

2025 HUSS AI 경진대회 출품작

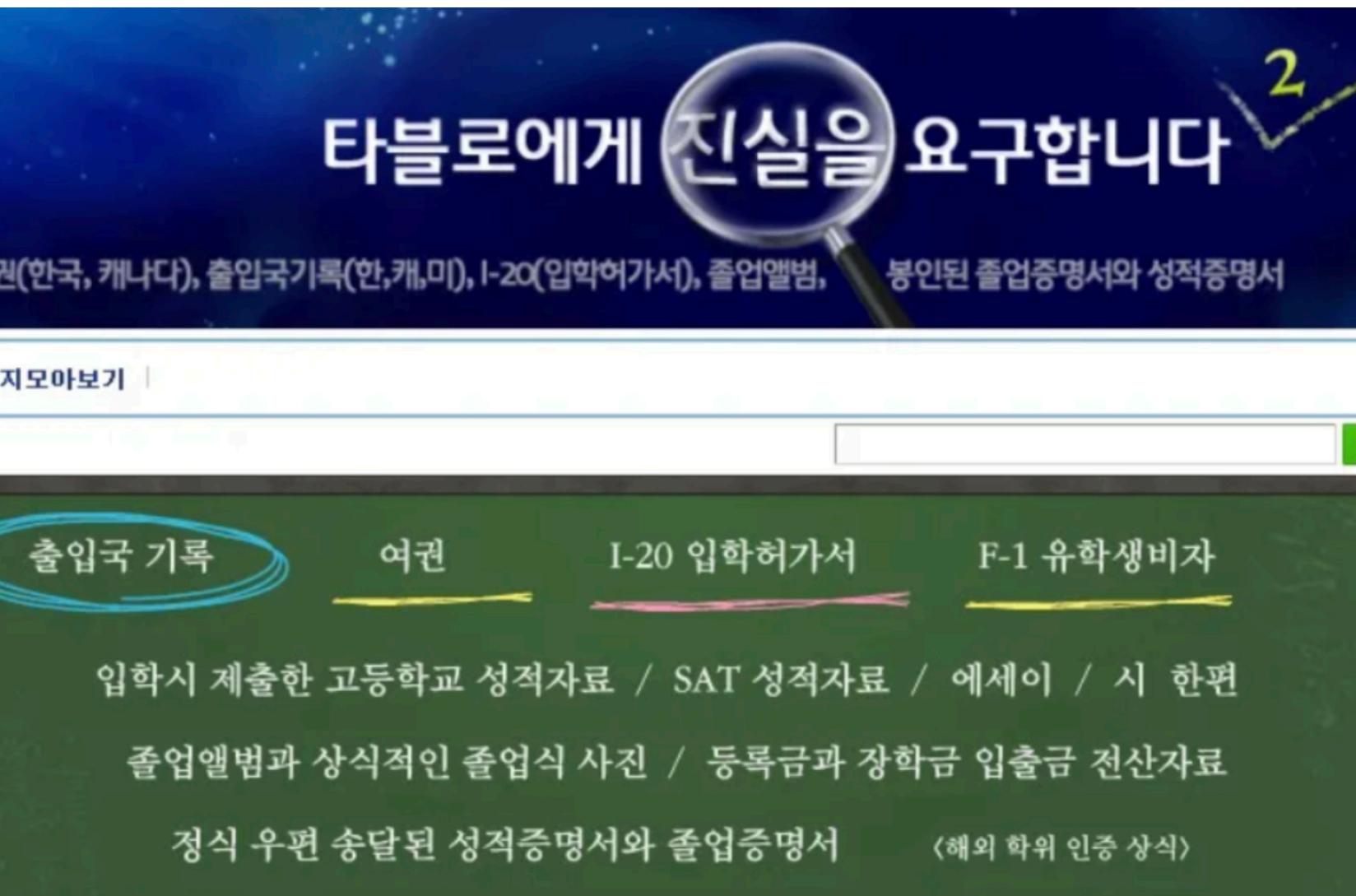
# Info-Guard

정보 신뢰성 진단 AI 알고리즘



광운대학교 | 정만교 조유진 이교원 이준희

# Intro



경향신문

## 타블로 비방 타진요 회원들 실형

서울중앙지법 형사14단독 곽윤경 판사는 6일 명예훼손 혐의로 불구속 기소된 타진요 회원 원모 씨와 이모씨에게 각각 징역 10월을 선고하고 법정구속했다.

