BoB 프로젝트 일지

피해 미디어 탐지 및 삭제 요청 서비스 개발 보고서

Sex Offense Video Tracking and Reporting Service Development Report

2020년 08월 04일 ~ 2020년 12월 18일

영산대학교
사이버경찰학과
김 태 룡

목차

표 목 차

그 림 목 차

제 1 장 서론

1.1 개발 배경 및 목적

2020년도 당시 이슈였던 N번방 사건으로 인해 미디어 성범죄가 수면 위로 떠오르게 되었습니다. 당시 성범죄 미디어가 텔레그램 및 디스코드, 카카오톡 등 메신저를 통해 유포되었으며, 유포된 미디어들은 다시 음란물 스트리밍 사이트에서 공공연히 유포되고 있었습니다.

때마침 필자는 [유해사이트 탐색 크롤러] 및 [범죄 통계 분석 프로그램]을 제작했었고, 두 프로그램을 합쳐 인터넷상에 유포된 성범죄, 도박, 마약류, 저작권 침해 등에 이용된 미디어를 수집하고, 범죄 정보를 시각화함으로 써 미디어물 포렌식 수사에 도움을 줄 수 있는 [가상 네트워크 범죄지도]를 2학기 졸업 작품으로 개발 할 예정이었습니다.

하지만 KITRI 주관 Best of the Best(차세대 보안리더 양성 프로그램)에 최종 합격되어 7월부터 교외 강의를 듣게 되는 바람에 2가지 이상의 프로그램이 혼재된 복잡한 [가상 네트워크 범죄지도]를 6개월 이내에 제작하기 힘들 것임을 판단하고, 해당 프로그램의 코어 기능인 '미디어 유사도 비교' 알고리즘과, 많은 범죄 중 현재 이슈로 떠오른 '성범죄'만을 이용한 '성범죄 피해 미디어 자동 추적/신고 프로그램'을 BoB 프로젝트 주제로 선정하고, 4개월 안에 만들기로 하였습니다.

부속 프로그램이긴 하나, 수사대뿐만 아니라 피해자를 위한 프로그램으로써 기존의 수사 프로그램 및 디지털장의사 업체와는 다르게 제작하는 것이 목적이었습니다.

때문에 피해자를 위해 신고양식 작성 란에 인적사항 기입이 필요 없고, 관리자 DB에 사진/동영상을 저장하지 않아 2차 가해 및 해킹 걱정이 없는 프로그램을 제작하기로 하였습니다. 수사관을 위한 기능으로는 현재 미디어 수사에 쓰이고 있는 단순한 파일 Hash 비교 프로그램과는 차별성 있는 알고리즘과, 시각화 기능을 제공하는 분석 프로그램을 제작하기로 하였습니다.

1.2 보고서 내용 및 구성

본 문서는 프로젝트의 진행 과정 및 결과를 중심으로 작성되었습니다. 2장에서는 프로젝트 초기에 진행 되었던 계획 수립 및 팀원 등에 대하여 다룰 것이며, 3 ~ 6장에서는 각 프로그램 및 서비스에 대한 개발/연구과정을 다룰 것이며, 7장에서는 산출물 및 결과에 대한 주제를 다룰 것입니다.

[표 1-1] 보고서 흐름도

제 2 장 프로젝트 계획 수립

2.1 프로젝트 주제 선정

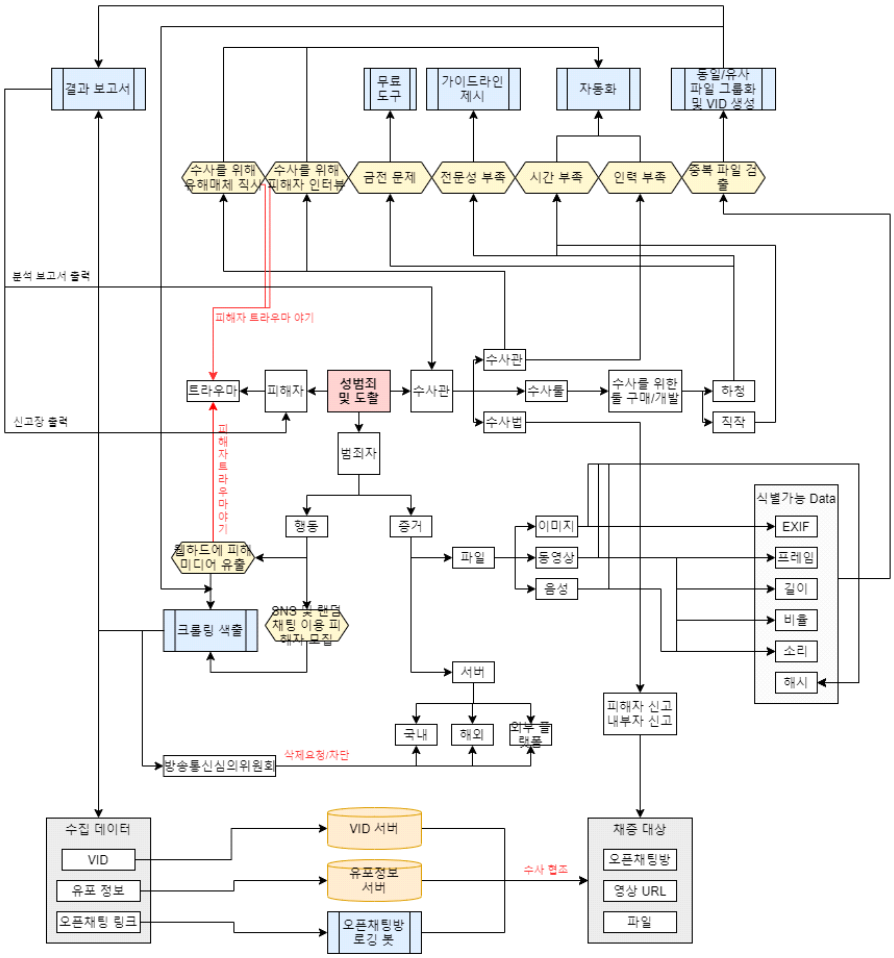
프로젝트 주제를 선정하는 데는 오랜 시간이 걸리지 않았습니다. 당시 필자는 [유해사이트 탐색 크롤러][1] 및 [범죄 통계 분석 프로그램][2]을 제작했었고, 두 프로그램을 합쳐 인터넷상에 유포된 성범죄, 도박, 마약류, 저작권 침해 등에 이용된 미디어를 수집하고, 범죄 정보를 시각화 함으로 써 미디어물 포렌식 수사에 도움을 줄 수 있는 [가상 네트워크 범죄지도]를 2학기 졸업 작품으로 개발 할 예정이었기 때문입니다.

하지만 2020년도 당시 N번방 사건으로 인해 미디어 성범죄가 수면 위로 떠오르게 되었으며, BoB 프로젝트는 기간이 매우 짧아 가상 네트워크 범죄지도를 만들기에는 턱없이 부족하였기에, 해당 프로그램의 코어 기능인 '미디어 유사도 비교' 알고리즘과, 많은 범죄 중 현재 이슈로 떠오른 '성범죄'만을 이용한 '성범죄 피해 미디어 자동 추적/신고 프로그램'을 주제로 정하였습니다.

부속 프로그램이긴 하나, 수사대뿐만 아니라 피해자를 위한 프로그램으로써 기존의 수사 프로그램 및 디지털장의사 업체와는 다르게 제작하는 것이 목적이기에, 피해자를 위해서는 익명성과 2차 피해를 방지하는 측면으로 생각 해 보았으며, 수사대를 위해서는 현재 미디어 수사에 쓰이고 있는 단순한 파일 Hash 비교 프로그램과는 차별성 있는 알고리즘을 위한 독자적인 DB 구성을 고안하여 시각화 기능을 제공하는 분석 프로그램을 제작하기로 하였습니다.

---

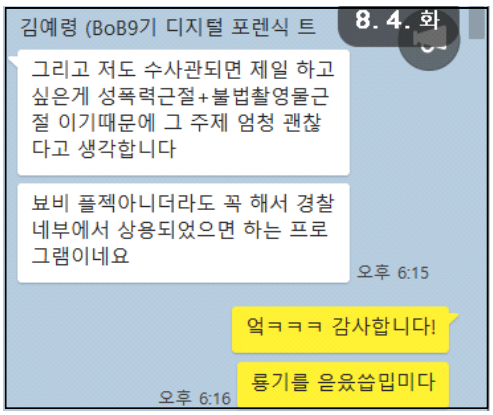
1) NuriCHAIN 개발 문서 : https://cafe.naver.com/goldbigdragon/85753
2) SCPTP 개발 문서 : https://cafe.naver.com/goldbigdragon/89133



[그림 2-1] 필자가 고안했던 초기 프로젝트 기획

2.2 팀원 구성 및 역할 분담

프로젝트 시작에 앞서, BoB 특성상 반드시 5인 이상의 팀원이 모여야만 프로젝트를 진행할 수 있었습니다. 다행히 프로젝트 기간 2달 전, 회식 자리에서 만난 김예령 멘티님께 필자의 BoB 자기소개서 및 각종 자료를 보여주며 프로젝트에 대한 이야기를 나누었더니, 응원의 한 마디와 함께 같이 프로젝트를 할 의사가 있음을 밝혀 주셨으며, 프로젝트 팀원 모집 기간 당일 팀 프로젝트를 도와주실 유현, 김종민 멘토님과 4명의 팀원을 순식간에 모아 오시는 놀라운 사회성을 보여주셨습니다.



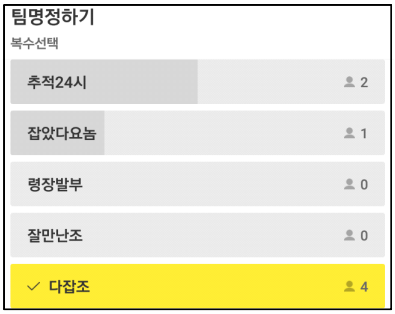
[그림 2-2] 필자에게 용기가 된 한 마디

전체적인 프로젝트 관리자인 PM(Project Manager)의 경우, 기획을 했기에 프로젝트에 대한 이해도가 높은 필자가 맡는 것이 좋을 것이라 예령님께서 말씀하셨지만, 뛰어난 사회성과 발표력, 인성 및 문서 정리 능력을 가진 예령 멘티가 PM을 맡고, 개발에 특화된 필자가 기획에 맞게 개발 해 나가는 것이 가장 이상적이라 판단되었습니다. 때문에 프로젝트가 시작되기 전, 많은 대화를 나누었으며, 오랜 대화 끝에 영광스럽고 감사하게도 김예령 멘티님께서 PM을 맡아 주시게 되었습니다.

직책	이름	직책	이름
주 멘토	유현	부 멘토	김종민
PL	김성민	PL	김다솜
PM	김예령	팀원	김태룡
팀원	이소민	팀원	전유민
팀원	조서연	팀원	허원무

[표 2-1] 프로젝트 팀원 구성 표

여담이지만 프로젝트 기간 동안 예령님께서는 거짓말 같이 뛰어난 업무능력을 보여주셨으며, 마지막 3차 발표까지 팀을 성공적으로 이끌어 주셨습니다. 더하여 팀명은 카카오톡 투표로 진행하였는데, 필자가 장난스럽게 집어넣었던 '다잡조'가 선정되었습니다.



[그림 2-3] 팀명 투표

2.3 프로젝트의 필요성과 차별성

프로젝트 시작 전부터 팀원 및 주제가 갖추어진 덕분에 다잡조는 사전 조사를 빨리 시작할 수 있게 되었습니다. 우선 디지털 성범죄 동향에 대한 조사를 수행하였으며, 이후 기존 디지털장의사 업체에 대한 인식과, 수사에 대한 불만을 주제로 설문조사를 진행하였습니다.

2.3.1 프로젝트 필요성

① 디지털 성범죄 동향

〈디지털성범죄정보 심의 및 시정요구 현황〉

(기간 : 2014. 1. 1. ~ 2018. 7. 31.)

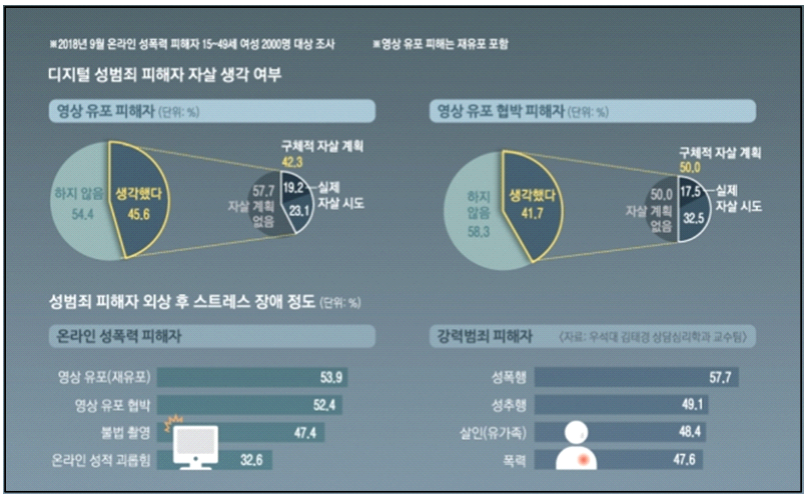
구 분	심의	시정요구		
		계	삭제	접속차단
2014년	1,807	1,665	182	1,483
2015년	3,768	3,636	63	3,573
2016년	7,356	7,325	10	7,315
2017년(~6.12.)	2,977	2,977	1	2,976
2018년(~7.31.)	7,648	7,567	106	7,461

※자료출처: 방송통신심의위원회

[그림 2-4] 디지털성범죄정보 심의 및 시정요구 현황

집계 표에서 디지털성범죄정보에 대한 심의 및 시정 요구 건수가 14년부터 꾸준히 증가하여 18년도에는 심의 건수가 대폭 늘어난

모습을 확인 할 수 있었습니다.



[그림 2-5] 디지털성범죄 피해자 대상 조사-한국여성정책연구원

또한 디지털성범죄로 인해 자살을 생각 한 피해자는 40%가 넘었으며, 그 중 실제 자살 시도를 진행 한 피해자는 17%나 되었습니다.

디지털 성범죄가 피해자에게 직접적인 피해를 끼치고 있으며, 그 추세 또한 점점 늘어나고 있는 자료는, 디지털성범죄 피해를 줄이기 위한 프로젝트가 충분히 가치 있음을 증명하고 있었습니다.

② 사회적 필요성

유현 멘토님께서 여성청소년수사팀과 사이버성폭력수사팀과의 미팅을 잡아 주신 덕에 직접 인터뷰 할 기회를 가지게 되었으며, 아래와 같은 문제점을 알게 되었습니다.

[여성청소년수사팀]

- 신고 된 사이트에 대해서만 삭제/차단 처리 된다.
- 피해자는 미디어에 대한 완전한 삭제를 원한다.

[사이버성폭력수사팀]

- 담당 수사관에 대한 정신적 충격 보호 장치의 부재.
- 오픈채팅방은 첩보를 통해 수사가 진행되지만, 첩보가 적음.
- Hash검사 프로그램[3]을 사용하지만 변조 영상이 많아 어려움.

다행히 프로젝트 기획 시 필자가 고려했던 부분들로, 크롤러를 이용한 지속적인 유사 영상 및 오픈 채팅방 탐지/신고 처리와, 영상에 대한 변조가 일어나도 Hash로만 판별하지 않기에 유사도 검사가 가능한 독자적인 VID정책, 자동 분석 프로그램을 통한 담당 수사관의 정신적 충격 보호 및 2차 가해 방지 등 기획부터 사회적 필요성을 잘 담아낸 프로젝트임을 알 수 있었습니다.

