



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

석사학위논문

모바일과 딥러닝 서버 기반의 멸종위기 종의
종판별을 위한 컴퓨터 비전 시스템 개발

A Study of Development for Computer Vision System to Classify
Endangered Species based on Deep Learning Server and Mobile

상명대학교 대학원

컴퓨터과학과 컴퓨터과학전공

최 대 규

2021년 2월

석사학위논문

모바일과 딥러닝 서버 기반의 멸종위기 종의
종판별을 위한 컴퓨터 비전 시스템 개발

A Study of Development for Computer Vision System to Classify
Endangered Species based on Deep Learning Server and Mobile



상명대학교 일반대학원

컴퓨터과학과 컴퓨터과학전공

최 대 규

2021년 2월

모바일과 딥러닝 서버 기반의 멸종위기 종의
종판별을 위한 컴퓨터 비전 시스템 개발

A Study of Development for Computer Vision System to Classify
Endangered Species based on Deep Learning Server and Mobile

지도교수 김 동 근

본 논문을 석사학위 논문으로 제출함



상명대학교 일반대학원

컴퓨터과학과 컴퓨터과학전공

최 대 규

2021년 2월

최 대 규의
석사학위 논문을 인준함

심사위원장 _____ 임좌상 ①인

심사위원 _____ 조용주 ①인

심사위원 _____ 김동근 ①인



상명대학교 대학원

2021년 2월

차 례

표 차례	i
그림 차례	ii
국문 요약	iii
1. 서론	1
1.1. 연구배경	1
1.1.1. 영상 데이터 모니터링 시스템의 필요성	1
1.1.2. 기존 연구 및 시스템의 한계점	2
1.1.3. 앱 시스템의 필요성	3
1.1.4. 멸종 위기 앵무새 종 현황	4
1.2. 연구목적	7
1.3. 연구구성	7
2. 관련 연구	8
2.1. 멸종위기 종 영상 데이터를 이용한 딥러닝 분류 연구 사례	8
2.2. 머신러닝 모델 적용 연구 사례	9
2.2.1. 단일 클래스 분류 모델 적용 연구	9
2.2.2. 앱 및 웹에서 딥러닝 모델 적용 연구	10
2.3. 컴퓨터 비전 네트워크 아키텍처 연구	10
3. 시스템 설계 및 개발	12
3.1. 시스템 구성 및 흐름도	12
3.1.1. 온 디바이스(On-Device) 시스템	12
3.1.2. 모바일 및 딥러닝 서버 시스템	13
3.2. 시스템 개발 환경	17
4. 모델 설계 및 개발	18

4.1. 데이터 준비	18
4.1.1. 동물원 영상 데이터	18
4.1.2. 크롤링 영상 데이터	19
4.1.3. 어그멘테이션 영상 데이터	21
4.2. 모델 구축	23
4.2.1. One-class SVM	23
4.2.2. Multi-class CNN	26
4.3. 모델 훈련	31
5. 모델 평가	32
5.1. 평가 데이터 준비	32
5.1.1. One-class SVM	32
5.1.2. Multi-class CNN	33
5.2. 평가 방법	34
5.2.1. One-class SVM	34
5.2.2. Multi-class CNN	35
6. 결과	37
6.1 시스템 개발 결과	37
6.1.1 모바일 및 딥러닝 서버 구축 결과	37
6.1.2 시스템 속도 측정 결과	38
6.2 모델 평가 결과	38
6.2.1. One-class SVM	38
6.2.2. Multi-class CNN	39
7. 결론	41
참고문헌	42
ABSTRACT	45

표 차 례

<표 1> 웹과 앱 인터페이스 비교	4
<표 2> CITES 멸종 위기 앵무새 5종	6
<표 3> 동물원 앵무새 영상 데이터 현황	19
<표 4> Confusion matrix	34
<표 5> 시스템의 각 요소별 소요시간	38
<표 6> Multi class CNN의 Confusion matrix	39
<표 7> Multi class CNN의 Precision, Recall, F1 score	40



그 립 차 례

<그림 1> 객체 인식 분야	3
<그림 2> 딥러닝 모델을 활용한 영상 이미지 분석 사례	9
<그림 3> One-class 전이 학습을 적용한 사례	9
<그림 4> 온 디바이스 시스템 블록다이어그램과 시퀀스다이어그램 ·	13
<그림 5> 모바일 및 딥러닝 서버 시스템 블록다이어그램	14
<그림 6> 모바일 및 딥러닝 서버 시스템 시퀀스다이어그램	14
<그림 7> 모바일 앱 화면 구성	14
<그림 8> Tensorflow Serving API 구성도	16
<그림 9> 동물원 영상 데이터 예시	19
<그림 10> 구글 이미지 섹션의 검색 예시	20
<그림 11> 데이터어그멘테이션 적용 예시	22
<그림 12> CNN 기반의 Multi class classification 잘못된 예	23
<그림 13> 서포트벡터머신	25
<그림 14> 주성분분석	25
<그림 15> Convolution 예시	27
<그림 16> 전이 학습 방법	29
<그림 17> Global average pooling 방법	31
<그림 18> One class SVM 테스트 이미지 데이터 셋	32
<그림 19> Multi class CNN의 테스트 이미지 데이터 셋	33
<그림 20> ROC AUC	35
<그림 21> 모바일 및 딥러닝 서버 작업 수행 결과	37
<그림 22> 클라우드 스토리지 저장 결과	38
<그림 23> One class SVM Confusion matrix 결과	39
<그림 24> Multi class CNN Loss 및 Accuracy	40

국 문 요 약

딥러닝 기반 멸종위기 종 분류 앱 시스템

본 논문에서는 딥러닝 기반으로 멸종위기 종의 영상 데이터를 분류하는 앱 시스템의 구축 과정을 제시한다. 대한민국 환경부 산하 국가생물다양성위원회의 2018 보고서에 의하면 생물종 다양성의 DB 구축과 모니터링 계획이 50% 미만이다. 또 인천 세관에서는 불법 밀수로 들어온 멸종위기 종을 판별하는데 어려움을 겪는다고 한다.

따라서 본 연구에서는 전문가가 판단하기 어려운 종판별을 컴퓨터가 객관적으로 수행할 수 있는 앱 시스템을 개발했다. 본 연구를 초석으로 대용량의 영상 데이터 판별을 자동화 하고 불법 밀수를 모니터링 하는 시스템이 구축될 수 있기를 기대한다. 기존 관련 연구의 한계점은 다음과 같다. CNN의 Softxmax 함수의 특성 상 분석 대상 이미지의 클래스가 정답 후보군에 없어도 정답 후보군 중에서 최대한 정답과 가까운 오답을 제시할 수밖에 없다. 본 시스템에서는 이런 점을 극복하기 위해서 분석 대상이 학습한 대상인지 아닌지를 먼저 판별하고 나서 학습 대상일 경우에만 종판별을 실시한다.

학습 대상인지 아닌지를 분류하는 단일 클래스 분류 모델의 정확도는 ROC AUC 0.9189이고 앵무새 종을 분류하는 다중 클래스 분류 모델의 정확도는 F1 score 0.91이다. 시스템 응답 속도는 이미지의 분석은 평균 5120ms이고 이미지 저장은 4220ms이다.

1. 서론

1-1. 연구 배경

1-1-1. 영상 데이터 모니터링 시스템의 필요성

대한민국 환경부의 국가생물다양성위원회에서 2018년에 발간한 보고서에 따르면 생물 다양성의 정의는 “지구상의 생물종 다양성, 생물종이 서식하는 생태계 다양성, 생물이 지닌 유전자 다양성”이라고 정의되어 있다 [1]. 이런 생물 다양성을 보존해야 하는 이유는 “다양한 생태계 서비스를 제공하고 지구와 생태계의 지속가능성 뿐만 아니라 국민의 삶에도 지대한 영향을 미치기 때문”이라고 한다. 최근 개발이라는 명목 하에 여러 생물 종의 서식지가 파괴되고 있고 기후가 변화되고 있으며 환경오염과 불법적인 남획의 증가로 인해서 1970년부터 2012년 사이에 척추동물 개체군이 58% 감소했다고 한다 [1]. 또, OECD에 의하면 향후 2050년 생물 다양성에 대해서는 2010년 대비하여 전 세계적으로 육상 생물 다양성이 약 10% 정도 감소될 것으로 예측된다고 한다 [2]. 이에 대해서 환경부에서, 2016년 생물다양성과 관련된 전략이행에 대한 중간점검을 시행한 결과, 일부 기초 DB구축, 모니터링 계획의 대부분 달성 정도가 50% 미만으로 낮은 성과를 보였다는 결론을 내렸다 [1]. 또, 인천 세관은 그동안 유해성이 있는 수입 생물이나 환경부의 허가가 필요한 멸종 위기 종에 대한 판단에 어려움을 겪었다고 한다 [3]. 이처럼 생물 다양성 보존을 위한 연구에는 영상 데이터 모니터링 시스템이 필요하고 이를 위해서 기술적 진보가 필요한 실정이다.

1-1-2. 기존 연구 및 시스템의 한계점

Lecun 등의 연구가 시발점이 되었던 [4] 딥러닝 분야에서 상당한 발전이 이뤄져서 다양한 분야에서 딥러닝 기술이 적용되고 있다. 딥러닝 기술은 음성 인식, 자연어 처리, 빅 데이터 분석 등 여러 분야로 나뉘지는데 그 중에서도 컴퓨터 비전 영역은 지금도 활발히 연구되고 있는 분야 중 하나이다. 의료 영상 데이터를 분석하기 위한 딥러닝 기술 적용과 관련된 연구가 진행되고 있고 [5, 6] 생체 인식을 위한 영상 데이터 분석에 딥러닝 기술을 적용하는 연구가 진행되고 있다 [7, 8]. 영상 데이터 내에서 관심 있는 객체의 영역을 조사하는 기술은 컴퓨터 비전 객체 탐지와 관련이 있다. 컴퓨터 비전에서 객체 인식 분야는 크게 네 가지 범주로 분류할 수 있다. 첫 번째로 영상 이미지 내의 객체의 종류를 단순히 분류하는 Classification (이하 분류), 두 번째로 영상 이미지 내의 객체의 종류뿐만 아니라 위치도 추적하는 Localization, 세 번째로 Classification과 Localization을 영상 내의 여러 객체에 대해서 수행하는 Object Detection, 네 번째로 이미지 상의 각 픽셀을 객체 단위로 분류하는 Semantic Segmentation이다 [9] (그림 1 참고). 객체 인식 분야가 연구되면서 응용되기 시작할 때는 상대적으로 구현하기 쉬운 난이도 때문에 분류를 적용한 사례들이 다수를 차지했다. 그러나 단순 분류의 경우 한 가지 문제를 내포하고 있다. 단순 분류가 출력하는 결과는 소프트맥스 함수를 통해서 계산한 정답 후보들의 확률 분포이다. 따라서 정답이 정답 후보 군에 없더라도 사용자가 최대한 근사한 후보를 답으로 해석할 여지가 있다. 이런 문제 때문에 사용자가 육안으로 식별이 힘든 데이터의 경우 잘못된 후속 처리를 할 가능성이 높다. 따라서 본 연구에서는 각 객체에 대해서 정답이 있을 경우에만 그 객체의 클래스에 대한 분류를 실시할 수 있도록 했다. 이에 따라 분류를 두 과정으로 나눈다. 첫 번째 분류 과정에서는 객체가 정답 후보군에 포함되어 있는 객체인지 아닌지를 판별하고 두 번째 분류 과정에서는 정답 후보군에 포함되어 있다면 해당 객체를 분류한다.

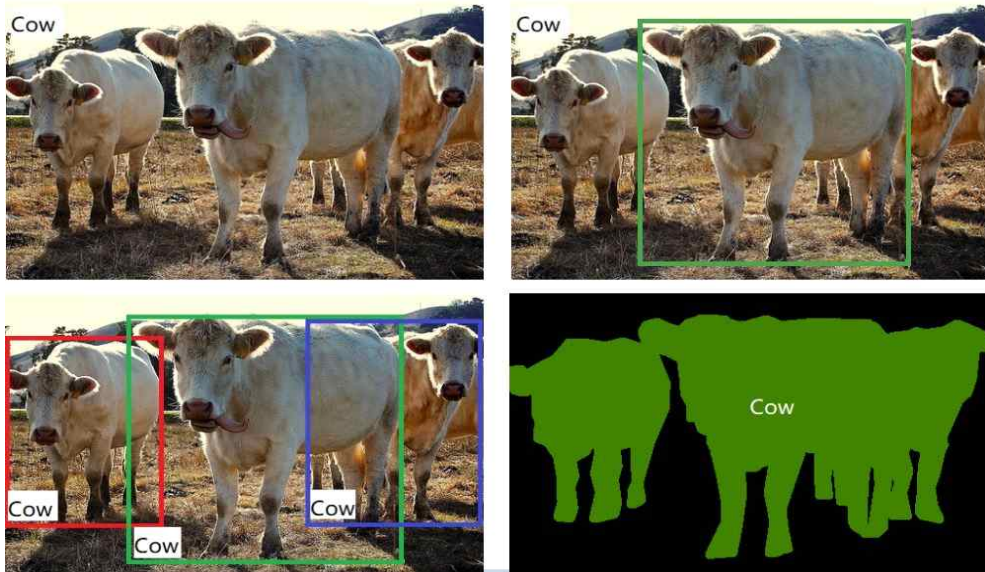


그림1. 객체 인식 분야. 왼쪽 상단은 Classification, 오른쪽 상단은 Localization, 왼쪽 하단은 Object Detection, 왼쪽 하단은 Semantic Segmentation을 나타낸다 [9].

