Networking Day

online

2021/06/22
10:00-17:20

boostcamp ai tech

무려20kg감량

Date: 2021.06.22
Time: 12:20 ~ 12:40

김상현 T1022

안현진 T1120

< Presenter >
반영성 T1082

송민기 T1107

최재하 T1215

Introduction

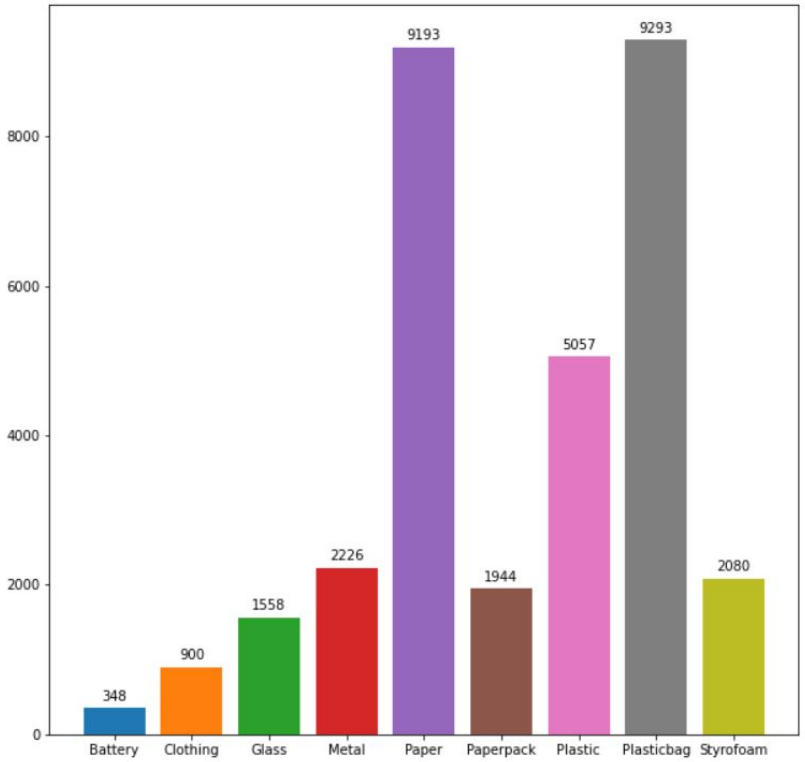
Ranking	User (Team)	score	f1	macs	Entries
2	 모델최적화_4조	0.3606	0.6002	1083210.0000	70

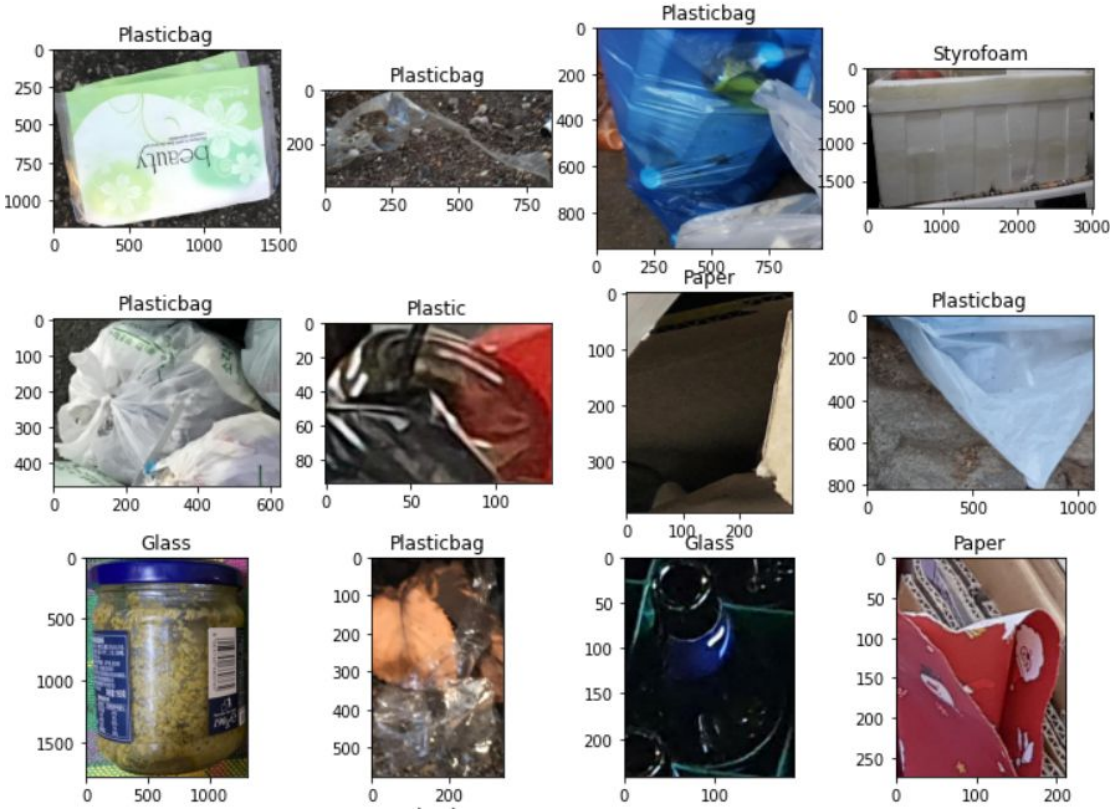
🏆 2nd Solution 🏆

ShuffleNet
+
Pruning
+
Decomposition
+
Knowledge Distillation

Introduction

- Task
  - Image Classification. 이미지를 9개의 재활용 품목 카테고리로 분류하는 문제
  - Model 경량화. 어느 정도의 성능을 유지하며 크기가 작은 모델을 만드는 문제





- MACs(Multiply-accumulate operations)
  - 합,곱연산의횟수
  - MACs -> 약 0.5FLOPs

- 채점 기준
  - $score_{LB} = score_{MACs} + score_{F1}$
  - $score_{MACs} = \frac{\text{제출모델 } MACs}{\text{기준모델 } MACs}$
  - $score_{F1} = \begin{cases} 1 - \frac{\text{제출모델 } F1\ score}{\text{기준모델 } F1\ score} & if\ \text{제출모델 } F1\ score < \text{기준모델 } F1\ score \\ 0.5 * \left(1 - \frac{\text{제출모델 } F1\ score}{\text{기준모델 } F1\ score}\right) & if\ \text{제출모델 } F1\ score \geq \text{기준모델 } F1\ score \end{cases}$

1. Top-down

1.1 Pretrained Model Search

1.2 Model compression

1.3 Vanilla Knowledge Distillation

2. Bottom-up

2.1 NAS & MuxConv

2.2 Scratch Training

3. Additional

3.1 Autoencoder

3.2 Channel attention

3.3 Attention Branch Network

3.4 Arc Face

3.5 Auxiliary Classifier

3.6 Residual Knowledge Distillation

4. Conclusion

1. Top-down

1.1 Pretrained Model Search

1.2 Model Compression

1.3 Vanilla Knowledge Distillation

1.1 Pretrained Model Search

	Input size	MACs	F1
ShuffleNetV1_x0.25	80	1688940.0	0.53
ShuffleNetV2_x0.5	64	3425673.0	0.54
FDMobileNet_x0.25	80	2179592.0	0.51
DiceNet_x0.2	80	2671977.0	0.51
MobileNetV3_x0.1	64	2490808.0	0.49
MNasNet_small_x0.25	64	6537616.0	0.54
MixNet_s_x0.2	64	4150758.0	0.55

1.1 Pretrained Model Search

	Input size	MACs	F1
ShuffleNetV1_x0.25	80	1688940.0	0.53
ShuffleNetV2_x0.5	64	3425673.0	0.54
FDMobileNet_x0.25	80	2179592.0	0.51
DiceNet_x0.2	80	2671977.0	0.51
MobileNetV3_x0.1	64	2490808.0	0.49
MNasNet_small_x0.25	64	6537616.0	0.54
MixNet_s_x0.2	64	4150758.0	0.55

ShuffleNet	Output size	KSize	Stride	Repeat	Output channels
	80x80				groups=3
Conv1 MaxPool	40x40 20x20	3x3 3x3	2 2	1	6
Stage1	10x10 10x10		2 1	1 3 **4**	60 60
Stage2	5x5 5x5		2 1	1 7 **8**	120 120
Stage3	3x3 3x3		2 1	1 3 **4**	240 240
GlobalAvgPool	1x1				
FC (Conv)					9
MACs					1688940.0

