비디오 벡터를 이용한

내용 기반 유튜브 영상 추천 시스템

**경희대학교 컴퓨터공학과 2020년 1학기 캡스톤디자인1**

**팀 이름 : Profit Hunter**

**2015104192 윤영빈**

**2015104193 윤준현**

**2015104208 이태현**

**2015104209 이현규**

**요약**

비디오 벡터를 이용한 내용 기반 유튜브 영상 추천 시스템(Context-based Youtube Video Recommendation System Using Video Vector)은 기존 유튜브의 영상 추천 알고리즘을 개선하기 위한 시스템이다. 기존 알고리즘은 동영상 투고자가 직접 남긴 태그와 이용자의 과거 시청 데이터에만 의거하여 영상을 추천한다는 한계점이 존재한다. 이에 본 논문에서는 영상의 내용을 대표하는 비디오 벡터를 통해 내용 상으로 연관성 높은 영상을 추천하는 시스템을 제안한다. 해당 시스템은 동영상 세그먼트 분석 모델을 사용하여 동영상에 태그를 할당하고, 태그를 기반으로 비디오 벡터를 생성한 후, 각 비디오 벡터 간의 유사도를 계산하여 관련 영상을 추천한다. 본 시스템을 사용하면 기존 유튜브 알고리즘과 달리 내용 상으로 관련이 깊은 동영상 목록을 추천할 수 있을 것이다.

**1. 서론**

**1.1. 연구배경**

2010년 초반 스마트폰과 무선 인터넷의 보급률이 높아지면서 가정에서는 물론, 대중교통, 카페 등 심지어 길거리에서 걸으면서까지 다양한 인터넷 동영상을 훨씬 더 간편히 시청할 수 있게 되었다. 이러한 변화 가운데 동영상들을 쉽고 빠르게 검색, 접근 및 시청할 수 있는 'Youtube'(이하 유튜브) 가 각광받고 있다.

업로드한 동영상이 높은 조회 수를 기록할수록, 그리고 동영상에 붙어 있는 광고 재생 시간과 횟수가 높을수록 동영상 업로더(이하 유튜버)는 유튜브 본사로부터 더 높은 동영상 투고 금액(수익)을 받는 시스템이다. 이 때문에 유튜버들은 단순히 재미 있는 동영상을 제작해야 하는 것뿐만 아니라, 자신들이 올린 동영상과 유튜브 페이지를 적절한 유튜브 시청자들에게 노출 및 홍보해야 한다.

또한 시청자들의 자체 영상 키워드 검색 말고도, 유튜브 쪽에서 직접 시청자들 각자가 좋아할만한 동영상의 카테고리를 분류, 파악하여 시청자들에게 영상을 추천해주는 시스템인 일명 ‘유튜브 알고리즘’이 존재한다. 업로드 된 영상들의 주제와 장르를 범주화시켜서 같은 범주를 자주 보았던 다른 시청자들의 유튜브 메인 화면에 영상을 추천해주는 것인데, 유튜버들 입장에서는 이 알고리즘이 ‘자신의 유튜브 페이지와 동영상을 모르는 새로운 시청자들’의 시선을 끌 수 있는, 중요한 시스템이라는 것이다.

한편, 현재 시행중인 동영상 업로드 방식을 살펴보면 유튜버가 동영상을 업로드하기 전에 자신이 직접 동영상의 카테고리를 분류하고 태그를 기입한다. 우리는 이러한 업로드 방식을 개선하기 위해 딥러닝 모델을 통하여 업로드된 동영상의 내용을 분석하여 생성된 태그를 기반으로 영상의 내용을 대표하는 비디오벡터를 생성하고, 관련도 높은 유튜브 영상을 추천해주는 시스템을 개발한다.

**1.2. 연구목표**

목표 설명에 앞서 이 웹 서비스의 의의는 기존 유튜브 영상 검색 엔진과 유튜브 알고리즘에 부족함을 채워 넣기 위함에 있다. 우리는 업로드한 동영상을 최대한 노출시킬 수 있는 최적의 영상 태그, 그리고 해당 영상과 비슷한 내용의 유튜브 영상 링크를 이 프로그램 유저들에게 웹 UI로써 보여주는 것을 궁극적인 목표로 삼았다. 다시 말해서, 이른바 현재 유튜브 영상 검색을 더욱 효율적으로 도와줄 수 있는 태그 생성과 이 태그로 비슷한 내용의 유튜브 영상 링크를 출력시키고자 한다.

첫 번째 목표는 딥러닝 테스트를 위한 수많은 동영상 데이터셋을 분석, 분류할 수 있는 머신러닝/딥러닝 모델을 조사하는 것이다. 영상 데이터셋으로는 사용 가능한 수백만 개의 유튜브 동영상을 1,000개의 클래스로 정의한 yt8m segment\_level 데이터셋을 사용한다. 이 샘플을 읽어 들일 수 있는 딥러닝 모델을 사용하여 ‘내용 기반 영상 분류를 통한 영상 추천’을 할 것이다.

두 번째 목표는 사용자들에게 태그와 유튜브 관련 영상 링크를 보여주기 위한 웹 페이지 프론트엔드, 백엔드를 구축하는 것이다. Drag & Drop을 통하여 영상을 입력하고, 해당 영상을 백엔드 처리 과정으로서 첫 번째 목표에서 말한 모델로 분석시킨다. 그 결과로 얻은 동영상의 각 태그들의 유사도를 반영하여 비디오 벡터를 생성하고, 연관된 유튜브 링크들을 구한다.

끝으로, 위에서 말했듯 사용자가 이 웹 서비스를 유튜브와 함께 활용한다면, 유튜브 크리에이터는 영상 홍보를 위하여 혼자 깊이 고민할 필요 없이 유튜브 알고리즘에 잘 녹아들 수 있는 내용 기반 추출 태그를 얻을 것이고, 시청자는 자신이 좋아하는 주제의 동영상을 더 확실하고 자주 만날 수 있을 것이다.

**2. 관련연구**

**2.1. 컴퓨터 비전**

동영상은 여러 장의 이미지들이 시간의 흐름에 따라 연속적으로 재생되는 영상물 파일이다. 따라서 동영상에 사용되는 이미지를 인식할 수 있는 기술이 요구된다. 컴퓨터 비전은 기계의 시각에 해당하는 부분을 연구하는 분야로, 이미지에서 정보를 추출한다.

**2.1.1. 물체인식**

물체인식은 미리 학습한 정보를 바탕으로, 물체를 보고 그 종류와 크기, 방향이나 위치 등 3차원적 공간정보를 알아내는 기술이다. Convolutional Neural Network(CNN)과 같은 딥러닝 모델이 사용될 수 있으며, 이를 위해 수많은 이미지들을 데이터셋으로 활용해야만 한다.

**2.1.2. 패턴인식**

패턴인식은 컴퓨터와 같이 계산이 가능한 기계장치가 어떠한 대상을 인식하는 문제를 다루는 분야로, 문자나 도형, 음성 등의 정보로부터 특징을 찾아내는 것을 말한다. 동영상과 같은 영상정보의 경우 작은 화소로 분할한 후, 각 화소의 명암을 8비트의 수치로 대치한다. 그 후 대치한 값이 급격하게 변하는 화소들을 추출하여 외곽선에 대한 정보를 얻는다. 이를 이용해 방향, 원주 등을 인식한다. 오디오 정보의 경우 강도나 위상 등에 따라 입력정보를 디지털화 하고 이를 이용한다.

**2.2.** **머신러닝**

머신러닝 또는 기계학습은 인공지능의 한 분야로, 컴퓨터가 학습할 수 있도록 하는 알고리즘과 기술을 개발하는 분야를 말한다. 즉 기계가 코드로 명시되지 않은 동작을 데이터로부터 학습하여 새로운 데이터를 처리할 수 있도록 하는 것을 의미한다. 현재 머신러닝은 컴퓨터 과학을 포함한 대부분의 분야에서 활용되고 있는데, 컴퓨터 비전, 물체 인식 등의 분야에서 활발히 응용되고 있다.

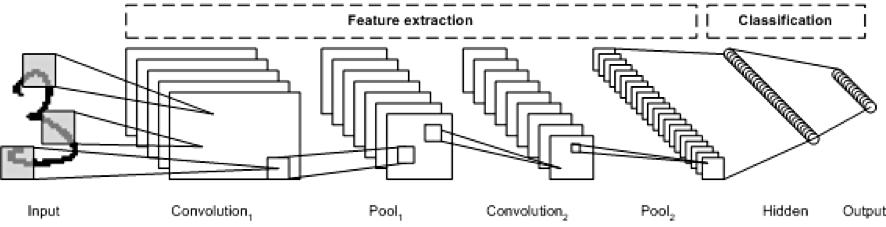
**2.2.1. 자연어 처리**

자연어 처리는 컴퓨터를 이용해 사람의 자연어를 분석하고 처리하는 기술로, 자연어를 분석, 이해, 생성하는 기술들을 사용한다. 자연어 분석은 그 정도에 따라 형태소 분석, 통사 분석, 의미 분석, 화용 분석의 네 가지로 나눌 수 있다. 자연어 이해는 컴퓨터가 자연어로 주어진 입력에 따라 동작하게 하는 기술이며, 자연어 생성은 영상이나 이미지, 표의 내용을 사람이 이해할 수 있는 자연어로 변환하는 기술이다. 영상 분석 후 가중치가 높은 태그들을 추천할 때 사용된다.

**2.3. 인공신경망 모델**

**2.3.1. Convolutional Neural Network (CNN)**

CNN(합성곱 신경망)은 이미지의 특징을 추출하여 학습하는 데 적합한 인공 신경망 모델이다. 만약 이미지 데이터로 전연결(Fully Connected) 신경망을 학습시켜야 할 경우, 3차원(공간 2차원 + RGB 1차원) 데이터를 1차원으로 평면화 시켜야 하는데, 차원을 축소하는 과정에서 필연적으로 공간 정보가 손실될 수밖에 없다. 즉 이미지를 전연결 신경망으로 학습시키는 경우에는 데이터의 특징을 잘 추출하지 못하여 결과가 좋지 못하게 된다. 이 때 이미지의 3차원 공간 정보를 손실하지 않은 상태로 학습이 가능한 모델이 CNN이다.



**[그림 1] CNN의 학습 구조 예시**[1]

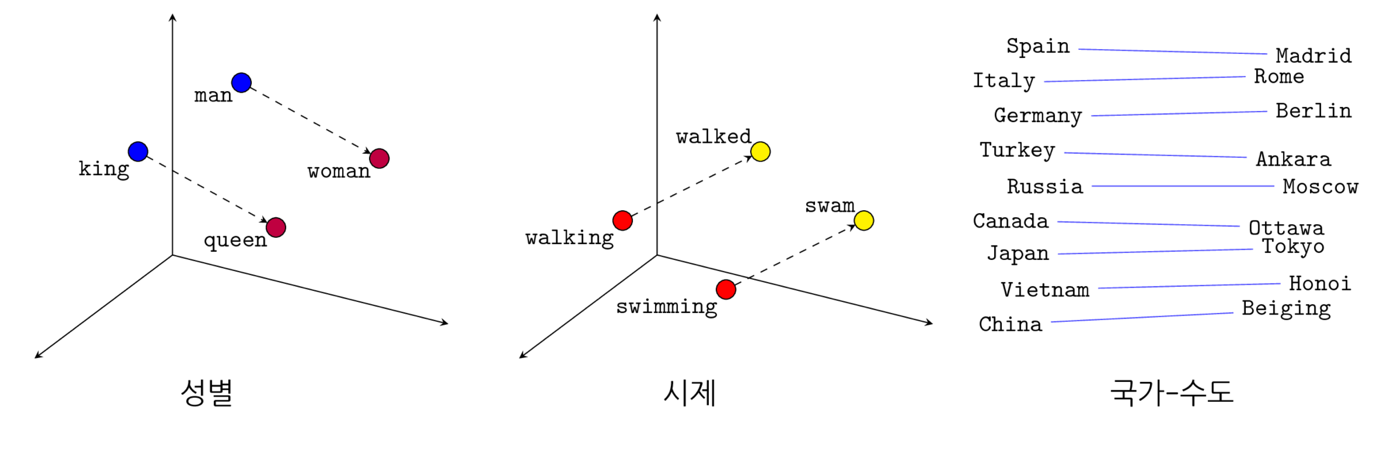
CNN의 신경망 계층은 Convolution Layer와 Pooling Layer가 반복되는 구조로 구성되어 있다. Convolution Layer에서는 합성곱 연산을 통해 이미지의 특성이 추출된 Feature Map을 생성하며, Pooling Layer에서는 피쳐 맵에 있는 특정 원소를 강조하거나, 중요하지 않다고 생각되는 원소를 버리는(Max Pooling) 등 최적화 연산을 수행한다. Convolution Layer와 Pooling Layer가 여러 번 반복되며 이미지의 특징을 효과적으로 추출하며, 마지막 Hidden Layer에서는 입력된 이미지가 각 category에 포함될 확률을 출력한다.

**2.3.2. Long Short-Term Memory (LSTM)**

LSTM(장단기 메모리)[2]는 RNN의 Vanishing Gradient Problem을 해결하기 위해 만들어진 RNN의 변형 모델이다. RNN은 사전 정보를 기억할 수 있는 능력은 갖추고 있으나, 사전 정보와 그 정보를 사용하는 지점 사이의 거리가 멀 경우 back-propagation 시 gradient가 소실되어 제대로 업데이트가 되지 않는 문제가 발생할 수 있다. 이 문제를 해결하기 위해 LSTM은 기존 RNN 구조에서 Gate를 추가하여 연상하게 된다. Gate가 추가된 LSTM은 직전의 정보를 버릴 것인지, 만약 정보를 보존한다면 강도를 얼마나 하여 새 데이터와 결합시킬 것인지 등을 추가로 판단하게 된다. 그다지 중요하지 않다고 판단된 기존 정보는 버려지게 되므로, 중요하다고 판단되는 적절한 값만 가중치(Gradient) 업데이트 시에 효과적으로 반영하게 된다.

**2.3.3. Word2Vec**

Word2Vec[3]은 단어를 vector로 변환(embedding)하는 2층 인공 신경망 모델이다. 다량의 텍스트 데이터(corpus)를 입력하면 해당 텍스트에 나온 단어들 모두를 n차원 형태의 vector로 변환한다. 예를 들어, ‘한국’이라는 단어를 [0.3, 0.5, 0.7] 이라는 3차원의 vector로 표현할 수 있는 것처럼 corpus에 나오는 모든 단어를 특정 vector에 대응시킨다. 이 때 나오는 vector 결과물은 단순히 단어를 숫자로 표현하는 것에 그치지 않고, 단어 간의 관계를 적절히 나타내는 형태로 vector space 안에 표현된다.



**[그림 2] Word Embedding 예시**

그림 1은 Word2Vec으로 embedding 된 단어들이 서로 어떤 관계에 있는지 보여 준다. 왼쪽 성별 그래프의 경우, 남자(man), 여자(woman), 왕(king), 여왕(queen)이 각각 벡터로 변환되어 vector space 안에 표현되어 있다. 이 때 man와 woman단어 간의 거리가 king과 queen 단어 간의 거리와 유사한데, 이것은 남자와 여자 간의 단어 관계가 왕과 여왕 간의 관계와 비슷하다는 것을 보여 주며, 실제로도 그러하다(남자:여자 = 왕:여왕). 동일한 맥락으로 걷다(walk)와 수영하다(swim) 단어 간의 진행형과 과거형 사이의 관계가 서로 비슷하게 배치되어 있으며, 국가-수도 간의 관계도 비슷하게끔 배치되어 있다. Word2Vec을 사용하면 단어 하나 당 100~500차원의 벡터에 대응되도록 할당할 수 있으며, 단어 사이의 관계를 cosine similarity 등의 방법으로 파악함과 동시에 자연어를 기계 학습용 수치 데이터로 사용할 수 있다.

**2.4. Youtube 8M Video Understanding Challenge**

Youtube 8M Video Understanding Challenge[4]는 Kaggle에서 개최된 유튜브 영상 자동 분류 대회이다. ImageNet과 유사하게, Youtube-8M이라는 유튜브 영상 데이터 세트로 모델을 학습시켜, 해당 모델이 영상 분류를 얼마나 잘 해내는지를 경쟁한다.

**2.5. 기존 연구**

**2.5.1. 영상 자동 태깅**

현재 유튜브는 동영상 업로드 시 제목을 기반으로 태그들을 추천한다. 또한 태그가 하나 이상 입력되어 있다면 입력된 태그들 또한 다른 태그들을 추천하는 데이터로 사용한다. 즉, 영상 자체를 분석하여 태그를 추천하지는 않는다는 뜻이다.

영상 자동 태깅과 비슷한 연구로 영상 분류, 영상 인식 등이 있다. 동영상을 특정 카테고리로 분류하거나 짧은 동영상 속 상황을 자연어로 출력하는 연구들인데, 결과를 Labeling한다는 점에서 태깅과 비슷한 점이 있다. 그러나 이들의 경우 가장 가중치가 높은 결과값 하나만을 출력하고, 결과값의 논리적 상위집합과 하위집합 간의 관계를 고려하지 않으므로 다수의 키워드를 지정할 수 있는 태깅과는 다소 차이가 있다. 일례로, 자동차 집합과 승용차 집합은 논리적 관계가 존재하지만 언급한 연구들의 결과만으로 두 집합 간 논리적 관계가 있다는 것은 알기 어렵다.

현재 동영상 연구를 위해 사용되고 있는 데이터셋은 10초 내외의 짧은 길이의 영상들로, 분 단위의 영상들을 여러 조각으로 나눈 것들이다. 또한 짧은 영상 내에서도 시간 효율을 위해 모든 프레임을 이용하지 않고, 일정 간격을 둔 뒤 추출한 프레임들의 집합을 생성해 이용한다.

**2.5.2. 유튜브 추천 알고리즘**

한국언론진흥재단의 논문 ‘유튜브 추천 알고리즘과 저널리즘’[5]에 따르면 현재의 기술로는 영상의 내용을 파악하기 어렵기 때문에, 유튜브의 추천 알고리즘에서 영상의 내용은 반영되지 않는다고 한다. 현재의 유튜브의 추천 영상 목록을 만드는 기준을 보면, 이용자가 과거에 시청했던 영상, 좋아요 버튼을 클릭했던 영상을 기반으로 관련 영상을 추천하고 있다. 그리고 전통적 언론사에 대한 선호 현상이나, 제목의 키워드 개수가 높을수록 가중 추천하는 현상 등, 개발자가 임의적으로 선택한 알고리즘에 따라 추천 영상이 선택되는 사례가 빈번히 관측된다. 이러한 방식은 영상 자체의 내용을 분석에 활용하지 않기 때문에 이용자의 성향에 맞지 않는 영상이 추천되기도 한다.

**2.6. 기존 연구의 문제점 및 해결 방안**

**2.6.1. 영상 자동 태깅 연구의 문제점**

동영상은 여러 장의 이미지들이 시간의 흐름에 따라 연속적으로 재생되는 영상 파일이지만 멀티미디어 매체이므로 영상만이 출력되는 경우는 드물고, 일반적으로 오디오와 함께 재생된다. 현재의 연구는 영상의 프레임들을 중점으로 진행되고 있는데, 오디오 정보 또한 중요하다. YouTube-8M Challenge에서 상위를 차지한 팀들은 영상과 오디오 정보를 모두 이용하는 모델을 사용했다.

그러나 단지 오디오 정보를 사용한다고 모델의 성능이 좋게 나오는 것은 아니다. 하나의 영상에 여러 주제가 담겨 있거나 여러 항목이 존재하는 데이터의 경우, 또는 영상은 움직이지 않지만 오디오만 재생되는 데이터의 경우와 같이 예외가 존재한다. 또한 모델이 분석하지 못할 정도로 짧은 효과음, 폭발음과 같이 모델의 성능을 저하시키는 경우도 있다.

**2.6.2. 유튜브 추천 알고리즘의 문제점**

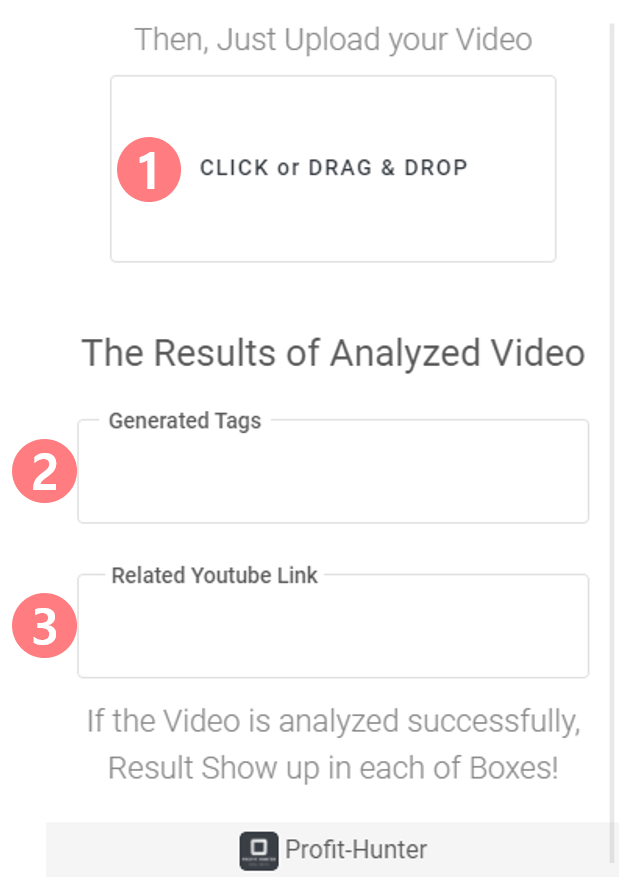
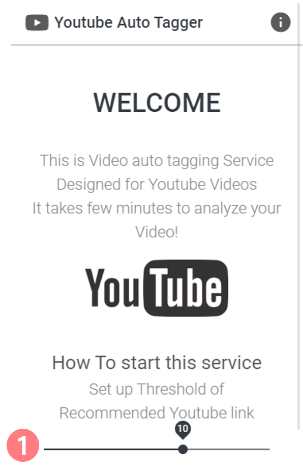
유튜브는 이용자 접속률을 위해 효율적인 영상 추천을 해야 할 필요가 있음에도 영상 내용 기반의 추천 알고리즘이 적용되지 않은 실정이다. 앞서 논한 바와 같이 현 유튜브는 사용자의 영상 감상 패턴이나 같은 크리에이터의 인기 영상 등 자의적인 기준에 따라 관련 영상을 추천하고 있는데, 영상과 직접적으로 관련이 없는 다른 영상들도 추천되는 경우가 간혹 발생한다. 그러므로 내용 상 연관된 동영상을 정확하게 추천한다면 동영상의 조회 수와 잔류 시간을 더욱 향상시킬 수 있을 것이라 예측할 수 있다.

**3. 프로젝트 내용**

**3.1. 서비스 소개**

이 서비스는 태그 생성 및 연관된 유튜브 영상 링크 추천의 본연의 기능을 중시하기 위해 UI/UX를 최대한 단순하고 직관적으로 구성하였다.

**3.1.1. 기본 화면**

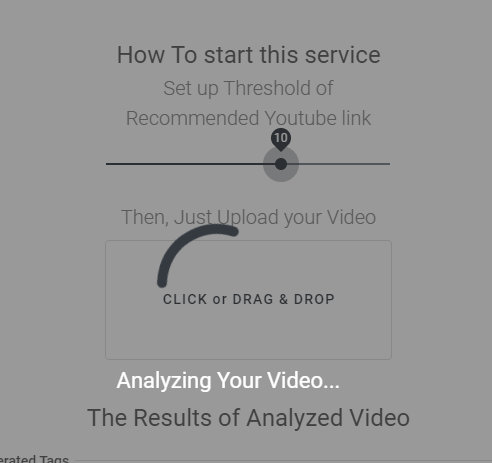
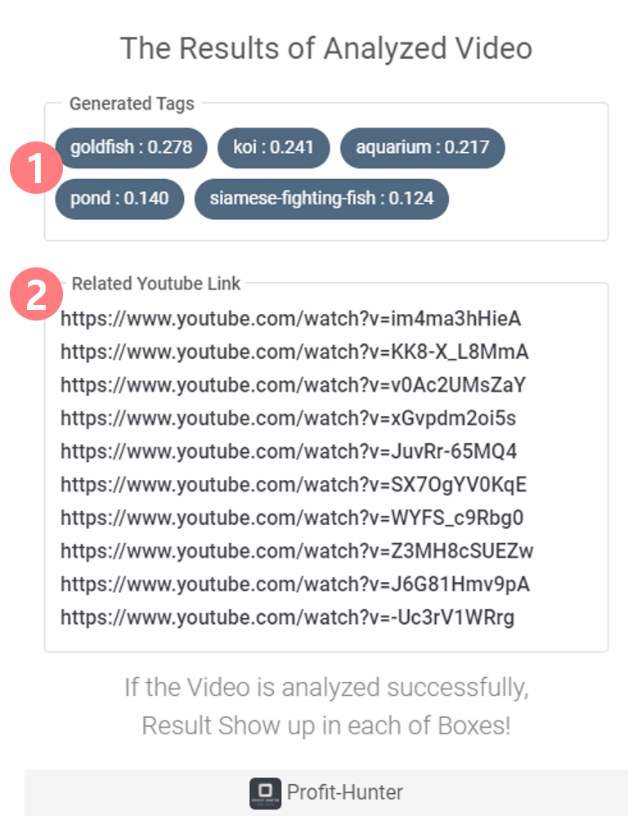


**[그림 3] 홈 화면1 [그림 4] 홈 화면2**

그림3은 사이트에 처음 접속했을 시 나오는 홈 화면이며. 서비스의 내용을 간략하게 소개하는 텍스트가 있다. 1번 항목에서 사용자는 슬라이드바를 이용해 추천받을 유튜브 영상 링크의 개수를 정할 수 있다.