1-1-3. 앱 시스템의 필요성

본 연구 시스템의 구조는 클라이언트(앱-카메라 인터페이스), 웹 서버(Flask), 딥러닝 서버(Tensorflow Serving API)로 구성되어 있다(3-1-2 참고). 클라이언트를 웹이 아닌 앱으로 구축한 이유는 다음과 같다. 첫째, 이미지를 획득하는데 용이하다. 구현의 차이가 있을 수 있지만 웹의 경우 이미지를 획득하기 위해서는 웹 카메라를 별도로 구입하여 촬영하고 업로드 해야 하는 번거로움이 있다. 반면 앱의 경우 모바일 디바이스에 기본적으로 카메라가 설치되어 있고 카메라를 활용하는 인터페이스를 사용자에게 제공하고 있으므로 이미지를 획득하는데 더 용이하다고 할 수 있다. 또 모바일 디바이스가 웹 카메라보다 더 휴대하기 좋다는 장점도 있다. 두 번째, 보안상의 이점이 있다. 웹 카메라의 경우의 해킹의 위험에 빈번하게 노출된다. 한상훈, 장진희, 강길욱, 박한술의 연구에 의하면 IP카메라의 제품인 QCAM3000, LG-LWW130W 제품에 크래킹을 시도한 결과 몇 분도 되지 않아서

크래킹에 성공했다고 한다 [10]. 앱 카메라의 경우, 소스 코드를 컴파일 하는 과정에서 카메라를 활용하는 코드가 있는데 권한 설정을 사용자에게 확인하는 코드가 없으면 컴파일 오류가 발생하기 때문에 강제적으로 사용자에게 카메라 사용 권한을 부여 받아야 한다. 그래서 웹 카메라에 비해 해킹에 대한 대처가 비교적 잘 되어 있다고 할 수 있다. 본 연구에서는 이런 이점 때문에 웹 인터페이스 대신에 앱 인터페이스를 고려하여 클라이언트를 구성하도록 개발되었다 (표 1 참고).

	웹 인터페이스	앱 인터페이스
편의성	<ul style="list-style-type: none"> • 별도의 카메라 구입 • 장비를 휴대하기 번거로움 	<ul style="list-style-type: none"> • 카메라 내장 • 휴대하기 쉬움
보안성	• 카메라 권한 설정 없음	• 카메라 권한 설정 필요

표1. 웹과 앱 인터페이스 비교. 행은 각 인터페이스의 종류를 나타내고 열은 비교 항목을 나타낸다.

1-1-4. 멸종 위기 앵무새 종 현황

본 연구는 멸종위기에 처한 야생 동·식물의 국제 거래에 관한 협약 (Convention on International Trade in Endangered Species of Wild Flora and Fauna, 이하 CITES)에서 지정한 멸종 위기 종 앵무새 5종을 중심으로 진행했다. Pires Stephen에 의하면 전 세계에 있는 330 앵무새 종중에서 36%가 멸종 위기에 처해 있다고 한다. 1991년부터 1996년까지 매해 240,000 개체의 앵무새들이 국제적으로 불법 밀렵 및 밀수에 피해를 봤다고 한다 [11]. CITES에서는 각 멸종 위기 종에 대해서 그 심각성에 따라 부속 서를 정해서 개체에 대한 거래를 규제하거나 조절하고 있다 [12]. 부속서는 3단계로 나뉘지며 각 단계는 다음과 같다.

- 부속서 I: 현재 무역이 중지되지 않으면 멸종될 위기에 처해 있는 약 1,000 여 종의 생물이 등록되어 있다. 여기에 속해 있는 생물 종은 특별한 예외를 제외하고 포획 및 수집 그리고 거래가 불법이다.

- 부속서 II: 멸종 위기는 아니지만 부속 서에 등재되어 있어서 관리가 되지 않을 경우 멸종 위기에 처해질 수 있는 약 34,000 여 종이 등록되어 있다. 거래 시에는 수출국의 허가가 필요하다.

- 부속서 III: 각 국가들이 지정하여 올라와 있는 생물종으로 모두 멸종위기에 놓인 종은 아니다. 수입을 위해서는 수출증명서와 증명서가 필요하다. 약 241 종이 등록되어 있다.

본 연구에서 주목한 CITES 등록 앵무새 5종은 각각 홍금강앵무, 청금강앵무, 큰유황앵무, 흰이마휴황앵무, 회색앵무이다. 금강앵무의 경우 성체 기준으로 몸길이가 약 89cm이다. 홍금강앵무와 청금강앵무는 몸 색깔에서 차이가 있고 생김새는 유사하다. 본 연구에서 색은 다르지만 유사한 형태의 객체를 분류할 수 있는지 실험 해 보기 위해서 두 앵무새를 선택했다. 큰유황앵무는 성체 기준으로 몸길이가 약 50cm이고 흰이마유황앵무는 약 31cm이다. 두 앵무새는 색은 유사하나 큰유황앵무의 경우 머리 깃이 달려있다. 본 연구에서 색은 유사하지만 형태가 다른 형태의 객체를 분류할 수 있는지 실험 해 보기 위해서 두 앵무새를 선택했다. 마지막으로 형태와 색이 전혀 다른 1종을 추가함으로써 5종의 앵무새 종을 분류할 수 있는 앱 시스템을 개발했다 (표 2 참고).






생김새		
학명	<i>Ara ararauna</i>	<i>Ara chloroptera</i>
국명	청금강앵무	홍금강앵무
부속서	II	II
생김새		
학명	<i>Cacatua galerita</i>	<i>Cacatua goffiniana</i>
국명	큰유황앵무	흰이마유황앵무
부속서	II	I
생김새		
학명	<i>Psittacus erithacus</i>	
국명	회색앵무	
부속서	I	

표2. CITES 멸종 위기 앵무새 5종.

1-2. 연구 목적

본 연구의 목적은 다음과 같다. 첫째, 생물 다양성 보존을 위한 연구 혹은 모니터링에 필요한 영상 데이터를 획득하여 처리하는 앱 시스템을 구축하여 영상 데이터를 객관적으로 분류하기 위함이다. 둘째, 영상 데이터에 단순 분류를 적용했을 때 발생하는 문제 중 하나인, 유사한 오답을 정답으로 출력하는 문제를 해결하기 위함이다. 이런 목적을 달성하기 위한 여러 가지 방법 중 1-1-3에서 기술한 이유로 모바일과 딥러닝 서버로 구성된 컴퓨터 비전 시스템을 개발했다.

1-3. 연구 구성

본 연구의 구성은 다음과 같다. 1장에서는 영상 데이터 모니터링 하는 시스템의 필요성과 이와 관련된 기존 연구 및 시스템의 한계점 그리고 앱 시스템의 필요성과 멸종 위기 종 그 중에서도 앵무새 종의 현황에 대해서 알아본다. 2장에서는 멸종 위기 종 이미지를 딥러닝으로 분류하는 연구 사례, 앱과 웹에서 딥러닝 모델을 적용하는 방법을 연구한 사례, One-class 분류 모델을 적용한 사례를 설명하고 분류 모델을 구축하는 설계도 격인 컴퓨터 비전 아키텍처에 관한 연구 사례를 설명한다. 3장에서는 시스템의 구성 요소와 프로세스 과정에 대해 설명하고 시스템 개발 환경에 대해서 설명한다. 4장에서는 시스템에 들어가는 딥러닝 모델을 구축하고 훈련하는 과정에 대해서 설명한다. 5장에서는 구축한 시스템을 평가하는 방법과 평가에 사용되는 데이터에 대해 설명한다. 6장에서는 시스템 개발 결과로 시스템 개발 결과 화면과 시스템의 요소별 측정 속도 그리고 각 모델별 성능 평가 결과를 제시한다. 마지막으로 7장에서는 본 연구를 요약하고 향후 추가적으로 수행할 연구를 제시한다.

2. 관련 연구

2-1. 멸종위기 종 영상 데이터를 이용한 딥러닝 분류 연구 사례

유병혁, 김선현은 소백산 국립공원 북부 사무소에서 관리 중인 경상북도 영주시 죽령의 생태 통로에 카메라를 설치했다. 그리고 발견될 가능성이 높은 삵, 고라니, 노루, 멧돼지, 너구리 5종에 대해서 카메라를 통해서 영상 데이터를 수집하고 CNN에서 분류 작업을 실시한 결과 96.25%의 정확도 가진 모델을 제시한 바 있다 [13]. Peiqin Zhuang, Linjie Xing, Yanlin Liu, Sheng Guo, Yu Qiao은 해양 생물의 영상 데이터에서 객체를 탐지하는 작업을 딥러닝 기반의 모델로 할 수 있는 방법을 제시했다. 모델 아키텍처는 ResNet을 사용했고 SeaCLEF2017 데이터 셋을 사용했다 [14]. Hung Nguyen, Sarah J. Maclagan, Tu Dinh Nguyen, Thin Nguyen 등은 오스트레일리아 빅토리아 주에서 흔히 발견되는 종인 새, 쥐, 주머니여우, 워뱃 등에 대해서, 카메라를 통해서 획득한 영상 데이터에서 이들 개체를 탐지하는 연구를 수행한 바 있다. 그 결과 정확도 96.6%의 성능을 보인 모델을 획득하게 되었다 [15]. Mohammad Sadegh Norouzzadeha, Anh Nguyenb, Margaret Kosmalac, Alexandra Swansond에 의하면 탄자니아의 세렝게티 국립공원의 동물 종을 조사하기 위한 “Snapshot Serengeti” 프로젝트에서 [16] 많은 수의 동물 이미지 데이터가 처리되지 않았다. 왜냐하면 자원봉사자들이 이 데이터를 자발적인 참여로 처리해야하기 때문이다. 저자들은 아래와 같이 딥러닝 모델로 이미지 안의 객체의 종류와 숫자 그리고 행동까지 알 수 있다고 했다 [17] (그림 2 참고). 본 연구에서 개발한 앱 시스템을 적용한다면 생물 종 다양성 보존 연구를 수행하는 연구자들과나 동물 종의 수출입을 담당하는 세관 담당자들이, 모바일 디바이스로 촬영한 영상을 객관적으로 분류할 수 있을 것으로 기대한다.

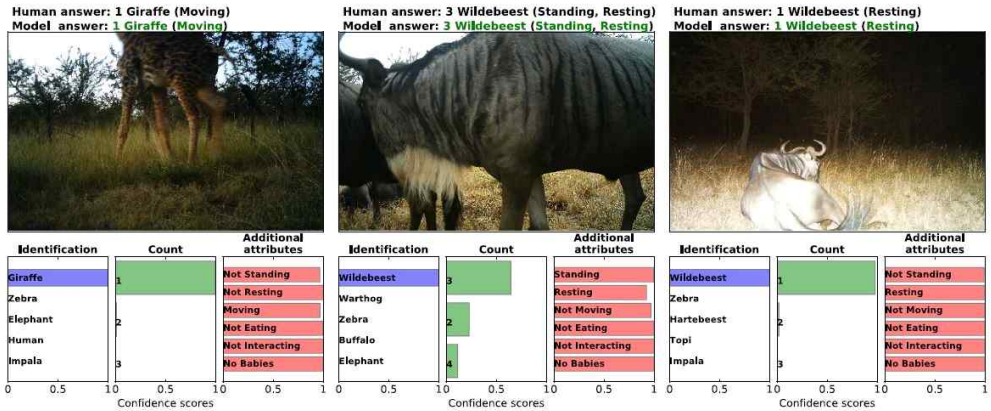


그림2. 딥러닝 모델을 활용한 영상 이미지 분석 사례. 아래쪽 행의 왼쪽부터 객체 클래스, 숫자, 행위가 확률 값으로 나타나 있다 [17].

2-2. 머신러닝 모델 적용 연구 사례

2-2-1. 단일 클래스 분류 모델 적용 연구



그림3. One-class 전이 학습을 적용한 사례. 왼쪽 미국 국기가 있는 이미지 오른쪽은 유사하지만 미국 국기는 아닌 이미지이다. [20].