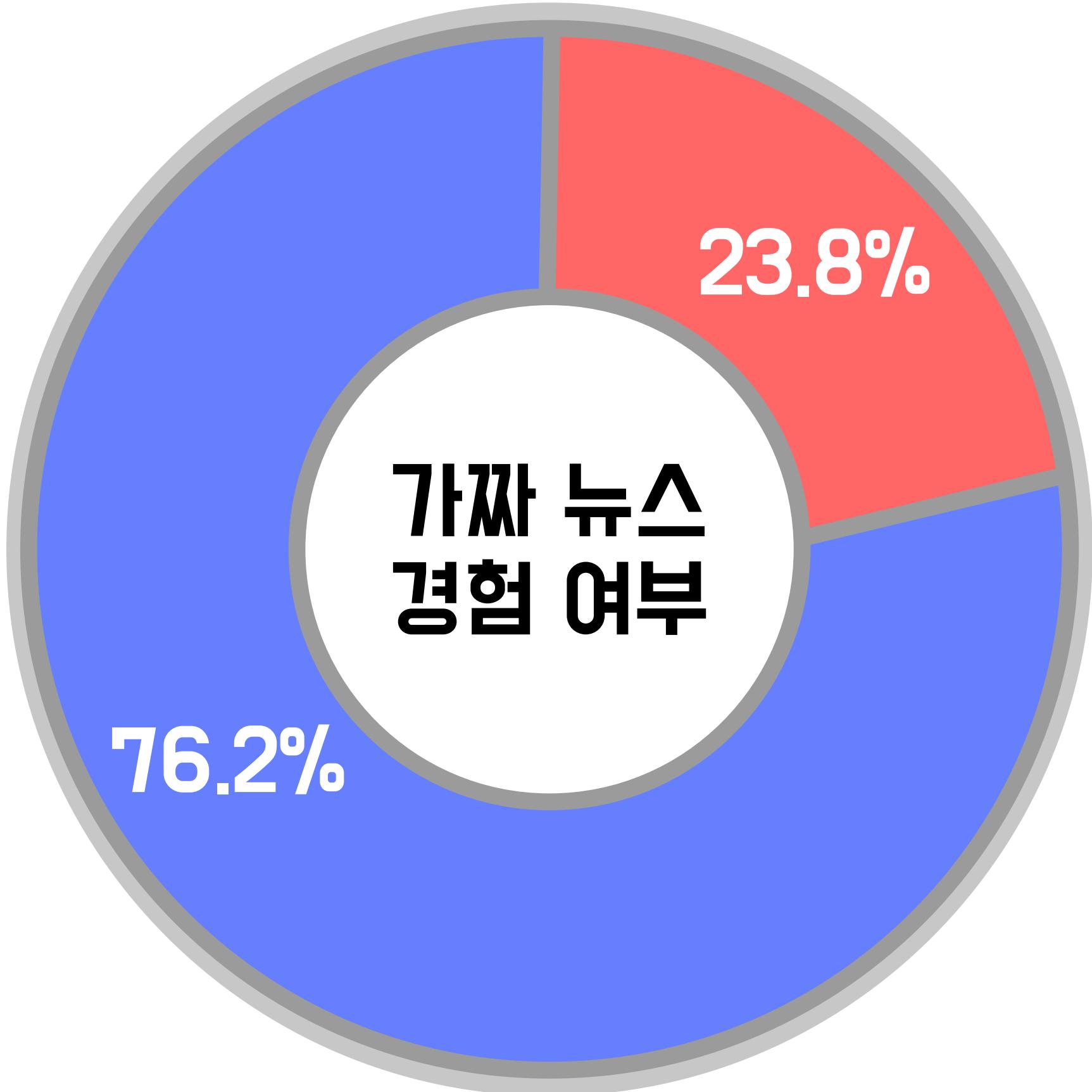
2012. 7. 6.

초기에 의혹이 퍼질 때. 정보를 접하는 사람들에게  
이거 사실이 아닐 수 있다고 알려주는 장치가 있었다면 어땠을까?

# 과제 1. 가짜 뉴스 문제

자극적인 썸네일과 출처 없는 '가더라'성 정보가 SNS, 유튜브 등으로 빠르게 확산되고 있음.

AI 기술의 발전에 따라 점점 교묘해지고 있음.



