

캡스톤 디자인 I

종합설계 프로젝트

프로젝트 명	동영상 연령제한 검열
팀 명	YouHi
문서 제목	중간보고서

Version	2.1
Date	2020-05-28

팀원	이태훈 (조장)
	이인평
	이주형
	김성수
	김민재
지도교수	임은진 교수

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22


CONFIDENTIALITY/SECURITY WARNING

이 문서에 포함되어 있는 정보는 국민대학교 전자정보통신대학 컴퓨터공학부 및 컴퓨터공학부 개설 교과목 캡스톤 디자인I 수강 학생 중 프로젝트 "You Hi"를 수행하는 팀 " You Hi "의 팀원들의 자산입니다. 국민대학교 컴퓨터공학부 및 팀 " You Hi "의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다.

문서 정보 / 수정 내역


Filename	2020-4조_중간보고서_2차.doc
원안작성자	이태훈
수정작성자	이인평, 이주형, 김성수, 김민재

수정날짜	대표수정자	Revision	추가/수정 항목	내 용
2020-5-15	이태훈	1.0	최초 작성	2차 중간보고서 초고 작성
2020-5-16	이주형	1.1	내용 추가	음성 필터링 관련 내용 추가
2020-5-18	이태훈	1.2	내용 추가	영상 필터링 관련 내용 추가
2020-5-18	이인평	1.3	내용 추가	프론트엔드 관련 내용 추가
2020-5-19	이태훈	1.4	내용 추가	백엔드 관련 내용 추가
2020-05-19	이주형	1.5	내용 추가	음성 마스킹, 자막 검열 내용 추가
2020-05-25	이주형	2.0	내용 수정	오탈자 및 비문 수정
2020-5-28	이태훈	2.1	내용 수정	최종 검토

	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22

목 차


1	프로젝트 목표	4
1.1	목표	4
1.2	목표에 따른 기대효과	5
2	수행 내용 및 중간결과	6
2.1	계획서 상의 연구내용	6
2.1.1	Object Detection	6
2.1.2	Instance Segmentation	7
2.1.3	이미지 분류	8
2.1.4	영상 검열	10
2.1.5	음성 인식	11
2.1.6	형태소 분석	12
2.1.7	욕설 검열	13
2.1.8	Web Front End	15
2.1.9	Back End	16
2.2	수행내용	16
2.2.1	Instance Segmentation	16
2.2.2	Object Detection	19
2.2.3	이미지 분류	20
2.2.4	영상 검열	22
2.2.5	음성 마스킹	25
2.2.6	자막 욕설 검열	25
2.2.7	Web Front End	27
2.2.8	Back End	30
3	수정된 연구내용 및 추진 방향	32
3.1	수정사항	32
3.1.1	영상 검열 방식 변경	32
3.1.2	음성 마스킹 및 자막 검열 기능 추가	33
4	향후 추진계획	34
4.1	향후 계획의 세부 내용	34
4.1.1	Front End	34
4.1.2	Back End	34
4.1.3	정확도 측정	34
5	고충 및 건의사항	35

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22

1 프로젝트 목표

1.1 목표

1. YouTube 및 실시간 방송 플랫폼, 방송통신위원회 등의 유해 기준을 참고하여 동영상에 미성년자 시청 불가능 장면/욕설이 있는지 판단한다.
2. 현재 YouTube <노란 딱지> 정책 기준의 모호성과 불공정성에 대한 문제를 해결하기 위해 사용자가 직접 자신의 영상 중 어떤 부분이 부적절한지 확인이 가능하게 한다. 영상 제작자(크리에이터) 등의 영상 제작의 효율성을 높인다.(<노란 딱지>는 YouTube에 존재하는 기존 AI 영상 등급 검열 장치이다. 최근, <노란 딱지> 시스템의 필터링 기준이 모호하여 동일한 내용의 영상들 사이에서도 유해 영상 판정을 받는 영상과 그렇지 않은 영상들이 존재하며 영상 제작자가 자신의 영상 내용 중 유해 판정을 받은 부분을 직접 확인할 수 가 없어 사용자들 간 많은 불만을 사고 있다.)
3. 동영상에 미성년자가 시청 불가능한 장면이 있는지 판단되면 해당 장면이 어떤 가이드 라인을 위반했는지 알려준다.
4. 현재 YouTube에 존재하는 다양한 가이드라인 중 선정성, 폭력성, 모방성(흡연, 욕설 등)에 대한 가이드라인을 충족시키는 지 중점적으로 확인하는 검열을 실시한다.
 - 1) "사람이 칼에 찔리는 장면이 명확하게 표현된 경우" - 폭력성
 - 2) "흡연하는 장면이 명확하게 표현된 경우" - 모방성
 - 3) "지나치게 특정 대상을 비방하거나 과도한 욕설이 포함된 경우" - 모방성
 - 4) "여성 혹은 남성의 노출 강도가 심한 경우" - 선정성
5. 본 프로젝트의 최종 목표는 다양한 영상 플랫폼에서 이 시스템을 사용할 수 있도록 정확도를 높이고, 검열 과정에 너무 긴 시간이 소요되지 않도록 개선하는 것이다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22

1.2 목표에 따른 기대효과

검열 시스템은 Youtube 의 가이드 라인에 맞춰서 영상과 음성을 검열한다. 이는 현재 신고 기반으로 이루어지는 수작업 검열 과정보다 효율적으로 작동된다. 업로드 이전에 일어나는 자동 검열 시스템이므로 다양한 기대효과와 활용방안이 있다.

1. 검열 시스템을 적용함으로써 청소년에게 부적합한 영상들을 일차적으로 검열할 수 있다. 이는 동영상 업로드 플랫폼에서 실시하는 신고 기반 검열 시스템보다 더 많은 영상을 검열할 수 있고, 작업량과 비용 측면에서 효율적이다.
2. 추후 Youtube 이외의 다양한 실시간 스트리밍 서비스(Twitch, Affreca TV 등)에서도 효과적으로 사용할 수 있다. 각 스트리밍 서비스들은 운영진이 직접 실시간 모니터링과 시청자들의 신고를 통해 제재를 가하는데, 모든 과정이 수동 작업으로 이루어지기 때문에 효과적으로 이루어지지 않는다. 따라서 방송되고 있는 장면들을 실시간으로 딥러닝 모델에 넣어 검열할 수 있다.
4. 현재는 미성년자들을 대상으로 부적합한 영상들을 검열하는 시스템을 목표로 하지만, 데이터셋이 많아지고 다양한 라벨에 대해서 학습을 시킨다면, 교통사고, 자연 재해, 길거리 싸움, 집단 구타 및 학대 등 검열 대상을 확대시켜 활용이 가능하다. 또한 유해한 동영상의 업로드를 제한함으로써 사이버 범죄 예방에 도움을 준다.
5. 현 음성 검열 시스템은 단순 욕설 단어에 대해서만 진행되지만, 데이터셋을 확대시켜 학습시킨다면 욕설을 포함하지 않은 비방 목적의 문장 또한 검열이 가능하다.
6. 이 시스템이 적극적으로 활용된다면, 동영상 업로드 플랫폼에 대한 사람들의 신뢰도와 인식의 향상에 도움이 된다.

2 수행 내용 및 중간결과

2.1 계획서 상의 연구내용


2.1.1 Object Detection

객체 탐지는 이미지에서 찾고 싶은 관심 객체를 배경과 구분해 식별하는 자동화 기법이다. 이 기술은 아래의 사진처럼 이미지에서 새와 사람 등의 객체를 자동으로 탐지해낸다. 이러한 객체 탐지는 딥러닝을 통해 이루어진다.

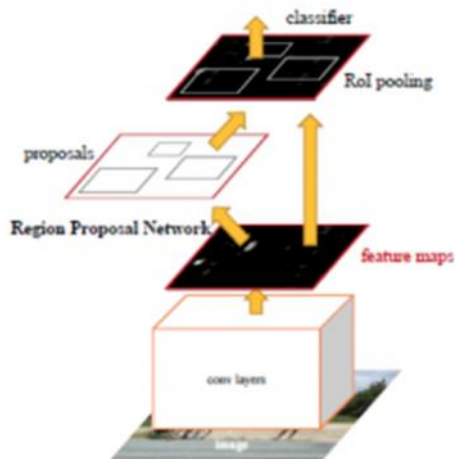


객체 탐지 기술은 2개 이상, 즉 N개의 객체를 탐지해 분류할 수 있어야 한다. 많은 객체를 탐지하는 데 한계가 있으므로 다수의 사각형 상자 위치와 크기를 가정해 컨볼루션 신경망을 변형한 후 이를 객체 분류(Object Classification)에 활용한다. 이러한 사각형 상자를 '윈도우(Window)'라고 부른다. 각 창의 크기와 위치는 객체의 존재 여부에 따라 결정될 수 있고 객체가 있는 경우에는 그 범주도 결정할 수 있다. 본 프로젝트에서는 다음과 같은 알고리즘을 사용한다.

2단계 방식의 객체 탐지 알고리즘, Faster RCNN 알고리즘 이름에 '빠른(Faster)'이라는 단어가 포함되어 있지만 이는 단일 단계 방식보다 빠른 처리가 된다는 뜻이 아닌 이전 버전이라 할 수 있는 RCNN 알고리즘과 Fast RCNN 알고리즘 보다 빠르다는 것을 뜻한다. 각 관심 영역(RoI; Region of Interest)에 대한 피쳐 추출의 계산을 공유하고 딥러닝 기반의

 <div> <p>국민대학교</p> <p>컴퓨터공학부</p> <p>캡스톤 디자인 I</p> </div>	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22

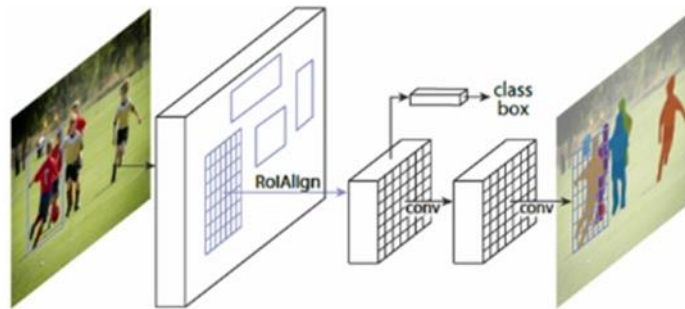
RPN을 도입해 구현한다.



