

23' 동계 현장설습 기획 프로젝트 보고

생산기술연구소 로봇기술그룹 송찬의

01. Introduction-Digit



Digit: vision 기반 tactile sensor

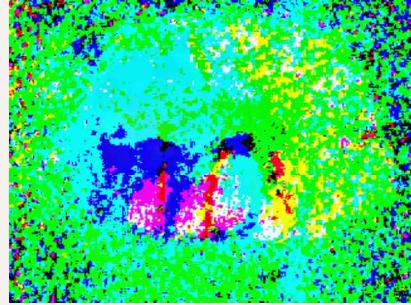
RGB 조명에 의해 화려한 무늬가 보임
표면에 물체가 닿으면 빛의 경로가 변경되어 무늬가 변경됨
프로젝트: Digit 활용 Material Classification 구현



Base Image
(320*240)

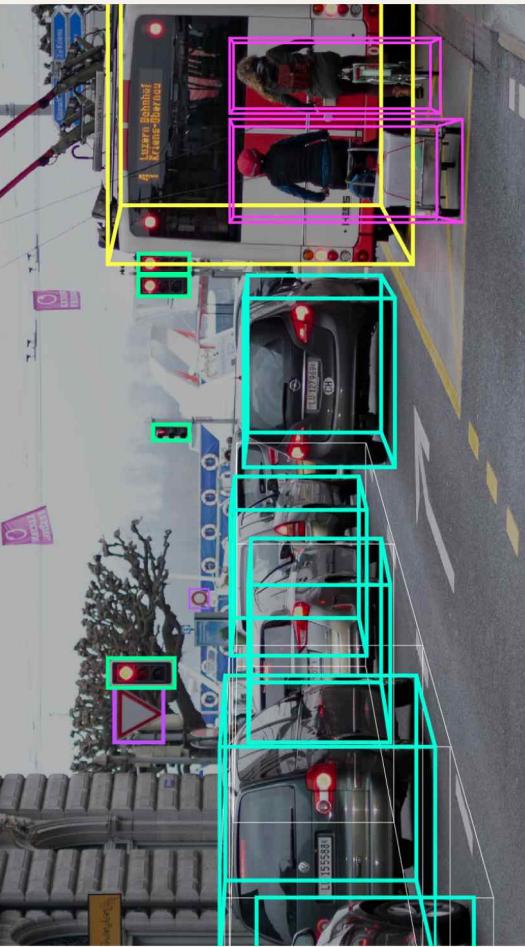


Material(100원 동전)과
접촉했을 때의 Raw Image



Raw Image에서 Base Image를 각
RGB값을 빼서 만든 Revised Image

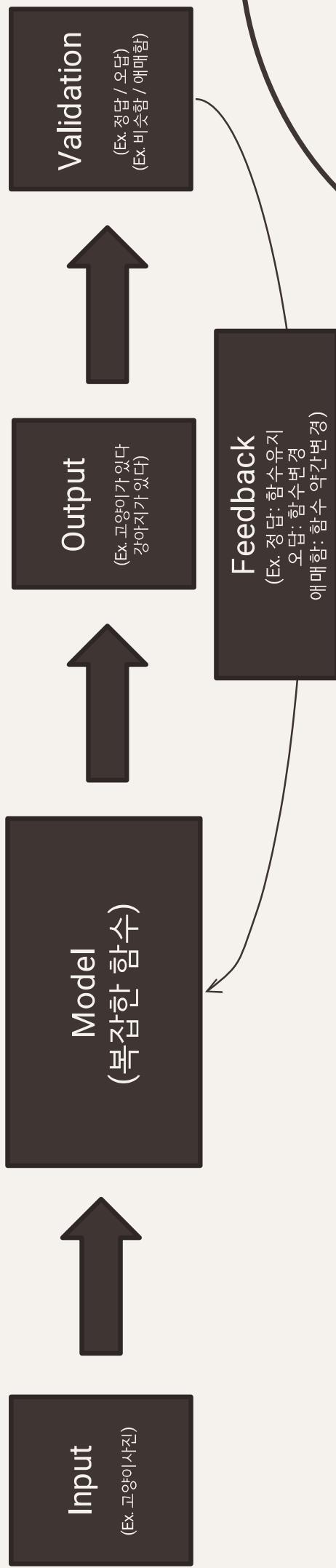
01. Introduction-Computer Vision



카메라를 통해 촬영된 이미지를 컴퓨터 알고리즘을 통해 해석 최근엔 다량의 이미지 데이터를 기반으로 한 머신러닝 기반의 컴퓨터 비전이 널리 사용됨 자율주행, 산업용 로봇 등 여러 분야에서 널리 사용되는 기술

Digit 역시 컴퓨터 비전을 통해 활용

01. Introduction-Machine Learning



- 목표: input(Ex. 사진)에 대한 원하는 output(Ex. 사진 속의 물체)를 출력하는 **학습(모델) 찾기**
- 방법: input에 대한 험수의 output을 보고, 실제 정답과 비교하여 험수를 계속해서 수정해나감
- 기계를 **학습**시킨다

01. Introduction-Machine Learning



- 충분히 학습된 모델을 이미지| 분류 등에 활용함

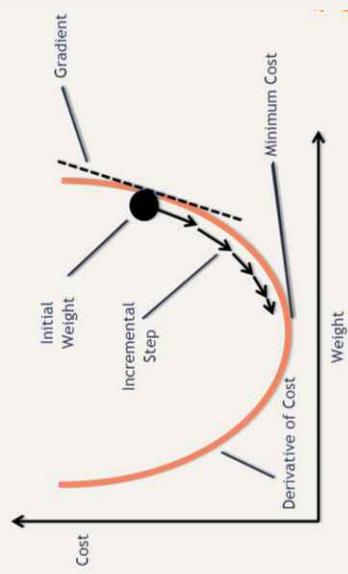
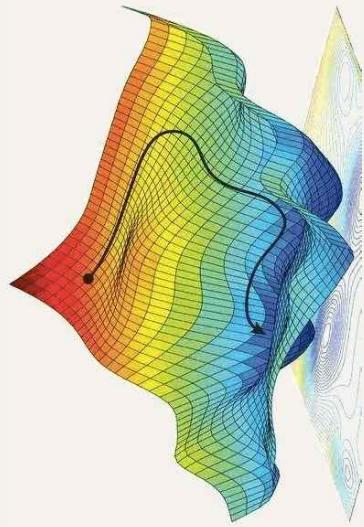
중요한 것

- 어떠한 형태의 model을 사용할 것인가
- Feedback 과정을 어떻게 설계할 것인가

- 어떠한 형태의 Model이 어떠한 목적에 제일 효율적인지| 연구자들 사이에서 널리 공유되는 종
- Feedback 과정 역시 대부분의 사람들이 활용하는 방법이 널리 알려짐

01. Introduction-Machine Learning

- Feedback 과정의 기본 아이디어
- 1. 적절한 Loss 함수를 찾는다.
 - 이 함수는 정답이면 낮은 값을, 오답이면 높은 값을 배출하도록 설정
 - Loss 값을 최소화하는 것이 정확한 모델을 설계하는 것과 동치
 - 널리 사용되는 함수들이 존재 (Ex. Cross Entropy Loss)
- 2. Loss를 최소화하기 위한 Gradient Descent 진행
 - 아래 그림에선 변수가 1개/2개만 보통 모델에선 수만개의 parameter 존재



01. Introduction-Machine Learning

Machine Learning의 특징

- Loss를 최소화하는 과정에서 많은 데이터가 필요하다
- 데이터가 많을수록 유리하다
- Loss와 parameter 사이에 관계가 직관적이지 않다: Loss가 감소하다가 갑자기 증가할 수도 있음

별표에서 사용한 용어

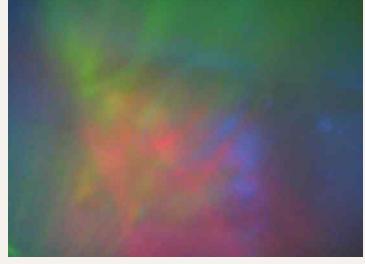
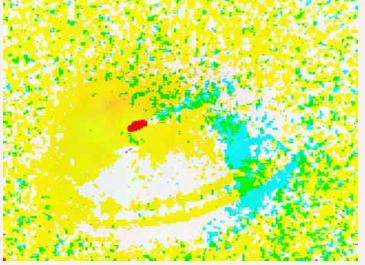
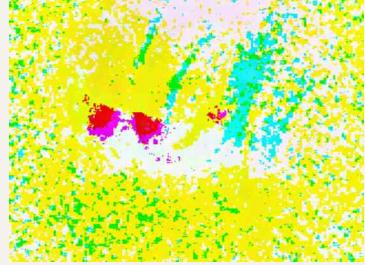
- Training Loss: Feedback 과정에서의 Loss Function. 날을수록 정답에 가까운 모델이 됐다는 뜻.
- Accuracy: 중간중간 모델을 test했을 때 모델의 정답률
- Learning rate: 모델 feedback을 얼마나 강하게 할 것인가
Learning rate가 작으면 모델 feedback이 소극적으로 반영됨
Learning rate가 크면 모델 feedback이 빠름. 정답에 가까운 점을 지나칠 가능성이 높아짐.
적당한 Learning rate를 여러 차례 실험을 통해 찾아내는 것이 중요
- Train data: 모델을 학습시키는 과정에서 쓰는 데이터. 정답을 제시하여 모델 feedback에 반영.
- Test data: 모델을 평가하는 과정에서 쓰는 데이터. 정/오답 여부는 feedback에 반영안됨. Test data와 부르