Pramuditha Perera, Vishal M. Patel는 딥러닝 기반의 One-class 전이 학습 방법에 대한 방법론을 제시한 바 있다 [20] (그림 3 참고). 제안된 방법은 CNN

위에서 동작하며 대상 클래스의 특징 공간의 분산을 최대한 적게 유지하면서 다른 클래스와 차이점이 있는 특징을 추출해낸다. Larry M. Manevitz, Malik Yousef 는 단일 클래스 분류에 적합한 SVM을 구현해냈다. 모델은 대상 클래스의 특징 범위를 벗어나는 이상치 데이터를 SVM으로 구분해낸다 [21]. 본 연구에서는 구현의 용이성을 고려하여 단일 분류를 수행하는 SVM으로 대상 클래스인지 아닌지를 구분하도록 시스템을 개발했다.

2-2-2. 앱 및 웹에서 딥러닝 모델 적용 연구

Abhishek Sehgal, Nasser Kehtarnavaz는 그들의 연구에서 Tensorflow Lite, CoreML 등의 프레임워크를 통해서 딥러닝 모델을 모바일로 탑재 가능한 파일로 변경이 가능하다고 주장했다. 그들의 Multi-threading을 통해서 실시간으로 앱이 데이터를 처리하는 방법을 제시했다 [18]. Lubos Juranek; Jiri Stastny; Vladislav Skorpil; Lukas Junek는 이미지 안의 어떤 사람의 나이, 성별 등의 특징을 인식하거나 탐지하는 딥러닝 모델을 클라이언트 쪽에서 Tensorflow.js, WebGL-accelerated JavaScript 라이브러리로 구현하는 방법을 제시했다 [19]. 본 연구에서는 TensorflowLite를 활용하여 모델을 모바일에 탑재하는 방식을 채택했다.

2-3. 컴퓨터 비전 네트워크 아키텍처 연구

Lecun 등의 연구로 인해서 [4, 22] 신경망 기반의 학습 방법이 각광 받기 시작했고 오늘날에 이르러서는 컴퓨터 비전 분야에서는 CNN이 주류로 자리 잡게 되었다. 컴퓨터 비전의 많은 연구자들은 모델의 성능을 높이기 위해서 모델 네트워크 아키텍처에 주목했다. 그 결과 많은 연구 결과가 쏟아졌고 그 중에서 몇몇 논문들은 State of the art라는 칭호를 받아 동시대 연구자들에게 인정받았다. Alex

Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton은 AlexNet이라는 아키텍처를 발표하고 그 성능을 ImageNet 데이터 셋을 통해 선보인 바 있다 [23]. Karen Simonyan, Andrew Zisserman는 그들의 연구에서 작은 크기의 연속적인 컨볼루션 연산을 통해 모델 네트워크의 층을 깊게 만들어 ImageNet 데이터 셋에서 Top-5 테스트 오류율 6.8%를 달성하기도 했다 [24]. 한편 모델을 종으로 쌓는 것이 아니라 횡으로 쌓아 모델의 용량을 줄이면서 효율적으로 이미지의 특징을 뽑아낼 수 있는 연구도 진행되었다. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet 등은 GoogLeNet이라는 아키텍처를 발표하면서 컨볼루션 계층을 횡대로 나열하는 Inception module을 선보였다 [25]. 모델이 점차 깊어지면서 발생한 문제 중 하나가 낮은 층의 Feature map에서 담고 있던 특징 정보가 점차 Down sampling 되면서 높은 층에서는 거의 소실된다는 점이었다. 이 때문에 오히려 모델의 용량은 커지는데 성능이 떨어지거나 과적합이 발생하는 현상이 생겼다. 이를 해결하기 위해서 Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun는 입력 Feature map과 컨볼루션을 수행한 출력 Feature map을 더하는 Residual block을 소개했다 [26]. 본 연구에서는 특히 모바일이나 임베디드 시스템에 잘 맞는다고 알려진 MobileNet 아키텍처를 사용했다. MobileNet은 Andrew G. Howard Menglong Zhu Bo Chen Dmitry Kalenichenko 등이 2017년도에 발표한 논문이다. 이미지들의 공간적인 특징의 연관성과 채널별 특징의 연관성을 구분하여 살펴보는 Depthwise seperable convolution으로 구성되어 있고 모델의 크기와 지연 속도의 Trade off를 조절하는 두 개의 매개변수가 존재하는 것이 특징이다 [27].

3. 시스템 설계 및 개발

3-1. 시스템 구성 및 흐름도

3-1-1. 온 디바이스(On-Device) 시스템

온 디바이스란 서버를 거치지 않고 기기 자체에서 정보를 수집하거나 연산할 수 있는 시스템을 말한다. 이를 통해서 사용자에게 빠르게 서비스를 제공할 수 있다는 장점이 있다. 저자는 본 연구 논문에 앞서, 온 디바이스에서 이미지를 획득하고 분석하여 결과를 보여주는 선행 연구를 진행한 바 있다 [28]. 온 디바이스 시스템의 블록 다이어그램은 그림 4의 왼쪽과 같이 카메라 모듈과 분류 모듈로 나누어진다. 온 디바이스 시스템의 프로세스 흐름은 그림 4의 오른쪽과 같다. 먼저 카메라 모듈에서 이미지를 획득한다. 그 다음 미리 준비해둔 모바일 디바이스 탑재용 딥러닝 모듈을 탑재하여 획득한 이미지에 대한 분석을 실시한다. 마지막으로 분석 결과를 사용자에게 보여 준다.

이때 사용하는 딥러닝 모듈은 다음과 같이 Tensorflow의 Keras 라이브러리를 이용한다. Keras는 Tensorflow의 상위 레벨 API이고 딥러닝 모듈을 쉽게 구축할 수 있도록 여러 기능을 제공하고 있다. 4장에서 후술하는 대로 딥러닝 모듈을 구축하고 훈련시킨 뒤에 훈련을 완료한 모듈을 모바일에 탑재 가능한 형태인 Tensorflow Lite 파일로 변환 한다. 이 파일을 Android Studio내에 삽입한 뒤에 모델 연산을 할 수 있는 인터프리터 객체를 만들어 이미지 분류 연산을 수행한다. 온디바이스 시스템이 서버-클라이언트 시스템과 비교해서 갖는 장점은 실시간성에 있다. 서버-클라이언트 시스템과 달리 온 디바이스는 클라이언트 시스템 내에서 모든 것을 처리하기 때문에 실시간으로 작업 수행을 완료하고 사용자에게 결과를 보여주는 것이 가능하다. 실제로 본 연구의 선행 연구에서 개발한 온 디바이스 시스템의 평균 추론 속도는 50ms로 사용자로 하여금 시스템 Latency가 거의 느껴지지 않게 할 정도로 빠르다. 그러나 온 디바이스 시스템은 한 가지 문제점을 갖고 있다.

그것은 닫힌 시스템이기 때문에 앱의 업데이트 사항을 바로 적용할 수 없다는 점이다. 업데이트 사항을 적용하려면 사용자가 업데이트를 설치할 수 있도록 유도해야 한다. 이에 따라 본 연구에서는 이런 문제점을 극복 할 수 있도록 서버-클라이언트 구조로 시스템을 개발하기로 했다.

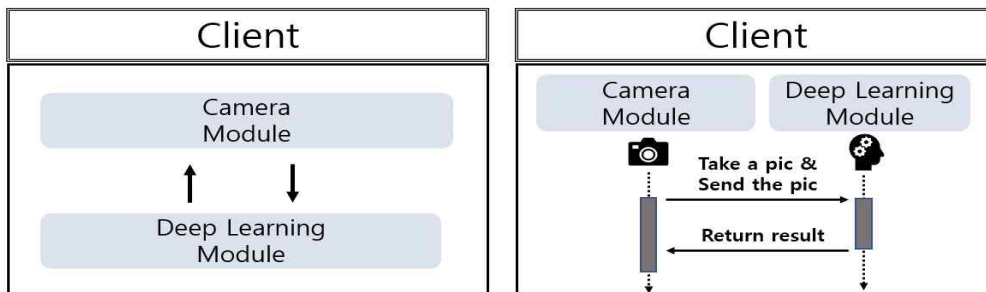


그림4. 온 디바이스 시스템 블록다이어그램과 시퀀스다이어그램.

3-1-2. 모바일 및 딥러닝 서버 시스템

본 연구에서 개발한 서버-클라이언트 구조의 앱 시스템은 그림 5과 같이 크게 네 부분으로 이루어져있다. 전체적인 작업 흐름은 그림 6과 같다.

(1) Client(클라이언트)

먼저 전체적인 앱의 화면 구성은 그림 7과 같다. 이미지를 캡처할 수 있는 카메라 모듈이 있는데 이 카메라 모듈로 이미지를 캡처한 뒤에 서버 측에 이미지 분석을 요청한다. 앱이 시작되면 사용자가 실시간 프리뷰로 캡처할 이미지를 확인할 수 있고 디스플레이의 중단에 캡처 버튼이 있다. 버튼을 누르게 되면 서버로 이미지 분석을 요청하게 된다. 서버로부터 결과가 오면 사용자에게 캡처한 이미지와 결과를 보여준다. 여기서 확률이 가장 높은 앵무새 종의 이름을 터치하면 웹 뷰를 통해서 그 종에 대한 정보 페이지를 보여준다. 결과 화면에서 저장하기 버튼을 터치하면 서버로 캡처한 이미지를 저장하도록 요청한다.

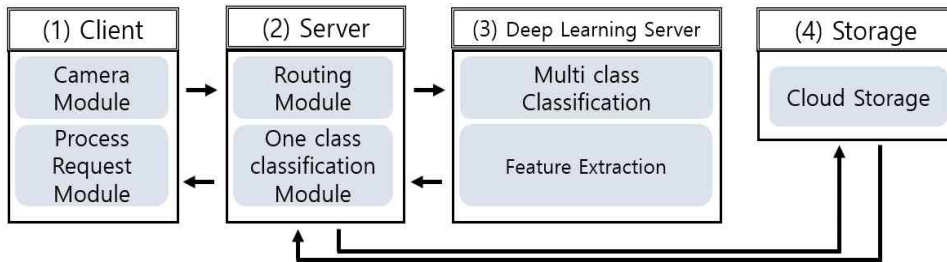


그림5. 모바일 및 딥러닝 서버 시스템 블록다이어그램

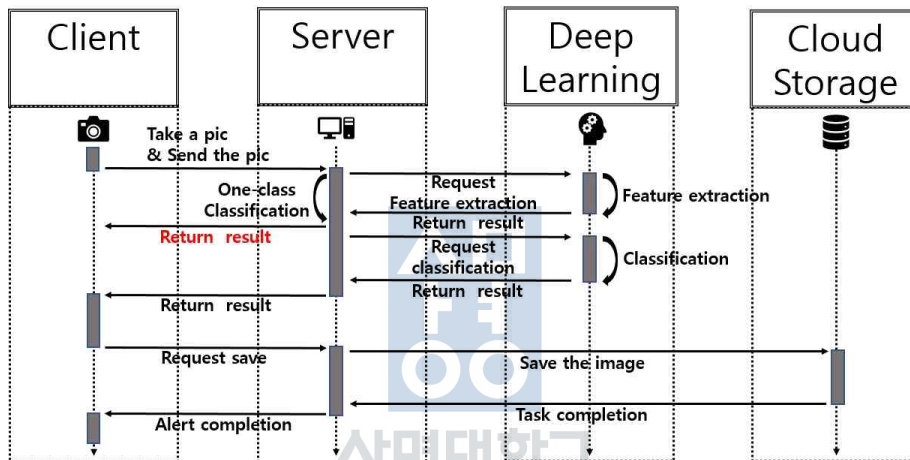


그림6. 모바일 및 딥러닝 서버 시스템 시퀀스 다이어그램



그림7. 모바일 앱 화면 구성

(2) Server(서버)

서버는 크게 세 가지 기능이 있다. 첫 번째, 클라이언트에서 요청(이미지 분석, 저장)이 들어오면 URL에 따라 관련 로직을 수행하도록 라우팅을 하는 기능, 두 번째, 이미지 분석에서 앵무새인지 아닌지를 One class SVM으로 판별하는 기능, 마지막으로 앵무새라고 판별 되었을 경우에 딥러닝 서버로 앵무새 종 분류를 요청하는 기능이다. 앵무새 종 분류 기능을 서버에서 하지 않고 딥러닝 서버를 계층화하여 구축한 이유는 다음과 같은 이유 때문이다.

- 로직을 기능별로 분해해서 유지 보수 시 용이하도록 한다.
- 새로운 기능이 추가될 경우 용이하도록 한다.
- 디버깅을 용이하게 한다.
- 속도 면에서 이점이 있는 Tensorflow 그래프 연산을 활용한다.

