프로젝트 계획서

분리수거 척척박사 - 2조

과목명	졸업프로젝트2
담당교수	김두현 교수님
학과	공과대학 컴퓨터공학부
학번 및 이름	201814116 권태윤
	201613187 송용수
	201613188 송태인

목차

1. 시작 반 작품에 대한 재분석

1학기에 진행했던 프로토타입의 주안점은 다음과 같다.

첫째, 재활용 쓰레기 사진 데이터 셋 부족

둘째, Object Detection 모델의 정확도

셋째, 분류 모델과 안드로이드와의 연동

프로토타입 모델을 개발하며 대부분의 문제점을 해결하였지만 같은 카테고리에도 다양한 모양이 존재한다는 점이 정확도를 높이기 어려웠다. 또한, 캔이나 건전지와 같이 비슷한 모양들도 많으며 빨대의 경우에도 구부림 유무에 따라서도 정확도 차이가 많이 났다. 프로토타입 모델에서는 아직 정확도 면에서 부족한 면이 많았다.

따라서 해당 카테고리에 대해서는 추가적인 데이터셋을 수집하고 다양한 각도에서 바라본 물체의 사진을 수집할 생각이다. 데이터셋이 부족한 경우 직접 사진을 찍어 부족한 부분을 채우거나 기존 사진에 회전과 같은 변화를 준 데이터를 추가로 학습시킬 예정이다.

2. 변경 (개선 내역)

1학기에 진행했던 프로토타입과 큰 틀은 같으며 분리수거 분류 항목 개수만 6개에서 10개로 확장할 계획이다.

기존에는 빨대, 컵라면, 페트병, 캔, 종이컵, 건전지 6가지 항목이었지만 젖병, 고무 장갑, 우유팩, 아이스 팩과 같이 일반인들이 잘 알지 못하는 분리수거 항목들에 대해서 추가했다.

기존 프로토타입 모델에서는 분류 결과만 나왔지만 이번 2학기 프로젝트에서는 분류 결과 뿐만 아니라 자세한 분리수거 방법도 제공해 분리수거하기 어려운 사람들에게 도움이 되는 어플리케이션이 될 예정이다.

하지만 기존 6개에서 10개로 카테고리가 추가돼 학습시키는 양이 늘어 모델도 무거워 질것으로 예상된다. 따라서 다양한 모델 중 만족할 만한 정확도가 나오고 무게도 가벼운 모델로 변경할 예정이다. 또한, 학습시킬 때에도 배치사이즈를 조정하는 등 알고리즘을 변경해 학습시킬 예정이다. 부족한 데이터셋은 회전이나 반전을 통해 확장 시키고 정규화해 모델을 가볍게 만들 예정이다.

학습시키는 사진이 많고 카테고리 별로도 모양이 다양해 상당한 시간과 노력이 필요할 것으로 예상된다.

3. 대표 시연 시나리오

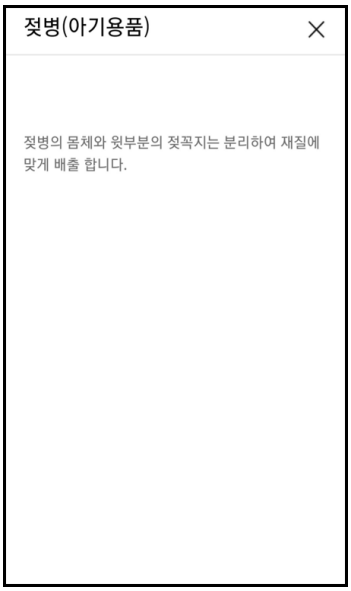
1. 앱 실행 후 "사진 선택" 버튼을 클릭한다.

2. 재활용품 이미지를 업로드할 업로드 방식을 선택한다.

   (새로운 사진 촬영 or 앨범에서 불러오기)

3. 업로드한 이미지의 분류 결과를 확인한다.

4. "상세한 분리수거 Tip" 버튼을 클릭해 해당 항목에 대한 분리수거 방법을 확인한다.



4. 문제 식별 및 해결 전략

첫번째 문제점으로는 재활용 쓰레기 데이터셋이 부족하다는 것이다. 문제를 해결하기 위해 하나의 사진에 대해 여러 조도, 명도, 각도의 사진을 촬영하며 부족한 경우 직접 촬영을 통해 데이터셋을 보강할 생각이다.