02. Dataset Collection

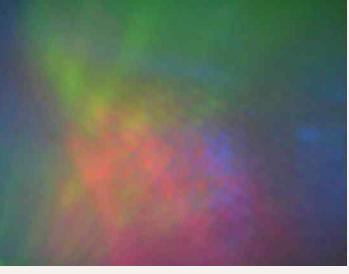
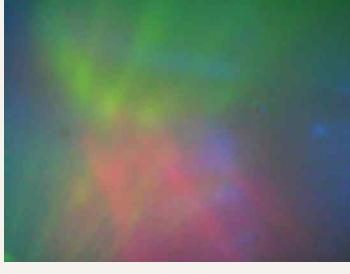
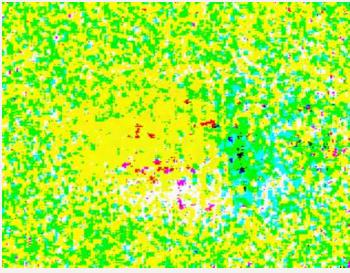
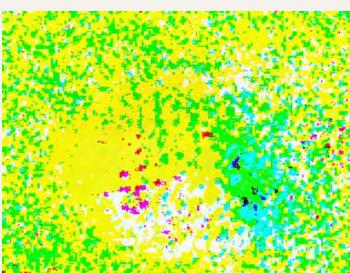
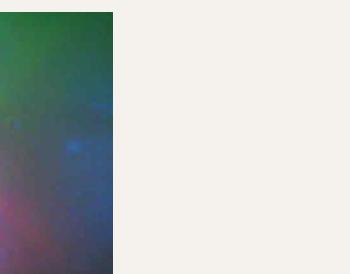
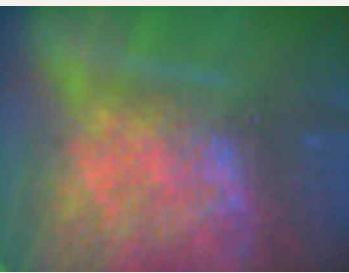
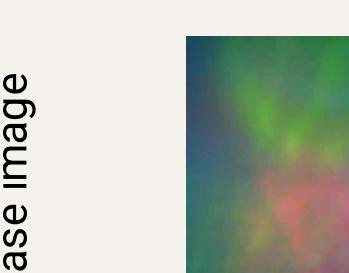
- Image Classification 구현 위한 데이터셋 수집
- Raw image 및 초기 base image와의 RGB값의 차를 계산한 Revised image 수집(Simple Subtraction)
 - $r_{ss,i} = r_{raw,i} - r_{base,i}$
- 100원 동전 / 안경과 표면 데이터 각각을 3차례에 걸쳐 100개씩 수집
 - (Set1: 동전 100개 / 안경 100개)
 - (Set2: 동전 100개 / 안경 100개)
 - (Set3: 동전 100개 / 안경 100개): 추가수집

```
iter=100
ret= input('Do you want to collect dataset?')
if ret=='y':
    for i in range(iter):
        dataset_raw_img_path = '{}/_raw_{}.png'.format(i)
        dataset_revised_img_path='{}/_revised_{}.png'.format(i)
        input('Press any key to continue. Dataset would be saved.')
        time.sleep(2)
        d.save_frame(dataset_raw_img_path)
        image=d.get_frame()
        diff=np.asarray(image)-np.asarray(base_img)
        print(diff.shape)
        rev_img=Image.fromarray(diff)
        rev_img.save(dataset_revised_img_path)
        print(f'{i+1} Dataset collected.')
    print("Dataset All collected. Shut Down.")
dataset_collect.py
```

02. Dataset Collection (동전)

1 번째 set의 base image	Raw data 10	Raw data 72	2 번째 set의 Base image	Raw data 110	Raw data 172
					

02. Dataset Collection (환경학)

1 번째 set의 base image	Raw data 10	Raw data 72	2 번째 set의 Base image	Raw data 110	Raw data 172
					
					

02. Dataset Collection

수집 데이터 특이사항

Digit 센서의 안정성에 문제가 있는 것으로 예상됨
같은 100원 동전에 대한 데이터를 얻어도 처음 100개 이미지와 그다음 100개 이미지 사이의 차이가 큼.

특히 저체적인 색감차이가 육안으로 식별가능할 정도로 큽

Base Image 측정에 서의 오차발생이 원인이 될 가능성이 높음
센서 재연결 시 rgb 조명의 특성이 달라지는 것이 원인일 수도 있다.

1번 째 set와 2번 째 set에서 얻은 데이터를 같은 물체를 나타내는 특성이이라고 보기 어려움
다만 각 set 내에서는 revised image의 형상이 비슷함

03. Data Augmentation

데이터셋을 수기로 수집하므로, 많은 시간 소요
동전 / 안경과는 대칭된 모습을 사용해도 그 특성이 변하지
않음
한 이미지를 상하/좌우/상하좌우 대칭하여 4배로 증대

```
for i in range(len(raw_images)):
    # 한 이미지를 상하좌우로 4번 대칭이동 시킨다.
    images_fliaug = []
    for image in raw_images[i]:
        aug_images[i].append(image)
        left_image=cv2.flip(image, 0)
        aug_images[i].append(left_image)
        up_image=cv2.flip(image, 1)
        aug_images[i].append(up_image)
        diag_image=cv2.flip(image, -1)
        aug_images[i].append(diag_image)
    random.shuffle(aug_images[i])
    for j in range(len(aug_images[i])):
        if j<0.9*len(aug_images[i]):
            train_images[i].append(aug_images[i][j])
        else:
            test_images[i].append(aug_images[i][j])
print("Dataset Augmented")
```

resnetver3.py 中 data augmentation 부분

04. Model Selection

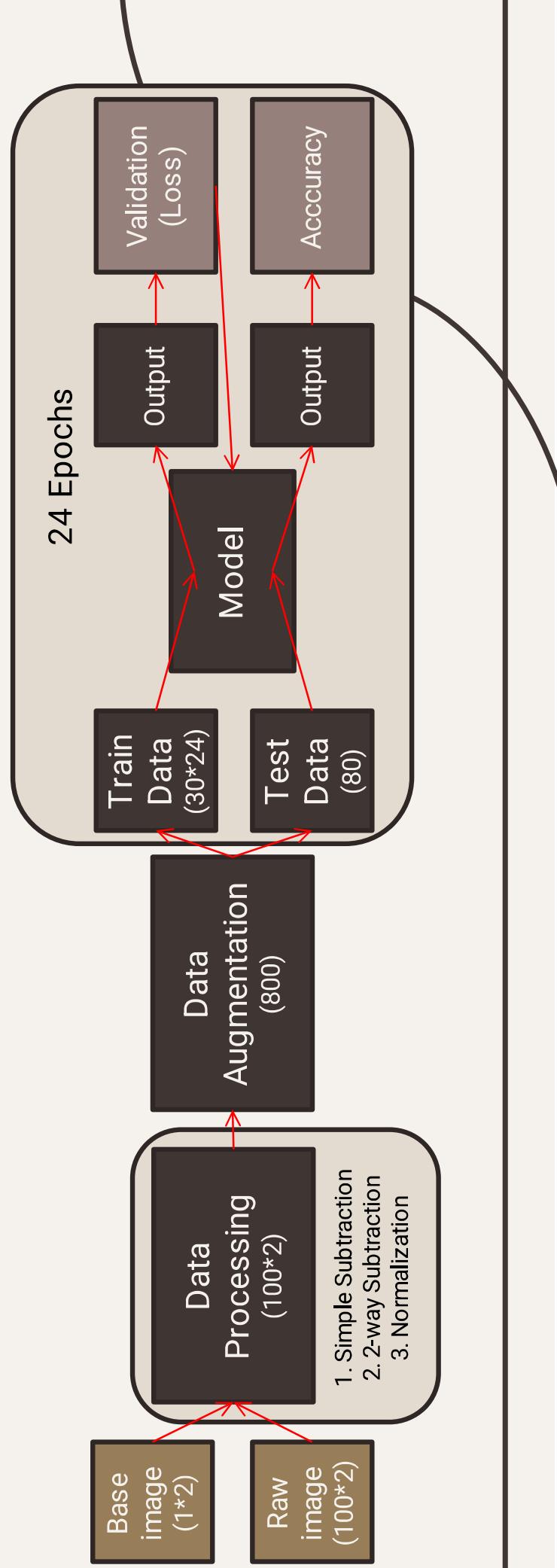
적은 데이터를 가지고 높은 성능을 내는 Chimeramix 모델
오픈 소스를 수정하여 구현하는데에 어려움 (개발역량 부족)

ResNet56 모델 활용하여 우선구현

학부수업에서 활용했던 코드를 수정하는 것이기에 다루기 용이
ImageNet에서 높은 성과를 보여왔던 모델
많은 연산량으로 인한 긴 학습시간 (모델 선정 이후 알게됨)

