8,5)	(TranslateY,0.0,2)
Sub-policy 16	(ShearY,0.7,6)	(Solarize,0.4,8)
Sub-policy 17	(Invert,0.6,4)	(Rotate,0.8,4)
Sub-policy 18	(ShearY,0.3,7)	(TranslateX,0.9,3)
Sub-policy 19	(ShearX,0.1,6)	(Invert,0.6,5)
Sub-policy 20	(Solarize,0.7,2)	(TranslateY,0.6,7)
Sub-policy 21	(ShearY,0.8,4)	(Invert,0.8,8)
Sub-policy 22	(ShearX,0.7,9)	(TranslateY,0.8,3)
Sub-policy 23	(ShearY,0.8,5)	(AutoContrast,0.7,3)
Sub-policy 24	(ShearX,0.7,2)	(Invert,0.1,5)

Table 8. AutoAugment policy found on reduced SVHN.

SVHN final policy

Ref) Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2019). Autoaugment: Learning augmentation strategies from data. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 113-123).

AutoAugment(AA) _Cubuk et al.(2018)

❖ 성능 결과

- State-of-the-art 달성

Dataset	Model	Baseline	Cutout [12]	AutoAugment
CIFAR-10	Wide-ResNet-28-10 [67]	3.9	3.1	2.6±0.1
	Shake-Shake (26 2x32d) [17]	3.6	3.0	2.5±0.1
	Shake-Shake (26 2x96d) [17]	2.9	2.6	2.0±0.1
	Shake-Shake (26 2x112d) [17]	2.8	2.6	1.9±0.1
	AmoebaNet-B (6,128) [48]	3.0	2.1	1.8±0.1
	PyramidNet+ShakeDrop [65]	2.7	2.3	**1.5 ± 0.1**
Reduced CIFAR-10	Wide-ResNet-28-10 [67]	18.8	16.5	14.1±0.3
	Shake-Shake (26 2x96d) [17]	17.1	13.4	**10.0 ± 0.2**
CIFAR-100	Wide-ResNet-28-10 [67]	18.8	18.4	17.1±0.3
	Shake-Shake (26 2x96d) [17]	17.1	16.0	14.3±0.2
	PyramidNet+ShakeDrop [65]	14.0	12.2	**10.7 ± 0.2**
SVHN	Wide-ResNet-28-10 [67]	1.5	1.3	1.1
	Shake-Shake (26 2x96d) [17]	1.4	1.2	**1.0**
Reduced SVHN	Wide-ResNet-28-10 [67]	13.2	32.5	8.2
	Shake-Shake (26 2x96d) [17]	12.3	24.2	**5.9**

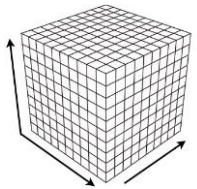
Ref) Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2019). Autoaugment: Learning augmentation strategies from data. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 113-123).

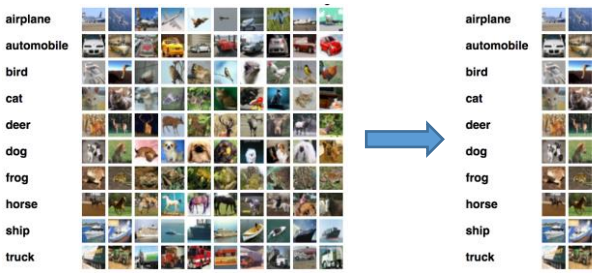
AutoAugment(AA) _Cubuk et al.(2018)

❖ 한계점

- 너무도 큰 Search space에 의한 연산량

Search Space = $(16 \times 10 \times 11)^{2\times5} \approx 2.9 \times 10^{32}$





50000 → 4000 samples

reduced dataset을 사용하여 AutoAugment policy 탐색

	Operation 1	Operation 2
Sub-policy 0	(Posterize,0.4,8)	(Rotate,0.6,9)
Sub-policy 1	(Solarize,0.6,5)	(AutoContrast,0.6,5)
Sub-policy 2	(Equalize,0.8,8)	(Equalize,0.6,3)
Sub-policy 3	(Posterize,0.6,7)	(Posterize,0.6,6)
Sub-policy 4	(Equalize,0.4,7)	(Solarize,0.2,4)
Sub-policy 5	(Equalize,0.4,4)	(Rotate,0.8,8)
Sub-policy 6	(Solarize,0.6,3)	(Equalize,0.6,7)
Sub-policy 7	(Posterize,0.8,5)	(Equalize,1.0,2)
Sub-policy 8	(Rotate,0.2,3)	(Solarize,0.6,8)
Sub-policy 9	(Equalize,0.6,8)	(Posterize,0.4,6)
Sub-policy 10	(Rotate,0.8,8)	(Color,0.4,0)
Sub-policy 11	(Rotate,0.4,9)	(Equalize,0.6,2)
Sub-policy 12	(Equalize,0.0,7)	(Equalize,0.8,8)
Sub-policy 13	(Invert,0.6,4)	(Equalize,1.0,8)
Sub-policy 14	(Color,0.6,4)	(Contrast,1.0,8)
Sub-policy 15	(Rotate,0.8,8)	(Color,1.0,2)
Sub-policy 16	(Color,0.8,8)	(Solarize,0.8,7)
Sub-policy 17	(Sharpness,0.4,7)	(Invert,0.6,8)
Sub-policy 18	(ShearX,0.6,5)	(Equalize,1.0,9)
Sub-policy 19	(Color,0.4,0)	(Equalize,0.6,3)
Sub-policy 20	(Equalize,0.4,7)	(Solarize,0.2,4)
Sub-policy 21	(Solarize,0.6,5)	(AutoContrast,0.6,5)
Sub-policy 22	(Invert,0.6,4)	(Equalize,1.0,8)
Sub-policy 23	(Color,0.6,4)	(Contrast,1.0,8)
Sub-policy 24	(Equalize,0.8,8)	(Equalize,0.6,3)

Table 9. AutoAugment policy found on reduced ImageNet.

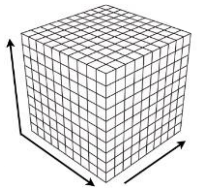
ImageNet으로 구한 Policy를 다른 도메인에 사용

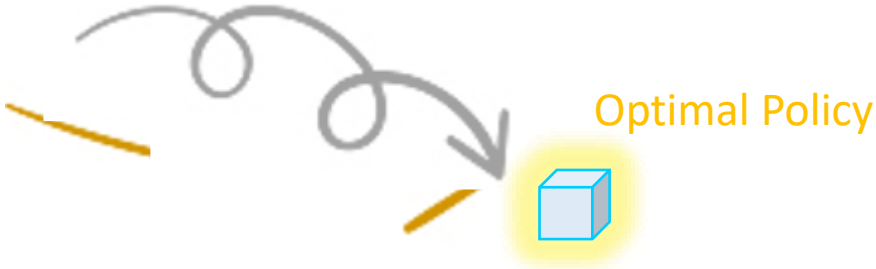
Ref) Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2019). Autoaugment: Learning augmentation strategies from data. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 113-123).

