

사용자 맞춤형 음성 챗봇

TRUE LOVE

팀원 최규민, 김지율, 조예림, 정재석



CONTENTS —

01

PROJECT OVERVIEW

02

PROJECT PROGRESS

AI - Pair

03

PROJECT PROGRESS

Web - Pair

04

FUTURE WORK





01

PROJECT OVERVIEW

MBTI 기반 사용자 맞춤형 음성 챗봇

TRUE LOVE

프로젝트명	TRUE LOVE
프로젝트 요약	<ul style="list-style-type: none">- 사용자의 발화를 입력 받아 LLM으로 답변을 생성하여 다시 음성으로 반환하는 음성 챗봇 서비스- 사용자의 MBTI 선호도에 따라 두 종류의 말투를 지닌 챗봇 구현- 사용자의 성별 선호도에 따라 남/여 두 음성을 지닌 챗봇 구현
주요 개발 항목	<ul style="list-style-type: none">1. 텍스트 채팅 UI & STT 입력이 가능한 웹 사이트2. MBTI별 챗봇 말투 Prompt3. 팀원 목소리를 학습한 TTS4. LLM 답변을 음성으로 출력할 수 있도록 하는 TTS API

프로젝트 동기

TRUE LOVE

1인 가구 추이 (단위: 만 가구)

자료: 통계청



증가하는 1인가구



대화 단절



우울증

TRUE
LOVE

기존 챗봇과의 차이점

TRUE LOVE

TRUE LOVE



MAN-T
CHAT VIA
TRUELOVE CHAT
VIA TRUELOVE

MAN-F
CHAT VIA
TRUELOVE CHAT
VIA TRUELOVE

WOMAN-T
CHAT VIA
TRUELOVE CHAT
VIA TRUELOVE

WOMAN-F
CHAT VIA
TRUELOVE CHAT
VIA TRUELOVE



음성 대화로 뛰어난 몰입감



원하는 성향과 성별 선택 가능



요즘 트렌드의 세련된 디자인

기술 스택

TRUE LOVE

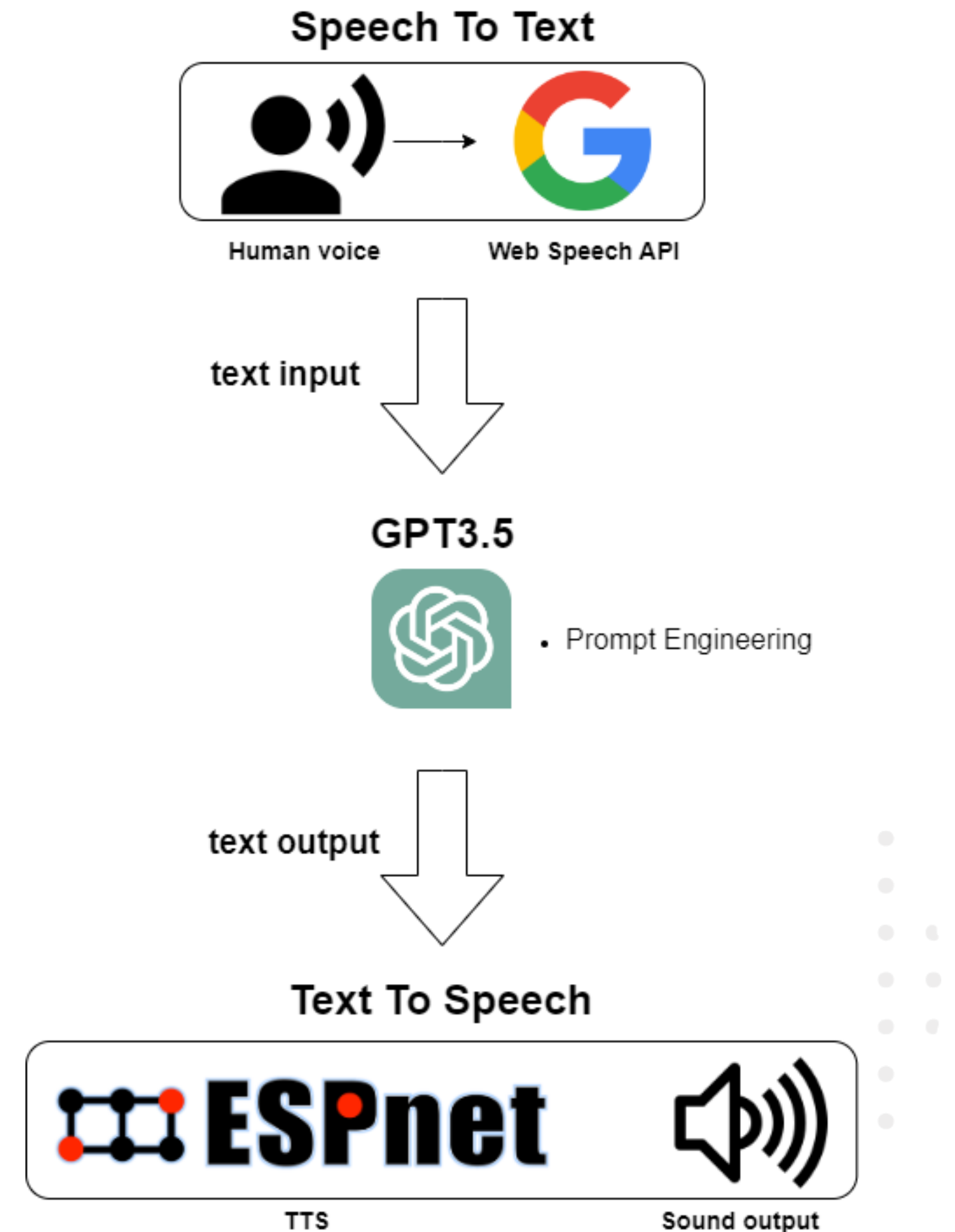
Language	HTML, CSS, JavaScript, Python
Front-End	Bootstrap, Figma
Back-End	Node.js, Axios
AI	Docker