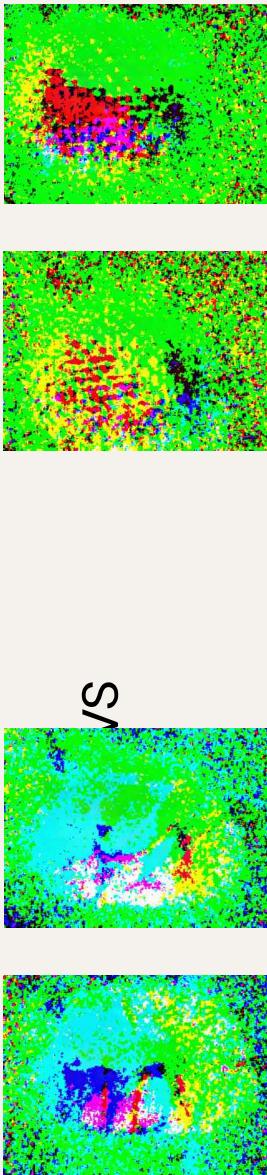
05. Image Classification

다양한 방법으로 동전과 안경과 구분하는 image classification 시도를 해봄



06. Simple Subtraction

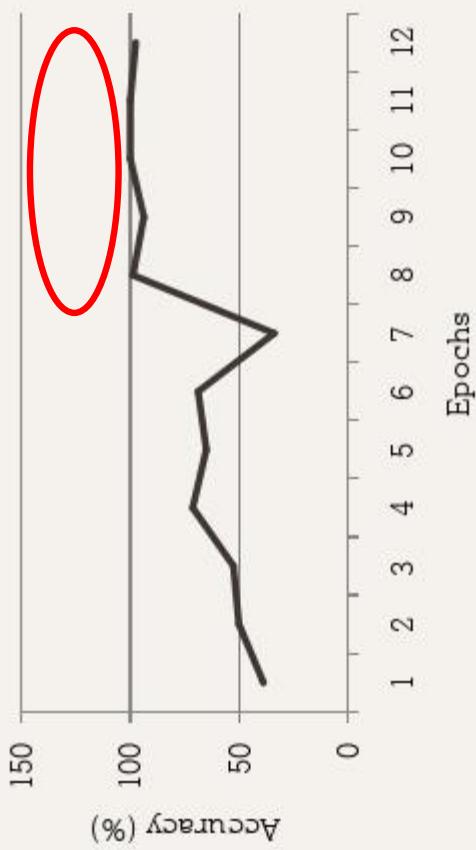
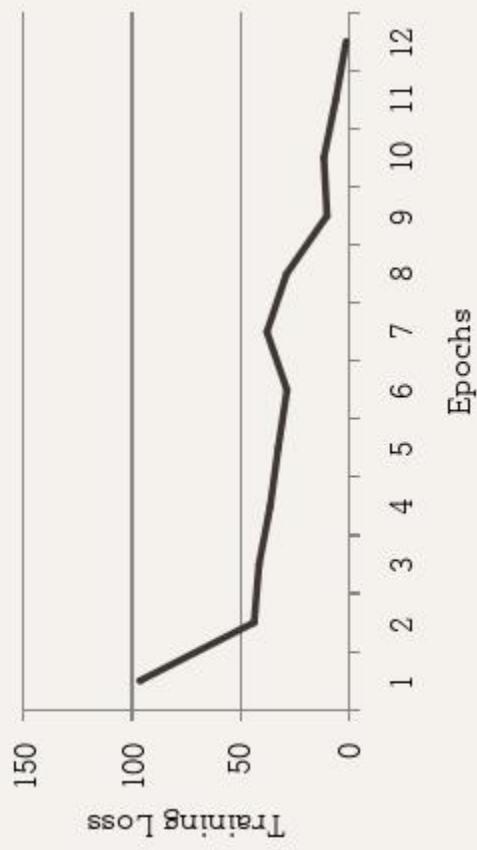
set1의 동전과 set1의 안경꼴을 구분하는 코드 resnetver2.py 구현
각 100개의 이미지, data augmentation으로 400개로 확장, 360개 train / 40개 test



특이사항
한장의 이미지를 학습하는데에 12~13초라는 매우 오랜 시간이 걸린다
작은 데이터와 epoch 수임에도 불구하고 총 2시간 이상의 오랜 시간이 걸림

더 가벼운 model 적용의 필요성이 있다

06. Simple Subtraction

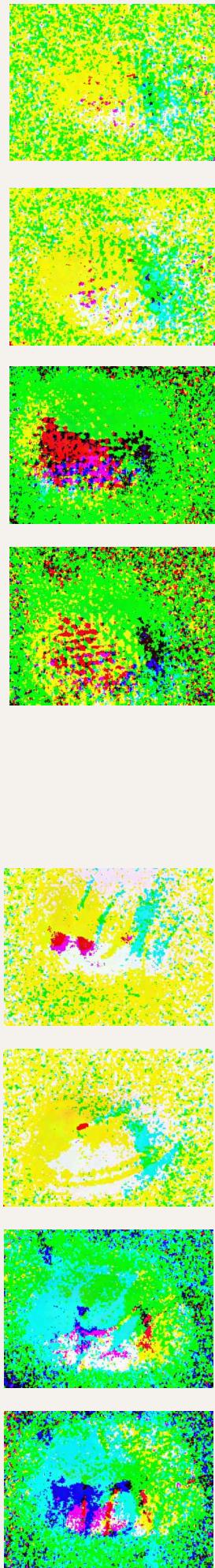


결과: Training Loss는 수렴하는 추세를 보였고, 8 epoch부터는 정답률이 100% 근처로 수렴
해석: 적은 데이터로 coin set1과 glasscase set1의 데이터를 구분 가능

다만 육안으로 확인할 때에도 data의 전체적인 색감이 다르기에, 물질의 특성에 의한 구분이
아니, 색감에 의존한 구분일 가능성이 높다.

06. Simple Subtraction

set1 & 2의 동전과 set1&2의 안경팔을 구분하는 코드 resnetver3.py 구현

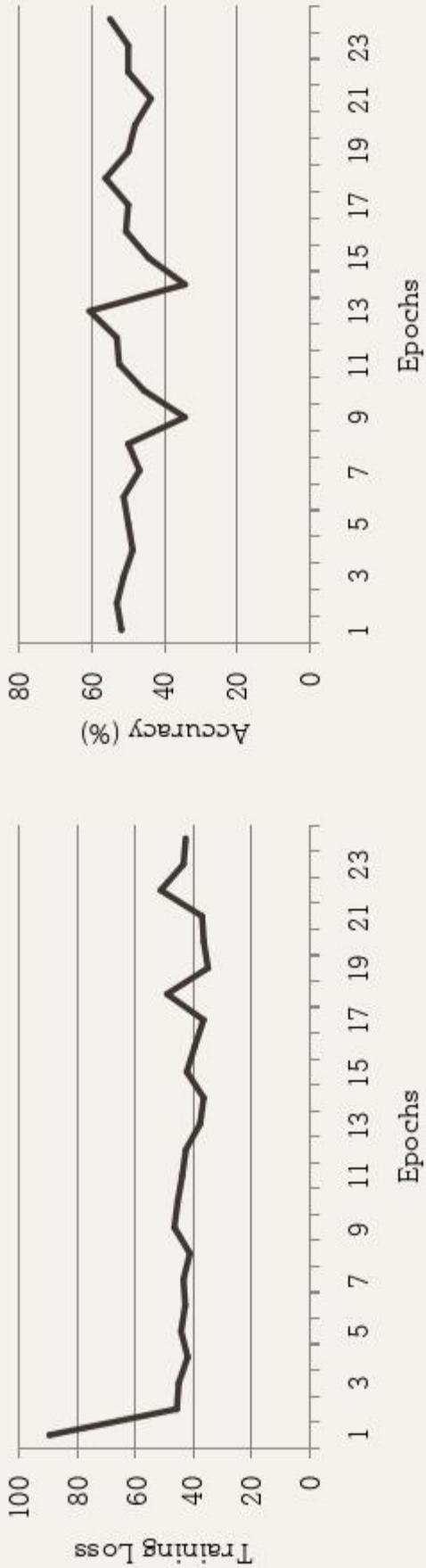


각 200개의 이미지, data augmentation으로 800개로 확장, 7207개 train / 807개 test

Learning rate=0.1 사용

역시 | 오랜 연산시간 소요 (5시간 이상)

06. Simple Subtraction



결과: Training Loss는 epoch 2 이후 쉽게 수렴하지 못함. 정답률 역시 50% 균형으로, 학습 실패

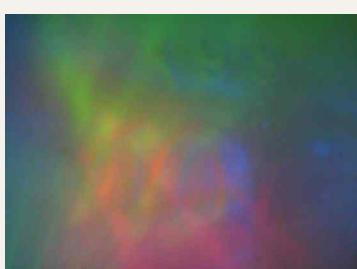
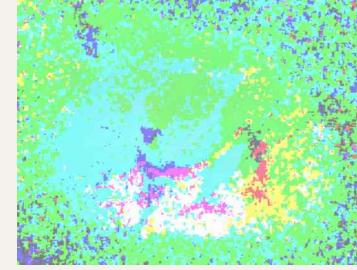
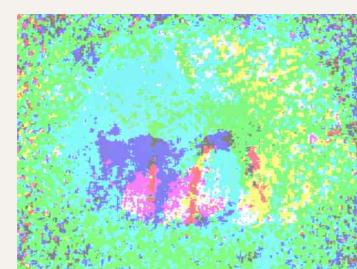
해석: set 1&2의 coin data 와 set 1&2의 안경과 데이터를 구분하지 못한다.
다만 learning rate 등의 optimization variable 조정을 통해 수렴이 가능하게 만들어질 수 있는 여지는 존재