두번째 문제점으로는 Object Detection 모델의 정확도를 들 수 있다. 이번 프로젝트의 분류 모델은 같은 카테고리 사이에서도 명확한 모양이 존재하지 않고 다양한 모양이 존재한다. 예를 들어 페트병 중 하나인 플라스틱 음료수병만 해도 기업들마다 다양한 디자인으로 생산한다. 이러한 문제를 해결하기 위해 최대한 많은 데이터 셋을 확보할 것이며 가이드라인을 추가로 제공해 정확도를 높일 생각이다. 모델 또한, 학습 후 가장 정확도가 높게 나온 모델에 대해서 사용할 예정이다.

세번째 문제점으로는 학습된 모델과 안드로이드 어플리케이션 사이에서 잘 연결되어야 한다는 것이다. 모델의 정확도를 높이기 위해 데이터 셋을 추가해 학습시키는 경우 모델이 무거워져 어플리케이션 내에서 결과가 나오기까지 지연 시간이 길어질 수 있다. 학습시킬 때 배치 사이즈를 늘리거나 사진을 정규화 하는 방법이 있으며 또는, 가벼운 모델로 변경해 학습시키거나 최악의 경우 사진의 개수를 줄이고 최대한의 성능을 뽑을 것으로 예상되는 사진들만 골라 학습시킬 예정이다.

또한, UI 부분과 분리수거 방법을 알려주는 자세한 분리수거 TIP 기능과 잘 연결하는 것이 관건이다.

5. 시스템 구축 계획

학습 시스템 : Google Colab / tensorflow 2.5.0 version

인식 시스템 : Android phone (minSdkVersion 28 / targetSdkVersion 30)

6. 소프트웨어 구현 계획

1. 활용도구 : Android Studio / Google Colab / Git / TensorFlow Lite

2. 오픈소스 : Tensorflow Lite (https://www.tensorflow.org/lite/guide?hl=ko)

TensorFlow Lite는 개발자가 모바일, 내장형 기기, IoT 기기에서 모델을 실행할 수 있도록 지원하여 기기 내 머신러닝을 사용할 수 있도록 하는 도구 모음이다.

주요 특징

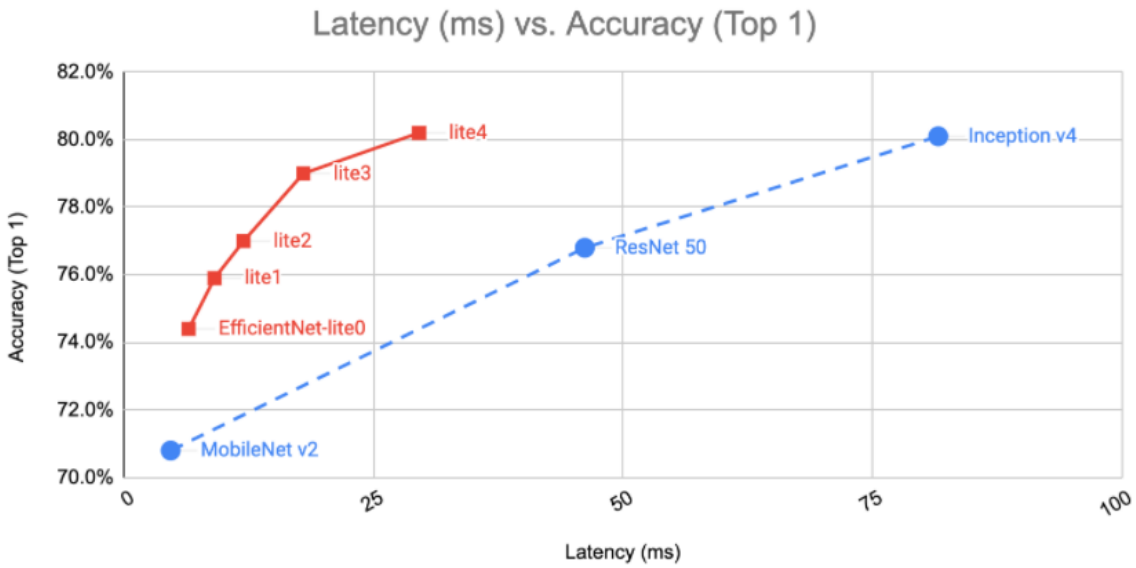
- 기기 내 머신러닝에 최적화됨, 5가지 핵심 제약사항 해결: 지연 시간(서버까지의 왕복 없음), 개인 정보 보호(기기에 개인 정보를 남기지 않음), 연결성(인터넷 연결이 필요하지 않음), 크기(모델 및 바이너리 크기 축소), 전력 소비(효율적인 추론 및 네트워크 연결 불필요)
- 여러 플랫폼 지원: Android 및 iOS 기기, 내장형 Linux 및 마이크로 컨트롤러 등
- 다양한 언어 지원: 자바, Swift, Objective-C, C++, Python 등
- 고성능: 하드웨어 가속 및 모델 최적화 사용
- 포괄적인 예: 다양한 플랫폼에서의 일반적인 머신러닝 작업(예: 이미지 분류, 객체 감지, 자세 추정, 질문 답변, 텍스트 분류 등)

