Cómo funcionan los LLM
Las 10 preguntas más frecuentes de los ejecutivos

Los líderes empresariales que toman decisiones relacionadas con la IA necesitan comprender los fundamentos del funcionamiento de los grandes modelos de lenguaje y las herramientas GenAI basadas en ellos. Infórmese sobre estos temas frecuentemente malinterpretados.

Rama Ramakrishnan

24 de septiembre de 2025

MIT Sloan Management Review

How LLMs work: Top 10 executive-level questions

sloanreview.mit.edu

EN MI TRABAJO en MIT Sloan School of Management he enseñado los conceptos básicos de cómo funcionan los modelos extensos de lenguaje (LLM, large language models) a muchos ejecutivos durante los últimos dos años.

Algunas personas postulan que los líderes empresariales no quieren ni necesitan saber cómo funcionan los LLM y las herramientas de IA generativa que impulsan, y que solo les interesan los resultados que estas herramientas pueden ofrecer. Esa no es mi experiencia. Los líderes con visión de futuro se preocupan por los resultados, por supuesto, pero también son plenamente conscientes de que un modelo mental claro y preciso de cómo funcionan los LLM es fundamental para tomar decisiones empresariales acertadas respecto al uso de tecnologías de IA en la empresa.

En esta columna comparto preguntas sobre 10 temas que suelen malinterpretarse y sobre los que me preguntan con frecuencia, junto con sus respuestas. No es necesario leer un libro sobre cada uno de estos temas, ni profundizar en los detalles técnicos, pero sí es necesario comprender lo esencial.

Considere esta lista como una referencia útil para usted y para sus equipos, colegas o clientes la próxima vez que surja una de estas preguntas en una conversación con ellos. He escuchado de mis estudiantes de nivel ejecutivo en el MIT que este conocimiento es especialmente útil para reflexionar sobre la realidad en las conversaciones con socios tecnológicos.

1. