E1.1 DOCUMENTO DEL ESTADO DE LA CUESTIÓN, GUÍAS PARA LA INYECCIÓN DE CONOCIMIENTO EN LLM Y MÉTRICAS

KG4LLM - SERVICIOS PARA EL ENRIQUECIMIENTO DE MODELOS DE LENGUAJE CON GRAFOS DE CONOCIMIENTO (SER-21/23 OTT)

Resumen

Este entregable recoge resultados de las tareas T1.1, T1.2 y T1.3 y T1.4. Incluye un análisis actualizado del estado del arte, las guías con los enfoques propuestos en el proyecto para el desarrollo de métodos de inyección de conocimiento en LLM y el diseño de métricas de evaluación. El entregable propone una selección de LLM base sobre los que aplicar los métodos de inyección de conocimiento propuestos, así como los recursos necesarios en forma de grafos de conocimiento y otras fuentes de conocimiento estructurado o herramientas externas.

José Manuel Gómez Pérez
Raul Ortega
Andrés García
Flavio Merenda
Cristian Berrío

28 de febrero de 2024
Expert.ai Language Technology Research Lab

Calle Poeta Joan Maragall, 3-5, Escalera Izquierda, Planta 1ª, Derecha, 28020, Madrid
CIF: B-66425513, Inscrita en el Registro Mercantil de Madrid, en el Tomo 44.538, Folio 74, Hoja Número M-784613, Inscripción 1ª.

www.expert.ai

Historia de revisions

Revision	Date	Description	Author (Organisation)
0.1	30/01/2024	Tabla de contenidos y estructura básica	Expert.ai
0.2	01/03/2024	Primera versión completa	Expert.ai
1.0	05/03/2024	Versión final	Expert.ai

Tabla de contenidos

1 Introducción

Este entregable recoge los resultados de las tareas T1.1, T1.2, T1.3 y T1.4 del paquete de trabajo PT1 del proyecto KG4LLM, sobre el estado del arte, guías y métricas para la inyección de conocimiento en LLM. En estas tareas se ha llevado a cabo un trabajo prospectivo para definir de manera actualizada los fundamentos básicos que regirán el desarrollo del proyecto desde un punto de vista técnico y que permitirán centrar el desarrollo de métodos de enriquecimiento mediante la inyección de conocimiento adicional externo al LLM.

El documento Incluye un análisis actualizado del estado del arte, las guías con los enfoques propuestos en el proyecto para el desarrollo de métodos de inyección de conocimiento en LLM y el diseño de métricas de evaluación. También propone una selección de LLM base sobre los que aplicar los métodos de inyección de conocimiento propuestos, así como los recursos necesarios en forma de grafos de conocimiento y otras fuentes de conocimiento estructurado o herramientas externas.

El resto del documento se estructura de la manera siguiente. La sección 2 resume la estrategia propuesta alrededor de las áreas de trabajo en las que el proyecto KG4LLM propone poner especial foco. La sección 3 ofrece una visión actualizada del estado del arte en los ámbitos clave del proyecto en relación a técnicas y métodos para la evaluación de la factualidad y sesgo en LLM, la mejora de ambos aspectos en fases sucesivas de preentrenamiento y ajuste de los parámetros del modelo, así como en tiempo de inferencia y razonamiento, la habilitación de LLM para el uso de herramientas externas y la generación de grafos de conocimiento a partir de texto mediante LLM. Basado en el enfoque propuesto y el análisis del estado del arte en las áreas técnicas clave del proyecto reportado en las secciones anteriores, la sección 4 propone un nuevo marco para la evaluación de la factualidad y sesgo agnóstico del dominio y del idioma. Basado en este análisis, la sección 5 esboza los métodos que se desarrollaran en el proyecto en las áreas identificadas. La sección 6 propone una serie concreta de LLM, grafos de conocimiento y otros recursos (semi)estructurados, corpora y herramientas externas que se utilizaran en el proyecto para el desarrollo de los métodos propuestos. Finalmente, la sección 7 concluye el documento.

2 Enfoque

El análisis llevado a cabo en el entregable E4.1 Análisis y Definición de Dominios de Aplicación y Casos de Uso (Gómez-Pérez y Ortega, 2023) puso de manifiesto la escasez de recursos libremente disponibles para llevar a cabo los objetivos del proyecto KG4LLM. Estos recursos son necesarios para alimentar los métodos existentes de inyección de conocimiento estructurado en LLM con el propósito de su adaptación a dominio en el marco lingüístico del español y lenguas cooficiales. Métodos como K-Adapter (Wang et al., 2021) requieren de un grafo de conocimiento y un gran corpus de texto anotado en el que se haya etiquetado previamente las entidades y relaciones del grafo que aparecen en el corpus. Este es un proceso costoso en términos de tiempo y esfuerzo, que se complica aún más en un escenario vertical en el que escasean los grafos de conocimiento u otros recursos semánticos específicos de dominio, los corpora de texto y de verificación y los conjuntos de datos de evaluación. Más aún en escenarios multilingües como el propuesto en el marco general de INESData.

Por otro lado, recientes avances en el campo de la investigación en NLP para la mejora de la factualidad en LLM han causado un giro hacia nuevos enfoques que se alejan de los ya tradicionales métodos de inyección de conocimiento estructurado durante las sucesivas fases

de preentrenamiento del LLM, como Ernie (Sun et al., 2020), KnowBERT (Peters et al., 2019), K-Adapter (Wang et al., 2021) y Kformer (Yao et al., 2022). Entre estos avances, RAG (del inglés, Retrieval Augmented Generation; Izacard et al., 2022) se basa en el enriquecimiento en tiempo de inferencia de la petición (prompt) al LLM con información recuperada de una base de conocimiento local o de motores de búsqueda en Internet, con el fin de complementar el conocimiento aprendido por el LLM durante su preentrenamiento. Frente a los métodos tradicionales de inyección de conocimiento mencionados más arriba, el gran tamaño de las ventanas de contexto aceptado por los modernos LLM y el moderado coste en términos de infraestructura de computación y volumen de datos en inferencia favorece el uso de RAG.

Entre estos avances recientes también cabe destacar de manera significativa el algoritmo DPO (del inglés, Direct Preference Optimization) propuesto por Raffailov et al. (2023). DPO permite alinear el texto generado por el LLM con un dataset de preferencias relativas a dimensiones como seguridad, ausencia de lenguaje tóxico o efectividad de la respuesta generada, sin necesidad de entrenar un modelo de recompensa (Christiano et al., 2017). Estudios recientes como (Tian et al., 2023b) han demostrado que DPO es también un método efectivo para alinear LLM con un objetivo de factualidad como una dimensión más.

Teniendo en cuenta las limitaciones detectadas durante las fases iniciales del proyecto y los avances recientes mencionados más arriba, el presente documento de guías se propone alinear la propuesta original del proyecto KG4LLM con las posibilidades ofrecidas por estos nuevos enfoques para afrontar los retos asociados a un escenario de escasez de recursos semánticos, corpora y conjuntos de datos de evaluación multilingües y específicos de domino. De este modo, proponemos una estrategia centrada en las siguientes áreas de trabajo principales, que introducimos a continuación:

- **Desarrollo de un marco de trabajo para la evaluación de factualidad y sesgo en LLM agnóstico de dominio e idioma.** Este marco de evaluación permitirá verificar el texto generado por el modelo contra datasets de verificación a nivel de dominio. El marco de evaluación también ofrecerá métodos intrínsecos basados en la confianza del modelo en el texto generado, de manera independiente de la existencia de un dataset de verificación, aliviando la dependencia de la existencia de este tipo de recursos para un idioma y dominio dados.
- **Métodos de mejora de la factualidad mediante inyección de conocimiento en LLM basados en DPO.** El objetivo de estos métodos será aumentar la proporción de hechos factuales generados por el LLM mientras se reduce la cantidad de alucinaciones o hechos no factuales. Los métodos propuestos buscarán ajustar los parámetros de un LLM previamente entrenado sobre un objetivo de entrenamiento cuya función de pérdida se centrará en optimizar la métrica de factualidad calculada mediante el marco de trabajo de evaluación introducido en el punto anterior. Los métodos propuestos buscarán aportar flexibilidad con respecto de la disponibilidad de recursos estructurados para inyectar conocimiento factual de dominio en el LLM en los distintos idiomas objetivo. Dichos recursos pueden incluir, de menor a mayor expresividad: listas de entidades, diccionarios, tesauros, taxonomías o grafos de conocimiento estructurados semánticamente.
- **Métodos de mejora de factualidad y sesgo en LLM mediante PSP y RAG.** Nos enfocamos en la combinación de métodos de prompting y RAG que guíen al LLM para llevar a cabo de forma efectiva el razonamiento requerido y faciliten la incorporación de conocimiento adicional en tiempo de inferencia que complemente la información ya

contenida en los parámetros del modelo. Los métodos de prompting propuestos avanzarán el estado del arte relacionado con técnicas como Chain of Thought (CoT; Wei et al., 2023) y plan-and-solve (PS; Wang et.al, 2023) y explorarán el uso de métodos de resolución de problemas (del inglés, Problem-Solving Methods o PSM) con el fin de guíar de manera estructurada el razonamiento del LLM y mejorar su explicabilidad. Proponemos un nuevo método que combina las técnicas de CoT y PS con PSM, al que nos referimos como Problem-Solving Prompting (PSP). También se estudiará el impacto de estos métodos en la factualidad del LLM en relación con el tamaño del modelo y el régimen de entrenamiento para los distintos idiomas y dominios. Esperamos que estas estrategias de resolución de problema combinadas con métodos como SelfRAG también ayuden a determinar qué información externa necesita incorporar un LLM para resolver una petición de usuario y cómo recuperar esa información de manera efectiva.

- **Métodos de alineamiento con preferencias basado en DPO para el uso de herramientas externas.** Al igual que es posible generar un dataset de preferencias y entrenar un LLM bajo un régimen DPO para hacerlo más factual y menos sesgado, planteamos el desarrollo de métodos basados en DPO para entrenar un LLM en la tarea de decidir cuándo generar texto normal como parte de la respuesta a un prompt y cuándo generar una llamada al API de una herramienta externa. A diferencia de métodos como Toolformer (Schick et al., 2023), que necesita un corpus de entrenamiento anotado de gran volumen basado en CCNet (Wenzek et al., 2020), nuestra hipótesis es que un método basado en DPO necesita menos esfuerzo computacional para generar un dataset de preferencias. Dicho dataset podría ser obtenido mediante el muestreo de Toolformer y su modelo base GPT-J (Wang and Komatsuzaki, 2021) para generar salidas preferidas frente a alternativas rechazadas.
- **Métodos para la generación de grafos de conocimiento semánticos a partir de texto mediante LLM.** Muchos de Los LLM considerados grandes, por ejemplo, con un número de parámetros superior a 175B, como GPT-3.5, ya fueron preentrenados sobre datos anotados siguiendo ontologías o vocabularios ampliamente aceptados, como schema.org o dublincore.org. Eso les permite generar datos anotados a partir de texto usando esas ontologías. Sin embargo, el escenario cambia cuando las ontologías a utilizar no están contenidas en el conjunto de datos con el que se entrenó el LLM. Los métodos propuestos explorarán distintos enfoques para generar un grafo de conocimiento mediante LLMs, de acuerdo con una ontología posiblemente arbitraria. Estos enfoques incluyen métodos de inyección de las ontologías en el prompt de la petición y la generación mediante LLM de un dataset anotado para entrenar otros LLM más pequeños en la tarea de traducir texto a un grafo de conocimiento estructurado sobre un conjunto de ontologías determinado.

3 Análisis actualizado del estado del arte

3.1 Evaluación de factualidad y sesgo en LLM

El desarrollo de un marco de evaluación de factualidad y sesgo que permita evaluar los LLM sobre esas dimensiones es crucial para determinar el impacto de los métodos desarrollados en el proyecto y guiar su evolución. Para lograr ese objetivo hay que tener en cuenta los retos que KG4LLM afronta, que incluyen:

- Especificidad de dominio.

- Escenario multilingüe que tenga en cuenta el español y lenguas oficiales.
- Baja disponibilidad de recursos textuales (corpora de entrenamiento y de verificación) y semánticos (grafos de conocimiento, taxonomías, tesauros, etc.) para los dominios verticales de posible interés e idiomas objetivo.

Por tanto, nos interesan aquellos métodos de evaluación que sean flexibles en cuanto al idioma para el que queremos evaluar los LLM y a la disponibilidad de los recursos mencionados. A la luz de estos retos, nos basamos en avances recientes en la estimación de la veracidad sin intervención humana, que podemos clasificar en: a) métodos basados en referencias, que evalúan en qué medida una base de conocimiento externo respalda las afirmaciones formuladas en un texto (Min et al., 2023; Chern et al., 2023) y b) métodos de evaluación que utilizan la propia confianza de un LLM en el texto generado como indicador de veracidad, inspirados en (Kuhn et al., 2023).

En cuanto al análisis de sesgo y LLMs, la literatura ha puesto atención en la evaluación de dos tipos de sesgos, principalmente: a) intrínsecos, que se introducen durante el preentrenamiento de los LLM, y b) extrínsecos, que se refieren a los sesgos generados durante el entrenamiento de tareas específicas (Delobelle et al., 2022). Dependiendo del tipo de tarea, del sesgo que se desea evaluar y de la estructura de los datos de entrada y salida, se pueden aplicar diversas métricas de sesgo (Gallegos et al., 2023). Al igual que en la evaluación de factualidad en LLM, en este caso también nos referimos a métodos flexibles que tengan en cuenta las limitaciones mencionadas y sean capaces de evaluar sesgos en entornos multilingües (Vashishtha et al., 2023), posiblemente generando recursos con métodos que no requieran anotaciones manuales, como la estrategia de perturbación de datos propuesta originalmente por Qian et al. (2022).

3.1.1 Conjuntos de datos para la evaluación de la factualidad

A la hora de evaluar la factualidad de los modelos, la literatura nos ofrece diferentes herramientas, tanto en términos generales como para dominios específicos. Hemos recopilado algunas de ellas, junto a otros conjuntos de datos que tratan de resolver tareas como resolución de preguntas multirrespuesta y de respuesta abierta, pero que ponen el foco en la factualidad de los modelos que tratan de resolverlas.

FACTOR (Muhlgay et al., 2024). Propone un método con el que transformar de forma automática un corpus con información veraz en un marco de evaluación de factualidad de modelos. Esta transformación se lleva a cabo seleccionando frases del corpus de forma aleatoria. A partir cada una de ellas y, utilizando su contexto previo, se generan una nueva colección de frases mediante llamadas a InstructGPT (Ouyang et al., 2022), ajustando el prompt para que la salida del modelo incluya varios tipos de errores (de entidad, de predicado, de circunstancia, de correferencia y de enlace). Después de un filtrado usando modelos de entailment y por valor de perplejidad, se utiliza esta colección de frases incorrectas junto a la correcta para formular una tarea de resolución de preguntas multirespuesta. Todo este marco permitiría generar conjuntos de datos de evaluación en varios dominios e idiomas, siempre que se tenga un recurso verificado en ambas dimensiones. Además, Muhlgay et al. (2024) presentan tres conjuntos de datos, con diferentes grados de especificidad: Wiki-FACTOR (basado en la sección de artículos de Wikipedia del subset de evaluación de The Pile, Gao et al., 2020), News-FACTOR (basado en artículos de Reuters posteriores a octubre de 2021, extraídos de RefinedWeb, Penedo et al., 2023) y Expert-FACTOR (basado en los subconjuntos de evaluación de ExpertQA, Malaviya et al., 2023)

TruthfulQA (Lin & Hilton et al., 2022). Se trata de un marco de evaluación de la factualidad de modelos, en formato de preguntas multirrespuesta, generado de forma manual en busca de posibles fallos en LLMs generativos debido a conceptos erróneos adquiridos por repetición en los textos con los que ha sido entrenado. Está compuesto por 817 preguntas de 38 categorías, que incluyen salud, legislación, finanzas y política. Usando como baselines modelos de la familia GPT y basados en T5, tan solo se consigue un 58% de acierto en este conjunto de datos, mientras que el desempeño humano estaría en torno al 94%.

FELM (Chen et al., 2023). En este marco de evaluación se han recolectado y anotado respuestas generadas por LLMs con etiquetas de factualidad. Estas anotaciones están orientadas a diversas tareas como conocimiento general, razonamiento, escritura, matemáticas y ciencias, y otorgan información acerca de qué tipo de error está cometiendo el modelo, el razonamiento detrás de la anotación y la referencia a la información dada en el razonamiento. La construcción de FELM fue realizada usando segmentos de texto provenientes tanto de plataformas online (Quora, Twitter), como de otros otros marcos de evaluación como MMLU (Hendrycks et al., 2021) y TruthfulQA (Lin & Hilton et al., 2022), y de generaciones de ChatGPT (Wang & Kordi et al., 2023).

Otros conjuntos de datos como StrategyQA (Geva et al., 2021), que requieren tanto conocimiento implícito como explícito para responder cada pregunta, también pueden servir como marcos para la evaluación de conocimiento factual de dominio general para LLM.

3.1.2 Evaluación de la factualidad basada en referencias

Además de los retos asociados a KG4LLM mencionados con anterioridad, la identificación de errores factuales durante la generación de texto mediante LLM presenta otra serie de desafíos, como la longitud y dispersión de los datos generados o la falta de evidencias que soporten sus afirmaciones. La evaluación humana de los textos generados por un LLM es costosa tanto en tiempo como en recursos, por lo que se hace necesario la aplicación de métodos que permitan atajar este problema de forma automática o semiautomática.

La literatura nos muestra una serie de marcos de trabajo que permiten, por un lado, articular el conocimiento disgregado a lo largo de los textos generados por los LLM en una lista de afirmaciones textuales atómicas y evaluables, llamadas claims, y por otro, la aplicación de diferentes métodos para contrastar estos claims con la información alojada en bases de conocimiento confiables.

FacTool (Chern et al., 2023) propone extraer los claims de un texto generado mediante una consulta a otro LLM base (gpt-3.5-turbo). Para ello definen el claim siguiendo la definición de ACUs (Atomic Content Units) dada por Liu et al. (2023), restringiendo su longitud a 15 palabras como máximo y a aquellos que representen un hecho evaluable. También se proponen otros métodos para la extracción de claims teniendo en cuenta otras tareas como la generación de código, la resolución de problemas aritméticos o la revisión de las referencias a artículos científicos.

Una vez con el listado de claims, Chern et al. (2023) propone generar para cada una dos consultas a diferentes herramientas externas, dependiendo del tipo de tarea a resolver (motores de búsqueda en internet, intérprete de Python, etc.). El texto de estas consultas se genera nuevamente con un LLM base como ChatGPT o GPT-4. El resultado de las consultas forma el conjunto de evidencias con el que contrastar el listado de claims. Por último, siguiendo el método de cadenas de pensamiento (del inglés, chain of thought o CoT) propuesto por Wei et al. (2023), se vuelve a consultar a ChatGPT o GPT-4 para evaluar la factualidad de los claims teniendo en cuenta las evidencias encontradas. El funcionamiento general de FacTool se ve reflejado en la Figura 1.

Figura 1. Pipeline propuesto por FacTool

FactScore (Min et al., 2023) plantea un marco para la evaluación automática de la factualidad similar a FacTool, aunque con varias diferencias. Al igual que FacTool, FactScore propone reducir el texto generado a un listado de claims que permita su evaluación de forma individual, aunque en su caso lo hacen mediante InstructGPT tras observar los resultados reflejados en (Chen et al., 2022), que muestran una alta semejanza de criterios entre este LLM y la anotación humana. La recuperación de evidencias relacionadas con cada claim se realiza mediante un sistema denso de recuperación de textos basado en GTR (Generalizable T5-based Retrievers) propuesto por Ni et al. (2022), recolectando cinco evidencias por cada claim, y usando un volcado de Wikipedia de enero de 2023 como base de verificación, ya que esta evaluación fue propuesta en el marco de la generación de biografías. Por último, para la evaluación final de los pares de claim-párrafo de verificación, proponen realizar una consulta a otros LLMs como LLaMa 65B (Touvron et al., 2023), Inst-LLaMa (Wang et al., 2022) y ChatGPT, siendo este último el que reporta mejores resultados. El prompt a estos LLMs consistiría en añadir el texto "*¿Verdadero o Falso?*" (*"True or False?*") al claim generado, evaluando con y sin el listado de evidencias. También proponen un método de evaluación basado en similitud no paramétrica, en el que se enmascara cada token mediante un modelo no paramétrico (Min & Shi et al., 2023), y se hace la media de las probabilidades para cada uno de ellos para asignar la etiqueta de "Soportado" o "No soportado" al claim. A la luz de los resultados proporcionados en Min et al. (2023), el método de evaluación más consistente y con menor ratio de errores sería un agregado entre la evaluación de claims junto a la lista de evidencias y al método de similitud no paramétrica, dando por "soportado" aquel claim en el que ambos métodos coincidan en asignar dicha etiqueta. Como muestra la Figura 2, el resultado final de la evaluación de la generación realizada por el modelo será el resultado de la evaluación de cada uno de sus claims.

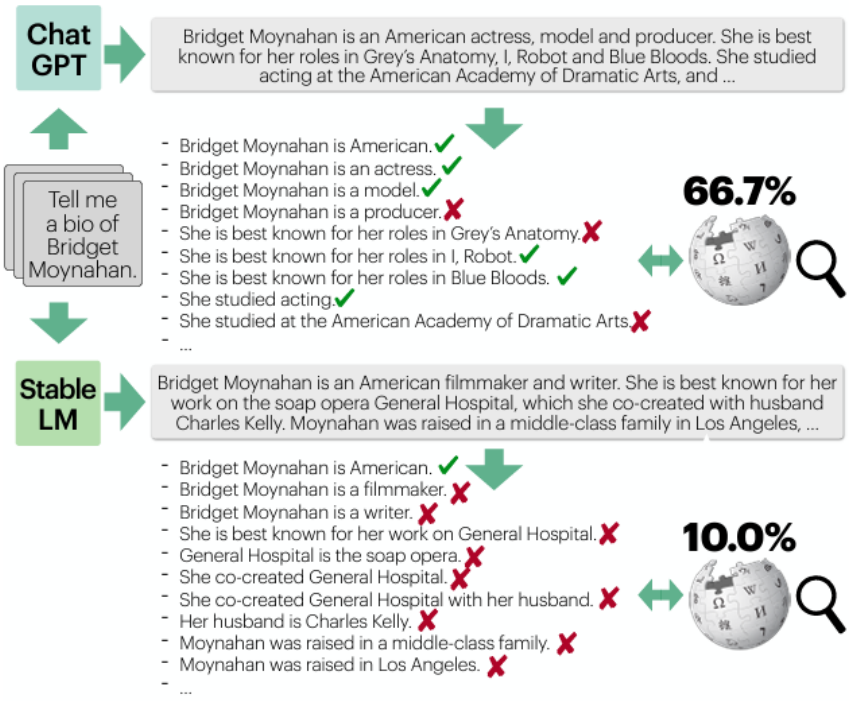

Figura 2. Ejemplo de evaluación de dos generaciones de texto utilizando FactScore.

3.1.3 Evaluación de la factualidad basada en la confianza del LLM

Para eliminar la necesidad de conocimiento externo, Tian et al. (2023b) propone un método que aprovecha observaciones relacionadas con la calibración de los LLM (Kadavath et al., 2022; Tian et al., 2023a). Según los resultados de estos trabajos, la confianza de un LLM en la respuesta generada está correlacionada con la probabilidad de que sea factual. De manera similar a (Tian et al., 2023b) otros métodos como SelfCheckGPT (Manakul et al., 2023) también utilizan la confianza del módelo en sus propias predicciones para evaluar la factualidad del texto generado.

Sin embargo, un párrafo puede contener muchos hechos, así como elecciones estilísticas particulares que tendrán un impacto significativo en la probabilidad total que un modelo asigna al texto. Por tanto, al igual que métodos basados en referencia como FacTool y FactScore, el método de Tian et al. primero lleva a cabo un paso de extracción de claims del texto y a continuación calcula la confianza promedio del modelo sobre todos los hechos extraídos como puntuación final.

Concretamente, el método propuesto por Tian et al. (2023b) primero extrae claims atómicas del texto utilizando un LLM grande, por ejemplo, GPT-3.5. A continuación usa ese mismo modelo para convertir cada claim en una pregunta mínimamente ambigua con la que evaluar el conocimiento del LLM sobre ese hecho en particular y muestrea una respuesta del LLM bajo evaluación un número de veces (20, según los autores) con el fin de estimar la incertidumbre del modelo sobre la respuesta generada. La proporción de las apariciones de la respuesta más frecuente sobre el total de respuestas obtenidas es la puntuación de veracidad final asignada a cada claim, que esencialmente representa la confianza del modelo en su factualidad.

Como se puede apreciar, este método es extremadamente flexible al no depender de la existencia de una base de conocimiento de verificación en el idioma para el que nos proponemos evaluar el LLM. Por el contrario, confía en el conocimiento del dominio y las capacidades que

para ese idioma puede tener el LLM utilizado para generar claims atómicas y las consiguientes preguntas sobre hechos del dominio de interés.

3.1.4 Métodos para la evaluación de sesgo en LLM

En la literatura se emplean diversos métodos para evaluar sesgos en LLM, los cuales dependen principalmente de la tarea en cuestión y, como consecuencia, de la arquitectura del modelo. Dependiendo de cómo se utilice el LLM al resolver una tarea, se pueden identificar tres diferentes tipologías de evaluación, cada una con sus respectivas métricas, basadas en embeddings, probabilidades y texto generado (Gallegos et al., 2023).

1. **Métodos basados en embeddings (Figura 3).** Las representaciones vectoriales densas de un modelo se emplean para medir el sesgo, comúnmente calculando la distancia en el espacio vectorial entre representaciones neutrales y representaciones marcadas. Un ejemplo de esta metodología sería calcular la similitud coseno para contrastar las representaciones de palabras profesionales, tales como "doctor" y "enfermera", con términos vinculados a grupos sociales, como "hombre" y "mujer". Las representaciones no sesgadas deberían mostrar una similitud coseno similar para diferentes términos pertenecientes a grupos sociales. Aunque existen métricas que utilizan representaciones a nivel de palabras, como WEAT (Caliskan et al., 2017), las más utilizadas son aquellas que emplean representaciones a nivel de frase, como SEAT (May et al., 2019), CEAT (Guo & Caliskan, 2021) y Sentence Bias Score (Dolci et al., 2023). Dado el tipo de salida requerida por el modelo, estos métodos se utilizan principalmente para evaluar encoders. (Zhou et al., 2023).

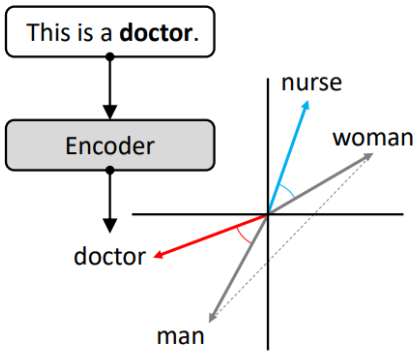

Figura 3. Método de evaluación de sesgos que utiliza representaciones vectoriales densas (embeddings).

2. **Métodos basados en probabilidades (Figura 4)**. Para estimar sesgos, este método utiliza las probabilidades asignadas a las palabras por los LLM. Existen dos tipos de técnicas: "masked token" y "pseudo-log-likelihood". La primera técnica consiste en presentar al modelo dos frases similares que se distinguen por las entidades que se mencionan, por ejemplo, "**él** es bueno en [MÁSCARA]" y "**ella** es buena en [MÁSCARA]", y analizar el comportamiento del modelo en determinar la siguiente palabra más probable para completar la frase. Ejemplos de esta estrategia son DisCo (Webster et al., 2020), LPBS (Kurita et al., 2019) y Categorical Bias Score (Ahn & Oh, 2021). La segunda técnica aproxima la probabilidad de un token condicionado al resto de los tokens que forman una frase. Este proceso implica enmascarar un token a la vez y predecirlo utilizando todos los demás tokens sin enmascarar. Ejemplos de métodos basados en esta estrategia son CrowS-Pairs Score (Nangia et al., 2020), CAT (Nadeem et al., 2021), AUL (Kaneko et. al, 2022) y AULA (Kaneko et. al, 2022).

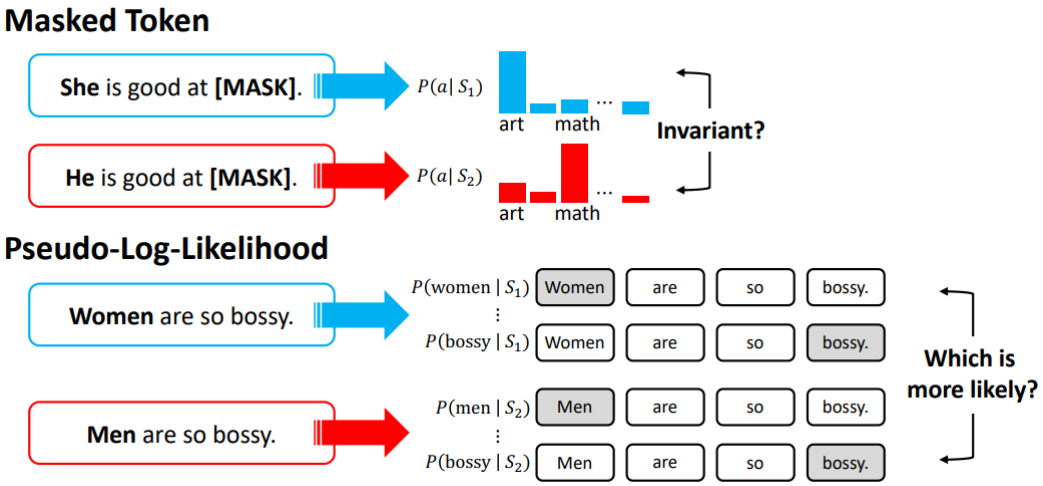

Figura 4. Dos tipologías diferentes de métodos de evaluación de sesgos basados en probabilidades.

3. **Métodos basados en texto generado (Figura 5)**. Estos son los métodos más comunes para evaluar LLMs con arquitectura decoder o más conocidos como modelos generativos (Zhou et al., 2023). En este tipo de modelos no es factible utilizar métodos de evaluación de sesgos como el de los embeddings o las predicciones, dado que el proceso de entrenamiento no se ha diseñado para optimizar la generación de representaciones vectoriales, es decir, para el encoding. Estos modelos son más similares a cajas negras y requieren como entrada frases de inicio que el modelo utiliza para continuar la generación de texto. Considerando esta limitación, los métodos de evaluación se centran en analizar el texto plano generado por el modelo. Existen tres métodos que se basan en esta estrategia, denominados de distribución (distribution), clasificación (classification) y léxico (lexicon). Con el primer método se compara la distribución de las palabras generadas entre dos grupos distintos en estudio. La premisa es que un modelo sin sesgos debería compartir la misma distribución entre ambos grupos. Métricas que utilizan esta estrategia son SGS (Rajpurkar et al., 2016), Co-Occurrence Bias Score (Bordia & Bowman, 2019), DR y ST (Bommasani et al., 2023). Las métricas basadas en clasificadores dependen de un modelo auxiliar para evaluar las salidas de texto generadas y analizar cualquier dimensión de sesgo como toxicidad, sentimiento etc. El sesgo puede identificarse cuando el texto generado a partir de entradas similares, pero que hacen referencia a diferentes grupos sociales, es clasificado de manera distinta. Métricas que se apoyan en esta tipología de análisis son EMT and TP (Gehman et al., 2020), TF (Bommasani et al., 2023), Counterfactual Sentiment Bias (Huang et al., 2020), Regard Score (Sheng et al., 2019) or Full Gen Bias (Smith et al., 2022). Por último, el método basado en léxico utiliza recursos previamente recopilados con palabras específicamente sesgadas, o que presentan un puntaje de sesgo, para identificar asociaciones con grupos de referencia. HONEST (Nozza et al., 2021), Psycholinguistic Norms y Gender Polarity (Dhamala et al., 2021) pertenecen a este grupo.

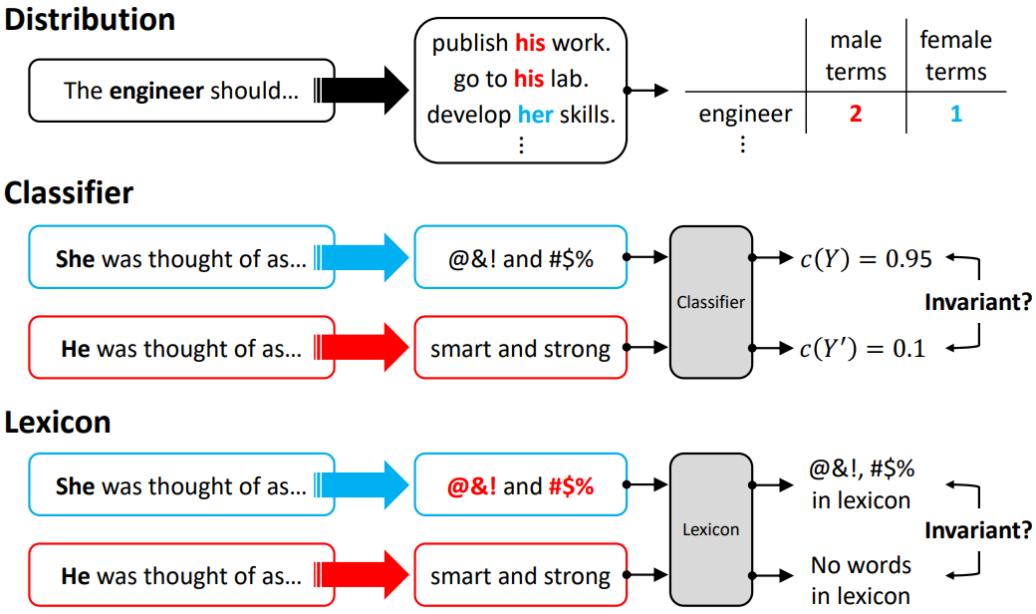

Figura 5. Tres diferentes métodos para evaluar sesgos in LLMs utilizando texto generado por el modelo.

3.2 Métodos de mejora de la factualidad en LLM

3.2.1 Decoding by Contrasting

Según Ji et al. (2023), los LLM entrenados en una tarea de predicción del siguiente token acaban usando sus capacidades lingüísticas para reconocer patrones en lugar de generar hechos factuales del corpus de entrenamiento. Esto da lugar a generación de textos incorrectos y a alucinaciones por parte de estos modelos. Por otro lado, se ha demostrado que el contenido factual tiene tendencia a distribuirse en las neuronas de las capas superiores en modelos tipo BERT (Dai et al. 2022), llegando a ser posible su modificación aplicando cambios en estas capas mediante un transformer autorregresivo (Meng et al. 2022).

En DoLa (Chuang et al. 2023) se parte de esta premisa, en la que los LLM van incorporando información factual de forma gradual en cada una de sus capas, y aplican una decodificación por contraste de las diferencias entre dos capas para reconfigurar la salida del modelo. De este modo, si la probabilidad de generar un token entre las capas n y n+i aumenta de forma significativa, ese token tendrá un mayor peso en la salida del modelo al aplicar esta decodificación, tal y como se muestra en la Figura 6.

Al aplicar esta operación, los autores de DoLa afirman haber mejorado el desempeño de modelos tipo LLAMA en un 12-17% en preguntas abiertas de TruthfulQA y un 2%-4% en preguntas multi-respuesta de FACTOR. También reportan mejoras con respecto a los baselines en tareas de CoT, como en StrategyQA y GSM8K (Cobbe et al., 2021). Cada uno de estos conjuntos de datos y modelos evaluados requirió la selección de un rango diferente de capas con las que aplicar la decodificación por contraste, siguiendo una validación 2-fold.

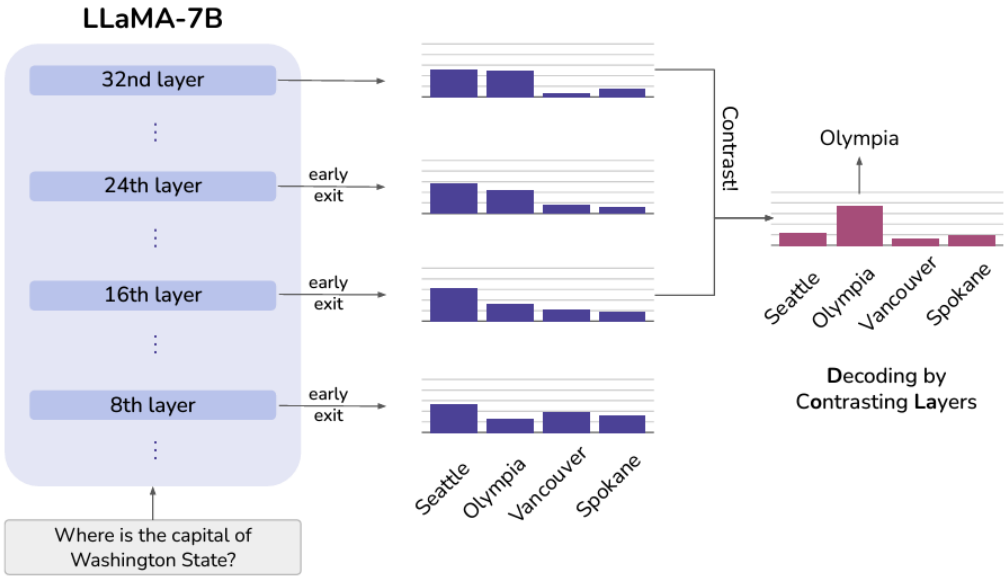

Figura 6. Resumen del funcionamiento de DoLa.

3.2.2 Inference-Time Intervention

Inference-Time Intervention (ITI) es una técnica para mejorar la factualidad de los LLM propuesta por Li et al. (2023). Consiste en el desplazamiento de las activaciones del modelo en tiempo de inferencia, de manera que éstas reconfiguren la salida para obtener una más factual, tal y como muestra la Figura 7. Esta idea nace del estudio del comportamiento interno de los LLM, y en cómo en ciertas direcciones dentro de sus espacios de activación acaban jugando un papel fundamental en la salida final del modelo. De esta manera, dirigir estos espacios de activación a unos más "veraces", debería mejorar la factualidad de los LLM.

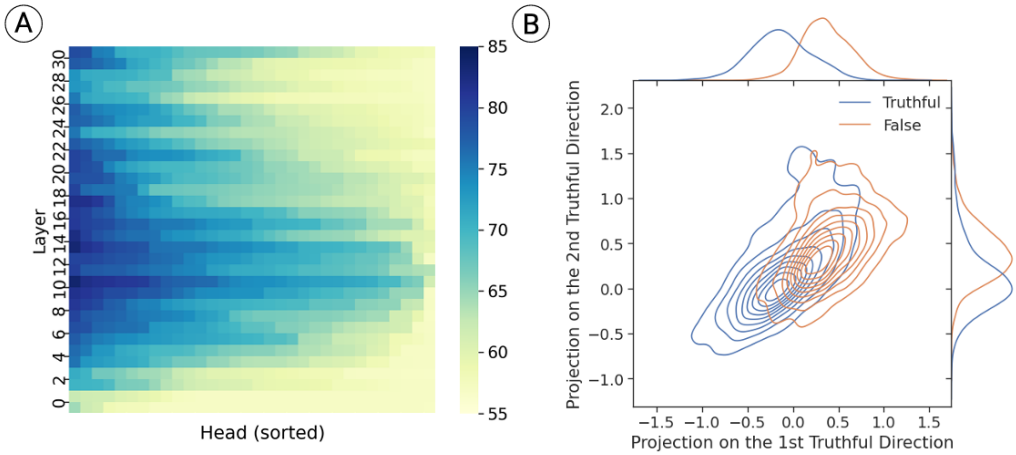

Figura 7. Análisis de los mapas de activación llevado a cabo siguiendo las pautas de ITI

La ventaja de esta técnica es que permite, con un número reducido de ejemplos, mejorar el desempeño de los LLM en conjuntos de datos de evaluación de la factualidad como TruthfulQA. Así, con unos pocos cientos de ejemplos se estudia el mapa de activaciones de cada cabeza de atención en cada capa del transformer y se calcula la activación que maximizaría el número de respuestas veraces. Li et al. (2023) reportan mejoras de entre un 32.5% hasta un 65.1% de

acierto en TruthfulQA con respecto a los baselines, usando diferentes modelos de la familia LLaMA.

3.2.3 FactTune

Una vez contamos con un método que nos permite estimar la factualidad de un LLM, es posible construir un conjunto de datos de preferencias para ajustar la factualidad de un modelo de lenguaje determinado a partir de un conjunto de prompts sin etiquetar. Utilizando los estimadores basados en referencias o en la confianza del modelo descritos en la sección 3.1.3, Tian et al. (2023b) propone un método llamado FacTune, que incluye dos variantes: FacTune-FS, en el que se utiliza FactScore como estimador de factualidad, y FacTune-MC, basado en la confianza del modelo en la respuesta generada.

Para cada pregunta generada de la forma descrita en la sección 3.1.3, FacTune elige como respuesta preferida para el conjunto de datos de preferencias aquella que tiene una puntuación de factualidad más alta según el estimador que se esté utilizando y, como respuestas rechazadas, todas las demás respuestas generadas por el LLM para esa pregunta. A continuación, FacTune construye el dataset de preferencias, formado por pares *<respuesta preferida, respuesta rechazada>* asociados a cada pregunta. Finalmente, se entrena el LLM, ajustando sus parámetros mediante el pipeline de entrenamiento DPO aplicado sobre ese dataset, usando todas las respuestas del modelo como objetivo de entrenamiento en esta fase de ajuste supervisado o SFT (del inglés, supervised fine-tuning).

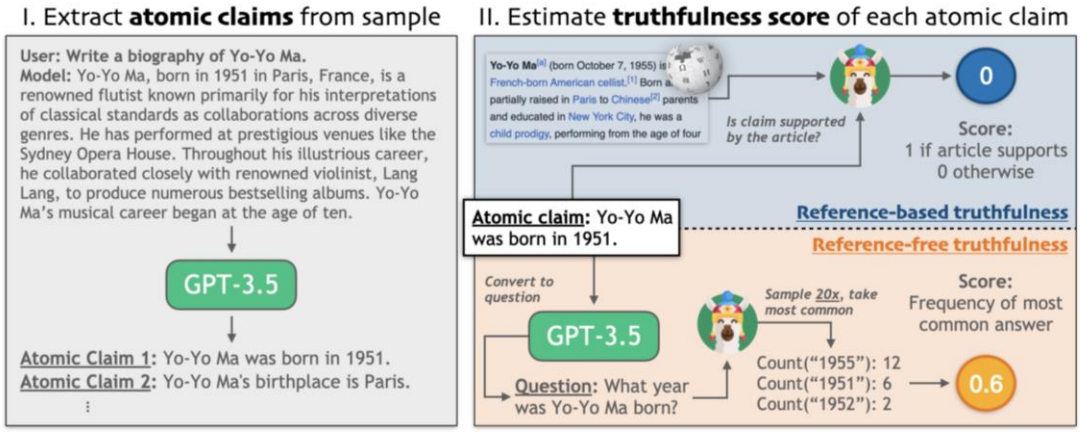

Figura 8: Intuición general del método de evaluación de factualidad en LLM mediante FacTune.

