OpenAI

**Universidad Alfonso X**

# TÉCNICAS DE OPTIMIZACIÓN Y CONTROL

*Práctica 5*

Autores:
Pelayo Huerta Mijares
Rubén Nogueras González

Octubre 2023

# 1. Introducción a ChatGPT

ChatGPT es un modelo de lenguaje avanzado desarrollado por OpenAI que utiliza técnicas de aprendizaje profundo para generar respuestas coherentes y contextuales en conversaciones de chat. Aunque es muy potente, a menudo necesita acceso a información contextual específica para brindar respuestas más precisas y relevantes. Los grafos de conocimiento son una herramienta valiosa para mejorar la capacidad de ChatGPT para comprender y generar respuestas contextualmente precisas.

# 2. Limitaciones de ChatGPT

La limitación más visible de ChatGPT y que conocemos todos, es que únicamente tiene información hasta 2021, por lo que su información puede ser veraz, pero probablemente no este actualizada. Esto es muy fácil de comprobar, simplemente le preguntamos algo que haya sucedido en fechas posteriores a 2021 y veremos que su respuesta es su información llega hasta 2021.

Si profundizamos un poco, nos daremos cuenta que en ocasiones ChatGPT genera información ficticia para así defender sus argumentos.

En el ejemplo que se nos presenta, unos abogados se apoyaron en ChatGPT para la demanda de una aerolínea. Lo que sucedió fue que ChatGPT les dio unos precedentes judiciales inventados, que no fueron revisados por los abogados y se presentaron en el juicio. Como consecuencia el jurado consideró que los abogados actuaron de mala fe por ofrecer declaraciones .$^{en}$gañosasz recibieron una multa de 5000$.

Apoyarse en ChatGPT o cualquier otra IA no es malo, ni mucho menos está prohibido. Sin embargo, su información no siempre es veraz, por lo que es muy recomendable contrastarla con alguna fuente más fiable y así asegurarse de que nuestra información es la correcta.

# 3. Diseño de la propuesta

## 3.1. Descripción del uso de grafos de conocimiento

Los grafos de conocimiento son representaciones estructuradas de datos que organizan información de manera semántica. Consisten en nodos que representan entidades y relaciones que conectan estos nodos. Cada nodo puede contener información detallada sobre una entidad específica, como

una persona, lugar, concepto o evento, y las relaciones entre nodos capturan la interconexión de estas entidades en un dominio de conocimiento particular.

Al usar un contexto estructurado nos asegura que los resultados sean precisos y actualizados.

### 3.2. Capturando Relaciones Semánticas

Uno de los principales beneficios de los grafos de conocimiento es que permiten capturar relaciones semánticas entre entidades. En el contexto de ChatGPT, esto significa que el modelo puede entender no solo la entidad en sí, sino también cómo se relaciona con otras entidades. Por ejemplo, si el grafo de conocimiento contiene información sobre una película y su director, ChatGPT puede utilizar esta estructura para comprender la relación "director de.[en]tre la entidad "películaz la entidad "director". Esto proporciona un contexto valioso para generar respuestas relevantes.

### 3.3. Mejorando la Búsqueda de Información

Los grafos de conocimiento también facilitan la búsqueda y recuperación de información relevante. Cuando un usuario hace una pregunta, ChatGPT puede usar el grafo para navegar de manera eficiente a través de las entidades y relaciones para encontrar la información necesaria. Esto es particularmente útil cuando se trata de consultas complejas que requieren conocimiento específico, como preguntas sobre eventos históricos, datos científicos o detalles de la cultura pop.

### 3.4. Contextualizando la Conversación

ChatGPT puede utilizar el grafo de conocimiento para mantener el contexto en una conversación. Cuando el modelo recibe una serie de mensajes en una conversación, puede referirse a entidades y relaciones previamente mencionadas en el grafo para generar respuestas más precisas y coherentes. Esto ayuda a evitar respuestas ambiguas o fuera de contexto.

### 3.5. Ampliando el Conocimiento de ChatGPT

Los grafos de conocimiento también pueden servir como una fuente de aprendizaje para ChatGPT. A medida que el modelo interactúa con usuarios y consulta el grafo, puede incorporar nuevos conocimientos y actualizar su comprensión del mundo. Esto lo hace más adaptable y capaz de mantenerse al día con la información en constante cambio.

### 3.6. Conclusiones

En resumen, los grafos de conocimiento ofrecen una herramienta valiosa para mejorar ChatGPT al proporcionar contexto estructurado en las conversaciones. Esto permite que el modelo comprenda mejor las relaciones semánticas entre entidades, mejore la búsqueda de información, mantenga el contexto en las conversaciones y amplíe su base de conocimientos. La combinación de técnicas de procesamiento de lenguaje natural con la estructura de los grafos de conocimiento puede llevar a conversaciones más informadas y relevantes con ChatGPT.

## 4. Implementación

Para nuestro experimento le preguntaremos sobre qué equipo ganó la pasada champions. Como es de esperar, no sabrá dicha respuesta y nos responderá que únicamente tiene información hasta 2021.