한국언론진흥재단 (2023)

# 과제 2. 과장된 감정 편향으로 인한 선동 문제



지금까지 논란 안터진게 이상하다는, 피  
식대학 인성 드러나는 순간들

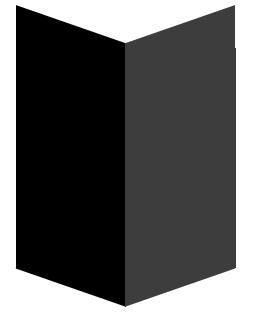
395K views • 1 year ago



**분노**      **혐오**      **과도한 찬양**

→ 사회 갈등, 혐오 조장 등의 **공공 문제** 야기 가능성

# Info-Guard



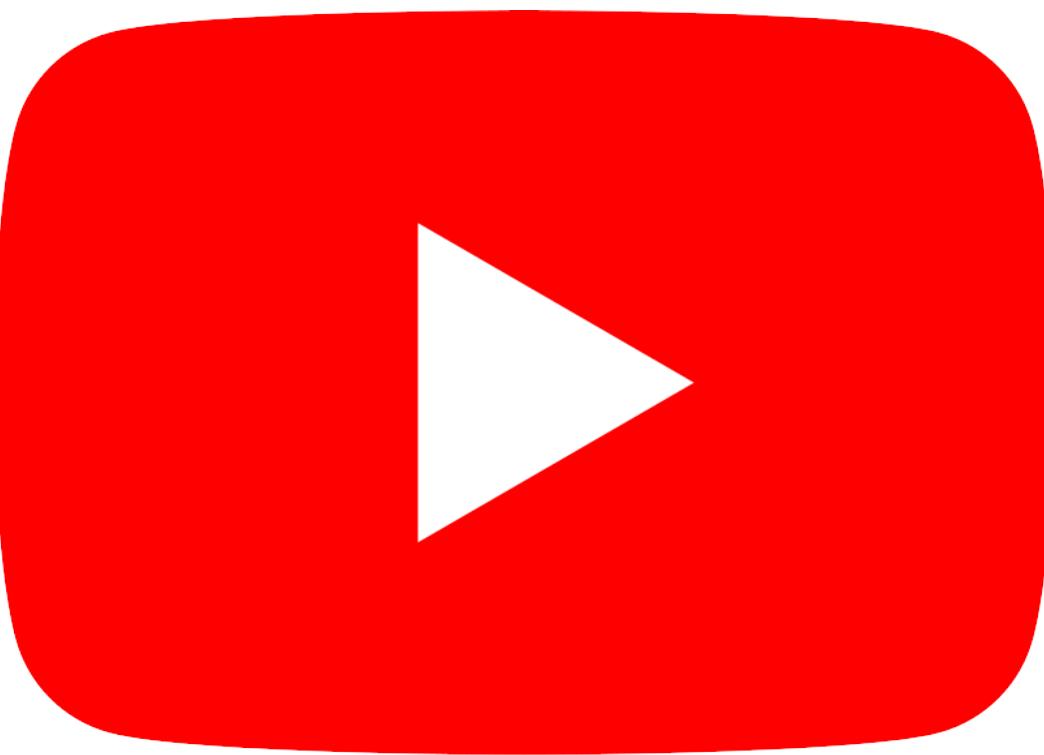
## 정보 신뢰성 진단 AI 알고리즘

---

가짜 뉴스 및 선동에 취약한 정치, 경제, 연예 분야를 대상으로 **유튜브** 영상 신뢰도 평가

정보의 왜곡과 선동성 컨텐츠로부터 사용자를 보호

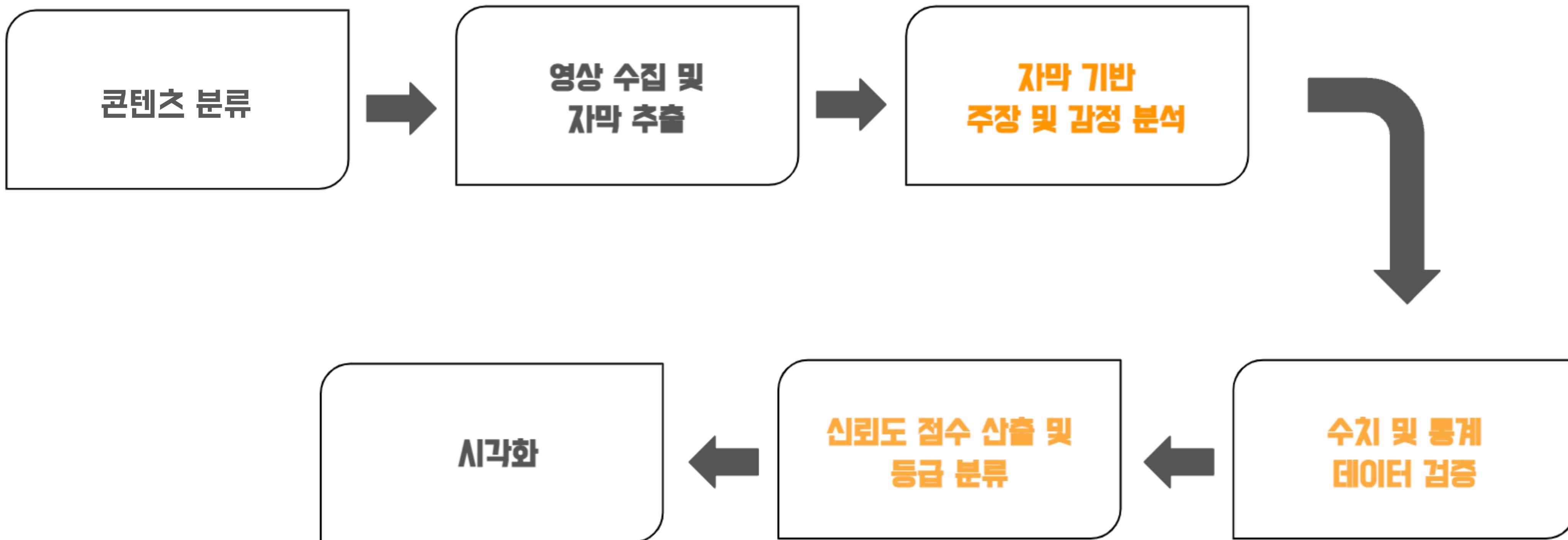
# 왜 유튜브인가?