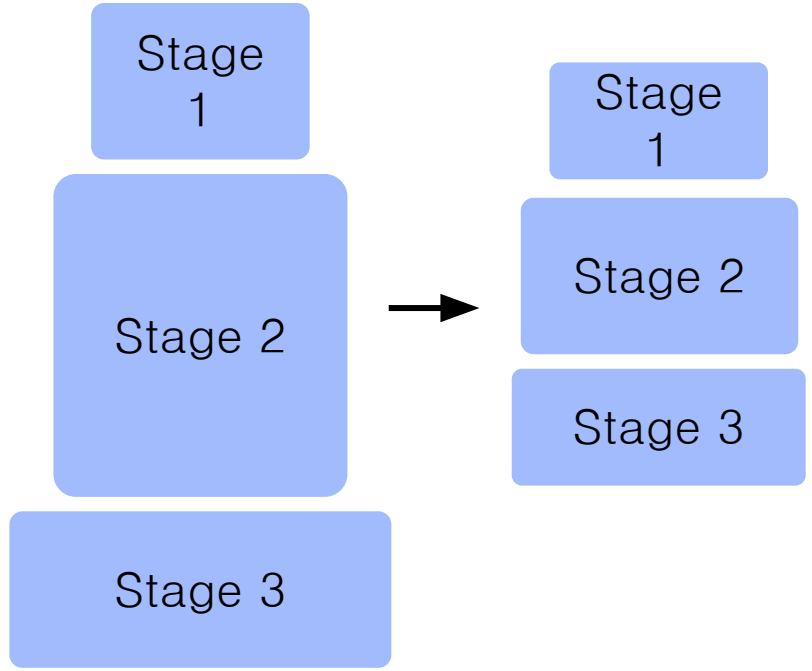
1.2 Model Compression

Motivation & Strategy

- MACs를 직접적으로 감소시키기 위해 Input size를 80x80으로 매우 낮게 설정

- Input size 축소로 인해 추출할 feature 수가 줄어들었다고 판단하여 Network 사이즈를 축소

- 대회 특성을 고려하면 unstructured pruning은 불가능하고 **structured pruning** (layer, channel) 및 **weight decomposition** 시도



1.2 Model Compression

How to apply

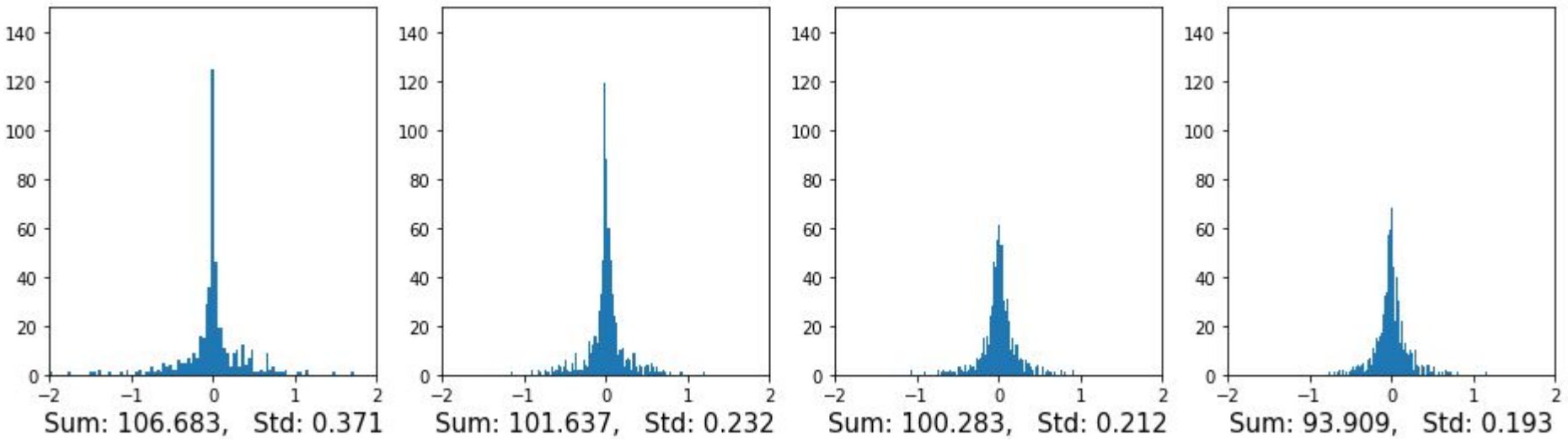
- 각 stage 후반부로 갈수록 0에 가까운 weight가 많이 분포
- 각 레이어의 후반부를 잘라내기로 결정