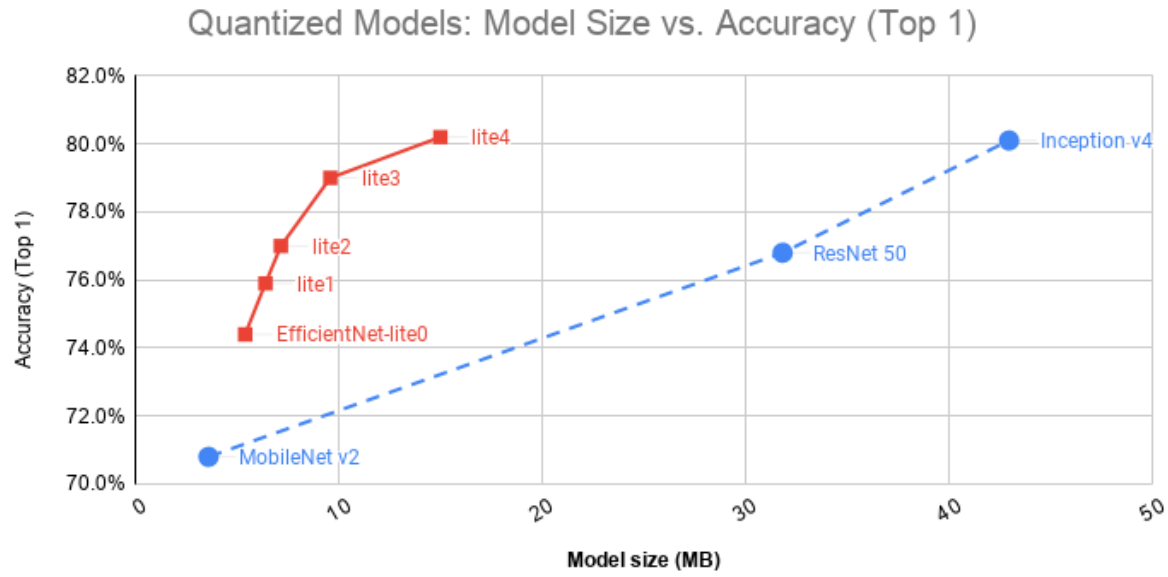
3. 데이터

- 구글링을 통한 분리수거 이미지 데이터셋 수집
- 부족한 사진의 경우 직접 촬영

4. 알고리즘 : EfficientNet-Lite

EfficientNet-Lite는 계산량과 매개변수 수를 현격하게 줄이면서도 최고의 정확도를 달성하는 이미지 분류 모델 모음이다. EfficientNet-Lite 모델은 양자화를 포함한 TensorFlow Lite에 맞게 최적화되어 있어 정확도 손실을 무시할 수 있을 정도의 수준에서 더 빠르게 추론할 수 있고 CPU, GPU 또는 Edge TPU에서 실행 가능하다.





Model	params	MAdds	FP32 accuracy	FP32 CPU latency	FP32 GPU latency	FP16 GPU latency	INT8 accuracy	INT8 CPU latency	INT8 TPU latency
efficientnet-lite0 ckpt	4.7M	407M	75.1%	12ms	9.0ms	6.0ms	74.4%	6.5ms	3.8ms
efficientnet-lite1 ckpt	5.4M	631M	76.7%	18ms	12ms	8.0ms	75.9%	9.1ms	5.4ms
efficientnet-lite2 ckpt	6.1M	899M	77.6%	26ms	16ms	10ms	77.0%	12ms	7.9ms
efficientnet-lite3 ckpt	8.2M	1.44B	79.8%	41ms	23ms	14ms	79.0%	18ms	9.7ms
efficientnet-lite4 ckpt	13.0M	2.64B	81.5%	76ms	36ms	21ms	80.2%	30ms	-

MobileNetV2, ResNet 50 들과 비교시 Efficientnet-lite2 모델이 용량 대비 정확도에서 훨씬 더 나은 결과를 가져와 Efficientnet-lite2모델을 사용했다.

