10차 회의

날짜	@2025년 5월 5일
사람	민영 찬주 오 Suyeon Han
장소	멋사 디스코드
회의 URL	https://mignonieee.notion.site/10-1ecef1aece7f81978e7feef6a6bad3ed?pvs=4

공유 및 논의 안건

- [x] ~~AI 구현 방법 논의~~

▼ Meeting Note @2025/05/05

1. AI 구현 방법 논의

지금 상황: 어떤 감정인지는 도출하는건 구현이 됨. but 원인 분석 아직 구현 중 (문장이 길어지면 정확도 떨어짐)

- 조교님의 조언
  - 파인튜닝해놓은 모델 너무 믿지 말기 > 공격 들어올 수 있음
- 우리가 직접 모델 구축해서 파인튜닝할건지
- 수연, 민영 모델 중 어떤 모델 쓸건지
  - 수연이가 사용하고 있는거:
    - 감정 분석 : hun3359/klue-bert-base-sentiment

▼ 코드

```
from transformers import pipeline
import re
from collections import Counter
```

```
sentiment_pipeline = pipeline(
    "text-classification",
    model="hun3359/klue-bert-base-sentiment",
    tokenizer="hun3359/klue-bert-base-sentiment",
    return_all_scores=True
)
#긍정감정 하나 추가하면 좋을듯.
emotion_mapping = {
    '기쁨': ['기쁨','감사하는', '신뢰하는', '만족스러운', '흥분', '신이 난', '
    '당황': ['당황','고립된', '남의 시선을 의식하는', '열등감', '부끄러운', '
    '분노': ['분노','툴툴대는', '좌절한', '짜증내는', '방어적인', '악의적인',
    '불안': ['불안','두려운', '스트레스 받는', '취약한', '혼란스러운', '당혹스
    '상처': ['상처', '질투하는', '배신당한', '고립된', '충격 받은', '가난한 불
    '슬픔': ['슬픔', '실망한', '비통한', '후회되는', '우울한', '마비된', '염세
}

def find_main_emotion(detail_emotion):
    for main, details in emotion_mapping.items():
        if detail_emotion in details:
            return main
    return "기타"

def split_sentences(text):
    sentences = re.split(r'(?<=[.!?])+', text)
    return [s.strip() for s in sentences if s.strip()]

def sentiment(content, accuracy=0.1):
    sentences = split_sentences(content)

    main_emotions = []  #6개감정
    sub_emotions = []   #상세감정

    for sentence in sentences:
        result = sentiment_pipeline(sentence)
        print(result)
        #리스트 중에서 score가 가장 높은 항목
        best_result = max(result[0], key=lambda x: x['score'])
        #감정명
```

```
        best_label = best_result['label']
        #점수
        best_score = best_result['score']

        #6개 감정 선별
        main_emotion = find_main_emotion(best_label)
        if best_score >= accuracy:
            main_emotions.append(main_emotion)
        #상세감정
        sub_emotions.append(best_label)

    #최빈값 찾기
    result_main_emotions = [e for e in main_emotions if e != "기타

    if result_main_emotions:
        emotion_counter = Counter(result_main_emotions)
        most_common = emotion_counter.most_common()

        top_count = most_common[0][1]
        result_emotion = [emotion for emotion, count in most_com

        if len(result_emotion) == 1:
            most_common_emotion = result_emotion[0]
        else:
            print("Tie")
            best_score = -1
            best_emotion = None
            for sentence in sentences:
                result = sentiment_pipeline(sentence)
                best_result = max(result[0], key=lambda x: x['score'])
                best_label = best_result['label']
                best_score_candidate = best_result['score']
                main_emotion_candidate = find_main_emotion(best_la

                if main_emotion_candidate in result_emotion and best
                    best_score = best_score_candidate
                    best_emotion = main_emotion_candidate
```

```
            most_common_emotion = best_emotion
    else:
        return {
            'status': 'success',
            'main_emotion': '기타',
    }

    return {
        'status': 'success',
        'main_emotion': most_common_emotion,
    }
```