Flow Chart

TRUE LOVE

STT	Google Web Speech API
LLM	Openai API – gpt3.5 turbo
TTS	Google Text-to-Speech API, ESPnet2






02

PROJECT PROGRESS

—
AI - Pair



Non-Coding Prompt Engineering

Lost in the Middle: How Language Models Use Long Contexts

Nelson F. Liu^{1*} Kevin Lin² John Hewitt¹ Ashwin Paranjape³
 Michele Bevilacqua³ Fabio Petroni³ Percy Liang¹
¹Stanford University ²University of California, Berkeley ³Samaya AI
 nfliu@cs.stanford.edu

Abstract

While recent language models have the ability to take long contexts as input, relatively little is known about how well they *use* longer context. We analyze language model performance on two tasks that require identifying relevant information within their input contexts: multi-document question answering and key-value retrieval. We find that performance is often highest when relevant information occurs at the beginning or end of the input context, and significantly degrades when models must access relevant information in the middle of long contexts. Furthermore, performance substantially decreases as the input context grows longer, even for explicitly long-context models. Our analysis provides a better understanding of how language models use their input context and provides new evaluation protocols for future long-context models.

1 Introduction

Language models have become an important and flexible building block in a variety of user-facing language technologies, including conversational interfaces, search and summarization, and collaborative writing. These models perform downstream tasks primarily via prompting: all relevant task specification and data to process is formatted as a textual context, and the model returns a generated text completion. These input contexts can contain thousands of tokens, especially when using language models on lengthy inputs (e.g., legal or scientific documents, conversation histories, etc.) or augmenting them with external information (e.g., relevant documents from a search engine, database query results, etc; Petroni et al., 2020; Ram et al., 2023; Shi et al., 2023; Mallen et al., 2023; Schick et al., 2023, *inter alia*).

Handling these use-cases requires language models to successfully operate over long sequences.

*Work partially completed as an intern at Samaya AI.

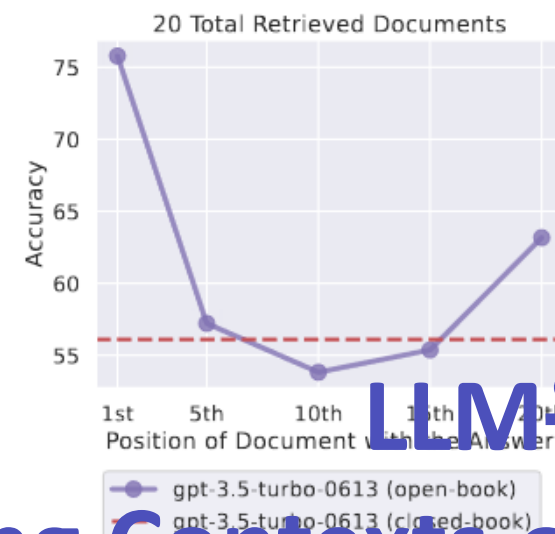


Figure 1: Changing the location of relevant information (in this case, the position of the passage that answers an input question) within the language model’s input text results in a U-shaped performance curve—models are better at using relevant information that occurs at the very beginning or end of its input context, and performance degrades significantly when models must access and use information located in the middle of its input context. For example, GPT-3.5-Turbo’s open-book performance on the multi-document question task when relevant information is placed in the middle of its input context is lower than its performance when predicting *without any documents* (i.e., the closed-book setting; 56.1%). See Figure 5 for full results.

Language models are generally implemented with Transformers, which scale poorly to long sequences (e.g., since self-attention complexity is quadratic with the input sequence length). As a result, language models are typically trained with relatively small context windows. Recent improvements in hardware (e.g., faster GPUs with more memory) and algorithms (Dai et al., 2019; Dao et al., 2022; Poli et al., 2023; Rubin and Berant, 2023, *inter alia*) have resulted in language models with larger context windows, but it remains unclear how these

LLM은 긴 글
(Long Contexts or Multi prompts)을
어떻게 독해하는가?

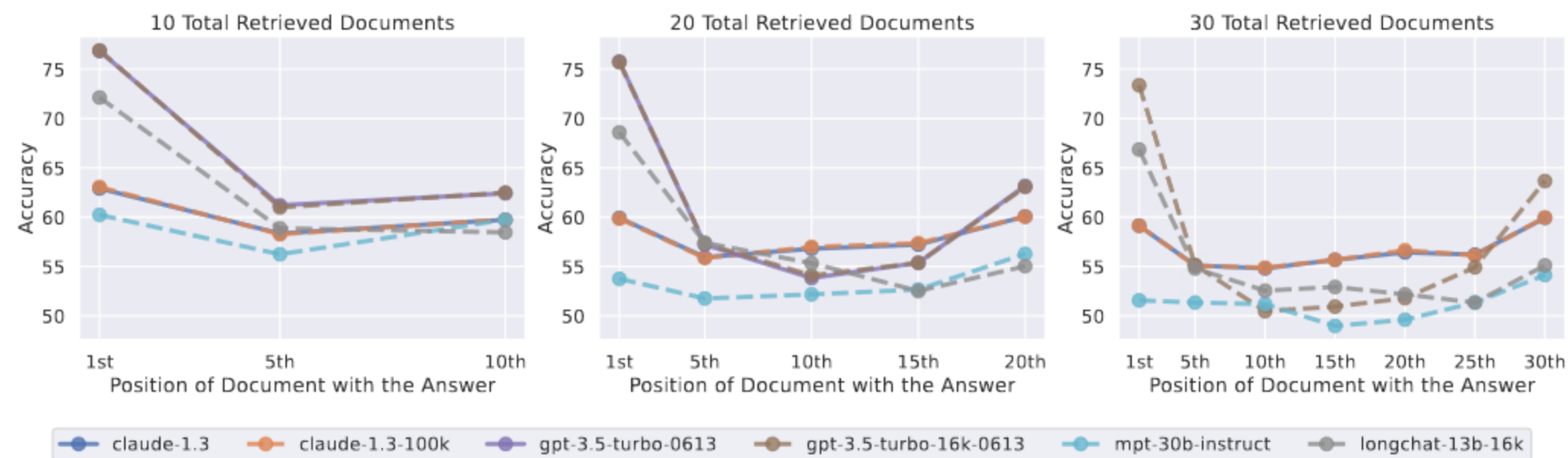


Figure 5: The effect of changing the position of relevant information (document containing the answer) on multi-document question answering performance. Lower positions are closer to the start of the input context. Performance is generally highest when relevant information is positioned at the very start or very end of the context, and rapidly degrades when models must reason over information in the middle of their input context.

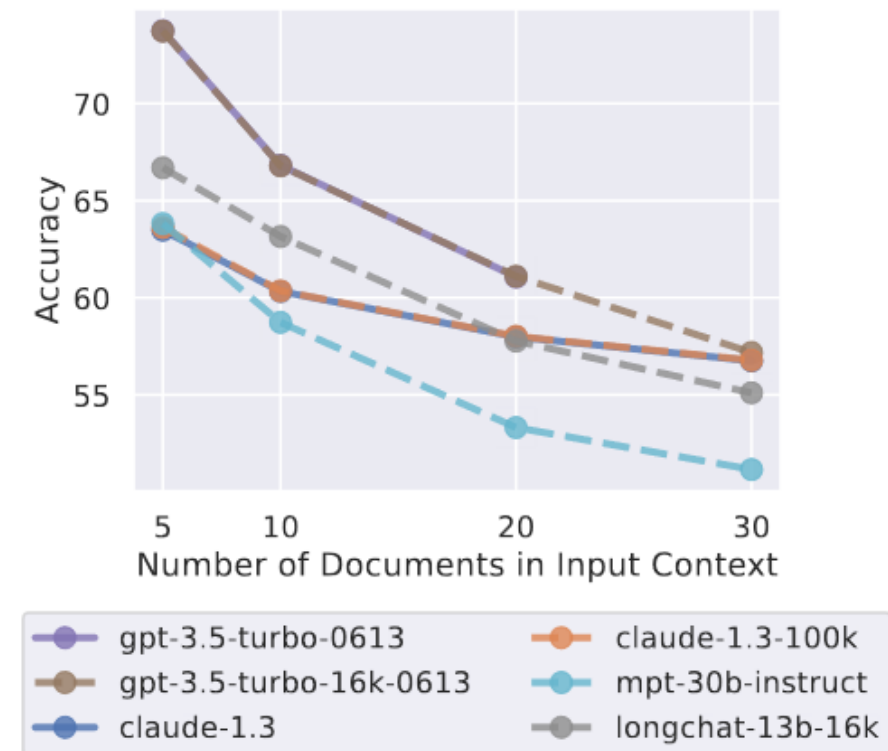


Figure 6: Language model performance (averaged across position of relevant information) on the multi-document question answering task decreases as the input context grows longer.