Learning Rate 수정 VS 새로운 Data Processing 도입

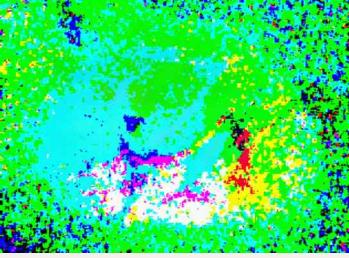
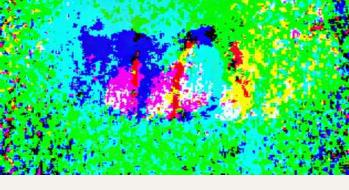
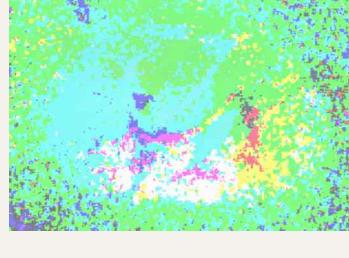
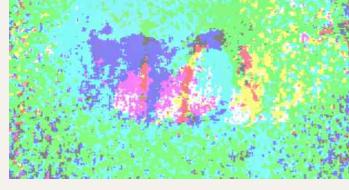
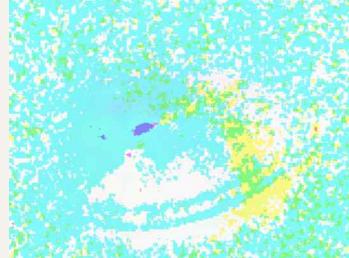
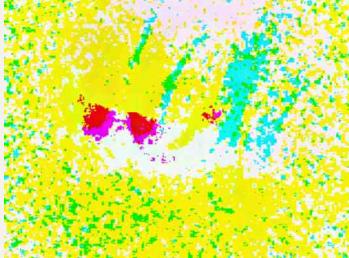
07. 2-Way Subtraction

- 최대 문제점: 데이터 후처리(raw data-base image) 과정에서 각 trial마다 특유의 색감을 낙타냄
- 센서데이터를 reliable하게 얻을 수 있는 방법을 찾아내는 것이 중요
 - 개인의견으로는 센서의 품질문제로 의심됨
 - 현 상황에서 시도해볼 수 있는 개선방안들 위주로 시도
- 개선방안: 2-Way Subtraction
 - Simple Subtraction은 raw data에서 base image를 그대로 뺀 값을 사용
 - raw data값이 base image보다 rgb값이 작을 경우 그 차이가 얼마인지에 무관하게 0으로 사용됨: 유의미한 데이터 손실
 - $r_{ss,i} = r_{raw,i} - r_{base,i}$ (*Simple Subtraction*)
 - $\Rightarrow r_{2s,i} = 127 + \frac{r_{raw,i} - r_{base,i}}{2}$ (*2-way Subtraction*)

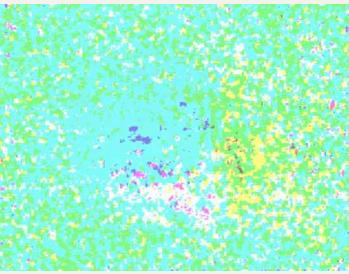
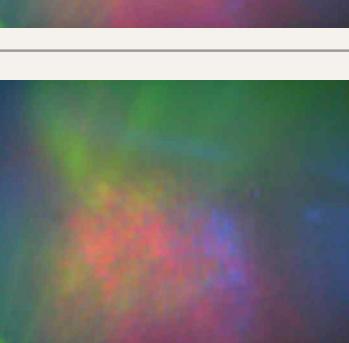
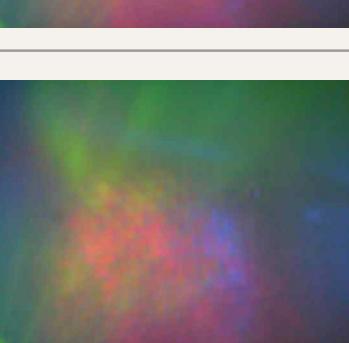
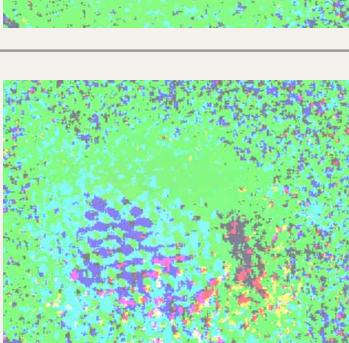
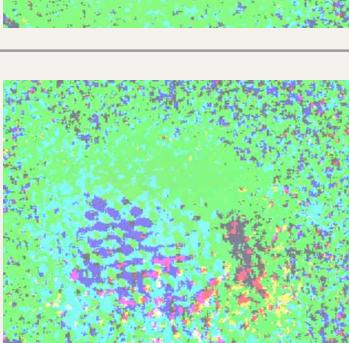
07. 2-Way Subtraction (동전)

1 번째 set의 base image	Raw data 10	Raw data 72	2 번째 set의 Base image	Raw data 110	Raw data 172
					
					

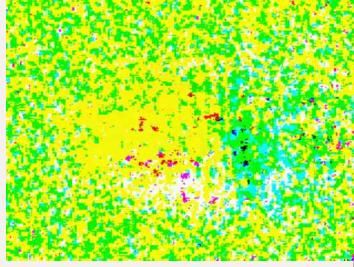
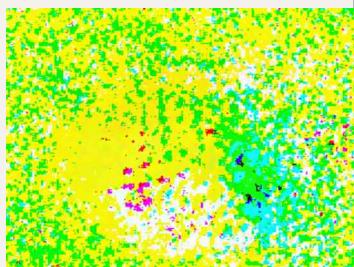
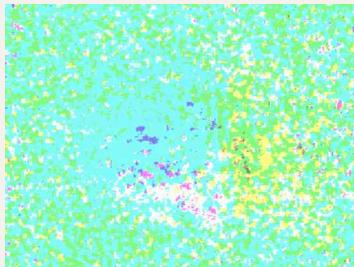
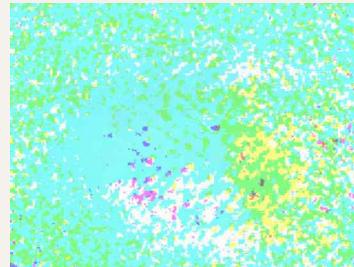
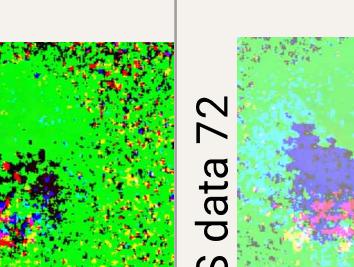
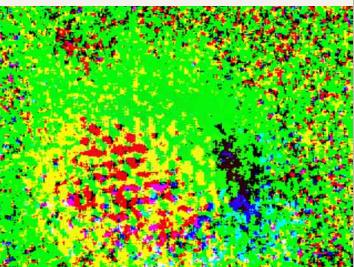
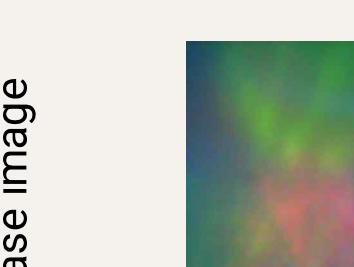
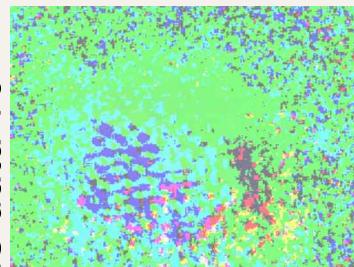
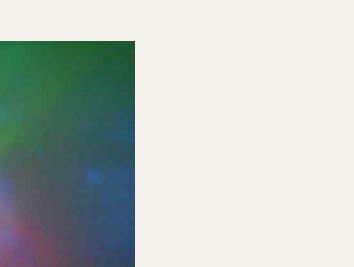
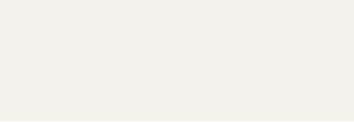
07. 2-Way Subtraction (동전)

1번짜 set의 base image	SS data 10	SS data 72	2번짜 set의 Base image	2S data 110	SS data 110	SS data 172
						

07. 2-Way Subtraction (한경각)

1번짜 set의 base image	Raw data 10	Raw data 72	2번짜 set의 Base image	Raw data 110	Raw data 172
					
					

07. 2-Way Subtraction (한경각)

1번 째 set의 base image	SS data 10	SS data 72	2번 째 set의 Base image	SS data 110	SS data 172
					
					