Así que para proporcionarle la información de una manera estructurada crearemos nuestro grafo de conocimiento, en el que añadiremos la información que necesitamos. Además, añadiremos más información sobre otros ganadores de otros títulos de fútbol para así, intentar dificultar su comprensión de nuestro grafo. Si finalmente nos da la respuesta esperada, sabremos que la implementación de grafos de conocimiento en ChatGPT realmente funciona y mejora su funcionamiento.

Lo primero que tenemos que hacer es importar las librerías que necesitamos para preguntarle quién ganó la última champions.

```
import os
import openai
```

Ahora establecemos conexión con Openai a través de nuestra API key, que podemos obtener en la web de Openai.

```
os.environ["OPENAI_API_KEY"] = "Aqui␣ponemos␣nuestro␣API␣KEY"
openai.api_key = os.environ["OPENAI_API_KEY"]
```

Formulamos la pregunta que queremos preguntarle y ejecutamos.

```
question = "What␣team␣won␣the␣2022␣champions␣league?"
```

```
completion = openai.ChatCompletion.create(model="gpt-3.5",
                                          temperature=0,
                                          messages=[{"role":"user",
                                                     "content":question,}])

print(completion["choices"][0]["message"]["content"])
```

Obviamente la respuesta que nos dará ya nos la podíamos esperar: que no tiene acceso a una información actualizada.

Para solucionar este problema, crearemos nuestro grafo de conocimiento. Para ello necesitamos importar otras librerías:

```
from langchain.graphs.networkx_graph import KnowledgeTriple
from langchain.llms import OpenAI
from langchain.indexes import GraphIndexCreator
import networkx as nx
import matplotlib.pyplot as plt
```

A continuación, le proporcionamos toda la información que queremos que contenga nuestro grafo de conocimiento, y creamos el grafo:

```
kg = [
    ("Real␣Madrid", "won", "2022␣champions␣league"),
    ("Chelsea", "won", "2021␣champions␣league"),
    ("Manchester␣City", "won", "2021␣champions␣league"),
    ("Barsa", "won", "2023␣La␣Liga"),
    ("Real␣Madrid", "won", "2022␣La␣Liga"),
    ("Atl tico␣de␣Madrid", "won", "2021␣La␣Liga"),
    ("Sevilla", "won", "2023␣europa␣league"),
    ("Frankfurt", "won", "2022␣europa␣league"),
    ("Villareal", "won", "2021␣europa␣league"),
    ("Manchester␣City", "won", "2023␣premier␣league"),
    ("Manchester␣United", "won", "2022␣premier␣league"),
    ("Manchester␣City", "won", "2021␣premier␣league"),
]

index_creator = GraphIndexCreator(llm=OpenAI(temperature=0))

graph = index_creator.from_text('')
for (node1, relation, node2) in kg:
    graph.add_triple(KnowledgeTriple(node1, relation, node2))
```

Para la visualización de nuestro grafo nos apoyamos en las librerías: matplotlib y networkx:

```
# Create directed graph
G = nx.DiGraph()
for node1, relation, node2 in kg:
    G.add_edge(node1, node2, label=relation)

# Plot the graph
plt.figure(figsize=(25, 25), dpi=300)
pos = nx.spring_layout(G, k=2, iterations=50, seed=0)

nx.draw_networkx_nodes(G, pos, node_size=5000)
nx.draw_networkx_edges(G, pos, edge_color='gray', edgelist=G.edges(), width=2)
nx.draw_networkx_labels(G, pos, font_size=12)
edge_labels = nx.get_edge_attributes(G, 'label')
nx.draw_networkx_edge_labels(G, pos, edge_labels=edge_labels, font_size=12)

# Display the plot
plt.axis('off')
plt.show()
```

Por último, le formulamos la pregunta de nuevo y veremos como ahora sí nos da la respuesta que queremos:

```
chain = GraphQAChain.from_llm(OpenAI(temperature=0), graph=graph, verbose=True)
chain.run(question)
```

## 5. Evaluación

Como hemos visto, conseguimos que ChatGPT nos diera la respuesta correcta a una pregunta sobre un evento posterior a su conocimiento. Sin embargo, su fiabilidad se puede poner en duda, ya que somos nosotros los que le hemos proporcionado esa información, pero y si hubiese sido falsa, sabría diferenciar información que es real de la que no lo es? Pues la respuesta es sencilla no, somos nosotros los que le proporcionamos la información y los que le .ªctualizamos"su conocimiento con la información que nosotros queramos. Por lo que los grafos de conocimiento son una gran herramienta para ayudar a ChatGPT en sus limitaciones, aunque se puede proporcionar información "falsa", que para él sería real.

# 6. Reflexión y Recomendaciones

Los resultados obtenidos son muy positivos, ya que podemos solucionar algunas de las limitaciones de ChatGPT. Y para finalizar, podríamos sugerir que se debería de ampliar la base de conocimientos, para incluir información más ampliada y actualizada. También se podría mejorar la comprensión y la generación de las respuestas para así conseguir una mayor precisión.