많은 CNN 레이어를 사용해 피쳐 맵 추출이 완료되면 RPN을 통해 개체를 포함하고 있을 가능성이 높은 윈도우가 다량으로 생성된다. 이 후 각 윈도우에 있는 피쳐 맵을 검색하고, 고정 크기로 조정된 뒤(RoI 풀링) 클래스 확률과 해당 객체에 대한 더 정확한 경계박스를 예측한다. RPN은 YOLO와 같은 방식의 앵커 박스를 사용한다. 하지만 YOLO 알고리즘과는 앵커 박스가 데이터로부터 생성되는 것이 아니라 고정된 크기와 형태로 생성된다는 차이가 있다. 이 앵커 박스는 이미지를 보다 조밀하게 커버할 수 있다. RPN은 여러 객체 카테고리들에 대한 분류 대신 윈도우의 객체 포함 유무에 대한 이진 분류(Binary Classification)만 수행한다.

2.1.2 Instance Segmentation

Instance segmentation(객체 분할)이란 경계 상자(Bounding Box)로 각 클래스들을 탐지하는 Object Detection(객체 탐지)보다 더 발전하여, Box 형태 안에 존재하는 실제 고유의 클래스들이 가지는 모양 그대로 객체의 영역만을 탐지하는 기술이다. Instance segmentation은 주어진 이미지 내 각 위치 상의 픽셀들을 하나씩 조사하며, 현재 조사 대상인 픽셀이 어느 특정한 라벨에 해당하는 사물의 일부인 경우 해당 픽셀의 위치에 그 클래스를 나타내는 '값'을 표기하는 방식으로 예측 결과물을 생성한다. 반대로 조사 대상 픽셀이 어느 클래스에도 해당하지 않는다면 어떠한 값도 부여하지 않게 되고 이러한 과정으로 생성된 결과물을 mask라고 한다.




The Mask R-CNN framework for instance segmentation.

Instance segmentation에 사용되는 모델은 R-CNN 계열에 속하는 Mask R-CNN이다. Mask R-CNN 모델의 동작 원리는 다음과 같다. Input으로 들어온 이미지는 Feature Pyramid Network를 통해 여러 feature map들을 생성한다. 생성된 feature map들은 RPN(Region proposal network)에 전달되고 이진 분류기를 이용하여 다중 ROI를 생성한다. 그리고 총 9개 anchor box를 이미지 위에 적용하게 된다. 이진 분류기는 classification score를 반환하게 되고, Non Max suppression 알고리즘을 적용하여 가장 score가 높은 anchor를 제외한 주위에 다른 anchor들은 모두 지운다. RoI Align network에서는 feature map에서 하나가 아닌 여러 개의 bounding box를 출력하고 고정된 차원으로 만든다. 그 후 feature map들은 fully connected layer에 전달되어 softmax를 통해 분류하고 bounding box 예측은 회귀 모델을 이용해 더욱 세분화된다. feature map들은 또한 CNN 2개로 구성된 mask 분류기로 전달되고 각 ROI에 대해 이진 mask를 출력한다. mask 분류기를 통해 network는 클래스 간 경쟁없이 모든 클래스에 대해 mask 생성이 가능하다.

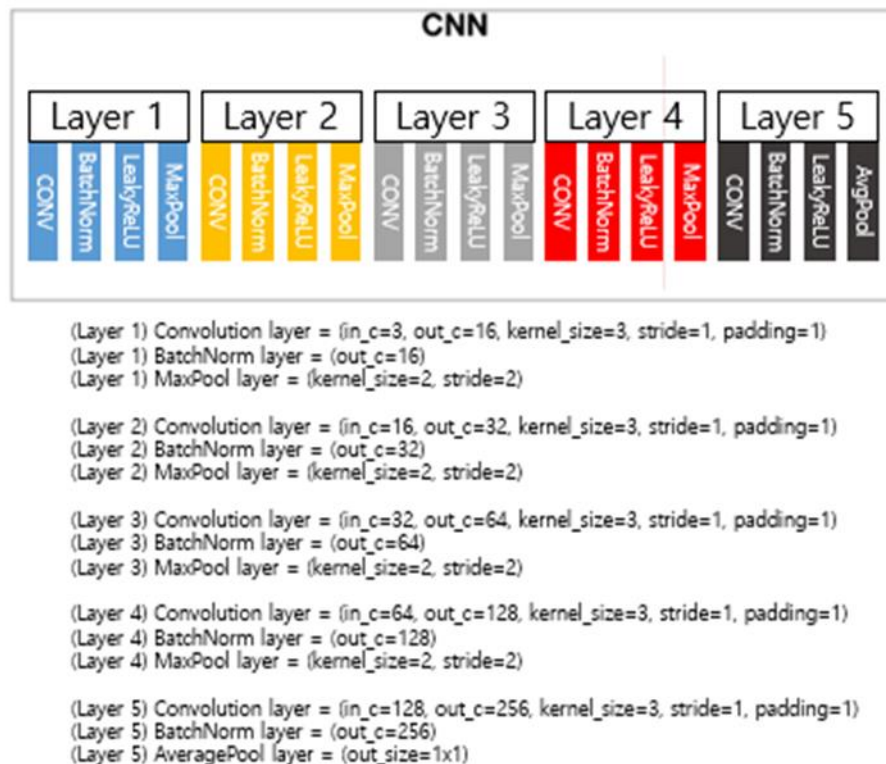
2.1.3 이미지 분류

Image Classification이란 딥러닝을 통해 시각적 내용에 따라 이미지를 분류하는 것을 말한다. 딥러닝 신경망의 한 종류인 CNN(Convolutional Neural Network)을 이용하며 이미지를 입력으로 받고 그에 대한 class 값과 입력 이미지가 특정 class라는 것에 대한 확률 값을 출력한다. CNN은 크게 input layer, output layer, hidden layers로 나눌 수 있고, hidden layers는 보통 convolution 연산을 입력 이미지에 적용 후 다음 층으로 전달하는 Convolutional layers, CNN의 출력을 결정하는 ReLU layers, 뉴런 군집의 출력을 단일 뉴런으로 결합하여 데이터의 차수를 감소시키는 Pooling layers, 이전 층의 모든 노드를 다음

	국민대학교 컴퓨터공학부 캡스톤 디자인 I	중간보고서	
		프로젝트 명	동영상 연령제한 필터링
		팀 명	YouHi
		Confidential Restricted	Version 1.5 2020-APR-22

층의 모든 노드에 연결하는 Fully connected layer으로 구성된다.

이번 프로젝트에서 원래 Object Detection만을 이용하여 게임 장면에서 등장하는 혈흔과 여성의 과도한 노출을 검열하려 하였으나, 빨간 옷을 입고 있는 게임 캐릭터 그리고 살색과 비슷한 옷을 입은 여성도 검열되는 문제를 확인하였다. 이러한 문제를 해결하기 위해서는 각 class마다 특징점을 찾아 이미지를 분류하는 Image Classification이 적합하다고 판단했고, 따라서 Object Detection 검열 이후 해당 프레임에 대한 Image Classification 검열 과정을 추가하여 혈흔과 빨간 색의 옷, 나체와 살색의 옷의 구분이 가능하도록 했다. 다음은 우리 조가 구축한, Image Classification에 사용되는 CNN모델의 구조이다.



이번 프로젝트에서 구축된 CNN Layer에는 앞서 설명하지 않은 BatchNorm, LeakyReLU, AvgPool 등의 기법이 적용되어 그 정확도를 높였다.

2.1.4 영상 검열

영상 검열은 Two Stream Convolution Network 모델로 진행한다.

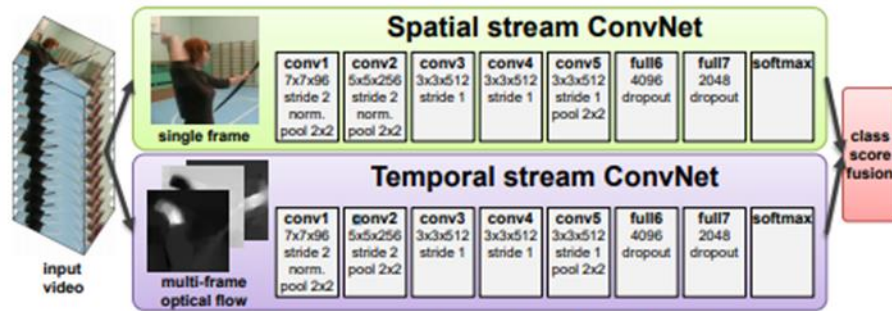
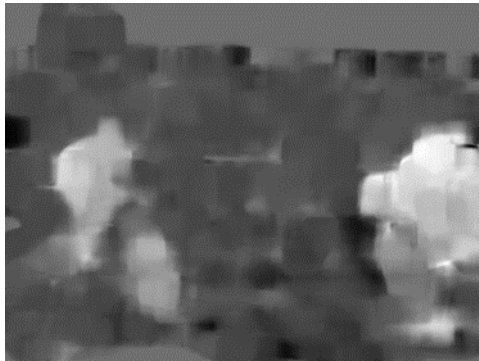


Figure 1: Two-stream architecture for video classification.

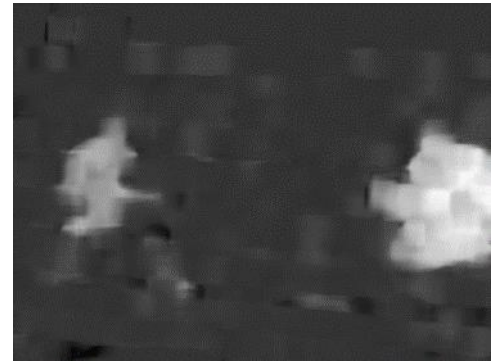
위 그림처럼 Spatial Stream ConvNet과 Temporal Stream ConvNet으로 각각 예측된 결과를 Average, Conv Fusion을 통해 결합한다. 먼저 Spatial Stream ConvNet는, 하나의 RGB 프레임에서 작동 되며 해당 프레임에서 사람의 Action을 인식한다. 즉, 정적 이미지에서 결과를 예측하는 것이다. 반면 Temporal Stream ConvNet은 일정 길이의 이미지(프레임)을 취합한 내용을 통해 예측하는 것이다. Optical Flow ConvNet이라고도 불리는데, 그 이유는 RGB 이미지가 아닌 Optical Flow로 표현된 이미지로 학습과 예측을 진행하기 때문이다. Optical Flow는 Vertical Flow와 Horizontal Flow로 구성되어있다. 따라서 학습과 예측을 진행하기 전에 RGB이미지를 Vertical Flow, Horizontal Flow로 표현해야한다. Vertical Flow, Horizontal Flow은 다음과 같이 표현된다.



