**캡스톤 디자인 I**

**종합설계 프로젝트**

|  |  |
| --- | --- |
| 프로젝트 명 | 동영상 연령제한 검열 |
| 팀 명 | YouHi |
| 문서 제목 | 중간보고서 |

|  |  |
| --- | --- |
| **Version** | 1.2 |
| **Date** | 2020-04-24 |

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

|  |
| --- |
| **CONFIDENTIALITY/SECURITY WARNING**  이 문서에 포함되어 있는 정보는 국민대학교 전자정보통신대학 컴퓨터공학부 및 컴퓨터공학부 개설 교과목 캡스톤 디자인I 수강 학생 중 프로젝트 “xxxx xxxx”를 수행하는 팀 “xxxxx”의 팀원들의 자산입니다. 국민대학교 컴퓨터공학부 및 팀 “xxxxxx”의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다. |

**문서 정보 / 수정 내역**

|  |  |
| --- | --- |
| **Filename** | 중간보고서-동영상 연령제한 검열.doc |
| **원안작성자** | 이태훈 |
| **수정작업자** | 이인평, 이주형, 김성수, 김민재 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 수정날짜 | 대표수정자 | Revision | 추가/수정 항목 | 내 용 |
| 2020-04-02 | 이태훈 | 1.0 | 최초 작성 |  |
| 2020-04-10 | 김성수 | 1.1 | 내용 추가 | WEB 관련 연구 내용 추가 |
| 2020-04-12 | 이인평 | 1.2 | 내용 추가 | 이미지 분류 관련 내용 추가 |
| 2020-04-14 | 김민재 | 1.3 | 내용 수정 | 오탈자 및 비문 수정 |
| 2020-04-17 | 이주형 | 1.4 | 내용 추가 | 음성 인식 및 STT 내용 추가 |
|  |  |  |  |  |
|  |  |  |  |  |

**본 양식은 컴퓨터공학부 캡스톤 디자인 I 과목의 프로젝트 중간보고서 작성을 위한 기본 양식입니다. 문서의 필수 항목을 제시하는 것이니 폰트, 문단 구조 등의 디자인 부분은 자유롭게 설정하기 바랍니다. 양식 내에 붉은 색으로 기술한 부분은 지우고 작성하기 바랍니다.**

**목 차**

[1 프로젝트 목표 4](#_Toc37882199)

[**1.1 목표** 4](#_Toc37882200)

[**1.2 목표에 따른 기대효과** 5](#_Toc37882201)

[2 수행 내용 및 중간결과 6](#_Toc37882202)

[2.1 계획서 상의 연구내용 6](#_Toc37882203)

[**2.1.1 객체 탐지** 6](#_Toc37882204)

[**2.1.2 영상 검열** 7](#_Toc37882205)

[**2.1.3 음성 인식** 9](#_Toc37882206)

[**2.1.4 형태소 분석** 10](#_Toc37882207)

[**2.1.5 욕설 검열** 11](#_Toc37882208)

[**2.1.6 Web Front End** 13](#_Toc37882209)

[**2.1.7 Back End** 13](#_Toc37882210)

[2.2 수행내용 13](#_Toc37882211)

[**2.2.1 객체 탐지** 13](#_Toc37882212)

[**2.2.2 영상 검열** 14](#_Toc37882213)

[**2.2.3 음성 추출 및 분할** 15](#_Toc37882214)

[**2.2.4 Speech To Text** 15](#_Toc37882215)

[**2.2.5 Web Front End** 15](#_Toc37882216)

[**2.2.6 Back End** 17](#_Toc37882217)

[3 수정된 연구내용 및 추진 방향 20](#_Toc37882218)

[3.1 수정사항 20](#_Toc37882219)

[**3.1.1 동영상 업로드 방식 변경** 20](#_Toc37882220)

[**3.1.2 영상 검열 방식 변경** 21](#_Toc37882221)

[**3.1.3 웹페이지 설계 및 UI 변경** 22](#_Toc37882222)

[4 향후 추진계획 22](#_Toc37882223)

[4.1 향후 계획의 세부 내용 22](#_Toc37882224)

[5 고충 및 건의사항 22](#_Toc37882225)

[**6. 참고 문헌** 23](#_Toc37882226)

# 프로젝트 목표

**1.1 목표**

- YouTube 및 실시간 방송 플랫폼, 방송통신위원회 등 여러 기준들을 참고하여 동영상에 미성년자 시청 불가능 장면/욕설이 있는지 판단한다.

- 또한 현재 YouTube "노란 딱지" 정책 기준의 모호성과 불공정성에 대한 문제를 해결하기 위하여 사용자가 직접 자신의 영상 중 어떤 부분이 부적합한지 확인이 가능하게 하여 크리에이터들의 영상 제작에 있어서 효율성을 높힌다.(여기서 "노란 딱지"는 YouTube에 존재하는 기존 AI 영상 검열 장치이다.)

- 동영상에 미성년자 시청 불가능 장면/욕설이 있는지 판단되면 해당 장면이 어떤 가이드라인을 위반했는지 알려준다.

- 현재 YouTube에 존재하는 다양한 가이드라인 중 선정성, 폭력성, 모방성(흡연, 욕설 등)에 대한 가이드라인을 충족시키는 지 중점적으로 확인하는 검열을 실시한다.