그림 4는 그림 3에 이어지는 홈 화면이며, 마찬가지로 서비스의 소개 텍스트가 있다. 1번 항목은 사용자가 요청할 동영상을 입력하는 부분이며, 드래그 & 드랍 또는 버튼을 클릭하여 동영상을 입력할 수 있다. 2번 항목은 동영상 분석이 완료되었을 때, 분석을 요청한 동영상의 태그가 생성되는 부분이다. 3번 항목은 동영상 분석이 완료되었을 때, 분석을 요청한 동영상과 유사한 내용을 담은 유튜브의 영상 링크가 생성되는 부분이다.

**3.1.2. 로딩 화면과 결과 화면**

**[그림 5] 로딩 화면 [그림 6] 결과 화면**

그림 5는 그림 4의 1번 항목을 통하여, 사용자가 서비스에 동영상을 입력하여 동영상 분석을 요청했을 시 등장하는 로딩화면이다.

그림 6은 사용자가 요청한 동영상 분석이 종료되었을 때 나타나는 화면이다. 1번 항목에서 영상의 태그가 생성된 것을 확인할 수 있으며, 2번 항목에서 연관된 유튜브 영상 링크가 생성된 것을 확인할 수 있다.

**3.2. 요구사항**

**3.2.1 영상의 피쳐맵 추출**

영상의 내용을 분석하기 위해서는 해당 영상의 피쳐맵이 필요하다. 이를 위하여 내용기반 영상 추천 시스템에서는 구글의 Mediapipe[6]를 활용하여 영상의 피쳐맵을 추출하여 영상의 내용 분석에 활용한다.

**3.2.2 태그 추출을 위한 동영상 분류 학습 모델**

Mediapipe를 이용해 추출한 데이터는 사전에 학습시킨 classification 모델을 통해 태그를 추출한다. 본 프로젝트에서는 Youtube 8m 데이터셋으로 학습시킨 Dbof(Deep Bag of Frames) 모델을 이용했다. 학습한 모델은 Validation시 80% 전후의 정확도를 보였다.

**3.2.3. 비디오 벡터 생성**

다른 영상과 내용을 비교해 유사도를 비교하기 위해서는 영상의 내용을 나타낸 값을 가진 변수가 필요하다. 본 프로젝트에서는 비디오 벡터라는 이름의 벡터를 이용해 영상의 내용을 나타내고자 했으며, 학습시킨 classification 모델과 word vector 모델을 통해 이를 생성한다.

**3.2.4. 웹페이지를 위한 백엔드 API서버, 프론트엔드 구현**

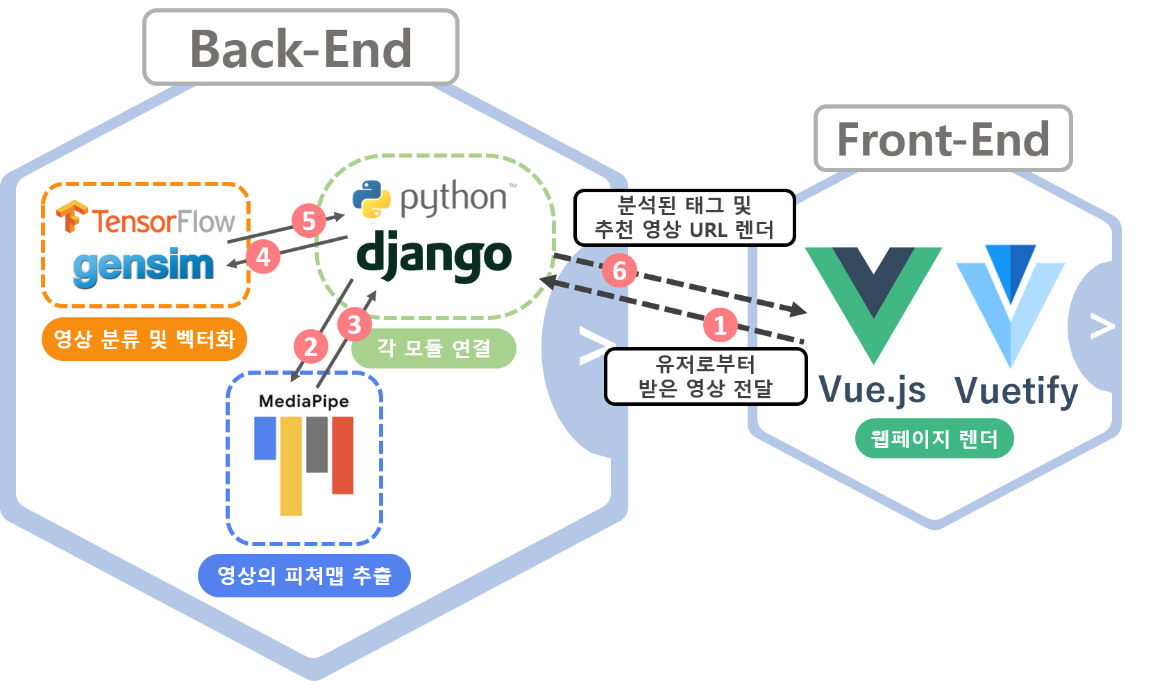
현재의 웹은 다양한 해상도를 가진 디바이스들이 접속하기 때문에, 웹 서비스를 만드는데 있어 반응형 웹 디자인[7]은 필수적으로 필요한 요소 중 하나이다. 우리가 사용할 프레임 워크인 Vue.js는 다른 프레임워크에 비해 가볍고, 성능 또한 좋기 때문에 현재 널리 이용 중인 반응형 웹 프레임워크이다. 그리고 Vuetify는 Google의 Material Design을 이용하기 쉽게 만들어진 Vue 전용 CSS 프레임워크이며, 우리는 이 두 가지 프레임워크를 결합하여 간단하고 직관적으로 프론트엔드를 구현할 것이다. API서버는 영상 분류 및 태그 추출에 python 라이브러리 중 하나인 tensorflow와 gensim이 사용되기 때문에 호환성을 위하여 python의 웹 서버 프레임워크인 django를 사용하였다.

**3.2.5. 각 모듈의 연결**

이 시스템을 구현하기 위해서 최종적으로 앞서 설명했던 각 항목들을 연결할 필요가 있다. API서버에서 각 항목의 연결을 진행하며, 영상의 피쳐맵 추출을 담당하는 Mediapipe와는 python의 라이브러리인 subprocess를 활용하여 외부 쉘 스크립트를 실행하는 방법을 사용하였다. 영상 분류 및 벡터화에 사용되는 모듈은 같은 python 이기 때문에 해당 모듈을 Import하여 API서버와 연결하였다.

**3.3. 시스템 설계**

**3.3.1. 서비스 Architecture**

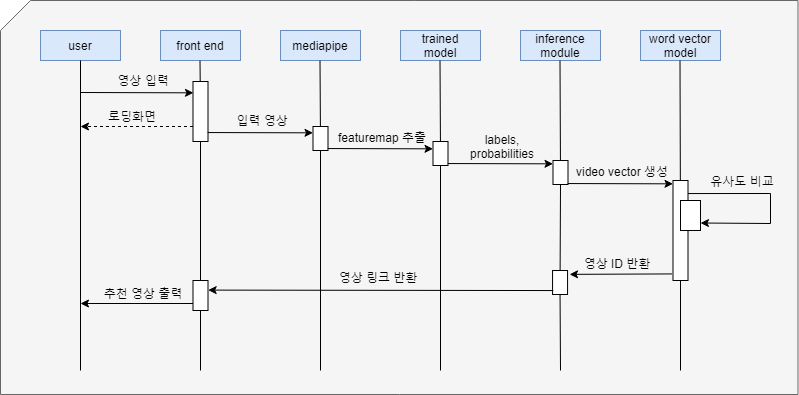


**[그림 7] 서비스Architecture**

최종적으로 우리가 구현할 서비스의 Architecture는 그림 7에서 보이는 바와 같다. 프론트엔드는 Vue.js와 Vuetify를 사용해 구성했고, 프론트엔드에서 받은 입력을 서버로 전송하고 결과를 출력하며 각 모듈들을 연결하기 위한 웹서버로 django를 사용했다. 또한 사용자로부터 받은 영상으로부터 피쳐맵을 추출하기 위해 google의 mediapipe모듈을 사용했으며, 영상의 labeling을 위한 딥러닝 모델은 tensorflow로 구현되었다. 비디오벡터 생성을 위해 word vector를 embedding 하기 위한 모듈로는 gensim을 사용했다. 위와같이 구성된 시스템의 작동 순서는 다음 절과 같다.