<Original Image>



<Horizontal>

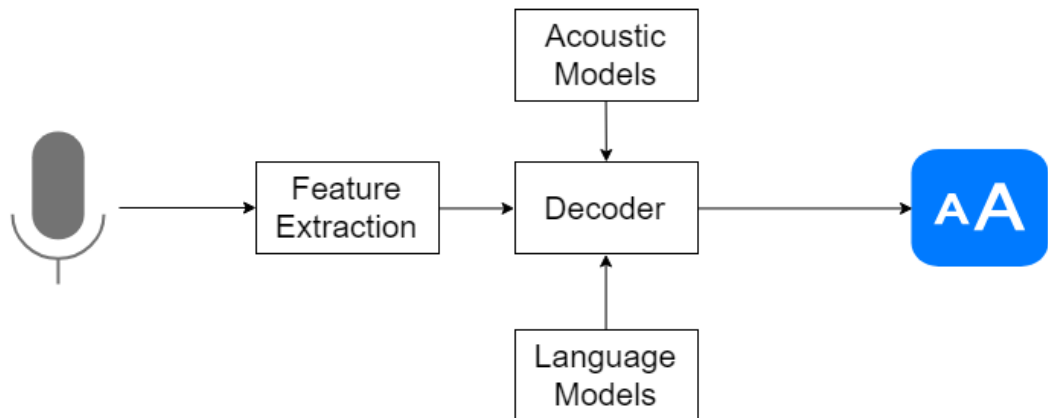


<Vertical>

이처럼 표현된 연속된 이미지들에서 행동을 추적하고 어떤 행동인지 예측한다. 그러나 이 두 가지 모델은 각각 단점이 있는데, Spatial의 경우 연속된 이미지를 고려하지 않다보니 단순 이미지 Classification의 결과와 크게 다르지 않다라는 것이다. 또한 Temporal의 경우 유사한 행동 패턴이 들어왔을 때 예측이 틀린 경우가 발생할 가능성이 높다. 따라서 Temporal과 Spatial을 Fusion을 진행하면 각각의 단점을 상호보완 해주기 때문에, 높은 정확도를 얻을 수 있다.

2.1.5 음성 인식

딥러닝 기반의 음성인식 기술은 크게 언어모델과 음향모델이라는 두 가지 중요한 지식원 (Knowledge source)을 사용해 음성 신호로부터 문자 정보를 출력한다. 언어모델은 단어 시퀀스에 확률을 할당(assign) 하는 일을 하는 모델이다. 이는 가장 자연스러운 단어 시퀀스를 찾는 역할을 한다. 음향모델은 언어의 소리단위를 딥러닝을 통해 학습하여 어떤 단어가 어떤 소리로 나는지를 확률적으로 변환한다. 이 두가지 모델이 동시에 작용하여 높은 확률을 보인 단어를 출력한다.



STT(Speech To Text) 는 잡음처리와 특징 추출과정을 거친 음성 데이터가 언어모델과 음향 모델이 결합된 디코더를 통과한 후 문장 형태로 출력되면 이를 텍스트로 저장한다.

2.1.6 형태소 분석

형태소 분석기는 텍스트 형태의 데이터를 형태소 단위로 분리한다. 형태소 단위로 분리된 데이터는 비속어 여부를 판단하기에 유리하다.

카카오에서 개발한 세 번째 형태소 분석기인 khaiii는 형태소 분석 시 입력된 각 음절에 대해 하나의 출력 태그를 결정하는 분류 문제로 접근하게 된다. 일정 텍스트의 형태소 분석 결과는 다음 이미지와 같이 생성된다.

```

kmsuw@kmsuw-ThinkPad-T440: ~/바탕화면/khaiii/build
kmsuw@kmsuw-ThinkPad-T440:~/바탕화면/khaiii/build$ khaiii --input input.txt
[2020-04-03 18:03:30.567] [Resource] [info] NN model loaded
[2020-04-03 18:03:30.567] [Preanal] [info] preanal dictionary opened
[2020-04-03 18:03:30.567] [ErrPatch] [info] errpatch dictionary opened
[2020-04-03 18:03:30.567] [Restore] [info] restore dictionary opened
[2020-04-03 18:03:30.567] [Resource] [info] PoS tagger opened
좋았던 좋 /VA + 았 /EP + 던 /ETM
기억만 기억 /NNG + 만 /JX
그리운 그리 /VA + ㄴ /ETM
마음만 마음 /NNG + 만 /JX
나가 나 /NP + 가 /JKS
떠나간 떠나 /VV + ㄴ /ETM
그 그 /MM
길 길 /NNG
위에 위 /NNG + 예 /JKB
이렇게 이 형 /VA + 게 /EC
남아 남 /VV + 아 /EC
서있다 서 /VV + 어 /EC + 있 /VX + 다 /EC
잊혀질 잊히 /VV + 어 /EC + 지 /VX + ㄴ /ETM
만큼만 만큼 /NNB + 만 /JX
괜찮을 괜찮 /VA + 을 /ETM
만큼만 만큼 /NNB + 만 /JX
눈물 눈물 /NNG
머금고 머금 /VV + 고 /EC
기다린 기다리 /VV + ㄴ /ETM
떨림 떨 /NNG + 리 /VV + ㄴ /ETN
끝 끝 /NNG + 예 /JKB
다시 다시 /MAG
나들 나 /NP + 들 /JKO
피우리라 피우 /VV + 리라 /EC
  
```



khaiii는 기계학습 기반의 알고리즘을 이용하여 형태소를 분석한다. 형태소 분석은 자연어 처리를 위한 가장 기본적인 전처리 과정이기 때문에 빠른 시간 내에 이루어져야 하므로 신경망 알고리즘들 중 빠른 속도로 진행되는 Convolutional Neural Network(CNN)을 사용하였다.

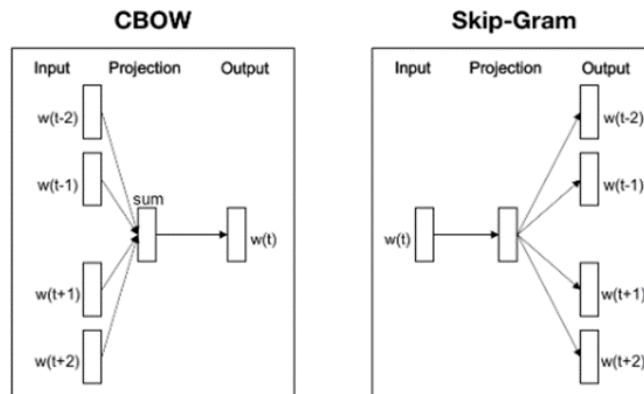
khaiii는 신경망 알고리즘의 앞단에 기분석 사전, 뒷단에 오분석 패치라는 두 가지 사용자 사전 장치를 제공한다. 기분석 사전은 단일 어절에 대해 문맥에 상관없이 일괄적인 분석 결과를 갖게 하고 싶을 경우 사용하고 오분석 패치는 여러 어절에 걸쳐 충분한 문맥과 함께 오분석을 바로잡아야 할 경우에 활용한다. 이러한 두 기능으로 형태소 오분석 확률을 줄일 수 있고 본 프로젝트에 맞게 형태소 분석을 진행할 수 있다.

2.1.7 욕설 검열

khaiii 형태소 분석기로 추출된 형태소들을 사전 훈련된 FastText 모델을 이용하여 욕설 데이터 리스트의 원소와 cosine similarity를 계산한다.

FastText는 단어를 벡터로 만드는 Word2Vec과 비슷한 매커니즘을 가진다. 하지만 Word2Vec은 어휘를 최소단위로 보는 반면에 FastText는 텍스트의 최소 단위를 어휘를 구성하는 글자 n-gram으로 하였다. 즉 Word2Vec은 임베딩 벡터를 어휘마다 하나씩 할당하고 이를 학습했다면, FastText는 어휘를 구성하고 있는 n-gram마다 하나씩 이를 할당하고, 어휘를 구성하는 모든 n-gram 벡터의 평균 벡터를 어휘 임베딩으로 보게 된다. 이러한 방법은 동일한 텍스트 데이터에서 더 많은 정보를 활용하기 때문에, 더 적은 양의 학습 데이터로도 높은 성능을 낼 수 있다.

FastText는 corpus 안에 존재하는 모든 단어를 지정해 놓은 윈도우 크기로 슬라이딩 하여 학습을 진행한다. 그리고 임베딩 기법에 따라 주변 단어를 보고 중심 단어가 무엇인지를 예측하는 CBOW모델과 중심 단어를 보고 어떤 주변 단어가 등장했는지를 맞추는 Skip-Gram 모델로 나뉜다.



<CBOW와 Skip-Gram>


만약 윈도우의 크기를 2라고 가정하였을 때, CBOW 모델의 경우 중심 단어의 벡터는 주변 단어로부터 단 한번의 업데이트 기회를 갖게 된다. 하지만 Skip-gram 모델의 경우 중심 단어 벡터를 주변 단어 개수만큼, 즉 네 번을 업데이트할 수 있게 한다. 따라서 같은 corpus 크기를 갖더라도 학습량이 네 배 차이가 나게 되어 CBOW 모델에 비해 Skip-Gram 모델이 더 높은 성능을 보인다. 따라서 본 프로젝트에서는 FastText Skip-Gram 모델 방식을 채택하였다.

FastText 모델 학습 시 dictionary에 관한 주요 argument는 다음과 같다.