07. 2-Way Subtraction

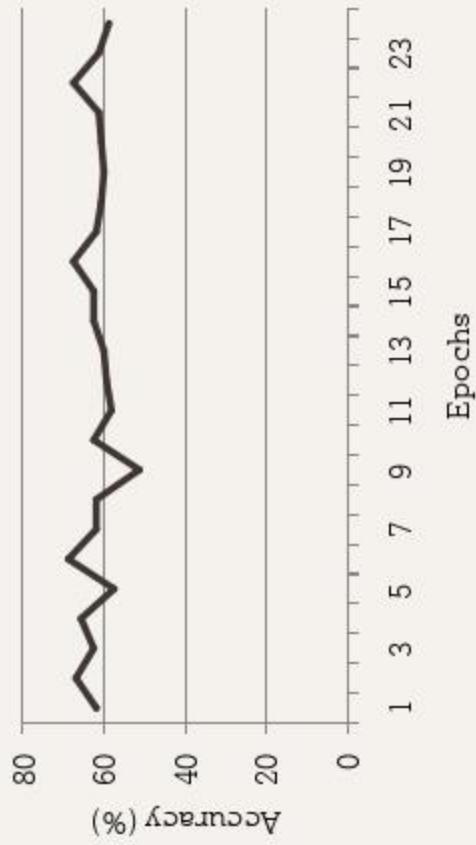
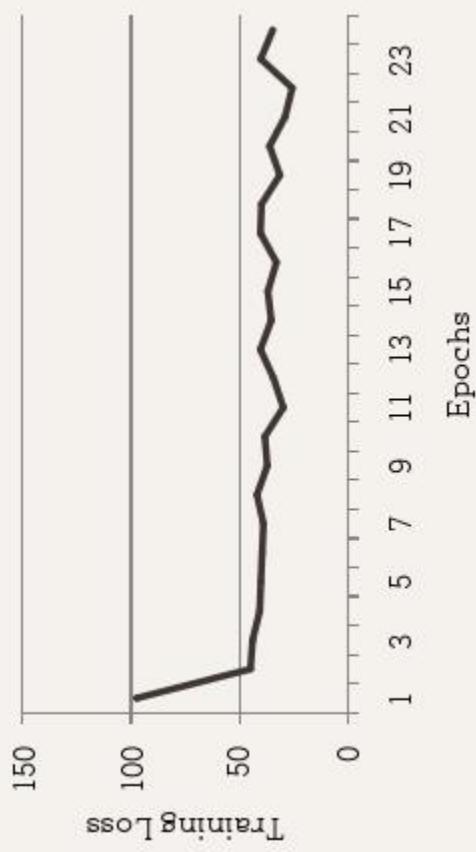
색감 차이 존재하지만, 전보단 데일한 차이

set1&2의 동전과 set1&2의 안경팔 데이터를 구분하는 코드 resnetver4.py 구현
Simple Subtraction 대신 2-Way Subtraction 적용
타 조건 전과 동일



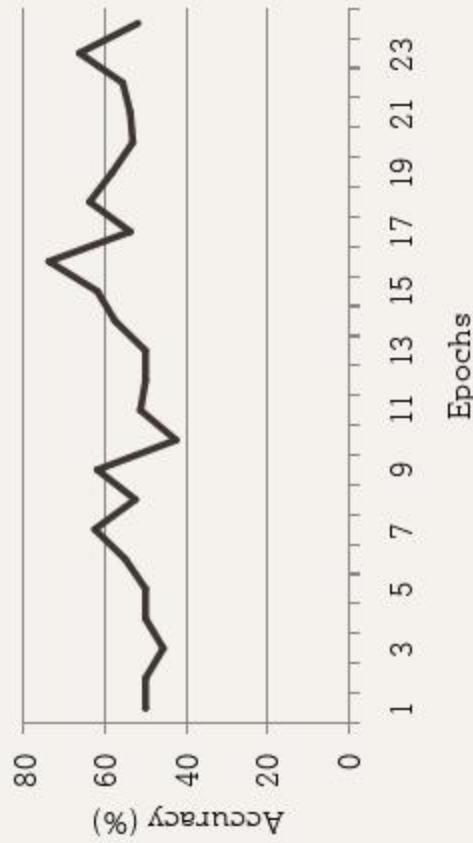
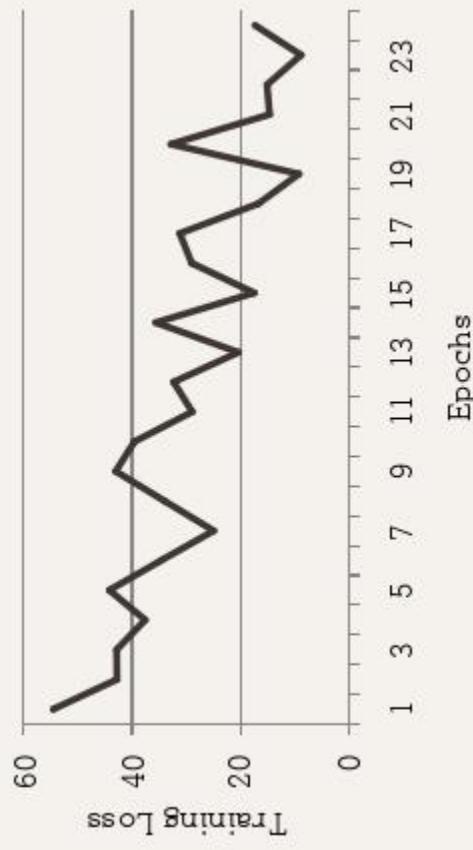
역시 오랜 연산시간 소요 (6시간 이상)

07. 2-Way Subtraction



결과: Simple Subtraction에 비해 loss가 조금 더 낮고 수렴하는 추세. 정답률은 평균 60%정도이지만 높아지는 추세는 아님.
해석: Loss 수렴이 덜 되는 것으로 분석. Learning rate를 0.1배 하여 재시도
초반부터 60%라는 유의미하게 높은 정답률을 보이는 것이 의문점

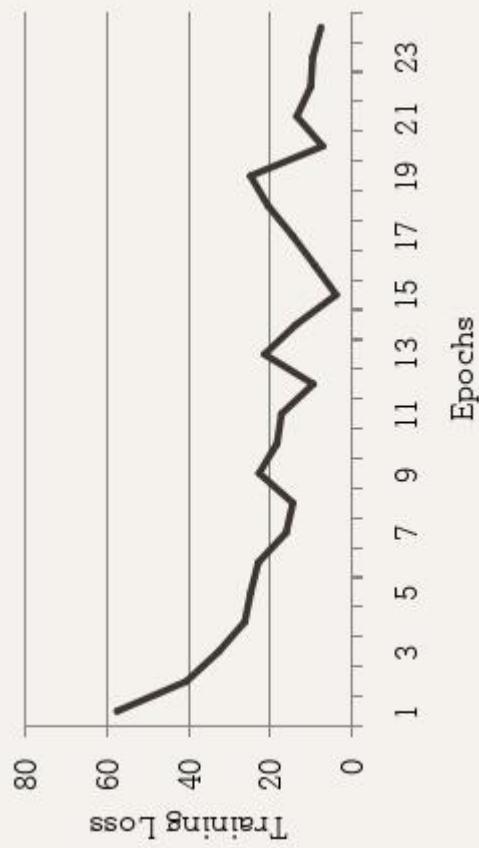
07. 2-Way Subtraction



결과: lr=0.1일 때에 비해 더 낮아지는 loss. 그러나 아직 수렴하지 않았음. 정확도는 약간씩 증가하는 추세지만 lr=0.1 일 때의 비정상적인 높은 정답률 보단 낮음

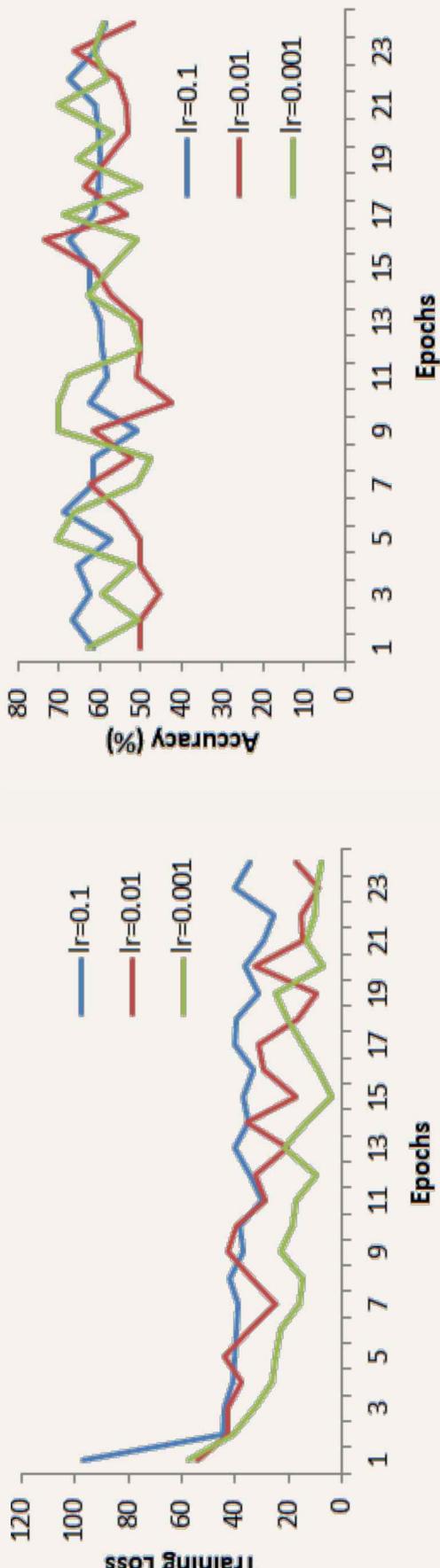
Learning rate를 0.1 배 하여 재시도

07. 2-Way Subtraction



결과: lr=0.01 일때에 비해 Loss가 더 수렴. 정확도도 소폭 증가하지만 유의미한 수치는 아님.