1) "칼에 찔리는 장면이 명확하게 보이는 경우" - 폭력성

2) "담배를 입에 물고 흡연하는 장면이 명확하게 표출될 경우" - 모방성

3) "지나치게 상대를 비하하거나 과도한 욕설이 표현될 경우" - 모방성

4) "여성이 노출의 강도가 심하거나 나체인 경우" - 선정성

- 우리 프로젝트의 최종 목표는 다양한 영상 플랫폼에서 이 시스템을 사용할 만큼 정확도를 높이고, 많은 시간이 소요되지 않게 하는 것이다.

**1.2 목표에 따른 기대효과**

검열 시스템은 Youtube의 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 청소년들에 있어서 부적합한 영상들을 일차적으로 검열할 수 있어 동영상 업로드 플랫폼에서 실시하는 신고 기반 검열 시스템보다 더 많은 영상을 검열할 수 있고, 작업량과 비용 측면에서 효율적이다.

2. 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 제재가 가해지는데, 그 숫자가 매우 많아 효과적으로 이루어지지 않는다. 따라서 방송되고 있는 장면들을 실시간으로 딥러닝 모델에 넣어 검열할 수 있다.

3. 이러한 검열 작업이 원활하게 이루어지면, 현재 미성년자들이 무분별하게 시청하고 있는 영상들이 자동으로 검열됨으로써 청소년에게 유해한 매체물과 약물 등이 청소년에게 유통되는 것과 청소년이 유해한 업소에 출입하는 것 등을 규제하고 청소년을 유해한 환경으로부터 보호ㆍ구제함으로써 청소년이 건전한 인격체로 성장할 수 있도록 함을 목적으로 한다.

4. 현재는 미성년자들을 대상으로 부적합한 영상들을 검열하는 시스템을 만들었지만, 점차 데이터셋이 많아지고 다양한 라벨에 대해서 학습을 시킨다면, 교통사고, 자연 재해, 길거리 싸움, 집단 구타 및 학대 등 검열 대상을 확대시켜 활용이 가능하다.

5. 현 음성 검열은 단순 욕설에 대해서만 진행되지만, 데이터셋 양을 확대시켜 학습시킨다면 본질적 속성을 토대로 한 악의적인 모욕(인종차별적 비방), 미성년자에게 수치심을 주거나 모욕감을 주는 등 검열 대상을 확대시킬 수 있다. 여기서 본질적 속성이란, 보호 대상 집단 신분, 신체적 특징 또는 성폭행, 가정폭력, 아동 학대 등의 피해자 신분이다.

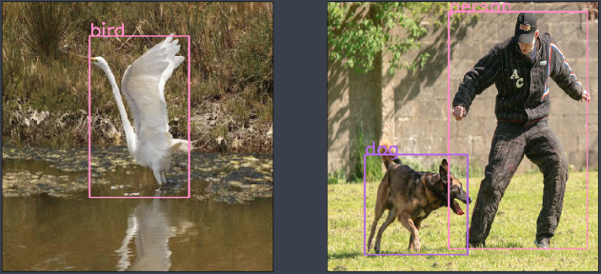
6. 이 시스템이 적극적으로 활용된다면, 해당 동영상 업로드 플랫폼에 대한 사람들의 신뢰도와 인식의 향상에 도움이 된다. 또한 미성년자 뿐만 아니라 일반인에 대해서도 각종 범죄에 쉽게 노출되지 않게 함으로써, 많은 사회적 범죄를 예방할 수 있다.

# 수행 내용 및 중간결과

## 계획서 상의 연구내용

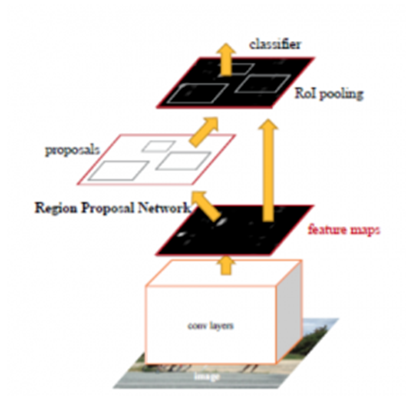
**2.1.1 객체 탐지**

객체 탐지는 이미지에서 관심 객체를 배경과 구분해 식별하는 자동화 기법이다. 아래와 같이 새와 사람 등 객체를 탐지하는 것을 의미한다. 이러한 객체 탐지는 딥러닝을 이용해 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 콘볼루션 신경망을 변형한 후 이를 객체 분류(Object Classification)에 활용한다. 이러한 사각형 상자들을 ‘윈도우(Window)’라고 부르고 이미지 상의 가능한 모든 위치와 크기를 포함한다. 각 창의 크기와 위치는 객체의 존재 여부에 따라 결정될 수 있고 객체가 있는 경우에는 그 범주도 결정할 수 있다. 우리는 다음과 같은 알고리즘을 사용한다.