**3.3.2. Sequence & Activity Diagram**

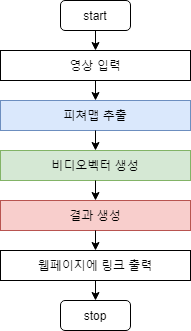
앞서 소개한 서비스 Architecture의 설명을 시퀀스 다이어그램으로 나타내면 다음과 같다.



**[그림 8] 서비스 Sequence diagram**

위 다이어그램에서 보이는 바와 같이 프론트엔드에서 사용자에게 동영상을 입력 받아 API서버로 보낸다. 사용자가 입력한 영상을 Mediapipe로 보내어 영상의 피쳐맵을 추출하여 그 결과를 trained model에 보낸다. Trained model에서 label 추출 후 비디오 벡터화가 완료되어 태그 및 추천 유튜브 영상 링크가 생성되면 서버는 해당 결과를 받아 태그와 링크를 웹에 전송한다

이를 액티비티 다이어그램으로 다음과 같이 나타낼 수 있는데, 아래 다섯개의 액티비티 중 피쳐맵 추출, 비디오벡터 생성, 결과 생성 액티비티를 자세히 다뤄보겠다.

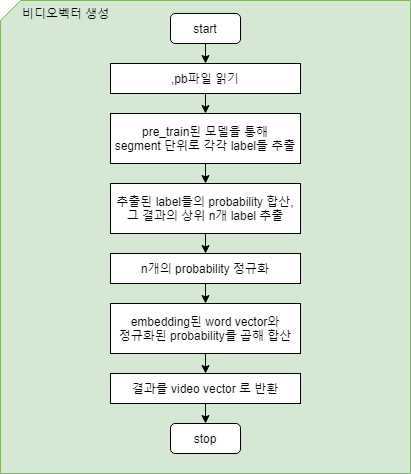


**[그림 9] 서비스 Activity diagram**



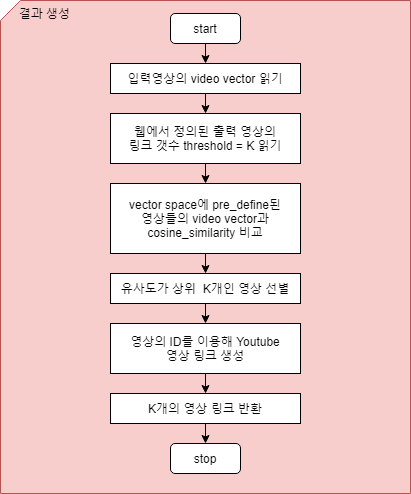
**[그림 10] 피쳐맵 추출 Activity diagram**

피쳐맵을 추출하는 단계는 다음의 과정을 따른다. 사용자로부터 입력 받은 영상을 mediapipe의 입력으로 사용한다. 영상은 사전에 mediapipe에 정의된 frame\_rate 값을 길이로 갖는 segment들로 나뉘어 각 segment마다 피쳐맵을 추출해 pb확장자 파일에 쓰게 된다. 본 프로젝트의 테스트 단계에서 사용된 frame\_rate는 5로 설정되었다. 이렇게 쓰기가 완료된 pb 확장자 파일은 다음 단계의 입력으로 사용된다.



**[그림 11] 비디오벡터 생성 Activity diagram**

비디오벡터를 생성하는 단계는 다음의 과정을 따른다. 이전 단계의 출력으로 생성된 피쳐맵을 읽어 그 내용을 사전에 학습시킨 모델의 입력으로 사용한다. 모델은 세그먼트 단위로 가장 가중치가 높은 몇 개의 label들과 각각의 probability를 출력하고, 전체 segment에서 출력된 결과들 중 같은 label을 갖는 probability를 합산한다. 이후 합산 결과 중 가장 큰 값을 가지는 n개의 label들과 probability값을 따로 추출해 정규화 한 후, embedding된 label들의 word vector와 해당 label의 정규화된 가중치를 곱해 그 결과들을 모두 더한다. 이렇게 생성된 벡터를 입력 영상의 비디오 벡터로써 다음 단계의 입력으로 사용한다.



**[그림 11] 결과 생성 Activity diagram**

결과 생성 단계에서는 사용자가 입력한 threshold를 읽어 출력할 영상 링크의 개수로 사용한다. 이전 단계에서 생성된 입력 영상의 비디오 벡터와 사전에 정의된 데이터셋 영상들의 비디오벡터 간의 유사도를 word2vec 모듈에서 지원하는 코사인 유사도 함수를 이용해 비교하고, 유사도가 가장 높은 영상들에 대해 Youtube 링크를 생성하고, 사용자가 설정한 threshold에 맞는 개수만큼 반환한다.

**4. 프로젝트 결과**

**4.1 연구 결과**

본 프로젝트의 주안점은 영상에 대한 정확한 태깅을 하는 것에서 출발하여, 입력 영상과 내용 상으로 가장 가까운 유튜브 영상들을 추천하는 것이다. 이를 위해 입력 영상을 받아 피쳐맵 형태로 분할하고 태그를 추출하며, 추출된 태그로 나온 연관 영상들을 유저에게 URL 형태로 추천하여 실제로 확인할 수 있게 해야 한다.

테스트는 Intel i9-9900X CPU 3.50GHz와 NVIDIA GeForce RTX 2080 Ti 환경에서 진행되었으며, 학습에는 yt8m의 1/20 scaled frame level data(78GB)가 사용되었다.

**4.1.1 단어 벡터 모델, 비디오 벡터 모델 사전 생성**

영상 추천에 앞서 yt8m의 영상들을 추천할 수 있는 형태로 가공해야 한다. 즉 yt8m 각 영상들에 대한 비디오 벡터를 미리 하나씩 생성해야 한다. 우선 yt8m segment dataset 중 validate 데이터에 대해, 각 영상마다 상위 5개의 태그와, 그 태그의 가중치를 도출한 solution 파일을 시스템 구현 전 미리 생성한다. 이는 3rd yt8m challenge에 사용된 세그먼트 태깅 코드를 영상 단위 태깅 코드로 변경하여 생성할 수 있다.



**[그림 12] yt8m 영상 별 상위 5개의 태그와 가중치**

그림 12는 각 yt8m 영상이 어떤 태그와 연관이 있는지 나타내고 있다. 예를 들어 XwFj라는 영상은 food 태그에 43.6%, cooking 태그에 32.3% 등 요리에 관련된 태그에 높은 연관성을 가지고 있으므로 요리에 관련된 영상인 것을 예측해볼 수 있다. yt8m dataset에서 각 영상은 임의의 네 자리의 이니셜이 부여되는데, 이니셜을 적절히 사용하면 원본 유튜브 영상에 접근할 수 있는 URL을 얻을 수 있으며 이를 추후 사용자 출력으로 사용한다.

비디오 벡터를 만들기 위해 태그를 단어 벡터화시킬 필요가 있는데, 이 때 각 태그에 해당하는 단어 벡터 모델이 필요하다. 이를 위해 사전에 위키피디아 텍스트를 이용하여 미리 단어 벡터 모델을 만들어 둔다. 단어 벡터 모델의 사이즈는 100으로 지정하였으며, 학습 성능을 위해 각 태그에 해당하는 위키피디아 설명 텍스트만 넣어 학습하였다. 단어 벡터 학습은 Gensim의 Word2Vec 패키지를 이용하였으며, 학습이 완료된 단어 벡터 모델은 시스템 내에 저장하고 추후 사용한다.