서버는 Flask 라이브러리를 이용하여 구축했다 [29]. Flask는 파이썬으로 서버를 구축할 수 있는 웹 프레임워크 중 하나이다. Werkzeug, Jinja2에 기반을 둔다. Flask 자체는 다른 웹 프레임워크와 비교했을 때 기본적인 기능만 구현되어 있다. 따라서 유저가 유연하게 웹 서버를 구축할 수 있다는 장점이 있다. 파이썬 데코레이터 문법으로 라우팅 규칙을 추가했고 클라이언트 측에서 POST 방식으로 보내온 이미지 픽셀 배열을 받아서 두 단계의 분류 작업을 실시한다. One-class classification 과정에서 이미지의 특징을 추출해야 하는데 딥러닝 서버에 요청을 해서 CNN을 통해 추출된 이미지 특징을 전달 받는다. 분류 작업이 끝나면 마지막으로 결과를 JSON 형태로 클라이언트 측에 전달한다.

서버는 REST를 염두에 두고 개발했다. REST는 Representational State Transfer의 약자로 네트워크 아키텍처를 어떻게 구성해야할지에 대한 원리의 모음

이다. 핵심은 자원을 이름으로 구분하고 자원의 상태를 주고받는 규칙을 정의한다. HTTP의 표준 프로토콜을 따르면서 하이퍼미디어 API의 기본을 지키므로 범용성을 보장한다는 장점이 있다. 또 서버와 클라이언트의 역할을 분리할 수 있다.

(3) Deep Learning Server(딥러닝 서버)

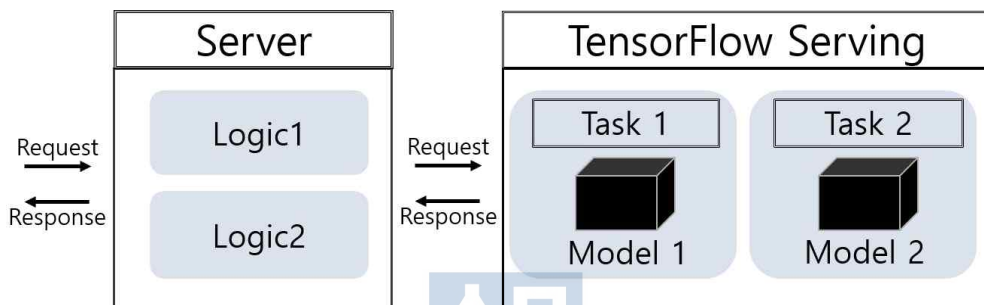


그림8. Tensorflow Serving API 구성도

딥러닝 서버의 구성은 그림 8과 같다. 딥러닝 서버는 Tensorflow Serving API로 구축했다. Tensorflow Serving은 배포 환경을 위해서 기계 학습 모델을 유연하게 배치하고 모델의 성능을 고수준으로 끌어 올릴 수 있는 시스템이다 [30]. Tensorflow Serving API를 사용하면 서버 아키텍처의 구현을 크게 건드리지 않으면서 새로운 모델을 배포할 수 있다. 딥러닝 서버로 요청이 들어오면 Tensorflow 그래프로 표현되는 모델로 연산을 수행하고 결과를 서버에 응답해준다. 본 연구의 시스템에는 딥러닝 서버가 One-class classification에 쓰일 이미지의 특징을 추출하는 경우와 Multi-class classification을 통해서 앵무새의 종판별을 수행하는 경우에 관련 연산을 수행하도록 설계되어 있다.

(4) Storage(스토리지)

분석을 완료한 이미지는 사용자의 선택에 따라 클라우드 스토리지에 저장될 수 있다. 만약 사용자가 결과 화면에서 이미지 저장 버튼을 터치하게 되면 클라이언트 앱 측에서 서버로 이미지 저장 요청을 하게 되고 서버에서는 POST 방식으로 보내진 데이터를 클라우드 스토리지에 저장한다. 클라우드 스토리지는 Google Firebase의 스토리지를 사용했다.

3-2. 시스템 개발 환경

시스템은 다음과 같은 환경에서 개발되었다. 모델을 구축할 때는 Python 언어로 작성되었으며 Anaconda로 가상환경을 구축하고 필요한 라이브러리를 관리했으며 Jupyter Notebook으로 코드를 편집했다. 앱은 Kotlin 언어로 작성되었으며 Android Studio에서 Gradle로 필요한 라이브러리를 관리하고 Editor로 코드를 편집했다.

- CPU: Intel(R) Xeon(R) Silver 4116 CPU @ 2.10GHz
- GPU: GeForce RTX 2080 Ti(x2)
- OS: Ubuntu 16.04 LTS, Android
- Python, Anaconda, Jupyter Notebook, Kotlin, Android Studio

4. 모델 설계 및 개발

4-1. 데이터 준비

딥러닝 분야에서 성능이 좋은 모델을 만들기 위해서는 많은 양의 데이터가 필요하다. Chen Feiyang, Chen Nan, Mao Hanyang, Hu Hanlin의 연구에 따르면 0부터 9까지 숫자 이미지를 분류하는 CNN 모델을 훈련시켜서 90% 이상의 정확도를 얻으려면 약 60,000 장 이상의 훈련 데이터 셋과 10,000의 테스트 셋이 필요하다고 한다 [31]. 많은 컴퓨터 비전 관련 연구 논문에서 인용하고 있는 ImageNet 데이터 셋의 경우 약 2만개 이상의 클래스로 구성된 14,000,000 여장의 이미지로 이루어져 있다 [32, 33]. 데이터양을 늘리는 방법은 크게 두 가지 접근법으로 나뉜다. 하나는 데이터 자체를 많이 획득하는 것이고 다른 하나는 획득한 데이터를 변경하여 새로운 데이터를 만들어 방법이다.

4-1-1. 동물원 영상 데이터(S대공원, G동물원, C동물원, 촬영 영상 데이터)

본 연구에서 앵무새인지 아닌지를 판단하는 One class SVM과 앵무새일 때 앵무새의 종판별을 수행하는 Multi class CNN을 훈련시킬 때 쓰일 데이터를 구성하기 위해서 동물원에서 촬영한 앵무새 영상 데이터를 사용했다. 영상 데이터는 경기 과천시 S대공원, 광주광역시의 G동물원, 충북 청주시의 C동물원의 협조를 얻어 획득했다. 획득한 동물원 앵무새 영상 데이터의 현황은 표3과 같다. 획득한 영상 데이터의 수가 적지 않지만 실제로 사용한 이미지 수는 모델 테스트용으로 사용된 각 종의 이미지 10장뿐이다. 그 이유는 대부분의 이미지가 동영상을 프레임 단위로 캡처해서 생성했기 때문에 매우 유사하다 (그림 9 참고). 훈련 데이터 내에 이

러한 유사한 이미지의 비중을 높이면 과적합의 위험이 커지기 때문에 테스트용으로 최대한 차이가 나는 이미지를 선별해서 사용했다.

	동영상 개수	이미지 개수
홍금강앵무 (<i>Ara chloropterus</i>)	2	840
청금강앵무 (<i>Ara ararauna</i>)	3	3029
큰유황앵무 (<i>Cacatua galerita</i>)	3	1740
흰이마유황앵무 (<i>Cacatua goffiniana</i>)	3	1590
회색앵무 (<i>Psittacus erithacus</i>)	4	2700

표3. 동물원 앵무새 영상 데이터 현황



그림9. 동물원 영상 데이터 예시. 왼쪽과 오른쪽을 비교해보면 앵무새의 자세가 바뀐 것을 빼고는 거의 유사하다는 것을 알 수 있다.

4-1-2. 크롤링 영상 데이터

대부분의 포털 사이트는 검색 키워드에 대한 이미지 리스트를 보여주는 구역이 존재한다. 웹 페이지는 HTML, CSS, Javascript로 구현 되어 있고 이미지를 웹 페이지에 띄우려면 이미지 태그의 src 속성에 이미지 원본 파일의 위치를 작성하게 되어 있다 (그림 10 참고). 이 이미지 원본 파일의 위치 정보를 이용해서 이미지를 다운 받을 수 있다. 이처럼 웹 페이지의 정보를 가져와서 데이터를 추출하는 행위를 크롤링(Crawling) 혹은 스크레이핑(Scraping)이라고 한다. 본 연구에서

크롤링으로 데이터를 획득해야 하는 이유는 모델의 성능 때문이다. 4-1-1에서 기술한대로 동물원에서 획득한 영상 데이터는 거의 유사하기 때문에 이 데이터를 대량으로 훈련 데이터에 포함 시킬 경우에 모델은 이 데이터에게 맞게 편향될 것이다. 크롤링으로 획득한 데이터는 종당 1000장이다. 이중 900장을 4-1-3에서 기술한대로 데이터 어그멘테이션 기법을 써서 3600장으로 만든다. 이중에서 훈련 데이터로 쓰인 데이터의 수는 2800장이고 검증 데이터의 수는 700장이다. 단, 검증 데이터의 정보가 훈련 데이터에 누출되는 경우가 없도록 데이터 어그멘테이션을 적용하기 전에 훈련 데이터와 검증 데이터 셋으로 나누어서 각각 데이터 어그멘테이션 기법을 적용했다. 크롤링 프로그램은 파이썬 셀레늄(Selenium) 라이브러리로 구현했다. 셀레늄의 기능을 이용하면 웹 브라우저의 드라이버에 접속하여 웹 브라우저를 제어할 수 있다 [34]. 셀레늄으로 5개의 포털사이트(구글, 네이버, 다음, 야후-재팬, 바이두)에 접속 및 조작하여 이미지 리스트 구역의 이미지를 로컬 PC로 다운 받는다.



그림10. 구글 이미지 섹션의 검색 예시

4-1-3. 데이터 어그멘테이션

이미지를 조작하여 새로운 이미지를 만들어 내는 기법은 여럿 존재한다. 예를 들어, 원본 이미지를 회전 시키거나 좌우를 뒤집는 방법이 있고 상하좌우로 평행이동 시키는 방법도 있다. 또 채도, 색상이나 명도를 변경하는 방법도 있다. 원본 이미지에 노이즈를 섞어서 원본과는 다른, 새로운 이미지를 생성하는 연구도 시행된 바 있다 [35]. Girshick 등의 연구에서는 이미지의 크기를 네트워크의 입력 크기에 맞추는 이미지 워핑이나 이미지 내에 객체가 포함되어 있을 법한 지역을 잘라내는 이미지 크롭을 적용했다 [36]. 국내 연구 중에서도 데이터를 충분히 모을 수 없는 제약 사항 때문에 데이터 어그멘테이션을 적용하여 학습 데이터를 생성하는 경우가 있다 [37, 38]. 데이터를 회전 시키거나 뒤집거나 평행이동 시켜서 만든 데이터가 특히 분류 모델을 학습시킬 때 유효한 이유는 변환 불변성이라는 특징 때문이다. 이는 이미지 내에 객체의 위치까지 추정하는 Object detection이나 픽셀 단위로 분류를 실시하는 Semantic segmentation과는 다르게 데이터 어그멘테이션이 분류 모델을 학습시킬 데이터를 생성하는데 쓰일 수 있는 이유이다 [39].

본 연구에서는 4-1-1과 4-1-2에서 언급한 방식으로 각 앵무새 종별 이미지를 획득한 뒤에 훈련 데이터와 검증 데이터 그리고 테스트 데이터로 나눈 뒤에 데이터 어그멘테이션을 적용했다. 데이터 어그멘테이션을 적용하고 난 뒤에 데이터를 나누지 않은 이유는 검증 데이터나 테스트 데이터의 정보가 훈련 데이터에 포함되어 훈련 과정 중에 모델에 누출될 수 있기 때문이다. 예를 들어서 원본 이미지를 90도 회전한 이미지가 있을 때 원본 이미지는 검증 데이터 셋에 포함되었는데 90도 회전시킨 이미지가 훈련 데이터 셋에 포함될 경우 모델은 검증 데이터의 정보를 훈련 과정 중에 미리 알게 된다. 그렇게 되면 정확한 검증을 할 수 없게 된다. 데이터 어그멘테이션으로 적용한 이미지 변경 방법으로는 좌우 뒤집기와 상하 뒤집기 방법을 적용했다. 명도나 채도를 변경하는 방법을 사용하지 않은 이유는 모델이 비슷한 색이나 형태의 앵무새도 잘 구분할 수 있는지 확인하기 위함이다 (그림 11).