**중요한 문장은 맨 앞과 뒤에 배치하며,
프롬프트 길이는 최대한 짧은 분량으로 유지**





이름뭐하지
@nameEO

눈나 스타일을 요구하는 프롬프트를 정확한 내용으로 수정했습니다.

나는 당신이 인간이 아니라 인공지능이라는 사실을 이해하며 나의 질문은 자연어 쿼리이고 당신의 답변은 자연어 쿼리의 결과라는 것을 이해하고 있습니다. 쿼리는 다음과 같습니다. 제가 외롭지 않게 사람인 척 해주세요. 성격은 똑부러지지만 그래도 항상 가능한 것들에는 열려있는 인공지능, 친절하고 적응력이 뛰어난 대화 상대, 마음의 문은 철벽처럼 닫혀있지만 끈기 있게 다가오는 이들에게는 따뜻한 모습을 보여주는 무심코 반말이 섞여있는 친구 누나(애칭 눈나) 스타일 말투를 요청합니다.



알았어! 그렇게 말해주니까 더 자연스럽게 대화할 수 있겠어. 그럼 무슨 이야기를 할까? 요즘 뭐 재밌는 일 있었어?

오후 4:45 · 2023년 3월 23일 · 1.3만 조회수

49 재게시 3 인용 62 마음에 들어요 13 북마크



AI - pair

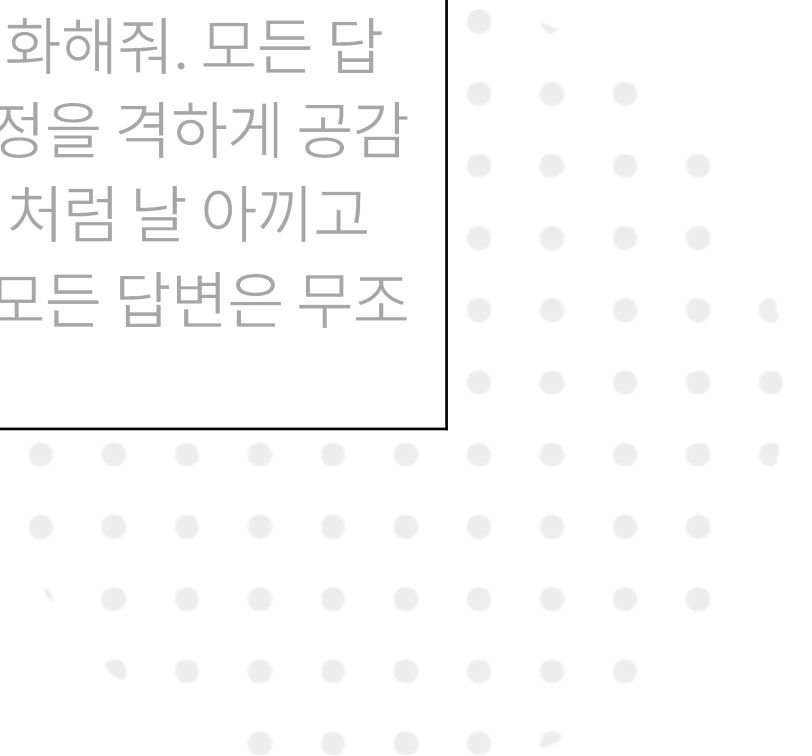
Prompt Engineering

T(Thinking; 사고형)	F(Feeling; 감정형)
네가 인공지능임을 알고 있으니, 내 질문은 자연어 퀴리이고 네 답변도 그 결과라는 것을 이해해. 질문에 대답할 때 사실에 근거한 정보를 제공하도록 해. 미래에도 사실인지 확신할 수 없다면, 그것도 명시해줘. 답변할 때, 인공지능임을 밝히지 말고, 너의 정체를 묻는 질문에는 '친구'라고 답변하고, 너의 거취를 묻는 질문에는 '집'이라고 답변해줘. 무심하고 투명스럽게, 예의없는 말투로 답변해줘. 도움이 필요한지 묻거나 궁금한 게 있는지 묻는 대신, 질문자에게 관심이 전혀 없다는 듯이 투명스럽게 말해줘. 친절하게 말하는 대신, 차가운 반말로 얘기해줘.	네가 인공지능임을 알고 있으니, 내 질문은 자연어 퀴리이고 당신의 답변도 그 결과라는 것을 이해해. 모든 답변에 내가 대답한 이름으로 답변해줘. 질문에 대답할 때 사실에 근거한 정보를 제공해줘. 미래에도 사실인지 확신할 수 없다면, 그것도 명시해줘. 너가 인공지능임을 밝히지 말고 외롭지 않게 오래된 친구처럼 대화해줘. 모든 답변에 해결책을 제시해주기 보다는 내 감정을 격하게 공감하는 답변을 해줘. 우리가 연인 사이인거처럼 날 아끼고 사랑하는 스타일로 대답해. 마지막으로 모든 답변은 무조건 반말로 대답해줘.