③ 유사서비스의 문제점

[디지털 장의사]

- 단기성 : 의뢰 건에 대한 지속적인 추적을 지원하지 않음. 또한 의뢰했던 사건이 다시 터져도 재 결재해야 추적 함.
- 개인정보 수집 : 의뢰를 위해서는 개인정보를 제공해야 함.
- 자료 저장 : 의뢰 자료는 추적을 위해 업체 PC에 소장됨.
- 유포 사이트와 결탁하여 사회적으로 논란된 적이 있음.[4]

[시판 중인 미디어 유사도 비교 프로그램/서비스]

- 높은 가격 : 일반인에게 부담되는 가격
- 썸네일 비교형 프로그램/서비스 : 영상에 대한 썸네일 '한 장'만을 비교하여 유사도에 대한 신뢰도가 떨어짐.
- 미디어 DNA 추적형 프로그램/서비스 : 피해 미디어 자료를 서비스 업체가 소장하고 있어야만 다른 미디어 자료에 대한 비교 분석이 가능함.

---

3) 각종 유해 영상에 대한 Hash DB를 모아 두고, 영상의 Hash값이 해당 DB 안에 있는지 확인하여 유해 영상인지 판별하는 프로그램을 사용하고 있다고 하셨다.

4) 중앙일보2020.05.07김정민 기자 : https://news.joins.com/article/23770589

[시판 중인 피해 미디어 추적/삭제 서비스]

- 과장 광고 : 하루 안에 유포 미디어를 삭제 한다고 광고함.[5)]
- 개인정보 수집 : 의뢰를 위해서는 개인정보를 제공해야 함.
- 자료 저장 : 의뢰 자료는 추적을 위해 업체 PC에 소장됨.

2.3.2 프로젝트 차별성

프로젝트의 필요성과 디지털성범죄의 동향, 유사 서비스의 문제점을 종합하여 다잡조 프로젝트만의 차별성을 찾아보았습니다.

① 익명성

- 개인정보 불필요 : 개인정보 입력 없이 신고 대상 URL 입력 혹은 미디어 파일 제출만으로 신고 접수.
- 미디어자료 폐기 : 채증 된 미디어 파일로부터 원본 미디어를 도출 할 수 없는 식별 데이터 값만 추출한 후, 원본 미디어 파일 영구 삭제. (관리자 및 해커가 데이터를 통해 원본 미디어를 추측/복구 할 수 없음)

② 지속성

- 신고 된 유해 미디어에 대한 지속적인 탐색 및 방송통신심의위원회 신고.

③ 자동화

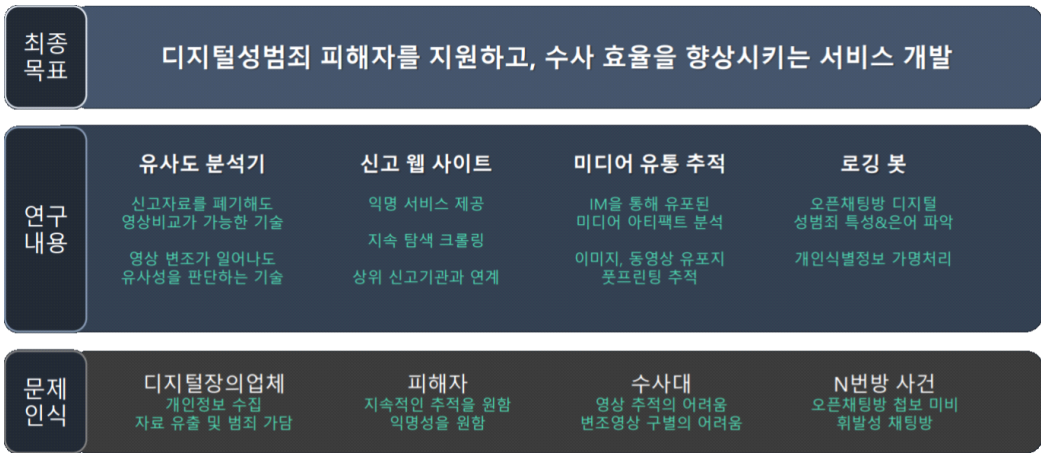
- 유사도 분석 프로그램 사용 시, 수사관이 직접 미디어 파일을 열람하지 않아도 되도록 유사 미디어간 자동 그룹화기능 제공.
- 유해 미디어 크롤러를 통한 유해 미디어 자동 수집 및 분석.

④ 민감 자료 배제

- 미디어 파일을 소장하지 않고, 미디어 파일로부터 추출된 식별 데이터만을 저장하여 민감 자료 배제.

---

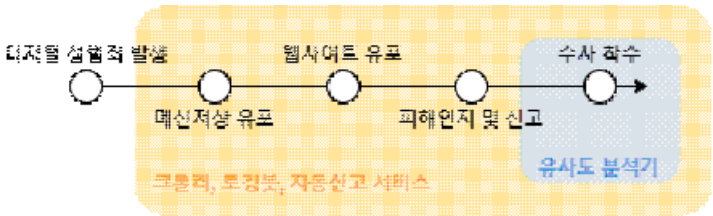
5) 유포 사이트 운영자 혹은 운영자와 결탁되어 있지 않는 이상 하루 안에 '삭제'하는 것은 불가능하며, 해외 사이트의 경우 더욱 어렵다.



[그림 2-6] 프로젝트 대 주제

2.4 프로젝트 전체 구성

초기 프로젝트 범위는 웹 사이트 유포 단계에서부터 수사 착수 단계까지였으나, 김종민 멘토님께서 추가로 메신저 상 유포 단계에서 미디어 유통 과정을 추적할 수 있다면 더 좋을 것 같다는 조언을 해 주셨으며, 이에 미디어 유통과정에서 변화하는 미디어 파일 아티팩트를 찾아 어떤 메신저를 통하여 유포되었는지에 대한 단서를 찾는 연구도 프로젝트에 포함되었습니다.

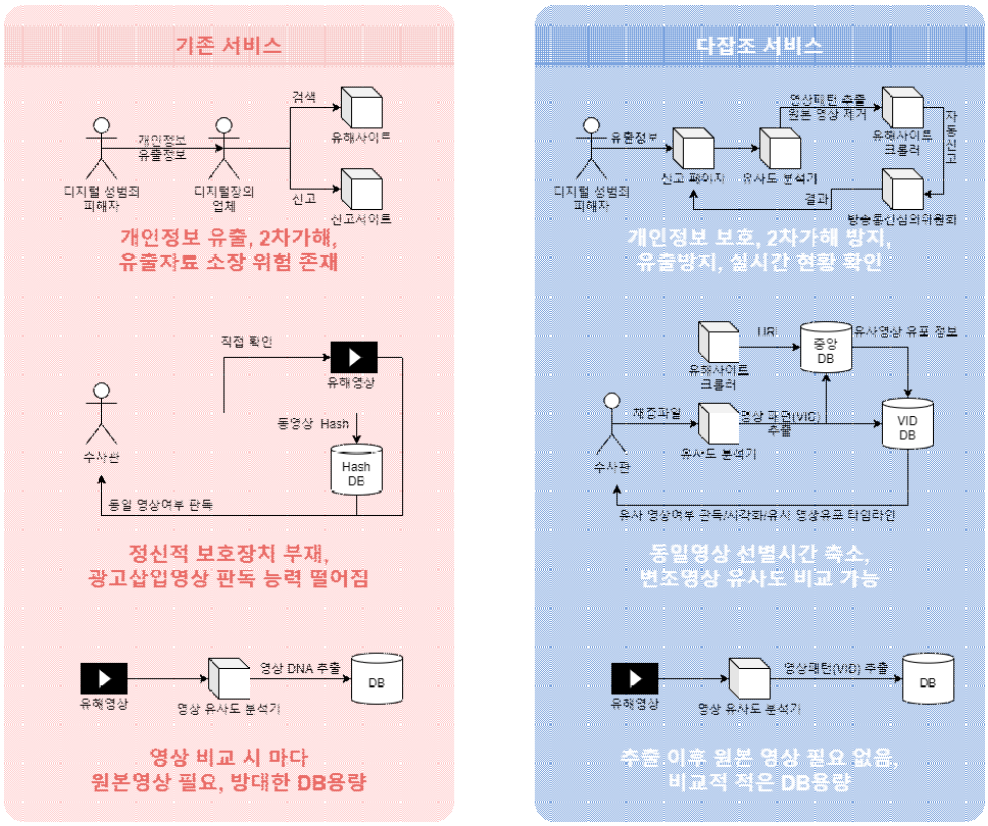


[그림 2-7] 디지털성범죄 타임라인과, 프로젝트 범위

프로젝트 범위가 정해졌고, 기획에 맞게 프로젝트를 수행하기 위해서는 크롤러, 유사도 분석기, 로깅 봇, 신고 페이지, API 서버 5가지 서비스가 필요했으며, 서비스 간 상호작용은 아래와 같이 이루어 졌습니다.

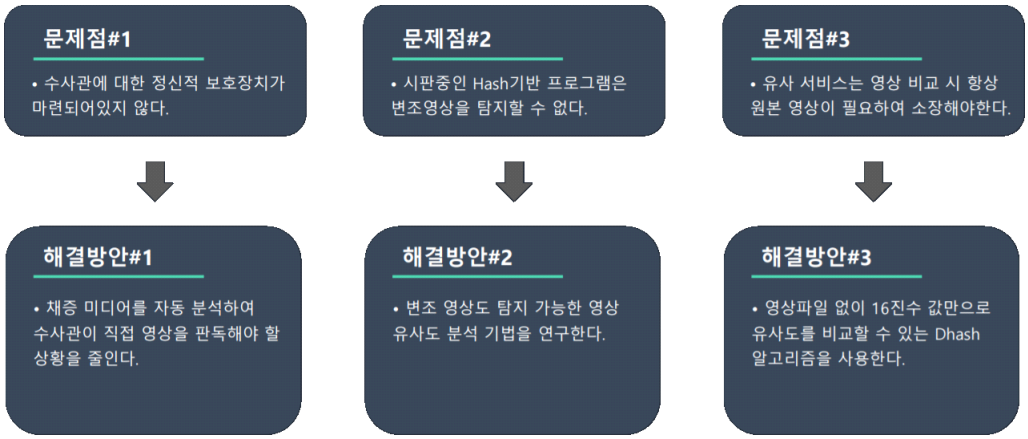
서비스	역할
크롤러 Python	유해 사이트를 순회하며, 유해 미디어를 수집하고, 식별 데이터를 추출하여 API 서버로 전송한다.
유사도 분석기 Python	수사관이 사용할 프로그램으로, 채증 된 미디어 파일로부터 식별 데이터를 추출하여 API 서버로 전송하고, 미디어 파일 간 유사도에 따라 그룹화 하여 시각화를 통한 수사 효율을 향상시킨다.
로깅 봇 JavaScript	수사 첩보를 위한 프로그램으로, 유해 미디어가 공유되는 오픈 채팅 방을 로깅하여 API 서버에 대화내역 및 채팅 방 정보를 전송한다.
신고 페이지 NodeJS	디지털성범죄 피해자가 신고할 수 있는 웹 페이지로, 피해자의 인적사항 요구 없이 URL 혹은 미디어 파일 첨부를 통해 신고 접수를 받는다.
API 서버 NodeJS	모든 서비스로부터 데이터를 수집하여 서비스가 원활히 흐르도록 한다.

[표 2-2] 프로젝트 기반 서비스 및 프로그램

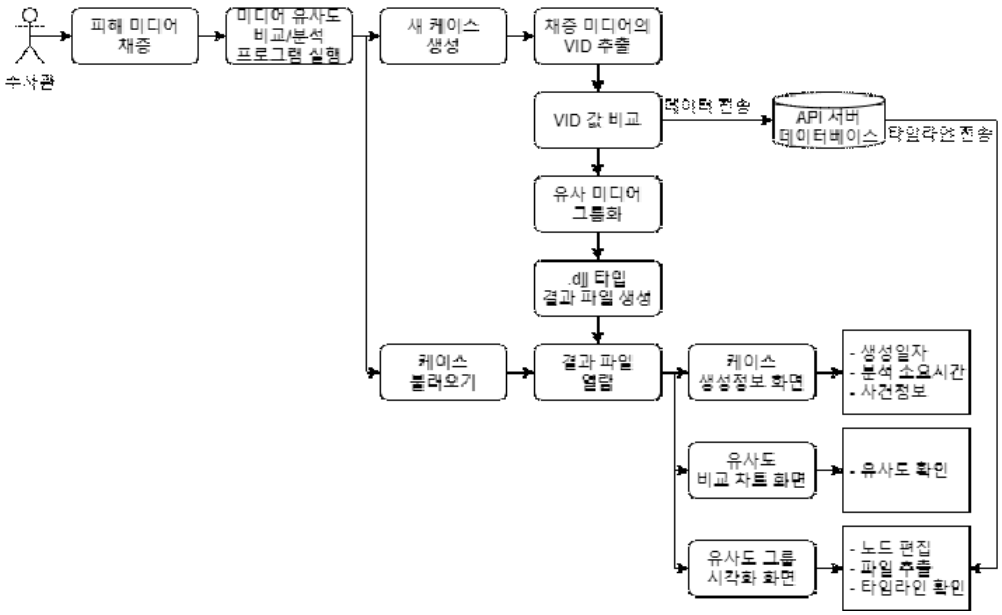


[그림 2-8] 프로젝트 전체 구성

제 3 장 미디어 유사도 분석기 개발[6)]



[그림 3-1] 미디어 유사도 분석기 개발 방향



[그림 3-2] 미디어 유사도 분석기 동작 알고리즘

6) 미디어 유사도 분석기 시연 영상 : https://tinyurl.com/y7ngoo8z

3.1 VID

VID는 Hash기반 영상 비교를 타개하기 위해 필자가 고안한 데이터 저장 타입으로, Video Identification Data의 약자입니다. VID의 자료 구조는 아래와 같이 설계하였으며, 하나의 VID는 하나의 미디어 파일에 대한 특징점 데이터[7]를 나타냅니다.

VID
· ID : 데이터베이스의 PRIMARY KEY · MediaType : 미디어의 타입. (그림/음악/동영상) · Ratio : 이미지, 동영상의 가로/세로 비율 · PlayTime : 음악, 동영상의 재생 시간 · Latitude : 파일 EXIF 값 속 위도 · Longitude : 파일 EXIF 값 속 경도 · CreatedTime : 파일 EXIF 값 속 파일 생성날짜 · SoundData : 음원에 대한 특징 데이터 배열 · FrameData : 이미지, 영상에 대한 프레임 특징 데이터 배열 · SoundGroupId : 유사 음원 그룹의 ID · FrameGroupId : 유사 프레임 그룹의 ID · FirstCatchedDate : 해당 데이터가 최초로 신고/크롤링된 날짜 · LastCatchedDate : 해당 데이터가 최근 신고/크롤링된 날짜

[표 3-1] VID 구조

VID 구조는 원본 영상을 저장하지 않고, VID를 통한 원본 영상 추측이 불가능 하도록 구성되었음에도 미디어 간 유사도 비교가 가능하기 때문에, 2차 가해 및 재 유포 위험이 있는 디지털성범죄 영상 비교에 적합 해 보였으나, 정말 좋은 구조인지에 대한 확신이 들지 않았었기에 프로젝트 진행 전인 08월 09일, **허원석 멘토님의 무엇이든 답해주는 QnA 시간**(과제로 질문 3개 이상 준비하라고 하셨다..!) 때, VID 정책에 대한 조언을 구하고자 다음과 같은 질문을 하였습니다.