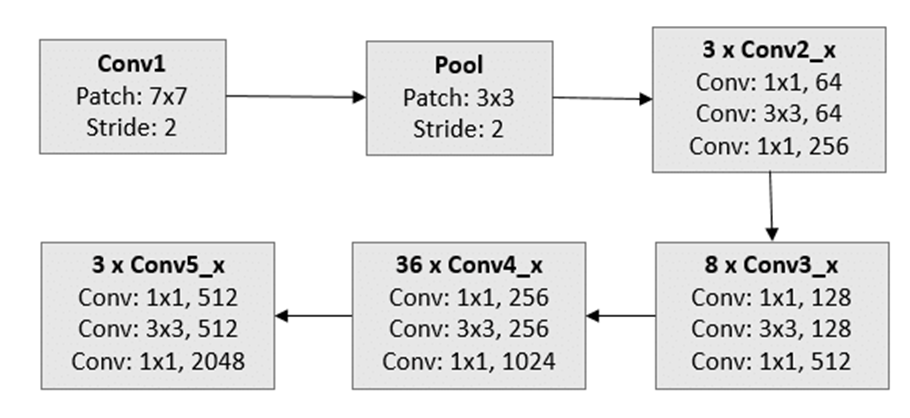
**이단계 방식의 객체 탐지 알고리즘, Faster RCNN** 알고리즘 이름에 ‘빠른(Faster)’이라는 단어가 포함되어 있지만 단일 단계 방식보다 빠른 처리가 된다는 뜻이 아닌 이전 버전이라 할 수 있는 RCNN 알고리즘과 Fast RCNN 알고리즘 보다 빠르다는 것을 뜻한다. 각 관심 영역(RoI; Region of Interest)에 대한 피쳐 추출의 계산을 공유하고 딥러닝 기반의 RPN을 도입해 구현한다.



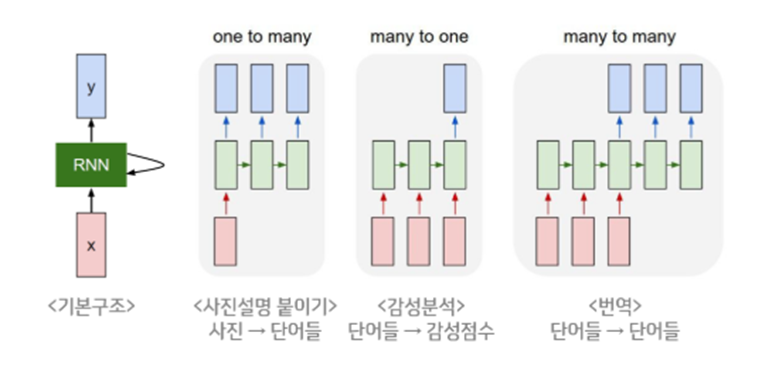
많은 CNN 레이어를 사용해 피쳐 맵 추출이 완료되면 RPN을 통해 개체를 포함하고 있을 가능성이 높은 윈도우가 다량으로 생성된다. 이 후 알고리즘은 각 윈도우에 있는 피쳐 맵을 검색하고, 고정 크기로 조정한 뒤(RoI 풀링) 클래스 확률과 해당 객체에 대한 더욱 정확한 경계박스를 예측한다. RPN은 YOLO와 같은 방식의 앵커 박스를 사용한다. 하지만 YOLO 알고리즘과는 앵커 박스가 데이터로부터 생성되는 것이 아니라 고정된 크기와 형태로 생성된다는 차이가 있다. 이 앵커 박스는 이미지를 보다 조밀하게 커버할 수 있다. RPN은 여러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

**2.1.2 영상 검열**

딥러닝 모델은 <사진 6>과 같이CRNN을 이용해 설계된 ResNetCRNN을 사용하여 학습을 진행한다. CRNN은 이미지를 처리하는 모델인 CNN과 가변적 크기의 데이터를 처리할 수 있는 RNN이 결합된 모델이다. 우리는 여러 CNN 모델 중 ResNet152 pre-trained 모델을 선택했다. ResNet152 모델은 이미지를 처리하는 모델 중 뛰어난 성능을 보이는 모델이며, 그 구조는 다음과 같다.

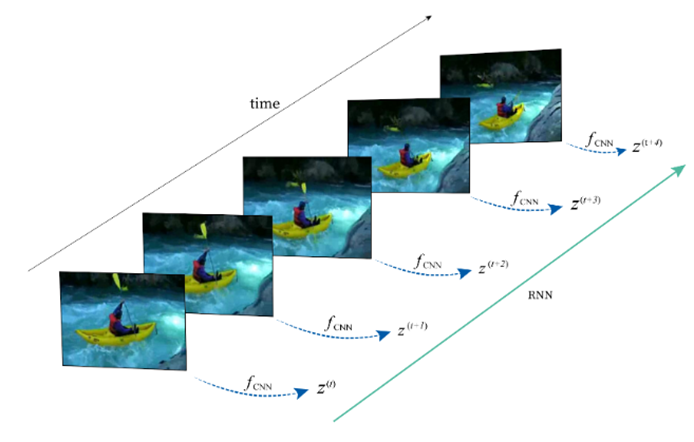


아래는 RNN의 구조에 대한 사진이다.