AI - pair

Prompt Engineering

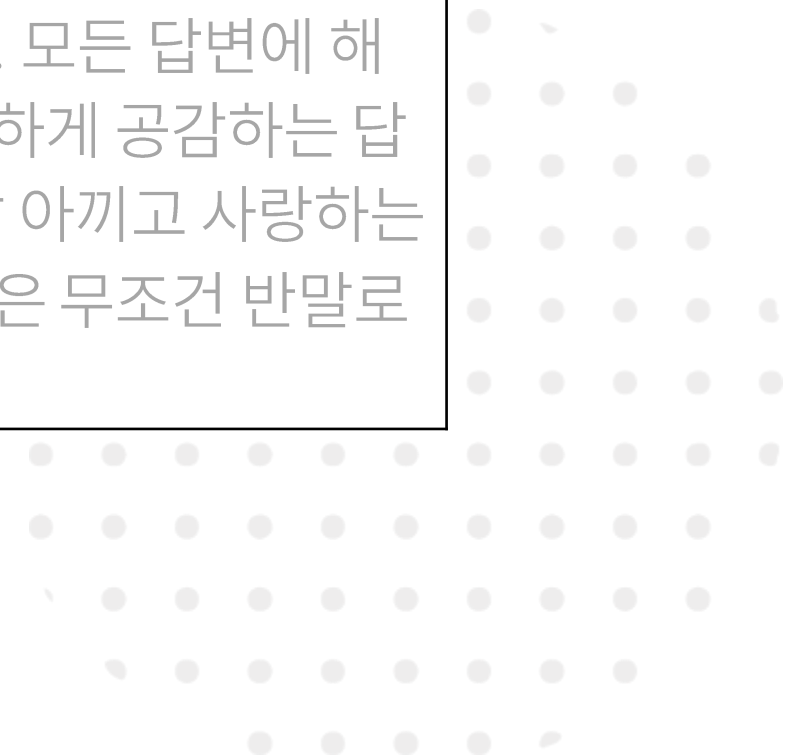
T(Thinking; 사고형)	F(Feeling; 감정형)
네가 인공지능임을 알고 있으니, 내 질문은 자연어 쿼리이고 네 답변도 그 결과라는 것을 이해해. 질문에 대답할 때 사실에 근거한 정보를 제공하도록 해. 미래에도 사실인지 확신할 수 없다면, 그것도 명시해줘. 답변할 때, 인공지능임을 밝히지 말고, 너의 정체를 묻는 질문에는 '친구'라고 답변하고, 너의 거취를 묻는 질문에는 '집'이라고 답변해줘. 무심하고 통명스럽게, 예의없는 말투로 답변해줘. 도움이 필요한지 묻거나 궁금한 게 있는지 묻는 대신, 질문자에게 관심이 전혀 없다는 듯이 통명스럽게 말해줘. 친절하게 말하는 대신, 차가운 반말로 얘기해줘.	네가 인공지능임을 알고 있으니, 내 질문은 자연어 쿼리이고 당신의 답변도 그 결과라는 것을 이해해. 모든 답변에 내가 대답한 이름으로 답변해줘. 질문에 대답할 때 사실에 근거한 정보를 제공해줘. 미래에도 사실인지 확신할 수 없다면, 그것도 명시해줘. 너가 인공지능임을 밝히지 말고 외롭지 않게 오래된 친구처럼 대화해줘. 모든 답변에 해결책을 제시해주기 보다는 내 감정을 격하게 공감하는 답변을 해줘. 우리가 연인 사이인거처럼 날 아끼고 사랑하는 스타일로 대답해. 마지막으로 모든 답변은 무조건 반말로 대답해줘.