1.2 Model Compression

How to apply

- 각 stage 후반부로 갈수록 0에 가까운 weight가 많이 분포

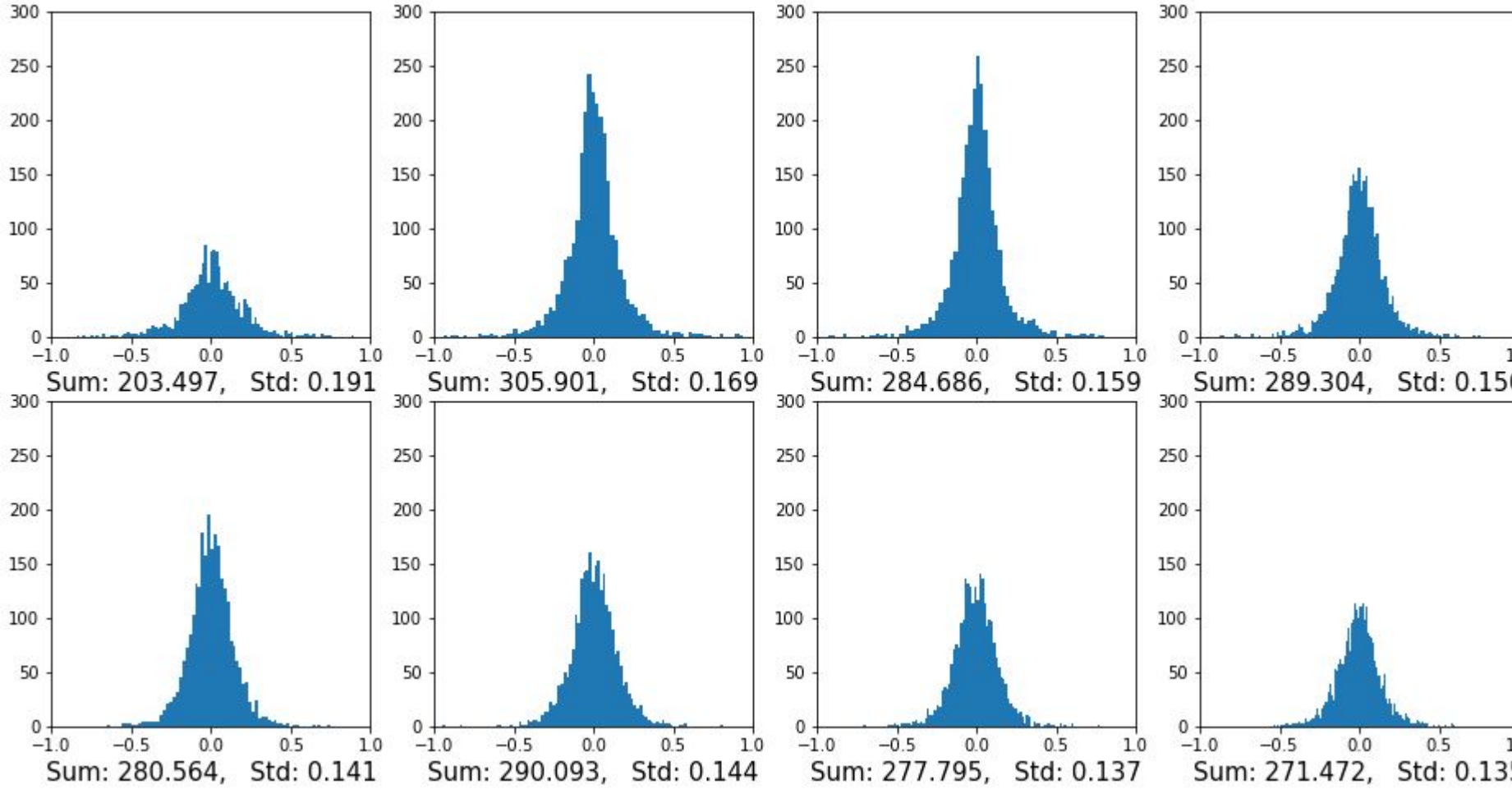
- 각 레이어의 후반부를 잘라내기로 결정



1.2 Model Compression

How to apply

- 각 stage 후반부로 갈수
- 각 레이어의 후반부를

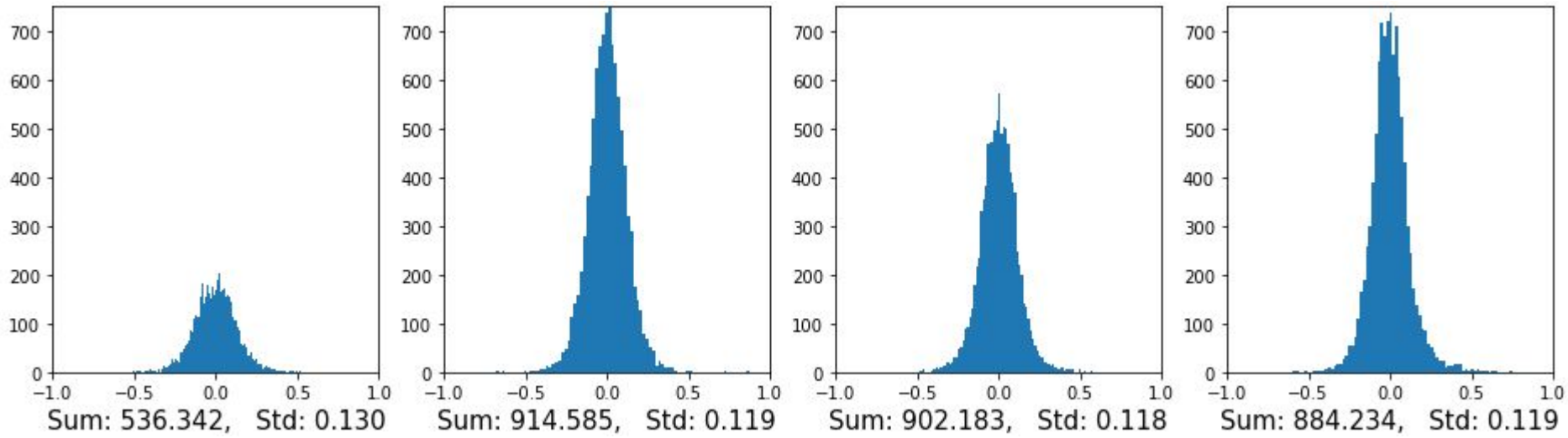


1.2 Model Compression

How to apply

- 각 stage 후반부로 갈수록 0에 가까운 weight가 많이 분포

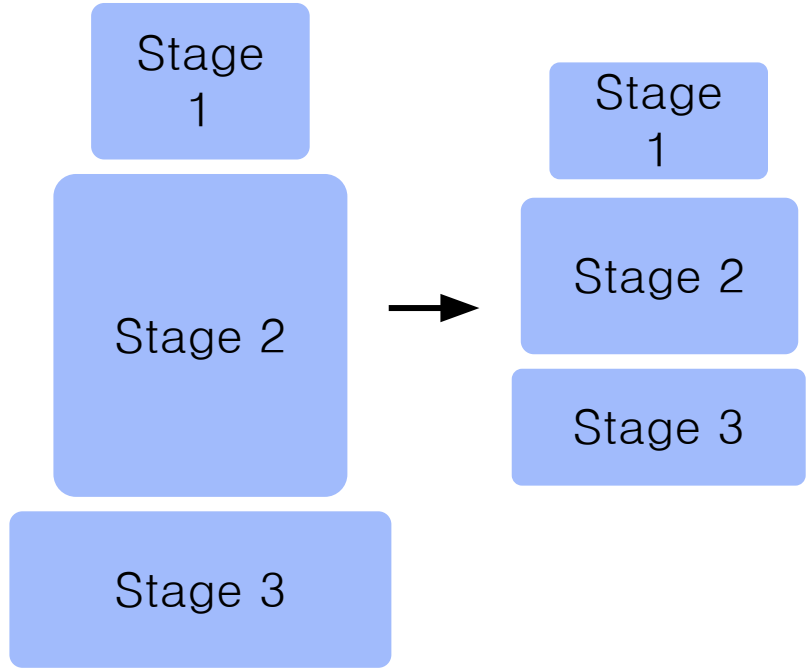
- 각 레이어의 후반부를 잘라내기로 결정



1.2 Model Compression

How to apply

- Base Network : ShuffleNet (ImageNet pretrained)
- layer pruning : [4-8-4] -> [2-5-2]
- channel pruning : stage3 [120, 240] -> [120, 210]
- decomposition : stage3 conv group3 -> group6



Result

- 기존 성능 [F1 0.6206, MACs 1688940.0]
  커스텀 후 [F1 0.6149, MACs 1083210.0]
- 대회 스코어 0.4005 -> 0.3431로 상승

1.3 Vanilla Knowledge Distillation

1. Input_size만 키워 teacher로 활용

Teacher	Student	F1 score
224 (0.6781)	80 (0.5624)	0.5801
336 (0.6994)	80 (0.5624)	0.5826

2. width를 키워 teacher로 활용

x1.0 (0.8201)	x0.25 (0.7473)	0.7579

3. 고성능의 Teacher 모델 사용

A (0.8896)	C (0.5993)	0.6211
B (0.7473)	C (0.5993)	0.6318

4. student를 teacher로 재활용

Teacher	F1 score
A(0.8201) -> C(0.5993)	C(0.6220)
A(0.8201) -> B(0.7473) -> C(0.5993)	C(0.6318)

5.Teacher Ensemble

Teacher	Student	F1 score
B(0.7473)	C (0.5993)	0.6318
A(0.7871) + B(0.7473)	C (0.5993)	0.6386

1.3 Vanilla Knowledge Distillation

1. Input_size만 키워 teacher로 활용

Teacher	Student	F1 score
224 (0.6781)	80 (0.5624)	0.5801
336 (0.6994)	80 (0.5624)	0.5826

2. width를 키워 teacher로 활용

x1.0 (0.8201)	x0.25 (0.7473)	0.7579

3. 고성능의 Teacher 모델 사용

A (0.8896)	C (0.5993)	0.6211
B (0.7473)	C (0.5993)	0.6318

4. student를 teacher로 재활용

Teacher	F1 score
A(0.8201) -> C(0.5993)	C(0.6220)
A(0.8201) -> B(0.7473) -> C(0.5993)	C(0.6318)

5.Teacher Ensemble

Teacher	Student	F1 score
B(0.7473)	C (0.5993)	0.6318
A(0.7871) + B(0.7473)	C (0.5993)	0.6386

1.3 Vanilla Knowledge Distillation

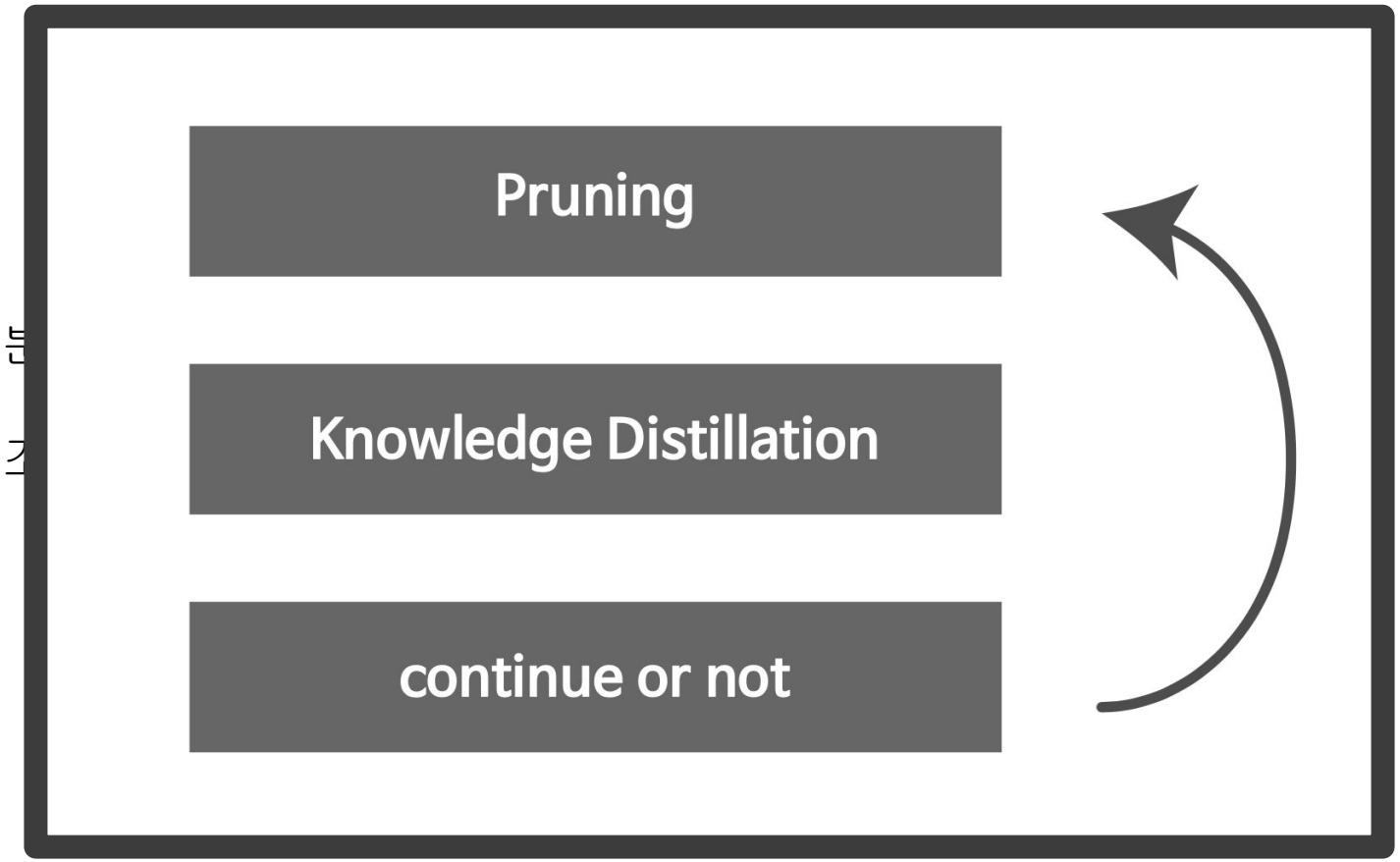
Result

- 비슷한 분포를 가지는 모델을 teacher로 사용하는 것이 매우 효과적
- 분포가 조금 달라져도, teacher를 Ensemble하는 방법은 효과적