최적화 알고리즘을 개선하는 방향

❖ **Optimal policy를 효과적으로 찾기위한 algorithm**

Search Space = $(16 \times 10 \times 11)^{2\times5} \approx 2.9 \times 10^{32}$





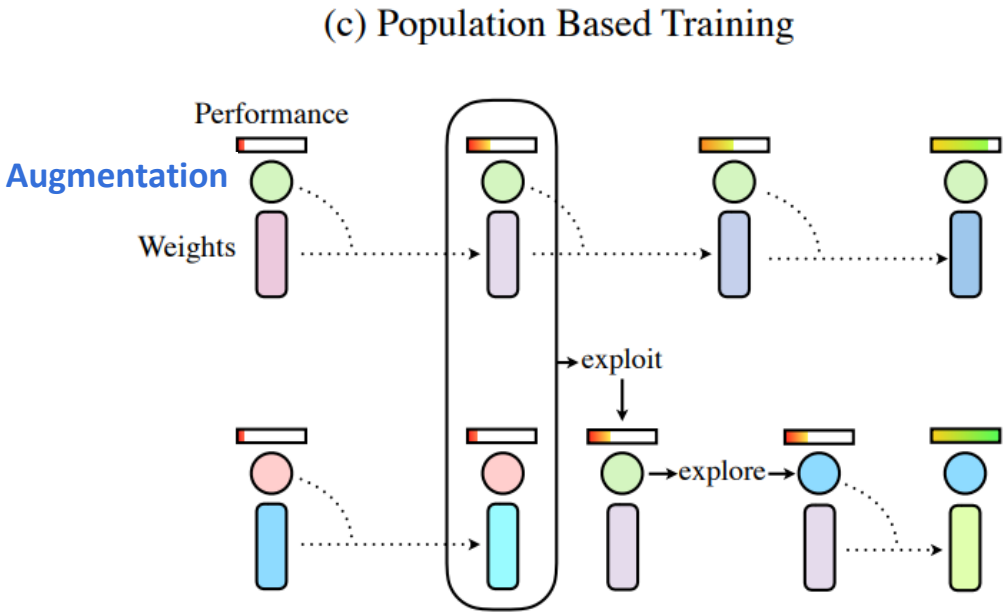
연산량↓, 빠르게
(최적화 알고리즘을 바꿔보자!)

Population-based Augmentation(PBA) _Ho et al.(2019)

❖ **Concept**

- Population Based Training(PBT) method를 적용하여 최적의 Augmentation을 찾음

Too heavy!

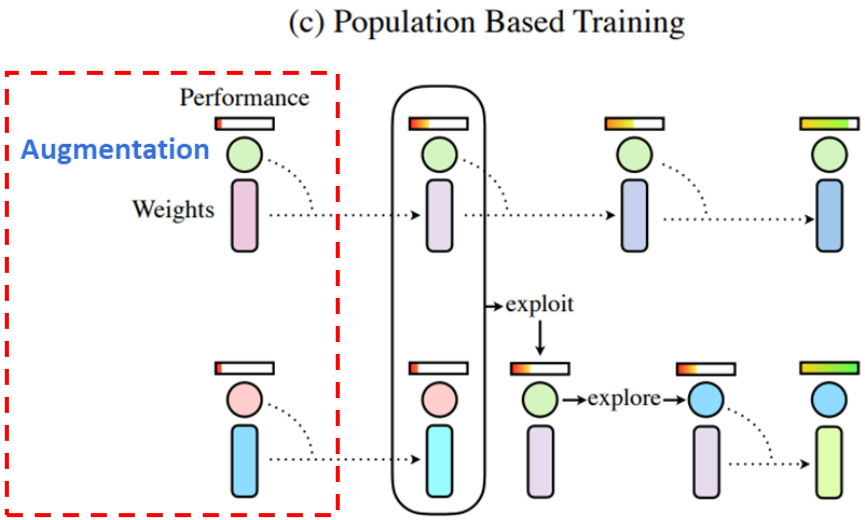


Ref) Jaderberg, M., Dalibard, V., Osindero, S., Czarnecki, W. M., Donahue, J., Razavi, A., ... & Kavukcuoglu, K. (2017). Population based training of neural networks. arXiv preprint arXiv:1711.09846.

Population-based Augmentation(PBA) _Ho et al.(2019)

❖ Method

- Population Based Training(PBT) method를 적용하여 최적의 Augmentation을 찾음



Augmentation의 종류에 맞춰 16개 모델 학습

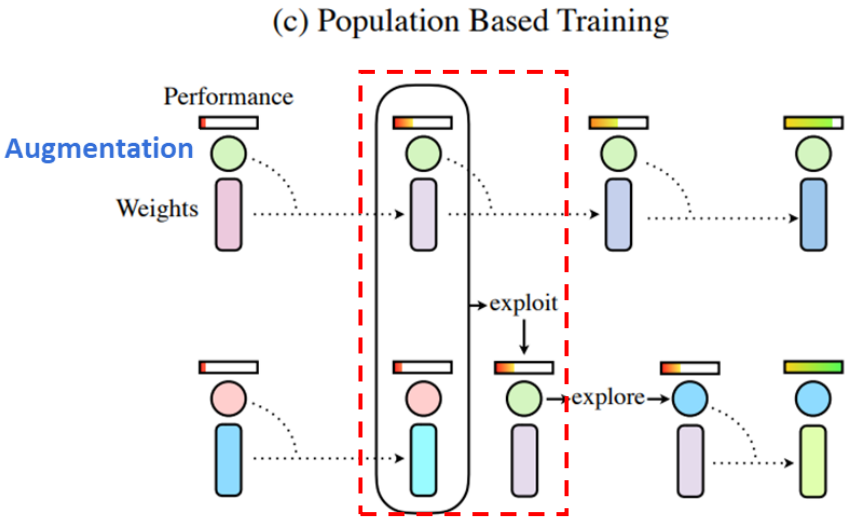
1. 여러개의 동일한 모델을
   다른 Augmentation을 적용하여 동시에 학습

Ref) Ho, D., Liang, E., Chen, X., Stoica, I., & Abbeel, P. (2019, May).
Population based augmentation: Efficient learning of augmentation policy schedules.
In International Conference on Machine Learning (pp. 2731-2741). PMLR.

Population-based Augmentation(PBA) _Ho et al.(2019)

❖ Method

- Population Based Training(PBT) method를 적용하여 최적의 Augmentation을 찾음



1. 여러개의 동일한 모델을
   다른 Augmentation을 적용하여 동시에 학습

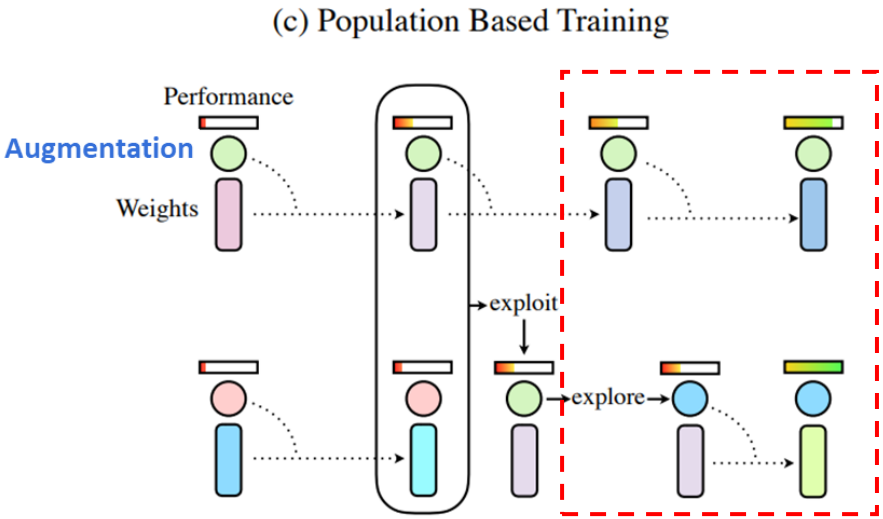
2. 학습 중간에 각 모델의 성능을 비교
   → 높은 성능의 모델 parameter를 다른 모델에 복제
   (하위 25% → 상위 25%에 있는 모델 weight을 복제)

Ref) Ho, D., Liang, E., Chen, X., Stoica, I., & Abbeel, P. (2019, May).
Population based augmentation: Efficient learning of augmentation policy schedules.
In International Conference on Machine Learning (pp. 2731-2741). PMLR.