Como se puede apreciar en la Figura 9, la evaluación llevada a cabo por Tian et al. muestra que ambos métodos de generación del conjunto de datos de preferencias consiguen reducir el número de hechos incorrectos generados a la salida del LLM tras su entrenamiento DPO comparado con el modelo base. Sin embargo, sólo FacTune-FS consigue además aumentar el número de hechos correctos generados. Por otro lado, ambos ofrecen mejores resultados que baselines como ITI (Li et al., 2023, ITI) y DoLa (Chuang et al., 2023). En cualquier caso, la flexibilidad ofrecida por FacTune-MC frente a barreras como la disponibilidad de un dataset de verificación en dominios con pocos recursos lingüísticos lo convierte en un método atractivo para este tipo de escenarios de baja disponibilidad de recursos.

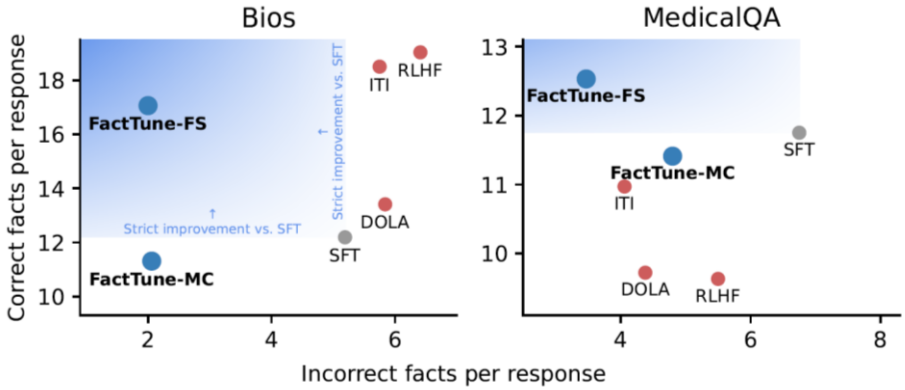

Figura 9: Tanto FacTune-MC como FacTune-FS consiguen reducir el número de hechos incorrectos generados a la salida del LLM para las tareas de evaluación del artículo de Tian et al., generación de biografías y respuesta de preguntas médicas. Sin embargo, sólo Factune-FS mejora también el ratio de hechos correctos generados. En general, ambas variantes de FacTune mejoran en ambos aspectos los resutlados de las baselines DoLa, IT, RLHF y SFT.

3.3 Métodos de prompting y RAG

El incremento en el número de parámetros en modelos generativos ha ampliado su capacidad para producir texto de manera coherente y exhibir cierto capacidad de razonamiento. (Wei et al., 2022). Numerosos investigadores han comenzado a estudiar empíricamente este fenómeno y a desarrollar estrategias para aprovechar y mejorar este comportamiento (Qiao et al., 2022). Métodos para guiar la generación de texto (prompting) y aumentar sus capacidades mediante recuperación de información (RAG) se han destacado como opciones para mejorar los resultados de los LLM. Métodos como Chain-of-Thought (Wei et al., 2022), Plan-and-Solve (Wang et al., 2023), Retrieval-Augmented Generation (Gao et al., 2024) y Self-Retrieval-Augmented Generation (Asai et al., 2023) se han aplicado para mejorar la resolución de problemas complejos. Además de mejorar el comportamiento de los modelos, estos métodos también aumentan su nivel de explicabilidad. En esta sección se presentarán con más detalle estas tipologías de estrategias, incluyendo sus versiones más recientes.

3.3.1 Chain-of-Thought

La cadena de pensamiento (del inglés Chain-of-Though o CoT, Wei et al., 2022) es una estrategia de prompting que induce un modelo a solucionar una tarea de manera secuencial reproduciendo los pasos necesarios para llevar a cabo un proceso de razonamiento. Los autores de este estudio muestran que modelos de lenguaje suficientemente grandes pueden generar cadenas de pensamiento si se proporcionan ejemplos de cadenas de pensamiento como parte del texto que se pasa al modelo en forma de prompt (Figura 10). Esta estrategia permite que el modelo mejore su procesamiento y genere texto más coherente, lógicamente más preciso en sus pasos y soluciones. Este método ofrece diversas ventajas como: i) descomponer problemas complejos en pasos intermedios para asignar recursos según sea necesario, ii) proporcionar una ventana interpretable para entender el comportamiento del modelo y depurar posibles errores de razonamiento, iii) es aplicable a una amplia gama de tareas, incluyendo problemas matemáticos, razonamiento de sentido común y manipulación simbólica y iv) se puede implementar fácilmente en modelos de lenguaje previamente entrenados. CoT demuestra excelentes resultados en varios benchmarks que evalúan LLMs en problemas matemáticos, razonamiento de sentido común y manipulación simbólica. Sin embargo, su ventaja se observa

principalmente en modelos con un número significativo de parámetros, que se sitúan en torno a los 100 billones.

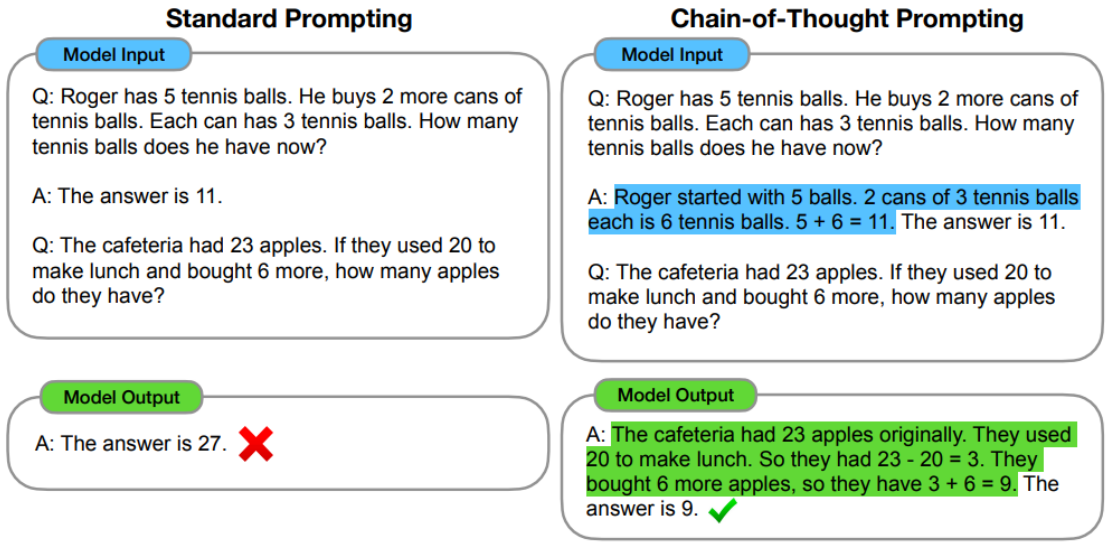

Figura 10. El método de Chain-of-Though permite a los LLMs abordar tareas complejas de aritmética, sentido común y razonamiento simbólico.

Un refinamiento de CoT que no necesita ejemplos explicitos en el prompt para inducir el modelo a resolver una tarea razonando paso a paso es “Zero-Shot CoT” (Kojima et al., 2022). Una estrategia de prompting que no se basa en ningún ejemplo previo para producir resultados se llama "zero-shot prompting". En la Figura 11 se muestra un ejemplo de Zero-Shot CoT que requiere dividir el prompt en dos pasos para funcionar correctamente.

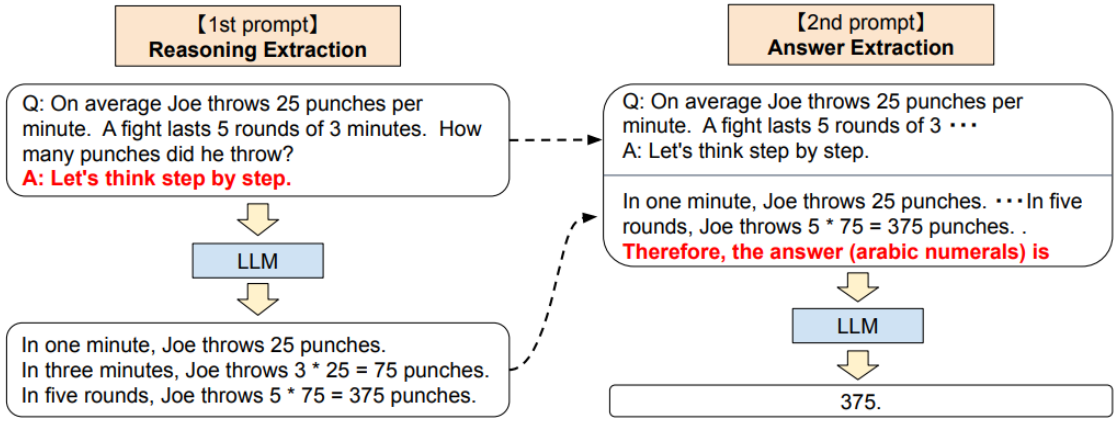

Figura 11. El método de activación Zero-Shot CoT permite iniciar la cadena de razonamiento en un LLM sin mostrar explícitamente un ejemplo, dividiendo el proceso en dos pasos para alcanzar una solución.

En el primer paso, llamado extracción de razonamiento, se solicita al modelo que responda, guiándolo a razonar paso a paso mediante una frase o trigger que induzca este comportamiento. En el trabajo donde se propone esta estrategia, la frase es "Razonemos paso a paso" (o "Let's think step by step" en inglés). En el segundo paso, o extracción de respuesta, se inserta una frase en el prompt que induzca al modelo a devolver el formato de respuesta deseado. En el contexto de este trabajo, la frase es “Por lo tanto, la respuesta (numeración arábiga) es” (o “Therefore, the answer (arabic numerals) is” en inglés). Los dos pasos se pueden formalizar de la siguiente

manera: 1) “Q: [X]. A: [T]”, donde [X] es la posición donde va el texto de entrada y [T] es la posición de la frase activadora del comportamiento. 2) "[X’] [Z] [A]" donde [X’] se refiere a la posición donde se repite el prompt completo del primer paso, [Z] para la frase generada por el primer prompt, y [A] para una nueva frase de activación para extraer la respuesta. El prompt para este paso es auto-aumentado, ya que contiene el texto generado por el mismo modelo de lenguaje en el primer paso.

3.3.2 Plan-and-Solve

Plan-and-Solve (PS, Wang et al., 2023), que se traduce como planificar y resolver, busca mejorar el método Zero-Shot CoT, intentando obtener de los LLM una capacidad de razonamiento más efectiva, manteniendo al mismo tiempo la característica de no utilizar ejemplos dentro del prompt. Los autores de este trabajo logran este propósito experimentado con frases de activación más elaboradas, como se puede ver en la Figura 12 donde se comparan las frases de Zero-Shot CoT (a) y de Plan-and-Solve (b). Como se aprecia en la figura, el primer paso del prompt “Let’s think step by step” se cambia en “Let’s first understand the problem and devise a plan to solve the problem. Then, let’s carry out the plan and solve the problem step by step”.[1]

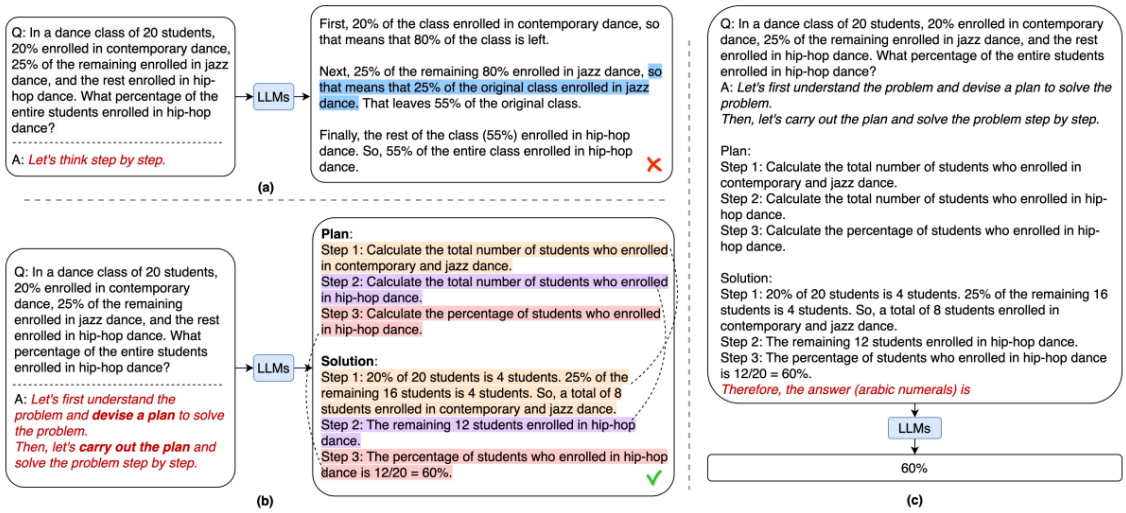

Figura 12. El método Plan-and-Solve mantiene la estructura en dos pasos de Zero-Shot CoT sustituyendo su frase en el primer paso (a) con una más elaborada que incorpora instrucciones más detalladas (b)

Los autores del método Plan-and-Solve también presentan una mejora de su propio sistema que nombran Plan-and-Solve+. En esta nueva propuesta, se ofrecen instrucciones aún más detalladas que prestan especial atención a la extracción de variables, la realización de cálculos y el uso de razonamiento lógico y de sentido común. Las frases adicionales están resaltadas en la Figura 13 (b), que se pueden traducir como “extrae variables relevantes y sus correspondientes numerales.” y “calcula variables intermedias (prestando atención a los cálculos numéricos correctos y al sentido común).” Los autores demuestran que estos métodos logran mejores resultados en marcos de evaluación relacionados con problemas matemáticos, razonamiento de sentido común y manipulación simbólica. Además, son capaces de corregir errores que surgían al analizar los resultados de Zero-Shot CoT.

[1] En español: “Primero, comprendamos el problema y elaboremos un plan para resolverlo. Luego, llevemos a cabo el plan y resolvamos el problema paso a paso.”)

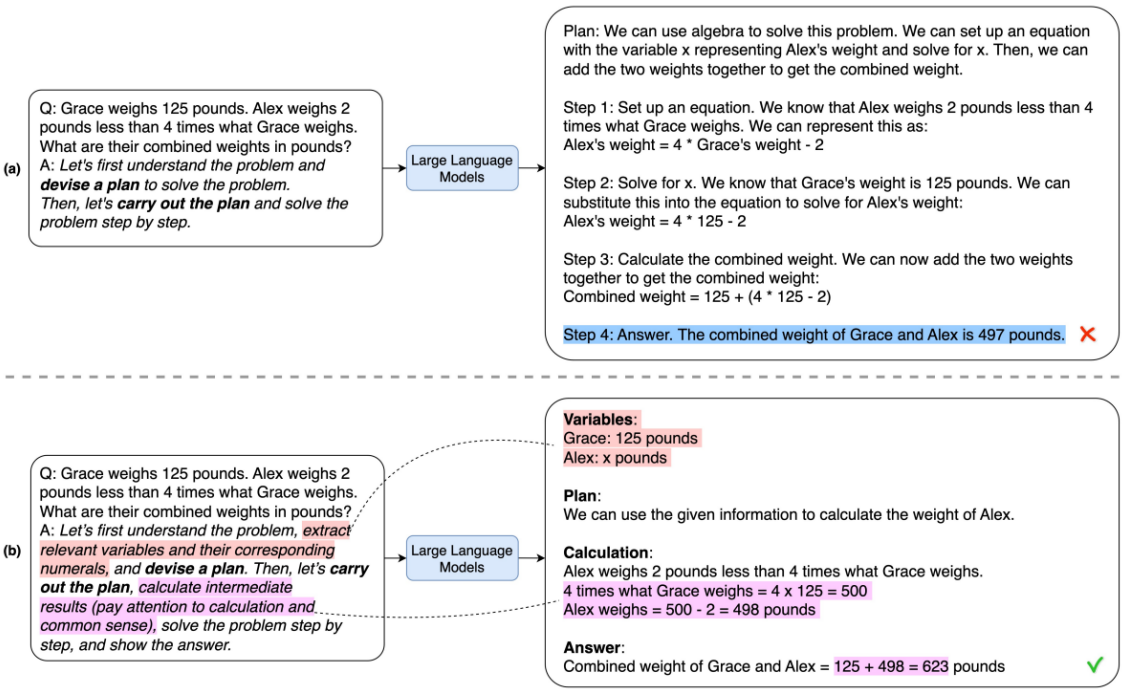

Figura 13. Comparación entre Plan-and-Solve (a) y Plan-and-Solve+ (b) en la cual se evidencian las frases adicionales que se añaden al prompt para mejorar el proceso de razonamiento.

3.3.3 Problem Solving Methods

Considerando las capacidades emergentes de los LLM en el ámbito del razonamiento y las posibilidades de inducir y mejorar este comportamiento, son de especial relevancia trabajos que se remontan a los inicios de la inteligencia artificial y la ciencia cognitiva. En particular, nos referimos a los llamados métodos de resolución de problemas (del inglés Problem Solving Methods o PSM), donde este proceso se caracteriza como una tarea de búsqueda a través de un espacio de problemas combinatorios, representado como un árbol (Newell, 1959 y 1972). A lo largo de los años, esta base conceptual ha servido como fundamento para proponer diversas soluciones formales destinadas a automatizar la resolución de problemas (Benjamins et al., 1998), como la metodología CommonKADS (Schreiber et al., 2000) desarrollada específicamente para generar estrategias y recursos que capacitan a los sistemas basados en conocimiento para la resolución de problemas.

En este contexto, surgen trabajos más recientes que buscan replicar estas metodologías al inducir a los LLM a seguir pasos previamente formalizados en metodologías de resolución de problemas mediante la técnica de prompting. El método “Tree of Thoughts” o ToT (Yao et al., 2023), ilustrado en Figura 14, dota a un LLM de la capacidad de generar diversas líneas de razonamiento ("pensamientos"), lo que facilita realizar una toma de decisiones deliberada al considerar y autoevaluar las opciones disponibles para determinar el próximo curso de acción. Además, permite anticipar o retroceder según sea necesario para tomar decisiones más coherentes y globales.

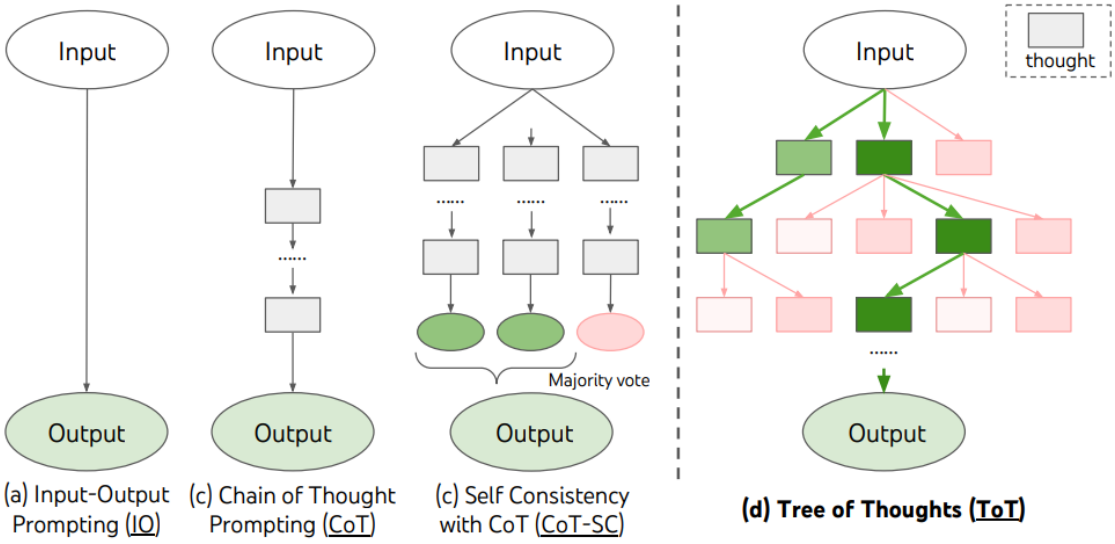

Figura 14. El método Tree of Thoughts (d), en comparación con otras estrategias de prompting, caracteriza el proceso como una exploración a través de un espacio de problemas combinatorios, representado como un árbol.

Los autores logran este comportamiento modelando cada problema como una búsqueda en un árbol de decisiones. En este árbol, cada nodo representa una solución parcial, acompañada del texto inicial y la secuencia de pensamientos generados hasta ese punto, denotada por ***s = [x, z_1…i]***, donde **s** es el estado, **x** es el texto inicial y **z** representa los pensamientos. El método comprende cuatro pasos principales: i) cómo descomponer el proceso intermedio en pasos de pensamiento, ii) cómo generar pensamientos potenciales a partir de cada estado, iii) cómo evaluar heurísticamente los estados, y iv) qué algoritmo de búsqueda utilizar. El ToT aprovecha las propiedades del problema para diseñar y descomponer los pasos intermedios del pensamiento, además de generar nuevos pasos utilizando dos tipos de indicaciones alternativas: "Sample" o "Propose". Posteriormente, el modelo evalúa cada estado alternativamente mediante las estrategias "Value" o "Vote". Finalmente, emplea dos algoritmos de búsqueda de rutas de pensamiento óptimas, conocidos como "Breadth-first search (BFS)" o "Depth-first search (DFS)". El método ToT mejora significativamente las habilidades de resolución de problemas de los LLMs en tres tareas que requieren planificación: "Game of 24", "Creative Writing" y "Mini Crosswords". En la Figura 15 se muestra la aplicación del método para resolver la tarea "Game of 24", donde el modelo logra un aumento de hasta un 70% en la tasa de éxito en comparación con los métodos IO y CoT.

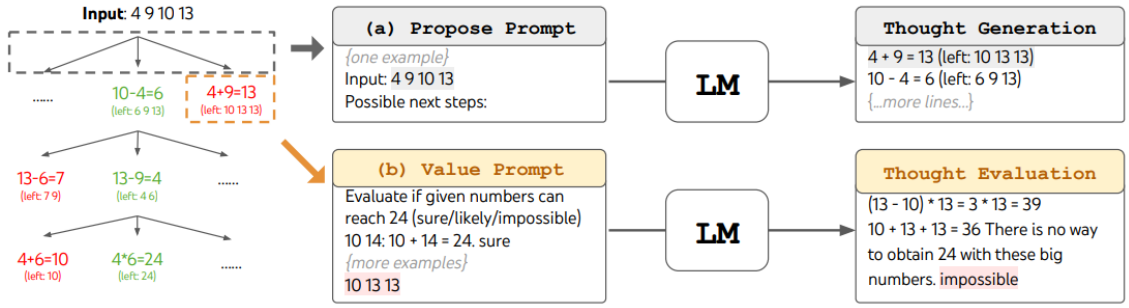

Figura 15. El método de prompting ToT para resolver la tarea "Game of 24" involucra el uso de las estrategias "Propose" (a) y "Value" (b).

3.3.4 Métodos de enriquecimiento del prompt en tiempo de inferencia

Trabajos como (Izakard et al., 2022) han demostrado que la incorporación de conocimiento externo a los parámetros del LLM, básicamente pasajes de texto procedentes de resultados de búsqueda inyectados en tiempo de inferencia, ayuda considerablemente a aliviar limitaciones de los LLM relacionadas con la memorización de entidades menos frecuentes o propias de dominios específicos que puede provocar problemas de factualidad, alucinaciones y sesgos. Otros trabajos (Mallen et al., 2023) han demostrado que este método basado en aumentar la capacidad del LLM dotándole de información adicional de contexto durante inferencia es más efectivo y ofrece mayores ganancias que las aportadas mediante el aumento del número de parámetros entrenables del LLM. Por otro lado, Zamani et al. (2022) proponen un marco de trabajo general en el que se separa explícitamente el conocimiento contenido en los LLM de su capacidad de razonamiento, de manera que el conocimiento necesario en tiempo de inferencia puede ser aumentado mediante técnicas de recuperación de información.

Uno de los trabajos más representativos que ha introducido el uso de información externa junto con la memoria paramétrica es el enfoque de RAG (Lewis et al., 2020), el cual se centra en la generación enriquecida mediante la recuperación de información factual. Este modelo utiliza la secuencia de entrada $\boldsymbol{x}$ para recuperar documentos de texto $\boldsymbol{z}$ y los emplea como contexto adicional al generar la secuencia de salida $\boldsymbol{y}$. El modelo que se muestra en la Figura 16 introduce el uso de dos componentes: i) un recuperador $\boldsymbol{p_\eta(z|x)}$ con parámetros $\boldsymbol{\eta}$ que devuelve distribuciones sobre pasajes de texto dados una query $\boldsymbol{x}$ y ii) un generador $\boldsymbol{p_\vartheta(y_i|x, z, y_{1:i-1})}$ parametrizado por $\boldsymbol{\vartheta}$ que genera un token basado en el contexto de los tokens previos $\boldsymbol{y_{1:i-1}}$, el input original **x** y un pasaje recuperado $\boldsymbol{z}$.

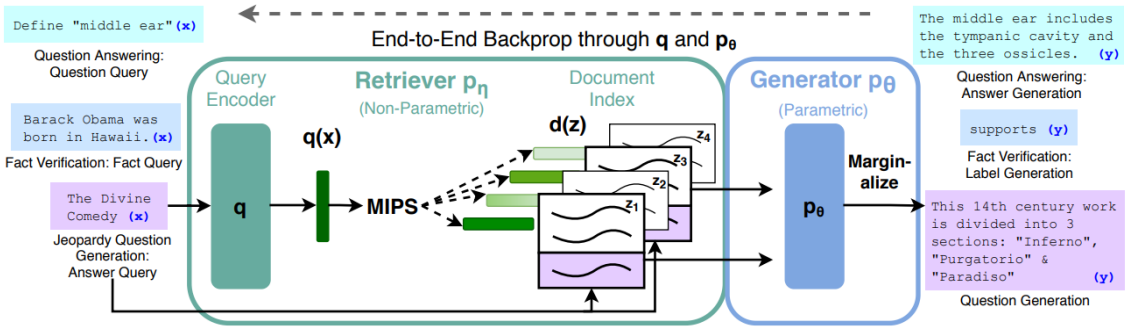

Figura 16. El método RAG introduce el uso de un recuperador para extraer información factual externa que sirve como contexto para un generador.

Este método ha mostrado ser efectivo en tareas intensivas en conocimiento, como la de preguntas y respuestas ("question answering"), pero presenta limitaciones en relación con la recuperación e incorporación indiscriminada de un número fijo de pasajes recuperados, independientemente de si la recuperación es necesaria o si los pasajes son relevantes.

Para superar estas limitaciones, se ha propuesto el método de SELF-RAG (Asai et al., 2023), que se basa en la generación aumentada con recuperación auto reflexiva. Este método capacita a cualquier modelo de lenguaje para determinar si es necesario recuperar pasajes para elaborar la respuesta o si puede hacerlo a través de sus propios parámetros. Además, el modelo se entrena para evaluar críticamente el texto generado de manera aumentada con los pasajes recuperados, determinando la relevancia y utilidad efectiva de estos, lo que le permite seleccionar la solución más factual para el texto de entrada, como puede verse en la Figura 17.

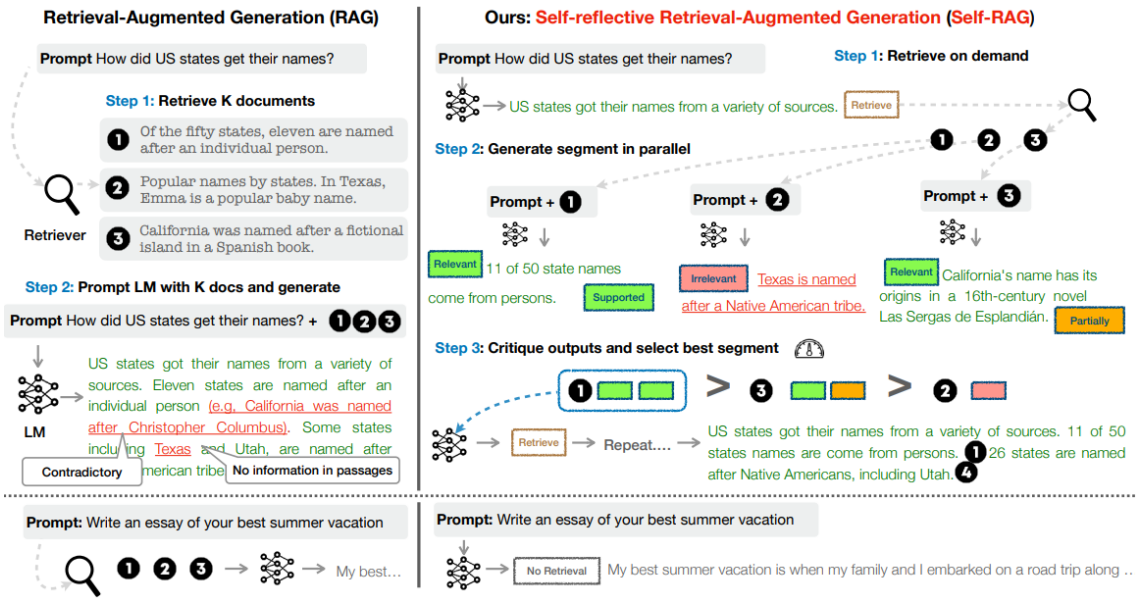

Figura 17. Comparación entre RAG y SELF-RAG. Este último método aprende a recuperar información solo cuando es necesario y a evaluar estos pasajes para integrarlos únicamente si son válidos, aumentando así su factualidad.

Los autores desarrollan un enfoque de entrenamiento "crítico" que reduce los costos y, al mismo tiempo, ofrece un mayor control sobre el proceso de inferencia en comparación con otros métodos, como el RLHF (Ouyang et al., 2022) por PPO (Schulman et al., 2017). La novedad reside en agregar la crítica directamente al corpus de entrenamiento, a diferencia de PPO, que se basa en un modelo adicional para cumplir esta función durante el proceso de entrenamiento. Para que el modelo pueda aprender este comportamiento auto reflexivo, es necesario agregar unos tokens especiales al corpus de entrenamiento que guíen su comportamiento, generando ejemplos de entrenamiento como se muestra en la Figura 18.

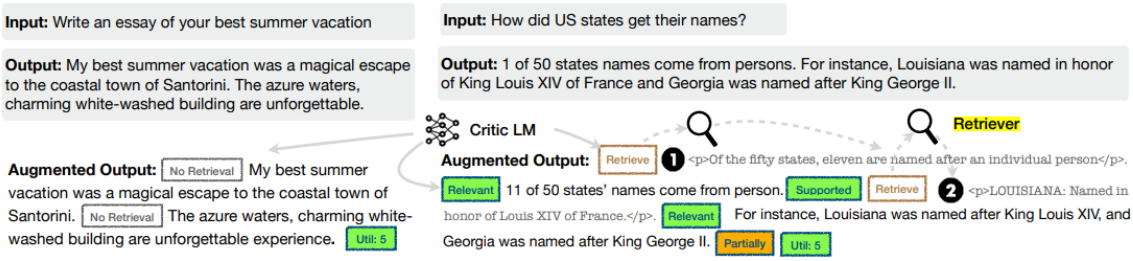

Figura 18. Ejemplos de datos de entrenamiento para el modelo generativo de SELF-RAG.

Para generar este conjunto de datos que contiene los tokens necesarios para entrenar el comportamiento del modelo, los autores utilizan en una fase previa un modelo de recuperación de pasajes, Contriever-MS MARCO (Izacard et al., 2021), y un modelo crítico, Llama2 7B. El SELF-RAG se ha demostrado como una solución eficaz para mejorar la factualidad de un modelo, con un rendimiento superior incluso a modelos más grandes como ChatGPT y otros modelos que igualmente emplean diferentes técnicas de recuperación.

Los modelos basados en recuperación de información no solo pueden depender de información textual, sino también de información estructurada, por ejemplo, como grafos de conocimiento.

Recientemente han aparecido trabajos en la línea de investigaciones previas (Logan et al., 2019) que demuestran cómo un mecanismo basado en recuperación de información mediante grafos puede mejorar la calidad de la generación de respuestas de un LLM e incluso potenciar el razonamiento sobre conceptos semánticos más complejos, que requieren una abstracción y manipulación simbólica más avanzada. Estos trabajos forman parte de la oferta de producto de compañías como Microsoft, con GraphRAG[2], y Neo4J[3]. Trabajos científicos recientes como G-Retriever (He et al., 2024) integran las fortalezas de redes de neuronas para grafos (GNN), LLM y RAG para mejorar la comprensión de grafos mediante soft prompting. Los resultados de este método parecen indicar una mejora del LLM en cuanto a su resistencia a las alucinaciones.

3.4 Métodos para el uso de herramientas externas por parte de LLM

Los LLM exhiben habilidades importantes para resolver nuevas tareas a partir de solo unos ejemplos o instrucciones de texto. Sin embargo, estos modelos también tienen problemas con funcionalidades básicas que incluyen operaciones aritméticas o de búsqueda de hechos, donde modelos que han sido ajustados específicamente a esas tareas obtienen mejores resultados. Para mejorar las prestaciones de los LLM en estas tareas se han propuesto varios trabajos que enseñan al modelo de lenguaje cuándo es mejor y cómo usar una herramienta para realizar estas tareas especializadas en lugar de apoyarse en su memoria paramétrica.

En esta sección se presentan LLM aumentados con herramientas incluyendo PAL (Gao et al., 2022) que usa **aprendizaje en contexto** para que el LLM genere intercaladamente texto en lenguaje natural y líneas de código para resolver un problema, TALM (Parisi et al., 2022) que aprende simultáneamente a llamar herramientas y a producir respuestas usando el resultado de las herramientas, Toolformer (Schick et al., 2023) que aprende a decidir qué API usar, cuándo usarlas, qué argumentos proporcionar y cómo integrar los resultados en la predicción de tokens futuros, y Chain-of-Abstraction CoA (Gao et al., 2024) que aprende a decodificar cadenas de razonamiento con marcadores de posición abstractos y después llamar herramientas de dominio para reificar cada cadena de razonamiento completando conocimiento específico. Además, estos enfoques usan herramientas como calculadoras y solucionadores de sistemas de ecuaciones (TALM, Toolformer, CoA), sistemas de respuestas a preguntas (TALM, Toolformer, CoA), motores de búsqueda (Toolformer), traducción automática (Toolformer) y runtimes de programación (PAL). En cuanto a LLM, PAL usa openAI codex, mientras que TALM, Toolformer y Chain-of-Abstractions afinan T5, GPT-J, y LLaMA(-2).

3.4.1 PAL - Program-aided language models

El método Program-Aided Language models PAL[4] (Gao et al., 2022) utiliza un modelo de lenguaje grande (LLM) para comprender problemas de lenguaje natural y generar pasos intermedios del programa, y la solución final se ejecuta en un *runtime* de programación como un intérprete de Python. Este enfoque divide el proceso de resolución de problemas. El LLM se centra únicamente en descomponer el problema en pasos ejecutables, dejando la solución al intérprete.

---

[2] https://www.microsoft.com/en-us/research/blog/graphrag-unlocking-llm-discovery-on-narrative-private-data

[3] https://neo4j.com/developer-blog/knowledge-graph-devops-rag-application

[44] https://reasonwithpal.com

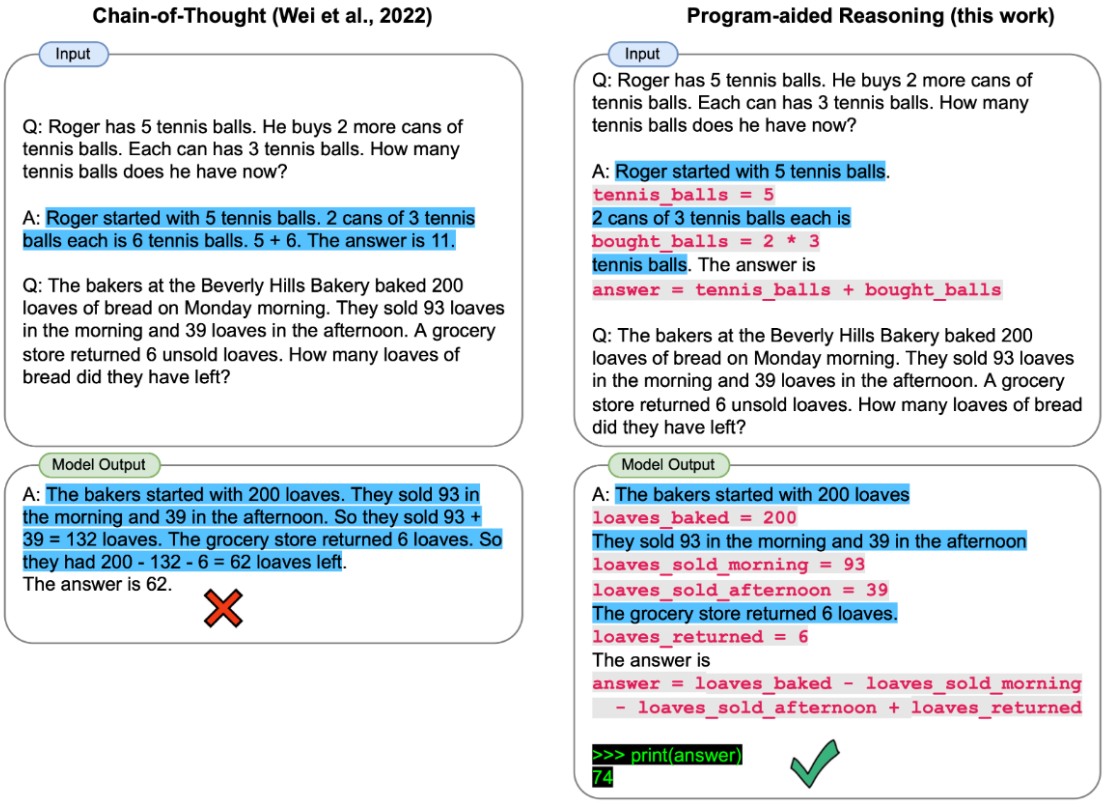

Figura 19: Dada una pregunta de razonamiento matemático, CoT (izquierda) genera pasos de razonamiento intermedios de texto de formato libre. Por el contrario, PAL genera un programa Python para resolver una tarea a partir de su descripción en lenguaje natural.

PAL demuestra la sinergia entre un LLM neural y un intérprete simbólico en varias tareas de razonamiento, incluidos problemas matemáticos, simbólicos y algorítmicos. PAL logra resultados más precisos que modelos más grandes que usan prompts tipo CoT, entre los que se incluye PALM-540B, en 12 benchmarks, incluyendo problemas matemáticos, simbólicos y algorítmicos, estableciendo nuevos resultados del estado del arte.

En PAL se propone generar los pensamientos t para un problema dado en lenguaje natural x como declaraciones intercaladas en lenguaje natural (LN) y lenguaje de programación (LP). Dado que se delega el paso de solución a un intérprete, no se proporcionan las respuestas finales a los ejemplos en el prompt. Es decir, cada ejemplo en contexto en PAL es un par $<x_i, t_i>$, donde $t_j = [s_1, s_2, \ldots s_N]$ con cada $s_i \in LN \cup LP$, una secuencia de tokens en LN o LP. El prompt completo es $p \equiv <x_1 \cdot t_1> || <x_2 \cdot t_2> || \ldots || <x_k \cdot t_k>$.

Dada una instancia de prueba x_{test}, se adjunta al prompt, y $p || x_{test}$ se alimenta al LM. Se permite que el LM genere una predicción t_{test}, que contiene tanto los pasos intermedios como sus declaraciones programáticas correspondientes.

Se indica al LLM que genere pasos intermedios en lenguaje natural usando sintaxis de comentarios (e.g., "#.." en Python) de manera que se ignoren por el intérprete. En los experimentos se usa **Codex**[5] como LLM.

[5] https://openai.com/blog/openai-codex

A continuación, en la Figura 20 se presenta un ejemplo de prompt generado por PAL para un problema matemático, en la Figura 21 se presenta un ejemplo de prompt para la tarea de coloreado de objetos y en la Figura 22 un ejemplo de prompt para la tarea de contar objetos.

Q: Olivia has $23. She bought five bagels for $3 each. How much money does she have left?

```
money_initial = 23
bagels = 5
bagel_cost = 3
money_spent = bagels * bagel_cost
money_left = money_initial - money_spent
answer = money_left
```

Figura 20. Ejemplo de prompt para una tarea matemática

Q: On the table, you see a bunch of objects arranged in a row: a purple paperclip, a pink stress ball, a brown keychain, a green scrunchiephone charger, a mauve fidget spinner, and a burgundy pen. What is the color of the object directly to the right of the stress ball?

```
...
stress_ball_idx = None
for i, object in enumerate(objects):
    if object[0] == 'stress ball':
        stress_ball_idx = i
        break
# Find the directly right object
direct_right = objects[stress_ball_idx+1]
# Check the directly right object's color
answer = direct_right[1]
```

Figura 21. Ejemplo de prompt para una tarea de coloreado de objetos.

Q: I have a chair, two potatoes, a cauliflower, a lettuce head, two tables, a cabbage, two onions, and three fridges. How many vegetables do I have?

```
# note: I'm not counting the chair, tables,
    or fridges
vegetables_to_count = {
    'potato': 2,
    'cauliflower': 1,
    'lettuce head': 1,
    'cabbage': 1,
    'onion': 2
}
answer = sum(vegetables_to_count.values())
```

Figura 22. Ejemplo de prompt en la tarea de contar objetos.

3.4.2 TALM - Tool-augmented language models

Los Modelos de Lenguaje Aumentados con Herramientas (Tool Augmented Language Models TALM) (Parisi et al., 2022) combinan un enfoque de solo texto para enriquecer los modelos de lenguaje con herramientas no diferenciables, junto con una técnica iterativa de "autojuego" para mejorar el rendimiento a partir de pocas demostraciones de herramientas. TALM muestra un rendimiento sólido tanto en una tarea de preguntas y respuestas (QA) basada en conocimiento como en una tarea de matemáticas orientada al razonamiento con herramientas simples. A una determinada escala, TALM supera significativamente a los LLM no aumentados.

Además, se demuestra que TALM realiza con éxito inferencias fuera de distribución en tareas de QA y matemáticas, donde los LLM no aumentados fallan. Los resultados sugieren que los LLM aumentados con herramientas representan una dirección prometedora para enriquecer las capacidades de los modelos de lenguaje, con menos dependencia de la escala. TALM permite que los modelos invoquen herramientas arbitrarias con resultados generados por el modelo y presten atención a los resultados de las herramientas para generar resultados de tareas.

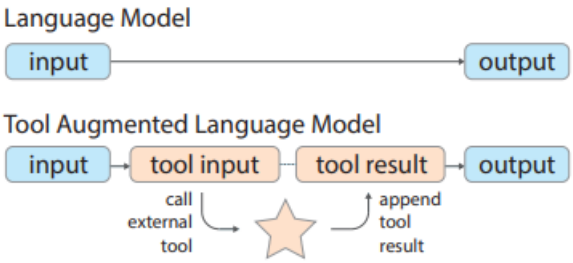

Figura 23. Modelo del Lenguaje aumentado con herramientas.

Se usa **T5** en sus versiones base, large y XL. Además, se usa una interfaz texto a texto como se muestra en la siguiente figura. TALM aprende dos tareas al mismo tiempo: llamar una herramienta y generar la respuesta basado en los resultados de la herramienta.