- 1) minCount : 주어진 값 이상으로 등장한 단어들만 임베딩을 실시할 수 있다.
- 2) wordNgrams : 위에서 말한 n-grams의 의미가 아닌 중심 어휘와 주변 어휘를 몇 개의 단어로 설정할 지에 대한 옵션이다.
- 3) bucket : 모델의 메모리 사용을 제한하기 위해 설정하는 n-grams이 hash된 bucket의 수이다.

FastText 모델 학습의 hyper parameter와 관련된 주요 argument는 다음과 같다.

- 1) lr : FastText 모델 학습 알고리즘의 learning rate를 설정한다. 0.1 ~ 1.0 사이의 값을 갖는 것이 좋다고 알려져 있다.
- 2) dim : 임베딩되는 단어 벡터들의 차원 수를 설정할 수 있다. dim이 큰 값을 가질수록,

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22

벡터들은 더 많은 정보를 가질 수 있지만 그만큼 더 많은 데이터를 학습해야 한다. 하지만 이 값이 너무 크게 되면, 학습 과정이 어려워지고 시간이 오래 걸린다. default 값으로 100을 가지지만 100부터 300사이의 값을 많이 이용한다.

3) ws : FastText 모델 학습 시 사용되는 window size의 크기를 설정할 수 있다.

4) epoch : FastText 모델 학습 시 수행하는 epoch 개수를 설정 가능하다.

5) loss : FastText 모델 학습 시 사용되는 loss function을 설정할 수 있다. 설정할수 있는 값으로 negative sampling을 뜻하는 ns, softmax의 softmax, hierarchical softmax을 뜻하는 hs가 있다.

FastText의 결과인 어휘 벡터들은 서로 의미가 유사할수록 그들이 갖는 방향 또한 같아진다. 이러한 점을 이용하여 욕설 검열 과정에서 형태소와 욕설의 유사도를 판단하기 위해 내적 공간의 두 벡터간 각도의 cosine값을 이용하여 측정된 벡터 간의 유사한 정도를 의미하는 cosine similarity를 이용하기로 했다.


2.1.8 Web Front End

React는 페이스북에서 개발한 유저인터페이스 라이브러리로 개발에게 재사용 가능한 UI를 생성 할 수 있도록 해준다. 이 라이브러리는 현재 페이스북, 인스타그램, 야후, 넷플릭스를 포함한 많은 서비스에서 사용되고 있다.

Web Page 특성상 기능별 컴포넌트들이 필요하고, 컴포넌트끼리의 연결도 용이해야한다. React Js는 컴포넌트 별 관리가 편하고, 디버깅 및 코드 수정이 비교적 간단하기 때문에 React Js로 구현을 진행한다.

React Js의 Rendering은 기존 HTML 문법과 상당히 유사하기 때문에, 많은 개발자에게 익숙하여 협업을 진행하기에 적합하다.

웹 구현에 있어 가장 중요한 것은 기능과 디자인이다. react는 다양한 라이브러리를 제공하는데 개발자는 이 라이브러리들을 쉽게 다운받아 쓸 수 있고 자체 커스텀하기도 편리하기 때문에 react js로 구현을 진행한다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22

2.1.9 Back End

딥러닝 모델 학습과 모델의 영상, 음성 검열 과정에 필요한 Amazon EC2 instance, Amazon S3, Amazon Lambda, Amazon Gateway API를 생성한다. 웹 페이지의 배포를 위해 웹서버를 구축한다. 웹서버, 웹페이지, AWS의 원활한 상호작용을 위해 Socket 통신을 활용한다.

동영상이 업로드 되는 공간, 업로드 된 동영상의 프레임 추출, 딥러닝 모델에 넣는 작업을 위한 Amazon S3, Amazon Lambda, Amazon Gateway API를 이용한다. 단순 S3에 업로드하는 방식을 채택할 경우, IAM 사용자의 Access Key 및 Secret Key가 노출될 가능성이 높으므로 Gateway Api와 Lambda를 통해 업로드를 진행한다. S3에 업로드가 완료되면 Lambda 함수에서는 프레임을 추출하고, 전처리를 진행하는 코드를 실행한다.

AWS EC2 instance는 AWS Deep Learning AMI를 채택하여 딥러닝 모델의 학습을 진행한다. 모델 학습에 있어 필요한 데이터셋은 Amazon S3에 저장한다. Instance내에 구축된 Apache 서버를 통해 웹페이지를 배포한다.

웹페이지 이용자와 AWS를 통해 얻어진 데이터의 주체가 동일인물인지 파악하고 Back End에서 일처리(검열 작업)의 종료를 웹페이지에 알려주기 위해 Socket 서버를 구축하여 통신한다. Socket은 SocketIo 라이브러리를 활용한다.

2.2 수행내용

2.2.1 Instance Segmentation

Knife, Smoke, Gun, Adult를 각각 Labelme 툴을 이용해 Annotation을 제작하여 데이터셋을 구축하였다. 각 라벨당 70장의 이미지가 사용되었으며, Annotation은 이미지의 이름과 동일한 이름을 가지는 json 파일로 생성된다.

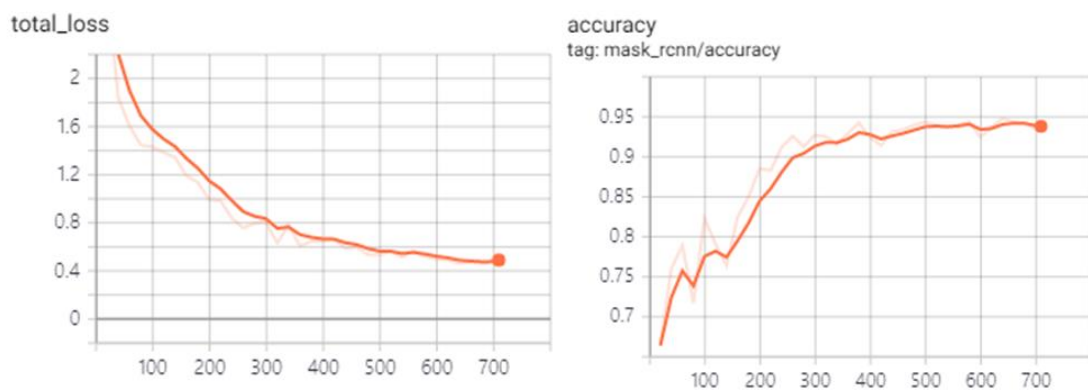


<Knife, Gun, Smoke Annotation>


칼과 같은 경우 담배와 비슷하게 인식하여 오분류되는 결과가 많이 발생하여 손을 포함시켜서 하나의 Segmentation으로 구성하였다.

모델은 Mask_RCNN_FPN_101_3x 모델을 사용하였고, Learning Rate는 0.0025, Iteration은 710으로 설정하여 학습을 진행하였다. 예측값이 최소 0.5이상(threshold) 결과만 출력시키며 가장 높은 값을 보이는 것을 해당 이미지에서의 라벨로 선정하였다.

아래 화면은 설정한 Hyper Parameter로 학습을 진행한 화면이다.



<Loss, Accuracy>

	국민대학교 컴퓨터공학부 캡스톤 디자인 I	중간보고서	
		프로젝트 명	동영상 연령제한 필터링
		팀 명	YouHi
		Confidential Restricted	Version 1.5 2020-APR-22

```

[05/21 19:45:26 d2.utils.events]: eta: 0:04:14 iter: 379 total_loss: 0.645
[05/21 19:45:42 d2.utils.events]: eta: 0:03:59 iter: 399 total_loss: 0.647
[05/21 19:45:57 d2.utils.events]: eta: 0:03:44 iter: 419 total_loss: 0.660
[05/21 19:46:13 d2.utils.events]: eta: 0:03:29 iter: 439 total_loss: 0.589
[05/21 19:46:29 d2.utils.events]: eta: 0:03:14 iter: 459 total_loss: 0.595
[05/21 19:46:44 d2.utils.events]: eta: 0:02:59 iter: 479 total_loss: 0.535
[05/21 19:47:00 d2.utils.events]: eta: 0:02:43 iter: 499 total_loss: 0.533
[05/21 19:47:16 d2.utils.events]: eta: 0:02:28 iter: 519 total_loss: 0.562
[05/21 19:47:31 d2.utils.events]: eta: 0:02:12 iter: 539 total_loss: 0.516
[05/21 19:47:47 d2.utils.events]: eta: 0:01:57 iter: 559 total_loss: 0.568
[05/21 19:48:02 d2.utils.events]: eta: 0:01:41 iter: 579 total_loss: 0.520
[05/21 19:48:18 d2.utils.events]: eta: 0:01:26 iter: 599 total_loss: 0.495
[05/21 19:48:33 d2.utils.events]: eta: 0:01:10 iter: 619 total_loss: 0.490
[05/21 19:48:49 d2.utils.events]: eta: 0:00:55 iter: 639 total_loss: 0.460
[05/21 19:49:05 d2.utils.events]: eta: 0:00:39 iter: 659 total_loss: 0.471
[05/21 19:49:20 d2.utils.events]: eta: 0:00:24 iter: 679 total_loss: 0.466
[05/21 19:49:36 d2.utils.events]: eta: 0:00:08 iter: 699 total_loss: 0.479
[05/21 19:49:45 d2.utils.events]: eta: 0:00:00 iter: 709 total_loss: 0.510

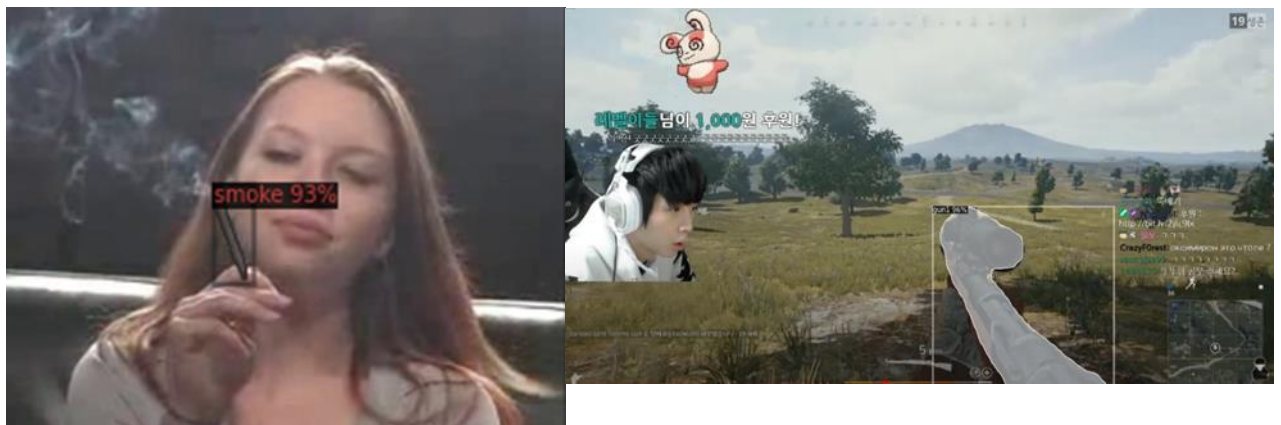
```

<Learning Progress>


다음은 학습이 끝난 뒤 테스트 데이터셋으로 예측을 진행한 결과이다.