1.3 Vanilla Knowledge Distillation

Result

- 비슷한 분
- 분포가 ㅈ



2. Bottom-up

2.1 NAS & MuxConv

2.2 Scratch Training

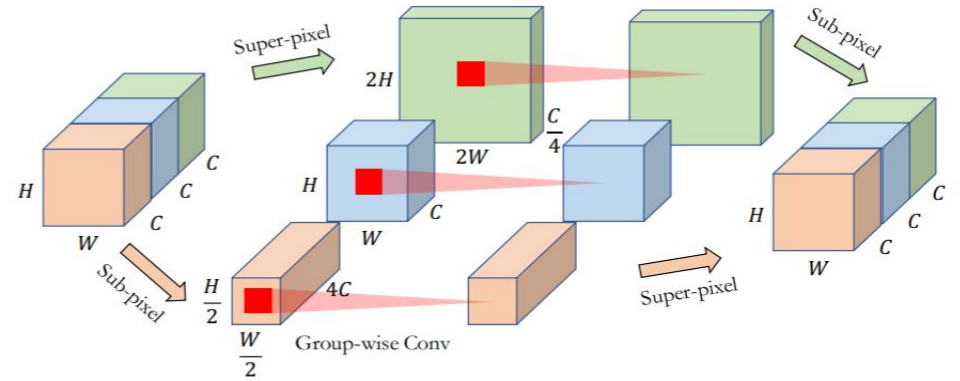
2.1 NAS & MuxConv

NAS Strategy - MuxConv

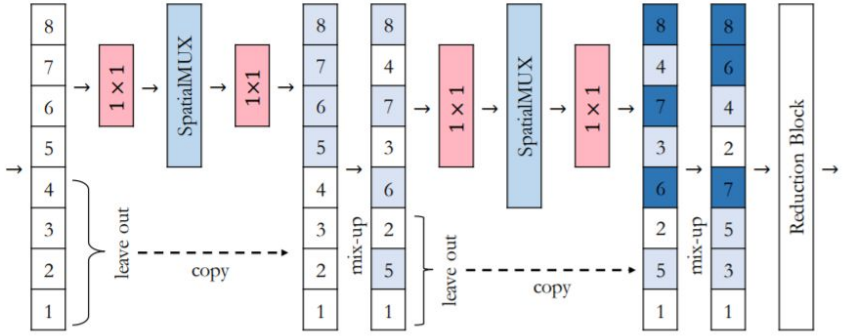
- Search block에 MuxConv 추가
- Objective : score (f1 score + MACs)
- MuxConv
  - Spatial multiplexing
    - 여러 스케일로 채널을 매핑하여 모델의 표현력 상승
  - Channel multiplexing
    - Selective processing과 Channel Shuffling을 통한 낮은 복잡도

Expected Results

- Spatial multiplexing과 Channel multiplexing을 통해 작은 input size에서 네트워크의 표현력을 상승 시킬것이라 기대



Spatial multiplexing



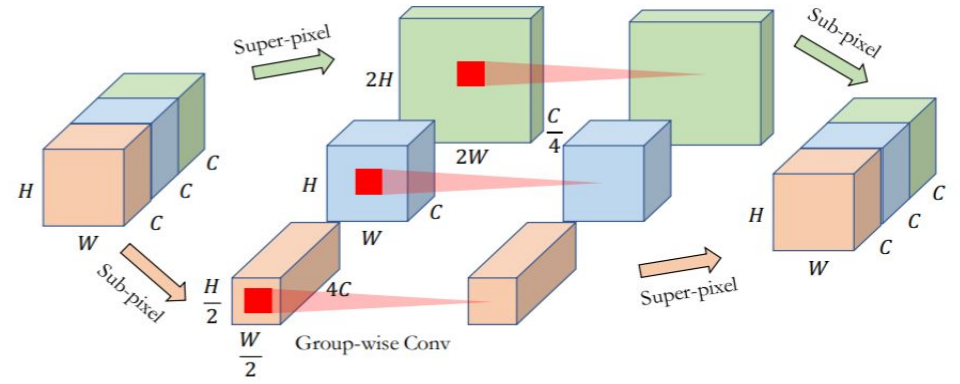
Channel multiplexing

MUXConv: Information Multiplexing in Convolutional Neural Networks, CVPR2020

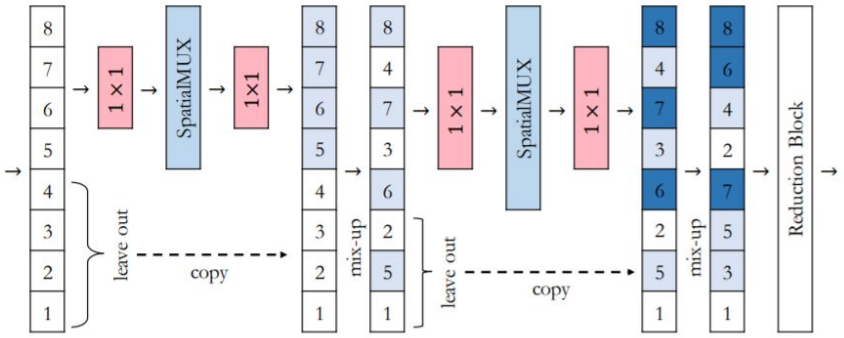
2.1 NAS & MuxConv

Results

- 기대와는 달리 MuxConv로 인한 MACs 상승 대비 f1 score의 상승이 크지 않아 MuxConv를 제외한 block 위주로 search가 진행됨



Spatial multiplexing



Channel multiplexing

MUXConv: Information Multiplexing in Convolutional Neural Networks, CVPR2020

2.2 Scratch training

Motivation & Strategy

- Pretrained weight가 존재하지 않아 taco dataset에서 낮은 성능을 보임
  - Cifar10 dataset과 Cifar100 dataset을 이용해 pretrained weight model을 생성 후 taco dataset에서 transfer learning과 Knowledge distillation 적용
- Student가 Teacher의 오답과 정답을 모두 학습하여 효율적이지 못함
  - Knowledge Distillation 적용 시 teacher가 맞췄을 때에만 loss 적용 (True label KD)

Model & macs

- Mobilenet v3의 InvertedResidualv3의 반복 개수와 filter의 개수를 customizing
- Input 64 기준, 650,073 MACs

2.2 Scratch training

데이터셋	val f1	pretrained	Teacher	비고	KD	Cutout remove
Taco	0.424					
Taco	0.425	CIFAR100_to_10				
Taco	0.433	CIFAR10_to_100				
Taco	0.458	CIFAR10_to_100	mobilev3 large	optimizer : AdamW(1e-2) scheduler : ReduceLROnPlateau(patient =20) loss : f1focal (f1rate = 0.6) epochs : 200	KD	
Taco	0.461	CIFAR10_to_100	mobilev3 large	optimizer : AdamW(1e-2) scheduler : ReduceLROnPlateau(patient =20) loss : f1focal (f1rate = 0.6) epochs : 200	True Label KD	
Taco	0.464	CIFAR10_to_100	mobilev3 large	optimizer : AdamW(1e-2) scheduler : ReduceLROnPlateau(patient =20) loss : f1focal (f1rate = 0.6) epochs : 200	KD	cutout remove
Taco	0.466	CIFAR10_to_100	mobilev3 large	optimizer : AdamW(1e-2) scheduler : ReduceLROnPlateau(patient =20) loss : f1focal (f1rate = 0.6) epochs : 200	True Label KD	cutout remove
Taco	0.466	CIFAR10_to_100	mobilev3 large	optimizer : Adam(1e-2) scheduler : ReduceLROnPlateau(patient =20) loss : f1focal (f1rate = 0.6) epochs : 200	KD	cutout remove
Taco	0.472	CIFAR10_to_100	mobilev3 large	optimizer : Adam(1e-2) scheduler : ReduceLROnPlateau(patient =20) loss : f1focal (f1rate = 0.6) epochs : 200	True Label KD	cutout remove

2.2 Scratch training

데이터셋	val f1	pretrained	Teacher	비고	KD	Cutout remove
Taco	0.424					
Taco	0.425	CIFAR100_to_10				
Taco	0.433	CIFAR10_to_100				
Taco	0.458	CIFAR10_to_100	mobilev3 large	optimizer : AdamW(1e-2) scheduler : ReduceLROnPlateau(patient =20) loss : f1focal (f1rate = 0.6) epochs : 200	KD	
Taco	0.461	CIFAR10_to_100	mobilev3 large	optimizer : AdamW(1e-2) scheduler : ReduceLROnPlateau(patient =20) loss : f1focal (f1rate = 0.6) epochs : 200	True Label KD	
Taco	0.464	CIFAR10_to_100	mobilev3 large	optimizer : AdamW(1e-2) scheduler : ReduceLROnPlateau(patient =20) loss : f1focal (f1rate = 0.6) epochs : 200	KD	cutout remove
Taco	0.466	CIFAR10_to_100	mobilev3 large	optimizer : AdamW(1e-2) scheduler : ReduceLROnPlateau(patient =20) loss : f1focal (f1rate = 0.6) epochs : 200	True Label KD	cutout remove
Taco	0.466	CIFAR10_to_100	mobilev3 large	optimizer : Adam(1e-2) scheduler : ReduceLROnPlateau(patient =20) loss : f1focal (f1rate = 0.6) epochs : 200	KD	cutout remove
Taco	0.472	CIFAR10_to_100	mobilev3 large	optimizer : Adam(1e-2) scheduler : ReduceLROnPlateau(patient =20) loss : f1focal (f1rate = 0.6) epochs : 200	True Label KD	cutout remove