07. 2-Way Subtraction

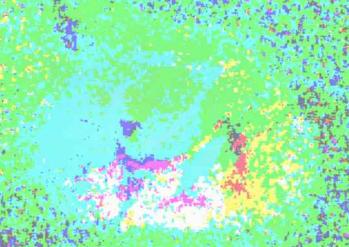
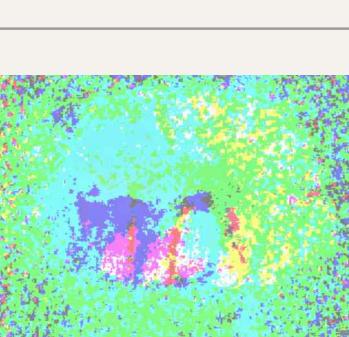
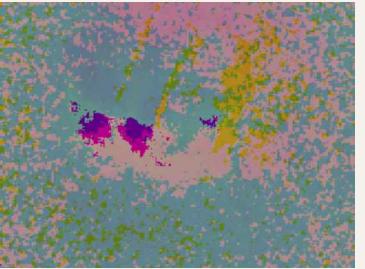
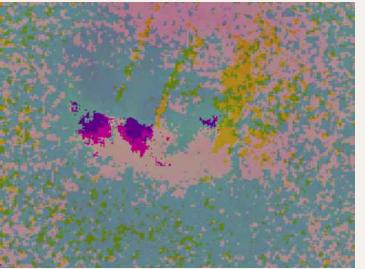


learning rate가 작을수록 Training Loss가 더 잘 수렴하는 경향을 보임
lr=0.1 일땐 Accuracy가 초반부터 높은 추이를 보임
이는 초기 parameter가 우연히 잘 설정된 원인으로 추정
lr이 작을수록 accuracy가 더 높지만, 그 차이가 뚜렷하지 않음
Data Processing 과정 수정

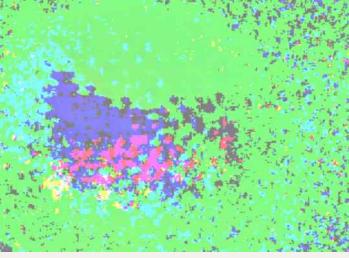
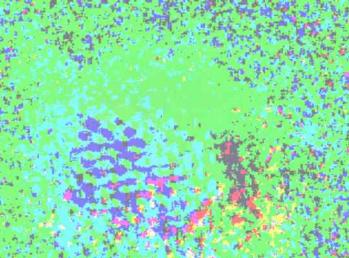
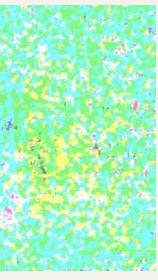
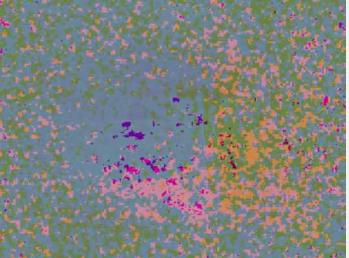
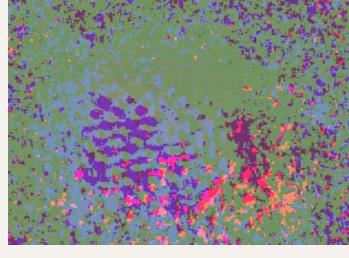
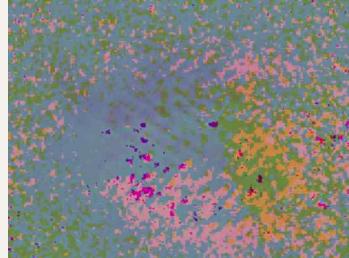
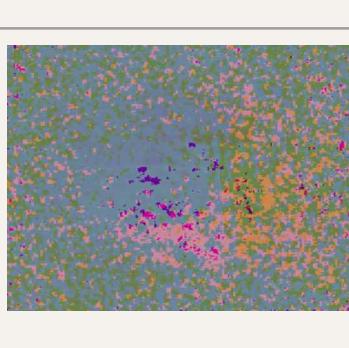
08. Normalization

- 앞선 data processing 과정에선 data간의 색감차이가 육안으로 확인됨
 - 색감에 의존한 data classification이 이뤄지면 새로운 측정data에 대한 활용이 어려움
 - Classification 정확도 역시 낮아짐
 - 결국 중요한 것은 픽셀 색깔 그 자체가 아닌 주변 픽셀과의 차이
- Data Normalization을 진행하자
 - 모든 이미지의 전체픽셀 평균값을 하나(127,127,127), 표준편차를 40으로 통일
 - 표준편차 40은 3σ 범위에 대부분의 표본이 들어갈 것이라는 가정하에 설정
 - Data processing 상황을 보고 변경가능
 - $r_{2s,i} = 127 + \frac{r_{raw,i} - r_{base,i}}{2} (2-way Subtraction)$
 - $(r_{2s,i}, g_{2s,i}, b_{2s,i})$ 의 $320 * 240 * 3$ 데이터의 평균을 $(r_{avg}, g_{avg}, b_{avg})$, 표준편차를 $(r_{std}, g_{std}, b_{std})$
 - $r'_{n,i} = 127 + 40 \times \frac{r_{2s,i} - r_{avg}}{r_{std}}$ (cf. 고등학교 확률과 통계 정규분포의 표준화)

08. Normalization (동전)

1 번째 set의 base image	2S data 10	2S data 72	2 번째 set의 Base image	2S data 110	2S data 172
					

08. Normalization (안경 꽉)

1 번째 set의 base image	2S data 10	2S data 72	2 번째 set의 Base image	2S data 110	2S data 172	Norm data 10	Norm data 72	Norm data 110	Norm data 172
									

08. Normalization

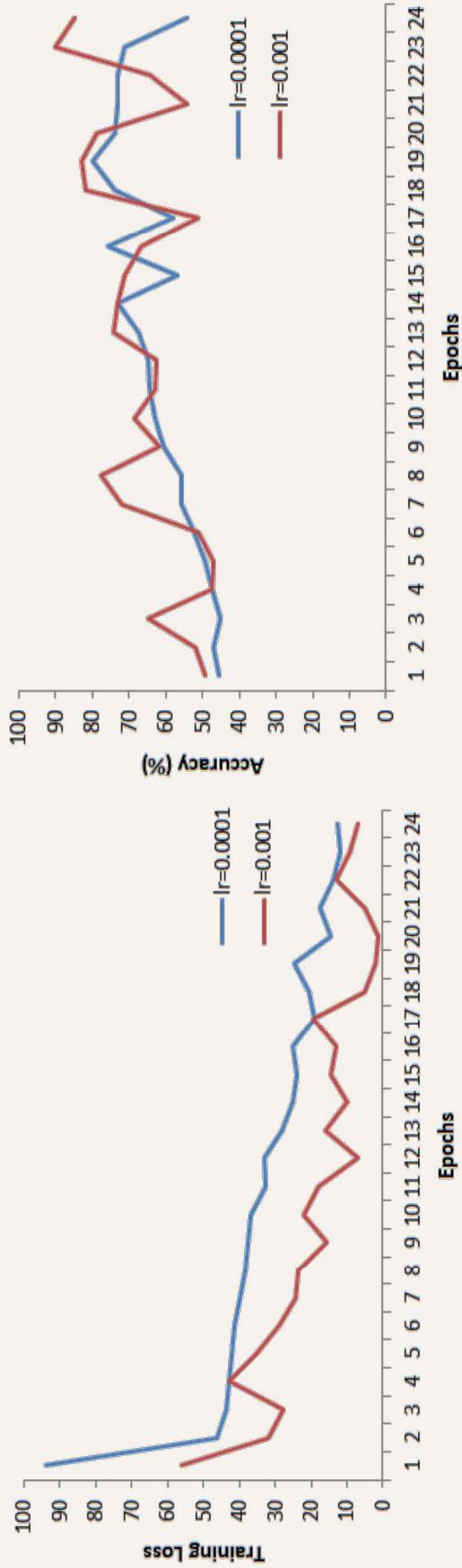
Set 1/2의 색감이 비슷해짐
Rgb값이 [0, 255]의 범위를 초과하는 픽셀수를 세어도 1%이내로 std=40을 관찰한 결과로
판단됨

set1&2의 동전과 set1&2의 외경곽을 구분하는 코드 resnetver5.py 구현
Data Normalization 적용
타 조건 전과 동일



오랜 연산시간 소요 (9시간 이상, 두 모델 동시가동)

08. Normalization

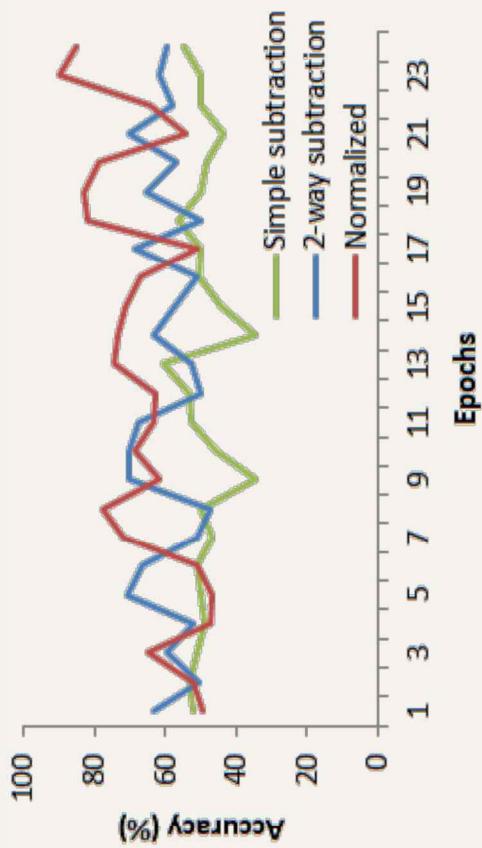
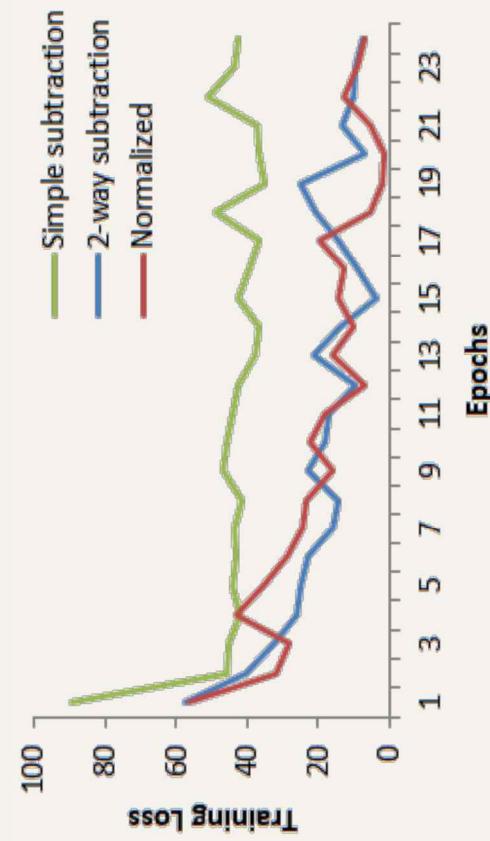


결과: 확실하게 줄어들며 수렴하는 loss, 최대 90%까지 올라가는 높은 정확도. 정확도가 안정적으로 높진 않음.