<Knife 예측 결과>

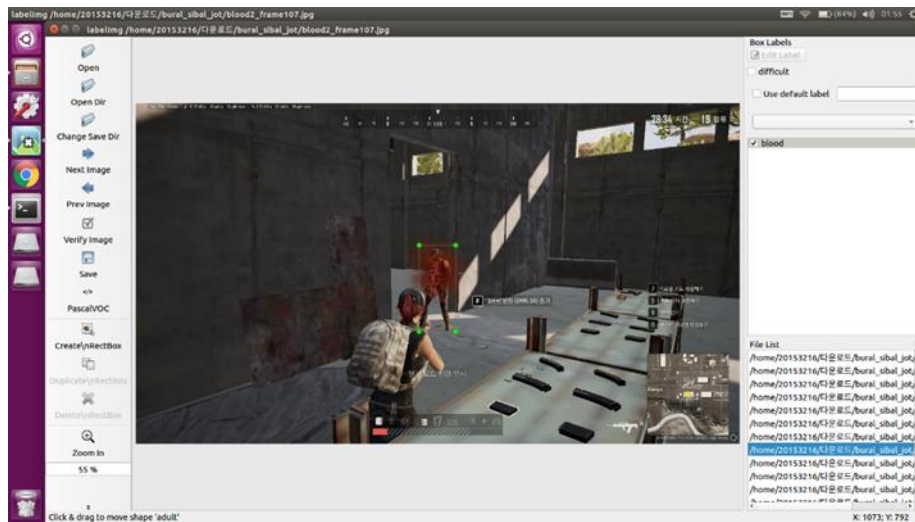


<Smoke, Gun 예측 결과>

	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22

2.2.2 Object Detection

Object Detection은 Blood 라벨에 대해서만 Detect를 진행한다. 즉 FPS 게임에서 총에 맞아 피를 흘리고 있는 사람을 검출한다. 이를 위해 Labellmg 툴을 이용해 Annotation을 진행하여 약 200장의 게임 데이터셋을 구축하였고, Faster-RCNN을 이용해 학습을 진행하였다.

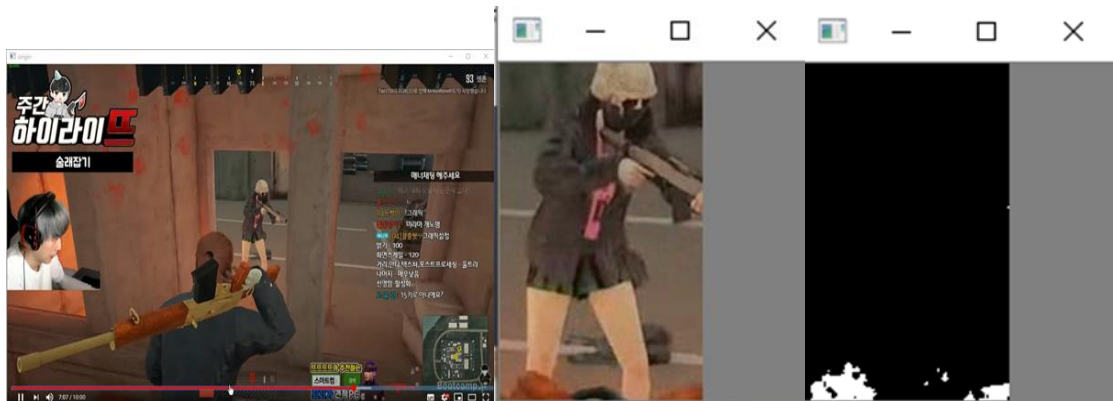


<Labellmg를 통한 데이터셋 구축>



<예측 결과>

그러나 위 사진과 같이 피를 흘리고 있는 신체와 일반 신체를 전혀 구별하지 못해 해당 객체만큼 ROI를 설정하고, 빨간색의 픽셀 비율을 계산한 이후 조건에 맞는 객체들에 한해서 Image Classification을 수행한다.



<ROI 추출과 빨간색 픽셀 계산>

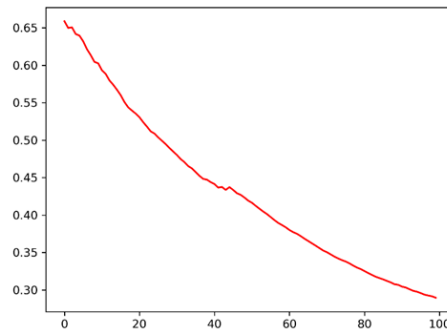
2.2.3 이미지 분류

Adult의 경우 옷을 입고있는 것과 그렇지 않은 것에 대한 구별을 정확하게 하지 못하고, Blood의 경우 빨간 옷과 피에 대한 구별을 제대로 하지 못해 이미지 분류 학습을 진행하고 각 라벨에 대한 예측값을 출력한다. Adult와 Blood의 경우 동일한 모델을 사용했지만, Epoch와 Learning Rate, Batch Size를 다르게 설정하여 보다 정확도를 높였다. Adult의 경우 100번의 Epoch 중 88번째 Epoch가 가장 뛰어난 성능을 보였으며, Batch Size는 8, Learning Rate는 0.0005로 설정하였다.

```
hyper_param_epoch = 100
hyper_param_batch = 8
hyper_param_learning_rate = 0.0005
```

<Hyper Paramter 설정>

```
Epoch [82/100], Loss: 0.1067
Epoch [83/100], Loss: 0.1153
Epoch [84/100], Loss: 0.1165
Epoch [85/100], Loss: 0.1670
Epoch [86/100], Loss: 0.1632
Epoch [87/100], Loss: 0.1376
Epoch [88/100], Loss: 0.1408
Test Accuracy of the model on the 402 test images: 95.5223880597015 %
```

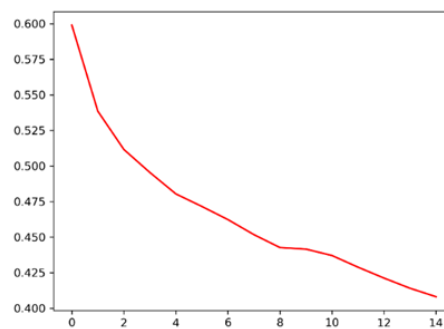
< 학습에 따른 Loss 변화 과정 >

Blood의 경우 15번의 Epoch 중 10번째 Epoch가 가장 뛰어난 성능을 보였으며, Batch Size는 10, Learning Rate는 0.001로 설정하였다.


```
hyper_param_epoch = 15
hyper_param_batch = 10
hyper_param_learning_rate = 0.001
```

<Hyper Parameter 설정>

```
Epoch [10/15], Loss: 0.4101
Epoch [10/15], Loss: 0.3596
Epoch [10/15], Loss: 0.3714
Epoch [10/15], Loss: 0.5857
Test Accuracy of the model on the 296 test images: 100.0 %
```



< 학습에 따른 Loss 변화 과정 >

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22

2.2.4 영상 검열

UCF101 데이터셋에서 23개의 라벨을 가져와 Optical Flow로 변환을 실시했다. 기존에 다른 개발자가 Optical Flow로 변환해놓은 데이터셋이 존재했지만, 성능이 매우 떨어지게 구축되어있어서 직접 변환하였다. 또한 기존 담배피는 행동 데이터셋으로 학습을 진행하였을 때, 사탕이나 빨대 등 유사한 행위를 담배피는 장면으로 오분류하는 결과가 나타나 데이터셋을 수정하였다. 담배를 피며 연기가 내뿜어지는 장면으로 구성하였다.