# YouTube

- 2024년 1월 기준 **한국인 1인당 평균 사용시간 40시간**으로 압도적 1위
- 뉴스 소비 플랫폼의 역할을 향과 동시에 **정보 유통의 핵심 경로**
- **전 세대에 걸친 사용 연령**

# Info-Guard 핵심 기능 및 워크플로우



# 경제 예시

**“테슬라 주가가 일주일 동안 35% 넘게 상승했습니다.  
이는 전례 없는 일입니다.”**

# 경제 예시 (워크플로우)

## 콘텐츠 분류

'경제', '주식' ...

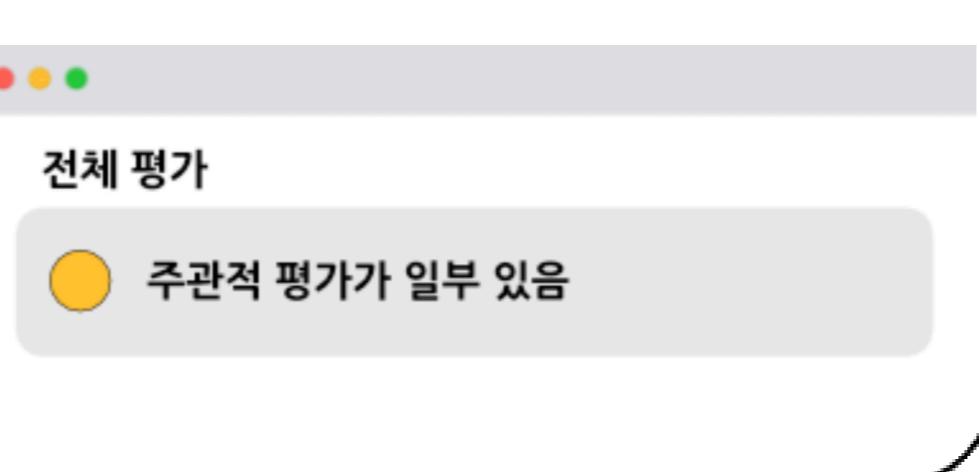
## 영상 수집 및 자막 추출

테슬라. 35%.  
일주일...

## 주장 및 감정 분석

감정 분석 시행

## 시각화



## 등급 분류

팩트 체커 점수 낮음  
**낮은 점수 부여**

## 데이터 검증

실제 금융 데이터  
분석 및 대조

# 콘텐츠 분류

```
from bs4 import BeautifulSoup
import requests

def extract_youtube_tags(video_url):

    headers = {'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.0.0 Safari/537.36'}
    response = requests.get(video_url, headers=headers)

    if response.status_code != 200:
        print('영상 페이지 접근 실패.')
        return []

    soup = BeautifulSoup(response.text, 'html.parser')
    keywords_tag = soup.find('meta', {'name': 'keywords'})

    if keywords_tag:
        tags = keywords_tag.get('content', '').split(',')
        return tags
    else:
        print('태그가 존재하지 않거나 추출 불가')
        return []

video_url = 'https://www.youtube.com/watch?v=dQw4w9WgXcQ'
```

1. 유튜브 태그를 통한 주제 분류
2. 정보 신뢰성 검증이 필요한 영상 타겟  
↳ 신뢰성 검증이 필요없는 영상 제외

## 실행 결과 예제

```
['MBC', 'MBC뉴스', '뉴스데스크', 'newsdesk', '뉴스투데이', 'newstoday', '8시 뉴스', '아침 뉴스', '뉴스', '정오 뉴스', 'news', '실시간', '실시간 뉴스', '엠비씨 뉴스', '엠비씨', '뉴스 실시간', '뉴스 속보', 'K리그', '전남', '오프사이트', 'VAR', '심판', '정강민', '오심 논란', '축구 협회']
```

# 텍스트 추출 - 영상 수집 및 자막 추출

1. 유튜브 API를 활용해 우선적으로 자막 추출
2. 생성된 자막이 없을 경우 **yt-dlp** 라이브러리 사용  
↳ 상대적으로 시간이 오래 걸리므로 나중에 수행



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

새로운 기능 및 개선 사항에 대한 최신 PowerShell을 설치하세요! https://aka.ms/PSWindows

PS C:\Users\xcrui> yt-dlp -x --audio-format mp3 --audio-quality 192K "https://www.youtube.com/watch?v=t_Y90Kb2Wts"
yt-dlp 라이브러리 사용 예제
```

# 분석 엔진 - 감정 분석

- 모델: KLUE BERT-base (한국어 특화)
- 목적: 영상 콘텐츠의 감정적 톤 분석
- 라벨: positive, neutral, negative

```
▶ sentiment_predict('처음 써봤는데 대박 좋아요.')
sentiment_predict('원래 배송이 이렇게 늦나요?')
sentiment_predict('좋은거 인정! 추가 구매 의향 있습니다.')
sentiment_predict('이건 정말 리뷰 쓰는게 아깝네요.')
```

⇨ 처음 써봤는데 대박 좋아요. -> 긍정(98.25%)
원래 배송이 이렇게 늦나요? -> 부정(94.65%)
좋은거 인정! 추가 구매 의향 있습니다. -> 긍정(96.48%)
이건 정말 리뷰 쓰는게 아깝네요.. -> 부정(97.94%)

## 2.2 감정 분석 모델 (app/ai/sentiment.py)

```
# 구현 위치: src/python-server/app/ai/sentiment.py
# 구현 클래스: SentimentAnalyzer

from .base import BaseAIModel
from transformers import pipeline
from typing import Dict, Any

class SentimentAnalyzer(BaseAIModel):
    def __init__(self):
        """감정 분석 모델 로드"""
        self.analyzer = pipeline(
            "sentiment-analysis",
            model="sbert/bert-base-multilingual-uncased-sentiment",
            device=0 if self.device == "cuda" else -1
        )

    def predict(self, text: str) -> Dict[str, Any]:
        """감정 분석"""
        processed_text = self.preprocess_text(text)
        result = self.analyzer(processed_text)

        # 감정 점수 계산 (1-5 스케일을 0-100으로 변환)
        score = float(result[0]['label'].split()[0]) * 20

        return {
            "sentiment_score": score,
            "sentiment_label": result[0]['label'],
            "confidence": result[0]['score'],
            "text": text
        }
```

GPU 가속 사용하여  
감정분석 모델 로딩

데이터 전처리 및 분석  
0~100점으로 신뢰도 측정

# 분석 엔진 - 감정 분석

## AI 모델 사용 프로토타입

하고싶은 말을 입력해주세요 : 어두운게 너무 무섭더라  
/usr/local/lib/python3.7/dist-packages/torch/utils/data/dataloader.py:477:  
cpuset\_checked)  
>> 입력하신 내용에서 **공포**가 느껴집니다.

하고싶은 말을 입력해주세요 : 주식이 엄청 올랐더라?  
>> 입력하신 내용에서 **놀람**이 느껴집니다.

하고싶은 말을 입력해주세요 : 마스크 때문에 너무 짜증나  
>> 입력하신 내용에서 **혐오**가 느껴집니다.

하고싶은 말을 입력해주세요 : 코로나는 대체 언제 없어지는거야?  
>> 입력하신 내용에서 **놀람**이 느껴집니다.

하고싶은 말을 입력해주세요 : 요즘 화가 많이 나  
>> 입력하신 내용에서 **분노**가 느껴집니다.

하고싶은 말을 입력해주세요 : 어제 헤어졌어ㅜ  
>> 입력하신 내용에서 **슬픔**이 느껴집니다.

하고싶은 말을 입력해주세요 : 우산 챙겨가~  
>> 입력하신 내용에서 **중립**이 느껴집니다.

하고싶은 말을 입력해주세요 : 오늘 날씨가 너무 맑다!  
>> 입력하신 내용에서 **행복**이 느껴집니다.

하고싶은 말을 입력해주세요 : 으 벌레 너무 싫어  
>> 입력하신 내용에서 **혐오**가 느껴집니다.

### 1. 감정 분석 모델 (Sentiment Analysis)