Learning Rate=0.0001도 시도해보았으나, 수렴하기 전에 데이터를 다 써버림.

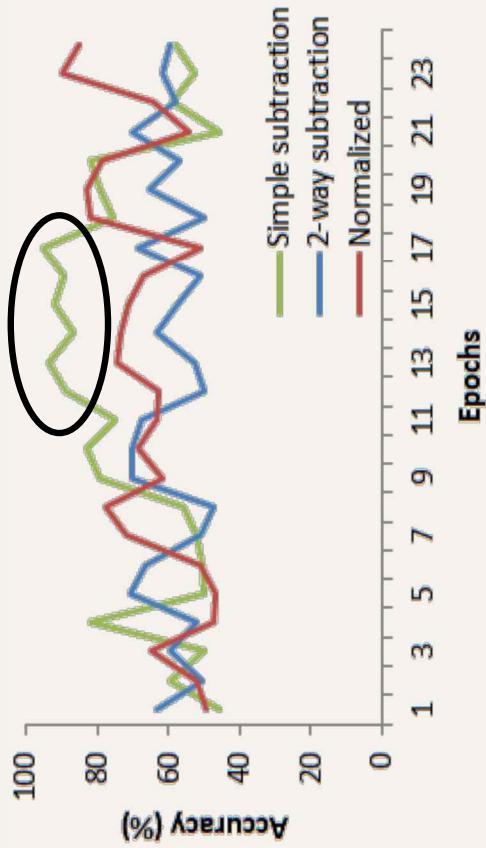
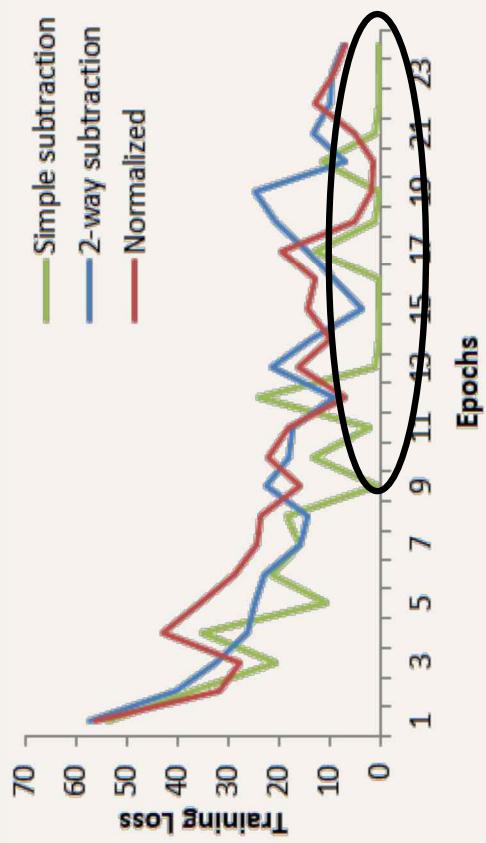
해석: Data normalization을 한 이미지를 기반으로 학습하는 것이 더 높은 정확도를 나타낸다.

08. Normalization



normalize하지 않은 데이터에 비해 normalize된 데이터가 비슷한 loss 수렴에도 불구하고 더 높은 정확도를 보임
학습된 데이터를 기반으로 다른 임의의 데이터도 잘 맞추기 용이하다는 의미
normalized 된 데이터를 기반으로 학습 / test data 역시 normalize 이후에 연산할 필요
Simple Subtraction의 단순 전처리 데이터는 loss가 수렴하지 않음 (lr 조정필요)

09. Unexpected Results



Simple Subtraction 역시 $\|r=0.001\)$ 로 조정해서 진행해봄
훨씬 더 오정적으로 잘아지는 LOSS 및 90%를 상회하는 높은 정확도를 학습 도중 보임
초후 날은 Loss임에도 다시 급락한 정확도 역시 확인 가능

10. Application on New Dataset

결국 실제 로봇에 사용되기 위해선 다양한 종류의 센서의 여러 set의 데이터를 활용할 수 있어야 함

현재까지의 data는 1개의 센서의 2set의 데이터만 활용

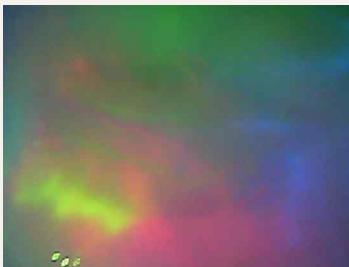
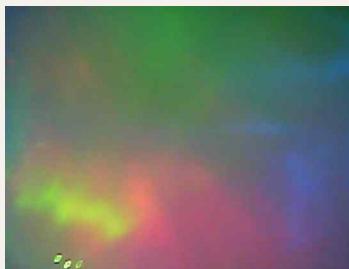
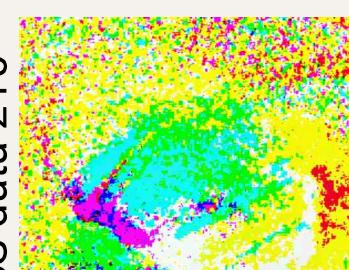
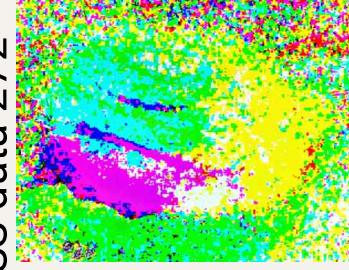
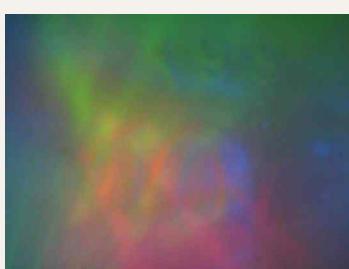
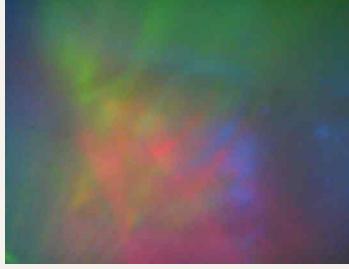
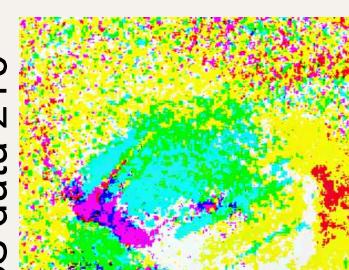
새로운 센서를 활용하여 각 100개의 데이터를 추가 수집

학습은 기존 200개의 데이터로 하여 새로운 100개의 데이터를 구분할 수 있을지를 실험

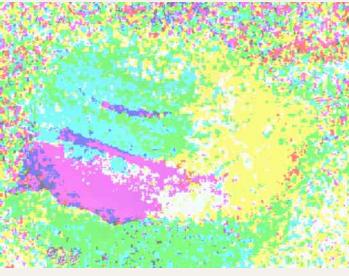
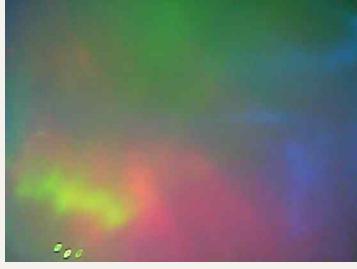
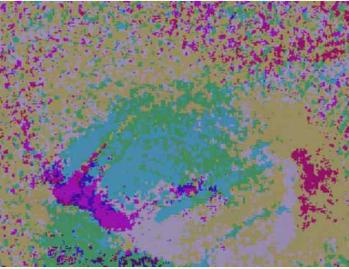
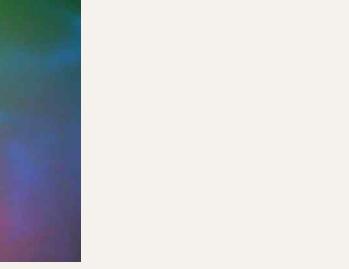
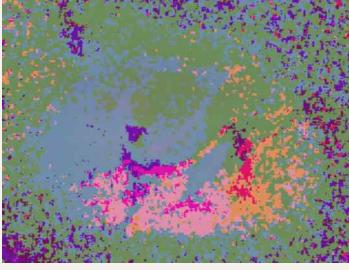
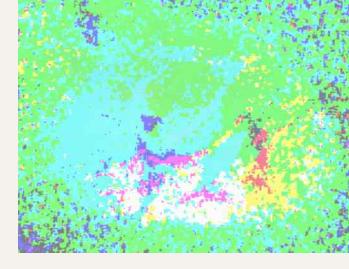
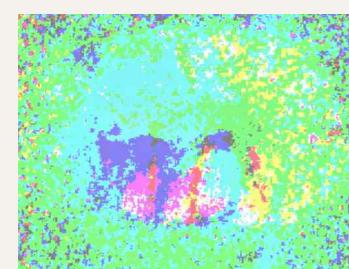
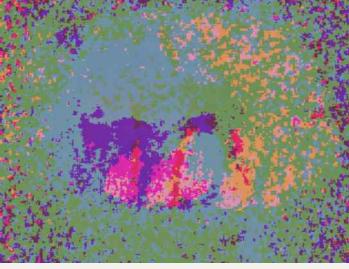
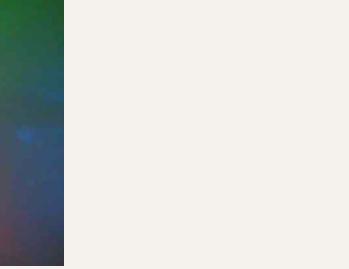
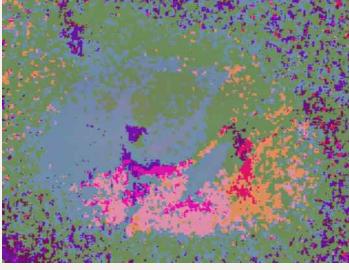
시간관계상 Simple Subtraction 및 Normalization만 측정하여 비교.