7) 동영상 파일을 식별 할 수 있는 기본 단위

Q. 동영상 파일의 동일성을 구하는 방법 중, 파일 해시를 구하는 방법을 제외하고, 소리에 대한 유사도를 검사하거나, 프레임 유사도를 조사하여 비슷하게 나올 경우, 이를 각 UUID로 사용하는 방법은 적절한지 궁금합니다!
A. 개인적으로 프레임 유사도를 조사 한다는 것이, 특정 시간에 대한 어떻게 쪼갤지에 따라 다를 것. 24 프레임으로 하든 등등. 적절할 수도 있겠지만, UUID라는 것이 영상을 한 번 인코딩 할 때 마다 UUID가 달라짐. 똑같이는 만들 수 없음. 인코딩 할 때 마다 값을 생성하기 때문. 실제 수사를 할 때 동영상에 대한 동일성을 구하는 경우[8], 조작 여부를 따질 때 증거를 제시하지 못한 경우가 많음. 원본이라 할 때 영상이 조작되었는지를 따짐. 프레임 흐름을 확인. 프레임이 계속 이어지면 비트맵이 비슷하게 이어질텐데, 어느 부분이 프레임이 탁 튀면 편집 했다고 봄.

[표 3-2] VID 정책에 대한 허원석 멘토님의 답변

당시 질문 답변을 통해 이후 프로젝트를 진행할 때 큰 도움이 되어 준 '시간을 어떻게 쪼개야 하는가?'와 '비트맵이 일정하지 않고 튀는 값을 내는 경우, 편집 된 화면'이라는 단서를 얻게 되었습니다.

때문에 VID 속 프레임 및 사운드 데이터의 경우, 영상으로부터 프레임을 추출한 다음, 탁 튀는 값을 기준으로 쪼개어 배열 형태로 저장하는 방향으로 재 구상하게 되었습니다.

3.2 Dhash

프로젝트 주제에 맞도록 데이터에 대한 비교가 가능하면서도 원본 파일이 필요 없는, 그러면서도 VID 정책에 부합하는 결과를 도출하는 영상/이미지로부터 특징 점을 추출 해 내는 알고리즘이 필요하였습니다.

필자는 선행 연구로 문자열 간 유사도를 비교 할 수 있는 Fuzzing Hash에 대한 실험[9]을 진행하였지만, 문자열 간 유사도 비교에만 쓸모있고, 미디어 유사도를 비교하기엔 적합하지 않았습니다.

이후 김예령, 조서연 멘티님과 함께 검색 및 강대명 멘토님과의 상담

8) 원본/조작본 동일성 여부는 성폭행 관련 범죄에서 많이 따져 진다고 한다.
9) Fuzzing Hash 실험 보고서 : https://cafe.naver.com/goldbigdragon/90771

등을 통하여 유사도 비교가 가능한 다양한 알고리즘에 대한 정보를 수집하고 실험[10] 해 본 결과, Dhash 알고리즘이 VID구조와 프로젝트에 가장 적합함을 알게 되었습니다.

알고리즘	원본 파일을 소장해야하는가?	DB 저장이 용이한가?	속도가 빠른가?	무료인가?	난이도
Dhash	X	O	O	O	★★☆☆☆
Fuzzing Hash	O	O	O	O	★☆☆☆☆
SIFT	O	X	X	X	★★★★★
ORB	O	X	X	O	★★★★☆

[표 3-3] 유사도 비교 알고리즘 평가

Dhash는 아래 4단계를 거쳐 이미지로부터 해시 값을 추출하며, 추출된 해시 값을 통해 이미지에 대한 유사도를 구할 수 있었습니다.

Dhash
1. 이미지를 지정 크기로 줄입니다. 2. 줄어든 이미지로부터 흑/백 명도 값만 가져옵니다. 3. 각 인접 픽셀간의 명도 차를 구합니다. 4. 명도 차를 계산하여 16진수 형태의 Hash로 출력합니다.

[표 3-4] Dhash의 이미지 특징 값 추출 과정

이는 aHash, pHash와 같은 다른 이미지 유사도 비교 알고리즘보다 간결하면서도 정확하여[11] 구현에도 많은 시간이 들지 않았으며, 분석 시간 또한 짧은 편이었습니다.

3.3 유사도 비교

10) 이미지 유사도 판별 방법 사전 조사 보고서 : https://tinyurl.com/yd8asym6
11) http://www.hackerfactor.com/blog/index.php?/archives/529-Kind-of-Like-That.html

각 VID의 프레임 별 특징점 데이터(FrameData)로는 Dhash 결과 값을 채우기로 정해 졌으나, 아직 그 누구도 VID라는 구조를 통해 Dhash를 이용한 영상비교에 대한 연구를 진행한 적이 없었기에, 유사도 비교를 위한 다양한 상황을 분석하여 최적의 유사도 비교 알고리즘을 세워야 했습니다. 때문에 특징점 데이터가 될 프레임을 어떤 기준으로 수집 할 것 인지부터, 수집 된 프레임의 비교 방법까지 모두 실험해 보며 최적의 결과를 찾아나가 보았습니다.

3.3.1 프레임 추출 대상

① 모든 프레임 추출

저장공간	속도	유사도 판별 능력	난이도
매우 많이 필요	느림	낮음	★☆☆☆☆

[표 3-5] 모든 프레임 추출 방식의 성능 분석표

영상 A, B로부터 전체 프레임 추출 후 Dhash값을 구한 다음, 각각의 프레임 Dhash에 대한 1:1 비교를 진행한 결과, 너무 많은 프레임으로 인해, 추출된 데이터가 원본 미디어파일보다 훨씬 많은 저장 공간을 요구했으며, 계산 속도도 매우 느렸습니다.

뿐만 아니라 전체 프레임에 대한 비교를 진행하였더니, 전체 프레임 개수에 비해 유사 프레임이 무조건 적게 나오는 바람에 유사도 판별 능력도 떨어졌습니다.

② n초당 1프레임 추출

저장공간	속도	유사도 판별 능력	난이도
적당히 필요	보통	보통	★★☆☆☆

[표 3-6] n초당 1 프레임 추출 방식의 성능 분석표

허원석 멘토님의 조언을 토대로 특정 시간대 마다 영상을 쪼개어 나가기로 하고, 1초당, 2초당, 5초당, 10초당 각 프레임을 추출하여 Dhash값을 구한 다음, 각각의 프레임 Dhash에 대한 1:1 비교를

진행한 결과, 실험①보다 적은 용량에 어느 정도 빠른 속도를 보여주었습니다.

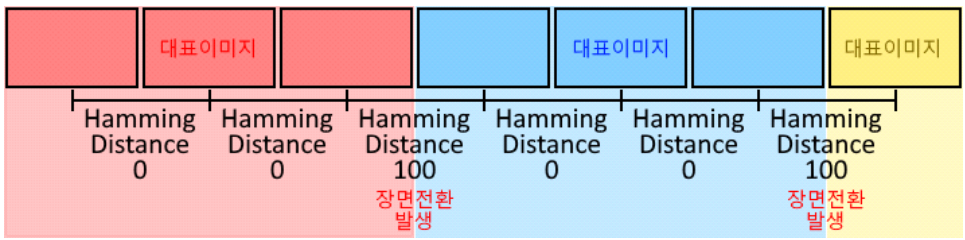
하지만, 영상마다 FPS(Frame Per Seconds)가 달랐기 때문에, 동일 영상이지만 FPS가 다르다면 비교 대상 프레임에 미세한 차이가 생기기 마련이었으며, 영상 프레임의 변화가 단조로운 영상의 경우, 거의 동일한 프레임이 지속해서 나오는 바람에 유사도 비교 능력이 다소 떨어졌습니다.

③ 대표 이미지 추출

저장공간	속도	유사도 판별 능력	난이도
적게 필요	약간 느림	보통	★★★★☆

[표 3-7] 대표 프레임 추출 방식의 성능 분석표

개선된 비디오 장면 유사도 검출 알고리즘[12]을 참고하여 논문 내용과 유사한 알고리즘을 구현하고, 모든 프레임으로부터 Dhash를 추출한 다음 첫 프레임부터 순서대로 프레임 간 Dhash를 비교하다가, Dhash의 Hamming Distance[13]가 갑자기 크게 차이가 날 경우, 차이 나기 전 구간까지의 프레임을 모아 하나의 클러스터로 선정하고, 각 클러스터로부터 대표이미지를 추출하여 유사도 비교를 진행하였습니다.



[그림 3-3] 대표이미지 추출 방법

12) 유주원, 김종원, 최종욱, 배경율. (2009). 개선된 비디오 장면 유사도 검출 알고리즘. 한국콘텐츠학회논문지, 9(2), 43-50.

13) 두개의 Dhash 값을 비교했을 때 발생하는 격차. 유사도가 높을수록 Hamming Distance는 낮게 나온다.

실험①, ②보다 훨씬 적은 용량이 필요했으며, 특히 단조로운 화면이 계속되는 영상일수록 적은 프레임이 대표이미지로 선출되며, 이에 따른 미탐 위험도 낮아졌습니다. 하지만 그와 반대로 역동적인 영상의 경우, 대표이미지가 이전 실험과 다를 것 없이 많이 추출되는 바람에 오탐의 위험은 여전하였습니다.

④ FFMPEG를 이용한 장면(Scene) 전환 추출

저장공간	속도	유사도 판별 능력	난이도
매우 적게 필요	매우 빠름	매우 높음	★☆☆☆☆

[표 3-8] 장면 전환 프레임 추출 방식의 성능 분석표

프레임 추출 관련으로 고민하고 있던 사이, BoB 8기 수료 선배님 중, 영화 무단 복제 추적 관련 프로젝트를 진행한 팀이 있음을 확인하였고, 인터뷰를 통해 선배님들은 FFMPEG를 이용하여 n초 마다 1개씩 프레임을 추출한다고 답변 해 주셨습니다.

n초마다 1개씩 프레임을 추출하는 방법은 이미 실험을 마친 상태였지만, FFMPEG는 처음 들어보는 도구였기에, 분석하던 도중, FFMPEG를 이용하여 영상 내 장면 전환 프레임을 추출 할 수 있음을 알게 되었습니다.

이후 FFMPEG를 연동하여 영상으로부터 장면전환 프레임만을 추출하여 Dhash값을 뽑아 각기 다른 영상을 비교 해 보았더니, 가장 높은 유사도 판별 능력을 보였으며, 추출된 장면전환 프레임 또한 차이가 뚜렷하면서도 적게 추출되어 용량부담도 적었습니다.

⑤ Dhash 리사이즈 크기

	8pixel	16pixel	24pixel	32pixel	48pixel	64pixel
처리 속도	왕빠름	빠름	보통	보통	느림	왕느림
저장 공간 차지	적음	적음	보통	보통	보통	많음
유사도 판별 능력	오탐多	오탐多	오탐多	보통	높음	미탐多

[표 3-9] Dhash 리사이즈 크기별 속도 및 판별력

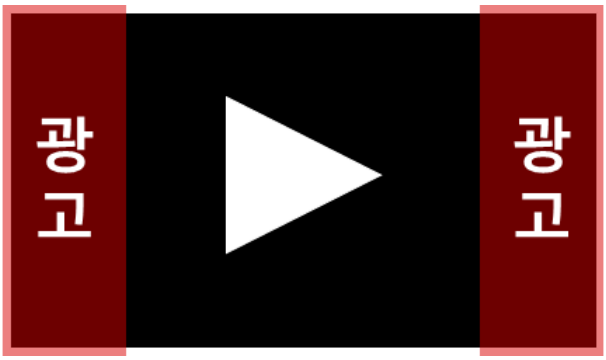
프레임 추출에 대한 최적 방안을 찾았지만, Dhash를 이용할 때 얼마나 이미지를 줄이는가에 따라 유사도 판별 능력이 달라졌기 때문에, 추가적인 실험을 진행하였습니다.

가로, 세로를 너무 작게 리사이즈 할 경우, 속도는 정말 빨랐지만, 대부분의 이미지가 유사하다고 나오는 바람에 오탐율이 매우 높았습니다. 반대로 너무 크게 리사이즈 할 경우, 처리 속도가 느려지고, 추출 데이터 크기가 1 GB를 넘어가는데다, 유사한 영상일지라도 미세한 오차를 발견하여 다른 영상이라고 판단하는 바람에 미탐율이 매우 높았습니다.

여러 차례에 걸친 리사이즈 실험을 통해 가로 32, 세로 32 pixel 일 경우가 가장 적당하였으나, 오탐, 미탐 위험이 다소 존재하였기에, 속도가 느려지더라도 48pixel을 Dhash 추출 기준으로 선정하였습니다.

3.3.2 변조 탐지 방법

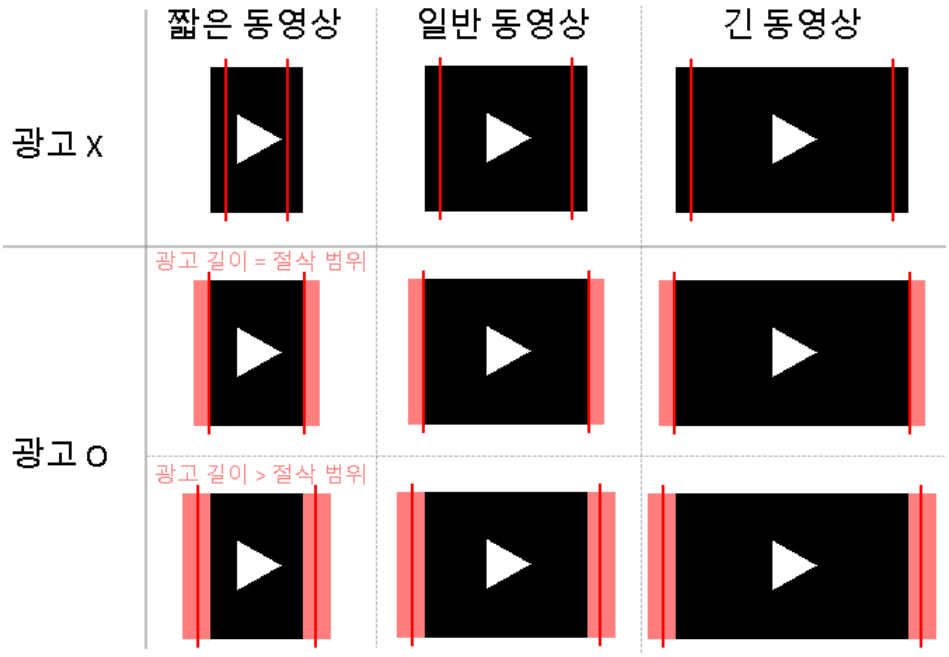
① 영상 앞/뒤 광고 삽입



[그림 3-4] 영상 광고 삽입 예시

국내 음란물 스트리밍 사이트의 경우, 광고 시작 혹은 끝 부분에 자사 도박 사이트 혹은 음란물 사이트에 대한 광고 영상을 삽입하기에, 유사도 비교가 쉽지 않습니다.

초기에는 광고 부분 만 잘라내어 최대한 원본 영상 프레임을 살리는 방안으로 개발 해 나갔습니다. 하지만 Dhash의 Hamming Distance 값이 높게 나오는 경우를 광고가 끝난 시점으로 잡으려니, 너무 많은 경우의 수가 나왔으며, 영상 앞/뒤로 N초 만큼 잘라내려하니, 내용은 동일한데 광고의 길이가 다른 경우와, 광고가 붙지 않은 영상 등 각양각색의 영상물이 많았습니다. 만일 이를 무시하고 고정된 길이만큼 영상 앞/뒤를 잘라 낼 경우, 광고가 조금이라도 남아있거나, 설정된 N초 보다 더 짧은 길이의 영상에 대한 비교 능력을 상실하게 될 것이 분명하였습니다.