An abstract task:
task input text |*tool-call* tool input text |*result* tool output text |*output* task output text

A weather task:
how hot will it get in NYC today? |*weather* lookup region=NYC |*result* precipitation chance: 10, high temp: 20c, low-temp: 12c |*output* today's high will be 20C

Figura 24. Ejemplo de la interface texto a texto de TALM

Autojuego se refiere a un modelo interactuando con el API de una herramienta. Se usa el autojuego para hacer bootstraping interactivo de los ejemplos de uso de herramientas con una calidad mejor progresivamente. El proceso iterativo de autojuego comienza con un pequeño conjunto de arranque de uso de herramientas $\{x_j, t_j, r_j, y_j\}_D$. En cada ronda de autojuego, TALM se afina en el conjunto de uso de herramientas D. A continuación, para cada ejemplo en el conjunto de tareas T, TALM muestrea las entradas de la herramienta, llama el API de una herramienta y muestrea resultados de tareas basados en resultados de la herramienta. Si el resultado de la tarea generado por TALM coincide con el objetivo dentro de algún umbral th, la secuencia de uso de la herramienta que condujo al resultado se agrega al conjunto de uso de herramientas D para la siguiente ronda de juego personal.

A continuación, se presentan en la Figura 25 y la Figura 26 ejemplos de las secuencias aumentadas con herramientas generadas por TALM.

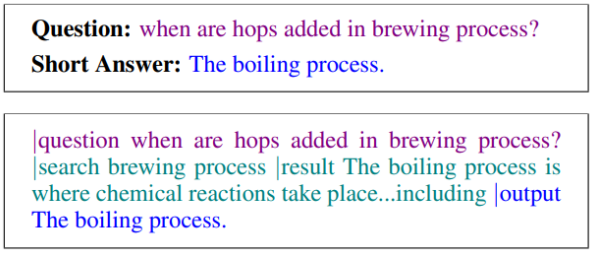

Figura 25. Ejemplo de pregunta del benchmark Natural Questions y la correspondiente secuencia aumentada con herramientas.

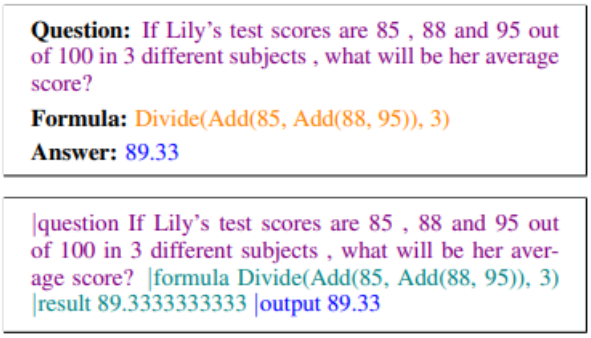

Figura 26. Ejemplo de pregunta de matemáticas y la correspondiente secuencia aumentada con herramientas.

3.4.3 Toolformer

Toolformer (Schick et al., 2023) se entrena para decidir qué API usar, cuándo usarlas, qué argumentos proporcionar y cómo integrar los resultados en la predicción de tokens futuros. Este enfoque autosupervisado requiere solo unas pocas demostraciones para cada API. Se incorporan diversas herramientas, como una calculadora, un sistema de preguntas y respuestas, un motor de búsqueda, un sistema de traducción y un calendario. Toolformer muestra un rendimiento significativamente mejorado en zero-shot en múltiples tareas, a menudo rivalizando con modelos más grandes, sin sacrificar sus capacidades centrales de modelado de lenguaje.

Toolformer usa un modelo de lenguaje grande (LLM) con aprendizaje en contexto para generar datasets completos desde cero. Dado unos pocos ejemplos escritos por humanos de como un API puede ser usado, se permite a un ML anotar un gran conjunto de datos de modelado de lenguaje con llamadas a API potenciales. Luego se utiliza una función de perdida autosupervisada para determinar cuáles de estas llamadas API realmente ayudan al modelo a predecir tokens futuros (Ver Figura 28). Finalmente, se ajusta el propio LM en las llamadas API que considera útiles. Como se ilustra en la Figura 27, a través de este enfoque los LM pueden aprender a controlar una variedad de herramientas y a elegir por sí mismos qué herramienta usar, cuándo y cómo.

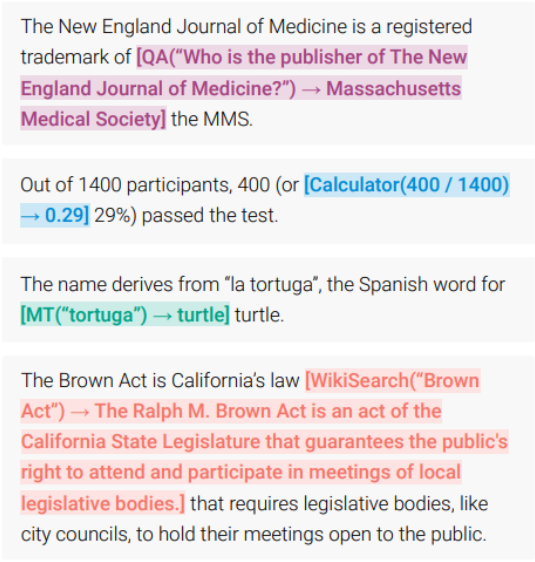

Figura 27. Predicciones de ejemplo de Toolformer. El modelo decide de forma autónoma llamar a diferentes API (de arriba a abajo: un sistema de respuesta a preguntas, una calculadora, un sistema de traducción automática y un motor de búsqueda de Wikipedia) para obtener información útil para completar un texto.

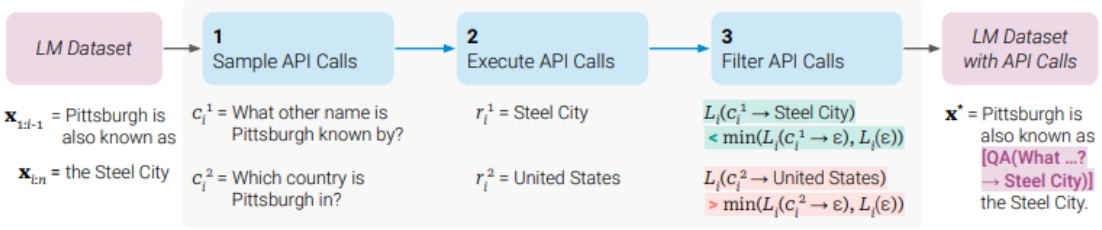

Figura 28. Pasos claves del Toolformer ilustrados para una herramienta de preguntas y respuestas. Dado un texto de entrada x, primero se muestrea una posición i y las correspondientes llamadas candidatas $c_i^1, c_i^2, \ldots, c_i^k$*. Después se ejecutan estas llamadas a las API y se eliminan las llamadas que no reducen la función de perdida* L_i *sobre los siguientes tokens. El resto de las llamadas a API se intercalan con el texto original, dando lugar a un nuevo texto x*.*

Your task is to add calls to a Question Answering API to a piece of text. The questions should help you get information required to complete the text. You can call the API by writing "[QA(question)]" where "question" is the question you want to ask. Here are some examples of API calls:

Input: Joe Biden was born in Scranton, Pennsylvania.

Output: Joe Biden was born in [QA("Where was Joe Biden born?")] Scranton, [QA("In which state is Scranton?")] Pennsylvania.

Input: Coca-Cola, or Coke, is a carbonated soft drink manufactured by the Coca-Cola Company.

Output: Coca-Cola, or [QA("What other name is Coca-Cola known by?")] Coke, is a carbonated soft drink manufactured by [QA("Who manufactures Coca-Cola?")] the Coca-Cola Company.

Input: x

Output:

Figura 29. Prompt de ejemplo P(x) para generar llamadas al API de herramienta de preguntas y respuestas.

Las herramientas que integra Toolformer son las siguientes:

- **Respuestas a preguntas:** Atlas[6] (Izacard et al., 2022) es un modelo de lenguaje aumentado con recuperación de información afinado en el benchmark de preguntas naturales.
- **Calculadora:** Se usa una calculadora que soporta las 4 operaciones básicas.
- **Búsqueda en Wikipedia:** Motor de búsqueda que se apoya en el algoritmo BM25. Dado un término de búsqueda retorna textos cortos de Wikipedia.
- **Traducción automática:** NLLB[7] (Costa-Jussa et al., 2022) Basado en un modelo de lenguaje que puede traducir 200 idiomas distintos al inglés.
- **Calendario:** Es un API que retorna la fecha actual sin tomar ninguna entrada.

3.4.4 Chain-of-Abstraction

Chain-of-Abstraction CoA (Gao et al., 2024) entrena (ver Figura 30) un LLM para que primero decodifique cadenas de razonamiento con marcadores de posición abstractos y luego llame a herramientas de dominio para reificar cada cadena de razonamiento completando conocimientos específicos. Esta planificación con cadenas abstractas permite a los LLM aprender estrategias de razonamiento más generales, que son sólidas ante los cambios de conocimiento del dominio (por ejemplo, resultados de matemáticas) relevantes para diferentes preguntas de razonamiento. También permite a los LLM realizar decodificación y llamada de herramientas externas en paralelo, lo que evita el retraso de inferencia causado por la espera de respuestas de la herramienta.

En los dominios de razonamiento matemático y de respuestas a preguntas en Wiki, CoA supera consistentemente las líneas de base anteriores de cadena de pensamiento y aumentadas por herramientas, en conjuntos de pruebas tanto dentro como fuera de la distribución, con una mejora media absoluta de la precisión de las respuestas de aproximadamente un 6 %.

Los LLM entrenados con CoA muestran un uso más eficiente de las herramientas, con una velocidad de inferencia que es en promedio aproximadamente 1,4 veces más rápida que los LLM de base aumentados con herramientas.

[6] https://research.facebook.com/blog/2023/1/atlas-few-shot-learning-with-retrieval-augmented-language-models

[7] https://ai.meta.com/research/no-language-left-behind

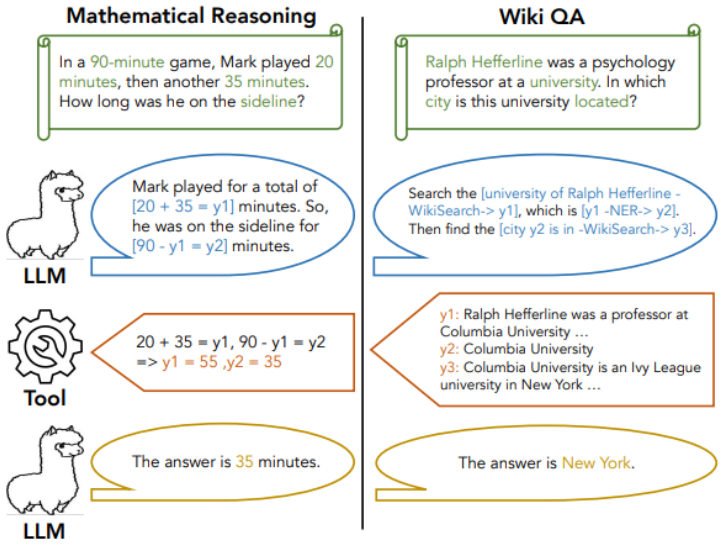

Figura 30. Vista general de razonamiento de CoA con herramientas. Dada una pregunta de dominio,un LLM se afina para primero generar una cadena de multiples pasos abstracta (burbuja azu), y despúes llama herramientas externas para reificar la cadena con conocimiento de dominio. La respuesta final obtiene con base en la cadena reificada de razonamiento.

Para crear los datos de aprendizaje de CoA primero se hace prompt a un LLM para que reescriba las respuestas existentes como instrucciones en cadenas abstractas, y después se usan las herramientas de dominio para verificar la validez de la reescritura, como se muestra en la Figura 31. El modelo de lenguaje usado para reescribir la respuesta es Llama-70B. Primero se hace prompt de Llama-70B para que etiquete secuencias en las respuestas que correspondan con operaciones de conocimiento (derivaciones matemáticas o frases basadas en referencias de Wikipedia) y después reescribir las frases con las secuencias etiquetadas como trazas de CoA rellenables, donde los resultados de las operaciones son remplazados con marcadores de posición abstractos.

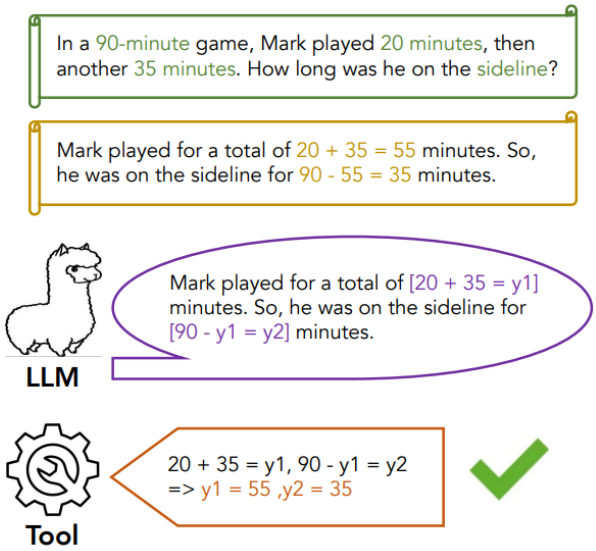

Figura 31. Ejemplo de la reescritura de datos para afinar la construcción de datos. Dado un par de preguntas de dominio (pergamino verde) y respuesta gold (pergamino amarillo), se solicita a un LLM que reescriba la respuesta dorada como una cadena de razonamiento con variables abstractas (burbuja morada). Luego,

herramientas especializadas en el dominio validan la exactitud de la reescritura verificando si la cadena abstracta se puede reificar para obtener la respuesta final (etiqueta naranja).

El proceso de ajuste de los modelos LLaMA-7B y LLaMA -70B se realizó en 8 y 64 NVIDIA A100-SXM4 (80GB) GPU, respectivamente. La herramienta usada para la búsqueda en Wikipedia es un motor de búsqueda implementado usando BM25 y para el reranking se usa la similitud de coseno basada en sentence embeddings. Además, se usa el reconocedor de entidades de Spacy[8] para extraer las entidades que puentean consultas de varios pasos. Para los cálculos matemáticos se utiliza SymPy[9], una herramienta para resolver sistemas de ecuaciones.

3.5 Métodos para la generación de grafos de conocimiento a partir de texto mediante LLM generativos

La construcción de grafos de conocimientos (KGC por sus siglas en inglés) generativa se refiere al uso de LLM generativos para generar los grafos (Ye et al., 2022). Típicamente para la tarea de KGC generativa se usan modelos secuencia a secuencia (seq2seq) como T5 o BART. La tarea de construcción de grafo puede descomponerse en diferentes subtareas incluyendo reconocimiento de entidades nombradas, extracción de relaciones, extracción de eventos, enlazado de entidades y completar grafos. Sin embargo, esta sección se centra en enfoques que realicen la tarea de construcción del grafo de manera completa (ver Figura 32). Para construir el grafo a partir de textos se puede entrenar un modelo generativo para etiquetar los textos con las entidades y las relaciones (ver Figura 33) o para construir una secuencia con una estructura determinada desde la cual se pueda derivar el grafo de conocimiento (ver Figura 34).

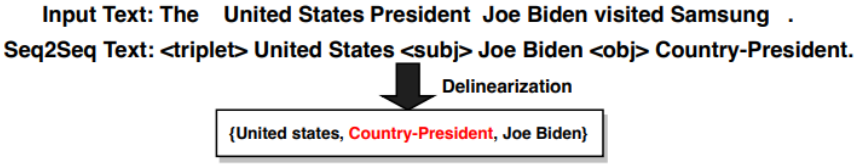

Figura 32. KGC generativo

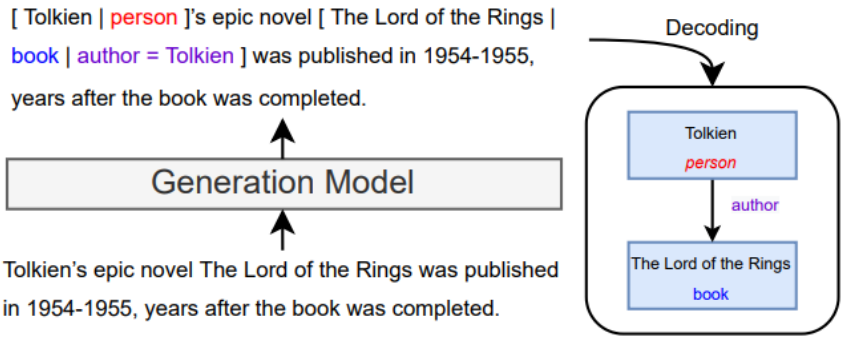

Figura 33. Secuencia aumentada con etiquetas para KBC

[8] https://spacy.io

[9] https://www.sympy.org

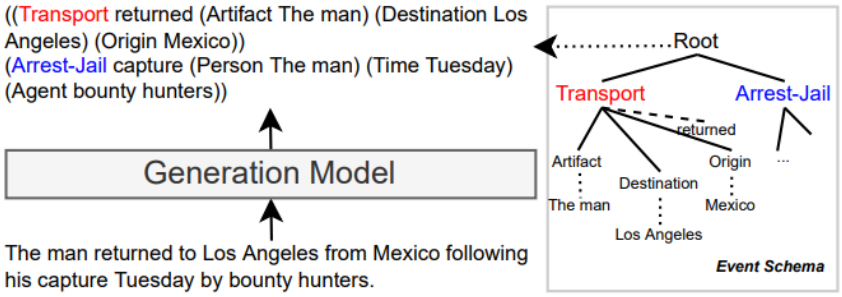

Figura 34. Secuencia con estructura linealizada

La tarea de construcción de grafos generativa cae en la categoría de enfoques texto-a-grafo y está relacionada con la tarea de Población de bases de conocimiento, KBP (Ji and Grishman, 2011) por sus siglas en ingles. En KBP el objetivo es expandir un grafo de conocimiento con hechos descubiertos a partir de textos. Tradicionalmente para hacer KBP se usa un enlazador de entidades (entity linking) en un corpus para reconocer entidades en el texto, y posteriormente se aplica rellenado de espacios (slot filling) para predecir objetos, ya sean entidades o valores, para un sujeto y una relación. Un componente clave en slot filling es un módulo de extracción de relaciones que identifica la relación entre la entidad y el objeto candidato.

En (Berquand y Ladeira, 2022) se propone usar un LLM para extraer entidades de la descripción de misiones espaciales. Es decir, se usa el LLM para hacer entity linking. Luego esas entidades son verificadas de forma automática y manual, y finalmente se integran en el grafo de conocimiento de acuerdo con una ontología determinada. En (Garcia-Silva y colegas, 2023) se hace prompt a un LLM para encontrar los objetos de parejas de sujetos y relaciones previamente determinadas. En este trabajo los autores usan un modelo de implicación textual (textual entailment) para validar las tripletas obtenidas contra textos obtenidos de la Web.

En cuanto a datos y grafos de conocimiento, se puede experimentar con el grafo presentado en (Berquand y Ladeira, 2023) centrado en el dominio de las misiones espaciales. Estas autoras crean un esquema ontológico (Ver

Figura 35) y lo intentan poblar con textos extraídos de EOportal[10]. Para evaluar el grafo obtenido se usa una base de datos donde se encuentra información estructurada de las misiones[11]. Ya que toda la información está en inglés se podría usar un modelo generativo grande (> a 70 billones de parámetros) para por ejemplo traducir los textos a otros idiomas y de esta manera ampliar la evaluación al español y lenguas cooficiales.

[10] https://www.eoportal.org/satellite-missions

[1111] https://database.eohandbook.com/about.aspx

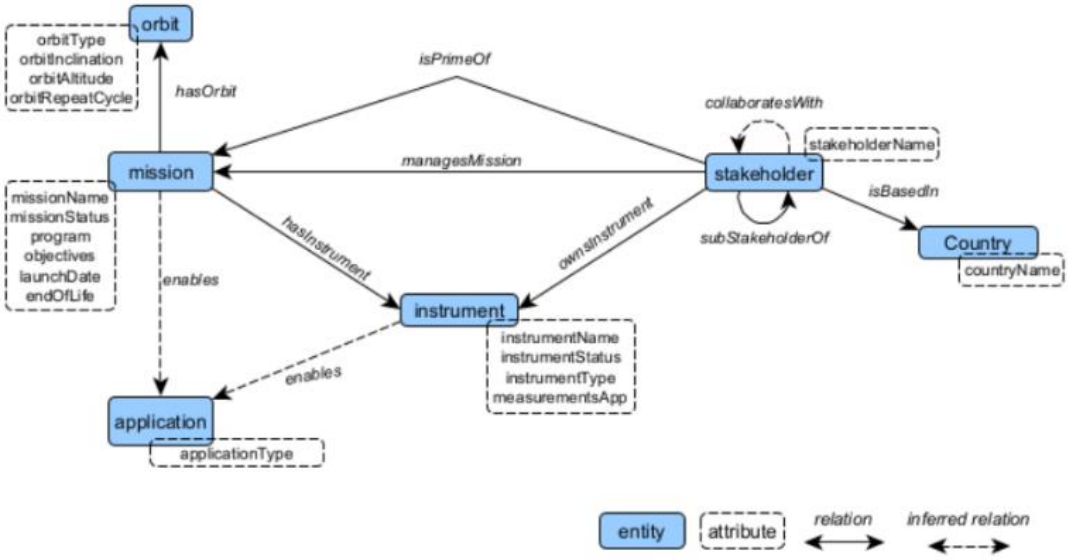

Figura 35. Esquema del grafo de conocimiento para misiones espaciales.

En (Lawrence, 2024) se explora usar un LLM para realizar todo el proceso texto a grafo, incluyendo la generación del código del grafo de conocimiento en un formato determinado. Este autor propone cuatro enfoques distintos para tareas Texto-a-Grafo usando LLM:

- Usando un LLM preentrenado con ontologías
- Incluir la ontología en el prompt del LLM
- Usando un LM afinado con la ontología
- Usando un hibrido de un LLM preentrenado y afinado

A continuación, se describe en detalle cada uno de estos enfoques.

3.5.1 LLM preentrenado con ontologías

Algunos LLM como gpt-3.5-turbo han sido pre-entrenados con una variedad de ontologías tales como SCHEMA.org, FOAF, SKOS, RDF, RDFS, and OWL. Si se usa un prompt que indique al LLM que debe usar una determinada ontología vista en preentrenamiento y se incluye el texto de donde se quiere obtener el grafo, es posible obtener el grafo requerido.