또한, 모든 EfficientNet-Lite모델들을 사용해본 결과 표와 달리 분리수거 데이터셋에서는 EfficientNet-Lite2가 다른 모델에 비해 학습속도 및 정확도, 용량이 가장 최적이라고 판단해 EfficientNet-Lite V2를 사용했다.

$$
\begin{aligned}
\text{depth: } & d = \alpha^\phi \\
\text{width: } & w = \beta^\phi \\
\text{resolution: } & r = \gamma^\phi \\
& \text{s.t. } \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2 \\
& \alpha \geq 1, \beta \geq 1, \gamma \geq 1
\end{aligned}
$$

CNN의 성능을 높일 수 있는 요소로는 Depth (d), Width (w), Resolution (r) 3가지가 있다.

EfficientNet 모델은 기본적으로 width scaling과 depth scaling, resolution scaling 3가지의 방식을 통해 모델을 학습시키며 EfficientNet의 알파, 베타, 감마 값은 간단한 grid search를 통해 구하는 방식을 제안하고 있으며, 처음 단계에서는 파이를 1로 고정한 뒤, 타겟 데이터셋에서 좋은 성능을 보이는 알파, 베타, 감마 값을 찾아낸다.

Table 2. **EfficientNet Performance Results on ImageNet** (Russakovsky et al., 2015). All EfficientNet models are scaled from our baseline EfficientNet-B0 using different compound coefficient ϕ in Equation 3. ConvNets with similar top-1/top-5 accuracy are grouped together for efficiency comparison. Our scaled EfficientNet models consistently reduce parameters and FLOPS by an order of magnitude (up to 8.4x parameter reduction and up to 16x FLOPS reduction) than existing ConvNets.

Model	Top-1 Acc.	Top-5 Acc.	#Params	Ratio-to-EfficientNet	#FLOPS	Ratio-to-EfficientNet
EfficientNet-B0	**76.3%**	**93.2%**	**5.3M**	**1x**	**0.39B**	**1x**
ResNet-50 (He et al., 2016)	76.0%	93.0%	26M	4.9x	4.1B	11x
DenseNet-169 (Huang et al., 2017)	76.2%	93.2%	14M	2.6x	3.5B	8.9x
EfficientNet-B1	**78.8%**	**94.4%**	**7.8M**	**1x**	**0.70B**	**1x**
ResNet-152 (He et al., 2016)	77.8%	93.8%	60M	7.6x	11B	16x
DenseNet-264 (Huang et al., 2017)	77.9%	93.9%	34M	4.3x	6.0B	8.6x
Inception-v3 (Szegedy et al., 2016)	78.8%	94.4%	24M	3.0x	5.7B	8.1x
Xception (Chollet, 2017)	79.0%	94.5%	23M	3.0x	8.4B	12x
EfficientNet-B2	**79.8%**	**94.9%**	**9.2M**	**1x**	**1.0B**	**1x**
Inception-v4 (Szegedy et al., 2017)	80.0%	95.0%	48M	5.2x	13B	13x
Inception-resnet-v2 (Szegedy et al., 2017)	80.1%	95.1%	56M	6.1x	13B	13x
EfficientNet-B3	**81.1%**	**95.5%**	**12M**	**1x**	**1.8B**	**1x**
ResNeXt-101 (Xie et al., 2017)	80.9%	95.6%	84M	7.0x	32B	18x
PolyNet (Zhang et al., 2017)	81.3%	95.8%	92M	7.7x	35B	19x
EfficientNet-B4	**82.6%**	**96.3%**	**19M**	**1x**	**4.2B**	**1x**
SENet (Hu et al., 2018)	82.7%	96.2%	146M	7.7x	42B	10x
NASNet-A (Zoph et al., 2018)	82.7%	96.2%	89M	4.7x	24B	5.7x
AmoebaNet-A (Real et al., 2019)	82.8%	96.1%	87M	4.6x	23B	5.5x
PNASNet (Liu et al., 2018)	82.9%	96.2%	86M	4.5x	23B	6.0x
EfficientNet-B5	**83.3%**	**96.7%**	**30M**	**1x**	**9.9B**	**1x**
AmoebaNet-C (Cubuk et al., 2019)	83.5%	96.5%	155M	5.2x	41B	4.1x
EfficientNet-B6	**84.0%**	**96.9%**	**43M**	**1x**	**19B**	**1x**
EfficientNet-B7	**84.4%**	**97.1%**	**66M**	**1x**	**37B**	**1x**
GPipe (Huang et al., 2018)	84.3%	97.0%	557M	8.4x	-	-

We omit ensemble and multi-crop models (Hu et al., 2018), or models pretrained on 3.5B Instagram images (Mahajan et al., 2018).