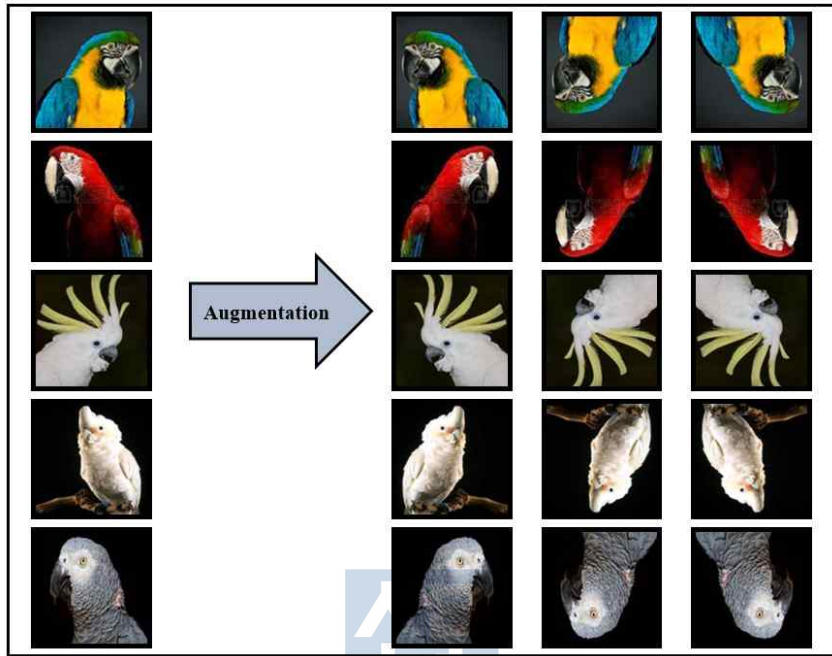


그림11. 데이터어그멘테이션 적용 예시

본 연구에서는 imgaug 라이브러리를 이용해서 데이터 어그멘테이션을 적용했다 [40]. imgaug는 기계 학습 모델 실험을 필요한 이미지의 수를 늘리기 위해서 여러 가지 기능을 제공하는 라이브러리이다. 이 기능을 결합하거나 임의의 순서로 적용하도록 설정할 수 있으며 CPU에서도 수행될 수 있다. 또 인터페이스가 직관적이어서 코드를 쉽게 짤 수 있다. 지원하는 어그멘테이션 유형에는 Bounding box, Segmentation map, Keypoint, Heatmap, Landmark 등이 있다.

4-2. 모델 구축

4-2-1. One class SVM



그림 12. CNN 기반의 Multi class classification 잘못된 예. 홍금강앵무는 첫 번째 열에 있는 이미지이지만 CNN의 Softmax 함수의 한계 때문에 두 번째에서 네 번째 열에 있는 새들도 모두 홍금강앵무라고 분류하고 있다.

Multi class CNN으로 분류 작업을 수행하기에 앞서 One class SVM으로 One class 분류를 수행해야 하는 이유는 Multi class CNN의 한계 때문이다. 그림 12를 보면 홍금강앵무가 아님에도 불구하고 색이 유사하다는 이유로 홍금강앵무라고 분류하고 있고 그 확률도 모두 97퍼센트 이상으로 강하게 확신하고 있다. 생물종 다양성 연구나 모니터링 시에 비전문가가 최종적으로 판단을 한다면 이는 문제가 될 수 있다. 이를 방지하고자 One class 분류를 수행해서 Multi class CNN이 학습한 대상이 아님을 사용자에게 알려 추가적으로 모델을 학습시키도록 유도할 필요가 있다.

데이터 셋과 문제에 따라 종종 이진 분류를 수행해야 하는데 한 개의 클래스로만 되어 있거나 클래스 간의 불균형이 심한 경우가 있다. 또는 한 개의 클래스와 그 밖의 클래스로 나뉘는 경우에 Negative 클래스의 범주가 너무 커서 데이터를

명확하게 모으기 힘든 경우가 존재한다. 이런 경우가 클래스 A와 이상치를 구분하는 분류로 대표적인 도메인으로 Anomaly Detection이 있다. Raghavendra Chalapathy, Sanjay Chawla는 그들의 Survey 논문에서 딥러닝 기반의 Anomaly Detection 알고리즘을 소개하고 다양한 도메인에서 적용 사례를 제시한 바 있다 [41]. 본 섹션에서는 SVM에 대해서 소개하고 어떻게 One class 분류를 수행할 것인지 제시할 것이다.

SVM(Support vector machine)은 딥러닝 기반이 아닌 머신러닝 지도 학습 모델 중 하나이다. 주로 데이터 분류나 회귀 분석할 때 사용한다. 예를 들어 이진 분류 모델의 경우 SVM은 새로운 데이터가 어느 클래스에 속할지 비확률적으로 판단한다. 모델은 데이터가 표현되는 공간에서 어떤 특징으로 나누어지는 경계를 찾게 된다. 이때 결정 경계와 가장 가까이 있는 데이터 포인트들(Support vectors)과의 거리가 가장 먼 결정 경계를 찾는다 (그림 13 참고).

SVM을 사용한 One class classification은 다음과 같은 과정을 통해 수행한다. 먼저 ImageNet 데이터 셋으로 미리 훈련시킨 ResNet50 모델의 최상층 계층을 제거하고 이미지의 특징만을 추출할 수 있도록 한다. ResNet50 모델은 계층이 점점 깊어지면서 소실되는 정보를 Residual block으로 어느 정도 보완 가능하기 때문에 깊은 모델을 쌓을 수 있어서 표현력이 풍부하다는 특징이 있다 [26]. 또 ImageNet 데이터 셋에는 새의 이미지가 다수 포함되어 있어서 앵무새의 미세 특징들을 잘 잡아낼 수 있을 것으로 기대한다. 미리 훈련된 ResNet50으로 이미지의 특징을 추출하고 나서 이를 벡터로 만든다. 그 다음에 표준정규화로 특징 값을 평균 0, 분산 1의 분포를 따르도록 만든다. 그리고 나서 주성분 분석(Principal component analysis)을 통해서 특징들의 분산이 가장 잘 드러나는 주성분을 뽑아서 이미지 특징 공간의 차원 수를 줄인다. 주성분 분석을 사용하면 데이터 특성 공간의 차원 수를 줄여 모델의 훈련이나 추론 속도를 크게 개선할 수 있다는 장점이 있다 [42] (그림 14 참고).

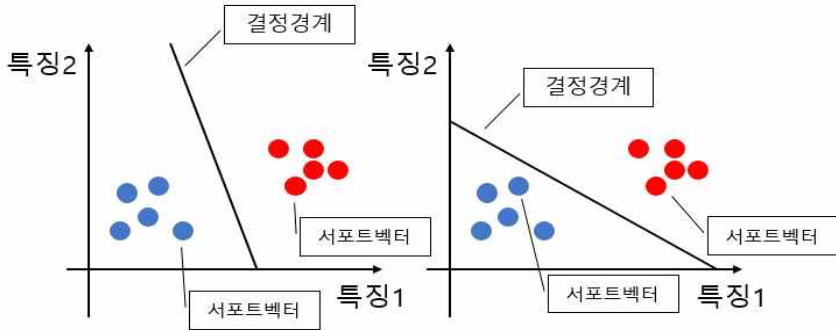


그림 13. 서포트벡터머신

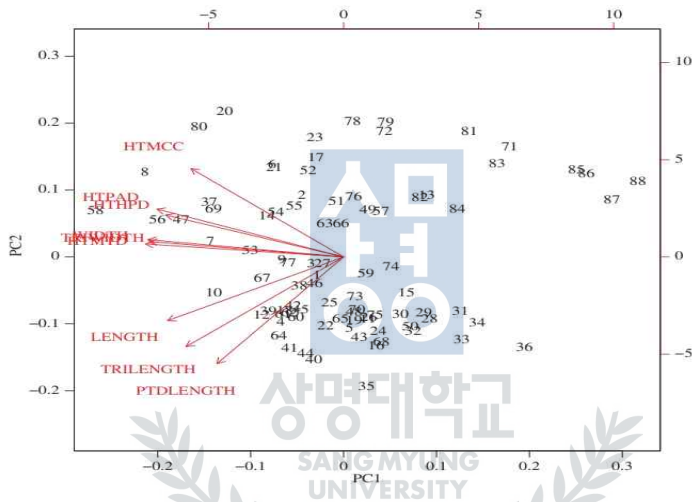


그림 14. 주성분분석 [42]

주성분 분석으로 추출한 주성분들로 SVM을 훈련시킨다. SVM에서 데이터를 고차원 특징 공간에 매핑 하는데 사용하는 방법은 크게 두 가지가 있다.

- Polynomial kernel: 원래 특성들을 조합하여 사용자가 지정한 차수까지 가능한 모든 조합을 계산한다.
- RBF(Radial basis function) 혹은 Gaussian kernel: 지수 함수를 테일러 전개를 사용하여 거의 무한한 다항식 차원으로 전사한다. 가능한 모든 차수의 모든 다항식을 고려한다. 고차 항이 될수록 그 특성의 중요도가 줄어들게 된다.

본 연구에서는 RBF kernel을 적용하여 SVM을 훈련시켰고 이때 커널의 폭을 결정하는 매개변수인 γ 를 0.001로 설정했다. ResNet50으로 특징을 추출하는 과정은 Tensorflow Keras 라이브러리를 이용했고 표준 정규화와 주성분분석 그리고 One class SVM은 Scikit-learn 라이브러리를 사용했다. Scikit-learn 라이브러리는 여러 가지 데이터 분석을 위한 도구를 제공한다. Numpy, Scipy, Matplotlib로 구축되었으며 오픈 소스이고 BSD 라이선스를 갖는다 [43]. 본 연구에서는 시간 관계상 One class classification의 모델 최적화를 수행하지 않았다. 다만 Grid Search 등을 통해서 모델 최적화를 수행한다면 더 결과가 좋아질 것으로 기대한다.

4-2-2. Multi class CNN

(1) 모델 아키텍처



앵무새인지 아닌지에 대한 One class 분류 과정이 끝나고 앵무새라고 판단될 경우 이 앵무새에 대한 종판별을 수행한다. 앵무새 종판별은 다중 클래스 분류 CNN을 이용한다. 본 연구에서 사용한 모델 아키텍처는 MobileNet이다. MobileNet 아키텍처의 주요한 특징은 입력 계층을 제외하고 나머지 Convolution 계층을 모두 Depthwise separable convolution으로 바꾼 것이다. François Chollet에 의하면 [44] 보통의 Convolution은 입력 채널마다 커널이 할당되고 이 커널이 입력 Feature map을 순회하면서 원소별 곱셈을 하고 이를 더한 것의 선형 결합으로 출력 Feature map의 한 채널이 결정된다. 그렇기 때문에 Cross-channel correlations와 Spatial correlations을 동시에 고려하게 된다. 그런데 이를 분리하면 모델의 연산 량과 파라미터 수를 줄이면서 더 좋은 성능을 낼 수 있다고 한다. MobileNet에서는 이를 연구 내용을 반영하여 일반 Convolution을 채널별 Spatial correlations를 계산하는 Depthwise convolution과

Cross-channel correlations를 계산하는 Pointwise Convolution을 연속으로 수행하는 Depthwise separable convolution으로 바꾼다 (그림 15 참고).

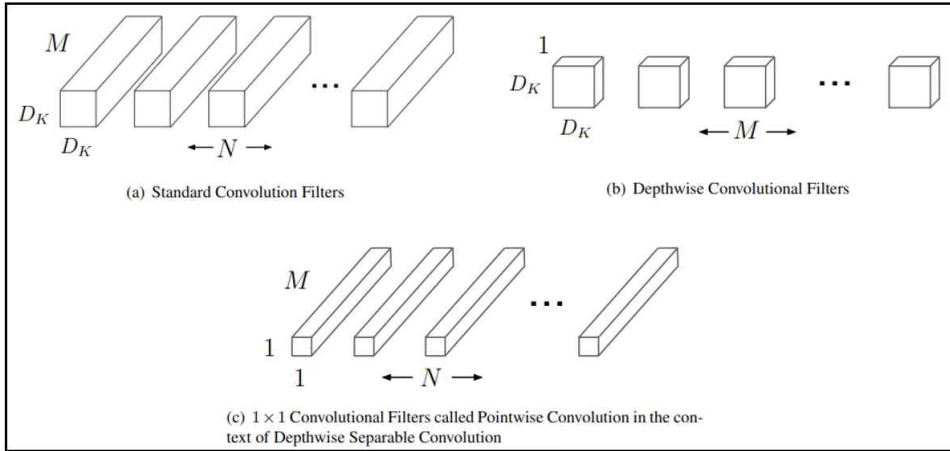


그림 15. Convolution 예시. 왼쪽 상단은 Regular convolution, 오른쪽 상단은 Depthwise convolution, 하단은 Pointwise convolution [27]

MobileNet을 제안한 저자들이 논문에서 주장한 바에 따르면 Regular convolution을 사용할 경우 각 계층에서의 원래 계산량은 식 1과 같다.

$$D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F \quad (1)$$

여기서 D_K 는 입력 Feature map의 너비와 높이이고 M 은 채널수이다. D_F 는 커널의 너비와 높이이고 N 은 채널수이다. 그런데 Depthwise separable convolution으로 바꾸면 연산량이 아래의 식2와 같이 바뀐다.

$$D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F \quad (2)$$

줄어드는 연산량은 식3과 같다.

$$\frac{D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F}{D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F} = \frac{1}{N} + \frac{1}{D_K^2} \quad (3)$$

결국 출력 채널수와 입력 Feature map의 크기에 좌우되는데 입력 Feature map의 크기는 고정이므로 출력 채널수의 역수만큼 계산량이 줄어들게 된다.