신문이(가) 표시된 사진

자동 생성된 설명

**[그림 13] 워드 벡터 학습에 사용된 태그별 설명**

그림 13은 각 태그별로 수집된 영문 위키피디아 설명을 보여 준다. 워드 벡터를 학습하기 위해 위키피디아의 설명 텍스트인 WikiDescription 컬럼을 입력 말뭉치로 사용한다. 이 때 설명 텍스트를 그대로 입력으로 사용하면 의미상 중요하지 않은 단어들이 과도하게 들어가거나, 복수의 단어로 이루어진 태그가 각 단어별로 따로 학습되거나, 학습이 부사 등에 편중되는 등 단어 벡터 모델이 잘못 학습될 가능성이 있다. 그러므로 정규화 원칙을 만들어 WikiDescription 항목을 올바른 형태로 학습되게 유도했다.

테이블이(가) 표시된 사진

자동 생성된 설명

**[그림 14] 텍스트 전처리 팀 내부 규칙**

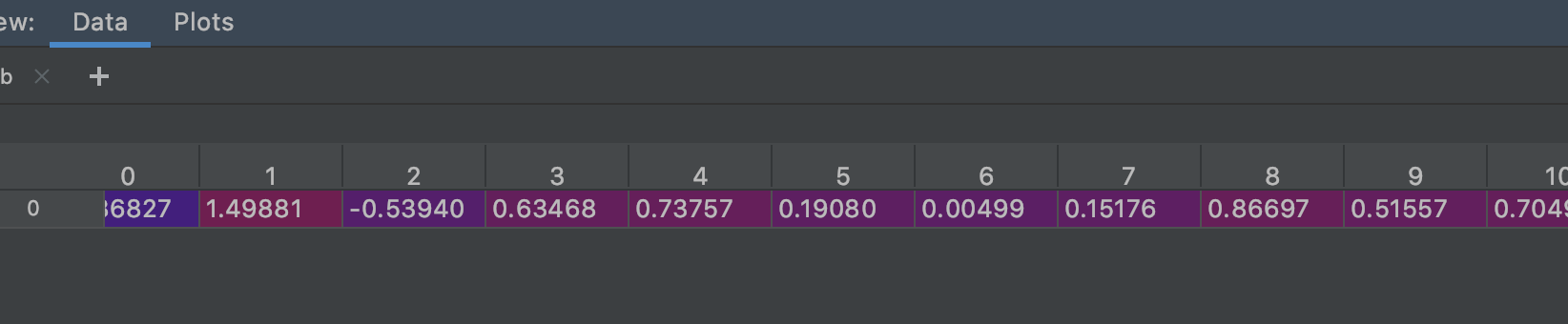
텍스트를 정규화하는 규칙 세 가지는 다음과 같다.

1. 영문 텍스트 전체를 소문자로 치환한다. 대문자가 섞인 단어와 소문자가 섞인 단어가 따로 학습되지 않도록 방지한다.
2. 태그 내에 소괄호로 부가 설명이 붙어 있는 부분을 삭제한다.
3. 태그에 들어간 띄어쓰기를 일괄적으로 하이픈(-)으로 치환한다. 태그에 들어간 별개의 단어가 따로 학습되지 않도록 방지한다.

위의 과정에 따라 단어 벡터를 생성하고, 생성된 단어 벡터를 기반으로 가중치를 반영하여 비디오 벡터를 생성한다. 비디오 벡터 모델은 단어 벡터 모델과 마찬가지로 Gensim 패키지의 Word2Vec 모델 형태로 저장한다.

**4.1.2 태그 및 영상 추천까지의 과정**

유저가 입력한 영상은 mediapipe를 통해 태그 추천 모델에 입력할 수 있는 피쳐 맵 형태로 변환된다. 피쳐 맵은 영상을 초 단위로 분해한 정보를 가지고 있는데, 1초당 RGB 값 1024개, AUDIO 값 128개의 데이터를 벡터 형태로 추출하게 된다.



**[그림 15] 피쳐 맵 추출 예시**

그림 12은 영상의 피쳐 맵 추출 예시이다. 피쳐 맵을 Youtube 8M Challenge의 태깅 모델에 입력하면 영상과 관련이 깊은 복수의 태그가 추출된다. 영상 태깅 모델에서는 1152 SIZE의 입력을 요구하는데, 1024개의 RGB 값과 128개의 AUDIO 값을 서로 연결하여 모델에 입력한다.

영상에서 추출된 단어 벡터를 종합하여 비디오 벡터를 생성하고 이를 다른 비디오 벡터와 유사도를 비교한 후 가장 가까운 영상을 추천한다. 비디오 벡터를 생성할 때에는 영상을 5초 단위의 세그먼트로 분할하여 각 세그먼트 별로 추출된 태그와 가중치를 합산한다. 본 프로젝트에서는 비디오 벡터의 사이즈를 100으로 설정하였다.

|  |  |
| --- | --- |
| Target Video | Video Vector |
| Goldfish | [ 0.0152, -0.0926, 0.0722, -0.0198, -0.0151, 0.0121, 0.0185 , … ] |
| Galaxy 10 | [ 0.0061, -0.0310, 0.0282, -0.0067, -0.0037, 0.0057, 0.0074, … ] |
| Beauty | [ 0.0025, -0.0063, 0.0067, -0.0018, -0.0013, 0.0022, 0.0008, … ] |
| Dog | [ 0.0098, -0.0671, 0.0545, -0.016, -0.0124, 0.0097, 0.0160, … ] |

**[표 1] 생성된 비디오 벡터 예시**

표 1은 영상을 입력했을 때 위의 과정을 토대로 생성된 비디오 벡터들이다. 기존 유튜브 영상들도 사전에 비디오 벡터 형태로 변환해 둔 다음, 생성된 비디오 벡터를 토대로 기존 유튜브 영상들과 비교하여 유튜브 영상 링크를 최종 도출한다.

**4.1.3 생성된 태그와 추천된 유튜브 영상 링크**

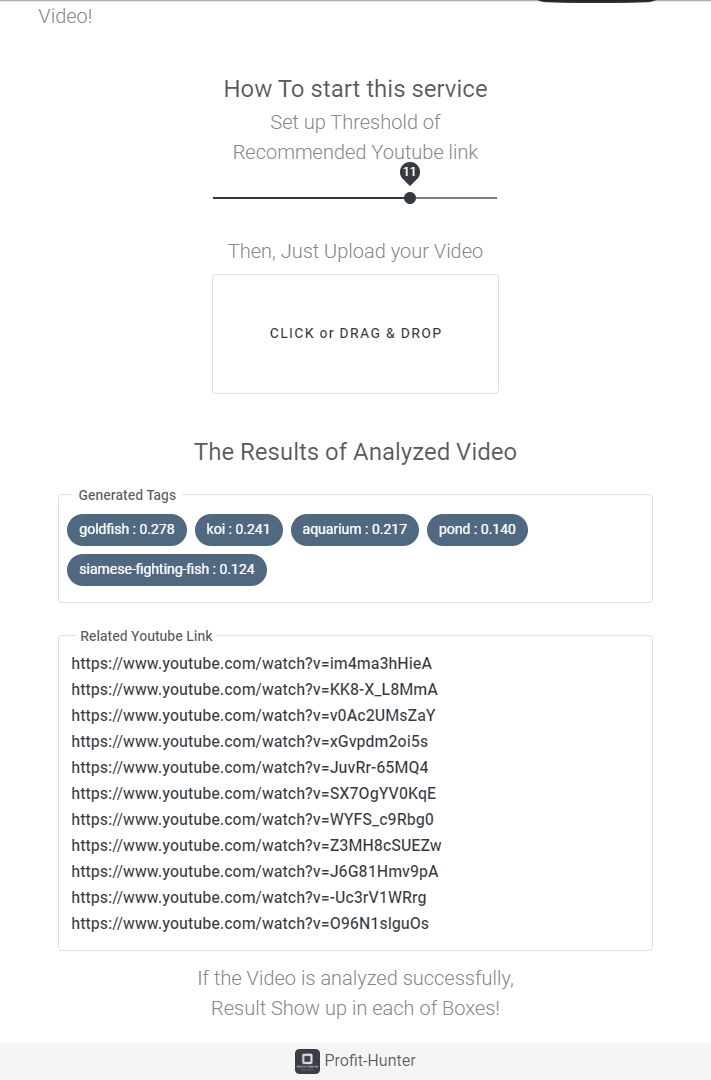
|  |  |  |
| --- | --- | --- |
| Input Video | Generated Tag | Youtube Url |
| Goldfish | goldfish: 0.278  koi: 0.241  aquarium: 0.217  pond: 0.140  siamese-fighting-fish: 0.124 | <https://www.youtube.com/watch?v=im4ma3hHieA>  <https://www.youtube.com/watch?v=KK8-X_L8MmA>  <https://www.youtube.com/watch?v=v0Ac2UMsZaY>  <https://www.youtube.com/watch?v=xGvpdm2oi5s>  <https://www.youtube.com/watch?v=JuvRr-65MQ4> |
| Galaxy 10 | mobile-phone: 0.302  sony-xperia: 0.215  samsung-galaxy-s4: 0.195  google-nexus: 0.155  samsung-galaxy-note2: 0.114 | <https://www.youtube.com/watch?v=H6YFe1p_I9c>  <https://www.youtube.com/watch?v=AMEkbpzo8LQ>  <https://www.youtube.com/watch?v=r372pwnghTk>  <https://www.youtube.com/watch?v=0xHdp9EVk_w>  <https://www.youtube.com/watch?v=IE-FdDsJcPw> |
| Beauty | eye-shadow: 0.254  eyelash-extensions: 0.248  lipstick: 0.222  eye-liner: 0.154  brush: 0.122 | <https://www.youtube.com/watch?v=kqXyMfkaYOo>  <https://www.youtube.com/watch?v=OrjT0-olLBQ>  <https://www.youtube.com/watch?v=Nu5xlVXi3dQ>  <https://www.youtube.com/watch?v=mIycwt01s9Q>  <https://www.youtube.com/watch?v=R38y6UjXSCE> |
| Dog | Border-collie: 0.288  collie: 0.280  golden-retriever: 0.202  canoe: 0.146  laptop: 0.084 | <https://www.youtube.com/watch?v=cCy2RiWpop4>  <https://www.youtube.com/watch?v=jQo4LZqYCuY>  <https://www.youtube.com/watch?v=KzFLVT92BPM>  <https://www.youtube.com/watch?v=YHxGg6il8So>  <https://www.youtube.com/watch?v=b7NV4MUgxak> |