2.2 Scratch training

Result of Transfer learning

Pretrained	F1 score
None	0.424
CIFAR 100 to 10	0.425
CIFAR 10 to 100	0.433

Result of Knowledge Distillation

KD Method	Optimizer	F1 score
KD	AdamW	0.464
True Label KD	AdamW	0.466

KD Method	Optimizer	F1 score
KD	Adam	0.466
True Label KD	Adam	0.472

3. Additional

3.1 Autoencoder

3.2 Channel attention

3.3 Attention Branch Network

3.4 ArcFace

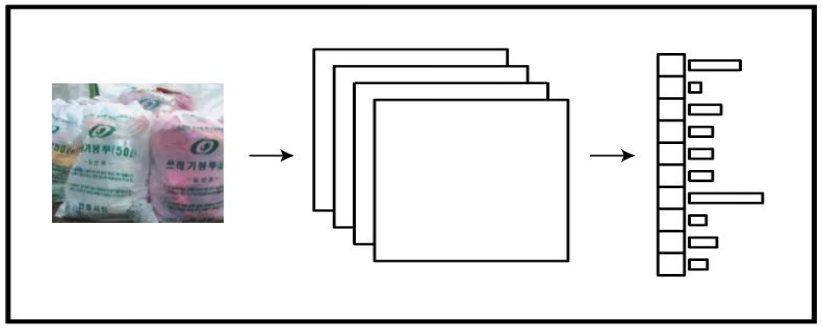
3.5 Auxiliary Classifier

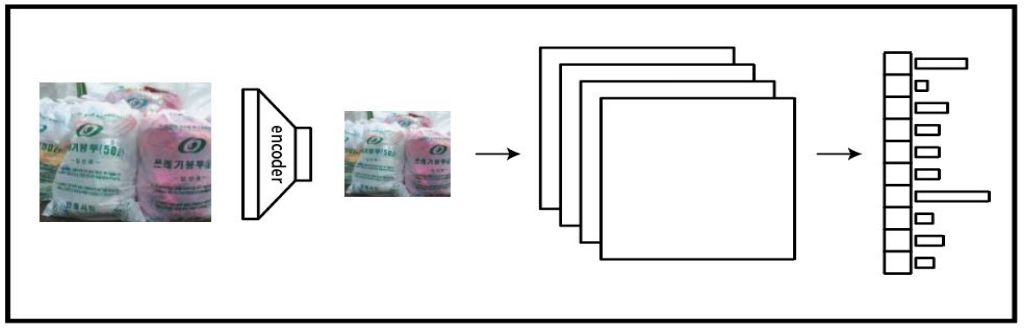
3.6 Residual Knowledge Distillation

3.1 AutoEncoder

Motivation & Strategy

- 이미지의 압축된 정보만으로도 분류가 충분히 가능할 수 있다는 가설
- 입력 이미지에 따라 연산회수가 기하급수적으로 변함
- autoencoder의 인코더를 이용해 압축된 정보를 통해 분류 문제를 해결
- 기존(좌측)의 입력을 auto encoder의 encoder로 압축 후 CNN을 통해 분류하는 형태(우측)로 변형
- F1 score를 유지하되 압축된 정보를 입력으로 전달해 전반적인 MACs 감소 예상





3.1 AutoEncoder

Results

- (encoder의 MACs) + (감소된 backbone의 MACs) ≒ (기존 backbone의 MACs)
- f1 score 유지
- input image size 변화를 통한 최적화 네트워크를 찾는 것이 더 낫다는 판단
- backbone network의 MACs가 클수록 MACs 감소량이 커지기에 모델이 커질수록 더 높은 효율이 기대됨

3.2 Channel Attention

Motivation & Strategy

- channel 내의 정보 중 중요한 feature를 선택적으로 강조하여 성능을 높일 수 있다는 가정
- 적은 추가 연산으로 유의미한 성능 향상을 얻을 수 있다고 기대
- 네트워크의 일부 모듈에서 channel의 feature를 attention하는 block을 추가하여 feature의 표현력을 높임
- SE(Squeeze and Excitation) block

Global average pooling, 두개의 FC-layer를 통해 중요한 정보를 압축, Re-calibration 단계를 거침

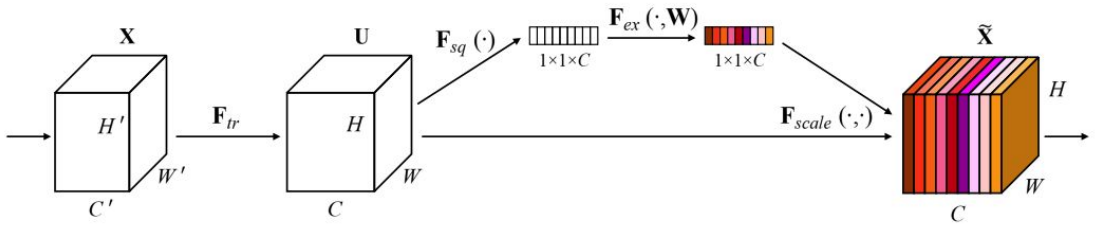


Figure 1. SE block [He. 2018]

3.2 Channel Attention

- ECA(Efficient Channel Attention) block

  2개의 FC-layer 대신 1D Convolution을 사용하여
  효율적으로 Channel Attention을 수행함

- 그 외 모듈 : SA block, FCA block 등

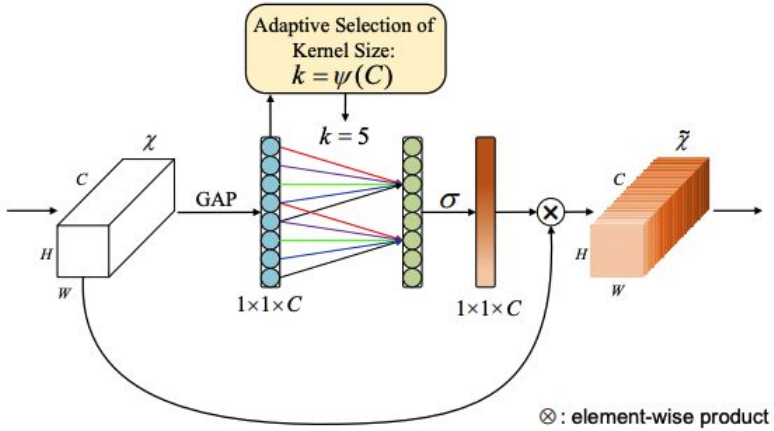


Figure 2. ECA block [Wang. 2019]

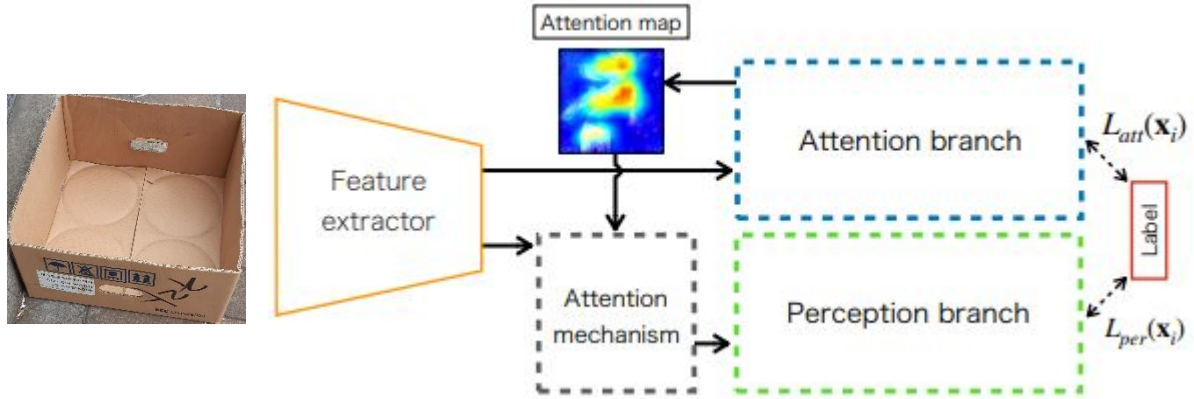
Result

- MACs 소폭 상승
- f1 score 소폭 하락
- 기존 pretrained weight가 layer의 추가로 인해 재학습이 필요한 것이 이유로 생각됨

3.3 Attention Branch Network(ABN)

Motivation & Strategy

- Image recognition 과 Visual explanation 을 동시에 end-to-end로 학습 하는 방법.



Expected Result

- MACs가 약간 증가하지만, 그에 상응하는 성능 향상이 있을 것으로 예상.

Result

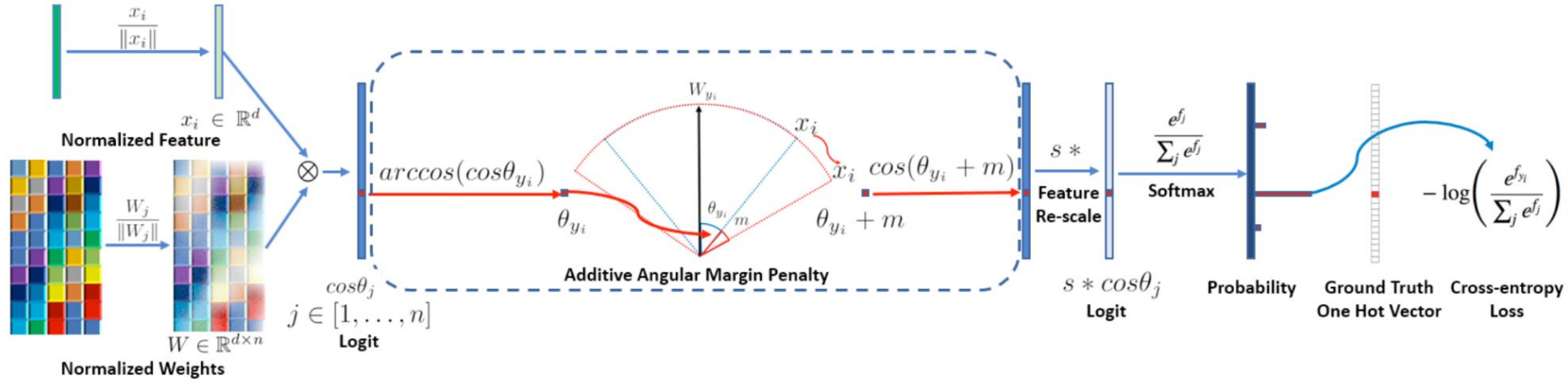
- MACs만 커지고, 성능의 변화가 없었다.