Population-based Augmentation(PBA) _Ho et al.(2019)

❖ Method

- Population Based Training(PBT) method를 적용하여 최적의 Augmentation을 찾음



1. 여러개의 동일한 모델을
   다른 Augmentation을 적용하여 동시에 학습

2. 학습 중간에 각 모델의 성능을 비교
   → 높은 성능의 모델 parameter를 다른 모델에 복제
   (하위 25% → 상위 25%에 있는 모델weight을 복제)

3. 적용된 Augmentation에 변형을 주어 재탐색

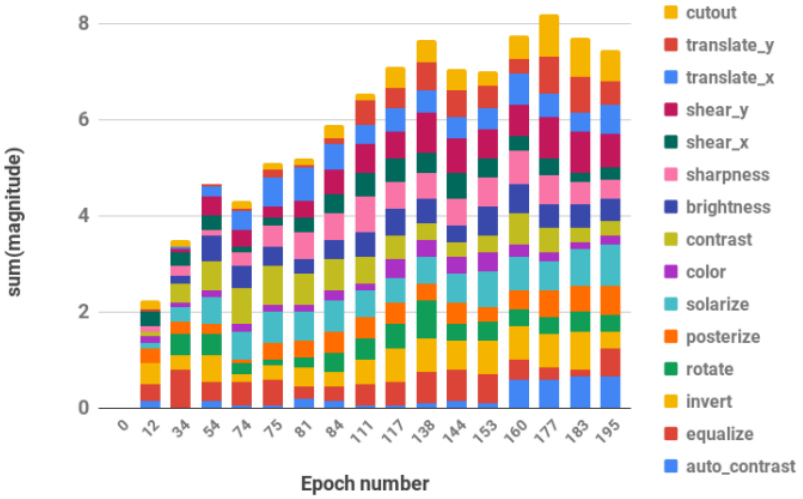
4. 위 과정을 반복하며 학습을 완료

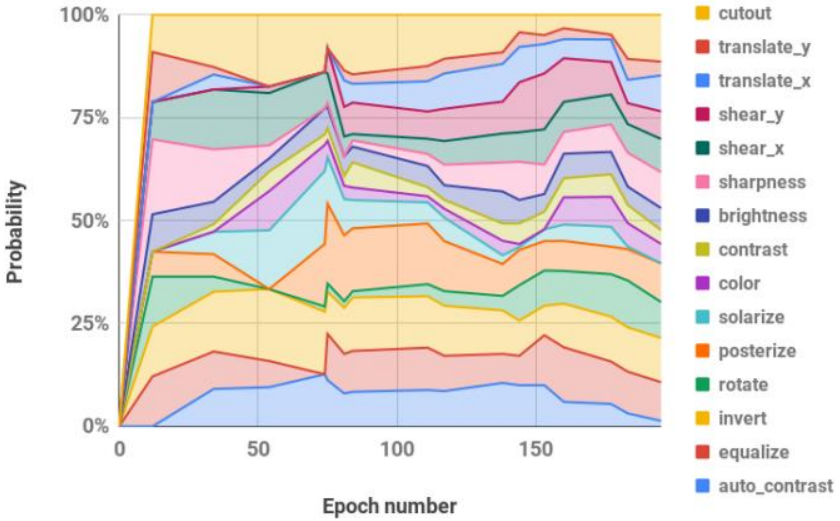
Ref) Ho, D., Liang, E., Chen, X., Stoica, I., & Abbeel, P. (2019, May).
Population based augmentation: Efficient learning of augmentation policy schedules.
In International Conference on Machine Learning (pp. 2731-2741). PMLR.

Population-based Augmentation(PBA) _Ho et al.(2019)

❖ **실험 결과**

- Augmentation magnitude가 130 epoch에서 수렴하며 결정됨
- Probability를 보았을 때, 모든 augmentation이 적당히 섞이면서도, 특정 방법은 강조됨(cutout 등)



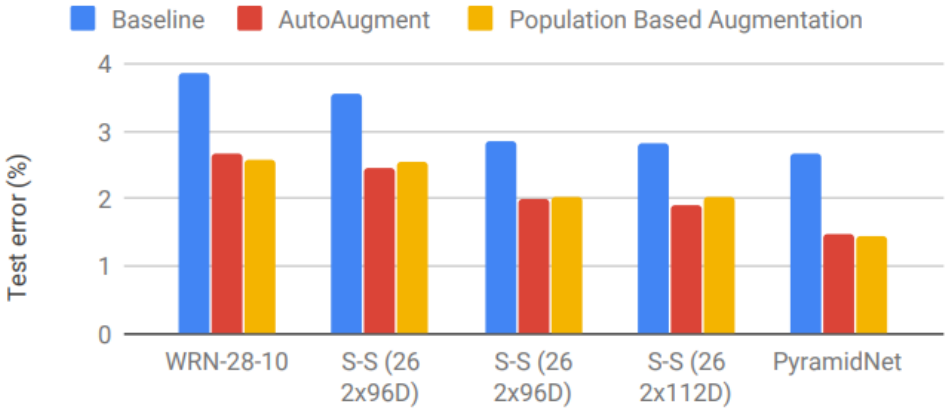


Ref) Ho, D., Liang, E., Chen, X., Stoica, I., & Abbeel, P. (2019, May).
Population based augmentation: Efficient learning of augmentation policy schedules.
In International Conference on Machine Learning (pp. 2731-2741). PMLR.

Population-based Augmentation(PBA) _Ho et al.(2019)

❖ **실험 결과**

- 준수한 성능(AA와 유사한 수준 & SOTA)
- 연산량이 1/1000배 수준으로 감소



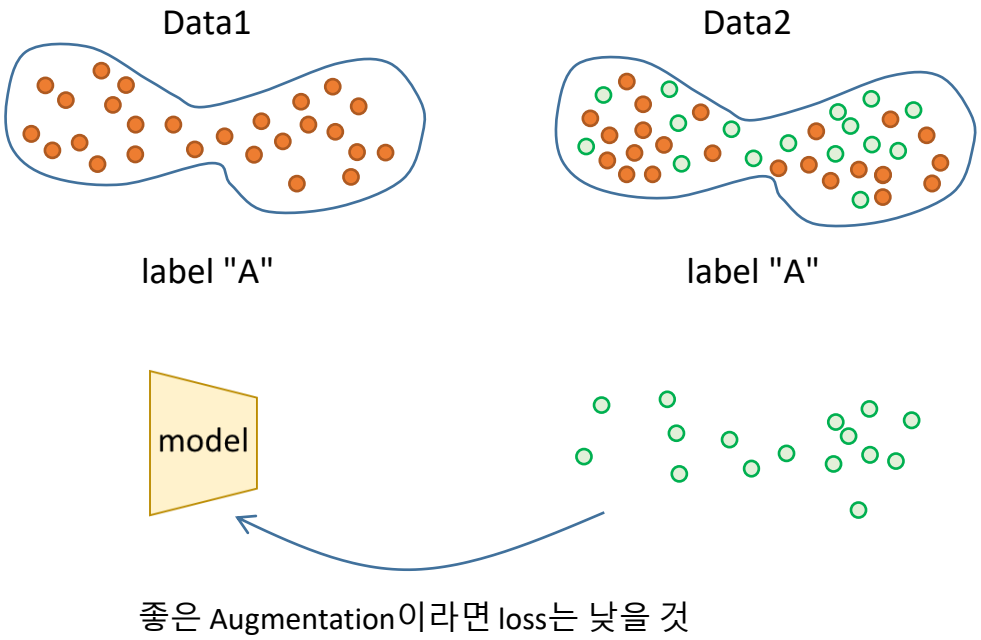
Dataset	Value	Previous Best	AA	PBA
CIFAR-10	GPU Hours	-	5000	5
	Test Error	2.1	1.48	1.46
CIFAR-100	GPU Hours	-	0*	0*
	Test Error	12.2	10.7	10.9
SVHN	GPU Hours	-	1000	1
	Test Error	1.3	1.0	1.1

Ref) Ho, D., Liang, E., Chen, X., Stoica, I., & Abbeel, P. (2019, May).
Population based augmentation: Efficient learning of augmentation policy schedules.
In International Conference on Machine Learning (pp. 2731-2741). PMLR.