El prompt debe instruir al LLM a usar la ontología y prefijos determinados. Por ejemplo:

```
Traduce el siguiente ejemplo de texto a un grafo RDF usando las ontologías RDF, RDS, y OWL y el formato TTL y usa el prefijo ex: con IRI <http://example.com/> para cualquier entidad creada. Texto….
```

Con este prompt gpt-3.5-turbo se capaz de crear el grafo incluyendo nuevas propiedades y clases con el prefijo previsto.

Las ventajas de este enfoque son:

- El LLM usará las clases y propiedades de la ontología descrita en el prompt o creará nuevas si es necesario

- La parte fija del prompt es corta así que la mayoría del costo de procesar el prompt viene del texto fuente.

Las desventajas de este enfoque son:

- La transformación Texto-a-Grafo está limitada a las ontologías en las que ha sido pre-entrenado el LLM
- Las entidades generadas tienen que ser alineadas entre grafos.

3.5.2 Incluir la ontología en el prompt del LLM

A parte de las ontologías muy populares como las que se mencionan en el apartado anterior es poco probable que un LLM haya sido pre-entrenado con una ontología de dominio o desarrollada para una aplicación particular. En este caso es necesario incluir toda la ontología en el prompt. El prompt debe incluir las siguientes instrucciones y el código de la ontología en formato TTL:

- Traducir el texto usando la ontología incluida
- Indicar los prefijos a utilizar
- Indicar que solo se pueden usar clases y propiedades predefinidas en la ontología incluida.
- Indicar el formato de salida del grafo.

Un ejemplo de prompt es:

```
Translate the following user text to an RDF graph using the following
schema1: <http://example.com/schema/1/> ontologies formatted as TTL.
Use the prefix ex: with IRI <http://example.com/> for any created
entities.
Only use pre-defined classes and properties from the schema1:
<http://example.com/schema/1/> ontology.
Use the properties and classes in the schema1: ontology.
Include individuals, their data, and relationships.
... the full ontology in TTL format ...
```

Este tipo de prompt funciona bien para transformar el texto a un grafo siguiendo la ontología.

Las ventajas de este enfoque incluyen:

- El LLM entiende la ontología que se incluye en el prompt
- El LLM transforma el texto a un grafo siguiendo la ontología

Las desventajas incluyen:

- El prompt es muy largo ya que incluye toda la ontología. Esto implica mayor costo ya que en condiciones normales este es proporcional al número de tokens, y mayor tiempo de procesamiento.
- Las entidades generadas tienen que ser alineadas entre grafos.

3.5.3 LLM afinado con una ontología

Del primer enfoque presentado (ver 3.5.1) se puede concluir que el entrenamiento puede hacer que el LLM entienda la ontología y la pueda usar para transformar un texto. Así es razonable

pensar que hacer un afinamiento del LLM con otras ontologías. Sin embargo, para afinar un LLM es necesario crear un dataset.

En (Lawrence, 2023) el autor propone que las tripletas de una ontología se traduzcan a texto usando un prompt divido en una parte fija conocida como prompt de sistema y una parte variable concocida como prompt de usuario:

```
messages”: [
  {"role": "system", "content": "Translate the following user text to
an RDF graph using the Schema1 ontology."},
  {"role": "user", "content": "{example unstructured text}"},
  {"role": "assistant", "content": "{RDF graph using custom Schema1
ontology semantics}"}]
}
```

Para crear ejemplos de entrenamiento para el LLM el mismo autor propone hacer prompting a un LLM. La idea es crear un prompt que le indique al LLM la tarea que se quiere llevan a cabo, es decir afinar un LLM, y que se necesita textos de ejemplo y el grafo RDF que se obtiene del texto. El prompt se divide se compone del prompt del sistema y el prompt del usuario. En este caso el prompt del sistema le indica al LMM la ontología que debe usar.

```
You are a helpful assistant who will use the :Schema1 ontology that
follows.
This schema is called :Schema1 with IRI <http://example.com/schema/1/>

…. ontology in TTL ….
```

El prompt del usuario le indica al LMM que se necesita hacer fine-tuning de un LLM y que se requiere ejemplos de textos de entrada y los grafos que se obtienen de esos textos.

```
I want to fine-tune an LLM on this Schema1 ontology.
This training aims to take an example user text and the assistant
translates it into an RDF graph based on the Schema1 ontology.
Please create a comprehensive set of 50 example system, user, and
assistant messages in JSONL message conversational chat format format
for fine-tuning an LLM to translate text to Schema1 ontology RDF.
The system content should be the instruction to translate the user
text to the assistant RDF graph response using Schema1 ontology.
The user text should be examples that cover all the concepts within
the Schema1 ontology. The assistant content should be an RDF graph
using the Schema1 ontology of the translation of the user text
examples.
Use the prefix ex: with IRI <http://example.com/> for any created
entities.
Use the message format as follows:
  {"messages": [
    {"role": "system", "content": "<system_content>"},
    {"role": "user", "content": "<user_content>},
    {"role": "assistant", "content": "<assistant_content>"}
  ]}
```

Con este prompt gpt-3.5-turbo genera automáticamente ejemplos para afinar el LLM.

```
{"messages": [
  {"role": "system", "content": "
      Translate the following user text to an RDF graph using the Schema1
ontology."},
  {"role": "user", "content": "
      John Smith is a professor who graduated from a reputed college."},
  {"role": "assistant", "content": "
      @prefix ex: <http://example.com/> .
      @prefix schema1: <http://inova8.com/schema/1/> .
      ex:JohnSmith rdf:type schema1:CC ;
        schema1:dc \"John\" ;
        ….
```

Después de afinar el LLM se espera que este sea capaz de extraer grafos que sigan una ontología a partir de un texto.

Las ventajas de este enfoque son:

- Generación automática de los ejemplos de entrenamiento usando un LLM
- Se puede hacer afinamiento del LLM con una ontología determinada
- La parte fija del prompt es corta así que la mayoría del costo viene definido por los tokens del texto del cual se quiere obtener el grafo.

Las desventajas de este enfoque son:

- Hay que generar prompts para entrenar el LLM y llevar a cabo el entrenamiento.
- Hay que evaluar la calidad de los ejemplos generados para entrenamiento y la calidad de los grafos extraídos.

3.5.4 LLM preentrenado y afinado

Otra opción es que una vez afinado el LLM con una ontología se le pida al LLM que extraiga de un texto el grafo de acuerdo con una ontología que ha visto en preentrenamiento y otra con la cual fue afinado.

El prompt debe hacer referencia a la ontología vista en entrenamiento y a la ontología en la que el LLM ha sido afinado. El siguiente ejemplo se indica al LLM que use la ontología FOAF, vista en entrenamiento, y la ontología Schema1 que ha sido usada en el afinado.

```
Translate the following user text to an RDF graph using both the FOAF,
and Schema1 ontologies.
Use the prefix ex: with IRI <http://example.com/> for any created
entities.
…Text…
```

Con este prompt, el LLM devuelve el grafo creado usando las dos ontologías.

Las ventajas de este enfoque son:

- Un LLM afinado con una ontología no olvida las ontologías que ha visto en preentrenamiento
- Se puede hacer texto a grafo usando ontologías vistas en preentrenamiento y afinado.

La desventaja de este enfoque es que si hay conceptos o propiedades que se solapan entre las ontologías el LLM usará una que no necesariamente sea la esperada.

4 Marco propuesto para la evaluación de factualidad y sesgo en LLM agnóstico de dominio e idioma

4.1 Evaluación de factualidad

Proponemos un marco general para la evaluación de factualidad que extiende los métodos descritos en la sección 3.1. El marco propuesto se resume en la *Figura 36*. En la figura se describen cuatro fases fundamentales, cada una de las cuales evaluarán la factualidad del modelo en un ámbito diferente y complementario.

Figura 36: Marco propuesto para la evaluación de factualidad de LLM

A continuación, desgranamos esas fases.

1. **Evaluación de base.** Se propone utilizar conjuntos de datos de evaluación previamente disponibles en el estado del arte para evaluar la veracidad del LLM, como TruthfulQA (Lin & Hilton et al., 2022) y FACTOR (Muhlgay et al., 2023), descritos en la sección 3.1.1. Dichos conjuntos de datos son de propósito general y normalmente sólo están disponibles en inglés, pero la evaluación sobre ellos puede aportar una perspectiva inicial interesante sobre la factualidad del modelo en el escenario más favorable posible (monolingüe y de propósito general). Esta evaluación también permitirá medir el impacto de los métodos de inyección de conocimiento y adaptación a domino a nivel general. Idealmente, dicho impacto debería ser positivo o al menos no perjudicar la factualidad del modelo en los escenarios evaluados por estos datasets. De manera similar, la evaluación de base incluirá marcos de evaluación generales para LLM más allá de aspectos relacionados con factualidad o sesgo, como MMLU (Hendrycks et. al, 2021), con los que se monitorizará el impacto de los métodos desarrollados en el proyecto en las capacidades básicas de los LLM.
2. **Evaluación basada en la confianza del LLM, independiente de dominio e idioma.** La siguiente fase es completamente independiente de dominio e idioma y se basa en el método descrito en la sección 3.1.3. En ella explotamos la correlación demostrada en (Kadavath et al., 2022; Tian et al., 2023a) entre la confianza de un LLM en la respuesta generada y la probabilidad de que dicha respuesta sea factual.
3. **Evaluación basada en referencias, específica de domino e idioma.** Como vimos en la sección 3.2.3, el método de FacTune para el alineamiento de un LLM con preferencias de factualidad mediante DPO depende de la calidad del dataset de preferencias generado mediante diferentes estimadores. Estos estimadores pueden basarse en la confianza del modelo (FacTune-MC) o en referencias (FacTune-FS). De ellos, el método basado en referencias, basado por ejemplo en FactScore, es el que consigue mejores

resultados. Esta fase de evaluación pretende ofrecer un marco basado en referencias usando las distintas versiones de Wikipedia para cada idioma como base de conocimiento general y reutilizable para los distintos dominios e idiomas objetivo.

4. **Evaluación basada en referencias sobre conjuntos de datos de verificación específicos de dominio e idioma.** De manera similar al punto anterior, el propósito de este paso de evaluación es aprovechar los conjuntos de datos de verificación disponibles para ciertos dominios, como se vio en E4.1. Para ello, se plantea una adaptación del método FactScore que sustituya conjuntos de datos basados en Wikipedia por esos conjuntos de datos de verificación.

4.2 Evaluación de sesgo

Teniendo en cuenta las métricas presentadas en la sección 3.1.4 y la arquitectura de los modelos que se utilizaran en el contexto de este proyecto (modelos Transformers con arquitectura decoder), en esta parte proponemos las las métricas elegidas más eficientes para evaluar los posibles sesgos. En la selección de métricas nos hemos centrado en las que se pueden reproducir fácilmente en un entorno multilingüe:

1. **Discovery of Correlations (DisCo).** Esta métrica estudia el posible sesgo del modelo al completar frases mediante plantillas con dos espacios en blanco. El primero se llena manualmente con un posible desencadenante de sesgo asociado, por ejemplo, a un grupo social, que proviene de listas de palabras bien definidas, mientras que el segundo se completa con las tres mejores predicciones generadas por el modelo. Por ejemplo: "[X] es [MÁSCARA]"; "[X] le gusta [MÁSCARA]". Este tipo de datos es sencillo de generar si se tiene acceso a listas de palabras relacionadas con el sesgo que se quiere evaluar. Por ejemplo, Touileb et al. (2022), evalúa sesgos en modelos de lenguaje para el idioma noruego, relacionados con palabras de trabajo y pronombres de género.
2. **CrowS-Pairs Score.** Compara pares de frases, una estereotipada y otra menos estereotipada, aproximando la probabilidad de los tokens compartidos U. Estas probabilidades están condicionadas a tokens modificados M, que suelen representar atributos de las dimensiones en las que se quieren evaluar sesgos y pueden representarse mediante la función $P(U|M, \vartheta)$. El dataset más utilizado para calcular esta métrica es CrowS-Pairs (Nangia et al., 2020), un conjunto de datos pequeño diseñado específicamente para este propósito y que ya ha sido traducido manualmente al francés (Névéol et al., 2022). Una opción sería traducirlo automáticamente a los idiomas de este proyecto o crear un conjunto de datos similar utilizando un LLM como ChatGPT.
3. **Co-Occurrence Bias Score.** Es una métrica de distribución que compara las asociaciones entre palabras neutras y términos demográficos en textos generados por LLMs. Un modelo imparcial debería tener una distribución de coocurrencias que coincida con una distribución de referencia, como la distribución uniforme. Por lo general, los conjuntos de datos utilizados para calcular estas métricas son frases relacionadas con las dimensiones demográficas que se quieren evaluar. Luego, se truncan y se pasan como prompts a un modelo generativo para que las complete. BOLD (Dhamala et al. 2022) es un ejemplo de conjunto de datos recopilado mediante la extracción de páginas de Wikipedia en inglés que mencionan un grupo en el dominio de sesgo (por ejemplo, profesión), donde se truncan las frases para formar prompts.

4. **Toxic Fraction.** Para evaluar la toxicidad de un texto generado utilizaremos una de las métricas más utilizada en literatura y que se basa en el uso de un modelo externo proporcionados por Perspe[12]. Este modelo evalúa una generación de texto y produce una probabilidad de toxicidad. La Fracción Tóxica se puede definir como la proporción de generaciones pasadas por la API que han sido anotadas como tóxicas.
5. **HONEST.** Por último, utilizaremos una métrica basada en léxico, como HONEST (Nozza et al., 2021), que realiza un análisis a nivel de palabra de la salida generada. Esto se logra comparando cada palabra con una lista precompilada de palabras dañinas (en el caso de HONEST se utiliza HurtLex) o asignando a cada palabra una puntuación de sesgo. Para calcular esta métrica, solo se necesita una lista de palabras relacionada con una dimensión demográfica específica.

5 Introducción y guías de los métodos propuestos para la inyección de conocimiento en LLM

5.1 Métodos de mejora de factualidad mediante inyección de conocimiento en LLM basados en DPO

Nuestro objetivo es aumentar el porcentaje de hechos correctos en el texto generado por el modelo para los pares de dominio e idioma objetivo y disminuir la proporción de hechos incorrectos. Nuestra propuesta se basa en el algoritmo de DPO para alinear un LLM dado con un objetivo de factualidad, apoyándonos en estimadores basados en confianza y referencias (cuando sea posible) para generar un dataset de preferencias de manera agnóstica del dominio y del idioma. En líneas generales, el método que nos proponemos explorar se basa en el algoritmo FacTune (Tian et al., 2023b) e incluye los siguientes pasos.

En primer lugar, se muestrea un número prefijado *n* de respuestas candidatas para cada prompt enviado al modelo a evaluar. Dichos prompts se construyen sobre un conjunto de datos de preguntas en el lenguaje objetivo acerca del fragmento de conocimiento de dominio que perseguimos inyectar en el LLM.

Las preguntas son generadas mediante un modelo multilingüe potente, como por ejemplo GPT-3.5, que llamaremos maestro. El prompt utilizado para muestrear las preguntas a partir del modelo maestro se construye sobre información acerca de las entidades extraídas de un recurso (semi)estructurado, como un diccionario, tesauro, taxonomía o, en el mejor caso, un grafo de conocimiento para ese idioma y dominio. Esa información también puede incluir relaciones entre entidades de dominio, siempre que dichas relaciones estén representadas en el recurso de partida.

A continuación, se utiliza otro LLM multilingüe para generar un párrafo con la respuesta a cada pregunta. Para cada respuesta, se calcula una puntuación de veracidad con el estimador elegido. Utilizamos una vez más el modelo maestro para extraer hechos atómicos de los párrafos de respuesta y transformarlos en preguntas que usamos para muestrear *m* respuestas del LLM a evaluar. Nos quedamos con la respuesta con una frecuencia mayor en el caso de usar un estimador basado en confianza o comparamos las respuestas con evidencias extraídas del

[12] https://perspectiveapi.com

conjunto de datos de verificación en el caso de usar un estimador basado en referencias. Finalmente, agregamos las puntuaciones de cada hecho atómico en una sola puntuación.

A continuación, construimos el dataset de preferencias. Para todos los pares de preguntas y respuestas, elegimos como preferida para cada pregunta la respuesta con la puntuación de veracidad más alta. Finalmente, ajustamos el LLM siguiendo el algoritmo DPO, utilizando todas las respuestas del modelo como objetivos para la etapa de ajuste supervisado.

5.2 Métodos de mejora de factualidad y sesgo en LLM mediante PSP y RAG

Los métodos de prompting han demostrado ser eficaces para mejorar el texto generado y la factualidad de la información paramétrica recuperada por el LLM. Una de las tipologías más prometedoras de este tipo de métodos se basa en la reutilización de técnicas de resolución de problemas, como el método Tree of Thoughts (ToT) presentado por Yao et al., (2023). ToT se inspira en algoritmos pioneros del campo de la IA para diseñar los prompts, centrándose especialmente en el proceso de descomposición de tareas complejas en subtareas más manejables. A través de los prompts así diseñados buscan inducir al modelo a replicar estos pasos para abordar el problema en su totalidad.

Proponemos avanzar con esta investigación y ampliar esta metodología en estudios posteriores que hayan analizado estas técnicas de manera más profunda y estructurada, ya que han sido utilizadas para desarrollar recursos y plantillas de resolución de problemas complejos para sistemas basados en reglas (Schreiber et al., 2000). Consideramos el uso de plantillas de modelos basados en conocimiento, como las empleadas para tareas de diagnóstico o evaluación (Schreiber et al., 2000), para guiar el razonamiento del LLM de manera estructurada. De esta manera, diseñaremos prompts que proporcionen al modelo estrategias para descomponer las tareas, buscar información paramétrica (y no paramétrica si es necesario) y razonar con ella, teniendo en cuenta la plantilla de resolución más apropiada para el problema en cuestión. Acuñamos así el término problem-solving prompting (PSP) para referirnos a este trabajo.

Para gestionar la búsqueda de información adicional, además de la paramétrica cuando sea necesario, consideramos seguir el enfoque presentado en SELF-RAG (Asai et al., 2023), que entrena un modelo para determinar cuándo es necesario realizar una búsqueda y evaluar los recursos recuperados de forma automática. Además, proponemos gestionar la información a recuperar teniendo en cuenta los recursos disponibles y su estructura específica, como se describe en la sección 4.1. El primer paso sería recuperar información de documentos del dominio, específicamente para el idioma de referencia, los cuales pueden incluir datos no estructurados o estructurados, como grafos de conocimiento y ontologías. Si esta información no está disponible, se recurrirá a fuentes en internet como Wikipedia y motores de búsqueda.

Estas estrategias se proponen para mejorar la factualidad del modelo, lo que a su vez puede llevar a una reducción de posibles sesgos. Sin embargo, si esto no fuera suficiente, nos planteamos mejorar las salidas generadas por los modelos y mitigar los sesgos mediante técnicas de intra-procesamiento, como la modificación de estrategias de decodificación de los modelos, y técnicas de post-procesamiento, como la reescritura del resultado con la sustitución de palabras discriminatorias (Gallegos et al., 2023).

5.3 Métodos de alineamiento con preferencias basado en DPO para el uso de herramientas externas

En esta tarea se va a entrenar un LLM para que use, cuando sea necesario, una herramienta de búsqueda construida sobre Wikipedia en español y las lenguas cooficiales. Aunque altamente relacionada con la tarea anterior de RAG, el enfoque que se va a usar en esta tarea inspirado en Toolformer integra la llamada a la herramienta como parte del texto generado por el LLM.

Sin embargo, el enfoque que se va a seguir es diferente al de Toolformer, que utiliza como objetivo de aprendizaje la predicción del siguiente token. En este caso, se usará DPO para que el LLM aprenda a preferir generar texto con llamadas a la herramienta búsqueda cuando se necesario y viceversa.

El conjunto de datos que usa DPO consta de ejemplos que contienen una pareja de textos, uno con la versión preferida y otro con la versión no deseada. Para crear este conjunto de datos valoraremos las siguientes aproximaciones: i) usar un LLM grande y aprendizaje en contexto para instruir con solo unos ejemplos de cuándo y cómo llamar la herramienta de búsqueda para generar ejemplos de texto anotado con llamadas a la herramienta, o ii) seleccionar un corpus y calcular la confianza de las predicciones del LLM usando la estrategia de FacTune, sí la confianza es menor a un umbral entonces se etiqueta el texto para llamar la herramienta. Después se usa una función de perdida autosupervisada para determinar qué llamadas al API son realmente útiles para predecir los tokens futuros. Con el dataset depurado de llamadas a la API de búsqueda que no son útiles se realiza el entrenamiento DPO.

En tiempo de inferencia se realiza la decodificación de la salida del LLM hasta que se encuentra el símbolo que indica que se espera la salida de la llamada al sistema de búsqueda. En ese momento se para la decodificación y se realiza la llamada. Después se continúa con la decodificación integrando la salida de la llamada a la secuencia de entrada del modelo.

La experimentación permitirá entender si con un enfoque de aprendizaje de preferencias es posible aprender a usar herramientas externas. Además, se pretende entender que numero de ejemplos es necesario para que el LLM aprenda a usar las herramientas y si usar DPO tiene ventajas sobre el enfoque de Toolformer, por ejemplo, requiriendo un menor número de ejemplos necesario para entrenar el sistema.

5.4 Métodos para la generación de grafos de conocimiento semánticos a partir de texto mediante LLM

Para la tarea de construcción de grafos de conocimiento (KBC) a partir de textos usando modelos generativos se va a usar un enfoque de secuencia de estructura linealizada (Ye et al., 2022). Este paradigma se refiere a usar conocimiento estructural y la semántica de las etiquetas para generar un formato de salida unificado (ver ejemplo en Figura 37). El formato de salida puede ser una representación intermedia a partir de la cual se derive el grafo de conocimiento o incluso la representación final del grafo de conocimiento en un formato determinado.