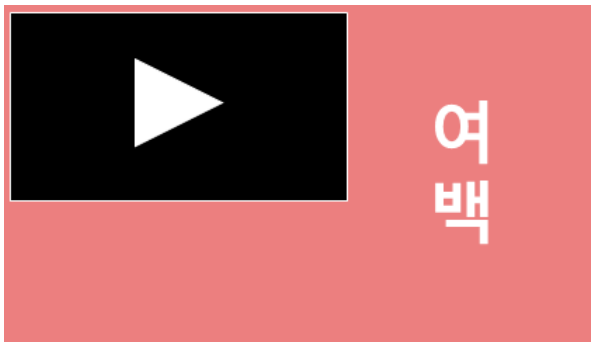


[그림 3-5] 고정 절삭범위 적용 시 발생할 상황

때문에 영상 길이의 N%만큼 앞/뒤를 잘라내는 알고리즘을 구현하

여, 영상 길이에 비례하도록 앞/뒤를 잘라내었습니다. 그 결과, 미디어 길이에 상관없이 원본 재생구간이 확보할 수 있게 되어 10초 내외의 짧은 영상일지라도 유사도 비교가 가능하였으며, 대부분의 광고 영역이 확실히 제거되었습니다.

② 영상 리사이즈



[그림 3-6] 영상 리사이즈 예시

원본 영상의 가로, 세로가 짧은 상태에서 원본 영상보다 큰 광고사진/동영상을 첨부할 경우, 원본 영상 겉 부분이 검은 여백으로 표시되는 경우가 많습니다. 뿐만 아니라 저작권 침해를 우회하기 위해 원본 영상을 임의대로 축소하여 영상의 가장자리에 배치하고, 겉면에는 단색 배경을 넣거나 상기된 바와 같이 검은 여백처리를 가하며, 이를 모두 통틀어 리사이즈 변조로 일컫습니다.

영상에 대한 리사이즈가 일어날 경우, 원본 이미지가 뭉개지는 현상이 일어나게 되므로, 대부분의 영상 유사도 비교 프로그램으로는 비교가 불가능하지만, Dhash의 경우 연산 과정에 이미 비교하고자 하는 이미지를 축소하여 뭉개는 과정이 들어가기 때문에 원본 영상이 뭉개지는 경우는 영향이 거의 없습니다.

하지만 문제는 역시나 주변의 여백으로, 원본 영상 옆에 붙은 검은 여백으로 인해 유사도 비교 능력이 뛰어난 Dhash 알고리즘에서도 완전히 다른 영상으로 판단하기 일쑤였습니다.

여백 제거를 위한 알고리즘으로 Dhash를 추출하기 위해 흑/백 명

도로만 이루어진 이미지를 사용하여, 상/하/좌/우의 명도 값이 0~4 혹은 251 ~ 255 사이인 픽셀이 70% 이상 이어질 경우, 여백으로 인식하는 코드를 작성하였습니다.

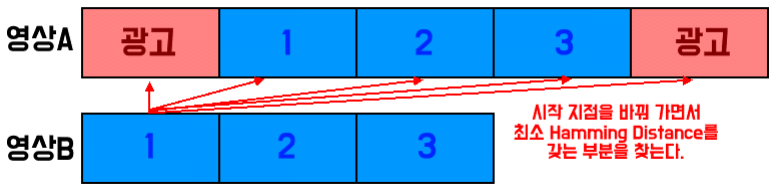
[그림 3-7] 명도 계산(좌), RGB 계산(우)

하지만 명도계산법으로는 검은 여백은 잘 지워졌으나, 배경 위에 워터마크로 글이 작성되어있는 경우, 잘려나가지 않는 결과를 보여주었습니다. 때문에 명도 추출 단계 이전인 RGB 색상 값을 이용하여, R + G + B ÷ 3 의 오차 값이 5 이내인 픽셀이 70% 이상 이어질 경우, 여백으로 인식하는 코드를 작성하여 워터마크까지 여백으로 인식하도록 하였더니, 검은 여백은 물론 여백 안의 워터마크까지 깔끔하게 제거되었습니다.

여백 제거 코드를 이미지 추출단계에서 실행하고, 이를 통해 새로 만들어진 영상의 시작 x, 시작 y, 끝 x, 끝 y 값을 이용하여 해당 부분만 크롭(crop)하여 여백을 제외한 원본 영상만을 추출하고, 이를 통해 유사도 비교를 진행하였더니 좋은 성능을 발휘하였습니다.

3.3.3 프레임 비교 방법

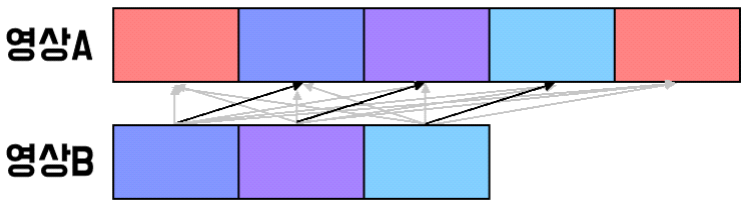
① 영상 흐름(Stream) 비교



[그림 3-8] 영상 흐름(Stream) 비교법

고급진 알고리즘을 구현하기 전, 가장 기초적인 방법으로, 각 프레임을 순서대로 대조하는 방식을 사용하였습니다. 하지만 위와 같이 더 짧은 영상을 기준으로, 긴 영상의 대표 프레임을 한 장 씩 대입하는 방식은 긴 영상의 앞이나 뒤, 혹은 사이에 짧은 영상의 첫 프레임이 끼어있을 경우, 전체적인 유사도를 떨어뜨렸습니다.

② 첫 프레임 우선 식 비교

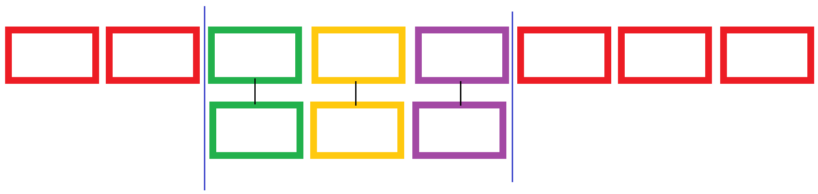


[그림 3-9] 첫 프레임 우선 식 비교법

첫 단추부터 잘 맞지 않으면 유사도 비교 효율이 확 내려가는 영상 흐름 비교법으로부터 첫 단추부터 잘 끼우자는 교훈을 얻고, 이후 짧은 영상의 1번 프레임을 긴 영상의 모든 프레임에 대입한 뒤, 가장 높은 유사도를 보이는 프레임부터 2번째 프레임 대입을 시작하도록 설계하였습니다만, 유사 영상 비교 효율이 어느 정도 상승하였어도 여전히 일부 영상에 대한 유사도 비교 효율이 낮았기에, 좋은 방법이 아님을 깨닫게 되었습니다.

③ 스펙트럼 산출식 비교

첫 프레임 우선 비교 알고리즘이 제대로 작동하지 않은 원인은, 미디어 프레임 구성 상황에 따른 한계에 부딪혔기 때문이었습니다.

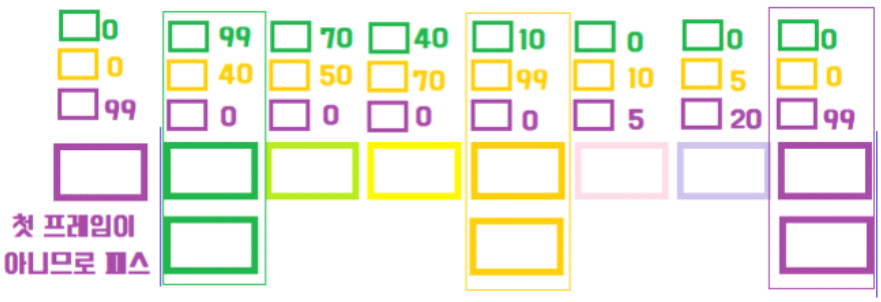


[그림 3-10] 완전히 동일한 프레임이 정확히 추출 되었을 경우

첫 프레임 우선 비교 알고리즘이 제대로 동작하려면, 완전히 동일한 프레임이 정확히 추출되어야 하며, 유사하거나 비슷한 프레임이 단 1장도 존재해서는 안 됩니다.

[그림 3-11] 유사 프레임의 연속

하지만 대부분의 미디어 프레임은 [그림 3-9]와 같이 유사 프레임의 연속입니다. 때문에 녹색 구역에서는 녹색 프레임이 짝을 맞추고, 노란색 구역에서는 노란색 프레임을 짝을 맞춰야 하지만, 바로 다음 구역인 연두색 구역과 노란색 프레임의 유사도가 매우 높기 때문에, 2번째 프레임부터 비교 대상이 꼬이게 됩니다.



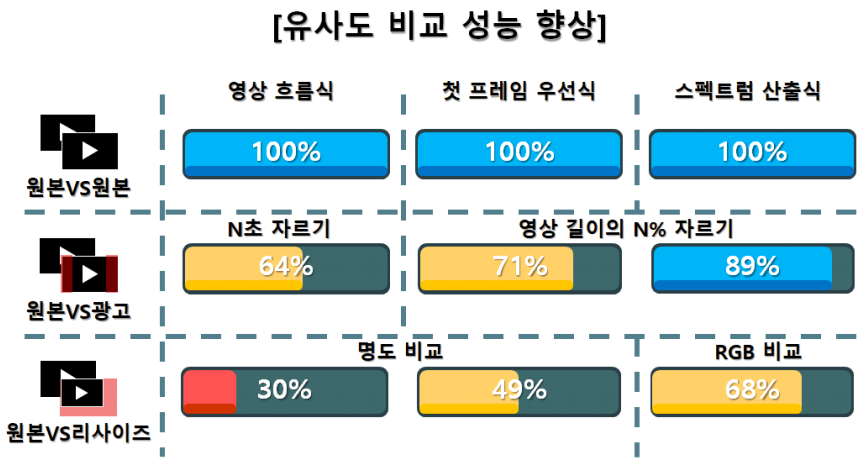
[그림 3-12] 스펙트럼 산출식 비교

이에 유사도 비교 시 처음부터 영상 A의 모든 프레임에 대한 영상 B의 모든 프레임을 비교하여 각 프레임과 얼마나 유사한지 구한 다음, 해당 프레임과 가장 유사했던 프레임을 짝 지어 내고, 첫 프레임 우선 식 비교 알고리즘을 일부 계승하여, 먼저 나와야 할 프레임보다 뒤에 있는 프레임과 닮은 부분을 찾더라도, 먼저 나와야 할 프레임부터 짝을 찾아주도록 하였습니다.

3.3.4 성능

실험 환경
- Intel(R) Core(TM) i7-6700HQ (RAM 16 GB) - Dhash 가로/세로 리사이즈 크기 : 각 48 pixel - 영상 앞/뒤 소거 비율 : 25% - 섬네일 추출 : 활성화 - 스레드 개수 : 8개 - 대상 미디어 : 동영상 84건, 사진 0건, 음악 0건 아시아 계열 포르노 동영상 (전문 배우가 합의 하에 대본대로 촬영한 영상이며, 아시아 계열 배우가 한국인과 가장 유사하기에 해당 미디어를 선정함) 총 길이 : 11시간 34분 07초 (41,647초), 3.23GB (3,469,567,823 바이트) - VID Group 기준 유사도 : 80% - 비교 알고리즘 : 스펙트럼 산출식

[표 3-10] 실험 환경



[그림 3-13] 유사도 비교 성능 향상 표

MCC 계수

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

- 디렉터리 기준 전체 평균 MCC : 0.85 (인식 98.13 %) (오탐/미탐 1.87 %)
- VID 그룹 기준 전체 평균 MCC : 0.99 (인식 99.91 %) (오탐/미탐 0.09 %)
- 지정파일 기준 전체 평균 MCC : 0.88 (인식 98.15 %) (오탐/미탐 1.85 %)
- 전체 평균 신뢰도 : 90.9 % (인식 98.73 %) (미인식 1.27 %)

[표 3-11] MCC 계수

검사목록	내용	결과
강인성	오탐/미탐에 대한 비율이 5% 이하인가?	1.34%
일관성	반복 추출된 특징 정보(dhash)가 동일한가?	100%
부분매칭	변조영상에 대한 유사도 판단의 정확도가 95%이상인가?	86.8%
고속추출	특징 정보를 추출하는데 걸리는 시간(평균특징 정보 추출시간(S)/콘텐츠 크기) 가 5% 이하인가?	3.6%

[표 3-12] 성능평가확인서를 받기위한 기준 검사 표[14)]

아쉽게도 부분매칭 항목이 8.2% 부족하여 기준에 미치지 못하였지만, 새로운 알고리즘을 개발 해 나가며 맞춰 나가기로 하였습니다.

3.4 프로그램 개발

3.4.1 개발환경

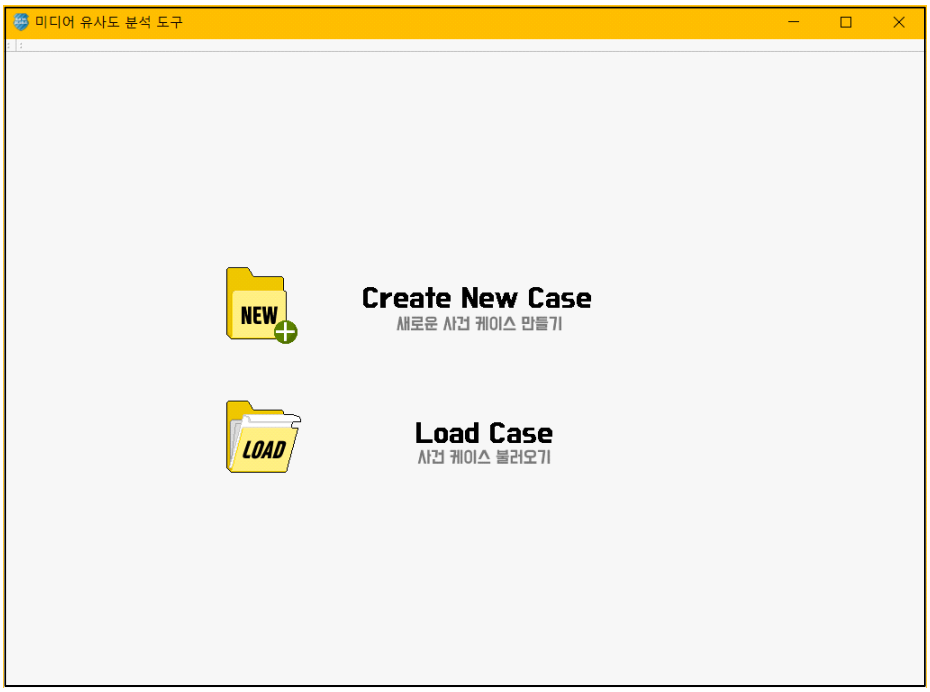
미디어 유사도 분석기는 모든 서비스의 기반이 될 VID 추출 모듈과 유사 미디어 그룹화 알고리즘을 구현하여 모듈화 시켜야했기에, 라이브러리가 다양하며 작성이 쉬운 Python을 기반으로 제작하였습니다.

