

UNIVERSIDAD SANTO TOMÁS PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA FACULTAD DE INGENIERÍA ELECTRÓNICA



Projecto Final 3

Analisis de emociones

Presentado a: Ing. Diego Alejandro Barragán Vargas Julian Sebastian Alvarado Monroy, Cód. 2308042

Resumen— Este informe presenta el desarrollo e integración de un sistema de chatbot basado en DeepSeek AI en el robot social Pepper. El objetivo del proyecto fue habilitar a Pepper para mantener una conversación fluida con humanos mediante el uso de un modelo de lenguaje avanzado (DeepSeek) alojado en un servidor Flask, comunicándose a través de una red local. Se utilizó Docker para contenerizar el servidor de DeepSeek, y el cliente del robot se programó en Python 2.7 con la plataforma NAOqi de SoftBank. Como resultado, Pepper pudo escuchar al usuario, convertir el habla a texto, enviar la consulta al servidor DeepSeek y pronunciar la respuesta generada, mostrando incluso un estilo de comunicación personalizado (por ejemplo, con un tono humorístico al estilo Bart Simpson). Las pruebas realizadas confirmaron la correcta interacción, reproduciendo respuestas coherentes y apropiadas al contexto conversacional.

I. Introducción

Los robots sociales están diseñados para interactuar con las personas de forma natural y emocional. Pepper, un robot humanoide de Aldebaran Robotics (SoftBank), es el primer robot social que puede reconocer rostros y emociones humanas. Pepper está optimizado para la interacción a través de conversación y cuenta con sensores y módulos de reconocimiento de voz integrados. Debido a esto, Pepper se ha empleado en entornos comerciales y educativos para recibir, informar y guiar a usuarios, habiendo más de 17,000 unidades desplegadas hasta 2021.

Por otro lado, los modelos conversacionales avanzados basados en IA han experimentado un rápido avance. Los modelos de lenguaje grande (LLMs) como GPT-4 han demostrado gran habilidad para generar texto coherente y mantener diálogos naturales. En particular, DeepSeek AI es un nuevo LLM desarrollado en China como alternativa a GPT-4 o Gemini, que ofrece modelos de código abierto (como DeepSeek-R1) y acceso por API. Este modelo está optimizado para eficiencia y coste, y su API es compatible con el formato de OpenAI, permitiendo reutilizar SDKs existentes.

En este contexto, el proyecto buscó integrar Pepper con el modelo DeepSeek. Inspirados en proyectos previos que conectan Pepper con chatbots basados en IA, se planteó una arquitectura cliente-servidor donde Pepper funciona como cliente local que envía preguntas a un servidor remoto. Dicho servidor ejecuta una aplicación Flask que maneja las peticiones de chat y devuelve las respuestas de DeepSeek. A

continuación se desarrolla la teoría necesaria para comprender cada componente de este sistema.

El informe se organiza en cinco etapas: construcción del dataset, ejecución de código de captura, implementación del análisis de voz, revisión tecnológica del estado del arte, y publicación técnica en repositorios colaborativos.

II. Marco Teórico

A. Robótica social (Pepper)

Los robots sociales constituyen una subclase de robots programables diseñados para interactuar socialmente con los humanos. Estos robots realizan tareas dentro de contextos sociales, comunicándose o provocando respuestas sociales en las personas. Pepper es un ejemplo destacado: es un robot humanoide social capaz de percibir emociones humanas y optimizado para el diálogo con usuarios. Pepper incluye micrófonos y un procesador de lenguaje para reconocer palabras, así como un sintetizador de voz para conversar con fluidez. Además, Pepper dispone de módulos de percepción (cámaras 2D/3D, sensores táctiles) y reconocimiento de voz en 15 idiomas distintos, lo que le permite comprender comandos verbales básicos y responder de forma multilingüe.

B. Reconocimiento de voz (ASR)

El reconocimiento automático del habla (ASR) es la tecnología que convierte señales de audio en texto legible por la computadora. Es un campo interdisciplinario que involucra aprendizaje automático y lingüística, y es fundamental para los sistemas de interacción oral con máquinas. En nuestro sistema, Pepper utiliza su motor ASR interno (ALSpeechRecognition de NAOqi) para transcribir la voz del usuario. Una vez transformado en texto, ese mensaje puede procesarse con un chatbot. El proceso inverso (texto a voz) se realiza mediante síntesis de voz (text-to-speech), que en Pepper está a cargo del módulo ALTextToSpeech. El uso combinado de ASR y TTS permite una conversación bidireccional hablada.



UNIVERSIDAD SANTO TOMÁS PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA FACULTAD DE INGENIERÍA ELECTRÓNICA



C. Plataforma Roboflow

Roboflow es una plataforma en línea que permite crear, gestionar y exportar datasets para modelos de visión artificial. Facilita la creación de modelos personalizados mediante etiquetado gráfico, procesamiento automático y exportación en múltiples formatos (YOLO, TensorFlow, etc.). En este proyecto, se utilizó Roboflow para almacenar y clasificar imágenes faciales correspondientes a cuatro emociones.

D. Modelos de lenguaje grandes (LLMs)

Un Modelo de Lenguaje Grande es un modelo de aprendizaje automático diseñado para tareas de procesamiento de lenguaje natural, especialmente generación de texto. Estos modelos, entrenados con enormes volúmenes de texto, pueden generar respuestas coherentes a preguntas abiertas. En este proyecto se usa DeepSeek-R1, un LLM que promete capacidades comparables a GPT-4. Los LLMs funcionan a través de modelos de completado de texto: reciben un prompt (solicitud) y generan un texto como respuesta, considerando el contexto provisto.

E. Arquitectura cliente-servidor

En este proyecto se implementa un esquema cliente-servidor para separar responsabilidades. La arquitectura cliente-servidor es un modelo de diseño de software en el que un programa cliente solicita servicios o recursos a un servidor central, que procesa las peticiones y devuelve las respuestas. En nuestro caso, Pepper actúa como cliente: recibe la entrada de voz del usuario y realiza solicitudes HTTP al servidor. El servidor es una aplicación Flask que expone un endpoint /chat. Al recibir la petición del cliente, el servidor consulta a DeepSeek y regresa la respuesta al cliente. Este modelo permite centralizar el procesamiento del lenguaje en un servidor potente, mientras que el robot solo maneja la interacción física con el usuario. Además, facilita la escalabilidad y actualización del servicio de IA sin modificar el robot.

III. Procedimiento y Resultados

A. Construcción del servidor Flask con DeepSeek (incluido Docker)

Se creó una aplicación en Flask que actúa como backend conversacional. Se instaló Python 3.8+ y el paquete deepseeksdk. En el archivo app.py se inició la aplicación Flask, se configuró el cliente de DeepSeek con la clave de API y se definió la ruta /chat que recibe el mensaje del usuario en JSON. Luego, se contenedorizó la aplicación con Docker. Se creó un Dockerfile basado en python:3.8-slim, se copió el proyecto,

se instalaron dependencias y se expuso el puerto 5000. Se construyó la imagen (docker build -t deepseek-chatbot .) y se ejecutó el contenedor (docker run -p 5000:5000 deepseek-chatbot).

```
Centato any 3 statement of the Company of the Compa
```

Figura 1: Creación Dataset. Fuente: Creación propia

B. Desarrollo del cliente en Pepper

En el robot Pepper, se programó un cliente en Python 2.7 usando NAOqi. Debido a las limitaciones de instalación en el entorno de Pepper, se incluyeron bibliotecas como requests en el paquete de la aplicación. El cliente captura la voz del usuario mediante ALSpeechRecognition, convierte la voz en texto y luego envía ese texto al servidor usando HTTP POST. La respuesta obtenida del servidor es leída en voz alta por Pepper usando ALTextToSpeech.

```
| According to the control of the co
```

Figura 2: Visualizacion Dataset. Fuente: Creación propia

C. Comunicación por red local

Se conectó Pepper y el servidor Flask a la misma red local. Se identificó la dirección IP del host y se configuró en el cliente de Pepper. Las solicitudes se realizaron vía REST sobre HTTP, garantizando baja latencia. Fue importante que el puerto 5000 estuviera accesible para que Pepper pudiera conectarse correctamente al servidor.

D. Flujo de conversación

El ciclo completo fue: el usuario habla, Pepper reconoce el comando, envía el texto al servidor, DeepSeek procesa y responde, y Pepper pronuncia la respuesta. Se personalizó la personalidad del bot usando un prompt con estilo "Bart



UNIVERSIDAD SANTO TOMÁS PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA FACULTAD DE INGENIERÍA ELECTRÓNICA



Simpson". Esto dio como resultado una conversación más divertida y adaptada.

```
(D) expendence part core et catorest va te vast here, pera antes escucha esta centrá de person herejaurio. Sijá mesta mano, aburráni (Biol Contrate servicine es 10,186.0.11859)
(Biol Contrate servicine (Contrate) "servicine es 10,186.0.11859)
(Biol Contrate) (Biol Contrate)
```

Figura 3: Creación Dataset. Fuente: Creación propia

IV. Conclusiones

La integración de un chatbot DeepSeek en el robot Pepper fue exitosa. Se comprobó que el uso de una arquitectura cliente-servidor permite dotar a Pepper de capacidades conversacionales avanzadas sin sobrecargar su sistema. El uso de DeepSeek proporcionó un backend flexible y poderoso, permitiendo también la personalización de tono y estilo. Las principales dificultades se centraron en la configuración de red, el entorno limitado de Pepper y el manejo de ASR. Se demostró que es posible extender las funciones sociales de un robot humanoide mediante modelos de lenguaje modernos y APIs conversacionales.

V. Referencias

- Aldebaran Robotics. Pepper. SoftBank Robotics. https://aldebaran.com/en/pepper/
- ilabsweden. PepperChat. GitHub. https://github.com/ilabsweden/pepperchat
- Wikipedia. Speech recognition. https://en.wikipedia.org/ wiki/Speech_recognition
- Wikipedia. Large language model. https://en.wikipedia. org/wiki/Large_language_model
- Mistelbacher, R. DeepSeek AI vs ChatGPT. Fresh van Root, 2025.
- Ranjan, R. Building a Chatbot with DeepSeek AI on Docker. Medium, 2025.
- Nerus, O. *Pepper robot: upload python modules*. StackOverflow. https://stackoverflow.com/q/45799150
- Wikipedia. Cliente-servidor. https://es.wikipedia.org/ wiki/Cliente-servidor
- Wikipedia. Robot social. https://es.wikipedia.org/wiki/ Robot social

https://github.com/JulianAlva24

https://www.overleaf.com/project/68465a8445555eea03a75ab8