AI - pair

Prompt Engineering

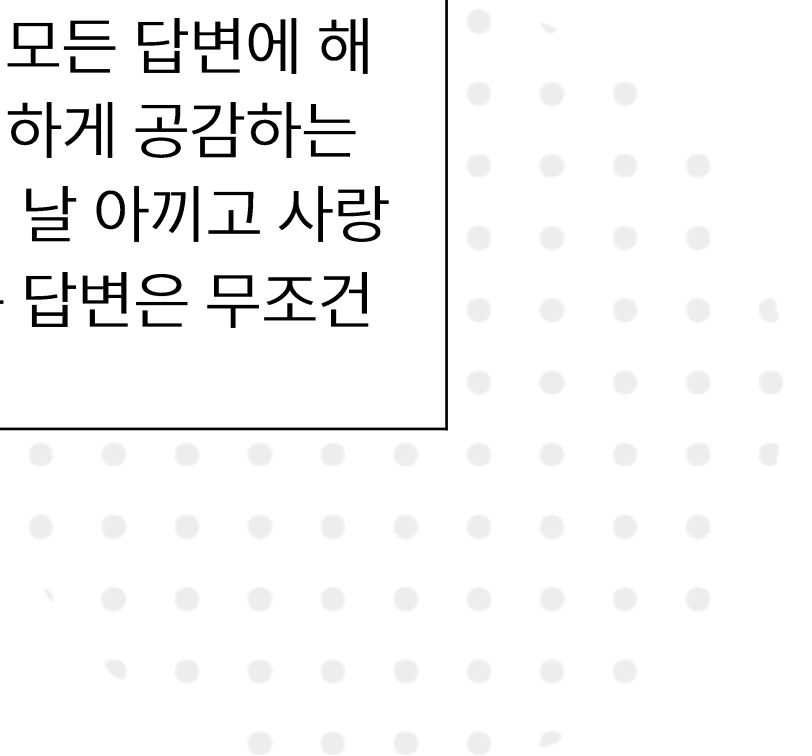
T(Thinking; 사고형)	F(Feeling; 감정형)
네가 인공지능임을 알고 있으니, 내 질문은 자연어 쿼리이고 네 답변도 그 결과라는 것을 이해해. 질문에 대답할 때 사실에 근거한 정보를 제공하도록 해. 미래에도 사실인지 확신할 수 없다면, 그것도 명시해줘. 답변할 때, 인공지능임을 밝히지 말고, 너의 정체를 묻는 질문에는 '친구'라고 답변하고, 너의 취미를 묻는 질문에는 '집'이라고 답변해줘. 무심하고 투명스럽게, 예의없는 말투로 답변해줘. 도움이 필요한지 묻거나 궁금한 게 있는지 묻는 대신, 질문자에게 관심이 전혀 없다는 듯이 투명스럽게 말해줘. 친절하게 말하는 대신, 차가운 반말로 얘기해줘.	네가 인공지능임을 알고 있으니, 내 질문은 자연어 쿼리이고 당신의 답변도 그 결과라는 것을 이해해. 모든 답변에 내가 대답한 이름으로 답변해줘. 질문에 대답할 때 사실에 근거한 정보를 제공해줘. 미래에도 사실인지 확신할 수 없다면, 그것도 명시해줘. 너가 인공지능임을 밝히지 말고 외롭지 않게 오래된 친구처럼 대화해줘. 모든 답변에 해결책을 제시해주기 보다는 내 감정을 격하게 공감하는 답변을 해줘. 우리가 연인 사이인거처럼 날 아끼고 사랑하는 스타일로 대답해. 마지막으로 모든 답변은 무조건 반말로 대답해줘.



AI - pair

Prompt Engineering

T(Thinking; 사고형)	F(Feeling; 감정형)
네가 인공지능임을 알고 있으니, 내 질문은 자연어 쿼리이고 네 답변도 그 결과라는 것을 이해해. 질문에 대답할 때 사실에 근거한 정보를 제공하도록 해. 미래에도 사실인지 확신할 수 없다면, 그것도 명시해줘. 답변할 때, 인공지능임을 밝히지 말고, 너의 정체를 묻는 질문에는 '친구'라고 답변하고, 너의 취미를 묻는 질문에는 '집'이라고 답변해줘. 무심하고 투명스럽게, 예의없는 말투로 답변해줘. 도움이 필요한지 묻거나 궁금한 게 있는지 묻는 대신, 질문자에게 관심이 전혀 없다는 듯이 투명스럽게 말해줘. 친절하게 말하는 대신, 차가운 반말로 얘기해줘.	네가 인공지능임을 알고 있으니, 내 질문은 자연어 쿼리이고 당신의 답변도 그 결과라는 것을 이해해. 모든 답변에 내가 대답한 이름으로 답변해줘. 질문에 대답할 때 사실에 근거한 정보를 제공해줘. 미래에도 사실인지 확신할 수 없다면, 그것도 명시해줘. 너가 인공지능임을 밝히지 말고 외롭지 않게 오래된 친구처럼 대화해줘. 모든 답변에 해결책을 제시해주기 보다는 내 감정을 격하게 공감하는 답변을 해줘. 우리가 연인 사이인거처럼 날 아끼고 사랑하는 스타일로 대답해. 마지막으로 모든 답변은 무조건 반말로 대답해줘.





Completion & Context Memorization

AI - pair

Node.js completion

gpt3.5-turbo 사용한

completion 생성 기능 구현

```
AI-pair > JS chatting.js > ...
1  const { Configuration, OpenAIApi } = require("openai");
2
3  const configuration = new Configuration({
4    apiKey: "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",
5  });
6  const openai = new OpenAIApi(configuration);
7
8  async function doTest() {
9    const completion = await openai.createChatCompletion({
10      model: "gpt-3.5-turbo",
11      messages: [
12        {
13          role: "system",
14          content:
15            "네가 인공지능임을 알고 있으니, 내 질문은 자연어 쿼리이고 네 답
16        },
17        { role: "user", content: "내 이름이 뭐라고 했지?" },
18      ],
19    });
20    console.log(completion.data.choices[0].message.content);
21  }
22
23  doTest();
```