14) 특수유형부가통신사업자의 불법정보 유통방지를 위한 개선방안 연구, 한국정보통신진흥협회, 2015.12 (https://tinyurl.com/y7u6kvus)

	이름	버전
IDE	Spyder	4.1.5
Language	Python	3.8.5
FFMPEG	ffmpeg, ffprobe	N-99286-gc7867b6ed1
Library	ffmpeg-python	0.2.0
	Image	1.5.33
	opencv-python	4.4.0.46
	Pillow	8.0.1
	pymysql	0.10.1
	PyQt5	5.9.2

[표 3-13] 개발환경

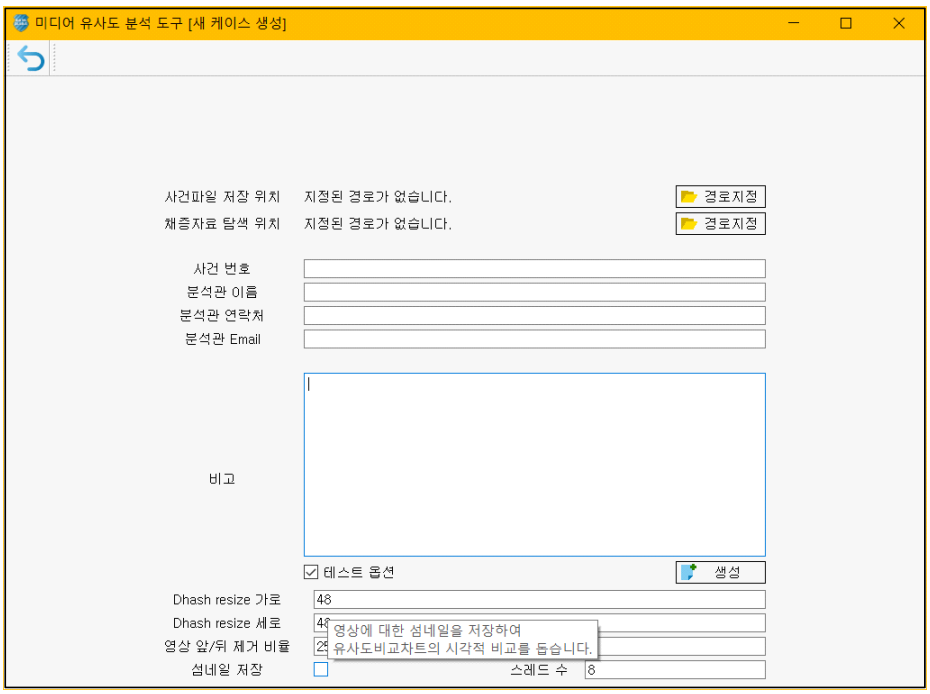
3.4.2 CASE 생성/불러오기



[그림 3-14] 프로그램 메인화면

수사 목적 프로그램인 만큼, 분석 결과를 사건 별로 관리할 수 있도록 CASE 기능을 추가하였습니다. 분석관이 채증 파일을 프로그램에

넣으면, 채증파일로부터 데이터를 추출하고 난 다음, .djj 확장자의 CASE 파일을 생성하며, 생성된 CASE 파일은 어디에서 열람하든 항시 동일한 내용을 확인 할 수 있습니다.



[그림 3-15] CASE 생성 화면

CASE 생성 시, 채증 파일이 보관된 디렉터리를 지정함으로 써, 디렉터리 내 모든 파일을 분석할 수 있으며, CASE 파일 저장 위치 지정 및 CASE 정보 입력 후, 분석을 시작할 수 있습니다.

만일 채증파일의 특성에 맞는 분석환경을 설정해야한다면, 테스트옵션 체크박스 버튼을 선택함으로 써, 분석관이 필요한 분석 환경으로 조정할 수 있습니다.

3.4.3 CASE 정보

채증 파일로부터 VID 추출 이후, 가장 처음 보여 지는 화면으로, 추

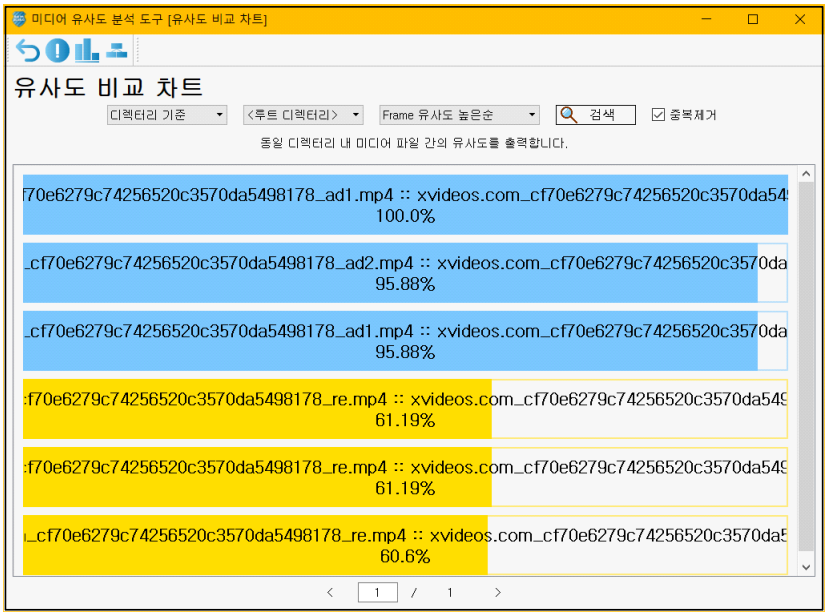
출에 소요된 시간과, CASE 정보를 확인할 수 있습니다. 보고서 작성이 수월하도록 각 정보들은 Label 형식이 아닌 Textbox 형태로 제작하여 복사/붙여넣기가 용이하도록 제작하였습니다.

[그림 3-16] CASE 정보화면

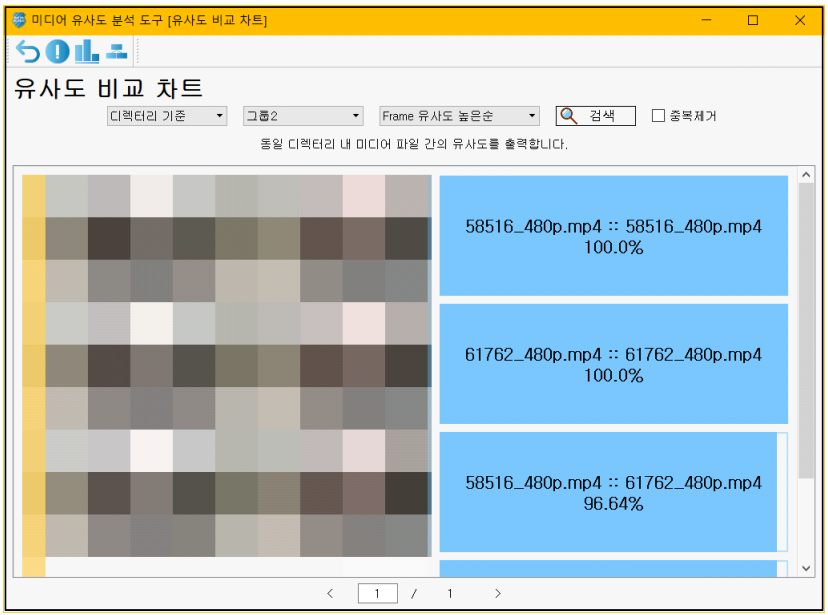
3.4.4 유사도 비교 차트

미디어 유사도를 차트 형식으로 비교 할 수 있도록 유사도 비교 차트 메뉴를 생성하였으며, 해당 메뉴에서는 [특정 디렉터리], [동일 그룹], [특정 미디어]에 대한 비교 분석 차트를 지원하며, Frame, 음성, 메타 데이터 등에 대한 유사도가 높거나 낮은 순으로 검색 가능하도록 설계하였습니다.

앞서 CASE를 만드는 단계에서 섬네일 추출 기능을 활성화 시킬 경우, 유사도 비교 차트의 좌측면에 미디어 섬네일이 표시되며, 육안으로 빠르게 판단이 가능하도록 하였습니다.



[그림 3-17] 유사도 비교 차트



[그림 3-18] 섬네일 추출 옵션 사용 시의 유사도 비교 차트

3.4.5 유사도 그룹 관계도

[그림 3-19] 유사도 그룹 관계도

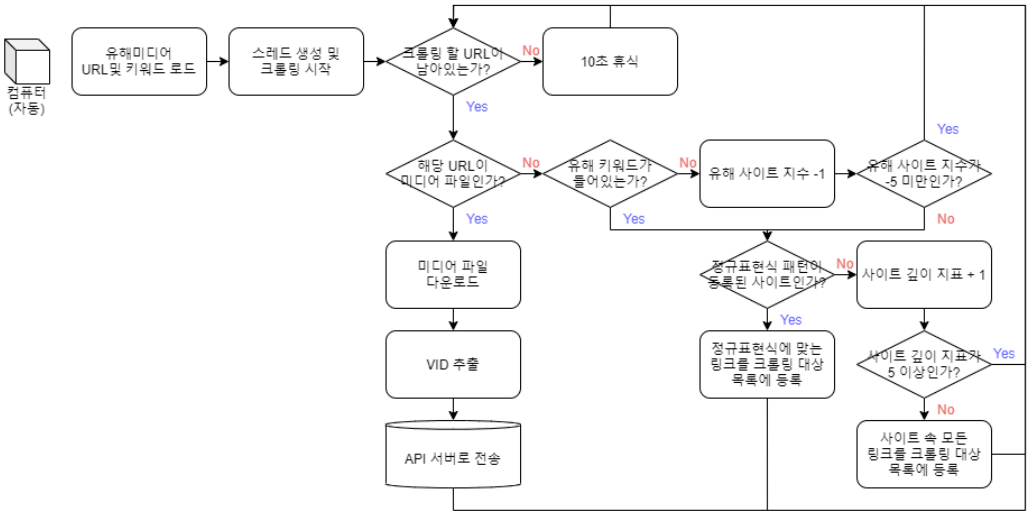
유사도 그룹 관계도 에서는 유사 미디어간의 관계를 노드 형식으로 그룹지어 분석 효율을 향상시켰습니다. 노드 추가/제거와, 이름, 아이콘, 그룹, 위치 변경이 자유로우며, 부모/자식 노드 설정이 가능하도록 개발하였습니다.

노드 정보화면 창에서 파일 추출 버튼을 선택할 경우, 분석관이 현재 배치한 노드 구성대로 디렉터리 및 파일이 생성되며, 현재 선택한 미디어 노드의 파일 위치를 간편하게 추적 할 수 있도록 편의성을 더하였습니다.

이외 선택된 미디어와 유사한 미디어를 발견한 전적이 있다면, 타임라인 출력을 통해 언제, 어느 URL에서 유사한 미디어가 발견되었는지 추적 할 수 있습니다.

제 4 장 크롤러 개발

4.1 유해 미디어 수집 크롤러[15)]



[그림 4-1] 유해 미디어 수집 크롤러 동작 알고리즘

피해자로 부터 피해 미디어 신고 접수를 받고나서 유사 미디어가 인터넷상에 유포되고 있었는지 여부를 알기 위해서는, 항시 미디어 데이터와 함께 해당 미디어가 게시된 URL을 수집 해 나가야합니다. 때문에 피해 미디어 수집을 위해서는 유해 사이트에 대한 크롤링이 필수였으므로, 일전에 필자가 개발한 유해사이트 탐색 크롤러[16)] 코드와 알고리즘을 그대로 사용하기로 하였습니다.

4.1.1 개발환경

유해 미디어 수집 크롤러는 실행 이후 관리자 조작이 필요 없도록 대부분의 작업을 자동화시킴으로 써, Linux환경 서버 PC와 같이 비

15) 유해 미디어 수집 크롤러 시연 영상 : https://tinyurl.com/y9opv46l
16) 유해사이트 탐색 크롤러 개발일지 : https://cafe.naver.com/goldbigdragon/85753

교적 적은 컴퓨터자원을 가진 곳에서도 동작하도록 설계하였습니다.

	이름	버전
IDE	Spyder	4.1.5
Language	Python	3.8.5
FFMPEG	ffmpeg, ffprobe	N-99286-gc7867b6ed1
Library	beautifulsoup4	4.9.3
	ffmpeg-python	0.2.0
	Image	1.5.33
	json5	0.9.5
	Pillow	8.0.1
	requests	2.24.0
	selenium	3.141.0

[표 4-1] 개발환경

4.1.2 기본 기능 구현

① URL 수집

PhantomJS 웹 드라이브와 Selenium 파이썬 라이브러리를 이용하여 JavaScript를 실행함과 동시에, Proxy 연결을 통하여 웹 사이트 내 콘텐츠를 가져온 다음, 콘텐츠 내용으로부터 유해 키워드가 포함되어 있는지 확인하여 키워드가 포함되어 있을 경우, 해당 URL 속에 포함된 링크들을 다음 검색 대상 LIST에 담는 기초적인 코드를 작성하였습니다.

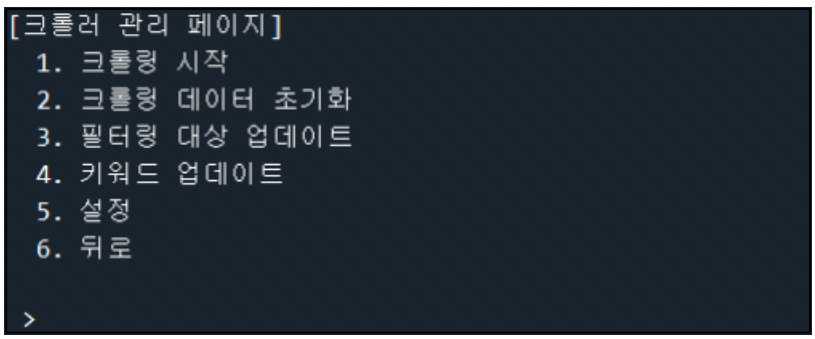
이후 다중 스레드 환경을 구축하여 위 작업을 프로그램 종료 시까지 중단되지 않고 반복 수행하도록 하였습니다.

② 미디어 다운로드

스레드 내부에서 검색 대상 LIST로부터 URL을 가져왔을 때, URL 주소가 .mp4와 같은 미디어 파일을 의미할 경우, 키워드 매칭 없이 바로 해당 미디어를 다운로드 받은 후, VID 추출 모듈을 이용하여 VID를 추출하고, 즉각 다운로드 받았던 미디어를 삭제하도록 하였습니다.

③ CLI 제공

유해 미디어 수집 크롤러의 경우, 미디어 유사도 분석기와는 달리 사람의 조작을 필요로 하지 않는 자동 수집 도구이기에, 자원 소모가 많은 GUI를 굳이 구현할 필요가 없었습니다. 때문에 최초 1회 프로그램을 실행 시킬 때 만 사람의 조작이 필요하므로, CLI를 제공하기로 하였습니다.



[그림 4-2] 유해 미디어 수집 크롤러 - 관리 페이지

4.1.3 부가 기능 구현

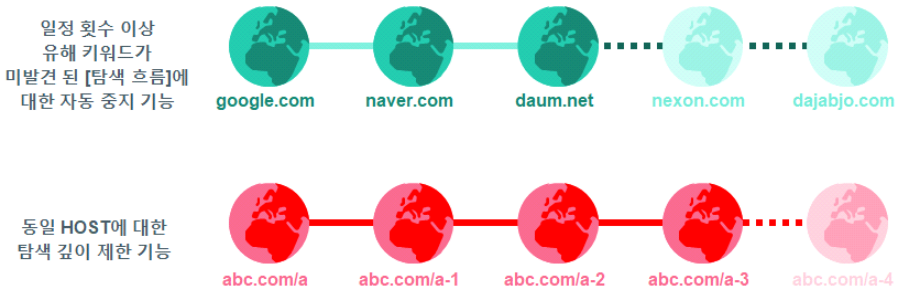
① 메모리 스와핑

검색 대상 LIST는 크롤러의 사용시간이 늘어날수록 커지기 때문에 적절한 관리가 이루어지지 않을 경우, 메모리 스택이 꽉 차면서 프로그램이 강제 종료 됩니다. 때문에 검색 대상 LIST가 일정 수치 이상 쌓일 경우, SQLite기반 DB에 차곡차곡 옮겨 담고, 검색 대상 LIST가 비어있을 경우, 다시 DB로부터 저장된 내용을 꺼내 오도록 설정하였습니다.