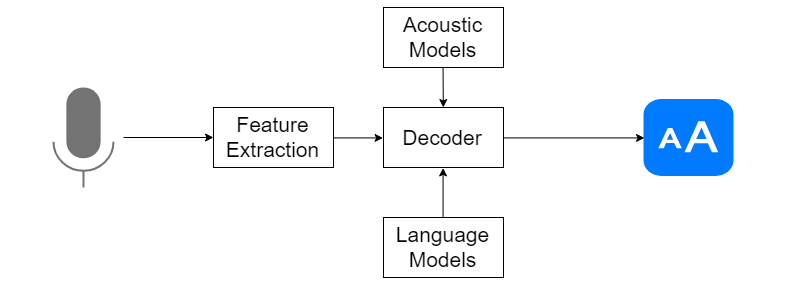
RNN은 시퀀스(연속적) 데이터를 모델링한다. RNN이 기존의 뉴럴 네트워크와 다르게 ‘기억’(다른 말로 hidden state)을 갖고있다. 여기서 기억은 지금까지의 입력 데이터를 요약한 정보라고 볼 수 있다. 즉 RNN은 영상에서 연속적인 프레임을 추출하고 이 프레임들에 대한 정보를 가지고 있다. 또한 새로운 프레임이 들어올때마다 모델은 자신의 기억을 조금씩 수정한다. 입력을 모두 처리하고 난 후 남겨진 기억은 시퀀스 전체를 요약하는 정보가 된다. 따라서 즉 순환적이라는 이름이 붙고, 이런 반복을 통해 영상을 처리할 수 있다.

다음은 적용된 CRNN 모델에 대한 간략한 구조이다.



**2.1.3 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.

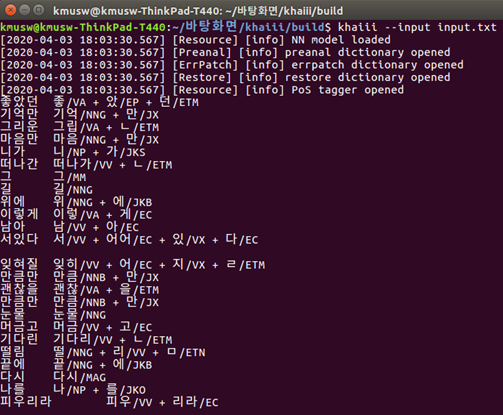


STT(Speech To Text) 는 잡음처리와 특징추출된 음성 데이터를 언어모델과 음향모델이 결합된 디코더를 통해 단어를 인식 후 출력하면 이를 텍스트 형태로 저장한다.

**2.1.4 형태소 분석**

형태소 분석기는 텍스트 형태의 데이터를 형태소 단위로 분리한다. 전처리 과정을 통해 형태소 단위로 분리된 데이터는 비속어의 여부를 판단하기에 유리하다. (수정필요)

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.



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

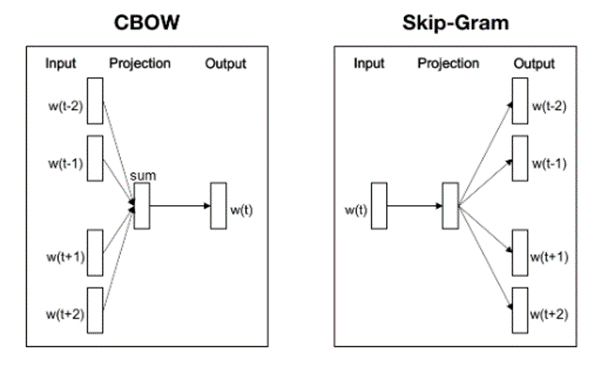
오분석이 발생할 경우 모델의 파라미터를 수정하여 바른 결과를 내도록 하는 것이 매우 어렵다. 이를 위해 khaiii는 신경망 알고리즘의 앞단에 기분석 사전을, 뒷단에 오분석 패치라는 두 가지 사용자 사전 장치를 마련하였다. 기분석 사전은 단일 어절에 대해 문맥에 상관없이 일괄적인 분석 결과를 갖게 하고 싶을 경우 사용하고 오분석 패치는 여러 어절에 걸쳐 충분한 문맥과 함께 오분석을 바로잡아야 할 경우에 활용한다. 이러한 두 기능으로 형태소 오분석의 확률을 줄일 수 있고 또한 본 프로젝트에 맞게 형태소 분석을 진행할 수 있다.

**2.1.5 욕설 검열**

khaiii 형태소 분석기로 텍스트에서 분석된 형태소들은 사전 훈련된 FastText 모델을 이용하여 욕설데이터 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 해가며 학습을 한다. 그리고 이러한 과정에서 주변 단어를 보고 중심 단어가 무엇인지를 예측하는 CBOW(continuous bag-of-words) 모델과 중심 단어를 보고 어떤 주변 단어가 등장했는지를 맞추는 Skip-Gram 모델로 임베딩 기법에 따라 나뉘게 된다.



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 된다. 따라서 같은 corpus 크기를 갖더라도 학습량이 네 배 차이가 나게 되어 CBOW 모델에 비해 Skip-Gram 모델의 성능이 더 좋다. 따라서 우리 팀은 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, 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 어휘들의 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.