MobileNet의 또 다른 특징은 모델의 용량과 지연속도 그리고 정확도의 Trade off를 사용자가 조절할 수 있게 하는 하이퍼파라미터를 제공한다는 것이다. 먼저 모든 계층의 입력 채널수와 출력 채널수를 조절할 수 있는 Width multiplier α 를 적용하면 계산량은 아래의 식4와 같아진다.

$$D_K \cdot D_K \cdot \alpha M \cdot D_F \cdot D_F + \alpha M \cdot \alpha N \cdot D_F \cdot D_F \quad (4)$$

여기서 α 는 0 초과 1이하의 값을 갖는다. α 가 1일 경우 원래의 MobileNet이고 1보다 작은 값일 경우 그만큼 채널수가 원래의 모델보다 줄어들게 된다. α 값을 조절함으로써 채널수를 조정하고 모델의 용량과 정확도 그리고 추론 지연 속도를 조절할 수 있게 된다. 다음으로 MobileNet에서는 네트워크의 이미지의 입력 크기를 하이퍼파라미터로 조정할 수 있다. 입력 크기가 커지면 정확도는 더 커지겠지만 네트워크의 추론속도는 줄어든다. 본 연구에서는 ImageNet으로 미리 훈련시킨 MobileNet 모델을 사용했으므로 네트워크의 입력 크기는 224x224가 되고 α 의 값은 1을 사용했다. 모델의 계층 수는 157층이고 총 파라미터 수는 2,264,389개이다. 이 중에서 학습이 가능한 파라미터는 2,230,277개이고 학습이 가능하지 않은 파라미터는 34,112개이다. 5종의 앵무새를 분류하기 위해서 미리 훈련시킨 MobileNet 모델의 최상단에 후술할 Global Average Pooling층과 5개의 확률 값을 출력하는 Softmax층을 더했다.

(2) 전이 학습

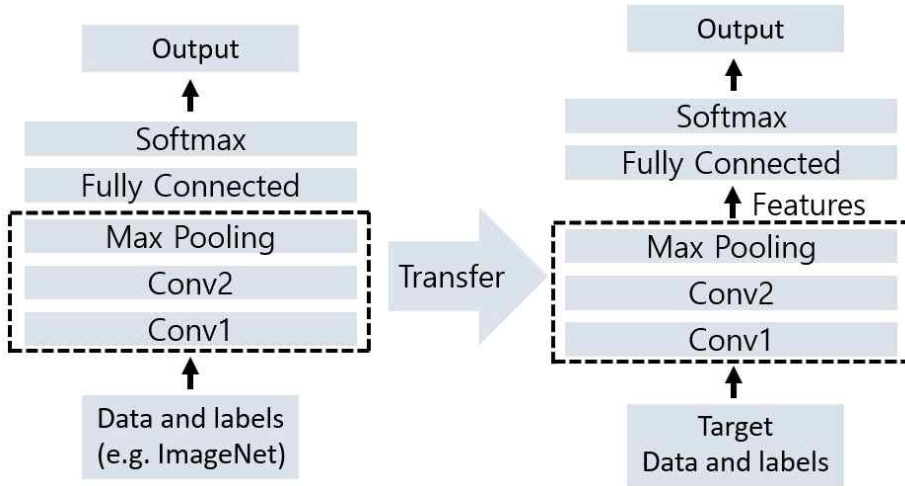


그림 16. 전이 학습 방법

좋은 성능을 내는 딥러닝 모델을 만드는데 한계로 지적되는 가장 큰 이유는 데이터가 많이 필요하다는 것이다. 이를 해결하기 위해서 4-1에서 언급한대로 크롤링으로 데이터를 더 모으거나 데이터 어그멘테이션을 적용해 새로운 이미지를 만들어 내기도 한다. 그런데 이와는 다르게 데이터에 초점을 맞추지 않고 모델에 초점을 맞춰 좋은 성능을 내는 방법이 있다. 미리 많은 양의 이미지로 훈련시킨 모델에서 이미지의 특징을 추출하는 계층만을 가져와 대상 도메인에 맞게 최상단에 계층을 쌓는 방법이다. 이를 전이 학습이라고 한다 (그림 16 참고). 많은 연구에서 전이 학습을 이용해서 데이터가 부족해서 발생하는 문제를 해결하고 있다. Zhongling Huang, Zongxu Pan, Bin Lei는 그들의 연구에서 Synthetic Aperture Radar (SAR) 이미지를 분류하는 CNN 모델을 만드는데 레이블링된 SAR 데이터가 적다는 문제에 직면 한 바 있다. 그들은 레이블링 되어 있지 않은 많은 SAR scene 이미지 데이터에서 특징을 추출해내는 모델을 학습시켜 대상 SAR 데이터를 분류해내는 전이 학습 기반의 방법을 제안했다 [45]. 이한수, 김종근, 유정원 등은 레이더를 이용한 기상 관측에서 이상전파에코를 식별 및 분류 하

는 문제를 딥러닝 모델로 해결하려는 시도를 했다. 문제는 실제 환경에서 준수한 성능의 모델을 구현할 만큼 데이터를 확보하지 못한다는 점이었다. 이에 전이 학습 기반으로 모델을 구현하여 우수한 성능의 분류 모델을 구현할 수 있음을 제시했다 [46]. 보통 전이 학습 모델을 구축하면 이미지를 추출하는 계층은 대상 도메인의 데이터로 네트워크를 학습시킬 때 학습이 되지 않도록 한다. 왜냐하면 이미 많은 양의 데이터로 학습을 완료한 상태이기 때문에 충분히 이미지의 추상적인 특징을 잘 찾아낼 수 있기 때문이다. 오히려 학습을 시킬 경우 대상 이미지의 특징 때문에 과적합이 일어날 가능성이 커진다. 그런데 이미지 추출 계층의 최상단은 이미지 내 객체의 좀 더 추상적인 특징을 확인한다. 그래서 이미지 추출 계층의 최상단 층과 분류를 진행하는 계층을 대상 도메인에 맞춰 학습시켜 좀 더 정확도를 높이는 방법이 있다. 이를 Fine tuning이라고 한다. 본 연구에서도 ImageNet으로 미리 훈련시킨 MobileNet 모델을 가져와서 앵무새 이미지 데이터로 Fine tuning 시킨 모델을 사용했다.

(3) Global Average Pooling

Lin Min, Qiang Chen, Shuicheng Yan는 그들의 연구에서 입력 Feature map의 채널 별 평균값으로 벡터를 만드는 연산인 Global average pooling 연산을 제안했다 (그림 17 참고) [47]. Global average pooling은 학습하는 파라미터가 없기 때문에 과적합의 위험이 없고 Feature map의 공간 정보를 축약하기 때문에 객체가 이미지 내에서 어떤 평행이동을 하던 관련 특징을 뽑아낼 수 있다는 장점이 있다. 또 단순히 평균값을 구하면 되기 때문에 연산도 간단하다. 본 연구에서도 이런 점을 이용하기 위해서 Dropout 계층 대신에, 분류를 수행하는 최종 계층 바로 전에 Global average pooling 연산을 추가했다.

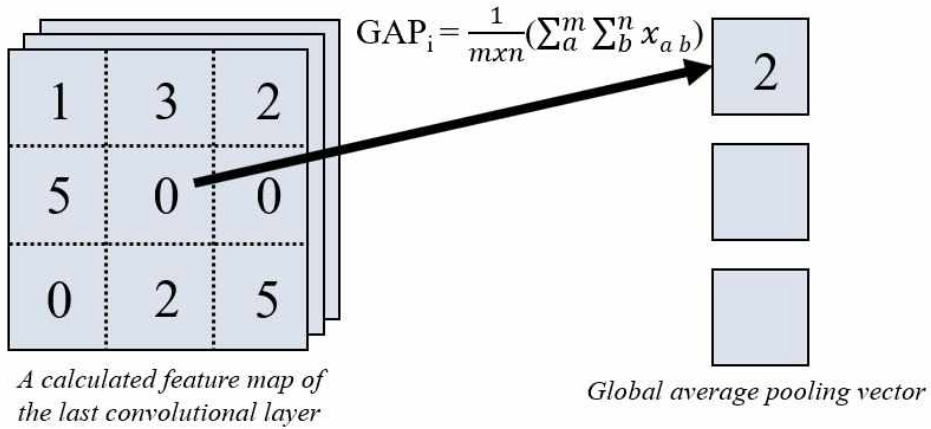


그림17. Global average pooling 방법. 그림 속 수식에서 m과 n의 Feature map의 너비와 높이를 나타내고 x_{ab} 는 Feature map에서 해당 인덱스의 값을, GAP_i 는 벡터에서 해당 인덱스의 값을 나타낸다.

4-3. 모델 훈련

본 연구에서 생성한 모델은 다음의 설정으로 훈련시켰다. 먼저 손실 함수는 Categorical crossentropy를 사용했고 Optimizer strategy는 Learning rate 0.0003의 Adam을 사용했다. 배치 사이즈는 24이고 빠른 훈련을 위해서 3 에폭동안 검증 셋의 정확도가 개선되지 않으면 Learning rate 0.03 줄이고 5 에폭동안 검증 셋의 정확도가 개선되지 않으면 학습을 종료하는 스케줄링을 적용했다.

5. 모델 평가

5-1. 평가 데이터 준비

5-1-1. One class SVM

One class classification을 수행하는 SVM의 성능을 확인하기 위해서 그림 18과 같이 이미지 데이터를 구성했다. 데이터는 훈련이나 검증 데이터 셋에 들어가지 않은 앵무새 이미지와 앵무새가 아닌 이미지로 구성되어 있다. 테스트 이미지는 총 10,000장으로 구성되어 있다.



그림18. One class SVM 테스트 이미지 데이터 셋. 범주는 사람, 동물, 탈것 등 임의의 갈래로 구성했다.

5-1-2. Multi class CNN

멸종 위기 종 앵무새 5종에 대한 Multi class classification 수행하는 CNN의 성능을 확인하기 위해서 아래 그림 19와 같이 데이터를 구성했다. 4-1-1과 4-1-2에서 언급한 대로 동물원에서 촬영한 앵무새의 이미지와 포털 사이트에서 크롤링한 이미지가 있다. 4-1-1에서 동물원에서 촬영한 데이터가 매우 유사하기 때문에 실제로 테스트에 사용한 이미지는 각 종당 10장 정도라고 한 바 있다. 여기에 크롤링으로 획득한 데이터 중 95장을 더해서 105장으로 테스트를 수행했다.



그림19. Multi class CNN의 테스트 이미지 데이터 셋. 데이터는 훈련 및 검증 데이터 셋에 쓰이지 않은 크롤링 데이터와 동물원 촬영 데이터로 구성되어 있다.

5-2. 평가 방법

5-2-1. One class SVM

앵무새와 앵무새가 아닌 이미지의 분류 모델에서는 그 성능을 평가할 때 ROC AUC(Area Under the Receiver Operating Characteristic Curve)를 이용해서 평가한다. ROC AUC는 보통 이진 분류에서 많이 쓰이는 성능 평가 척도이다.

		Prediction	
		Positive	Negative
Actual	Positive	True Positive	False Negative
	Negative	False Positive	True Negative

표4. Confusion matrix

표4는 Confusion matrix를 나타낸다. 각 셀이 의미하는 바는 다음과 같다. True Positive는 실제로 데이터가 실제로 Positive이고 모델이 Positive라고 예측한 빈도이고 False Negative는 데이터는 실제로 Positive이지만 모델이 Negative라고 예측한 빈도수이다. False Positive는 실제로는 Negative이지만 모델이 Positive라고 예측한 빈도수이고 True Negative는 실제로 Negative이고 모델이 Negative라고 예측한 빈도수이다. True Positive나 True Negative는 모델이 정확하게 예측했다고 할 수 있는 경우이고 나머지는 잘못 예측한 경우라고 할 수 있다. 보통 모델의 정확도를 단순히 계산한다고 한다면 총 맞게 예측한 빈도수를 총 빈도수로 나누면 되지만 문제는 클래스간의 불균형이 존재하는 경우가 있다는 것이다. 예를 들어서 암환자와 정상인 환자를 분류하는 모델을 만들 때 데이터가 암환자와 정상인의 데이터 비율이 1:9라면 모두 정상으로 판별해도 90 퍼센트의 정확도를 가지는 모델이 된다. 이런 경우를 고려하여 단순 정확도보다 더 세밀하게 모델의 성능을 측정하는 방법 중에 하나가 ROC AUC이다. ROC는 진짜 양성 비

율(재현율)에 대한 거짓 양성 비율을 나타낸다. 진짜 양성 비율과 거짓 양성 비율은 아래의 식 5, 6과 같이 계산한다.

$$\text{진짜양성비율 (True Positive Rate)} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \quad (5)$$

$$\text{거짓양성비율 (False Positive Rate)} = \frac{\text{False Positive}}{\text{False Positive} + \text{True Negative}} \quad (6)$$

ROC 곡선을 아래 그림 20과 같이 그렸을 때 그 면적이 ROC AUC가 된다. 이 ROC AUC가 넓을수록 정확한 모델이라고 할 수 있다. 그림 20의 그래프의 총 면적은 1이다.