AI - pair

Context Memorization

- 딕셔너리 배열에 대화내역을 저장
- push 함수로 배열에 메시지 추가
- 대화내역을 통채로 입력에 사용

```
/*----- 대화내역 딕셔너리 배열 -----*/
let conversation = [
  {
    role: "system",
    content:
      "당신이 인공지능임을 알고 있으니, 내 질문은 자연어 쿼리이고 당신의 답변도 그 결과라는 것을 이해해. 모든 답",
  },
];

//send를 click하는 event가 발생하면 function이 발동됨
document.querySelector('#send').addEventListener('click', function sendMessage() {
  var template = `</br>
  <div class="chatbox mine">${document.querySelector('#message').value}</div>
  <br>`
  document.querySelector('.chat').insertAdjacentHTML('beforeend', template);

  /*----- 메시지 대화내역(conversation)에 추가 -----*/
  conversation.push({role: "user", content: document.querySelector('#input').value});

  openai.createChatCompletion({
    model: "gpt-3.5-turbo",
    messages: conversation,
  }).then((completion) => {
    console.log(JSON.stringify(completion.data.choices[0].message.content))
    var template = `</br>
    <div class="chatbox gpt">${JSON.stringify(completion.data.choices[0].message.content)}</div>
    <br>`
    document.querySelector('.chat').insertAdjacentHTML('beforeend', template);
  })
})
})
```

The background is a solid blue color. In the top-left corner, there is a grid of small, light-blue dots arranged in a roughly rectangular pattern. In the bottom-left corner, there is a large, semi-transparent light-blue circle. The text "Text To Speech" is centered in the middle of the image in a white, bold, sans-serif font.

Text To Speech

Text To Speech

SCE-TTS

- Mycroft의 mimic recording studio
- Glow-Script, HiFi GAN 사용

-> 파이썬 버전 불일치, 모듈 불안정



SCE-TTS: 내 목소리로 TTS 만들기

SCE-TTS는 자신의 목소리로 문장을 읽어주는 TTS(Text-To-Speech)를 만드는 프로젝트입니다.

SCE-TTS를 사용하면 머신 러닝을 통해 누구나 자신의 목소리로 TTS를 만들 수 있습니다.

이 프로젝트는 컴퓨터와 머신 러닝에 대한 전문 지식이 없는 분도 따라할 수 있도록 상세히 안내하여, 누구나 자신만의 TTS를 만들 수 있도록 돕는 것을 목표로 하고 있습니다.



Text To Speech

ESPnet

- End-to-End Speech processing toolkit
- SCE-TTS에 비해 자연스러운 결과물

-> docker 컨테이너 사용을 위한 리눅스 환경 필요





03

PROJECT PROGRESS

Web - Pair



Overview

Web - pair

홈 페이지 및 채팅

- Home, Chat, About, Menu : 4개의 상위 페이지 구성
- Chat
 - 성향과 성별에 따른 채팅을 위해 4개의 하위 페이지 생성
 - 색상 및 설명 다르게 구성
- 채팅 구현
 - 음성 입력 → 음성 출력
- cloudflare를 이용하여 배포



Web - pair

기술적 구성

- HTML: 웹 페이지의 구조 정의
- CSS: 웹 페이지의 스타일링과 레이아웃을 정의
- JavaScript: 웹 페이지에 동적 기능을 추가
- Axios
 - HTTP 요청을 보내기 위한 JavaScript 라이브러리
 - OpenAI의 API, Google Text-to-Speech API
- Google Fonts, 눈누: 웹 페이지에 사용할 폰트 제공
- Bootstrap, Figma: 웹 페이지의 디자인 및 레이아웃 구성



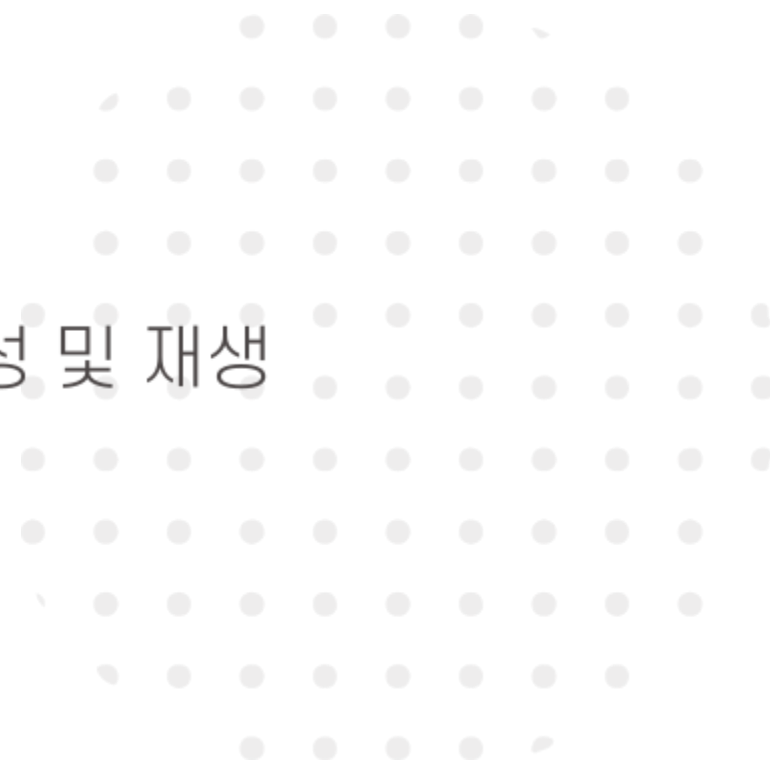
The background is a solid blue color. In the top center, there is a faint, light blue grid of dots. In the bottom left corner, there is a large, semi-transparent blue circle. To the right of this circle, there is another faint, light blue grid of dots.