Fast AutoAugment(FastAA) _Lim et al.(2019)

❖ Concept

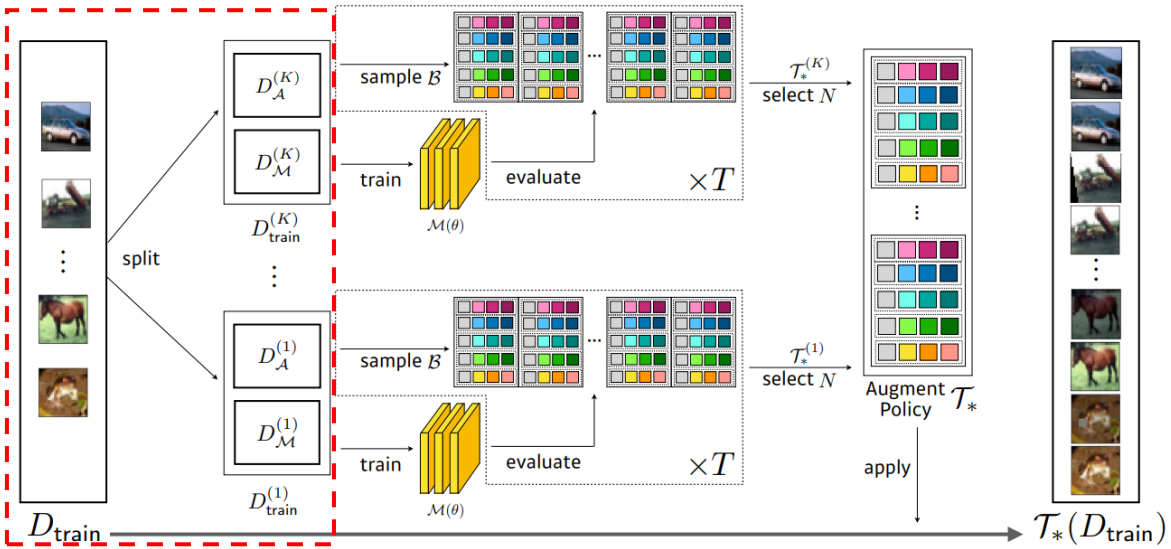
- Augmentation은 missing data points를 채워서, train data와 valid data의 "분포를 맞춰주는" 역할



Ref) Lim, S., Kim, I., Kim, T., Kim, C., & Kim, S. (2019). Fast autoaugment. Advances in Neural Information Processing Systems, 32.

Fast AutoAugment(FastAA) _Lim et al.(2019)

❖ **Tree-structured Parzen Estimator (TPE) algorithm :** Bayesian Optimization

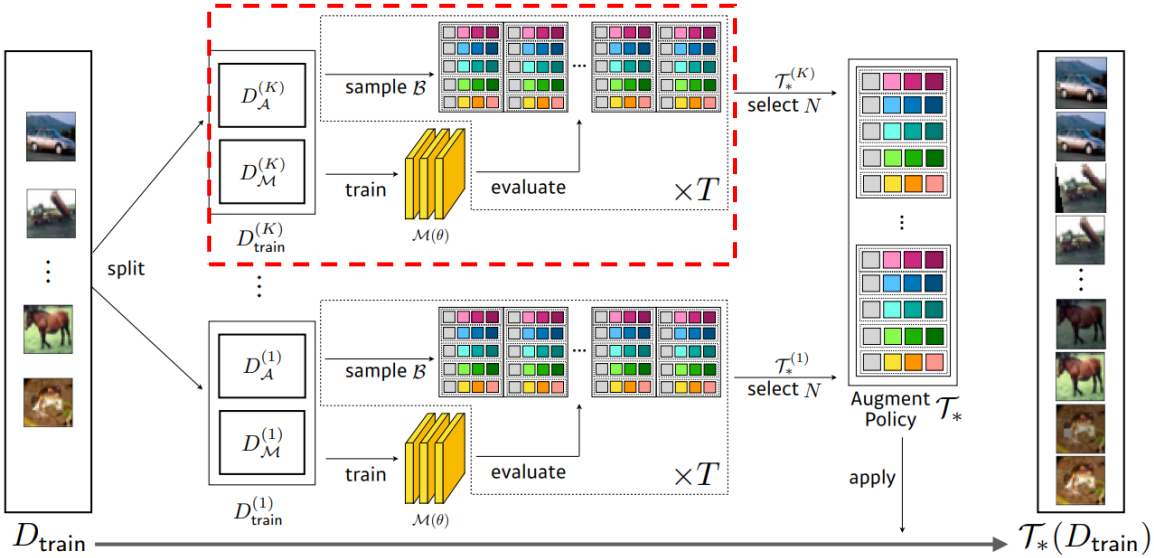


1. Train 데이터를 class 비율을 맞추어 K개로 나누고, 각 묶음은 D_M , D_A로 나눔
   - K = 5로 실험 진행

Ref) Lim, S., Kim, I., Kim, T., Kim, C., & Kim, S. (2019). Fast autoaugment. Advances in Neural Information Processing Systems, 32.

Fast AutoAugment(FastAA) _Lim et al.(2019)

❖ **Tree-structured Parzen Estimator (TPE) algorithm :** Bayesian Optimization



Algorithm 1: Fast AutoAugment

Input : $(\theta, D_{\text{train}}, K, T, B, N)$

Split D_{train} into K-fold data $D_{\text{train}}^{(k)} = \{(D_{\mathcal{M}}^{(k)}, D_{\mathcal{A}}^{(k)})\}$ // stratified shuffling

for $k \in \{1, \ldots, K\}$ **do**

  $\mathcal{T}_*^{(k)} \leftarrow \emptyset$, $(D_{\mathcal{M}}, D_{\mathcal{A}}) \leftarrow (D_{\mathcal{M}}^{(k)}, D_{\mathcal{A}}^{(k)})$ // initialize

  Train θ on $D_{\mathcal{M}}$

  for $t \in \{0, \ldots, T-1\}$ **do**

    $\mathcal{B} \leftarrow \text{BayesOptim}(\mathcal{T}, \mathcal{L}(\theta|\mathcal{T}(D_{\mathcal{A}})), B)$ // explore-and-exploit

    $\mathcal{T}_t \leftarrow$ Select top-N policies in $\mathcal{B}$

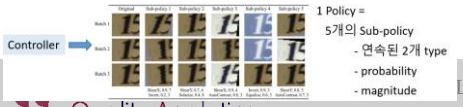
    $\mathcal{T}_*^{(k)} \leftarrow \mathcal{T}_*^{(k)} \cup \mathcal{T}_t$ // merge augmentation policies

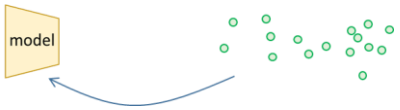
return $\mathcal{T}_* = \bigcup_k \mathcal{T}_*^{(k)}$

2. D_M으로만 model을 학습하고,D_A는 Augmentation Policy를 탐색

- policy에 의한 sample별 모델의 성능결과로 Bayesian Optimization 진행
- 성능이 좋은 N개 policy를 병합하여 하나의 policy($\mathcal{T}_*^K$)결정
- T번 반복
- N=5, T=2로 실험 진행

$$\text{EI}(\mathcal{T}) = \mathbb{E}[\min(\mathcal{L}(\theta|\mathcal{T}(D_{\mathcal{A}})) - \mathcal{L}^{\dagger}, 0)]$$