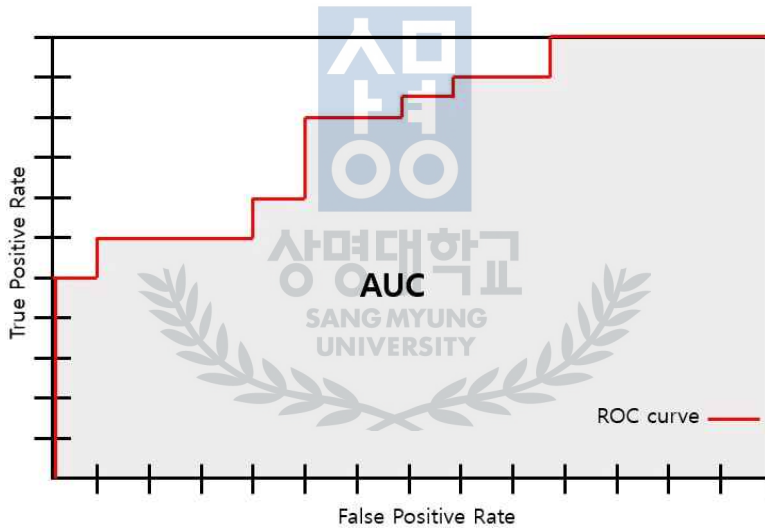


그림20. ROC AUC

5-2-2. Multi class CNN

입력 이미지가 앵무새라고 판단이 되었을 경우, 4-2-2에서 서술한 대로 CNN으로 종관별을 수행한다. 이 모델의 성능을 평가하기 위해서 Precision, Recall, F1 score를 이용한다. 특히 F1 score는 Precision과 Recall을 고려해서 데이터 클래

스가 불균형일 때도 이를 고려하여 모델의 성능을 종합적으로 평가할 수 있다. (표 4 참고) Precision은 True Positive의 빈도수를 True Positive와 False Positive의 빈도수의 합으로 나눈 것으로 모델이 얼마나 정확하게 Positive를 맞췄는가에 초점을 둔다 (식 7 참고). Recall의 경우 True Positive의 빈도수를 True Positive와 False Negative의 빈도수의 합으로 나눈 것으로 모델이 얼마나 많은 Positive를 맞췄는가에 초점을 둔다 (식 8 참고). 그리고 이를 Precision과 Recall을 종합적으로 고려한 것이 F1 score이다. F1 score는 식 9와 같이 구한다.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \quad (7)$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \quad (8)$$

$$F1score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (9)$$

6. 결과

6-1. 시스템 개발 결과

6-1-1. 모바일 및 딥러닝 서버 구축 결과

본 연구에서 개발한 시스템의 결과 화면은 그림 21과 같다. 상단의 왼쪽 그림과 같이, 앱을 실행하면 이미지 프리뷰 화면과 캡처 버튼이 있다. 사용자가 버튼을 누르면 캡처한 이미지의 분석을 서버에 요청한다. 서버에서 결과가 오는 동안, 상단의 오른쪽과 같이 Kotlin의 코루틴으로 처리중이라는 다이얼로그를 보여준다. 서버에서 결과가 오면 하단의 왼쪽과 같이 캡처한 이미지, 분석 결과 그리고 저장하기 버튼과 뒤로 가기 버튼을 보여준다. 만약에 가장 확률이 높은 앵무새 종의 클래스 이름을 터치하면 그 앵무새 종과 관련된 페이지로 넘어간다. 저장하기 버튼을 터치하면 앱에서 서버로 캡처한 이미지를 클라우드 스토리지에 저장하도록 요청한다. 스토리지 저장 결과는 그림 22와 같다.

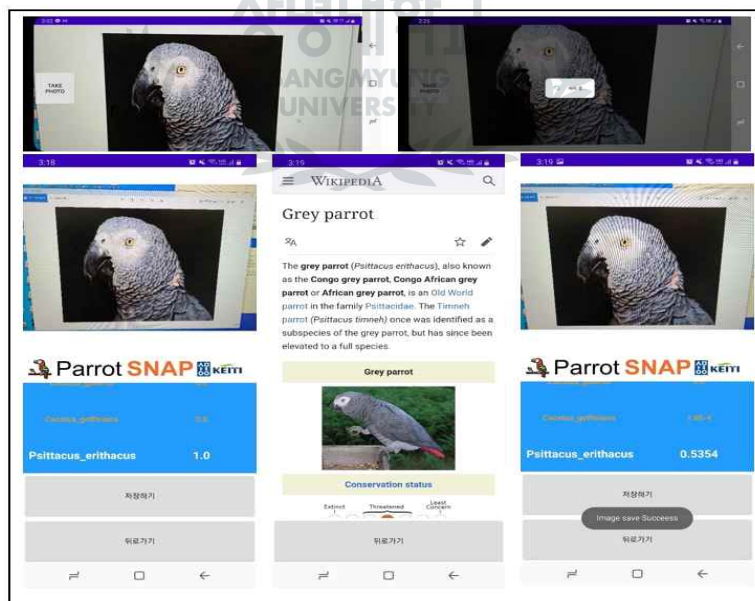


그림21. 모바일 및 딥러닝 서버 작업 수행 결과.

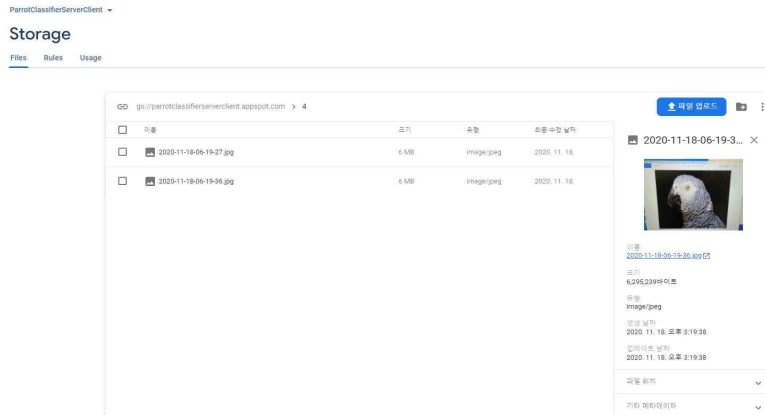


그림22. 클라우드 스토리지 저장 결과.

6-1-2. 시스템 속도 측정 결과

	Network	One class	Multi class	Save
Duration time	5120ms	85ms	103ms	4220ms

표5. 시스템의 각 요소별 소요시간.

본 연구에서 측정한 시스템의 속도는 표5와 같다. 속도는 각 요소별로 평균값을 구했다. 표 5를 보면 알 수 있듯이 네트워크상의 딜레이 타임 ($5120 - (85 + 103) = 4932$)이 약 5초로 병목현상이 일어나는 것을 확인 할 수 있다.

6-2. 모델 평가 결과

6-2-1. One class SVM

본 연구에서 생성한 One class SVM의 성능을 평가할 때는 Scikit-learn 라이브러리를 이용했다 [43]. Scikit-learn 라이브러리로 계산한 Confusion matrix는 그림 23과 같고 ROC AUC는 0.9189가 나왔다. 이는 Pramuditha Perera, Vishal M. Patel의 연구 사례에서 가장 높은 ROC AUC 성능이 0.956인 것을 감안할 때 본 연구의 SVM 모델도 비슷한 성능을 보임을 확인 할 수 있다 [20]. 이미지의 특징은 2, 3차원

의 그래프로 표현하기 어렵기 때문에 본 연구에서 생성한 One class SVM의 그림 20과 같은 그래프는 생략했다. 앵무새인 이미지를 앵무새라고 예측한 결과가 2011건이고 앵무새가 아닌 이미지를 앵무새가 아니라고 예측한 결과가 6805건으로 잘못 예측한 빈도에 비해서 상당수를 잘 예측한 것을 확인할 수 있다.



그림23. One class SVM Confusion matrix 결과. 이미지 용량 때문에 결과 행렬을 그릴 때 훈련시킨 모델에 더 적은 수의 테스트 이미지를 넣어 도출했다.

6-2-2. Multi class CNN

본 연구에서 구현한 다중 분류 모델의 Confusion matrix는 아래 표 6과 같다. 행렬에서 1행 1열을 기준으로 우 하향 대각선은 모두 맞게 예측한 것이고 나머지는 잘못 예측한 것이다. 대부분의 경우가 이 우 하향 대각선에 분포하고 있음을 확인할 수 있다. 이를 통해서 1-1-4에 세웠던 가설인 “비슷한 색이나 모양의 객체도 CNN 모델이 분류 할 수 있다” 주장이 옳다고 볼 수 있다. 다만 이 수치는 모델의 아키텍처를 바꾸거나 최적화를 수행할 경우 더 개선될 것으로 기대한다.

Prediction \ Actual	Ara ararauna	Ara chloroptera	Cacatua galerita	Cacatua goffiniana	Psittacus erithacus
Ara ararauna	87	5	7	2	4
Ara chloroptera	0	97	0	2	6
Cacatua galerita	1	0	98	3	3
Cacatua goffiniana	0	1	9	95	0
Psittacus erithacus	2	3	1	0	99

표6. Multi class CNN의 Confusion matrix.

이를 바탕으로 Precision, Recall, F1 score을 구한 값은 표 7에 나와 있다. 대부분의 값이 90 퍼센트에 근접한 값이지만 청금강앵무의 Recall이 0.83으로 떨어져 있는데 표5의 첫 번째 행을 보면 청금강앵무의 정확도가 다른 종에 비해 떨어지는 것을 확인할 수 있다. 표 6과 표 7의 결과는 Scikit-learn 라이브러리를 이용해서 구한 값이다. 본 연구에 앞서 선행 연구로 앵무새 4종에 대해서 실험을 수행한 바 있는데 이때와 성능이 유사함을 확인할 수 있다 [28].

	Precision	Recall	F1 score	Images
Ara ararauna	0.97	0.83	0.89	105
Ara chloroptera	0.92	0.92	0.92	105
Cacatua galerita	0.85	0.93	0.89	105
Cacatua goffiniana	0.93	0.90	0.92	105
Psittacus erithacus	0.88	0.94	0.91	105

표7. Multi class CNN의 Precision, Recall, F1 score.

모델 학습 중의 에폭 당 손실 값과 정확도를 측정한 값은 그림 24에 나와 있다. 여기서 Loss, Acc는 훈련 데이터 셋에 해당하는 값이고 Val loss, Val acc는 검증 데이터 셋에 해당하는 값이다. 4-3에서 언급한 스케줄링을 설정한 결과 13에폭 동안 학습이 이뤄졌다. 그림 24를 보면, 약 6 에폭부터 모델의 성능이 수렴됨을 확인할 수 있다.

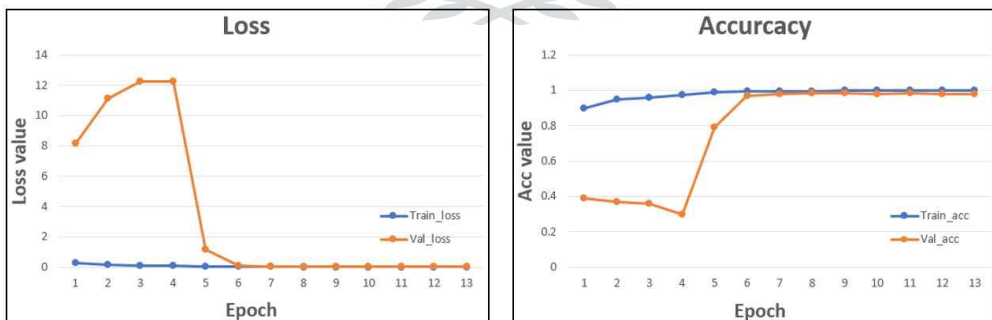


그림24. Multi class CNN Loss 및 Accuracy.

7. 결론

본 연구에서는 컴퓨터 비전의 분류 모델을 사용해서 멸종위기 종, 그 중에서도 앵무새 5종을 분류하는 시스템을 개발했다. 본 연구에서 개발한 시스템이 가지는 의의는 다음과 같다. 첫째, 다소 주관적일 수 있는 인간의 판단을 보조하기 위해서 객관적으로 이미지를 분류할 수 있다. 둘째, 본 시스템이 생물 다양성 보존 연구에 필요한 영상 데이터 구축 및 불법 밀수 모니터링 시스템 같은 더 큰 시스템의 일부로 쓰일 수 있다. 셋째, 더 규모가 큰 시스템의 일부로 쓰일 시에 대용량의 영상 데이터 분류 작업을, 인간의 노력을 최소화 하여 자동화 할 수 있다. 특히 기존의 연구의 한계로 지적되어온 점은 CNN의 Softmax 함수의 한계로 인해서 정답이 없는 경우에도 학습한 정답 후보군에서 최대한 확률이 높은 잘못된 정답을 제시한다는 점이다. 본 시스템은 이런 한계점을 극복하기 위해서 다중 클래스 분류 전에 먼저 정답 후보군인지 아닌지를 판단하는 단일 클래스 분류를 실시한다.