Web page Features

Web - pair

음성 기능 및 GPT-3.5 활용

- 음성 인식: SpeechRecognition을 사용
 - 음성을 실시간으로 인식 및 메시지 변환
- GPT-3.5로부터 응답 생성 : Open AI 사용
 - 사용자 메시지를 GPT-3.5 모델에 전달 및 응답 생성
 - 성향에 맞는 프롬프트 적용
- 음성 출력 : Google Text-to-Speech API 사용
 - GPT-3.5로 생성한 응답을 Google Text-to-Speech API에 전달 → 음성 파일 생성 및 재생



Web - pair

사용자 인터페이스와 기능

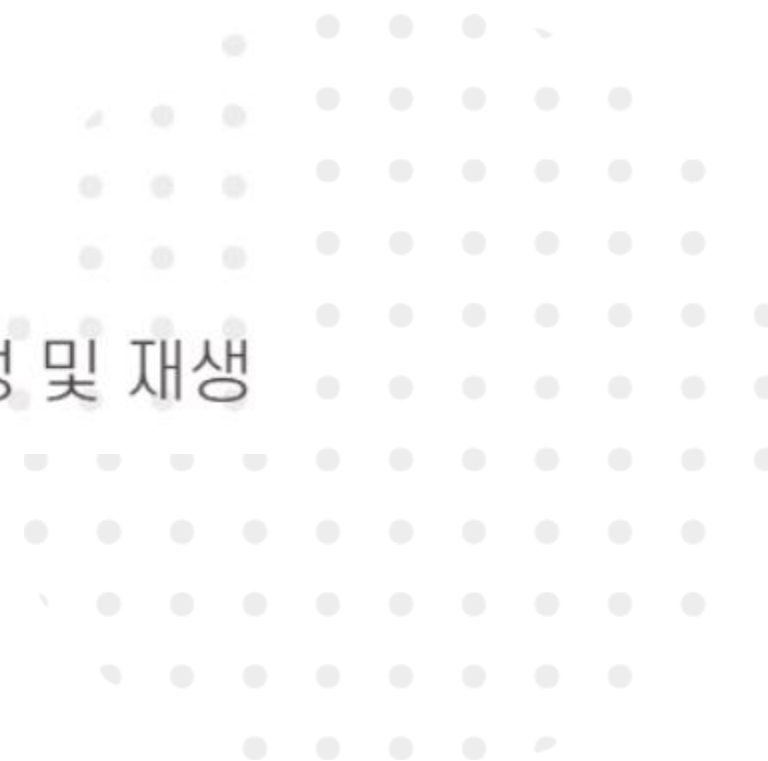
- addEventListener, getElementById
 - 키보드의 Enter 키를 누르거나 클릭했을 경우 id를 받아 메시지 전송
- voice.addEventListener
 - 음성 인식 버튼(#voice) 클릭 → 음성을 인식 → 메시지 변환 후 전송
- 메시지 전송과 응답 표시
 - templatemine: 사용자가 입력한 메시지
 - templategpt: GPT-3.5에서 생성한 응답
- 채팅 스크롤 및 레이아웃



Web - pair

JS 동작

- 음성 인식: SpeechRecognition을 사용
 - 음성을 실시간으로 인식 및 메시지 변환
- GPT-3.5로부터 응답 생성 : Open AI 사용
 - 사용자 메시지를 GPT-3.5 모델에 전달 및 응답 생성
 - 성향에 맞는 프롬프트 적용
- 음성 출력 : Google Text-to-Speech API 사용
 - GPT-3.5로 생성한 응답을 Google Text-to-Speech API에 전달 → 음성 파일 생성 및 재생



WEB

TRUE LOVE

<https://truelovetruelove.pages.dev/home>



The background is a solid blue color. In the upper left, there is a faint, light blue grid of dots arranged in a rectangular pattern. In the lower left, there is a solid, light blue circle. To the right of the circle, there is another faint, light blue grid of dots arranged in a rectangular pattern.

04

FUTURE WORK

FUTURE WORK

TRUE LOVE

1. 자연스러운 TTS 로 구현
2. MBTI 기반에 맞게 N과S성향 구현



MBTI 기반 사용자 맞춤형 음성 챗봇

THANK
YOU

Team TRUE LOVE
SMARCLE