<원본 이미지>



<Horizontal>



<Vertical>

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22

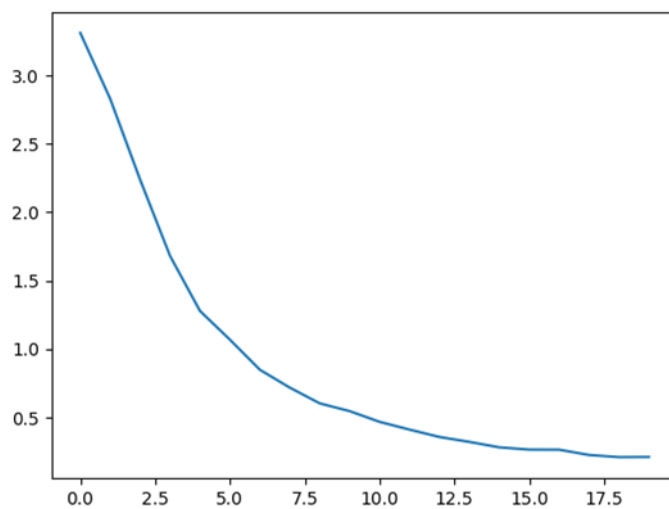
Spatial Stream Model의 경우 20 Epoch로 학습을 진행하였다.

```

Current learning rate is 0.001000:
Epoch: [19][5/128]      Time 6.512 (6.512)      Loss 0.2326 (0.2326)      Prec@1 92.000 (92.000)
Epoch: [19][10/128]     Time 6.536 (6.524)      Loss 0.2233 (0.2280)      Prec@1 92.800 (92.400)
Epoch: [19][15/128]     Time 6.537 (6.528)      Loss 0.2436 (0.2332)      Prec@1 92.000 (92.267)
Epoch: [19][20/128]     Time 6.540 (6.531)      Loss 0.1857 (0.2213)      Prec@1 95.200 (93.000)
Epoch: [19][25/128]     Time 6.541 (6.533)      Loss 0.1613 (0.2093)      Prec@1 96.800 (93.760)
Epoch: [19][30/128]     Time 6.535 (6.534)      Loss 0.1733 (0.2033)      Prec@1 96.000 (94.133)
Epoch: [19][35/128]     Time 6.535 (6.534)      Loss 0.1812 (0.2001)      Prec@1 96.000 (94.400)
Epoch: [19][40/128]     Time 6.535 (6.534)      Loss 0.2419 (0.2054)      Prec@1 94.400 (94.400)
Epoch: [19][45/128]     Time 6.526 (6.533)      Loss 0.1504 (0.1993)      Prec@1 97.600 (94.756)
Epoch: [19][50/128]     Time 6.534 (6.533)      Loss 0.3391 (0.2132)      Prec@1 94.400 (94.720)
Epoch: [19][55/128]     Time 6.539 (6.534)      Loss 0.2330 (0.2150)      Prec@1 95.200 (94.764)
Epoch: [19][60/128]     Time 6.534 (6.534)      Loss 0.1654 (0.2109)      Prec@1 96.000 (94.867)
Epoch: [19][65/128]     Time 6.539 (6.534)      Loss 0.2241 (0.2119)      Prec@1 93.600 (94.769)
Epoch: [19][70/128]     Time 6.542 (6.535)      Loss 0.2418 (0.2140)      Prec@1 94.400 (94.743)
Epoch: [19][75/128]     Time 6.540 (6.535)      Loss 0.1988 (0.2130)      Prec@1 95.200 (94.773)
Epoch: [19][80/128]     Time 6.565 (6.537)      Loss 0.1938 (0.2118)      Prec@1 94.400 (94.750)
Epoch: [19][85/128]     Time 6.532 (6.537)      Loss 0.1767 (0.2098)      Prec@1 97.600 (94.918)
Epoch: [19][90/128]     Time 6.557 (6.538)      Loss 0.1833 (0.2083)      Prec@1 96.000 (94.978)
Epoch: [19][95/128]     Time 6.551 (6.538)      Loss 0.2134 (0.2086)      Prec@1 95.200 (94.989)

```

<Spatial Stream Model 학습>



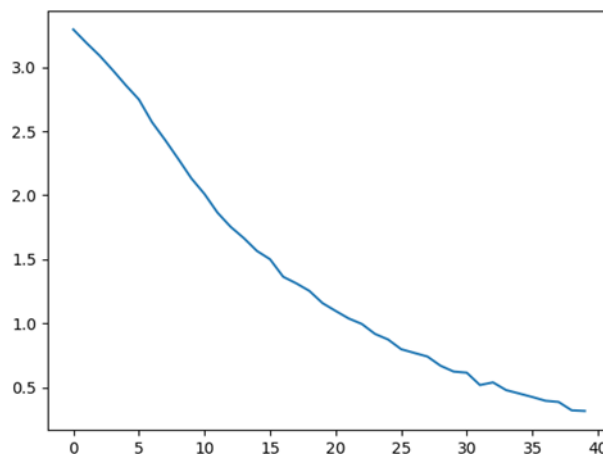
< 학습에 따른 Loss 변화 과정 >

Temporal Stream Model의 경우 40 Epoch로 학습을 진행하였다.

```

Current learning rate is 0.001000:
Epoch: [39][5/128]      Time 8.558 (8.558)      Loss 0.2884 (0.2884)      Prec@1 92.000 (92.000)
Epoch: [39][10/128]     Time 8.563 (8.561)      Loss 0.2803 (0.2844)      Prec@1 93.600 (92.800)
Epoch: [39][15/128]     Time 8.580 (8.567)      Loss 0.2825 (0.2837)      Prec@1 90.400 (92.000)
Epoch: [39][20/128]     Time 8.584 (8.571)      Loss 0.4543 (0.3264)      Prec@1 87.200 (90.800)
Epoch: [39][25/128]     Time 8.546 (8.566)      Loss 0.2392 (0.3089)      Prec@1 92.800 (91.200)
Epoch: [39][30/128]     Time 8.542 (8.562)      Loss 0.3669 (0.3186)      Prec@1 88.800 (90.800)
Epoch: [39][35/128]     Time 8.534 (8.558)      Loss 0.2582 (0.3100)      Prec@1 90.400 (90.743)
Epoch: [39][40/128]     Time 8.533 (8.555)      Loss 0.4001 (0.3212)      Prec@1 88.000 (90.400)
Epoch: [39][45/128]     Time 8.565 (8.556)      Loss 0.3272 (0.3219)      Prec@1 92.000 (90.578)
Epoch: [39][50/128]     Time 8.580 (8.558)      Loss 0.2670 (0.3164)      Prec@1 91.200 (90.640)
Epoch: [39][55/128]     Time 8.598 (8.562)      Loss 0.4145 (0.3253)      Prec@1 86.000 (90.578)
  
```

<Temporal Steam Model 학습>



<학습에 따른 Loss 변화 과정>

마지막으로 Spatial Stream Model과 Temporal Stream Model의 예측결과를 Average Fusion 하여 최종 결과를 출력한다. 현재 Spatial Stream Model과 Temporal Stream Model의 평균값을 계산하는 과정에서 동일한 가중으로 계산한다. 아래 사진처럼 두 모델의 출력값이 하나 차이나는 것을 확인할 수 있는데, 이는 Spatial Stream과 Temporal Stream Model이 결과를 예측할 때 서로 다른 숫자의 프레임을 확인하기 때문이다.



```
1 smoke4.mp4
2 0.912588
3 0.980637
4 0.993374
5 0.99166
6 0.996229
7 0.993992
8 0.986162
9 ----
10
```

```
1 smoke4.mp4
2 0.208731
3 0.391649
4 0.16849
5 0.98884
6 0.930769
7 0.981653
8 ----
9
```

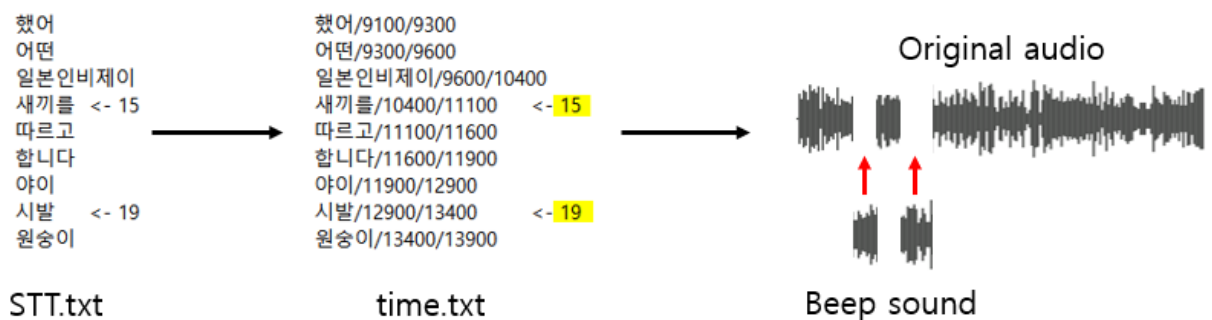
```
1 smoke4.mp4
2 0.9902500000000001
3 0.963499
4 0.9878225
5
```

<Spatial, Temporal Stream Model 출력>

<최종 출력 화면>

2.2.5 음성 마스킹

검열된 욕설은 STT 처리된 원본 텍스트 파일과 비교하여 몇 번째 라인에 위치해 있는지 확인 후 해당 위치를 배열에 저장한다. 이 후, 저장해둔 발화시간이 담겨있는 텍스트 파일을 통해 욕설의 발화 시작, 종료 시간을 추출한 뒤 원본 음성 파일 중 해당 부분을 묵음 처리 한다. 묵음 처리 과정에는 '빠-' 소리의 경고음 파일을 사용하며 욕설 단어가 발화된 부분을 잘라낸 뒤 해당 부분에 경고음 파일을 붙인다.



<음성 마스킹>

2.2.6 자막 욕설 검열

자막 추출은 영상을 프레임 단위로 나눈 사진에 Google Vision API를 적용시켜 이루어진다. 해당 API를 통해 자막을 텍스트로 변환한 후 자연어 부분과 이미지 내의 해당 단어 좌표를 구분지어 저장한다.



오늘은 축구 역사상 기1장 화려한 플레이를 보여준 선수,
오늘은 축구 역사상 1장 화려한 플레이를 보여준 선수,
호나우지뉴에 대한 야기)입니다.
그럼 재미있게 봐주요!
VIN |그럼 재미있게 봐주요!
맴피셜
맴훈이!
SEN EDIN AHK
맴훈이의 뇌피셜 풋들 N ES

<텍스트로 변환된 자막 파일>

오늘은 축구 역사상 기1장 화려한 플레이를 보여준 선수, +(68,228) (274,228)
(91,230) (91,244) (84,244)/축구+(95,229) (109,229) (109,244) (95,244)/역사
(143,243)/1+(148,230) (149,230) (149,243) (148,243)/장+(151,229) (158,229)
(192,230) (216,230) (216,244) (192,244)/를+(219,230) (225,230) (225,244) (:
(271,228) (271,247) (255,247)/,+(273,243) (274,243) (274,245) (273,245)/
오늘은 축구 역사상 1장 화려한 플레이를 보여준 선수, +(68,226) (274,226) (2
(91,244) (84,244)/축구+(95,229) (109,229) (109,244) (95,244)/역사삼+(114,2
(148,230) (158,230) (158,244) (148,244)/화려한+(163,229) (187,229) (187,24
(225,230) (225,244) (219,244)/보여준+(230,230) (253,230) (253,244) (230,24
(273,246)/

<자막의 단어별 좌표를 포함하고 있는 파일>

텍스트 파일은 앞선 음성 필터링과 같은 과정(Khahii를 통한 형태소 분석, FastText를 사용
한 코사인 유사도 계산, String Match 함수를 통해 욕설 감지)을 거친 후 좌표를 포함한 파
일과의 비교를 통해 욕설이 검출된 프레임 이미지의 번호와 해당 욕설의 좌표를 출력한다.
이 후, openCV 를 사용하여 해당 이미지의 욕설 부분에 BoundingBox 처리를 한다.