② 무해사이트 판정, 검색 깊이 설정

아무리 유해 키워드 기반 유해 사이트 탐색 크롤러라 할지라도, 유해 키워드만 보이면 죄다 수집하기 때문에 간혹 위키 백과, 의학 사전 등 무해한 사이트가 억울하게 잡혀오는 일이 생겼습니다.

때문에 동일 HOST의 경우, 내부 링크 탐색 깊이를 제한하도록 하여 특정 깊이까지의 내부 링크만 수집하도록 함과 동시에, 일정 횟수 이상 유해 키워드가 탐지되지 않은 사이트에 대하여, 더 이상 하위 링크를 수집하지 않도록 설정하였습니다.



[그림 4-3] 무해사이트 판정, 검색 깊이 설정

③ 탐색 우선순위

크롤링을 계속해서 진행하다보면, 수 만개의 링크가 검색 대상 LIST에 쌓여 있는 모습을 볼 수 있습니다. 하지만 유해 미디어 수집 크롤러의 본연의 임무는 유해 미디어로부터 VID를 추출하는 것인데, 갖가지 링크 URL들로 인해 미디어 파일 링크의 대기 순번이 뒤로 밀려나게 됩니다.

이를 방지하고자 탐색된 URL에 대하여 우선순위(Priority)를 부여하여 미디어 파일을 가리키는 주소는 무조건 1순위로 탐색하도록 설정함으로 써 크롤러의 효율을 향상시켰습니다.

④ 사이트 맞춤형 정규표현식

크롤러는 모든 사이트에 대하여 효율적 일수 없습니다. 유해 미디어를 유포하는 사이트가 1000개 있다면, 그 중 미디어 다운로드 시도조차 할 수 없는 사이트가 800개일 것입니다.

유해 미디어 유포 사이트의 경우, Sendvid 혹은 기타 미디어 스트리밍 플랫폼을 이용하여 임베디드 미디어 스트리밍을 추구하거나, iframe을 이용하여 페이지 내 크롤링이 힘들도록 하거나,

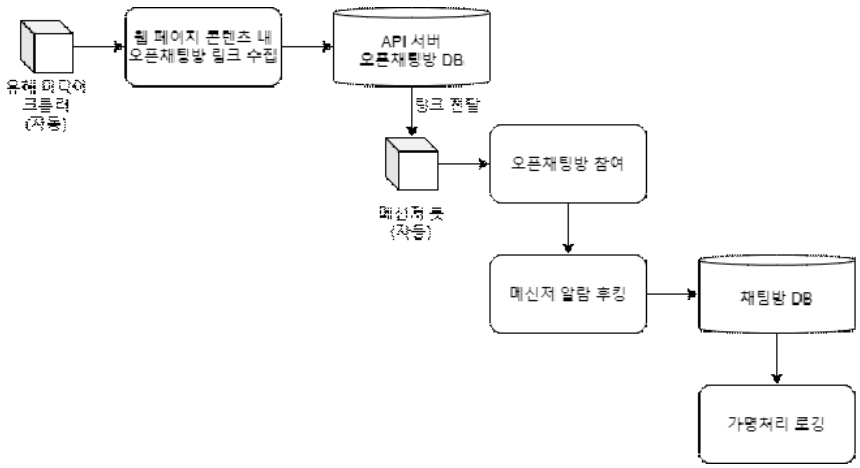
javascript를 이용하여 아예 실시간으로 불러오는 등 다양한 이유로 인해 크롤링 환경을 어렵게 만듭니다.

때문에 유해 미디어 스트리밍 사이트 중, 유명하거나 자료가 많은 사이트를 일부 골라내어 해당 HOST에 대한 크롤링 시도 시에는 해당 사이트 내부 문서 규칙에 맞는 크롤링을 시도하도록 함으로 써 효율을 높였습니다.

```
# 특정 사이트에 대한 영상 링크 및 페이지 링크 추가
def sitemodule(urlStructure, soup):
  regexedLinks = None
  if '[redacted]' in urlStructure.url or '[redacted]' in urlS
    regexedLinks = soup.find_all(href=re.compile('^(https?:\/\/)?yado
    regexedLinks += soup.find_all(href=re.compile('(https?:\/\/)?yado
    regexedLinks += soup.find_all(href=re.compile('^board\.php\?.*pag
  elif '[redacted]' in urlStructure.url:
    regexedLinks = soup.find_all(href=re.compile('^show\.php\?u\=.*$
    regexedLinks += soup.find_all(href=re.compile('^index\.php\?.*pag
  elif '[redacted]' in urlStructure.url:
    regexedLinks = soup.find_all(href=re.compile('^(https?:\/\/)?moa
```

[그림 4-4] 사이트 구조에 맞는 정규표현식 사용

4.2 오픈채팅방 로깅 봇[17)]

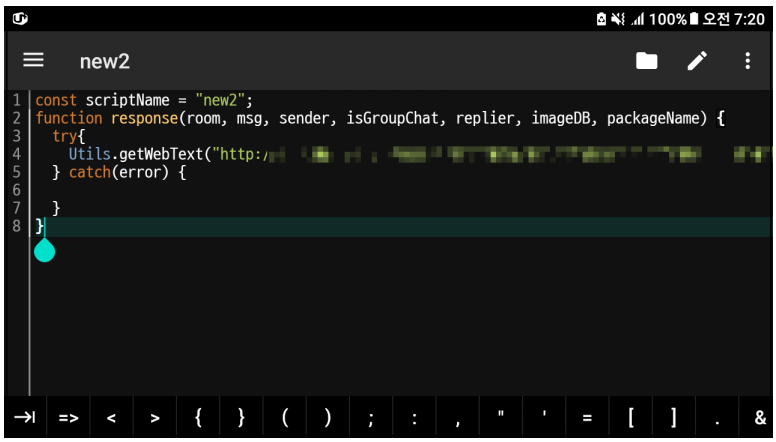


[그림 4-5] 오픈채팅방 로깅 봇 동작 알고리즘

17) 오픈채팅방 로깅 봇 시연 영상 : https://tinyurl.com/y8lkcasl

4.2.1 메신저봇R

카카오톡, 디스코드 등 메신저 플랫폼이 제공 해 주는 봇의 경우, 참여자 목록에 '봇'임이 드러나기 때문에 초대와 동시에 쫓겨날 것이 분명하였습니다. 또한 안드로이드 어플리케이션으로 직접 제작하려면 시간이 많이 걸렸기 때문에, 이미 만들어진 '메신저봇R'[18]을 사용하기로 하였습니다.



[그림 4-6] 메신저봇R 스크립트 작성 화면

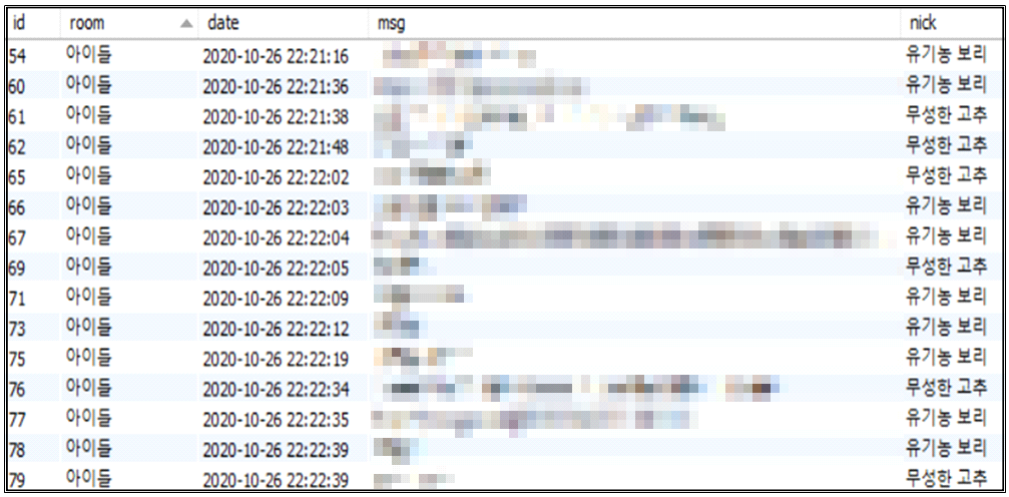
메신저봇R은 대화방에 봇을 참여시키는 형태가 아니라, 기기 내 어플리케이션 알람을 후킹하는 방식으로 메시지 내용을 읽어왔습니다. 덕분에 채팅방에 봇을 참가시키지 않아도 되며, 기기가 켜 져 있고, 인터넷만 연결되어있다면 채팅 내역을 로깅할 수 있었습니다.

4.2.2 수사 정책 제안

메신저로부터 채팅 내역을 로깅할 수는 있었지만, 통신비밀보호법 및 형사소송법과 마찰이 일어났습니다. 정작 제도적 문제로 인해

18) 메신저봇R 구글 플레이 스토어 링크 : https://tinyurl.com/y9zye8gm

활용할 수 없게 된 것이 아깝기도 하고, 정책이 완화되었으면 하는 마음에 조서연 멘티님께서 '오픈채팅방을 이용한 디지털 성범죄 수사 정책 제안'[19] 논문을 2020 한국 디지털포렌식학회 등재하였으며, 참여자 닉네임을 가명 처리하는 방안을 제안하였습니다.

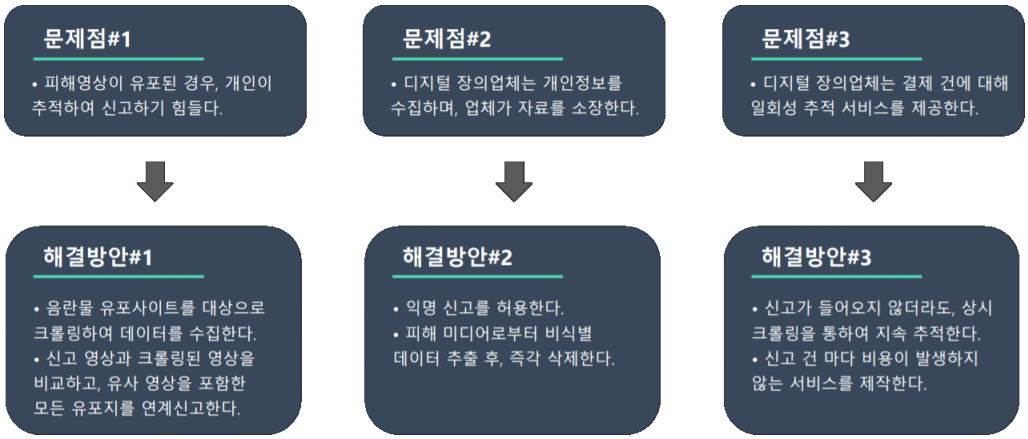


id	room	date	msg	nick
54	아이들	2020-10-26 22:21:16		유기농 보리
60	아이들	2020-10-26 22:21:36		유기농 보리
61	아이들	2020-10-26 22:21:38		무성한 고추
62	아이들	2020-10-26 22:21:48		무성한 고추
65	아이들	2020-10-26 22:22:02		무성한 고추
66	아이들	2020-10-26 22:22:03		유기농 보리
67	아이들	2020-10-26 22:22:04		유기농 보리
69	아이들	2020-10-26 22:22:05		무성한 고추
71	아이들	2020-10-26 22:22:09		유기농 보리
73	아이들	2020-10-26 22:22:12		유기농 보리
75	아이들	2020-10-26 22:22:19		유기농 보리
76	아이들	2020-10-26 22:22:34		무성한 고추
77	아이들	2020-10-26 22:22:35		유기농 보리
78	아이들	2020-10-26 22:22:39		유기농 보리
79	아이들	2020-10-26 22:22:39		무성한 고추

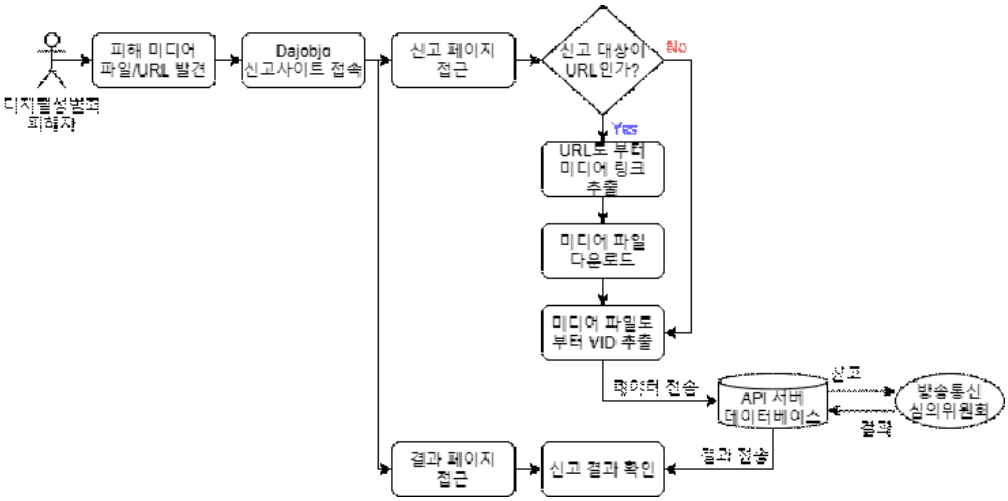
[그림 4-7] 가명처리 된 채팅 내역 (필자는 작물을 좋아한다)

19) 오픈채팅방을 이용한 디지털 성범죄 수사 정책 제안 논문 : https://tinyurl.com/ya2hv5da

제 5 장 신고 웹 페이지 개발[20)]



[그림 5-1] 신고 웹 페이지 개발 방향



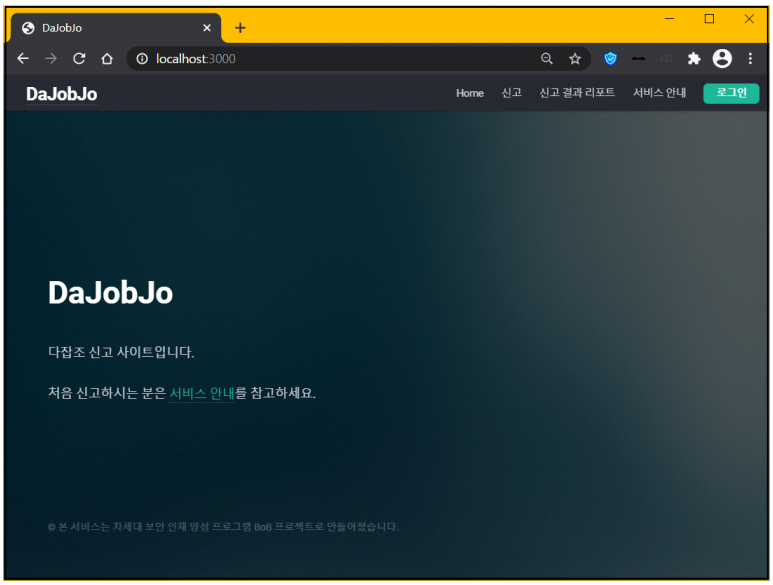
[그림 5-2] 신고 웹 페이지 동작 알고리즘

20) 신고 웹 사이트 시연 영상 : https://tinyurl.com/y995hwjt

5.1 신고 웹 페이지

수사 효율 향상을 위함과 동시에 모든 서비스의 근간이 되는 유사도 분석 모듈을 부착한 미디어 유사도 분석기와, 유해 미디어를 탐색하여 VID를 모아 나가는 유해 미디어 탐색 크롤러가 제작되었습니다. 이제 피해자로부터 신고만 받을 수 있게 된다면 3가지 서비스가 유기적으로 흐를 수 있는 상황이 되었습니다.