**2.1.6 Web Front End**

**2.1.7 Back End**

딥러닝 모델 학습과 모델의 영상과 음성 검열에 있어 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 Amzon 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. 또한 S3에서 업로드 완료되는 이벤트에 따라 Lambda함수를 프레임을 추출하고, 전처리를 진행하는 코드로 구현한다.

AWS EC2 instance는 AWS Deep Learning AMI를 채택하여 딥러닝 모델의 학습을 진행한다. 모델 학습에 있어 필요한 데이터셋은 Amazon S3에 저장한다.

Apache 서버를 통해 웹페이지를 배포한다.

웹페이지의 이용자와 AWS를 통해 얻어진 데이터의 주체가 동일인물인지 파악하는 것과 Back End에서 일처리(검열 작업)의 종료를 웹페이지에 알려주는 것의 진행을 위해 Socket 서버를 구축하여 통신한다. Socket은 SocketIo 라이브러리를 활용한다.

## 수행내용

**2.2.1 객체 탐지**

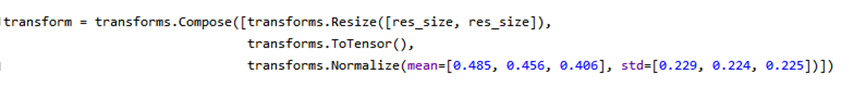
먼저 Img와 annotation된 xml 데이터셋의 구축을 위해 labelImg 라이브러리를 활용한다. labelImg를 이용한 데이터셋 구축은 다음과 같이 구축했다.

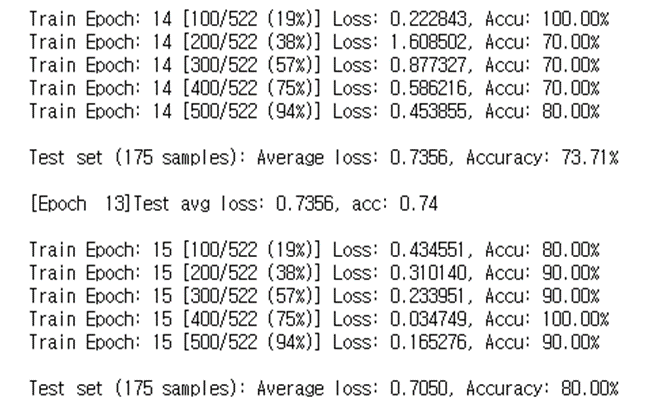
(사진 없어짐)

이후 구축이 완료된 데이터셋을 이용해 Faster R-CNN 딥러닝 모델이 적용된 detecto라는 라이브러리로 객체 탐지 학습을 진행한다.

**2.2.2 영상 검열**

"Knife", "Smoke", "Adult", "Game"에 해당되는 총 4종류의 CRNN 모델을 이용한다. 먼저 Encodder - Decoder로 구성된 CRNN 딥러닝 모델에 Dataset을 넣어서 학습시키기 이전에, 라벨링이라는 작업을 진행한다. 라벨링은 OneHotEncoder를 이용한다. 라벨링이 모두 끝난 이후 데이터 전처리를 통해 학습을 가능하게 만든다. 전처리는 Resize, ToTensor(), Normalize를 진행한다.





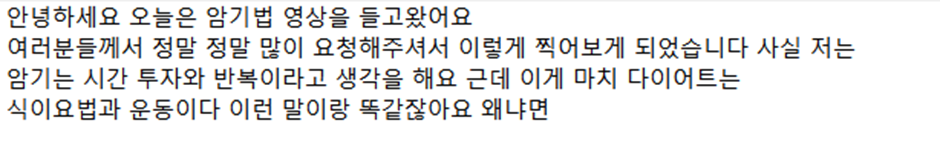
사전에 정의된 res\_size(224) \* res\_size(224)로 이미지 크기를 Resize하고, ToTensor()을 통해 Tensor차원으로 변환한다. 마지막으로 정규화를 진행한다. 이 전처리는 학습 데이터셋과 테스트 데이터셋 모두에 해당된다. 총 80 Epoch로 학습이 진행되었으며, 학습과 테스트가 병행해서 이루어지고, 가장 높은 정확도를 보였던 테스트 단계에서 모델이 저장된다.

**2.2.3 음성 추출 및 분할**

영상에서 moviepy 라이브러리를 통해 음성 데이터를 추출한 후, pydub 라이브러리를 사용해 음성 데이터를 분할하여 저장한다. 이때 각 음성데이터의 길이가 5초 단위로 저장될 수 있도록 분할한다.

**2.2.4 Speech To Text**

분할된 음성 데이터를 Google Cloud Speech API 를 통해 텍스트 파일로 변환한 후 지정된 형식의 파일(csv, txt 등) 로 저장한다.



**2.2.5 Web Front End**

초기 web page 구상 시 로그인, 파일 업로드, 업로드 결과 확인 등 여러 컴포넌트들이 필요했기 때문에 기능별 컴포넌트들을 편리하게 관리할 수 있고 무엇보다 여러 라이브러리들이 존재하고 그 라이브러리들이 필요 할때 마다 쉽게 다운로드 받고 사용할 수 있는 react.js를 선택했다.