<Bounding Box 결과>

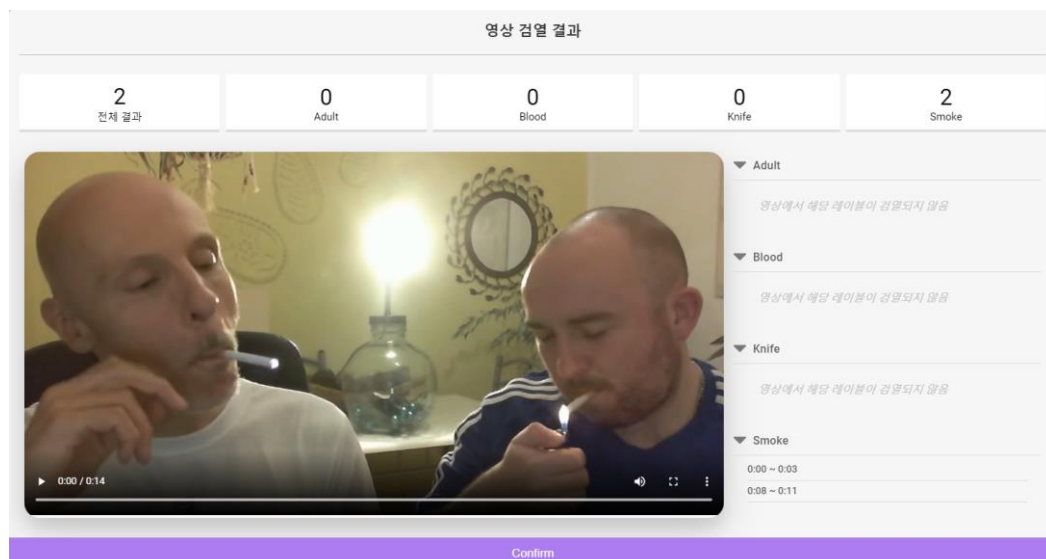
 국민대학교 컴퓨터공학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22

2.2.7 Web Front End


필터링 컴포넌트의 경우 영상 필터링과 음성 필터링 컨테이너를 각각 만들어 이미지를 클릭하면 새로운 팝업창을 띄워 각각의 필터링 결과를 확인할 수 있도록 할 수 있다.



각각의 영상, 음성 필터링 이미지를 클릭 시 새로운 팝업 창이 나오고 서버로부터 받은 필터링 된 결과가 화면에 나타난다. 영상 검열 팝업창의 모습은 다음과 같다.

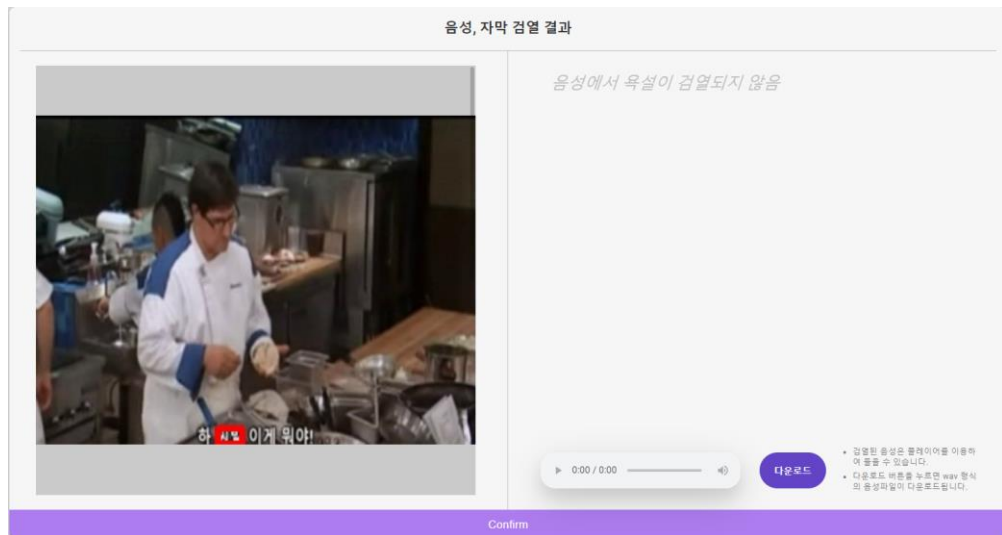


팝업의 제목 아래에는 5개의 박스가 위치하고 있다. 제일 왼쪽에 위치한 박스는 업로드 영상에서 모든 라벨들이 검출된 횟수의 합을, 그리고 나머지 4개의 박스들은 각 라벨마다


	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22

검열된 횟수를 나타내도록 하였다.

그 아래에는 왼쪽에 영상 플레이어가 위치하고 있으며, 클라이언트는 자신이 업로드한 영상을 재생할 수 있다. 오른쪽에는 각 라벨마다 검열된 구간들을 나타내며 drop down을 이용하여 쉽게 확인할 수 있도록 하였다. 남은 프로젝트 기간에는 영상이 재생될 때 특정 라벨이 검열된 구간에 들어오면 위에 위치한 해당 라벨 박스에 CSS 효과를 주어 클라이언트가 영상을 보며 검열 결과를 한 눈에 파악할 수 있도록 할 예정이다.

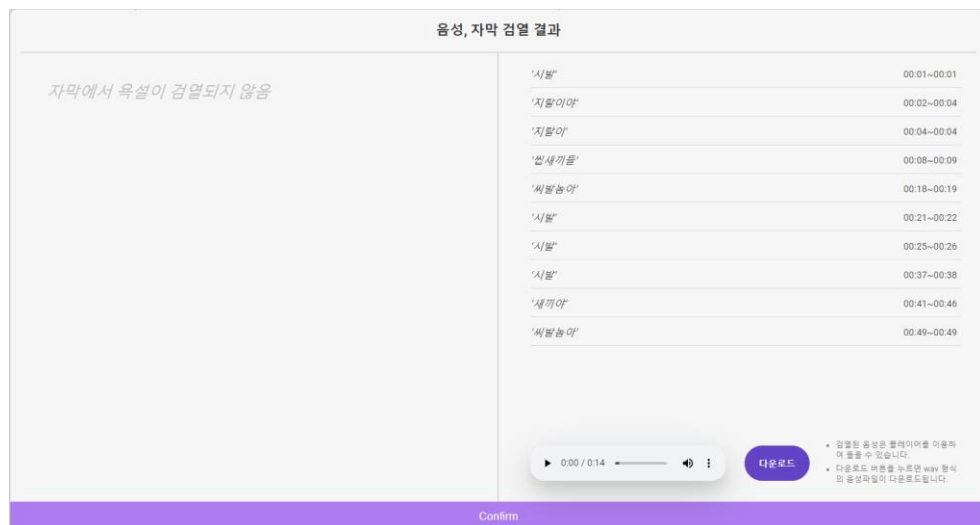


음성, 자막 검열 결과 팝업 창은 위의 이미지와 같다. 위의 경우는 자막에서 욕설이 검열된 것으로, 자막으로 나타난 욕설에 빨간색 박스가 씌워진 것을 확인할 수 있다. 이로써 클라이언트는 업로드한 영상의 자막에서 어느 부분이 부적절했는지 확인할 수 있다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22



위의 이미지와 같이, 자막에서의 검열 결과가 여러 개라면 스크롤 바를 이용하여 확인 가능하도록 하였다.



위 이미지는 음성, 자막 검열 결과 팝업 창에서 음성 검열 결과가 출력된 경우이다. 음성에서 검열된 욕설들이 나열되어 있으며, 각 욕설마다 발화 시간이 오른쪽에 나와있다. 검열 결과가 팝업 창 크기를 넘어갈 시, 스크롤 바를 이용하여 검열된 모든 욕설들을 확인 가능하다. 그리고 음성 검열 결과 아래에는 업로드 영상 음성에서 검열된 욕설들을 목록 처리한 음성 파일을 재생할 수 있는 오디오 플레이어가 있다. 그리고 그 옆에 위치한 다운로드 버튼을 클릭하여 목록 처리가 포함된 음성 파일을 다운로드할 수 있다.

 <div> <p>국민대학교</p> <p>컴퓨터공학부</p> <p>캡스톤 디자인 I</p> </div>	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22

2.2.8 Back End

웹에서 영상을 업로드 시, 다음의 코드가 백엔드에서 진행된다.

```
socket.on('complete', function (data) {
  var id = data.split('.');
  id = id[0];
  console.log(data);
  var a = "";
  a = String(data);
  io.to(clients[id]).emit("upload", "complete");
})
socket.on('filter', function (data) {
  console.log(data)
  var exec = require('child_process').exec,
  ls;
  ls = exec("python3.6 down.py " + data, function (error, stdout, stderr) {
    console.log('stderr: ' + stderr);
    if (error !== null) {
      console.log('exec error: ' + error);
    }
  });
  console.log("download done")
});
```

다음과 같이 구현된 소켓 서버가 EC2 인스턴스에서 구동되고 있고, S3의 업로드 완료 메시지를 수신하고, 웹에서 필터 버튼을 누르면 down.py를 실행한다.

```
os.system("aws s3 cp s3://uploadtest-s3/" + sys.argv[1] + " ./")
os.system("mkdir " + str)
os.system("mkdir ../static " + str)
os.system("mkdir ../static/blood_result")
os.system("aws s3 cp s3://youhi-project/" + str + " ./" + str + " --recursive")
os.system("python3.6 segmentation.py --dir " + sys.argv[1])
```

s3에서 영상과 추출된 프레임 폴더를 다운로드 받고, segmentation.py에 해당 영상이름이 argument로 들어가며 실행된다.

```
sio = socketio.Client()
sio.connect("http://3.34.55.213:1234")
os.system("aws s3 cp " + client_folder + "/blood.txt " + "s3://youhi-project/" + client_folder + "/blood.txt")
sio.emit("complete", args.dir)
sio.sleep(1)
sio.disconnect()
```