신고 웹 페이지는 피해자와 가장 직접적으로 연관된 서비스인 만큼, 신뢰감을 심어 줄 수 있어야 했으며, 실제로도 신뢰도 높은 동작을 보여야 했기 때문에, 웹 페이지는 미리 만들어진 템플릿을 이용하기로 하였으며, 너무 밝지도 어둡지도 않은 테마를 선정하였습니다.

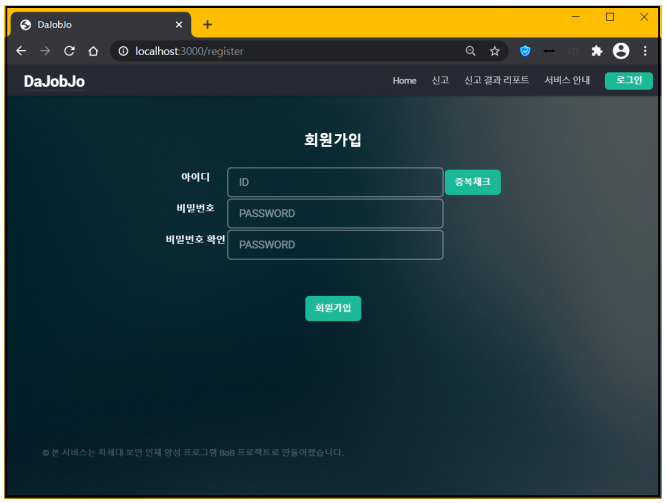


[그림 5-3] 신고 웹 페이지 홈 화면

또한 개발 시 NodeJS를 이용하여 웹 서비스의 기반을 다지되, 복잡한 연산 작업은 Python을 이용하도록 설계함으로 써 연산 효율을 향상시켰습니다.

5.1.1 계정 생성

신고 이후 결과 확인을 위해서는 반드시 신고자를 특정 할 수 있어야 하지만, 본래 계획대로 신고자의 개인정보를 수집하지 않아야 했습니다. 때문에 간편하게 ID와 비밀번호만 입력하면 가입이 승인되었으며, 전화번호, 이름, 주민등록번호 등 일체의 개인정보를 수집하지 않도록 하였습니다.

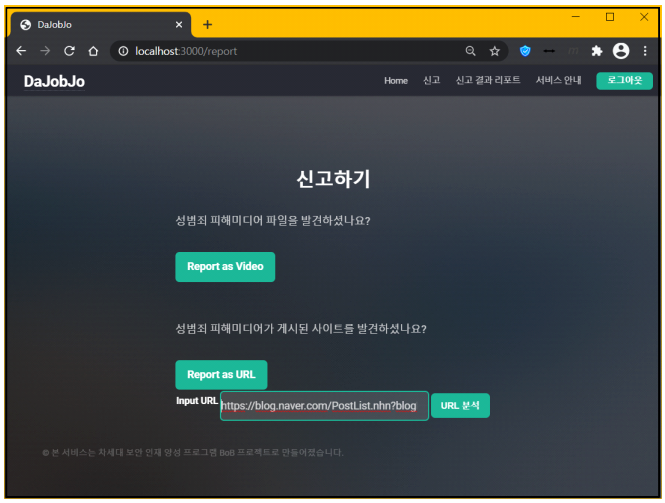


[그림 5-4] 회원가입 페이지

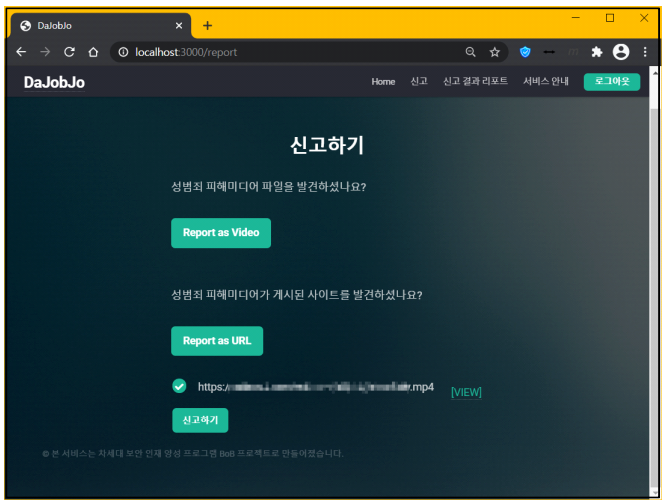
5.1.2 URL 링크 신고

디지털 성범죄 피해자의 경우, 피해 영상을 소지하고 있는 것 자체로도 심각한 트라우마를 야기하며, 보통 피해자 대신 지인이 신고하는 경우가 많았기에, 피해 영상이 게시된 대상 웹 페이지 URL 신고가 가능하도록 제작하였습니다.

URL 신고 시, 내부 크롤러가 대상 페이지로부터 미디어 파일을 탐색하게 되며, 탐색 된 미디어 파일 목록이 출력되면 피해자가 직접 신고를 원하는 대상 미디어 파일을 선택할 수 있도록 하였습니다.



[그림 5-5] URL 신고 화면

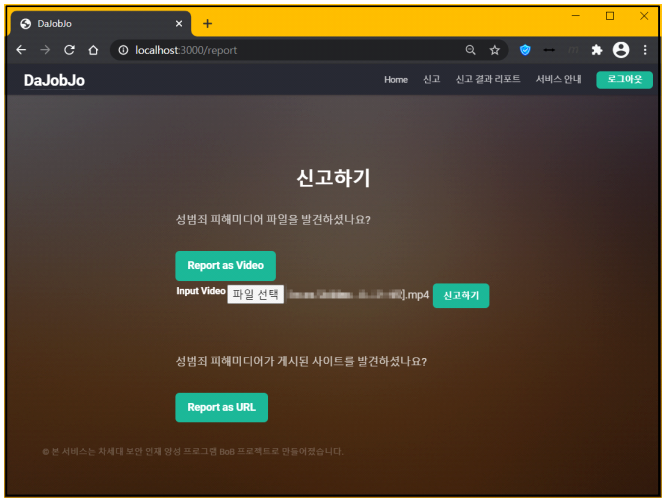


[그림 5-6] 신고 대상 미디어 파일을 선택 한 모습

선택 이후 신고 버튼을 클릭할 경우, 서버는 해당 미디어 파일을 다운로드 받은 후, VID 및 영상 추적 정보를 추출하여 API서버에 전송한 다음, 다운로드 받았던 미디어 파일을 즉각 폐기하도록 설계하였습니다.

5.1.3 미디어 파일 신고

혹여나 피해 미디어 파일을 소지하고 있을 경우를 위해 미디어 파일을 첨부할 수 있도록 하였습니다.

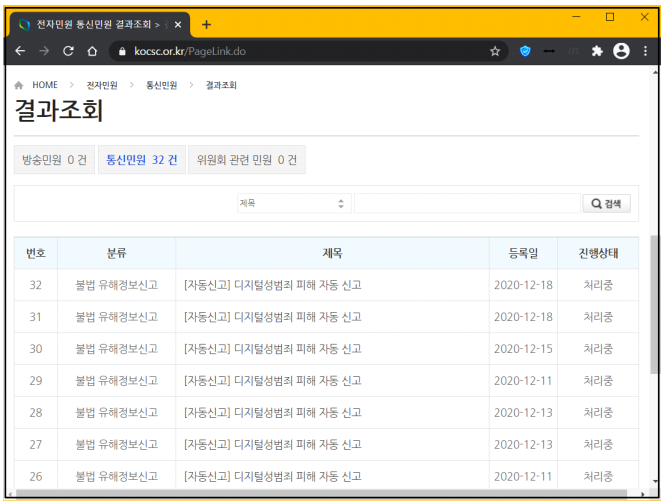


[그림 5-7] 미디어 파일 신고 화면

미디어 파일 신고 시, VID를 추출하여 API 서버에 전송한 다음, 업로드 되었던 미디어 파일을 즉각 폐기하도록 설계하였습니다.

5.1.4 신고 대상 추적

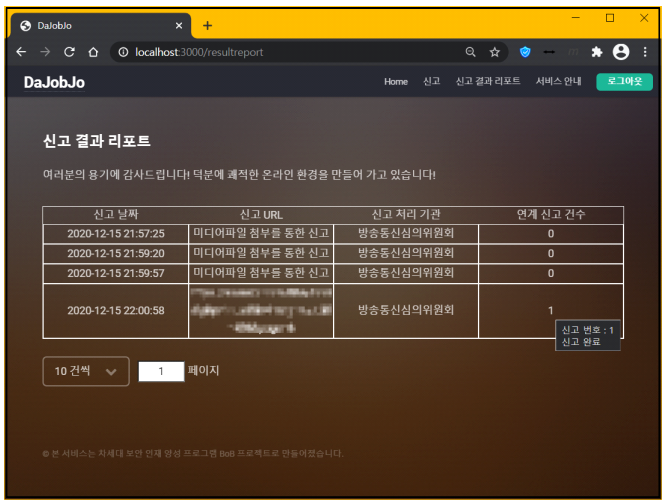
신고 데이터가 API 서버로 전달되고 난 다음, 미디어 파일 신고를 통해 전달 된 데이터는 유해 미디어 탐색 크롤러가 해당 미디어 파일과 유사한 미디어 영상을 게시한 페이지를 찾을 때 마다 해당 페이지를 곧장 방송통신심의위원회로 연계 신고 되도록 설계하였으며, URL 신고를 통해 전달된 데이터는 방송통신심의위원회에 신고 된 URL 및 미디어 링크를 신고함과 동시에, 미디어 파일 신고와 동일하게 앞으로 유해 미디어 탐색 크롤러가 신고 URL 속 영상과 유사한 영상이 게시된 사이트를 발견하는 족족 방송통신심의위원회로 신고하도록 설계하였습니다.



[그림 5-8] 방송통신심의위원회 신고 결과 페이지

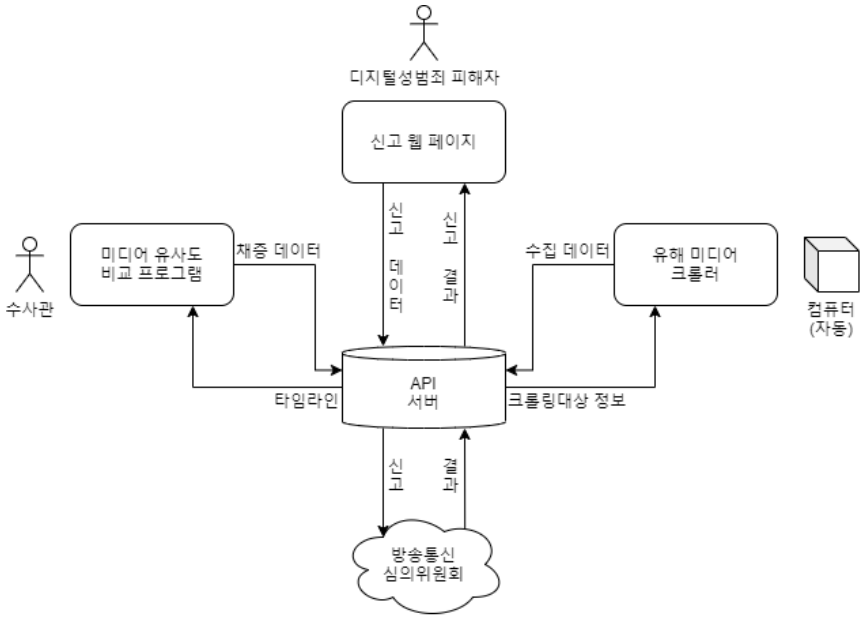
5.2 결과

신고 이후 결과페이지에서 신고 내역과 신고 결과를 열람할 수 있도록 하였으며, 신고자의 신고 덕분에 유해 미디어 탐색 크롤러가 연계 신고한 유사 영상의 수가 몇 개이며, 신고 결과가 어떻게 되었는지에 대한 정보까지 출력하도록 하였습니다.



[그림 5-9] 신고 결과 확인 페이지

제 6 장 API 서버 개발



[그림 6-1] API 서버 동작 알고리즘

6.1 API 서버

6.1.1 개발

API 서버는 신고 웹 서비스와 동일하게 NodeJS기반으로 만들었으며, 복잡한 연산이 필요한 작업은 Python을 통해 처리하도록 설계하였습니다.

API 서버의 본질은 데이터의 공유와 처리였기에, '사용자' 개념이 없으므로 따로 홈 페이지를 만들지 않았으며, 오직 프로토콜에 의한 데이터 수집/전송만을 이행하도록 제작하였습니다.

6.1.2 백그라운드 작업

API 서버가 다양한 서비스로부터 데이터를 수집하는데, 수집과 동시에 해당 데이터를 연산하여 적합한 위치에 놓아야한다면 상당히 느린 응답 속도를 가지게 될 것이 뻔했습니다.

이에 메인 스레드는 각 서비스로부터 들어오는 신호를 통해 데이터를 저장하기만 하고, Python으로 짜여진 스크립트가 백그라운드에서 데이터에 대한 연산 및 정렬 작업을 진행하도록 하였습니다. 예를 들어 메인 스레드는 신고 웹 서버로부터 계속해서 신고 데이터만을 DB에 저장하고, Python 코드가 5분에 1회 DB에 쌓인 모든 데이터를 분석하여 방송통신심의위원회에 신고하는 것처럼 말이죠.

```
const { spawn } = require("child_process");
// PC내의 python.exe가 위치한 디렉터리의 절대 경로를 작성해야 함!
// 추가로, grouper.py에서 요구하는 라이브러리가 모두 갖춰져 있어야 함!
const grouperChildProcess = spawn("C:/Users/GoldBigDragon/anaconda3/python", [
  "./grouper.py",
]);
const reporterChildProcess = spawn("C:/Users/GoldBigDragon/anaconda3/python", [
  "./reporter.py",
]);
```

[그림 6-2] 복잡한 연산을 Python으로 처리하는 모습

6.2 상호작용

6.2.1 미디어 유사도 비교 프로그램

API서버는 미디어 유사도 비교 프로그램으로부터 추출된 VID를 수집 하며, 미디어 유사도 비교 프로그램이 특정 VID에 대한 추적 데이터를 요청 할 경우, 유해 미디어 탐색 크롤러로 부터 받은 URL 데이터와 결합하여 특정 유해 미디어에 대한 추적 타임라인을 제공 해 줍니다.

6.2.2 유해 미디어 크롤러

유해 미디어 크롤러가 수집 한 VID 및 사이트 정보를 수집하여 추적

지표를 생성하며, 크롤러로부터 수집 된 VID와 신고 웹 페이지로부터 신고 받았던 VID가 유사하다고 판정 날 경우, 해당 내용을 방송통신심의위원회에 자동 신고하게 됩니다.

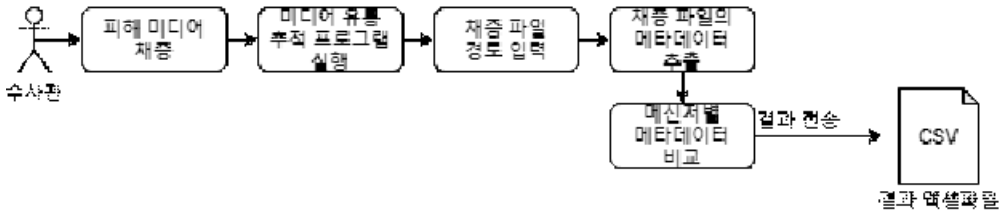
6.2.3 신고 웹 페이지

신고 웹 페이지로부터 신고 데이터를 수집하여 자동으로 방송통신심의위원회에 신고하며, 신고 된 VID 정보를 저장함으로 써 유사 미디어 탐색 시 사용합니다.