```
# 구현 위치: src/python-server/app/ai/sentiment.py
# 구현 클래스: SentimentAnalyzer

class SentimentAnalyzer(BaseModel):
    def __init__(self):
        self.model = self._load_model()
        self.tokenizer = self._load_tokenizer()

    def analyze(self, text: str) -> Dict[str, float]:
        """텍스트 감정 분석"""
        inputs = self.tokenizer(text, return_tensors="pt", truncation=True, max_length=512)
        outputs = self.model(**inputs)

        # 감정 점수 계산 (-100 ~ 100)
        sentiment_score = self._calculate_sentiment_score(outputs)

        return {
            "sentiment_score": sentiment_score,
            "confidence": self._get_confidence(outputs),
            "emotions": self._extract_emotions(outputs)
        }

    def _calculate_sentiment_score(self, outputs) -> float:
        """감정 점수 계산"""
        logits = outputs.logits
        probs = torch.softmax(logits, dim=-1)

        # 긍정/부정/중립 확률
        positive_prob = probs[0][1].item()
        negative_prob = probs[0][2].item()
        neutral_prob = probs[0][0].item()

        # -100 ~ 100 점수로 변환
        score = (positive_prob - negative_prob) * 100
        return round(score, 2)
```

# 분석 엔진 - 편향성 감지

감지 요소	신뢰도 외의 관계	예시
키워드 편향	<b>특정 단어, 표현이 반복 되어 프레임을 유도시</b> <b>사실성보다 설득, 선동 목적 가능성 ↑</b>	<b>"범죄자", "불법" 등</b> <b>→ 부정적인 키워드 반복</b>
정치 편향	<b>특정 정치 성향으로 치우친 정보는 객관성 ↓</b> <b>신뢰도 평가 시 감점요소</b>	<b>"OO 정권은 나라를 망쳤다"</b> <b>→ 한쪽만 비판한 경우</b>
언어 편향	<b>특정 집단, 지역, 성별 등에 대한 차별, 고정관념이 드러나면 공정성 ↓</b>	<b>"그들은 원래 게으르다"</b>

## 2. 편향 감지 모델 (Bias Detection)

```
# 구현 위치: src/python-server/app/ai/bias.py
# 구현 클래스: BiasDetector
```

```
class BiasDetector(BaseModel):
    def __init__(self):
        self.bias_keywords = self._load_bias_keywords()
        self.political_terms = self._load_political_terms()
```

```
def detect_bias(self, text: str) -> Dict[str, Any]:
    """편향성 감지"""
    bias_score = 0
    bias_types = []
```

```
# 키워드 기반 편향 감지
keyword_bias = self._detect_keyword_bias(text)
bias_score += keyword_bias['score']
bias_types.extend(keyword_bias['types'])
```

```
# 정치적 편향 감지
political_bias = self._detect_political_bias(text)
bias_score += political_bias['score']
bias_types.extend(political_bias['types'])
```

```
# 언어적 편향 감지
linguistic_bias = self._detect_linguistic_bias(text)
bias_score += linguistic_bias['score']
bias_types.extend(linguistic_bias['types'])
```

```
return {
    "bias_score": max(-100, min(100, bias_score)), # -100 ~ 100
    "bias_types": list(set(bias_types)),
    "confidence": self._calculate_confidence(bias_score),
    "details": {
        "keyword_bias": keyword_bias,
        "political_bias": political_bias,
        "linguistic_bias": linguistic_bias
    }
}
```

```
def _detect_keyword_bias(self, text: str) -> Dict[str, Any]:
    """키워드 기반 편향 감지"""
    bias_score = 0
    bias_types = []
```

**키워드 편향**

**정치 편향**

**언어 편향**

# 분석 엔진 - 팩트 체커

- 원문 분석
- 발언의 문구와 맥락 확인
- 정확한 수치 확인
- 근거 확인
- 사실 여부 확인

## 경제 예시

### Fake Text Example

"테슬라 주가가 이번 달에 35% 넘게 상승했습니다. 이는 전례 없는 일입니다."

#### 1. 음성 텍스트 추출

#### 2. 문장 분석

- 상호 : 테슬라 (Tesla)
- 수치 : 35%. 상승
- 기간 조건 : 이번 달

#### 3. 데이터 검증

- Yahoo Finance, Alpha vantage 등 금융 API

#### 4. AI 모델 활용 점수화



## 4. 팩트 체커 (Fact Checker)