참고 : https://arxiv.org/abs/1812.10025

3.4 Arc Face

Motivation & Strategy

- backbone의 이미지 분별력을 강화시키는 방향으로 학습한다면 MACs의 큰 변화 없이 f1 score를 높일 수 있음
- “ArcFace: Additive Angular Margin Loss for Deep Face Recognition” 내용 기반, 이미지 간 코사인 유사도 연산을 통해 변별력 향상
- 실험을 통해 찾은 네트워크의 backbone에 classifier를 ArcFace classifier로 변환 후 ArcFaceLoss를 이용해 학습



3.4 Arc Face

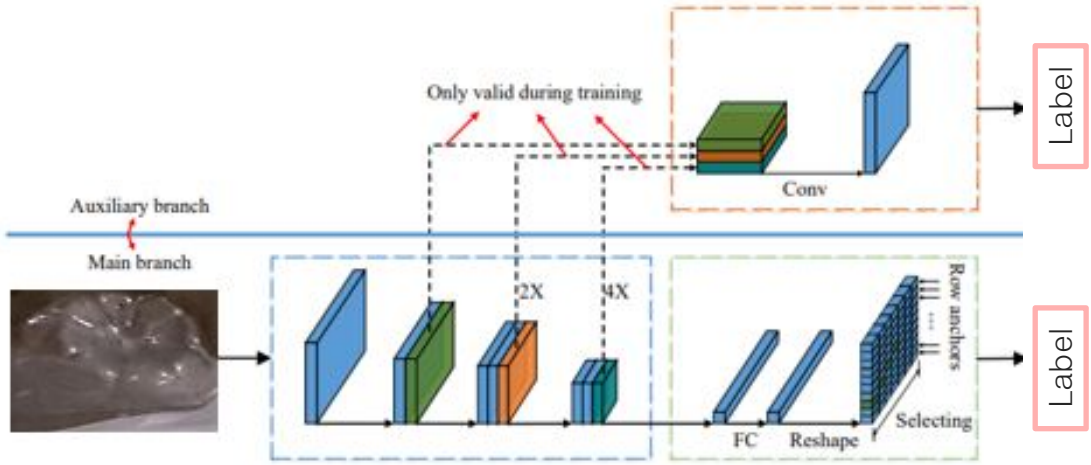
Results

- MACs 유지
- validation f1 score 향상(최소 0.0004, 최대 0.03)
- LB 상 f1 score 대폭 하락(0.2)
- ArcFaceLoss를 사용해 네트워크 학습만으로는 오히려 validation score가 낮아짐
- ArcFaceLoss 학습 -> classifier를 linear로 변경 후 재학습 할 경우 기존 linear layer로 직접 하는 것보다 성능 향상
- F1과 focal loss를 이용한 loss function 수정은 오히려 성능하락을 야기함

3.5 Auxiliary training

Motivation & Strategy

- 각 Block에서의 output을 합쳐 다양한 scale의 feature를 학습.



Expected Result

- MACs의 변화는 없지만, 다양한 scale의 feature를 보기 때문에 성능 향상이 있을 것이라고 예상.

Result

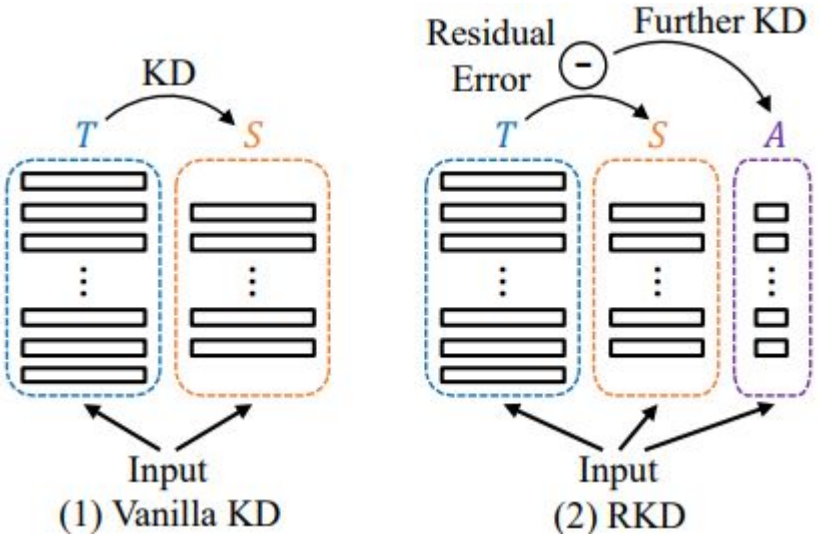
- MACs에서는 변화가 없었고, 성능 저하가 있었음.

참고 : https://arxiv.org/abs/2004.11757

3.6 Residual Knowledge Distillation

Residual Knowelege Distillation

- teacher와 student 사이의 residual error(분포 차이)를 학습하는 assistant 모델을 추가
- 단점 : Inference time에서 assistant 모델도 사용하기 때문에 MACs가 늘어나는 문제



How to apply & Results

- student 모델의 depth는 유지하고 width를 둘로 나눔
- MACs : 0.65M -> 0.45M + 0.19M
  F1 : 0.24 -> 0.24
- assistant가 너무 작아서 제대로 학습이 불가능하다고 판단
- MACs 1.68M의 더 큰 모델에서는 약간의 향상을 보임

Conclusion

1. Top-Down 방식과 Bottom-up, 그리고 두 방식 모두에 적용가능한 additional 방법을 통해 효율적인 실험과정을 계획하였다.

2. Pruning, Tensor decomposition, Knowledge Distillation, NAS 등 경량화 기법 외에도 Mux Conv, Channel Attention, ABN, Auto Encoder, ArcFace, Auxiliary training 등의 다양한 접근을 하였다.

3. 각 방식은 논문의 환경과 유사하였을 때 가장 좋은 효율을 나타냈다.

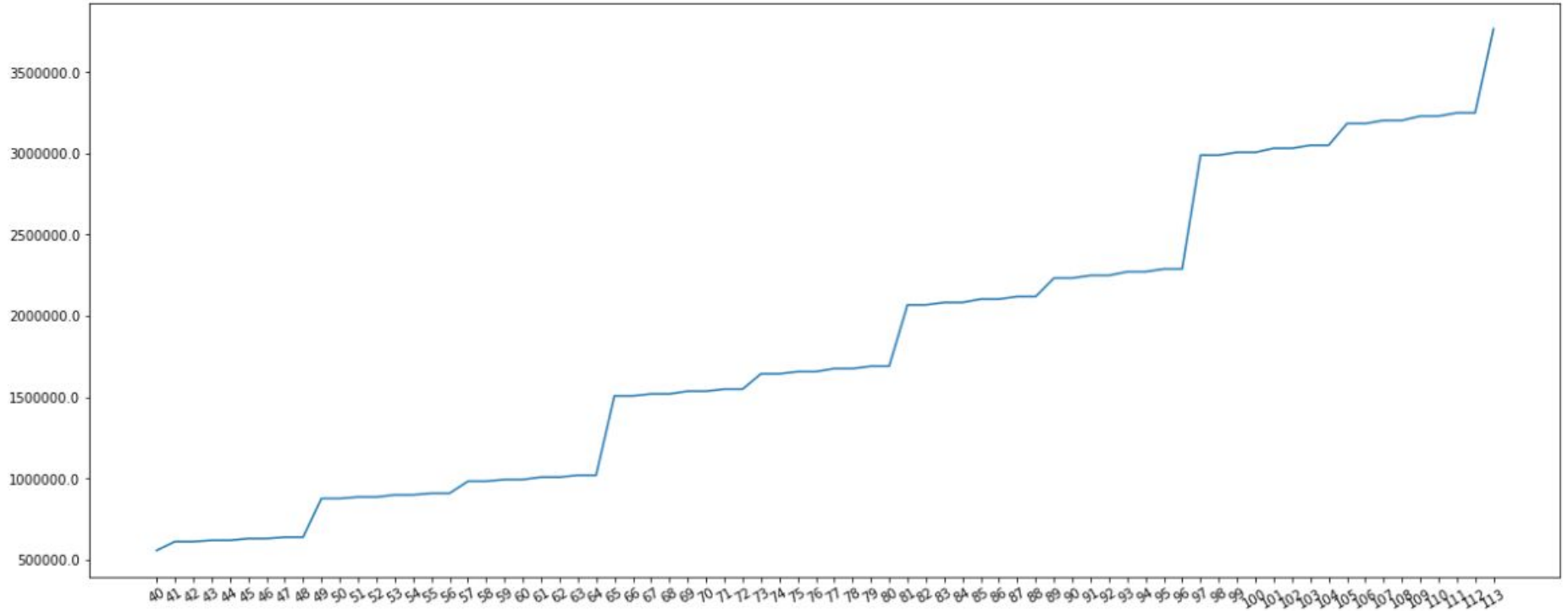
4. Pruning, Tensor decomposition, Knowledge Distillation을 사용한 Top-Down 방식의 결과가 가장 용이함을 보였다.