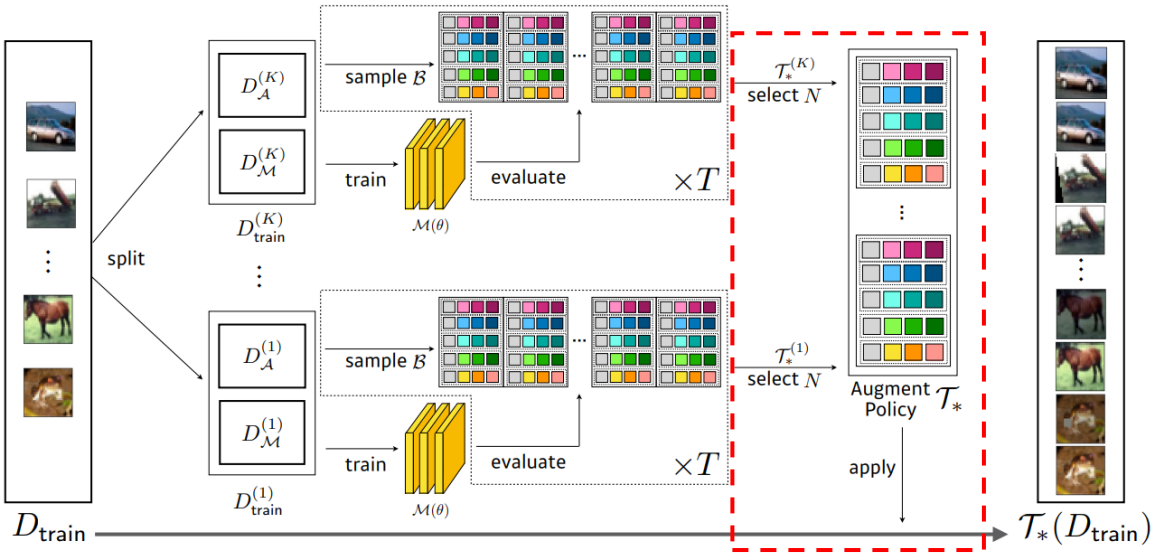




Lim, S., Kim, I., Kim, T., Kim, C., & Kim, S. (2019). Fast autoaugment. Advances in Neural Information Processing Systems, 32.

Fast AutoAugment(FastAA) _Lim et al.(2019)

❖ **Tree-structured Parzen Estimator (TPE) algorithm :** Bayesian Optimization



2. 각 묶음에서 구한 policy를 최종 Policy로 병합하여 Augmentation 진행

Ref) Lim, S., Kim, I., Kim, T., Kim, C., & Kim, S. (2019). Fast autoaugment. Advances in Neural Information Processing Systems, 32.

Fast AutoAugment(FastAA) _Lim et al.(2019)

❖ 성능 : AA, PBA와 동일 수준, 연산량(*GPU hours)에서 큰 효과

Model	Baseline	Cutout [5]	AA [3]	PBA [13]	Fast AA (transfer / direct)
Wide-ResNet-40-2	5.3	4.1	3.7	–	**3.6** / 3.7
Wide-ResNet-28-10	3.9	3.1	**2.6**	**2.6**	2.7 / 2.7
Shake-Shake(26 2×32d)	3.6	3.0	**2.5**	**2.5**	2.7 / **2.5**
Shake-Shake(26 2×96d)	2.9	2.6	**2.0**	**2.0**	**2.0** / **2.0**
Shake-Shake(26 2×112d)	2.8	2.6	**1.9**	2.0	2.0 / **1.9**
PyramidNet+ShakeDrop	2.7	2.3	**1.5**	**1.5**	1.8 / 1.7

Table 2: Test set error rate (%) on CIFAR-10.

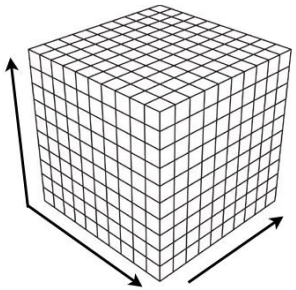
Dataset	AutoAugment [3]	Fast AutoAugment
CIFAR-10	5000	3.5
SVHN	1000	1.5
ImageNet	15000	450

Ref) Lim, S., Kim, I., Kim, T., Kim, C., & Kim, S. (2019). Fast autoaugment. Advances in Neural Information Processing Systems, 32.

Search space를 개선하는 방향

❖ 굳이 종류,확률,강도를 세부적으로 나누어서 찾아야할까?

Search Space = $(16 \times 10 \times 11)^{2\times5} \approx 2.9 \times 10^{32}$



Search space를 줄여보자!

RandAugment(RA) _Cubuk et al.(2019)

❖ **Concept : Augmentation 별로 probability, magnitude를 따로 설정하지 않고 hyperparameter를 통합**

AutoAugment

16 methods

Operation Name	Description	Range of magnitudes
ShearX(Y)	Shear the image along the horizontal (vertical) axis with rate *magnitude*.	[-0.3,0.3]
TranslateX(Y)	Translate the image in the horizontal (vertical) direction by *magnitude* number of pixels.	[-150,150]
Rotate	Rotate the image *magnitude* degrees.	[-30,30]
AutoContrast	Maximize the the image contrast, by making the darkest pixel black and lightest pixel white.	
Invert	Invert the pixels of the image.	
Equalize	Equalize the image histogram.	
Solarize	Invert all pixels above a threshold value of *magnitude*.	[0,256]
Posterize	Reduce the number of bits for each pixel to *magnitude* bits.	[4,8]
Contrast	Control the contrast of the image. A *magnitude*=0 gives a gray image, whereas *magnitude*=1 gives the original image.	[0.1,1.9]
Color	Adjust the color balance of the image, in a manner similar to the controls on a colour TV set. A *magnitude*=0 gives a black & white image, whereas *magnitude*=1 gives the original image.	[0.1,1.9]
Brightness	Adjust the brightness of the image. A *magnitude*=0 gives a black image, whereas *magnitude*=1 gives the original image.	[0.1,1.9]
Sharpness	Adjust the sharpness of the image. A *magnitude*=0 gives a blurred image, whereas *magnitude*=1 gives the original image.	[0.1,1.9]
Cutout [12, 69]	Set a random square patch of side-length *magnitude* pixels to gray.	[0,60]
Sample Pairing [24, 68]	Linearly add the image with another image (selected at random from the same mini-batch) with weight *magnitude*, without changing the label.	[0, 0.4]

<Operation(augmentation) type>

0.1~1.0 (**10** values)

<The probability of applying the operation>

0~10 (**11** values)

<the magnitude of the operation>

Search Space = $(16 \times 10 \times 11)^{2\times5} \approx 2.9 \times 10^{32}$

Ref) Cubuk, E. D., Zoph, B., Shlens, J., & Le, Q. V. (2020). Randaugment: Practical automated data augmentation with a reduced search space. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops (pp. 702-703).in Neural Information Processing Systems, 32.

RandAugment(RA) _Cubuk et al.(2019)

❖ **Concept : Augmentation 별로 probability, magnitude를 따로 설정하지 않고 hyperparameter를 통합**

14 methods

- identity
- rotate
- posterize
- sharpness
- translate-x
- autoContrast
- solarize
- contrast
- shear-x
- translate-y
- equalize
- color
- brightness
- shear-y

<Operation(augmentation) type>

$\frac{1}{14}$ (K=14) **1** value

<The probability of applying the operation>

N값만 결정
(몇가지 사용할것인지)

0~30 (30 values)

<the magnitude of the operation>

M 값 통일
(Type 상관없이
magnitude 통일 적용)

Search Space = $(16 \times 10 \times 11)^{2\times5} \approx 2.9 \times 10^{32}$ → $N \times M \approx 10^2$

최적화 grid search만으로 충분

Ref) Cubuk, E. D., Zoph, B., Shlens, J., & Le, Q. V. (2020). Randaugment: Practical automated data augmentation with a reduced search space. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops (pp. 702-703).in Neural Information Processing Systems, 32.