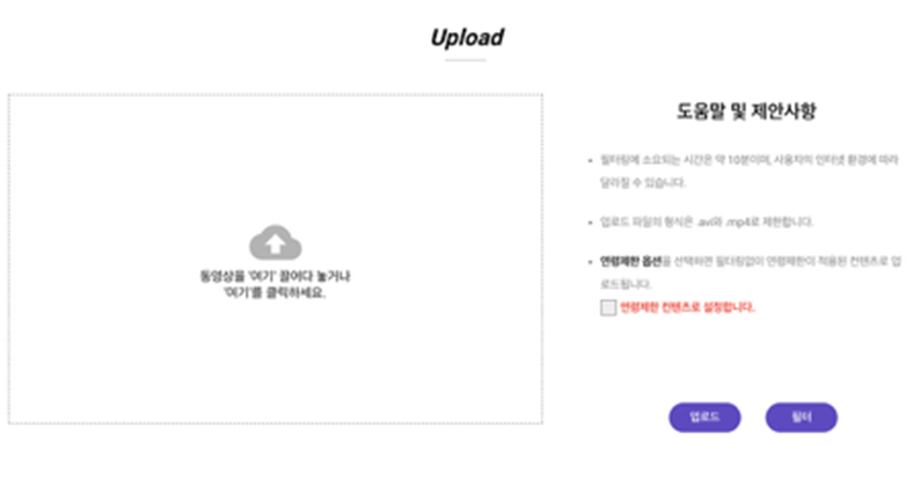


<YouHi WEB design 초안>

컴포넌트 종류는 다음과 같다.

1) 파일 업로드

파일 업로드의 경우 file type의 input form을 사용할 경우 직접 파일을 선택해 올리는 것만 가능하기 때문에 드래그 앤 드랍 형식의 파일 업로드를 추가하기 위해 react-dropzone 라이브러리를 사용해 구현했다.



파일을 선택했을 경우 하단 좌측 이미지와 같이 업로드 버튼 위의 파일명으로 파일이 정상적으로 선택되었는지 확인할 수 있다. 파일 선택을 완료했을 경우 업로드 버튼을 누르면 프로그래스 바가 나타나면서 파일이 어느정도 업로드 됐는지 확인 할 수 있다. 업로드가 완료되었다면 필터 버튼을 클릭해 필터링을 진행할 수 있다. (사진 수정)



2) 필터링 확인

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

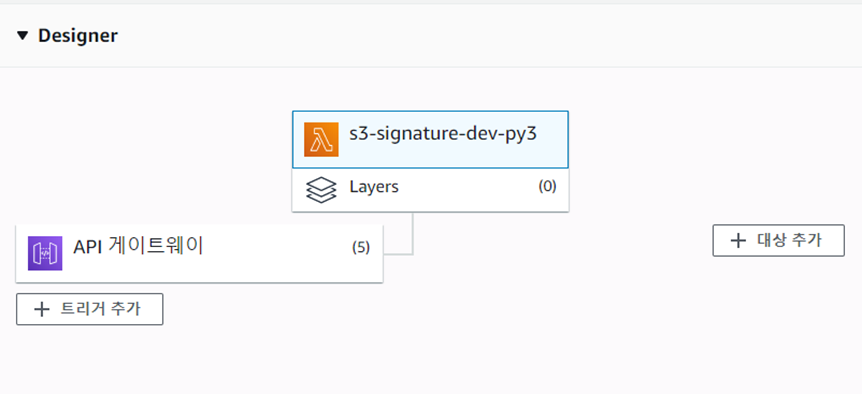
<Filter css 수정 후 이미지> (사진없음)

각각의 영상, 음성 필터링 이미지를 클릭 시 새로운 팝업 창이 뜨고 서버로 부터 받은 필터링 된 결과를 화면에 나타내준다. (어떤 식으로 필터링 되고 어떤식으로 결과가 나오는지는 다른 곳에서 설명할 것 같아서 안 넣음) 영상 업로더는 필터링 된 결과를 확인하고 문제가 있을 경우 이의신청 및 문의하기 버튼 클릭으로 문제를 제기할 수 있다.