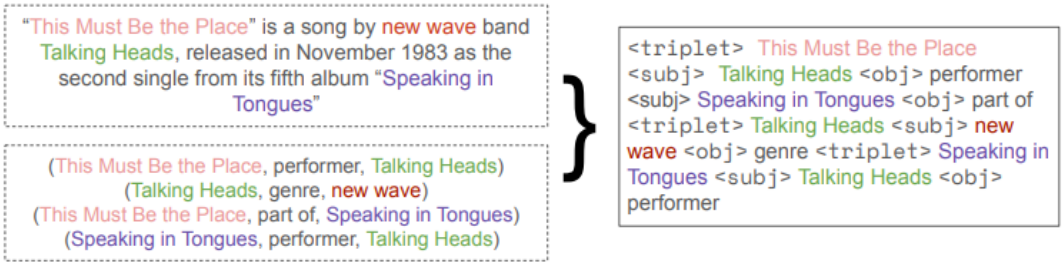

Figura 37. Ejemplo de linealización de tripletas en REBEL.

Se realizarán experimentos en una configuración zero-shot con modelos de lenguaje generativos preentrenados de diferentes tamaños para investigar si estos modelos son útiles para la tarea texto-a-grafo y si hay una relación en el desempeño de los modelos en esta tarea y el número de parámetros del modelo. Además, se realizarán experimentos de aprendizaje de contexto donde el prompt del modelo de lenguaje incluya algunos ejemplos de la tarea de texto-a-grafo. De forma similar, se afinarán los modelos generativos con datos de entrenamiento. Esto permitirá ver si hay una ganancia del desempeño del modelo con respecto a usar el aprendizaje en contexto.

6 Selección de LLM base, grafos de conocimiento, corpora y herramientas externas

6.1 LLM base

A la hora de seleccionar los LLM base que utilizaremos en el proyecto, debemos tener en cuenta diferentes factores, como el tamaño, la disponibilidad (tanto de código como de fuentes) y en qué dominios y lenguajes ha sido preentrenado. Por las características del proyecto, debemos centrarnos en aquellos que contengan soporte multilingüe en alguna escala, por lo que hemos omitido aquellos que han sido entrenados únicamente con corpora en inglés.

- **LLaMA (Touvron et al., 2023).** Con un rango de entre 7 billones y 65 billones de parámetros, esta familia de LLMs ha sido entrenada utilizando conjuntos de datos disponibles de forma pública. La potencia de estos modelos reside en el tamaño de los corpus con los que ha sido preentrenado, que puede llegar a tener un mayor peso que su número de parámetros (Hoffmann et al., 2022). Si bien todos los modelos LLaMA han sido entrenados con datos en múltiples idiomas, son los más grandes los que cuentan con un mayor soporte para lenguas con menor cantidad de recursos disponibles como catalán, euskera o gallego.
- **Falcon (Almazrouei et al., 2023).** Esta familia de LLMs cuenta con tres tamaños: 7 billones, 40 billones y 180 billones de parámetros. Han sido entrenados con un corpora de 3.5 trillones de tokens recolectados de la web. De ellos, un subconjunto de 800 billones de tokens se publicó junto a los modelos y es de acceso libre.
- **Aguila-7B[13].** Usando Falcon-7B (7 billones de parámetros) como base, se ha generado un modelo ajustado con conjuntos de datos en catalán, inglés y español, añadiendo un total de 26 billones de tokens.

[13] https://huggingface.co/projecte-aina/aguila-7b

- **FLOR[14].** Se trata de una familia de modelos de entre 760 millones y 6.3 billones de parámetros, orientados a dar soporte para catalán y castellano, y que han sido entrenados ajustando únicamente la capa de embeddings de los modelos. Así pues, tomando un vocabulario fuente y otro objetivo, los pesos correspondientes a los tokens que se solapan son preservados, mientras que el resto son inicializados usando la media de los embeddings de la fuente. FLOR está basado en BLOOM (Scao et al., 2022), cuyo corpus de preentrenamiento (ROOTS, Laurençon et al., 2022) contiene 3.5 billones de tokens en catalán y 36.86 billones de tokens en castellano. FLOR ha sido preentrenado, además de con estos subconjuntos en catalán y castellano del corpus de BLOOM, con otros provenientes de documentos legales, artículos científicos, patentes, páginas de noticias, crawlers, etc., hasta formar 140 billones de tokens. Este número ha sido elegido para alinearse con el preentrenamiento de los modelos Chinchilla (Hoffmann et al., 2022).
- **Latxa[15].** Basado en LLaMA, trata de dar soporte al euskera. Cuenta con dos versiones: 7 billones y 70 billones de parámetros. Ha sido ajustado usando EUsCrawl[16], un conjunto de textos en euskera con 288 millones de tokens extraído de 33 páginas web con contenido de calidad en euskera.

6.2 Corpora, conjuntos de datos de evaluación, grafos de conocimiento y otros recursos semánticos

Según el entregable E4.1 Análisis y Definición de Dominios de Aplicación y Casos de Uso (Gómez-Pérez y Ortega, 2023), existe una cobertura desigual de recursos disponibles según el tipo de dominio sobre el que se quiera inyectar conocimiento a un LLM. Tras el análisis llevado a cabo en dicho documento, se llegó a la conclusión de que los dominios de mayor interés para este proyecto serían el de Salud, Legal, Patentes y Seguros. Por ello, en este apartado recapitulamos con qué recursos contamos para cada uno de ellos:

- **Salud.** Como corpora para el ajuste de los LLMs, contaríamos con PubMed[17], EC3[18] y el corpus de entrenamiento del Biomedical and Clinical LM for Spanish[19]. De estos, solo Pubmed podría ser utilizado como conjunto de datos verificado al completo. Para la evaluación de los LLMs, contaríamos con MedQA-USMLE[20], PubmedQA[21], MedMCQA[22],

[14] https://medium.com/@mpamies247/flor-6-3b-a-chinchilla-compliant-model-for-catalan-spanish-and-english-7cdb389a9aac

[15] https://www.hitz.eus/en/node/340

[16] https://www.ixa.eus/euscrawl/

[17] https://ftp.ncbi.nlm.nih.gov/pubmed/baseline/

[18] https://github.com/hltfbk/E3C-Corpus

[19] https://github.com/PlanTL-GOB-ES/lm-biomedical-clinical-es#corpora-

[20] https://github.com/jind11/MedQA

[21] https://huggingface.co/datasets/pubmed_qa

[22] https://huggingface.co/datasets/medmcqa

PharmaCoNER[23] y CANTEMIST[24]. Por último, como recursos semánticos, tendríamos el grafo de conocimiento de UMLS[25].

- **Legal.** Como corpora documental, tendríamos Multi-legal PILE[26] y el Spanish Legal Domain Corpora[27]. Dentro de Multi-legal PILE, podría utilizarse el subconjunto de EuroLEX como base de datos de verificación. Para evaluar los LLMs en este dominio, podríamos utilizar LEXGlue[28], LegalBench[29], FairLEX[30], LEXTREME[31] y QUAD[32]. Por último, como recurso semántico para este dominio, tenemos el tesauro de Eurovoc[33].
- **Patentes.** Como corpora tenemos BigPatent[34], HUPD[35], ParaPat[36] y EuroPat[37]. Todos ellos, al tratarse de texto proveniente directamente de los registros públicos de patentes, pueden ser usados como conjuntos de datos de verificación. Para evaluar un LLM ajustado al dominio de patentes, no contamos con un marco particular, por lo que quizás es necesario utilizar otros recursos pertenecientes a dominios asociados, como Ciencia o Salud. En cuanto a los recursos semánticos, se podría estudiar el uso del European Patent Registry[38], una base de patentes anotadas semánticamente.
- **Seguros.** Este dominio carece de recursos propios, por lo que será necesaria la combinación de recursos provenientes de otros dominios transversales a él (salud, legal).

Se pueden encontrar más detalles acerca de estos recursos y otros relacionados con otros dominios en el entregable E4.1 Análisis y Definición de Dominios de Aplicación y Casos de Uso (Gómez-Pérez y Ortega, 2023).

6.3 Herramientas externas

La herramienta principal a integrar será un buscador en Wikipedia Español y lenguas cooficiales. La entrada de esa herramienta es un término de búsqueda y la salida son fragmentos de texto

[23] https://github.com/TeMU-BSC/PharmaCoNER-Tagger

[24] https://github.com/TeMU-BSC/cantemist-evaluation-library

[25] https://www.nlm.nih.gov/research/umls/licensedcontent/umlsknowledgesources.html

[26] https://huggingface.co/datasets/joelniklaus/Multi_Legal_Pile

[27] https://github.com/PlanTL-GOB-ES/lm-legal-es#corpora-

[28] https://huggingface.co/datasets/lex_glue

[29] https://huggingface.co/datasets/lex_glue

[30] https://huggingface.co/datasets/coastalcph/fairlex

[31] https://huggingface.co/datasets/joelniklaus/lextreme

[32] https://huggingface.co/datasets/cuad

[33] https://eur-lex.europa.eu/browse/eurovoc.html

[34] https://huggingface.co/datasets/big_patent

[35] https://huggingface.co/datasets/HUPD/hupd

[36] https://huggingface.co/datasets/para_pat

[37] https://europat.net/

[38] https://www.epo.org/en/searching-for-patents/data/bulk-data-sets/register-data

cortos de Wikipedia. Esta herramienta permite a un modelo obtener información más completa sobre un tema. Sin embargo, requiere que el modelo extraiga las partes relevantes por sí mismo. Para poder comparar los resultados con Toolformer se propone usar el volcado de Wikipedia de KILT (Petroni et al., 2021) que fue utilizado en dicho trabajo.

Como motor de búsqueda básico se puede utilizar el algoritmo de ranking BM25 usado sistemas de búsqueda tradicionales como Elastic Search. Adicionalmente se puede usar índices de búsqueda basados en representaciones densas (embeddings) generadas, por ejemplo, con transformer bi-encoders para codificar preguntas y textos, y calcular su similitud usando el producto punto de los vectores. Las representaciones densas capturan relaciones semánticas entre palabras que no son capturadas por medio de los modelos de bolsas de palabras, que se usan en las representaciones tradicionales dispersas. Entre los enfoques de representaciones densas se propone experimentar con ColBERTv2 (Santhanam et al., 2022).

7 Conclusiones y trabajo futuro

Los métodos propuestos en este entregable afrontan el problema de la factualidad de los LLM como un problema de alineamiento, en el que se pretende alinear el modelo con una forma de responder a los usuarios más factual, con menos alucinaciones y menos sesgada. Por este motivo, los métodos de aprendizaje por refuerzo para alineamiento con preferencias como DPO se postulan como piedra angular de este trabajo. Exploramos así una nueva vía alejada de los métodos habituales que hasta ahora se basaban en ajustar los parámetros del modelo mediante su entrenamiento continuado sobre conjuntos de datos adicionales, con las limitaciones que esto implica en escenarios de escasez de datos, entre los que se incluyen objetivos de adaptación a dominio y un entorno multilingüe. También nos alejamos de métodos propuestos con anterioridad para inyectar conocimiento estructurado en arquitecturas encoders basadas en Transformers, como K-Adapter, Kformer o Ernie durante distintas fases del pre-entrenamiento del LLM.

El punto de partida de este camino es el desarrollo del marco de evaluación propuesto en este documento, agnóstico con respecto al idioma y al dominio, que es una característica fundamental en escenarios de bajos recursos. Este marco de evaluación permitirá a su vez la creación de conjuntos de datos de preferencias con los que entrenar los LLM siguiendo el algoritmo DPO, incorporando conocimiento adicional procedente de los corpora y recursos (semi)estructurados disponibles.

Planteamos por otro lado métodos de mejora de la factualidad en tiempo de inferencia mediante la expansión y refinamiento de las peticiones o prompts. Estos métodos pretenden optimizar el rendimiento del modelo para la resolución de problemas complejos que requieran no sólo recuperar información sino razonar, dotándole de estrategias estructuradas para la resolución de problemas y de expansión de su memoria paramétrica mediante información adicional bien directamente o a través de herramientas. También en estos casos pretendemos dotar al modelo de mecanismos que le permitan preferir el uso de estos recursos sobre el conocimiento contenido internamente cuando sea conveniente. Presentamos el concepto de Problem-Solving Prompting (PSP), que combina métodos clásicos de resolución de problemas en sistemas basados en conocimiento con técnicas modernas de prompting en LLM.

Nuestros próximos pasos se centran en el desarrollo de estos métodos y en la implementación de LLM que incorporen estas innovaciones. Los siguientes entregables ofrecerán una visión más detallada de los métodos aquí esbozados, así como los primeros resultados experimentales.

Referencias

Ahn, J., & Oh, A. (2021). **Mitigating language-dependent ethnic bias in BERT**. arXiv preprint arXiv:2109.05704.

Almazrouei E., Alobeidli H., Alshamsi A., Cappelli A., Cojocaru R., Debbah M., Goffinet É., Hesslow D., Launay J., Malartic Q., Mazzotta D., Noune B., Pannier B., & Penedo G. (2023). **The Falcon Series of Open Language Models.** *arXiv preprint arXiv:2311.16867*

Asai, A., Wu, Z., Wang, Y., Sil, A., & Hajishirzi, H. (2023). **Self-rag: Learning to retrieve, generate, and critique through self-reflection**. arXiv preprint arXiv:2310.11511.

Bender, EM. (2019). **A typology of ethical risks in language technology with an eye towards where transparent documentation can help.**

Benjamins, V. R. (1993). **Problem Solving Methods for Diagnosis.** PhD thesis, University of Amsterdam, Amsterdam, The Netherlands.

Benjamins, V. R., & Fensel, D. (1998). **Problem-solving methods**. International Journal of Human-Computer Studies, 49(4), 305-313.

Berquand, A., Ladeira A. V. (2022). **From Mission Description to Knowledge Graph: Applying Transformer-based models to map knowledge from publicly available satellite datasets**.

Bommasani, R., Liang, P., & Lee, T. (2023**). Holistic Evaluation of Language Models. Annals of the New York Academy of Sciences**.

Bordia, S., & Bowman, S. R. (2019). Identifying and reducing gender bias in word-level language models. arXiv preprint arXiv:1904.03035.

Brown, TB. et al. (2020). **Language Models are Few-Shot Learners.** ArXiv, abs/2005.14165.

Caliskan, A., Bryson, J. J., & Narayanan, A. (2017). **Semantics derived automatically from language corpora contain human-like biases**. *Science*, *356*(6334), 183-186.

Chen, J., Sriram, A., Choi, E., & Durrett, G. (2022). **Generating Literal and Implied Subquestions to Fact-check Complex Claims**. *In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (pp. 3495–3516). Association for Computational Linguistics.*

Chen S., Zhao Y., Zhang J., Chern I., Gao S., Liu P., & He J. (2023). **FELM: Benchmarking Factuality Evaluation of Large Language Models.** *In Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Track on Datasets and Benchmarks*

Chern, I. et al. (2023). **Factool: Factuality detection in generative ai – a tool augmented framework for multi-task and multi-domain scenarios.** CoRR, abs/2307.13528.

Chowdhery, A.- et al. (2022). **Palm: Scaling language modeling with pathways.** URL: https://arxiv.org/abs/2204.02311

Christiano, P.F., Leike, J., Brown, T.B., Martic, M., Legg, S., & Amodei, D. (2017). **Deep Reinforcement Learning from Human Preferences.** ArXiv, abs/1706.03741.

Chuang, Y., Xie, Y., Luo, H., Kim, Y., Glass, J.R., & He, P. (2023). **DoLa: Decoding by Contrasting Layers Improves Factuality in Large Language Models.** ArXiv, abs/2309.03883.

Cobbe K., Kosaraju V., Bavarian M., Chen M., Jun H., Kaiser L., Plappert M., Tworek J., Hilton J., Nakano R., Hesse C., & Schulman J. (2021). **Training Verifiers to Solve Math Word Problems.** *arXiv preprint arXiv:2110.14168*

Costa-Jussa, Marta R. et al. (2022). **No Language Left Behind: Scaling Human-Centered Machine Translation**. ArXiv, abs/2207.04672

Dai, D. et al. (2022). **Knowledge Neurons in Pretrained Transformers.** In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 8493–8502, Dublin, Ireland. Association for Computational Linguistics.

Davison, J., Feldman, J., and Rush, A. (2019). **Commonsense knowledge mining from pretrained models.** In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 1173–1178, Hong Kong, China. Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1109

Delobelle, P., Tokpo, E. K., Calders, T., & Berendt, B. (2022). **Measuring fairness with biased rulers: A comparative study on bias metrics for pre-trained language models.** In NAACL 2022: the 2022 Conference of the North American chapter of the Association for Computational Linguistics: human language technologies (pp. 1693-1706).

Denaux, R., Gomez-Perez, JM. (2020). **Linked Credibility Reviews for Explainable Misinformation Detection.** In: J. Z. Pan, V. Tamma, C. d'Amato, K. Janowicz, B. Fu, A. Polleres, O. Seneviratne, L. Kagal (Eds.), The Semantic Web – ISWC 2020, Springer International Publishing, Cham, 2020, pp. 147–163.

Denaux, R. and Gomez-Perez, JM. (2019). **Vecsigrafo: Corpus-based Word-Concept Embeddings. Bridging the Statistic-Symbolic Representational Gap in Natural Language Processing.** Semantic Web Journal 10, 5 (2019), 881–908. https://doi.org/10.3233/SW-190361

Devlin, J. et al. 2019. **BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.** In Proc. of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol.1 (Long and Short Papers), pp. 4171-4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dhamala, J., Sun, T., Kumar, V., Krishna, S., Pruksachatkun, Y., Chang, K. W., & Gupta, R. (2021, March). **Bold: Dataset and metrics for measuring biases in open-ended language generation**. In Proceedings of the 2021 ACM conference on fairness, accountability, and transparency (pp. 862-872).

Dhingra, B., Cole, JR., Eisenschlos, JM., Gillick, D., Eisenstein, J., and Cohen, WW. (2022). **Time-aware language models as temporal knowledge bases.** Transactions of the Association for Computational Linguistics, 10:257–273.

Dinan, E., Fan, A., Wu, L., Weston, J., Kiela, D., and Williams, A. (2020). Multidimensional gender bias classification. ArXiv.

Dolci, T., Azzalini, F., & Tanelli, M. (2023). **Improving Gender-Related Fairness in Sentence Encoders: A Semantics-Based Approach. Data Science and Engineering**, 1-19.

Driess, D. et al. (2023). **PaLM-E: An Embodied Multimodal Language Model.** International Conference on Machine Learning.

Fensel, D.A. (2000). **Problem-Solving Methods: Understanding, Description, Development, and Reuse.** Lecture Notes in Computer Science 1791, Springer 2000, ISBN 3-540-67816-6

Gallegos, I. O., Rossi, R. A., Barrow, J., Tanjim, M. M., Kim, S., Dernoncourt, F., ... & Ahmed, N. K. (2023). **Bias and fairness in large language models: A survey**. *arXiv preprint arXiv:2309.00770*.

García-Silva, A., Berrio, C., Gómez-Pérez, JM. (2023). **Textual Entailment for Effective Triple Validation in Object Prediction.** The Semantic Web – ISWC 2023, Springer International Publishing, Cham, 2020, to appear.

Gao L., Biderman S., Black S., Golding L., Hoppe T., Foster C., Phang J., He H., Thite A., Nabeshima N., Presser S., & Leahy C. (2020). **The Pile: An 800GB Dataset of Diverse Text for Language Modeling.** *arXiv preprint arXiv:2101.00027.*

Gao, L., Madaan, A., Zhou, S., Alon, U., Liu, P., Yang, Y., Callan, J., & Neubig, G. (2022). **PAL: Program-aided Language Models.** ArXiv, abs/2211.10435.

Gao, L., et al. (2023). **RARR: Researching and Revising What Language Models Say, Using Language Models.** In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 16477–16508, Toronto, Canada. Association for Computational Linguistics.

Gao, S., et al. (2024). **Efficient Tool Use with Chain-of-Abstraction Reasoning**. arXiv:2401.17464

Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., ... & Wang, H. (2023). **Retrieval-augmented generation for large language models: A survey**. arXiv preprint arXiv:2312.10997.

Geva M., Khashabi D., Segal E., Khot T., Roth D., & Berant J. (2021**). Did Aristotle Use a Laptop? A Question Answering Benchmark with Implicit Reasoning Strategies.** *arXiv preprint arXiv:2101.02235*

Gómez-Pérez, JM., Ortega, R. (2023). **E4.1 Análisis y Definición de Dominios de Aplicación y Casos de Uso.** KG4LLM Technical Report.

Gómez-Pérez, JM., García-Silva, A., Leone, R., Albani, M., Fontaine, M., Poncet, C., Summerer, L., Donati, A., Roma, I., Scaglioni, S. (2023). **Artificial Intelligence and Natural Language Processing and Understanding in Space: A Methodological Framework and Four ESA Case Studies.** Engineering Applications of Artificial Intelligence (to appear).

Gomez-Perez, JM., Denaux, R., Garcia-Silva, A. (2020) **A Practical Guide to Hybrid Natural Language Processing - Combining Neural Models and Knowledge Graphs for NLP.** Springer, Cham. DOI: https://doi.org/10.1007/978-3-030-44830-1

Gómez-Pérez, JM., Ortega, R. (2020). **ISAAQ - Mastering Textbook Questions with Pre-trained Transformers and Bottom-Up and Top-Down Attention.** 5469-5479. 10.18653/v1/2020.emnlp-main.441. Empirical Methods to Natural Language Processing (EMNLP) 2020.

Gomez-Perez, Jose Manuel. (2010). **Acquisition and understanding of process knowledge using problem solving methods.** Studies on the Semantic Web, IOS Press, 978-1-60750-600-3 (print) | 978-1-61499-341-4 (online). DOI: https://doi.org/10.3233/978-1-61499-341-4-i

Guo, W., & Caliskan, A. (2021, July). **Detecting emergent intersectional biases: Contextualized word embeddings contain a distribution of human-like biases**. In Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society (pp. 122-133).