```
class FactChecker(BaseModel):
    def __init__(self):
        self.verification_sources = self._load_verification_sources()
        self.fact_patterns = self._load_fact_patterns()

    def check_facts(self, text: str) -> Dict[str, Any]:
        """사실 여부 검증"""
        facts = self._extract_facts(text)
        verified_facts = []
        unverified_facts = []

        for fact in facts:
            verification_result = self._verify_fact(fact)
            if verification_result['verified']:
                verified_facts.append(verification_result)
            else:
                unverified_facts.append(verification_result)

        # 팩트 체크 점수 계산
        fact_check_score = self._calculate_fact_check_score(verified_facts, unverified_facts)

        return {
            "score": fact_check_score,
            "verified_facts": verified_facts,
            "unverified_facts": unverified_facts,
            "total_facts": len(facts),
            "verification_rate": len(verified_facts) / len(facts) if facts else 0
        }

    def _extract_facts(self, text: str) -> List[str]:
        """텍스트에서 사실 추출"""
        facts = []

        # 숫자 기반 사실 패턴
        number_patterns = [
            r'(\d+)\s*(명|개|건|%)|원|달러|시간|분|초',
            r'(\d{4})년\s*(\d{1,2})월',
            r'(\d+)월\s*(\d+)일'
        ]

        for pattern in number_patterns:
            matches = re.findall(pattern, text)
            facts.extend([f"숫자 정보: {match}" for match in matches])
```

# 점수 엔진 - 신뢰도 점수 및 등급 분류

감정 분석

편향성 분석

팩트 체크

3가지 분석 점수를 기반으로 최종 평가

## 3. 신뢰도 분석 모델 (Credibility Analyzer)

```
class CredibilityAnalyzer(BaseModel):
    def __init__(self):
        self.fact_checker = FactChecker()
        self.bias_detector = BiasDetector()
        self.sentiment_analyzer = SentimentAnalyzer()

    def analyze_credibility(self, text: str, metadata: Dict[str, Any] = None) -> Dict[str, Any]:
        """종합 신뢰도 분석"""
        results = {}

        # 팩트 체크
        fact_check_result = self.fact_checker.check_facts(text)
        results['fact_check'] = fact_check_result

        # 편향성 분석
        bias_result = self.bias_detector.detect_bias(text)
        results['bias'] = bias_result

        # 감정 분석
        sentiment_result = self.sentiment_analyzer.analyze(text)
        results['sentiment'] = sentiment_result

        # 종합 신뢰도 점수 계산
        overall_score = self._calculate_overall_score(results)

        return {
            "overall_credibility": overall_score,
            "fact_check_score": fact_check_result['score'],
            "bias_score": bias_result['bias_score'],
            "sentiment_score": sentiment_result['sentiment_score'],
            "detailed_analysis": results,
            "confidence": self._calculate_confidence(results)
        }
```

영상 데이터  
분석

종합 신뢰도  
계산

# 점수 언진 - 신뢰도 점수 및 등급 분류

**가중치는 차후 학습하는 데이터를 기반으로 조정 예정**

```
def _calculate_overall_score(self, results: Dict) -> float:  
    """종합 신뢰도 점수 계산"""  
    fact_score = results['fact_check']['score'] * 0.4 팩트 체크 (40%)  
    bias_score = (100 - abs(results['bias']['bias_score'])) * 0.3 편향성 분석 (30%)  
    sentiment_score = (100 - abs(results['sentiment']['sentiment_score'])) * 0.2 감정 분석 (20%)  
    metadata_score = self._calculate_metadata_score(results.get('metadata', {})) * 0.1  
                                영상 출처 분석 (10%. 미구현)  
  