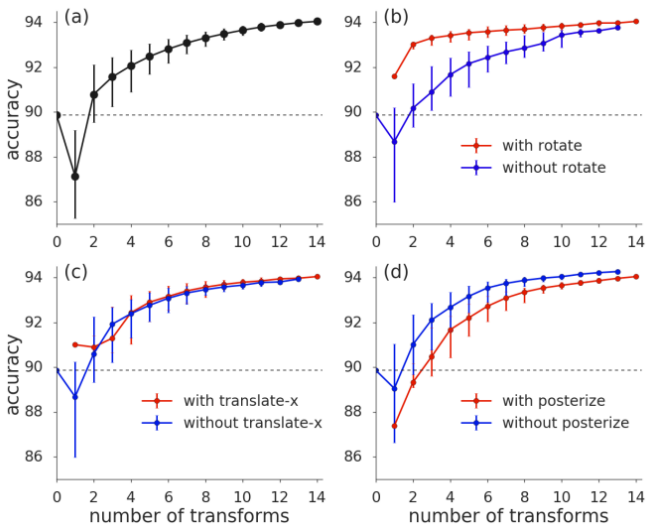
RandAugment(RA) _Cubuk et al.(2019)

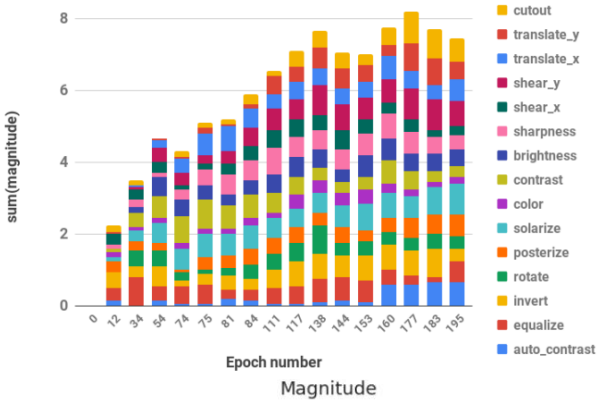
❖ **Concept : 모든 Augmentation을 동일하게 적용**

N=2, M=9

Random한 Augmentaion 2가지를
강도 9로 통일 적용하여 input으로 학습

model





Magnitude Method	Accuracy
Random Magnitude	97.3
Constant Magnitude	97.2
Linearly Increasing Magnitude	97.2
Random Magnitude with Increasing Upper Bound	97.3

Augmentation간의 유의차는 있으나 개수가 많아지면 비슷한 성능을 보임

Augmentation별로 magnitude를 변경해도 큰 차이가 없고,
Constant로 해도 변경시켜주는 것과 유사한 성능을 얻음

Ref) Cubuk, E. D., Zoph, B., Shlens, J., & Le, Q. V. (2020). Randaugment: Practical automated data augmentation with a reduced search space. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops (pp. 702-703).in Neural Information Processing Systems, 32.

RandAugment(RA) _Cubuk et al.(2019)

❖ 실험 결과 : 매우 간단한 코드, 준수한 성능

	baseline	PBA	Fast AA	AA	RA
CIFAR-10					
Wide-ResNet-28-2	94.9	-	-	**95.9**	95.8
Wide-ResNet-28-10	96.1	**97.4**	97.3	**97.4**	97.3
Shake-Shake	97.1	**98.0**	**98.0**	**98.0**	**98.0**
PyramidNet	97.3	**98.5**	98.3	**98.5**	**98.5**
CIFAR-100					
Wide-ResNet-28-2	75.4	-	-	**78.5**	78.3
Wide-ResNet-28-10	81.2	**83.3**	82.7	82.9	**83.3**
SVHN (core set)					
Wide-ResNet-28-2	96.7	-	-	98.0	**98.3**
Wide-ResNet-28-10	96.9	-	-	98.1	**98.3**
SVHN					
Wide-ResNet-28-2	98.2	-	-	**98.7**	**98.7**
Wide-ResNet-28-10	98.5	98.9	98.8	98.9	**99.0**

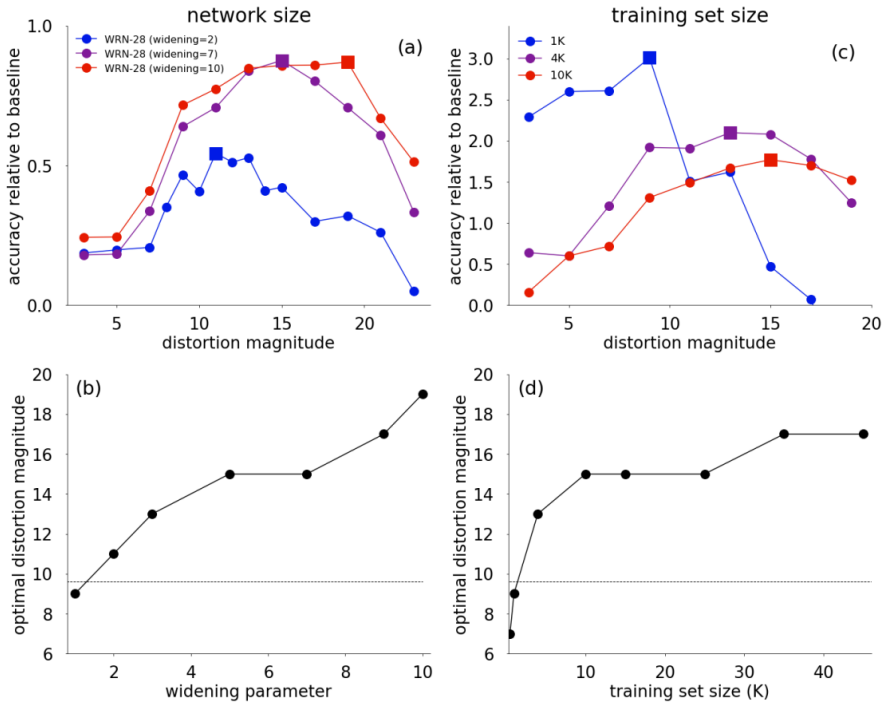
```
transforms = [
'Identity', 'AutoContrast', 'Equalize',
'Rotate', 'Solarize', 'Color', 'Posterize',
'Contrast', 'Brightness', 'Sharpness',
'ShearX', 'ShearY', 'TranslateX', 'TranslateY']

def randaugment(N, M):
"""Generate a set of distortions.

  Args:
    N: Number of augmentation transformations to
        apply sequentially.
    M: Magnitude for all the transformations.
"""

  sampled_ops = np.random.choice(transforms, N)
  return [(op, M) for op in sampled_ops]
```

Figure 2. Python code for RandAugment based on numpy.



- Model의 크기가 클수록 최적 Magnitude 값이 커짐
- Train Data의 크기가 클수록 최적 Magnitude 값 커짐

Ref) Cubuk, E. D., Zoph, B., Shlens, J., & Le, Q. V. (2020). Randaugment: Practical automated data augmentation with a reduced search space. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops (pp. 702-703).in Neural Information Processing Systems, 32.

