1. Problema

   Hemos orientado el proyecto en el problema de detección de emociones a través de una simple frase, es decir, tras un csv con 30 ejemplos de frases y su correspondiente sentimiento (positivo, neutral o negativo), le pasamos cada una de las frases en los diferentes prompts elegidos (base, zero shot, few shot y chain of thougth), y en función de lo que nos devuelve el modelo, identificamos si la salida contiene uno de los diferentes sentimientos, para poder clasificarlo nosotros mismos; y en caso de no poder identificar dicho sentimiento, se calificaría como desconocido.

2. Modelos
   - Google flan t5 base → 248M
   - Chat gpt 2 medium → 368M
   - Google flan t5 small → 77M
   - Google gemma-3 → 1B
   - Qwen 2.5 → 1.5B

3. Prompts utilizados

- Prompts Base usados por todos menos Gemma, para los modelos Google flan, Qwen y Chat gpt2:

- Base: *"Clasifica la siguiente opinión en base a como interpretes la actitud del mensaje, dicha actitud puede ser vista como positiva, negativa o neutral: {opinion}"*

- *Zero shot*: *"Clasifica la siguiente opinión como positiva, negativa o neutral: {opinion}"*

- *Few shot*: *"""*
  Aquí tienes algunas opiniones clasificadas:

  Opinión: 'Me encanta este producto. Es increíble.' → Sentimiento: positivo
  Opinión: 'Este servicio es terrible, nunca lo recomendaría.' → Sentimiento: negativo
  Opinión: 'El artículo está bien, pero no es excepcional.' → Sentimiento: neutral

  Clasifica esta opinión: {opinion}
  """

- *Chain of thought: """*
  Pensando en la siguiente opinión:

  Opinión: '{opinion}'

El sentimiento puede ser positivo, negativo o neutral. ¿Qué opinas?

"""

Los prompts utilizados para el modelo de Gemma-3 han sido iguales, pero con una indicación para que responda con una única palabra:
"...Responde con una sola palabra: 'positiva', 'negativa', 'neutral' o 'desconocida'."

4. Identificación del Sentimiento

```
  def assign_sentiment(text):
        normalized_response = text.lower().strip()

        palabras_positivas = ["positivo", "positiva", "positive"]
        palabras_negativas = ["negativo", "negativa", "negative"]
        palabras_neutras = ["neutral", "neutro", "neutra"]

        if any(word in normalized_response for word in
palabras_positivas):
            return "positiva"
        elif any(word in normalized_response for word in
palabras_negativas):
            return "negativa"
        elif any(word in normalized_response for word in
palabras_neutras):
            return "neutral"
        else:
            return "desconocida"
```

Para identificar el sentimiento hemos buscado palabras clave dentro de la respuesta del LLM a la prompt concreta. Puesto que el LLM puede generar respuestas a las prompts en inglés o en español (hemos hecho prompts en ambos idiomas), hemos elegido una serie de palabras clave en ambos idiomas (definidas arriba), para determinar si la respuesta del LLM, indica que la opinión de entrada tiene un sentimiento positivo, negativo, neutral o desconocido. Esta función es una especie de filtro que determina si la respuesta generada indica un sentimiento negativo, positivo o neutral.

Para el modelo Gemma-3 no ha sido necesario esto ya que, directamente responde con una única palabra directamente clasificable.

4. Resultados

Modelos / prompts	Google flan t5 base	Chat gpt 2 medium	Google flan t5 small	Qwen	Google gemma-3 (Respuesta de una sola palabra)	**Desempeño de los prompts (Media)**
base	0.63	0.67	0.43	0.87	0.87	**0.6940**
Zero shot	0.70	0.73	0.43	0.93	0.90	**0.738**
Few shot	0.83	0.17	0.43	0.87	1.00	**0.66**
Chain of thought	0.80	0.57	0.43	0.47	1.00	**0.654**
Median	**0.74**	**0.535**	**0.43**	**0.785**	**0.9425**	

5. Conclusiones

A pesar de haber utilizado diferentes modelos con múltiples técnicas, podemos observar que la cantidad de parámetros, si bien guarda cierta relación con el desempeño del modelo, a partir de cierta cantidad de parámetros, resulta mucho más interesante producir una prompt adecuada para el problema. Es por eso que observamos que hay modelos que, si bien tienen un número de parámetros muy inferior a otros, son capaces de generar respuestas de mayor calidad.

Es interesante destacar que, a pesar de que a priori, el método *chain of thought,* parece que debería dar mejor resultado, podemos observar que, de media, es el que peor desempeño entre todos los modelos. Sin embargo, podemos ver que, al forzar el resultado a que responda con una única palabra, el desempeño resulta mejor que el de otros. Tambien podemos ver que, aunque el método *zero shot* es el más simple, ya que no necesita de ejemplos o explicaciones, ha sido el que mejor desempeño ha tenido de media entre todos los modelos, esto quizá pueda deberse a que, al hacer la prompt más compleja, se añade ruido al sistema y eso sesgue la decisión del LLM de manera arbitraria.