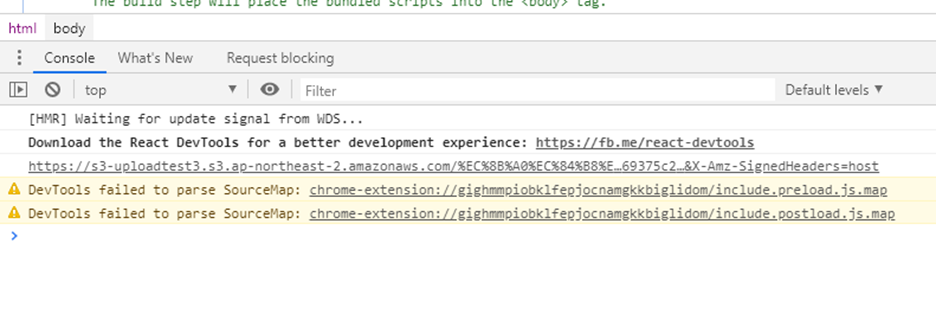
3) 로그인

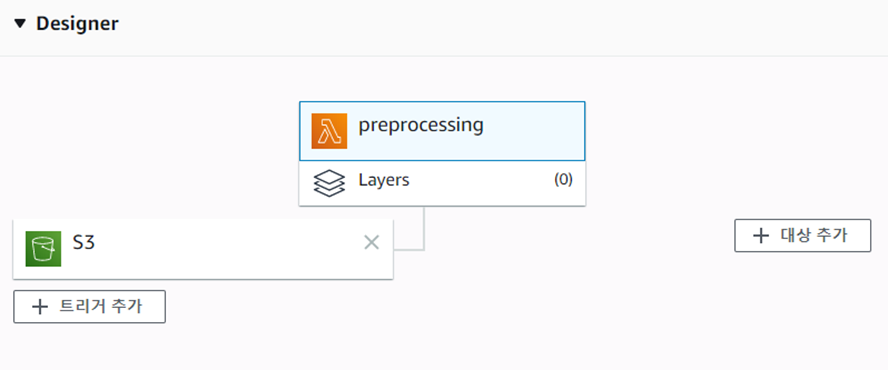
영상을 업로드 하기 위해 본인 인증이 가능해야 하기 때문에 로그인 기능을 구현하고자 한다. 직접 로그인 폼을 만들어 DB로 관리하기엔 문제가 많아 Google login api를 사용해 Google 계정으로 로그인 할 수 있게 구현할 것이다. 영상 업로드 기능은 로그인 후 이용할 수 있다.

**2.2.6 Back End**



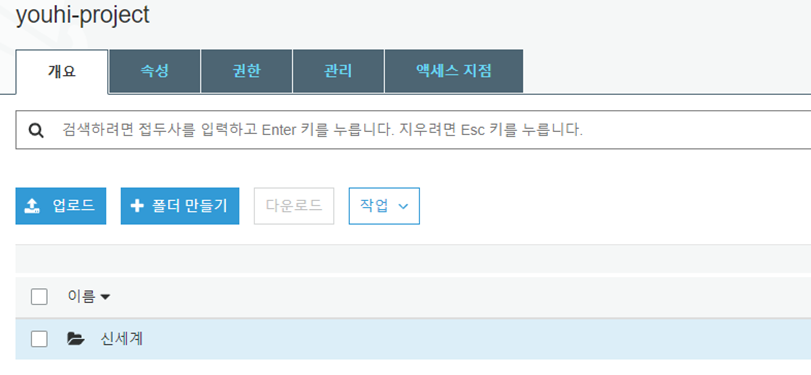
Aws Gateway Api와 Lambda를 통해 pre-signed url을 GET Http 메소드를 통해 얻어온다. 하단에 https://s3-uploadtest3로 시작하는 url이 pre-signed url이다.

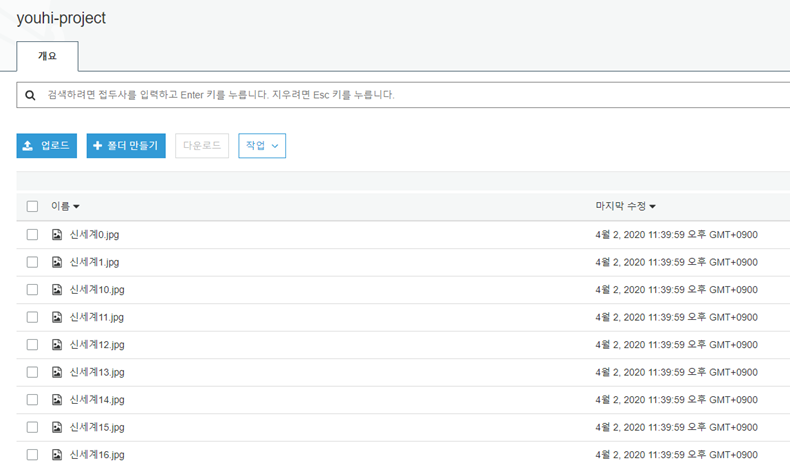






"신세계.avi"를 s3-uploadtest3에 업로드한 화면이다. 해당 업로드는 pre-signed url을 통해 업로드 하였다.





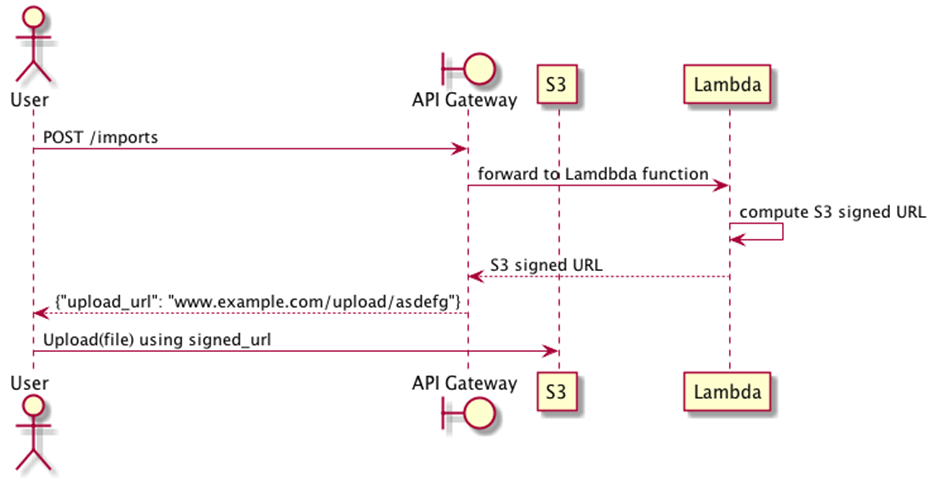
s3-uploadtest3 버킷에 업로드 완료되는 이벤트에 따라 youhi-project 버킷에 파일명에 해당하는 폴더를 생성하고 5초에 하나의 프레임을 추출한다.