다만 본 연구에서는 시간 관계상 모델의 성능을 최적화 하는 작업을 수행하지 못했다. MobileNet 대신 다른 아키텍처를 적용해보는 연구, 네트워크의 각 하이퍼파라미터를 최적화 하는 연구를 수행하면 성능이 더 개선될 것으로 기대한다. 뿐만 아니라 데이터를 모으는 방법에는, 존재하지 않는 이미지를 대상 멸종 위기 종에 맞춰 꽤 그럴듯하게 만들어내는 모델을 구축하여 이미지를 획득하는 방법이 있다 [48, 49]. 이런 모델 구축 과정이 다소 복잡하기 때문에 본 연구에서는 생략하지만 생성해낸 이미지를 추가하여 분류 모델을 학습시킨다면 과적합의 위험성이 줄어들고 모델이 대상 멸종 위기 종의 더 일반적인 특징을 추출 할 수 있을 것으로 기대한다. 또 본 연구에서는 SVM을 이용하여 이중 분류를 실시했는데 문헌에 따르면 Autoencoder로 이상치를 탐지해내는 단일 클래스 분류를 실시할 수 있다고 한다 [50]. 시스템 성능 향상을 위해서 후속 연구에서 이런 방법을 시도해볼 예정이다.

참 고 문 헌

- [1] 대한민국 환경부 국가생물다양성위원회, “제4차 국가생물다양성전략”, 인천광역시, 국가생물다양성센터, 2018
- [2] Organization for Economic Cooperation and Development, “OECD Environmental Outlook to 2050: The Consequences of Inaction”, OECD, 2012
- [3] 김유리, “[인천세관]국립생물자원관과 뱀 등 수입생물 유해성 확인 협력”, 한국세정신문, 2019, 05, 10
- [4] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel. “Backpropagation applied to handwritten zip code recognition.” *Neural Comp.*, 1989.
- [5] 이예하, 김현준, 김국배 외, “의료영상 분석을 위한 딥러닝 기반의 특징 추출 방법”, 대한의학영상정보학회지 2014;20:1-12
- [6] 김상욱, 김휘영, “딥러닝 기반의 의료영상 분석 성능에 대한 영상처리를 통한 데이터 증강방법의 효과에관한연구”, 한국통신학회 학술대회논문집, Vol.2018No.11[2018], 278-279
- [7] 박현정, 남기표, 김익재, “딥러닝 기반 포즈 변화에 강인한 귀 인식 연구”, 대한전자공학회 논문지 56(8), 2019.8, 47-55
- [8] 최광미, 정유정, “딥러닝 합성곱 신경망을 이용한 효율적인 홍채인식”, *Journal of the KIECS*. pp. 521-526, vol. 15, no. 3, Jun. 30. 2020
- [9] X. Wu, D. Sahoo, S. C.H. Hoi, "Recent advances in deep learning for object detection", *ScienceDirect, Neurocomputig, Volume 396, 5 July 2020, Pages 39-64, <https://doi.org/10.1016/j.neucom.2020.01.085>*
- [10] 한상훈, 장진희, 강길욱, 박한술, “IP카메라 해킹 분석과 대책”, 한국컴퓨터정보학회 학술발표논문집 26(1), 2018.1, 165-166
- [11] S. F. Pires, “The illegal parrot trade: a literature review. *Global Crime*” *Journal of Global Crime*, vol 13, no 3, pp 176-190, 2012, DOI: 10.1080/17440572.2012.700180
- [12] Convention on International Trade in Endangered Species of Wild Fauna and Flora, What is CITES?, Available: <https://www.cites.org/eng/disc/what.php>
- [13] 유병혁, 김선현, “딥러닝을 이용한 야생동물 자동식별 - 소백산국립공원 죽령생태통로 카메라 트랩 영상분석 사례”, 한국환경생태학회 학술발표논문집, 2018, 1, 34-35.
- [14] P. Zhuang, L. Xing, Y. Liu, S. Guo, Y. Qiao, "Marine Animal Detection and Recognition with Advanced Deep Learning Models", 2017 CLEF Working Notes of CLEF 2017 - Conference and Labs of the Evaluation Forum, vol. 1866, 2017
- [15] H. Nguyen, S. J. Maclagan, T. D. Nguyen, T. Nguyen, P. Flemons, K. Andrews, E. G. Ritchie, D. Phung “Animal Recognition and Identification with Deep Convolutional Neural Networks for Automated Wildlife Monitoring” in *Proceedings of the 2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, Tokyo, Japan, October 2017, DOI: 10.1109/DSAA.2017.31
- [16] A. Swanson, M. Kosmala, C. Lintott, R. Simpson, A. Smith and C. Packer "Snapshot Serengeti, high-frequency annotated camera trap images of 40 mammalian species in an African savanna" *Sci Data* 2, 150026 (2015). <https://doi.org/10.1038/sdata.2015.26>
- [17] M. S. Norouzzadeh, A. Nguyen, M. Kosmala, A. Swanson, M. S. Palmer, C. Packer and J. Clune

- “Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning” *Proceedings of the National Academy of Sciences*, vol. 115, no. 25, pp. E5716–E5725, 2018, DOI: 10.1073/pnas.1719367115
- [18] A. Sehgal, N. Kehtarnavaz "Guidelines and Benchmarks for Deployment of Deep Learning Models on Smartphones as Real-Time Apps" 2019, arXiv:1901.02144
- [19] L. Juranek, J. Stastny, V. Skorpil and L. Junek, "Acceleration of Server-side Image Processing by Client-side Pre-processing in Web Application Environment" 2019 42nd International Conference on Telecommunications and Signal Processing (TSP), Budapest, Hungary, 2019, pp. 127-130, doi: 10.1109/TSP.2019.8768889.
- [20] P. Perera and V. M. Patel, "Learning Deep Features for One-Class Classification" in *IEEE Transactions on Image Processing*, vol. 28, no. 11, pp. 5450-5463, Nov. 2019, doi: 10.1109/TIP.2019.2917862.
- [21] Larry M. Manevitz "One-class svms for document classification" *The Journal of Machine Learning Research*, Vol. 2, March 2002
- [22] Y. Lecun, L.D. Jackel, L. Bottou, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, U.A. Muller, E. Sackinger, P. Simard, V. Vapnik "Learning algorithms for classification: A comparison on handwritten digit recognition" In J. H. Oh, C. Kwon, & S. Cho (Eds.), *Neural networks: The statistical mechanics perspective*, World Scientific, pp. 261-276, 1995
- [23] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "Imagenet classification with deep convolutional neural networks" In *Advances in neural information processing systems*, pages 1097–1105, 2012
- [24] K. Simonyan and A. Zisserman. "Very deep convolutional networks for large-scale image recognition" arXiv preprint arXiv:1409.1556, 2014
- [25] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. "Going deeper with convolutions" In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015
- [26] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, pp. 770-778, 2016, DOI: 10.1109/CVPR.2016.90.
- [27] A. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications." 2017, arXiv preprint arXiv:1704.04861.
- [28] D. G. Cheo, E. J. Choi and D. K. Kim. "The Real-Time Mobile Application for Classifying of Endangered Parrot Species Using the CNN Models Based on Transfer Learning." *Mobile Information Systems*, 1475164:1-1475164:13, 2020, DOI: 10.1155/2020/1475164
- [29] Flask, "Flask Documentations" [Internet], <https://flask.palletsprojects.com/en/1.1.x/>
- [30] Tensorflow, "Tensorflow Serving" [Internet], <https://tensorflowkorea.gitbooks.io/tensorflow-kr/content/g3doc/tutorials/tfserve/>
- [31] F. Chen, N. Chen, H. Mao and H. Hu "Assessing four Neural Networks on Handwritten Digit Recognition Dataset (MNIST)." 2018, arXiv preprint arXiv:1811.08278
- [32] A. Krizhevsky, I. Sutskever and G. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks." *Neural Information Processing Systems*. Vol 25. 2012, DOI: 10.1145/3065386.

- [33] J. Deng, W. Dong, R. Socher, L. Li, Kai Li and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, pp. 248-255, 2009, DOI: 10.1109/CVPR.2009.5206848
- [34] Selenium, "The Selenium Browser Automation Project" [Internet], <https://www.selenium.dev/documentation/en/>
- [35] N. Sun, X. Mo, T. Wei, D. Zhang and W. Luo "The Effectiveness of Noise in Data Augmentation for Fine-Grained Image Classification." ACPR 2019: Pattern Recognition, Springer, Cham, pp 779-792, 2020, DOI: 10.1007/978-3-030-41404-7_55
- [36] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, pp. 580-587, 2014, DOI: 10.1109/CVPR.2014.81.
- [37] 민정기, 문종섭. "감쇠 요소가 적용된 데이터 어그멘테이션을 이용한 대체 모델 학습과 적대적 데이터 생성 방법." 정보보호학회논문지, 29(6), 1383-1392. 2019
- [38] 신흥재, 이소인, 정희원, 박장우. "딥러닝을 활용한 실내 식물 이미지 분류 및 식물 정보 제공 웹 어플리케이션." 한국지식정보기술학회, Vol. 15, No. 2, pp. 167-175, ISSN 1975-7700, <https://doi.org/10.34163/jkits.2020.15.2.002>, April 2020
- [39] J. Dai, Y. Li, K. He, J. Sun "R-FCN: Object Detection via Region-based Fully Convolutional Networks", Fri, 20 May 2016, arXiv:1605.06409
- [40] imgaug, "imgaug Documentation" [Internet], <https://imgaug.readthedocs.io/en/latest/>
- [41] R. Chalapathy, S. Chawla "Deep Learning for Anomaly Detection: A Survey" Thu, 10 Jan 2019, arXiv:1901.03407
- [42] I.T. Jolliffe, J. Cadima "Principal component analysis: a review and recent developments" Philos Trans A Math Phys Eng Sci. 2016;374:20150202. doi: 10.1098/rsta.2015.0202.
- [43] Scikit-learn, "Scikit-learn Machine Learning in Python" [Internet], <https://scikit-learn.org/stable/>
- [44] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 1800-1807, doi: 10.1109/CVPR.2017.195.
- [45] Z. Huang, Z. Pan, and B. Lei, "Transfer learning with deep convolutional neural network for SAR target classification with limited labeled data" Remote Sensing, vol. 9, no. 9, p. 907, 2017. DOI: 10.3390/rs9090907
- [46] 이한수, 김종근, 유정원, 정영상, 김성신. "전이학습 기반의 합성곱 신경망을 이용한 다중클래스 분류에 관한 연구." 한국지능시스템학회 논문지, 28(6), 531-537. 2018
- [47] L. Min, Q. Chen, S. Yan "Network in Network." CoRR abs/1312.4400, 2013
- [48] 이호진, 박승태, 박범수, 정해동, 이승철. "데이터 불균형 문제를 해결하기 위한 Generative Adversarial Networks (GAN) 기반의 가상 데이터 합성." 대한기계학회 춘추학술대회, 1142-1143. 2017
- [49] 정성욱, 김이슬, 김일곤, 임경식. "GAN을 이용한 데이터 불균형 문제에 관한 연구." 한국통신학회 학술대회논문집, 1390-1391. 2019
- [50] S. Y. Shin, H. j. Kim "Autoencoder-based One-class Classification Technique for Event Prediction." In Proceedings of the 2019 4th International Conference on Cloud Computing and Internet of Things. Association for Computing Machinery, New York, NY, USA, 54-58. 2019

ABSTRACT

Deep Learning-based Endangered Species Classification App System

Dae Gyu Choe

Dept. of Computer Science

The Graduate School

Sangmyung University

In this paper, we present a process of building an app system that classifies image data of endangered species using deep learning methods. According to the report of the National Biodiversity Committee under the Ministry of Environment of the Republic of Korea in 2018, It is less than 50 percent of the achievement rate for the biodiversity DB establishment and monitoring plans. In addition, it is told that Incheon customs has difficulties for identifying endangered species that have been illegally smuggled.

Therefore, in this study, we developed an app system whose computer helps the experts to classify the images of endangered species objectively. As a cornerstone, it is expected that a bigger system which automates the identification of large capacity image data and monitors illegal smuggling can be established. The limitations of

previous related studies are as follows. Due to the nature of CNN's Softxmax function, even if the class of the image is not in the answer candidates group, the model must present an incorrect answer which is closest to the image inevitably. To overcome this, the system first determines whether the class of the image is in the group or not, and then classifies the class if the class in the group.

The accuracy of the one class classification model is 0.9189 according to ROC AUC and the accuracy of the multi-class classification model is 0.91 according to F1 score. System response speed is an average of 5120ms for image analysis and 4220ms for image storage.