End of Document

Thank You.

QnA

QnA : Input size는 왜 80?



MACs가 확 증가하는 부분이 있습니다.
(80 -> 81의 경우 1688949.0에서 2066763.0로 MACs가 확 증가)

때문에 확 증가하기 직전의 Input size를 후보군으로 선택했고,
80사이즈에서 MACs 대비 F1 스코어가 가장 높았습니다.

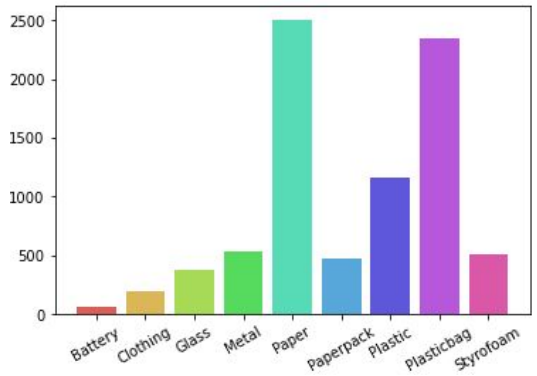
Input size	MACs	F1
64	1018137.0	0.47
80	**1688940.0**	**0.53**
96	2288097.0	0.56
112	3249021.0	0.59

QnA : Validation 전략?

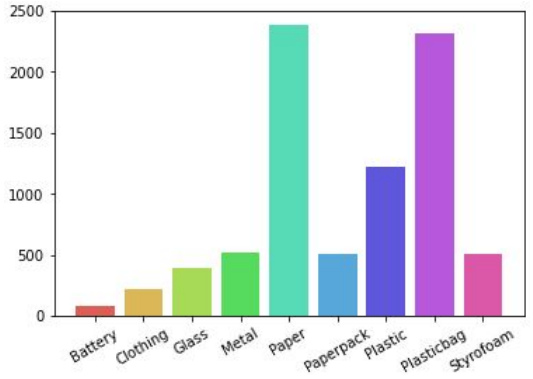
Motivation

- Inference 분포를 확인했을 때, 성능이 높을 때와 낮을 때 분포에서 많은 차이가 나지 않음

F1 : 0.6280



F1 : 0.8241

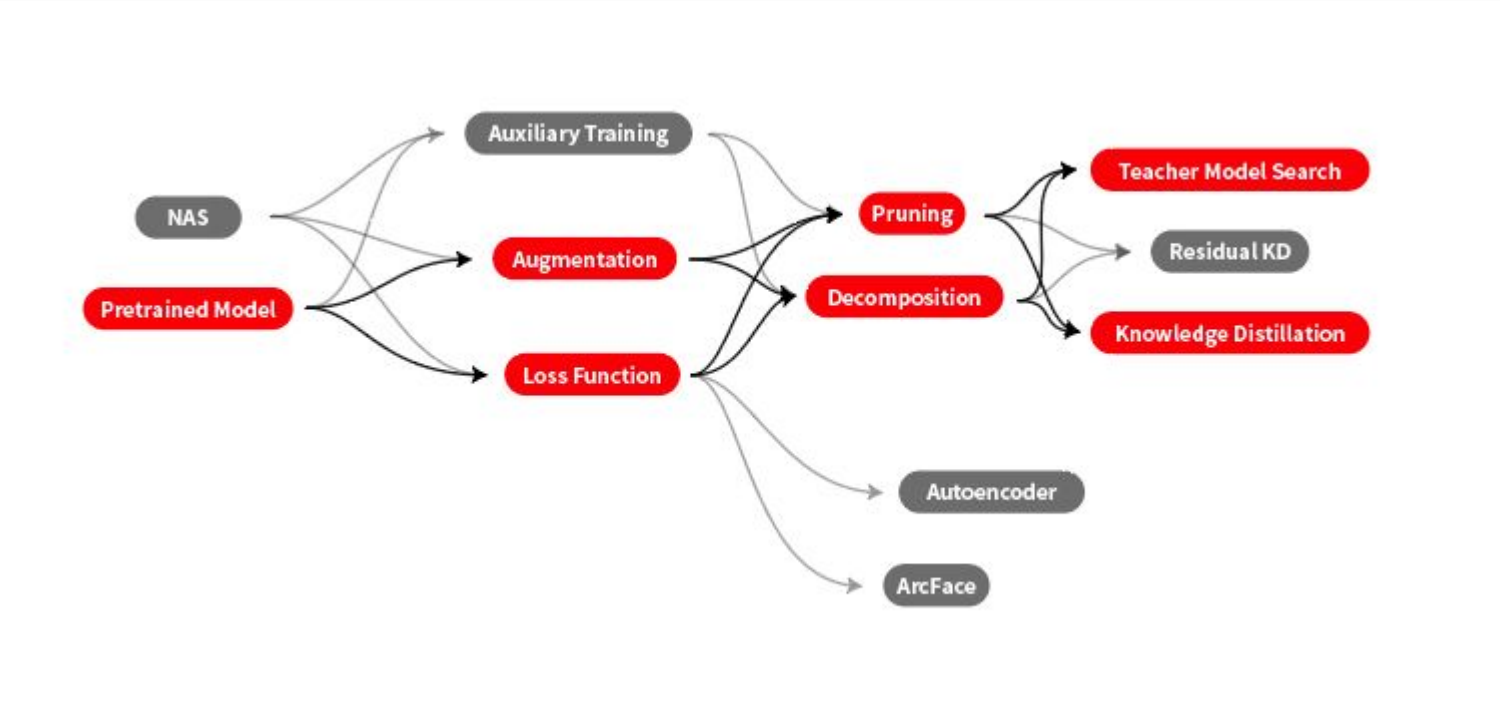


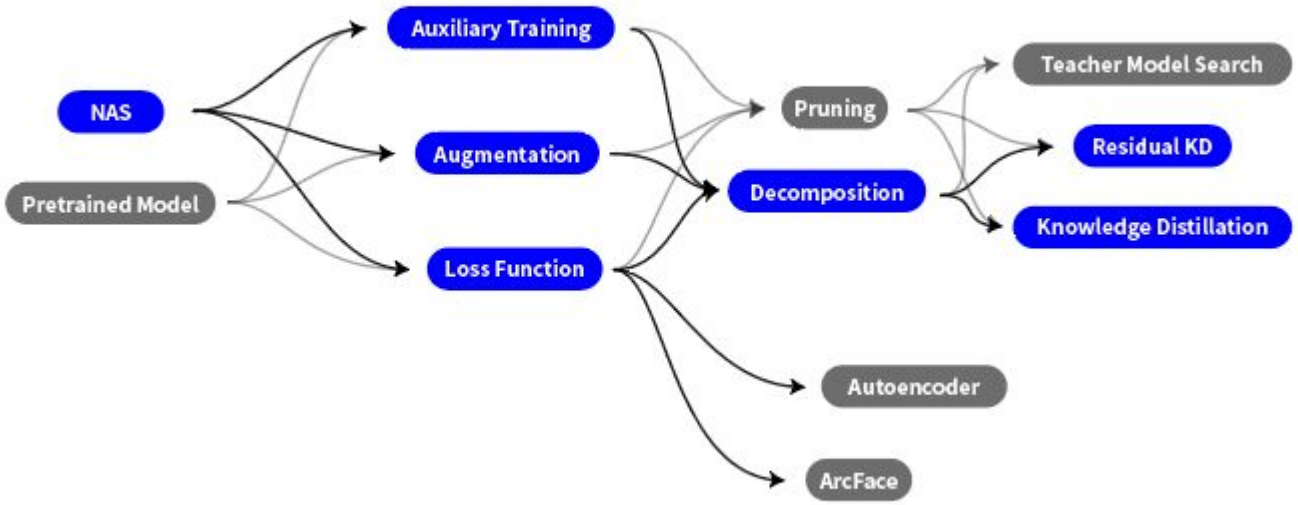
How to apply

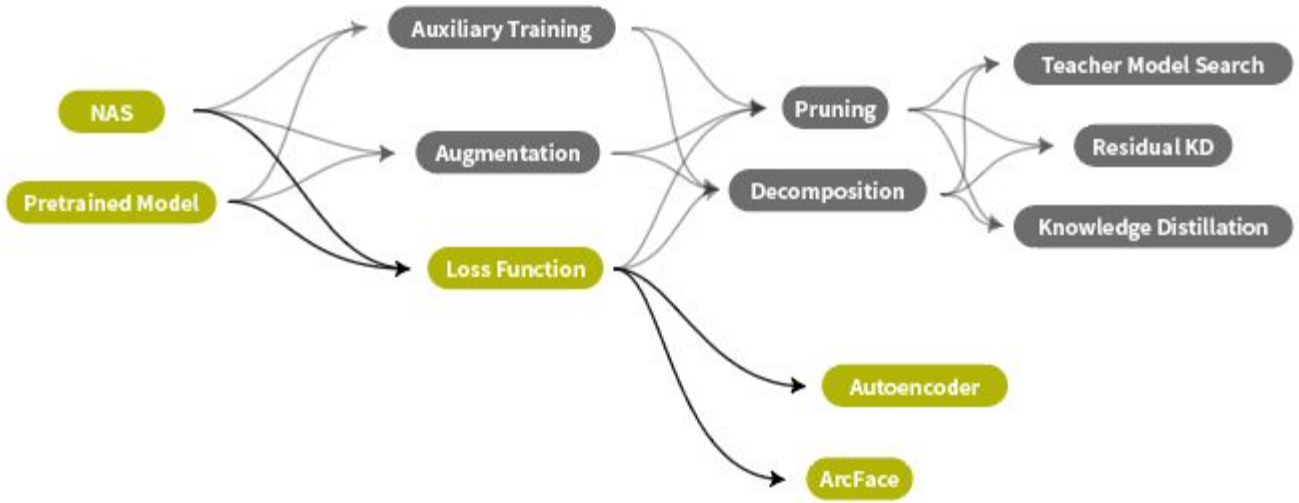
- 높은 성능의 모델 결과와 test data를 이용해서 validation.
- All Data Train : 기존 분리된 train data만 활용했을때보다 LB F1이 0.02정도 상승