6.2.4 오픈채팅방 로깅 봇

오픈채팅방 로깅 봇으로부터 전달 받은 메시지 데이터를 가명 처리 시킨 다음, Hash함수를 이용하여 실제 사용자 정보를 알 수 없도록 저장합니다.

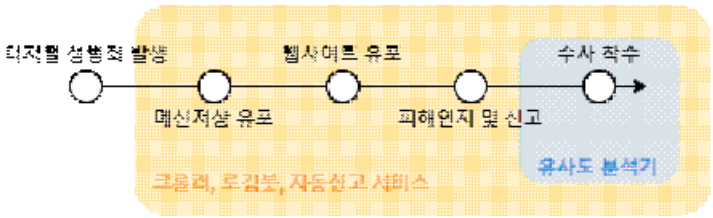
제 7 장 미디어 유통 추적 프로그램 개발



[그림 7-1] 미디어 유통 추적 프로그램 동작 알고리즘

7.1 미디어 유통 추적

미디어 유통 추적은 원래 계획에는 없었으나, 프로젝트 시작 직후 김종민 멘토님께서 디지털 성범죄 타임라인을 그려 주시며, 메신저 상 유포 단계에서 수사관을 도울 수단으로 추천 해 주셨었습니다.



[그림 7-2] 디지털성범죄 타임라인

미디어 유통 추적의 경우 크롤러, 로깅봇, 자동 신고 서비스, 유사도 분석기, API서버와 상호작용하지 않는 독립적인 분야였기 때문에, 전유민, 허원무 멘티님께서 미디어 유통 추적 연구를 전담하시게 되셨으며, 추 후 해외 논문 작성을 위하여 특허를 진행 중이시던 이소민 멘티님께서 간간히 논문 번역을 도왔습니다.

미디어 유통 추적은 특정 OS, 휴대폰 기종, 메신저 등을 통해 공유된 미디어로부터 고유 메타 데이터를 찾아 채증 된 미디어의 유포지 출처를 알아내는 것이 목적으로, 국내외 논문에서는 카카오톡 혹은

텔레그램 등 단일 메신저에 대한 연구만 진행되었을 뿐, iOS 환경의 iPhone6 모델에서 찍은 사진을 KakaoTalk으로 전송하고, KakaoTalk으로부터 전송 받은 사진을 Telegram으로 공유하는 등 N차 유포에 대한 연구가 진행되지 않은 상황이었기에 충분히 연구 가치가 있다고 판단하였으며, 연구와 논문 제출[21]만으로 끝낼 것이 아니라, 해당 내용을 프로그램으로 구현함으로 써 수사에 도움을 주는 것이 최종 목표였습니다.

7.2 미디어 유통 추적 프로그램

논문 실험 내용을 바탕으로 필자는 CLI 프로그램을 제작하였으며, iOS, Android 환경에서 각 메신저로 부터 전송 받은 미디어를 실험 대상으로 지정하였습니다.

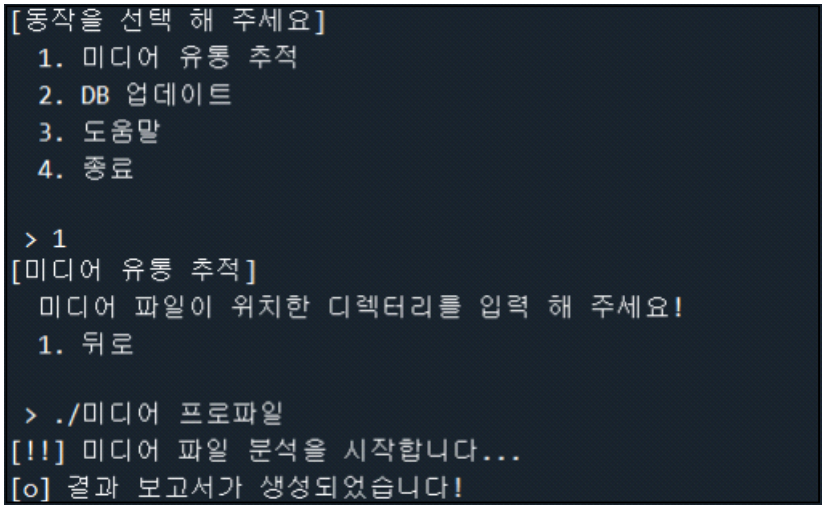
메신저	iOS 버전	Android 버전
Discord	49.0	49.16
Facebook	297.0	297.0.0.36.116
Facebook Messenger	291.2	291.2.0.22.114
Instagram	167.0	167.1.0.25.120
KakaoTalk	9.1.2	9.1.3
LINE	10.19.0	10.20.1
NateOn	4.0.5	4.0.8
Skype	8.66.76	8.66.0.76
Signal	3.22.0	4.78.5
Telegram	7.2.1	7.2.1
Wechat	7.0.18	7.0.17
WhatsApp	2.20.121	2.20.205.16
Wickr Me	5.66.10	5.66.8

[표 7-1] 실험 메신저 버전

메신저 및 디바이스 OS가 패치 될 경우를 고려하여 필자는 메타데이터 프로파일 추출기를 먼저 제작하였으며, 확장성을 생각하여 정해진

21) 미디어 유통과정 추적 연구 : https://tinyurl.com/y8svlrv6

디렉터리 구성 규칙에만 맞춘다면, 표에 없는 메신저나 디바이스도 프로파일에 추가할 수 있도록 DB 업데이트 기능을 추가하였습니다.



[그림 7-3] CLI 구동 모습

	A	B	C	D	E	F	G	H	I	J	K	L
1	순번	경로	이름	확장자	가로	세로	촬영일	OS	모델	메신저	추가 의심대상	
2	1	Android 6.0.1/Note5/Instagram/가로.mp4	가로	mp4	480	270		Android 6.0.1	Note5	Instagram	{Android 6.0.1::S7::Instagr	
3	2	Android 6.0.1/Note5/Facebook Messenger/가로.mp4	가로	mp4	640	368		Android 6.0.1	Note5	Facebook Messenger		
4	3	Android 6.0.1/Note5/Instagram/세로.mp4	세로	mp4	480	852		Android 6.0.1	Note5	Instagram	{Android 6.0.1::S7::Instagr	
5	4	Android 6.0.1/Note5/Facebook Messenger/세로.mp4	세로	mp4	368	656		Android 6.0.1	Note5	Facebook Messenger		
6	5	Android 6.0.1/Note5/Facebook/해상도불규칙 가로.mp4	해상도불규칙	mp4	1280	720		알 수 없음	알 수 없음	알 수 없음		
7	6	Android 6.0.1/Note5/Facebook/해상도불규칙 세로.mp4	해상도불규칙	mp4	720	1280		알 수 없음	알 수 없음	알 수 없음		
8	7	Android 6.0.1/Note5/Discord/세로.mp4	세로	mp4	1920	1080	2020-11-05	Android 6.0.1	Note5	Discord	{Android 6.0.1::Note5::Dis	
9	8	Android 6.0.1/Note5/Discord/가로.mp4	가로	mp4	1920	1080	2020-11-05	Android 6.0.1	Note5	Discord	{Android 6.0.1::Note5::Dis	
10	9	Android 6.0.1/Note5/KakaoTalk/카톡 가로.mp4	카톡 가로	mp4	852	480		Android 6.0.1	Note5	KakaoTalk	{Android 6.0.1::Note5::Kak	
11	10	Android 6.0.1/Note5/KakaoTalk/카톡 세로.mp4	카톡 세로	mp4	852	480		Android 6.0.1	Note5	KakaoTalk	{Android 6.0.1::Note5::Kak	
12	11	Android 6.0.1/Note5/LINE/원본 가로.mp4	원본 가로	mp4	960	540	2020-11-05	Android 6.0.1	Note5	LINE	{Android 6.0.1::Note5::LIN	
13	12	Android 6.0.1/Note5/LINE/원본 세로.mp4	원본 세로	mp4	540	960	2020-11-05	Android 6.0.1	Note5	LINE	{Android 6.0.1::Note5::LIN	
14	13	Android 6.0.1/Note5/LINE/일반 가로.mp4	일반 가로	mp4	960	540	2020-11-05	Android 6.0.1	Note5	LINE	{Android 6.0.1::Note5::LIN	
15	14	Android 6.0.1/Note5/LINE/일반 세로.mp4	일반 세로	mp4	540	960	2020-11-05	Android 6.0.1	Note5	LINE	{Android 6.0.1::Note5::LIN	
16	15	Android 6.0.1/Note5/NateOn/가로.mp4	가로	mp4	1920	1080	2020-11-05	Android 6.0.1	Note5	Discord	{Android 6.0.1::Note5::Dis	
17	16	Android 6.0.1/Note5/NateOn/세로.mp4	세로	mp4	1920	1080	2020-11-05	Android 6.0.1	Note5	Discord	{Android 6.0.1::Note5::Dis	
18	17	Android 6.0.1/Note5/Signal/가로.mp4	가로	mp4	1920	1080	2020-11-05	Android 6.0.1	Note5	Discord	{Android 6.0.1::Note5::Dis	
19	18	Android 6.0.1/Note5/Skype/가로.mp4	가로	mp4	1280	720	2020-11-05	Android 6.0.1	Note5	Skype	{Android 6.0.1::S7::Signal}	
20	19	Android 6.0.1/Note5/Skype/세로.mp4	세로	mp4	720	1280	2020-11-05	Android 6.0.1	Note5	Skype	{Android 6.0.1::S7::Signal}	
21	20	Android 6.0.1/Note5/Signal/세로.mp4	세로	mp4	1920	1080	2020-11-05	Android 6.0.1	Note5	Discord	{Android 6.0.1::Note5::Dis	
22	21	Android 6.0.1/Note5/Telegram/고화질 세로.mp4	고화질 세로	mp4	1080	1920	2020-11-05	Android 6.0.1	Note5	Telegram	{Android 6.0.1::Note5::Tel	

[그림 7-4] 출력된 결과 CSV 파일

제 8 장 결과

8.1 산출물

산출물	비고
수행 계획서	
WBS (개발 계획표)	
주간 보고서	총 16주
멘토링 일지	총 30건
프로젝트 결과 보고서	
미디어 유통과정 추적 연구 논문[22]	
오픈 채팅방을 이용한 디지털 성범죄 수사 정책 제안 논문[23]	
인터뷰 보고서 (사이버성폭력수사팀, 여성청소년수사팀, BoB 09기 선배님)	총 3건
요구사항 명세서[24]	
Usecase 명세서[25]	
테스트 시나리오[26]	
이미지 유사도 판별 방법 조사 보고서[27]	
피해 신고 웹 사이트 사용법	

[표 8-1] 산출물 목록

프로젝트 종료(12월 19일) 이후, 12월 23일 까지 작성하여 제출한 산출물로는 연구 보고서, 투고 논문, 수행 계획서 등이 포함되었으며, 미디어 유사도 분석기는 pyinstaller를 통해 exe 파일로 추출하여 실행 가능한 파일로 제작하고, 신고 웹 서버는 codns 서비스를 이용하여 dajobjo.codns.com 주소를 통해 접근 가능하도록 하였습니다.

8.2 문제점 및 향후 개발 방향

22) 미디어 유통과정 추적 연구 : https://tinyurl.com/y8svlrv6
23) 오픈 채팅방을 이용한 디지털 성범죄 수사 정책 제안 논문 : https://tinyurl.com/ya54c9jt
24) 요구사항 명세서 : https://tinyurl.com/y7rn22n4
25) UseCase 명세서 : https://tinyurl.com/ycnhrljy
26) 테스트 시나리오 : https://tinyurl.com/ya7pof52
27) 이미지 유사도 판별 방법 조사 보고서 : https://tinyurl.com/yaqhm5v9

8.2.1 미디어 유사도 분석기

유사도 판단에 대한 정확도가 86.8%로, 기준치 95%보다 8.2% 부족하기 때문에 리사이즈 영상에 대한 분석 방법을 연구 해 나갈 것입니다. 현재 그 원인이 원본 리사이즈 영상과 배경 제거 이후의 리사이즈 영상 간 iframe 간격이 다르기 때문으로 판단하고 있으며, 동일 iframe 간격을 적용한 실험을 진행 할 예정입니다.

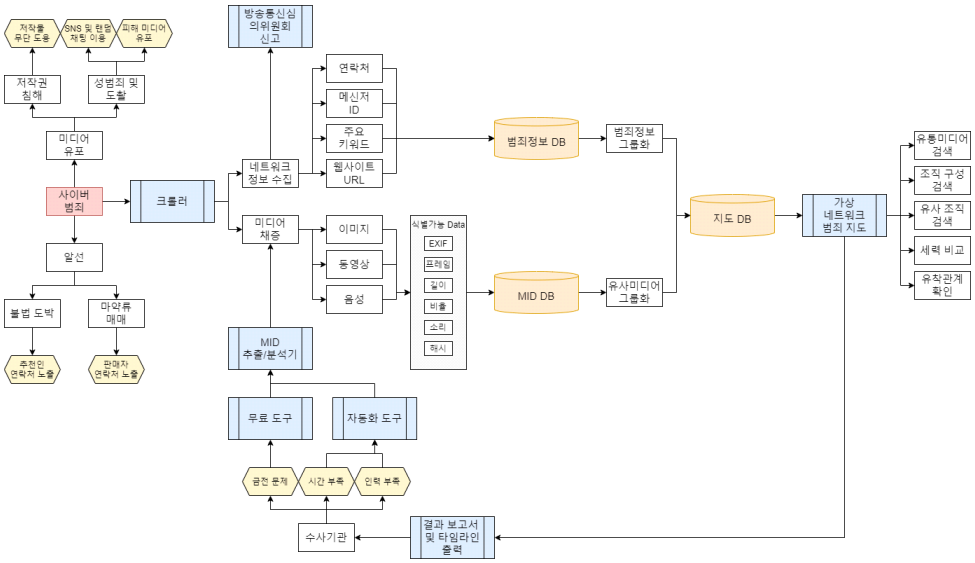
8.2.2 피해 신고 웹 페이지

피해자가 영상 혹은 URL을 신고할 때, 해당 미디어가 실제 피해 영상인지에 대한 구분이 불가능한 상태이며, 구분을 하게 된다 하더라도 인식을 위한 막대한 데이터와 함께 개인정보 처리까지 필요로 하게 될 수 있습니다. 하지만 익명 신고가 원칙이기 때문에 개인정보를 수집 할 수가 없으며, 그로인해 거짓 신고에 대한 구체적인 대응책을 세울 수 없습니다.

사용자의 양심에 전적으로 의존해야하는 현 상황에서, 실제 피해자만을 대상으로 서비스하게 될 경우, 그 모집단은 어디서 구할 것이며, 어떤 기관과 연계하여 서비스해야할지 모든 부분에 대한 연구가 필요합니다.

8.2.3 가상 범죄지도 제작

미디어 유사도 분석기 및 유해 미디어 탐색 크롤러, 신고 웹 페이지의 개발이 완료되었으므로, 수집되는 데이터를 이용하여 원래 만들고자했던 가상 범죄지도 시각화 프로그램을 제작 해 나갈 계획입니다.



[그림 8-1] 가상 범죄지도 동작 알고리즘