# 수정된 연구내용 및 추진 방향

## 수정사항

**3.1.1 동영상 업로드 방식 변경**

기존 동영상을 S3에 업로드할 때 Gateway Api와 Lambda를 통해 "업로드"를 바로 진행했으나, 5MB이상 파일을 업로드하지 못하는 문제점을 발견하였다. 또한 그 속도도 S3에 Direct하게 업로드 하는 것보다 느리다고 판단되었다. 따라서 다음과 같은 구조로 업로드 방식을 변경하였다.



User가 업로드를 시작하면 Gateway Api와 Lambda를 거쳐서 signed url을 전달받는다. 이 url은 S3에 Direct하게 업로드할 수 있는 S3 url이며 유효시간이 정해져있어, 일정 시간이 지난 이후에 이 url은 사용하지 못한다. 따라서 Direct하게 접근 가능하지만 기존과 마찬가지로 외부에 공개될 우려가 없어 보안을 안전하게 유지할 수 있다.

**3.1.2 영상 검열 방식 변경**

기존 Video Classification 단일 모델을 사용하는 것은 문제가 많은 것으로 판단되어 Object Detection를 결합하고 4개의 Video Classification 모델을 이용해서 영상 검열을 진행한다. 또한 모든 영상에 대한 프레임을 추출하여 Video Classification을 진행하는 것이 아닌 5초에 한 frame을 추출하여 Object Detection을 진행하고, Object Detection으로 칼, 총, 담배 등이 검출되면 Object에 따라 각 Video Classification이 진행되는 방식으로 변경했다. 예를 들어, Object Detection으로 칼이 검출되면, 해당 영상이 요리를 하는 영상이거나 칼에 대한 제품을 소개하는 영상 혹은 칼로 전투를 치르지만 15세 이상 적용 대상인 영상인지 Video Classification으로 최종 판단하는 과정을 거치게 된다. 이와 같이 이루어질 경우, 오분류가 매우 줄어들 것으로 예상되며 또한 기존 모든 프레임 추출 방식에 비해 시간과 메모리가 극적으로 줄어들 것이다.

**3.1.3 웹페이지 설계 및 UI 변경**

기존 웹페이지에 YouTube등 동영상 플랫폼에서 이용할 수 있는 시스템처럼 보이기 위해 연령 제한 옵션과 해당 플랫폼에 대한 설명 등을 설정하고 표출시켰지만, 계획 발표의 피드백인 "YouTube에 적용시키겠다라는 목표가 현실적으로 불가능하니 새롭게 설정해야된다"를 반영하여 "YouTube 등 영상 플랫폼에서 활용 가능한 시스템을 개발하겠다"로 새롭게 설정했다. 그에 따라 웹페이지에 있던 연령 제한 옵션 등을 삭제했다.

# 향후 추진계획

## 향후 계획의 세부 내용

1) 정확도 측정 매트릭스

2) 영상 및 음성 검열이 왼료된 것을 다시 웹페이지에 띄우는 것 구현

3) 여러 사용자가 동시에 영상 업로드 및 검열을 실시 했을 때 서버가 안정적인지 확인

4) 영싱 검열 및 이미지 분류기 딥러닝 모델의 지속적인 개선(정확도 향상) -> 모델 변경 및 기법 변화, 데이터셋 지속적으로 추가

5) 웹 디자인 및 UI 지속적으로 수정(이용자의 편의를 최대한 추구)

6) 로그인 기능 및 문의 사항, 이의 신청 기능 구현

# 고충 및 건의사항

유튜브 및 실시간 영상 송출 플랫폼과 한국 방송통신위원회 등 19세 이상 관람가능 조건이 각기 다 다르고, 불분명합니다. 기준이 명확하지 않습니다. 그래서 어떤 영상은 연령 제한없이 업로드되어있고, 동일한 내용임에도 불구하고 어떤 영상은 연령 제한이 걸려있는 것을 심심찮게 볼 수 있습니다. 이에 따른 여러 문제점도 있는 것으로 보입니다.

따라서 저희는 위 기준들을 고려해보고 여러 기사들을 참고해서 저희만의 기준을 독자적으로 마련했습니다. 만약 교수님들이나 다른 학생들이 봤을 때 "어 왜 저 장면이 19세지?"라고 생각이 들으신다면, 아 저 친구들은 저 장면이 19세 이상 관람기준으로 봤구나 라고 생각해주시면 되겠습니다.

**6. 참고 문헌**

*딥러닝 기반의 음성인식 기술 (한국데이터 산업진흥원, 2018)*