    total_score = fact_score + bias_score + sentiment_score + metadata_score  
    return max(0, min(100, round(total_score, 2)))
```

$$\text{Final} = w_1 \times \text{FactScore} + w_2 \times (1 - \text{BiasScore}) + w_3 \times (1 - \text{EmotionScore})$$

# 시각화

- 안전
- 주의
- 위험

점수에 따른 등급 분류  
분석 문장 개별 등급 구간화

The screenshot shows a YouTube video player interface. At the top, there are three cards with text and icons:

- Chief Scientist / Co-Founder 퇴사 후 컴백
- 전 CEO 퇴사
- 전 CTO / Co-Founder 퇴사

The main video frame shows two men, one in a blue jacket and one in a grey polo shirt, looking at the camera. Below the video is a progress bar at 1:20 / 5:26. A subtitle reads "아예 다른 성격의 회사가 되었습". The video title is "아이온큐는 더 이상 이전의 아이온큐가 아니다." and the channel is "T3chfeed - 테크피드 미국 ...".

A red circle highlights the "분석" button in the video controls. A mouse cursor hovers over this button. To the right, a white box contains the following analysis highlights:

- 전체 평가
- 주관적 평가가 일부 있음
- \$8,200만 매출 상향 조정
- 양자 내성 보안이 갖춰진 위성 군집을 전개하는 것이 목표인 것 같습니다.
- AQ #64는 양자 컴퓨터의 혁신이 될 것
- Example Text # 04

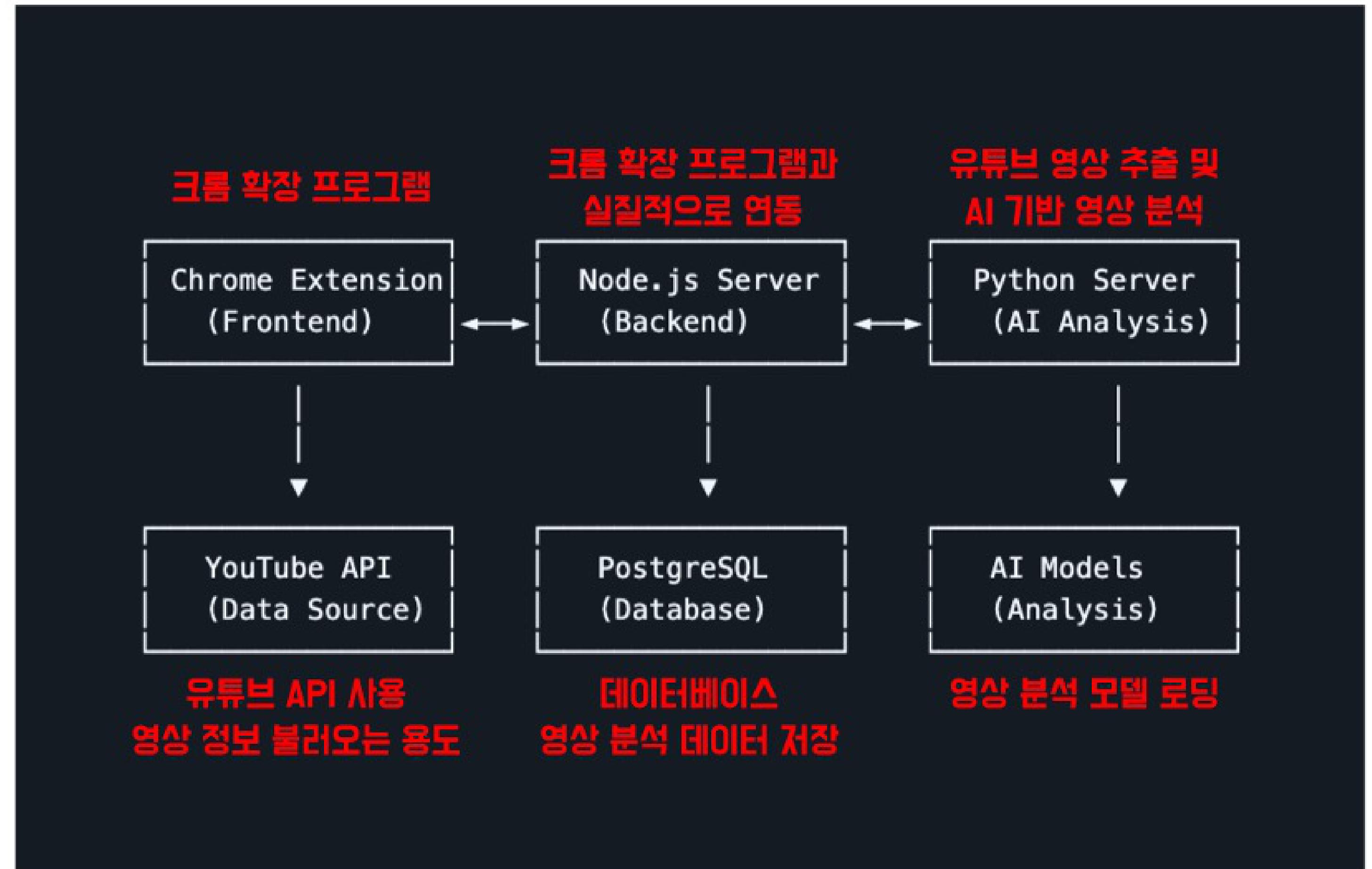
At the bottom of the video player, there is a link to the script: "스크립트" and "스크립트 표시" with the URL "www.youtube.com/watch?v=XOL-DgPelec".

# 배포 계획



chrome web store

크롬 확장 프로그램으로 배포 예정



느니적

# 감사합니다

**Info-Guard**  
정보 신뢰성 진단 AI 알고리즘



광운대학교 | 정만교 조유진 이교원 이준희



깃허브 주소 (QR)