위 표는 ImageNet에 대한 실험 결과이다. 기존 ConvNet들에 비해 EfficientNet 모델은 비슷한 정확도를 보이면서 parameter수와 FLOPS 수를 굉장히 많이 절약할 수 있는 것을 알 수 있다. 또한, 기존에 ImageNet 데이터셋에서 가장 높은 정확도를 달성했던 GPipe 보다 EfficientNet 모델은 더 높은 정확도를 달성하는 것을 확인할 수 있다. 동시에 parameter 수와 FLOPS수도 굉장히 많이 절약할 수 있어 굉장히 좋은 결과를 보여주는 것을 바로 알 수 있다.

7. 시험 계획 (시험 항목, 통과 조건)

<기능>

1. 처음 사용하는 사용자도 쉽게 사용할 수 있는가?
2. 앱 실행 시 메인 화면이 잘 실행되는가?
3. 메인 화면에서 '사진 선택' 버튼 클릭 시 사진 업로드 화면이 잘 실행되는가?
4. 사진 업로드 화면에서 '새로운 사진 촬영' 버튼 클릭 시 카메라가 잘 실행되는가?
5. 사진 업로드 화면에서 '앨범에서 불러오기' 버튼 클릭 시 갤러리가 잘 실행되는가?
6. 카메라를 통해 찍은 사진이 결과화면에 잘 나타나는가?
7. 갤러리에서 업로드한 사진이 결과화면에 잘 나타나는가?
8. 결과 화면에서 '상세한 분리수거 Tip' 버튼 클릭 시 분류 결과와 일치하는 재활용 품목의 분리수거 Tip 화면이 잘 나타나는가?
9. 메인 화면에서 '분리수거 Tip' 버튼 클릭 시 카테고리 화면이 잘 실행되는가?
10. 카테고리 화면에서 특정 카테고리 선택 시 해당 카테고리와 일치하는 재활용 품목의 분리수거 Tip 화면이 잘 나타나는가?

<모델 성능>

통과 조건은 모두 70%의 정확도로 정하였다.

1. 모든 카테고리별로 최소 요구 정확도를 만족하는가?
2. 같은 물체에 대해 다른 각도에서 동일한 결과를 출력하는가?
3. 한 물체에 대해서 다양한 형태를 동일하게 인식하는가?

8. 일정

주차	날짜	활동내용
1주차	08-30 ~ 09-04	데이터 셋 수집
2주차	09-06 ~ 09-11	데이터 셋 라벨링
3주차	09-13 ~ 09-18	분류 모델 학습 + 계획서 작성
4주차	09-20 ~ 09-25	분류 모델 학습 + 계획서 작성
5주차	09-27 ~ 10-02	분류 모델 학습 + 모델 안드로이드 탑재
6주차	10-04 ~ 10-09	자세한 분리수거 항목 추가
7주차	10-11 ~ 10-16	오류 해결
8주차	10-18 ~ 10-23	어플리케이션 UI개선 + 중간 발표
9주차	10-25 ~ 10-30	어플리케이션 1차 수정
10주차	11-01 ~ 11-06	어플리케이션 2차 수정
11주차	11-08 ~ 11-13	어플리케이션 3차 수정
12주차	11-15 ~ 11-20	시스템 테스트
13주차	11-22 ~ 11-27	시스템 성능 개선 + 오류 수정
14주차	11-29 ~ 12-04	최종 보고서 작성 + 시스템 안정화
15주차	12-06 ~ 12-11	최종 보고서 작성 + 최종 발표 준비
16주차	12-13 ~ 12-18	최종 발표 및 시연 + 최종 보고서 마무리

9. 팀원 역할 분배

- 권태윤 : 데이터셋 구축, 모델 설계, 모델 학습

- 송용수 : 데이터셋 구축, 라벨링 작업, 모델 학습

- 송태인 : 데이터셋 구축, 안드로이드 어플리케이션 개발, UI/UX 디자인