UniformAugment(UA) _LingChen et al. (2020)

❖ Concept : 모든 확률 분포는 Uniform, 몇가지 Augmentation을 선택할지만 결정

Transform	Narrow	Default	Wide
ShearX(Y)	[-0.15,0.15]	[-0.3,0.3]	[-0.9,0.9]
TranslateX(Y)	[-0.225,0.225]	[-0.45,0.45]	[-1,+1]
Rotate	[-15,15]	[-30,30]	[-90,90]
AutoContrast	N/A	N/A	N/A
Invert	N/A	N/A	N/A
Equalize	N/A	N/A	N/A
Solarize	[0,256]	[0,256]	[0,256]
Posterize	[6,8]	[4,8]	[2,8]
Contrast	[0.5,1.5]	[0.1,1.9]	[0.01,2]
Color	[0.5,1.5]	[0.1,1.9]	[0.01,2]
Brightness	[0.5,1.5]	[0.1,1.9]	[0.01,2]
Sharpness	[0.5,1.5]	[0.1,1.9]	[0.01,2]
Cutout	[0,0.1]	[0,0.2]	[0,0.6]

N개 선택

Algorithm 1: UniformAugment algorithm

Input : $(x, y) \in \{(x_i, y_i)\}_{i=1}^{n}$

Set : $(\hat{x}, \hat{y}) \leftarrow (x, y)$

for $j \leftarrow 1$ **to** $NumOps$ **do**

$t_{ij} \leftarrow t \in T \sim \mathbb{U}_{\{T\}}$;

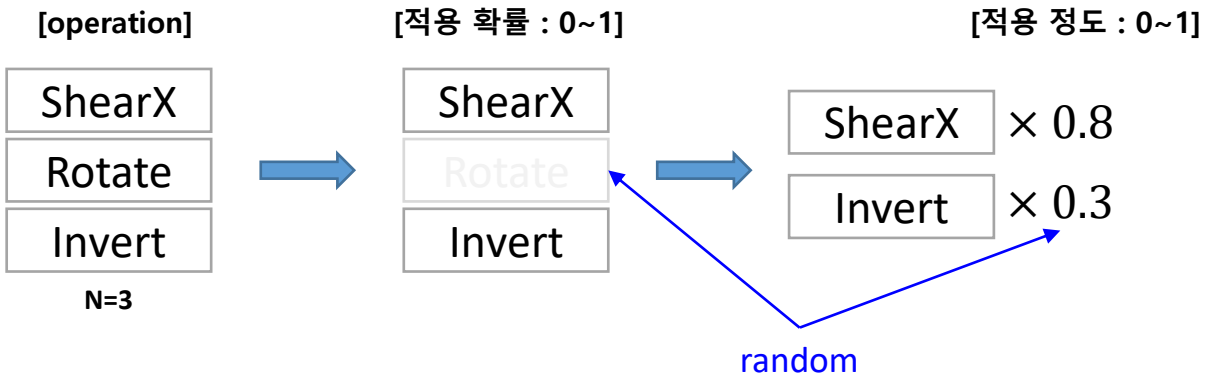
$p_{ij} \leftarrow p \sim \mathbb{U}_{(0,1)}$;

$\lambda_{ij} \leftarrow \lambda \sim \mathbb{U}_{(0,1)}$;

$(\hat{x}, \hat{y}) \leftarrow t_{ij}(\hat{x}, \hat{y}, p_{ij}, \lambda_{ij})$;

end

Return: $(\hat{x}, \hat{y})$



Ref) LingChen, T. C., Khonsari, A., Lashkari, A., Nazari, M. R., Sambee, J. S., & Nascimento, M. A. (2020). Uniformaugment: A search-free probabilistic data augmentation approach. arXiv preprint arXiv:2003.14348.

UniformAugment(UA) _LingChen et al. (2020)

❖ 실험결과 : Search-free data augmentation으로 준수한 성능

	Search space	CIFAR-10 on SS	CIFAR-100 on WRN	ImageNet on RN50
Baseline	0	97.1	81.2	76.3
AA	10^{32}	98.0	82.9	77.6
FAA	10^{32}	98.0	82.7	77.6
PBA	10^{61}	98.0	83.3	N/A
RA	10^{2}	98.0	83.3	77.6
UA	0	98.1	82.8	77.7

Table 2. Average error rate of different augmentation methods on CIFAR10

Model	Baseline	Cutout	AA	PBA	FAA	RA	UA
WRN-40-2	5.6	4.1	3.7	N/A	3.6	N/A	3.75
WRN-28-10	3.9	3.1	2.6	2.6	2.7	2.7	2.67
SS (26 2x32d)	3.6	3	2.5	2.5	2.5	N/A	2.49
SS (26 2x96d)	2.9	2.6	2	2	2	2	1.90

Ref) LingChen, T. C., Khonsari, A., Lashkari, A., Nazari, M. R., Sambee, J. S., & Nascimento, M. A. (2020). Uniformaugment: A search-free probabilistic data augmentation approach. arXiv preprint arXiv:2003.14348.

TrivialAugment(TA) _Müller & Hutter. (2021)

❖ **Concept : 하나의 Augmentation만을 선택**

Transform	Narrow	Default	Wide
ShearX(Y)	[-0.15,0.15]	[-0.3,0.3]	[-0.9,0.9]
TranslateX(Y)	[-0.225,0.225]	[-0.45,0.45]	[-1,+1]
Rotate	[-15,15]	[-30,30]	[-90,90]
AutoContrast	N/A	N/A	N/A
Invert	N/A	N/A	N/A
Equalize	N/A	N/A	N/A
Solarize	[0,256]	[0,256]	[0,256]
Posterize	[6,8]	[4,8]	[2,8]
Contrast	[0.5,1.5]	[0.1,1.9]	[0.01,2]
Color	[0.5,1.5]	[0.1,1.9]	[0.01,2]
Brightness	[0.5,1.5]	[0.1,1.9]	[0.01,2]
Sharpness	[0.5,1.5]	[0.1,1.9]	[0.01,2]
Cutout	[0,0.1]	[0,0.2]	[0,0.6]

1개 선택

Algorithm 1 TrivialAugment Procedure

1: **procedure** TA(x: image)
2: Sample an augmentation a from $\mathcal{A}$
3: Sample a strength m from $\{0, \ldots, 30\}$
4: Return $a(x, m)$
5: **end procedure**

[operation] **[적용 확률]** **[적용 강도]**

ShearX → ShearX → ShearX × 9

N=1 고정 **p=1 고정** random한 값

Ref) Müller, S. G., & Hutter, F. (2021). Trivialaugment: Tuning-free yet state-of-the-art data augmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 774-782).

TrivialAugment(TA) _Müller & Hutter. (2021)

❖ 실험결과

	Default	PBA	Fast AA	AA	RA	UA	TA (Wide)
CIFAR-10							
Wide-ResNet-40-2	96.16 ± .08	-	**96.4**	96.3	-	96.25	96.32 ± .05
Wide-ResNet-28-10	97.03 ± .07	**97.4**	97.3	**97.4**	97.3	97.33	**97.46** ± .06
ShakeShake-26-2x96d	97.54 ± .07	98.0	98.0	98.0	98.0	98.10	**98.21** ± .06
PyramidNet	97.95 ± .05	**98.5**	**98.5**	98.3	**98.5**	**98.5**	**98.58** ± .04
CIFAR-100							
Wide-ResNet-40-2	78.42 ± .31	-	79.4	79.3	-	79.01	**79.86** ± .19
Wide-ResNet-28-10	82.22 ± .25	83.3	82.7	82.9	83.3	82.82	**84.33** ± .17
ShakeShake-26-2x96d	83.28 ± .14	84.7	85.4	85.7	-	85.00	**86.19** ± .15
SVHN Core							
Wide-ResNet-28-10	97.12 ± .05	-	-	98.0	**98.3**	-	98.11 ± .03
SVHN							
Wide-ResNet-28-10	98.67 ± .02	98.9	98.8	98.9	**99.0**	-	98.9 ± .02
ImageNet							
ResNet-50	77.20 ± .32 (93.43 ± .11)	-	77.6 (93.7)	77.6 (**93.8**)	77.6 (**93.8**)	77.63 (-)	**78.07** ± .27 (**93.92** ± .09)