- 민영이가 사용하고 있는거:
  - 감정분석 : monologg/kobert
  - 원인분석 : paust/pko-t5-base

▼ 코드

```
import json
import re
from datasets import Dataset

# 1. 파일 불러오기
with open('/mnt/감성대화말뭉치(최종데이터)_Validation.json', 'r', e
    raw_data = json.load(f)

# 2. 데이터 추출
texts = []
labels = []

for item in raw_data:
    text = item['talk']['content'].get('HS01', None)
    label = item['profile']['emotion'].get('type', None)
    if text and label:
        texts.append(text)
        labels.append(label)

# 3. 감정 레이블 매핑 (7가지 기준)
```

```
emotion_map = {
    "E31": "우울함",
    "E25": "두려움",
    "E53": "슬픔",
    "E62": "행복",
    "E17": "화남",
    "E52": "불안",
    "E21": "놀람",
    "E01": "기타"
}


mapped_labels = [emotion_map.get(label, "기타") for label in lab

# 4. Hugging Face datasets로 변환
dataset = Dataset.from_dict({
    "text": texts,
    "label": mapped_labels
})

# 5. 감정 원인 추출 함수
def extract_cause(text):
    match = re.search(r"(.*?)(때문에|해서|탓에|으로 인해|바람에|덕분
    if match:
        return match.group(1).strip()

    sentences = re.split(r"[.!?]", text)
    sentences = [s.strip() for s in sentences if s.strip()]
    for sentence in sentences:
        if any(word in sentence for word in ["실수", "망쳤", "꼴찌", "
            return sentence
    return sentences[0] if sentences else text

# 6. 감정 예측 함수 (7가지 기준)
def predict_emotion(text):
    text = text.lower()
    # 평온 감정
    if any(word in text for word in ["고요", "차분", "편안", "평온", "느
```

```
        return "평온"
    # 행복
    elif any(word in text for word in ["좋아", "기쁘", "행복", "설렘", "
        return "행복"
    # 화남
    elif any(word in text for word in ["화나", "짜증", "열받", "분노"])
        return "화남"
    # 우울함
    elif any(word in text for word in ["우울", "상처", "속상", "좌절", "
        return "우울함"
    # 슬픔
    elif any(word in text for word in ["슬퍼", "눈물", "그리움", "외롭"
        return "슬픔"
    # 두려움
    elif any(word in text for word in ["무서워", "두려워", "겁나", "공포
        return "두려움"
    else:
        return "기타"

# 7. 감정 원인 추가
dataset = dataset.map(lambda example: {"cause": extract_caus

# 8. 사용자 입력 분석
def predict_emotion_and_cause(text):
    emotion = predict_emotion(text)
    cause = extract_cause(text)
    return emotion, cause

# ✏️ 사용 예시
input_text = input("\n✏️ 감정을 분석할 문장을 입력하세요: ")

predicted_emotion, predicted_cause = predict_emotion_and_ca

print(f"\n예측된 감정: {predicted_emotion}")
print(f"추출된 감정 원인: {predicted_cause}")
```

- 찬주가 사용하고 있는거:

- 감정 분석: https://huggingface.co/hun3359/klue-bert-base-sentiment (다른 사람이 파인튜닝 해놓은 모델에 프롬폼트 조정 정도)

▼ 코드

```
from transformers import pipeline
import json

# 상위 감정 매핑
fine2target = {
    "행복": ["기쁨", "감사하는", "신뢰하는", "편안한", "만족스러운", "흥
    "화남": ["분노", "툴툴대는", "좌절한", "짜증내는", "방어적인", "악의
    "우울함": ["슬픔", "실망한", "비통한", "후회되는", "우울한", "마비된
    "두려움": ["불안", "두려운", "스트레스 받는", "취약한", "조심스러운
    "놀람": ["혼란스러운", "당혹스러운", "충격 받은", "혼란스러운(당황
    "평온": ["편안한", "안도", "느긋"]
}

# 감정 매핑 함수
def get_coarse_label(fine_label, score):
    if score < 0.15:
        return "감정 측정 어려움"
    for coarse_label, fine_labels in fine2target.items():
        if fine_label in fine_labels:
            return coarse_label
    return "기타"

# 감정 분석기 불러오기
classifier = pipeline("sentiment-analysis", model="hun3359/klu

# 예시 문장 리스트
sentences = [
    "오늘 친구랑 약속이 취소돼서 너무 외로웠어.",
    "시험에서 100점을 받아서 기분이 좋았어.",
    "회의 시간에 발표를 망쳐서 정말 짜증났어.",
    "조용한 공원에서 산책하니까 마음이 편안했어.",
    "갑자기 상사가 소리를 질러서 너무 놀랐어."
]
```

```
# QA 데이터셋 생성
qa_dataset = []

for idx, text in enumerate(sentences):
    result = classifier(text)[0]
    fine_label = result["label"]
    score = result["score"]
    coarse_label = get_coarse_label(fine_label, score)

    if coarse_label == "감정 측정 어려움":
        continue

    question = f"{coarse_label}을 느끼게 만든 사건은 무엇인가요?"

    # SQuAD 스타일로 저장
    qa_dataset.append({
        "id": f"qa_{idx}",
        "context": text,
        "question": question,
        "answers": {
            "text": ["[정답 수동 기입 필요]"],
            "answer_start": [text.find("너무")]
        }
    })

# JSON 저장
with open("emotion_qa_dataset.json", "w", encoding="utf-8") a
    json.dump(qa_dataset, f, ensure_ascii=False, indent=2)

print("emotion_qa_dataset.json 생성 완료!")
```

⇒ 원인 분석 최대한 파인튜닝 해보기! 감정 분석은 세부 감정 추출 필요, 원인 분석은 여러개 문장 있어도 가능하도록!!

@2025년 5월 12일

- **민영, 찬주 → 감정 원인 분석 파인튜닝**
- **수연: 콘텐츠, 감정 분석 매핑**

- 구현 내용 이제 깃허브에 pr해야함!

- 어떤 알고리즘 썼고, 어떤 정확도 보였는지에 대한 내용도 중간발표에 포함되어야 함 !!
    - 콘텐츠와 감정 어떻게 매핑시켰는지
    - 감정 60가지 정도를 7가지로 어떻게 그룹화했는지