총이 검출된 결과가 담긴 txt 파일이 S3로 업로드되며, 해당 라벨에 대한 영상 검열과 음성 검열을 진행하는 EC2 인스턴스로 메시지를 전송한다.

```
os.system("python3.6 flow.py --demo --label smoke --video " +
  client_folder + "/smoke/ & python3.6 rgb.py --demo --label smoke
  --video " + client_folder + "/smoke/")
os.system("python3.6 fusion.py --video " + client_folder + "/
  smoke/ --label smoke")
```

이후 영상 검열을 진행한다.(예: smoke 진행)

	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22

```
os.system("mv " + list + label + "_result.txt ../static/" + client_folder + "/")
```

영상 검열과 음성 검열을 모두 진행한 이후 그 결과값을 웹에 출력하기 위해 그 결과 파일들을 웹서버가 있는 위치에 전송한다

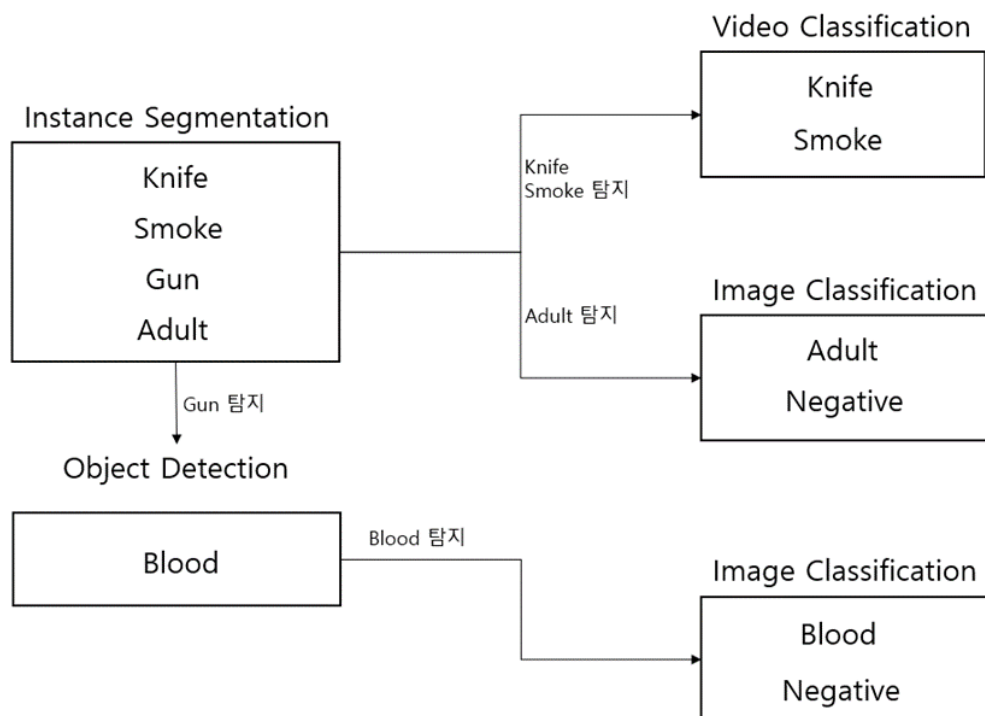
```
ubuntu@ip-172-31-33-143:/var/www/html/static/0$ ls
adult_result.txt  smoke_result.txt
```


3 수정된 연구내용 및 추진 방향

3.1 수정사항

3.1.1 영상 검열 방식 변경

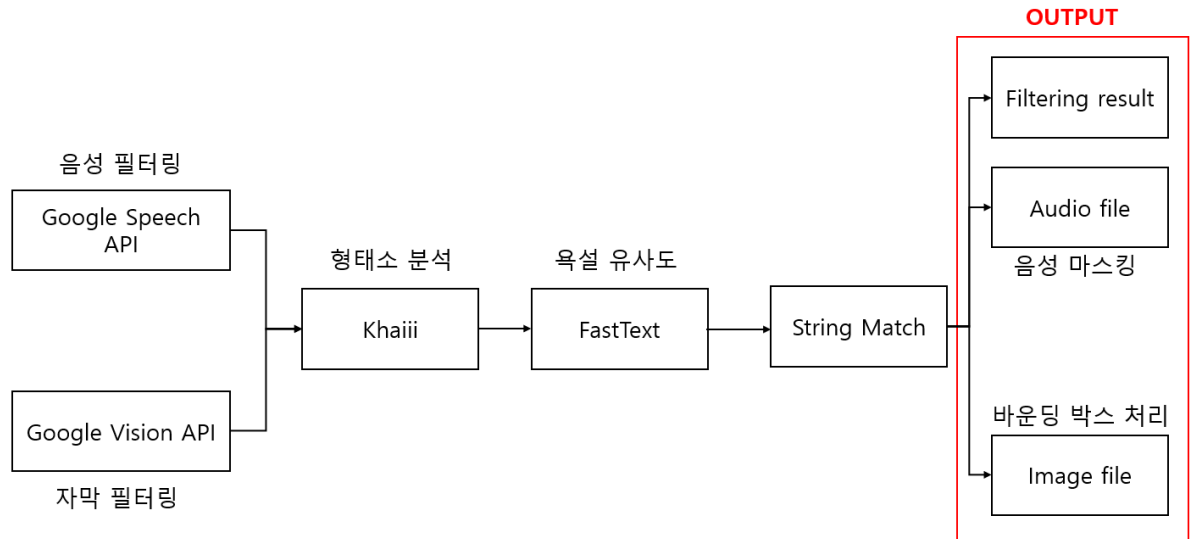
기존 Object Detection -> Video Classification으로 검열 과정을 구축하였는데, Object Detection에서 Smoke와 Knife에 대한 오분류가 너무 많이 발생하여 Instance Segmentation 수행과정을 추가하였다. 다음과 같이 진행된다.



<영상 검열 과정>

위 사진과 같이 기존에는 Knife, Smoke, Gun, Adult를 모두 Object Detection을 이용해서 탐지하고 Video Classification과 Image Classification을 진행할 예정이었지만, Knife와 Smoke 오분류 뿐만 아니라, Adult에서 옷을 입고있는 여자와 그렇지 않은 여자에 대한 오분류도 추가적으로 발견되어 각각 Instance Segmentation과 Image Classification 과정을 추가적으로 진행하여 해결하였다. 또한 Object Detection에서는 총을 맞고 피가 터져 나오는 Object만 탐지한다.

3.1.2 음성 마스킹 및 자막 욕설 검열 기능 추가



기존 음성 검열 방식에서 음성 마스킹 기능이 추가되었다. 구현 내용은 다음과 같다.

Google Speech API를 이용하여 음성에서 텍스트를 추출하고 각 단어마다 추출된 발화 시간을 얻는다. 그 후 기존 음성 검열 방식에 따라 형태소 분석기를 통해 텍스트 전처리를 실시하고 FastText Skipgram Model을 이용한 cosine similarity를 구하여 String Match 진행 여부를 판단한다. 여기서 욕설로 판단되는 단어들은 앞서 구한 발화 시간을 통해 기존 음성 파일의 어느 구간에서 등장했는지 알 수 있기 때문에, 등장한 구간에 마스킹을 씌우도록 한다.

1차 중간 자문 평가 피드백을 고려하여 자막 욕설 검열 기능을 추가하였다. 구현 내용은 다음과 같다.

Google Vision API를 통해 영상에 나타나는 자막, 그리고 자막이 추출된 시간과 위치 정보를 텍스트 파일로 저장한다. 그리고 이후에 일어나는 검열 과정은 String Match 까지 위에서 설명했던 음성 마스킹 기능과 동일하다. String Match가 완료되어 욕설로 판단된 자막들은 발견된 영상의 프레임과 해당 프레임 내 좌표를 이용하여 openCV 바운딩 박스 처리를 진행한다.

4 향후 추진계획

4.1 향후 계획의 세부 내용

4.1.1 Front End

지속적으로 UI/UX를 사용자 편의에 맞게 개선해나간다. 또한 현재 정돈되지 못한 코드를 Refactory 하여 정돈한다.

4.1.2 Back End

여러 사용자가 동시에 영상 업로드 및 검열을 실시할 때 서버가 안정적으로 작동하는지 테스트하여 문제 발생 시, 이를 개선한다.

4.1.3 정확도 측정

본 프로젝트에서 Precision / Recall 매트릭스를 통해 정확도를 측정한다.

- True Positive(TP) : 실제 True인 정답을 True라고 예측 (정답)
- False Positive(FP) : 실제 False인 정답을 True라고 예측 (오답)
- False Negative(FN) : 실제 True인 정답을 False라고 예측 (오답)
- True Negative(TN) : 실제 False인 정답을 False라고 예측 (정답)

		실제 정답		
		True	False	
분류 결과	True	True Positive	False Positive	$(Precision) = \frac{TP}{TP + FP}$ $(Recall) = \frac{TP}{TP + FN}$
	False	False Negative	True Negative	

True는 유해 요소, False는 유해하지 않은 요소로 설정하여 평가한다. 평가 방식은 Precision과 Recall을 이용한다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	중간보고서		
	프로젝트 명	동영상 연령제한 필터링	
	팀 명	YouHi	
	Confidential Restricted	Version 1.5	2020-APR-22

5 고충 및 건의사항

유튜브, 실시간 영상 송출 플랫폼, 한국 방송통신위원회에서는 각자 다른 영상등급 선정 기준을 사용하고 있습니다. 또한 영상 내용이 청소년 관람 불가 내용으로 판단하는 기준이 명확하게 설정되어있지 않기 때문에, 본 프로젝트에서는 각 기관의 선정 기준을 취합하고 각 기관의 모든 기준을 포괄하는 것을 목표로 자체적인 가이드라인을 설정했습니다. 따라서 저희의 가이드라인이 정확하지 않을 수 있지만 저희가 설정한 가이드라인에 충족되는 영상과 그렇지 않은 영상을 정확하게 검열하는 여부에 대해서 우선적으로 판단해주시면 감사하겠습니다.