6번만 Accuracy 측정

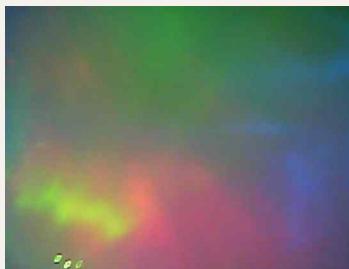
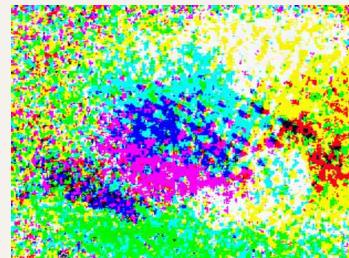
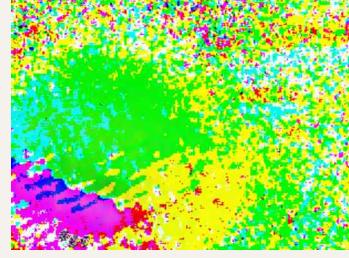
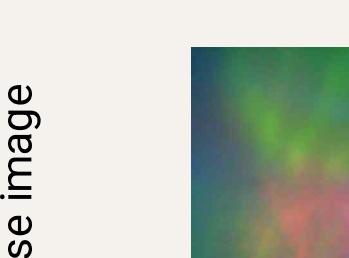
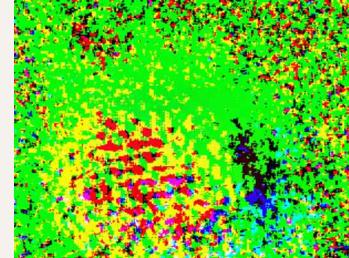
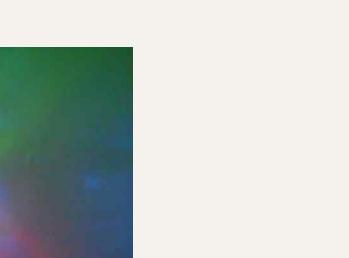
10. Application on New Dataset (동전)

1 번째 set의 base image	Raw data 10	Raw data 72	3 번째 set의 Base image	Raw data 210	Raw data 272
					
					

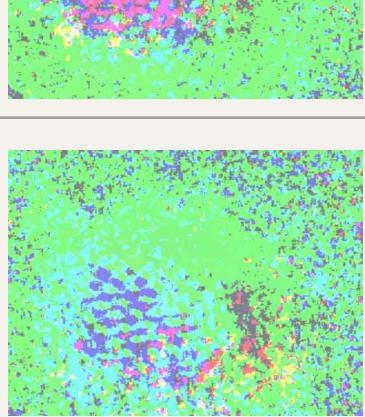
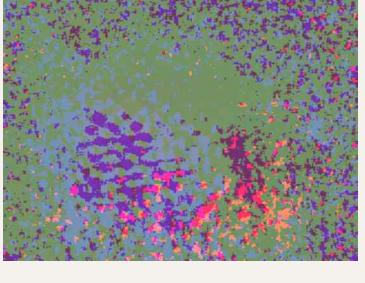
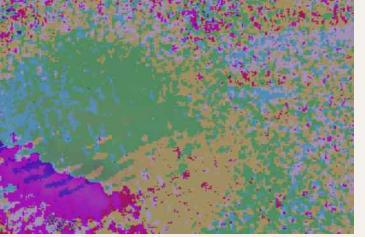
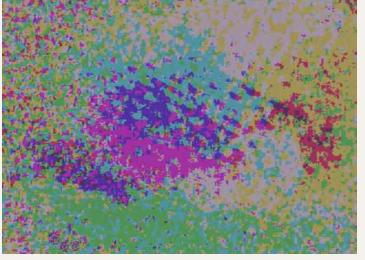
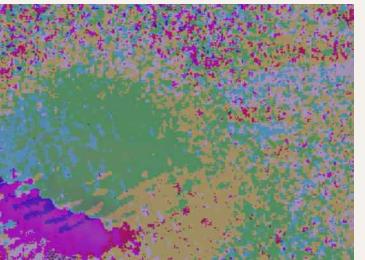
10. Application on New Dataset (Σ_{74})

1 번째 set의 base image	2S data 10	2S data 72	3 번째 set의 Base image	2S data 210	Norm data 272
					
					

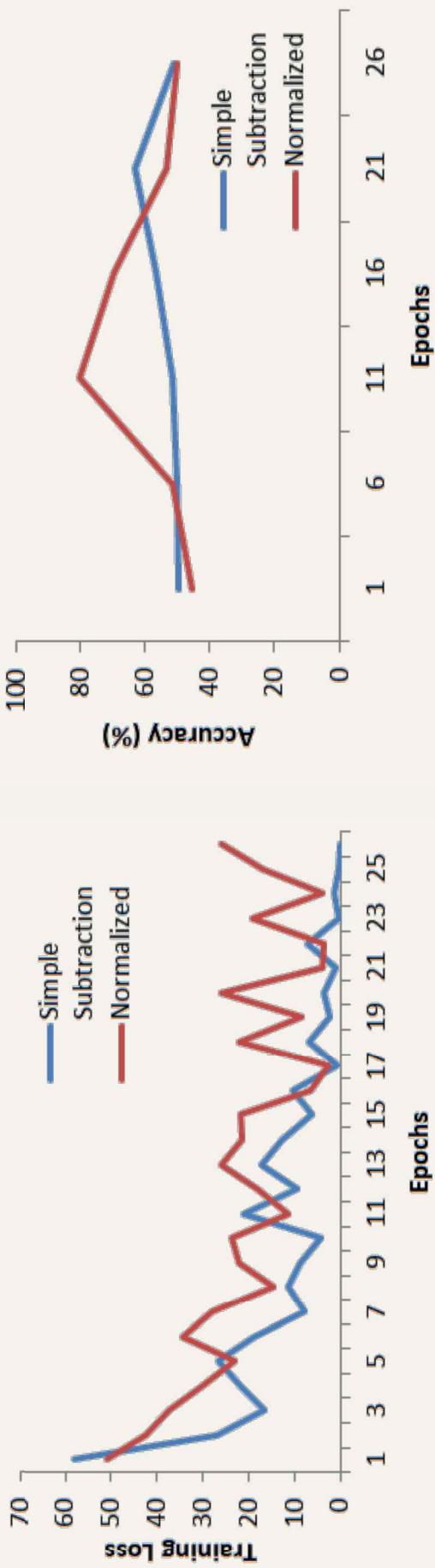
10. Application on New Dataset (안경 꽈)

1 번째 set의 base image	Raw data 10	Raw data 72	3 번째 set의 Base image	Raw data 210	Raw data 272
					
SS data 10	SS data 72	SS data 210	SS data 272		
					

10. Application on New Dataset (안경 꽈)

1 번째 set의 base image	2S data 10	2S data 72	3 번째 set의 Base image	2S data 210	2S data 272	Norm data 10	Norm data 72	Norm data 210	Norm data 272
									

10. Application on New Dataset



Normalized Train loss가 수렴하지 않았음: 이전과 같은 데이터 사용했기에 일시적인 문제
결과: Simple Subtraction의 경우 Loss가 수렴했음에도 불구하고 낮은 정확도를 나타냄.
Normalized의 경우 Loss가 수렴하지 않았지만, 높은 정확도를 나타내는 구간 조차

해석: learning rate 조절 & data 수 증가로 Normalized data를 늘려야 함.
이때 Normalized data의 정확도가 더 높아질 것으로 기대

11. Future Works & Conclusion

Digit 측정 데이터의 안정성만 확보되면 적은 데이터로도 동전 / 안경곽의 구분이 가능함

Digit은 측정시마다 같은 센서로 측정을 하여도 색감이 다르게 나오는 등 안정성에 문제가 있음

Data Normalization을 활용하면 Digit의 안정성 문제를 일부 보완할 수 있음.

그러나 데이터 당 5초이상의 시간이 소요됨

ImageNet에 잘 사용되는 ResNet56은 Digit 이미지 Classification에도 활용 가능

그러나 5초의 시간이 소요되며, 정확도는 최대 80% 정도

더 정밀하고 안정적인 센서 및 더 가볍고 강력한 모델이 필요할 것으로 생각됨

Thank You