**[표 2] 입력 영상으로 생성된 태그와 유튜브 영상링크**

표 2 는 각 영상을 내용 기반 영상 추천 시스템에 입력하였을 때 생성되는 태그와 추천되는 유튜브 영상 링크들을 보여준다.

**4.1.3 웹페이지에서 보이는 태그와 유튜브 링크**

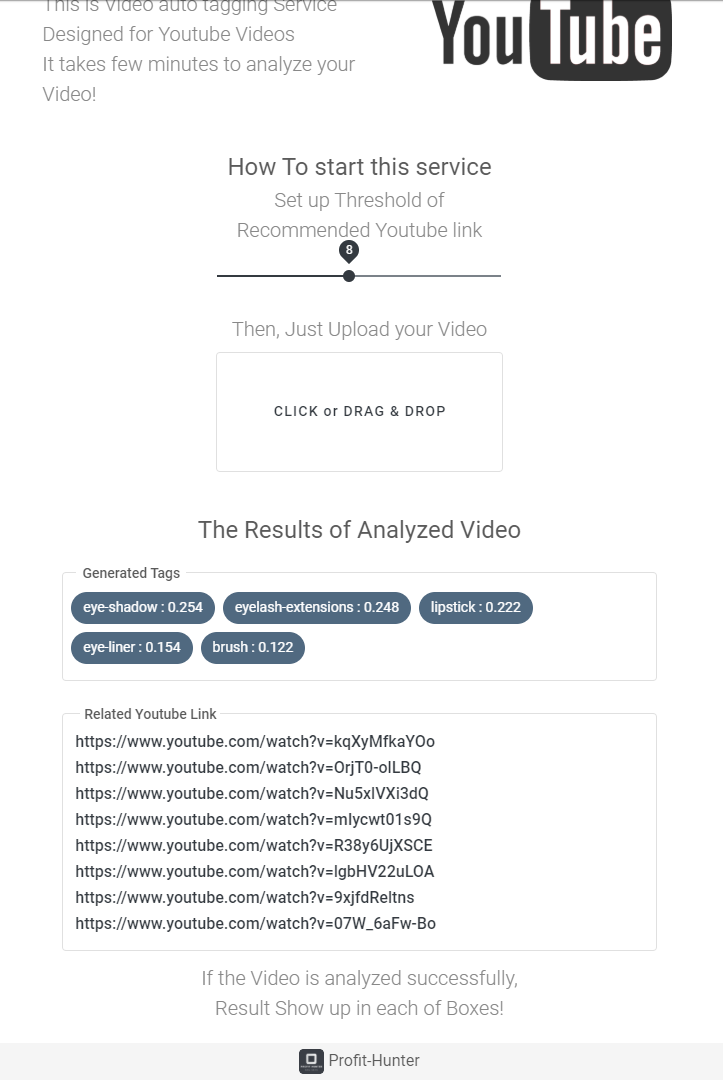
아래의 그림들은 시스템에 입력영상을 넣었을 때, 웹페이지에서 보이는 화면들이다. 그림 3의 1번에서 보이듯이, 사용자가 추천될 유튜브 링크 개수를 지정할 수 있음을 보여주기 위해 각 영상의 threshold를 다르게 설정하였다.



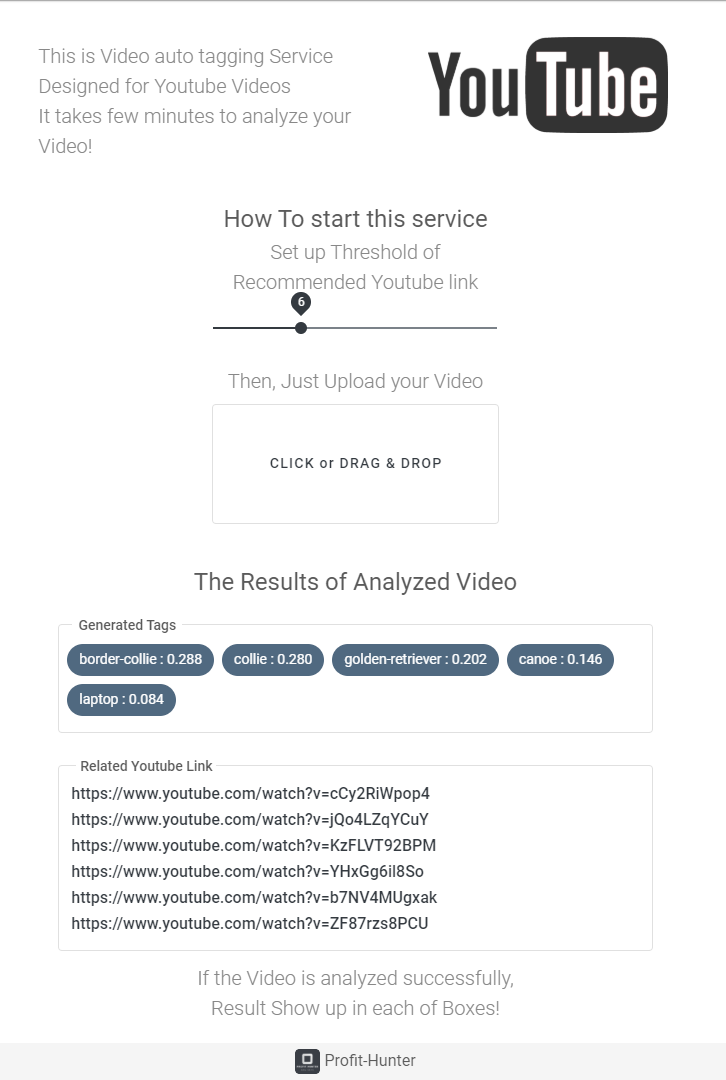
**[그림 16] Goldfish를 입력영상으로 하였을 때, 서비스의 출력 결과**



**[그림 17] Galaxy10을 입력영상으로 하였을 때, 서비스의 출력 결과**



**[그림 18] Beauty를 입력영상으로 하였을 때, 서비스의 출력 결과**



**[그림 19] Dog를 입력영상으로 하였을 때, 서비스의 출력 결과**

**4.2 성능 평가**

시스템의 성능평가는 첫 번째로 입력 영상과 생성된 태그의 연관성, 두 번째로 입력 영상과 추천된 유튜브 영상과의 연관성으로 진행한다. 4.2.1과 4.2.2에서 제시하는 입력 영상과 생성된 태그, 그리고 추천된 유튜브 영상의 연관성을 보았을 때, 의미 있는 결과를 도출함을 알 수 있었다.