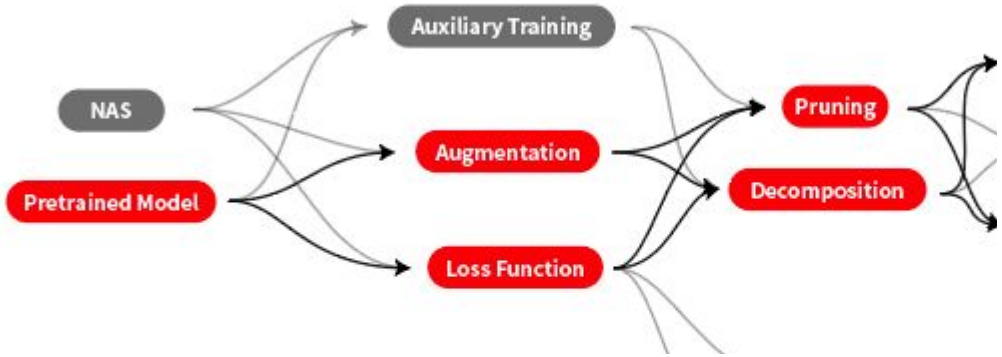
Used Model

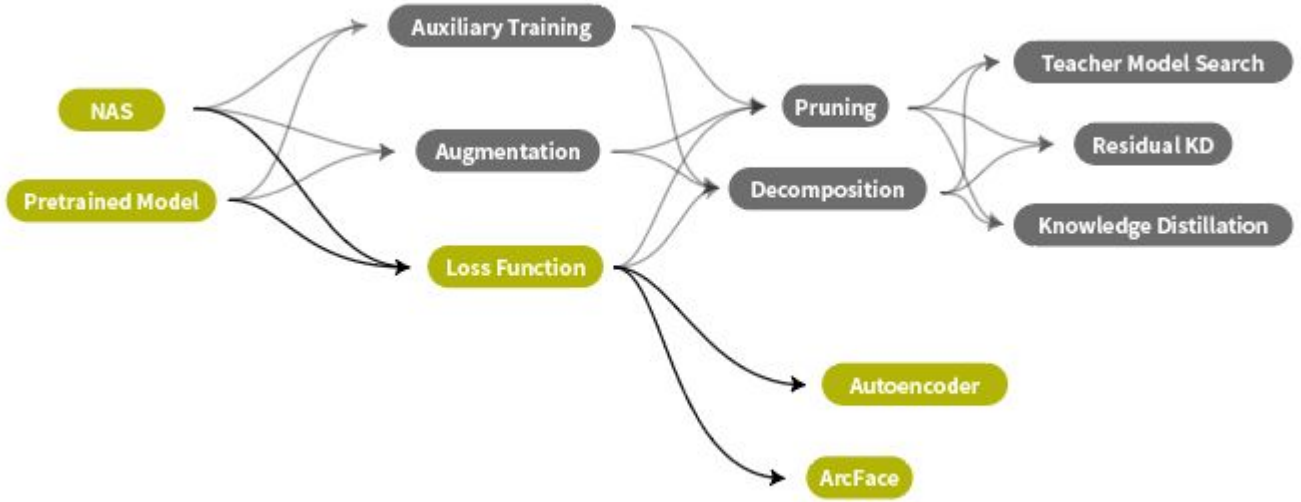
EfficientNet-V2 (ImageNet pretrained)
Random Box Cutmix
LB F1 : 0.8896

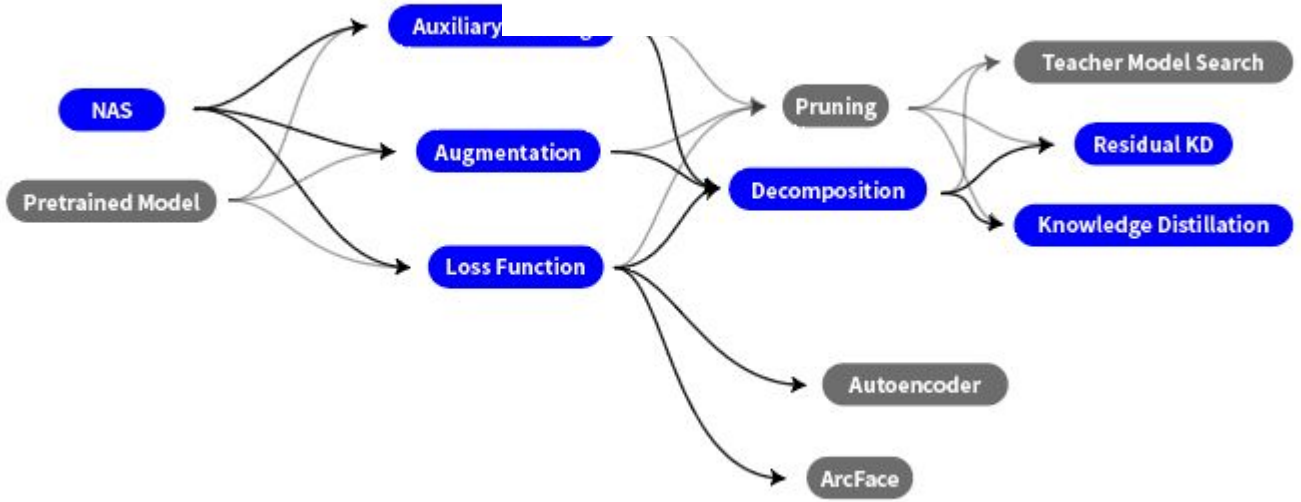

Auxiliary Training
NAS
Augmentation
Pretrained Model
Loss Function
Pruning
Decomposition
Teacher Model Search
Residual KD
Knowledge Distillation
Autoencoder
ArcFace


NAS
Pretrained Model
Auxiliary Training
Augmentation
Loss Function
Pruning
Decomposition
Teacher Model Search
Residual KD
Knowledge Distillation
Autoencoder
ArcFace




NAS
Pretrained Model
Auxiliary Training
Augmentation
Loss Function
Pruning
Decomposition


NAS
Pretrained Model
Auxiliary Training
Augmentation
Loss Function
Pruning
Decomposition
Teacher Model Search
Residual KD
Knowledge Distillation
Autoencoder
ArcFace

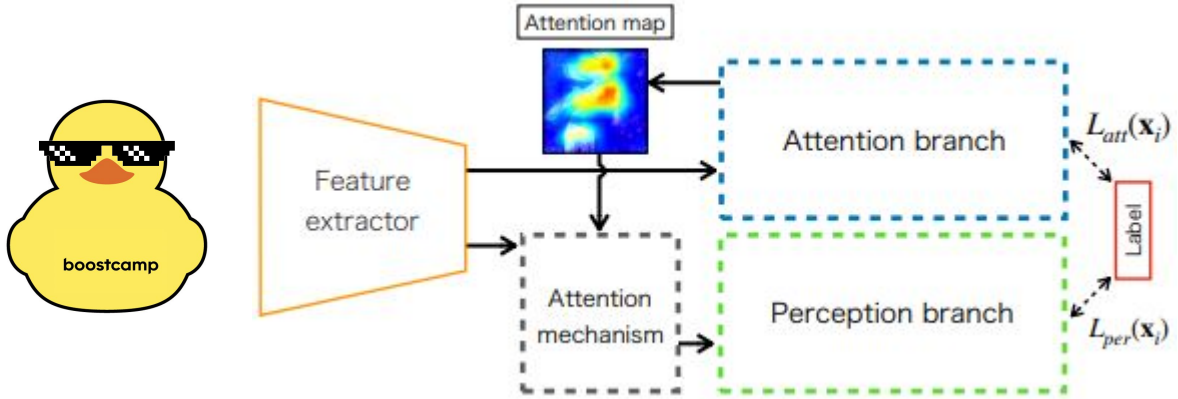

NAS
Pretrained Model
Augmentation
Loss Function
Pruning
Decomposition
Teacher Model Search
Residual KD
Knowledge Distillation
Autoencoder
ArcFace


숨긴거 아닙니다
boostcamp

3.3 Attention Branch Network(ABN)

Motivation & Strategy

- Image recognition 과 Visual explanation 을 동시에 end-to-end로 학습 하는 방법.



Expected Result

- MACs가 약간 증가하지만, 그에 상응하는 성능 향상이 있을 것으로 예상.

Result

- MACs만 커지고, 성능의 변화가 없었다.

참고 : https://arxiv.org/abs/1812.10025