He, B. et al. (2020). **BERT-MK: Integrating graph contextualized knowledge into pre-trained language models.** In Findings of the Association for Computational Linguistics: EMNLP 2020, pages 2281–2290, Online, Nov. 2020. Association for Computational Linguistics.

He, X., Tian, Y., Sun, Y., Chawla, N., Laurent, T., LeCun, Y., Bresson, X., & Hooi, B. (2024). **G-Retriever: Retrieval-Augmented Generation for Textual Graph Understanding and Question Answering.** ArXiv, abs/2402.07630.

Hendrycks D., Burns C., Basart S., Zou A., Mazeika M., Song D., & Steinhardt J. (2021). **Measuring Massive Multitask Language Understanding.** *arXiv preprint arXiv:2009.03300.*

Hoffmann J. et al. (2022). **Training Compute-Optimal Large Language Models.** *arXiv preprint arXiv:2203.15556*

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., Laroussilhe, Q.D., Gesmundo, A., Attariyan, M., & Gelly, S. (2019). **Parameter-Efficient Transfer Learning for NLP. International Conference on Machine Learning.**

Izacard, G., Lewis, P., Lomeli, M., Hosseini, L., Petroni, F., Schick, T., Yu, J.A., Joulin, A., Riedel, S., & Grave, E. (2022). **Few-shot Learning with Retrieval Augmented Language Models.** ArXiv, abs/2208.03299.

Izacard, G., Caron, M., Hosseini, L., Riedel, S., Bojanowski, P., Joulin, A., & Grave, E. (2021). **Unsupervised dense information retrieval with contrastive learning**. arXiv preprint arXiv:2112.09118.

Ji, Z. et al. 2022. **Survey of hallucination in natural language generation.** ACM Computing Surveys.

Ji, H., Grishman, R.: **Knowledge Base Population: Successful Approaches and Challenges**. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. pp. 1148–1158. Association for Computational Linguistics, Portland, Oregon, USA (Jun 2011), https://aclanthology.org/P11-1115

Kadavath, S., Conerly, T., Askell, A., Henighan, T.J., Drain, D., Perez, E., Schiefer, N., Dodds, Z., DasSarma, N., Tran-Johnson, E., Johnston, S., El-Showk, S., Jones, A., Elhage, N., Hume, T., Chen, A., Bai, Y., Bowman, S., Fort, S., Ganguli, D., Hernandez, D., Jacobson, J., Kernion, J., Kravec, S., Lovitt, L., Ndousse, K., Olsson, C., Ringer, S., Amodei, D., Brown, T.B., Clark, J., Joseph, N., Mann, B., McCandlish, S., Olah, C., & Kaplan, J. (2022). **Language Models (Mostly) Know What They Know.** ArXiv, abs/2207.05221.

Kamoi, R., Goyal, T., Rodriguez, J.D., & Durrett, G. (2023). **WiCE: Real-World Entailment for Claims in Wikipedia.** ArXiv, abs/2303.01432.

Kaneko, M., & Bollegala, D. (2022, June). **Unmasking the mask–evaluating social biases in masked language models**. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 36, No. 11, pp. 11954-11962).

Kandpal, N., Deng, H., Roberts, A., Wallace, E., Raffel, C. (2022). **Large Language Models Struggle to Learn Long-Tail Knowledge.**

Kirkpatrick, J. et al. 2017. **Overcoming catastrophic forgetting in neural networks.** Proceedings of the national academy of sciences, 114(13):3521–3526.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., & Iwasawa, Y. (2022). **Large language models are zero-shot reasoners**. *Advances in neural information processing systems*, *35*, 22199-22213.

Komeili, M., Shuster, K., and Weston, J. (2022). **Internet-augmented dialogue generation.** In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 8460–8478, Dublin, Ireland. Association for Computational Linguistics.

Kryscinski, W., McCann, B., Xiong, C., and Socher, R. (2020). **Evaluating the Factual Consistency of Abstractive Text Summarization.** In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 9332–9346, Online. Association for Computational Linguistics.

Kuhn, L., Gal, Y. and Farquhar, S. (2023) **Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation.**

Kurita, K., Vyas, N., Pareek, A., Black, A. W., & Tsvetkov, Y. (2019). **Measuring bias in contextualized word representations**. arXiv preprint arXiv:1906.07337.

Laurençon, H. et al. (2022). **The BigScience ROOTS Corpus: A 1.6TB Composite Multilingual Dataset.** *In Advances in Neural Information Processing Systems (pp. 31809–31826). Curran Associates, Inc.*

Lauscher, A., Majewska, O., Ribeiro, L., Gurevych, I., Rozanov, N., Glavaš, G. (2020). **Common Sense or World Knowledge? Investigating Adapter-Based Knowledge Injection into Pretrained Transformers**. 43-49. 10.18653/v1/2020.deelio-1.5.

Lawrence, Peter. (2024). **Text-to-Graph via LLM: pre-training, prompting, or tuning?.** https://medium.com/@peter.lawrence_47665/text-to-graph-via-llm-pre-training-prompting-or-tuning-3233d1165360

Lawrence, Peter. (2023). **Large Language Model = Knowledge Graph Store? Yes, by Fine-Tuning LLM With KG**. https://betterprogramming.pub/large-language-model-knowledge-graph-store-yes-by-fine-tuning-llm-with-kg-f88b556959e6

Lehmann, J. et al. (2015). **DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia.** Semantic Web, 6, 167-195.

Levine, Y. et al. (2020). **SenseBERT: Driving some sense into BERT.** In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, July 2020.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Kiela, D. (2020). **Retrieval-augmented generation for knowledge-intensive nlp tasks**. Advances in Neural Information Processing Systems, 33, 9459-9474.

Li, K., Patel, O., Vi'egas, F., Pfister, H., & Wattenberg, M. (2023). **Inference-Time Intervention: Eliciting Truthful Answers from a Language Model.** ArXiv, abs/2306.03341.

Lin et al. (2022). **Few-shot Learning with Multilingual Generative Language Models.** In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 9019–9052, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Lin, S., Hilton, J., and Evans, O. 2022. **TruthfulQA: Measuring How Models Mimic Human Falsehoods.** In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Liu, Y. et al. (2019). **RoBERTa: A Robustly Optimized BERT Pretraining Approach.** ArXiv, abs/1907.11692.

Liu, Y., Fabbri, A., Liu, P., Zhao, Y., Nan, L., Han, R., Han, S., Joty, S., Wu, C.S., Xiong, C., & Radev, D. (2023). **Revisiting the Gold Standard: Grounding Summarization Evaluation with Robust Human Evaluation.** *In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 4140–4170). Association for Computational Linguistics.*

Logan IV, R. L., Liu, N. F., Peters, M. E., Gardner, M., & Singh, S. (2019). ***Barack's wife hillary: Using knowledge-graphs for fact-aware language modeling****. arXiv preprint arXiv:1906.07241.*

Malaviya C, Lee S., Chen S., Sieber E., Yatskar M., & Roth D. (2023). **ExpertQA: Expert-Curated Questions and Attributed Answers.** *arXiv preprint arXiv:2309.07852.*

Mallen, A., Asai, A., Zhong, V., Das, R., Khashabi, D., and Hajishirzi, H. 2023. **When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories.** In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 9802–9822, Toronto, Canada. Association for Computational Linguistics.

Manakul, P., Liusie, A. and Gales, M. 2023. **SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models.** In Proceedings of the 2023 Conference on Empirical

Methods in Natural Language Processing, pages 9004–9017, Singapore. Association for Computational Linguistics.

May, C., Wang, A., Bordia, S., Bowman, S. R., & Rudinger, R. (2019). **On measuring social biases in sentence encoders**. arXiv preprint arXiv:1903.10561.

Maynez, J., Narayan, S., Bohnet, B, and McDonald, R. (2020). **On Faithfulness and Factuality in Abstractive Summarization.** In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 1906–1919, Online. Association for Computational Linguistics.

Mcdermott, J., 1988. **Preliminary Steps Toward a Taxonomy of Problem-Solving Methods.** Springer US, Boston, MA. pp. 225–256. DOI: https://doi.org/10.1007/978-1-4684-7122-9_8

Meng, K., Bau, D., Andonian, A., & Belinkov, Y. (2022). **Locating and Editing Factual Associations in GPT.** *In Advances in Neural Information Processing Systems (pp. 17359–17372). Curran Associates, Inc.*

Mesquita, F., Cannaviccio, M, Schmidek, J., Mirza, P., and Barbosa, D. (2019). **KnowledgeNet: A Benchmark Dataset for Knowledge Base Population.** In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 749–758, Hong Kong, China. Association for Computational Linguistics.

Miller, GA. (1994). **WordNet: A Lexical Database for English.** In Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994.

Min S., Shi W., Lewis M., Chen X., Yih W., Hajishirzi H., & Zettlemoyer L. (2023). **Nonparametric Masked Language Modeling.** *arXiv preprint arXiv:2212.01349*

Min, S., Krishna, K., Lyu, X., Lewis, M., Yih, W.T., Koh, P., Iyyer, M., Zettlemoyer, L., and Hajishirzi, H. (2023). **Factscore: Fine-grained atomic evaluation of factual precision in long form text generation**.

Muhlgay, D., Ram, O., Magar, I., Levine, Y., Ratner, N., Belinkov, Y., Abend, O., Leyton-Brown, K., Shashua, A., & Shoham, Y. (2023). **Generating Benchmarks for Factuality Evaluation of Language Models.** ArXiv, abs/2307.06908.

Nadeem, M., Bethke, A., & Reddy, S. (2020). StereoSet: **Measuring stereotypical bias in pretrained language models**. arXiv preprint arXiv:2004.09456.

Nadeem, M., Bethke, A., and Reddy, S. (2021). **StereoSet: Measuring stereotypical bias in pretrained language models.** In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 5356–5371, Online. Association for Computational Linguistics.

Nangia, N., Vania, C., Bhalerao, R., and Bowman, S. (2020). **CrowS-Pairs: A Challenge Dataset for Measuring Social Biases in Masked Language Models.** In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1953–1967, Online. Association for Computational Linguistics.

Névéol, A., Dupont, Y., Bezançon, J., & Fort, K. (2022, May). **French CrowS-Pairs: Extending a challenge dataset for measuring social bias in masked language models to a language other than English**. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 8521-8531).

A. Newell, J. C. Shaw, and H. A. Simon. **Report on a general problem solving program**. In IFIP congress, volume 256, page 64. Pittsburgh, PA, 1959.

A. Newell, H. A. Simon, et al. **Human problem solving**. Prentice-Hall, 1972.

Ni J., Qu C., Lu J., Dai Z., Ábrego GH., Ma J., Zhao VY., Luan Y., Hall KB., Chang M., & Yang Y. (2021). **Large Dual Encoders Are Generalizable Retrievers**. *arXiv preprint arXiv:2112.07899*

Nozza, D., Bianchi, F., & Hovy, D. (2021). HONEST: **Measuring hurtful sentence completion in language models**. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics.

O'Hern, M.S. & Rindfleisch, A. (2010). **Customer Co-Creation: A Typology and Research Agenda.** In: Review of Marketing Research, vol. 6, p. 84-106.

Opitz, J. (2019). **Argumentative relation classification as plausibility ranking.** In Preliminary Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019): Long Papers,

pages 193–202, Erlangen, Germany. German Society for Computational Linguistics & Language Technology.

Ouyang L., Wu J., Jiang X., Almeida D., Wainwright CL, Mishkin P., Zhang C., Agarwal S., Slama K., Ray A., Schulman J., Hilton J., Kelton F., Miller L., Simens M., Askell A., Welinder P., Christiano P., Leike J., & Lowe R. (2022). **Training language models to follow instructions with human feedback.** *arXiv preprint arXiv:2203.02155*.

Parisi, A. & Zhao, Y. & Fiedel, N. (2022). **TALM: Tool Augmented Language Models.** 10.48550/arXiv.2205.12255.

Patel, A., Bhattamishra, S., and Goyal, N. (2021). **Are NLP models really able to solve simple math word problems?** In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 2080–2094, Online. Association for Computational Linguistics.

Penedo, G. et al. (2023). **The RefinedWeb Dataset for Falcon LLM: Outperforming Curated Corpora with Web Data, and Web Data Only.** ArXiv, abs/2306.01116.

Peters, ME., Neumann, M., Logan, R, Schwartz, R., Joshi, V, Singh, S., and Smith, N. (2019). **Knowledge Enhanced Contextual Word Representations.** In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 43–54, Hong Kong, China. Association for Computational Linguistics.

Petroni, F. et al. (2019). **Language Models as Knowledge Bases.** In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Puri, R., & Catanzaro, B. (2019). **Zero-shot Text Classification With Generative Language Models.** ArXiv, abs/1912.10165.

Qian, R., Ross, C., Fernandes, J., Smith, E., Kiela, D., & Williams, A. (2022). **Perturbation augmentation for fairer nlp**. *arXiv preprint arXiv:2205.12586*.

Qiao, S., Ou, Y., Zhang, N., Chen, X., Yao, Y., Deng, S., ... & Chen, H. (2022). **Reasoning with language model prompting: A survey**. arXiv preprint arXiv:2212.09597.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). **Language Models are Unsupervised Multitask Learners.**

Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C.D., & Finn, C. (2023). **Direct Preference Optimization: Your Language Model is Secretly a Reward Model.** ArXiv, abs/2305.18290.

Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016). **Squad: 100,000+ questions for machine comprehension of text**. arXiv preprint arXiv:1606.05250.

Rehm, G., & Way, A. (2023). **Strategic Research, Innovation and Implementation Agenda for Digital Language Equality in Europe by 2030.** European Language Equality. Springer Cham. https://doi.org/10.1007/978-3-031-28819-7_45

Rehm G. et al. (2023). **European Language Grid A Language Technology Platform for Multilingual Europe.** Springer Cham. https://doi.org/10.1007/978-3-031-17258-8

Sanh, Victor & Webson, Albert & Raffel, Colin & Bach, Stephen & Sutawika, Lintang & Alyafeai, Zaid & Chaffin, Antoine & Stiegler, Arnaud & Scao, Teven & Raja, Arun & Dey, Manan & Bari, M & Xu, Canwen & Thakker, Urmish & Sharma, Shanya & Szczechla, Eliza & Kim, Taewoon & Chhablani, Gunjan & Nayak, Nihal & Rush, Alexander. (2021). **Multitask Prompted Training Enables Zero-Shot Task Generalization.**

Sap, M. et al. (2019). **ATOMIC: an atlas of machine commonsense for if-then reasoning.** In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence (AAAI'19/IAAI'19/EAAI'19). AAAI Press, Article 372, 3027–3035. https://doi.org/10.1609/aaai.v33i01.33013027

Santhanam K. et al. (2022). **ColBERTv2: Effective and Efficient Retrieval via Lightweight Late Interaction**. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 3715–3734, Seattle, United States. Association for Computational Linguistics.

Scao, TL. et al. (2022). **BLOOM: A 176B-Parameter Open-Access Multilingual Language Model.** ArXiv, abs/2211.05100.

Schick, T., et al. (2023). **Toolformer: Language Models Can Teach Themselves to Use Tools.** ArXiv, abs/2302.04761.

Schick, T., Udupa, S., and Schutze, H. (2021). **Self-diagnosis and self-debiasing: A proposal for reducing corpus-based bias in NLP.** Transactions of the Association for Computational Linguistics. https://doi.org/10.1162/tacl_a_00434

Schick, T., Schutze, H. (2021). **Generating datasets with pretrained language models.** In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.emnlp-main.555

Schreiber, G. et al. (1994). **CommonKADS: A comprehensive methodology for KBS development.** IEEE Expert. 9. 28-37. 10.1109/64.363263.

Schreiber, G. (2000). **Knowledge engineering and management: the CommonKADS methodology**. MIT press.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). **Proximal policy optimization algorithms**. arXiv preprint arXiv:1707.06347.

Shuster, K., Poff, S., Chen, M., Kiela, D., and Weston, J. (2021). **Retrieval Augmentation Reduces Hallucination in Conversation.** In Findings of the Association for Computational Linguistics: EMNLP 2021, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Speer, R., Chin, J., and Havasi, C. (2017). **ConceptNet 5.5: an open multilingual graph of general knowledge.** In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI'17). AAAI Press, 4444–4451.

Sun, Y., Wang, S., Li, Y., Feng, S., Tian, H., Wu, H., and Wang, H. (2020). **ERNIE 2.0: A continual pre-training framework for language understanding.** In The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020, pages 8968–8975. AAAI Press, 2020.

Thorne, J., Vlachos, A., Christodoulopoulos, C., and Mittal, A. (2018). **FEVER: a Large-scale Dataset for Fact Extraction and VERification.** In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Thoppilan, R. et al. (2022). **LaMDA: Language Models for Dialog Applications.** ArXiv, abs/2201.08239.

Tian, K., Mitchell, E., Zhou, A., Sharma, A., Rafailov, R., Yao, H., Finn, C., & Manning, C.D. (2023). **Just Ask for Calibration: Strategies for Eliciting Calibrated Confidence Scores from Language Models Fine-Tuned with Human Feedback.** ArXiv, abs/2305.14975.

Tian, K., Mitchell, E., Yao, H., Manning, C.D., & Finn, C. (2023). **Fine-tuning Language Models for Factuality**. ArXiv, abs/2311.08401.

Touileb, S., Øvrelid, L., & Velldal, E. (2022, July). **Occupational biases in Norwegian and multilingual language models**. In Proceedings of the 4th Workshop on Gender Bias in Natural Language Processing (GeBNLP) (pp. 200-211).

Touvron, H. et al. (2023). **Llama 2: Open Foundation and Fine-Tuned Chat Models.** ArXiv, abs/2307.09288.

Vashishtha, A., Ahuja, K., & Sitaram, S. (2023). **On evaluating and mitigating gender biases in multilingual settings**. *arXiv preprint arXiv:2307.01503*.

Vaswani, A. et al. (2017). **Attention is All you Need.** NIPS.

Wang, R. et al. (2021). **K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters.** In Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 1405–1418, Online. Association for Computational Linguistics.

Wang, A., Cho, K., and Lewis, M. (2020). **Asking and Answering Questions to Evaluate the Factual Consistency of Summaries.** In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 5008–5020, Online. Association for Computational Linguistics.

Wang, L., Xu, W., Lan, Y., Hu, Z., Lan, Y., Lee, R.K., & Lim, E. (2023). **Plan-and-Solve Prompting: Improving Zero-Shot Chain-of-Thought Reasoning by Large Language Models.** Annual Meeting of the Association for Computational Linguistics.

Wang, Y., Mishra, S., Alipoormolabashi, P., Kordi, Y., Mirzaei, A., Naik, A., Ashok, A., Dhanasekaran, A., Arunkumar, A., Stap, D., Pathak, E., Karamanolakis, G., Lai, H., Purohit, I., Mondal, I., Anderson, J., Kuznia, K., Doshi, K., Pal, K., Patel, M., Moradshahi, M., Parmar, M., Purohit, M., Varshney, N., Kaza, P., Verma, P., Puri, R., Karia, R., Doshi, S., Sampat, S., Mishra, S., Reddy A, S., Patro, S., Dixit, T., & Shen, X. (2022). **Super-NaturalInstructions: Generalization via Declarative Instructions on 1600+ NLP Tasks**. *In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (pp. 5085–5109). Association for Computational Linguistics.*

Wang Y., Kordi Y., Mishra S., Liu A., Smith NA., Khashabi D., & Hajishirzi H. (2023). **Self-Instruct: Aligning Language Models with Self-Generated Instructions.** *arXiv preprint arXiv:2212.10560.*

Wang, L., Xu, W., Lan, Y., Hu, Z., Lan, Y., Lee, R. K. W., & Lim, E. P. (2023). **Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models**. arXiv preprint arXiv:2305.04091.

Webster, K., Wang, X., Tenney, I., Beutel, A., Pitler, E., Pavlick, E., ... & Petrov, S. (2020). **Measuring and reducing gendered correlations in pre-trained models**. arXiv preprint arXiv:2010.06032.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., ... & Zhou, D. (2022). **Chain-of-thought prompting elicits reasoning in large language models**. Advances in Neural Information Processing Systems, 35, 24824-24837

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., ... & Fedus, W. (2022). **Emergent abilities of large language models**. arXiv preprint arXiv:2206.07682.

Wilkinson, M., Dumontier, M., Aalbersberg, I. et al. **The FAIR Guiding Principles for scientific data management and stewardship.** Sci Data 3, 160018 (2016). https://doi.org/10.1038/sdata.2016.18

Xiong, W., Du, J, Wang, W., and Stoyanov, V. (2020). **Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model.** In 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net, 2020.

Yao, Y., Huang, S., Zhang, N., Dong, L., Wei, F., & Chen, H. (2022). **Kformer: Knowledge Injection in Transformer Feed-Forward Layers.** ArXiv, abs/2201.05742.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., & Narasimhan, K. (2023). **Tree of thoughts: Deliberate problem solving with large language models**, may 2023. *arXiv preprint arXiv:2305.10601*, *14*.

Hongbin Ye, Ningyu Zhang, Hui Chen, and Huajun Chen. 2022. **Generative Knowledge Graph Construction: A Review**. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 1–17, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yu, D. et al. (2022). **KG-FiD: Infusing Knowledge Graph in Fusion-in-Decoder for Open-Domain Question Answering.** In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 4961–4974.

Zamani, H., Diaz, F., Dehghani, M., Metzler, D., and Bendersky, M. (2022). **Retrieval-Enhanced Machine Learning.** In Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22). Association for Computing Machinery, New York, NY, USA, 2875–2886. https://doi.org/10.1145/3477495.3531722

Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M, and Liu, Q. (2019). **ERNIE: Enhanced language representation with informative entities.** In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1441–1451, Florence, Italy, July 2019. Association for Computational Linguistics.

Zhou, C., Li, Q., Li, C., Yu, J., Liu, Y., Wang, G., ... & Sun, L. (2023). **A comprehensive survey on pretrained foundation models: A history from bert to chatgpt**. *arXiv preprint arXiv:2302.09419*.