**4.2.1 입력 영상과 생성된 태그의 연관성**

|  |  |  |
| --- | --- | --- |
| Input Video | Input Video Thumbnail | Generated Tag |
| Goldfish |  | goldfish: 0.278  koi: 0.241  aquarium: 0.217  pond: 0.140  siamese-fighting-fish: 0.124 |
| Galaxy 10 |  | mobile-phone: 0.302  sony-xperia: 0.215  samsung-galaxy-s4: 0.195  google-nexus: 0.155  samsung-galaxy-note2: 0.114 |
| Beauty |  | eye-shadow: 0.254  eyelash-extensions: 0.248  lipstick: 0.222  eye-liner: 0.154  brush: 0.122 |
| Dog |  | Border-collie: 0.288  collie: 0.280  golden-retriever: 0.202  canoe: 0.146  laptop: 0.084 |

**[표 3] 입력 영상과 생성된 태그**

표 3은 사용자가 입력한 영상의 썸네일, 그리고 입력 영상으로부터 생성된 태그들이다. Goldfish 영상부터 보자면, 영상의 주요 내용인 goldfish, aquarium이 태그로 생성되었음을 알 수 있고, goldfish와 생김새가 유사한 동물인 koi, simese fighting fish가 태그로 생성되었음을 볼 수 있다. Galaxy 10, Beauty, Dog의 입력 영상의 썸네일과 생성된 태그 또한 Goldfish 입력 영상과 마찬가지로 영상의 내용에 맞는 태그가 생성되었음을 알 수 있다. 이를 통하여 내용 기반 영상 추천 시스템이 입력한 영상과 높은 연관성을 가진 태그가 생성하는 것을 확인할 수 있다.

**4.2.2 입력 영상과 추천된 유튜브 영상과의 연관성**

|  |  |  |
| --- | --- | --- |
| Input Video | Input Video Thumbnail | Youtube Video Thumbnail |
| Goldfish | Beauty of Goldfish - HD 1080p | Elephant Ear Betta or Dumbo Fighter and Horse Faced Loach |
| Galaxy 10 | Samsung Galaxy S10 5G Review: A Great 4G Phone [4K] | AT&T Nokia Lumia 1520 Hands-on |
| Beauty | 눈꼬리 내리는 성형후 화장한 모습 | GLAM BRIDAL MAKEUP BY MISBAH B MUA |
| Dog | 폭염에 아이스풀장을 만난 썰매견 | 2012 Para 2013 Fogos na lage do Beto Escadaria da Nike Borel |

**[표 4] 입력 영상과 추천된 유튜브 영상**

표 4은 사용자가 입력한 영상의 썸네일, 그리고 추천된 유튜브 영상의 썸네일들이다. Goldfish에서는 입력 영상의 썸네일과 추천된 유튜브 영상은 어항에서 움직이는 물고기라는 공통적인 내용이 담긴 것을 볼 수 있으며, Galaxy 10 에서는 입력 영상의 썸네일과 입력 영상으로부터 추천된 유튜브 영상은 휴대폰 리뷰라는 공통적인 내용을 담긴 것을 확인할 수 있다. 그리고 Beauty에서 입력 영상의 썸네일과 입력 영상으로부터 추천된 유튜브 영상은 화장하는 모습이라는 공통적인 내용을 담고 있었다. 그러나 Dog에서는 입력 영상의 내용과 추천된 유튜브 영상의 내용이 다른 것을 알 수 있다. 위의 예시 이외에도, 표 2의 추천된 유튜브 영상 링크를 확인해본다면, 전체적으로 보았을 때 입력 영상과 비슷한 내용이 담긴 유튜브 영상이 추천됨을 알 수 있었다.

추천된 영상 중에서 입력 영상과 무관한 영상도 있었는데, 이는 크게 두 가지의 문제로 해석해볼 수 있다. 첫 번째는 모델 학습에 사용된 데이터셋 크기가 작다는 것이다. 본 프로젝트에서는 1/20 scaled의 yt8m 데이터 셋을 사용하였는데, 전체 yt8m 데이터를 사용하지 않아서 데이터가 적은 특정 클래스의 내용을 제대로 반영하지 못했을 가능성이 있다. 두 번째는 태그 벡터가 부정확하게 학습되었다는 것이다. 위키피디아의 설명 텍스트만을 토대로 태그 벡터를 학습시켰는데, 말뭉치의 양이 적다 보니 의미상 연관된 태그들이 벡터 공간에 충분히 밀집되지 못했을 수 있다. 위의 두 가지 문제점을 개선한다면 입력 영상과의 연관성이 높은 유튜브 영상이 추천될 것으로 기대된다.

**4.3. 역할 분담**

이현규 : Word Vector 학습, Video Vector 생성

윤영빈 : 영상 태깅, 태그별 가중치 부여

윤준현 : 프론트 화면 구성, 영상 피쳐맵 추출

이태현 : 백엔드 API서버 구현 및 각 모듈 연결

**5. 결론**

**5.1 기대효과**

2020년 현재에도 유튜브 이용률은 꾸준히 증가하고 있다. 2019년 초등학생 대상 희망직업 조사 통계에서 크리에이터(유튜버)가 운동선수, 교사에 이어 3위[8]를 차지할 정도로 유튜브는 우리 삶의 중요한 요소가 되었다.

우리가 개발한 이 웹 서비스, ‘비디오 벡터를 이용한 내용 기반 영상 추천 시스템’은 이러한 사회 속에서 새롭게 등장한 유튜버라는 직업에 필요할 것이다. 업로드한 영상의 내용을 정확하게 분석하고, 유튜브 검색 기능과 유튜브 알고리즘을 위한 적절한 태그를 달아준다면 동영상의 조회 수는 이전보다 더 빠른 속도로 늘어날 것이다.

유용한 것은 유튜버 뿐만이 아니다. 시청자들에게 자신의 취향에 맞는 영상을 더 정확하게 추천해준다는 것은 좋아하는 주제의 영상을 접할 수 있는 시청자들에게도 이점으로 작용할 것이다. 또한 영상 태그와 내용 기반으로 분석한 관련 영상 유튜브 링크를 유튜브 상용 서비스에 적용할 수 있다면 유튜브 이용률과 만족도가 현재보다 상승할 것이라고 생각한다.

**5.2 추후 연구 방향**

향후 연구로는 비디오 벡터를 구하기 위한 전처리 과정에서의 시간 단축 방법을 모색하고자 한다. 가령, 영상 입력 단계 프론트엔드 UI에서 프로그램 사용자에게 자신이 중요하다고 생각하는 동영상 구간을 함께 입력 받아 해당 구간의 프레임을 집중적으로 분석시킬 수 있다. 이렇듯 분석을 위하여 조금 더 세밀한 조건을 도입한다면, 피쳐맵 추출 시간과 비디오 벡터 생성 시간을 절약시킬 뿐만 아니라, 더 정확한 유사도 비교를 행할 수 있을 것이다.

또 텍스트 기반이 아닌 영상 분석 과정 자체에서 태그 벡터 모델을 생성하는 방안도 고려하고 있다. 최근 Youtube 8M Challenge에서 활발하게 연구되고 있는 모델 중에는 영상을 학습시키면서 동시에 태그 벡터를 생성하는 모델도 등장하고 있다. 영상 정보에 기반하여 태그 벡터를 생성할 수 있다면 비디오 벡터를 더 정확하게 생성할 수 있을 것이라 예상한다.

그리고 연관성이 높은 태그와 영상을 추천하기 위해 데이터 학습에 사용된 딥러닝 모델을 변경하는 방안이 있을 것이다. 본 프로젝트에서는 빠르게 결과를 얻기 위해 비교적 간단한 구조의 Logistic regression모델을 사용하였다. 이 시스템에 Youtube 8M Challenge에서 높은 성능을 보인 CCLR[9]과 같은 모델을 적용한다면 더 정확하고, 연관성이 높은 결과가 도출될 것이라 생각한다.

**6. 참고 문헌**

[1] CNN, Convolutional Neural Network 요약 (<http://taewan.kim/post/cnn/>)

[2] LSTM(RNN) 소개 (<https://brunch.co.kr/@chris-song/9>)

[3] Tomas Mikolov, Kai Chen, Gregory S. Corrado, Jeffrey A. Dean, "Computing numeric representations of words in a high-dimensional space", Google, 2013.

[4] Youtube-8M, “Youtube-8M Large-Scale Video Understanding Challenge”, Google, 2019.

[5] 오세욱, 송해엽, "유튜브 추천 알고리즘과 저널리즘", 한국언론진흥재단, 2019.

[6] Mediapipe (<https://github.com/google/mediapipe/>)

[7] 반응형 웹 (<https://www.samsungsds.com/global/ko/support/insights/Responsive_web_1.html>)

[8] 2019년 초등학생 장래희망 통계 (<https://news.joins.com/article/23653857>)

[9] CCRL (<https://arxiv.org/pdf/1911.08548.pdf>)