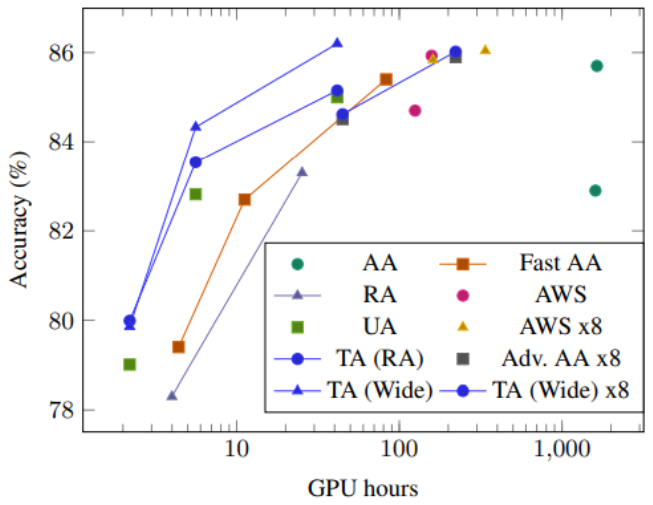
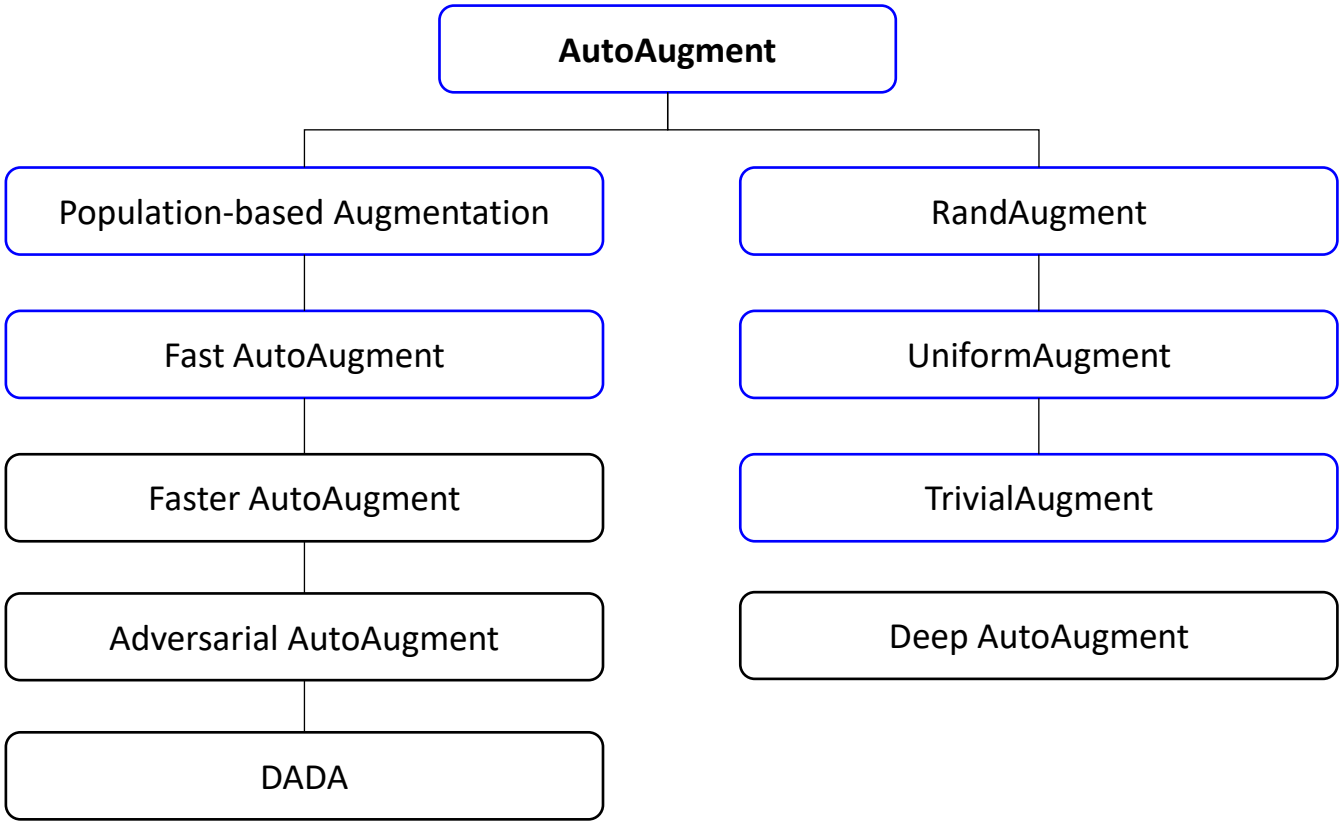


Figure 3: Comparison of the final test accuracy on CIFAR-100 in comparison to RTX2080ti GPU-hours compute invested for *augmentation search* and *final model training* across a set of models. Methods marked with *x8* use batch augmentations[10].

※ limitation!
Image classfication 에서만 좋은 성능을 보이며,
Object detection과 같은 다른 vision 분야에서는 제대로 작동하지 않는 것이 발견

Conclusion

❖ **Optimal Augmentation** : 모델을 통해 찾아내자!



Conclusion

❖ **Auto Augmentation은 모델을 통해 Optimal Augmentation을 찾아가는 분야**

- Augmentation은 좋은 성능을 위해 반드시 고려해야할 필수 요소 중 하나
- 모델을 통한 최적화는 더 높은 성능을 위한 효과적인 방법이 될 수 있음

❖ **데이터 도메인에 대한 사전지식 없이도 적용가능**

- 특별한 사전지식 없이도, 모델을 통해 Augmentation을 최적화할 수 있음
- Augmentation 종류에 따른 성능을 통해 도메인마다 어떤 Augmentation이 효과적인지 정량적으로 확인해볼 수 있음

❖ **Auto Augmentation의 다양한 확장 적용**

- Image 뿐만 아니라, Text, 시계열 데이터, Gragh등 다양한 분야로 확장 연구해볼 여지가 있음

References

1. Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of big data*, *6*(1), 1-48.
2. Ho, D., Liang, E., Chen, X., Stoica, I., & Abbeel, P. (2019, May). Population based augmentation: Efficient learning of augmentation policy schedules. *In International Conference on Machine Learning* (pp. 2731-2741). PMLR.
3. Lim, S., Kim, I., Kim, T., Kim, C., & Kim, S. (2019). Fast autoaugment. *Advances in Neural Information Processing Systems*, 32.
4. Hataya, R., Zdenek, J., Yoshizoe, K., & Nakayama, H. (2020, August). Faster autoaugment: Learning augmentation strategies using backpropagation. *In European Conference on Computer Vision* (pp. 1-16). Springer, Cham.
5. Cubuk, E. D., Zoph, B., Shlens, J., & Le, Q. V. (2020). Randaugment: Practical automated data augmentation with a reduced search space. *In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops* (pp. 702-703).
6. LingChen, T. C., Khonsari, A., Lashkari, A., Nazari, M. R., Sambee, J. S., & Nascimento, M. A. (2020). Uniformaugment: A search-free probabilistic data augmentation approach. *arXiv preprint arXiv*:2003.14348..
7. Müller, S. G., & Hutter, F. (2021). Trivialaugment: Tuning-free yet state-of-the-art data augmentation. *In Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 774-782).
8. Zheng, Y., Zhang, Z., Yan, S., & Zhang, M. (2022). Deep autoaugment. *arXiv preprint arXiv*:2203.06172.
9. https://hoya012.github.io/blog/Image-Data-Augmentation-Overview/
10. https://deepkerry.tistory.com/27
11. http://dmqm.korea.ac.kr/activity/seminar/307

감사합니다.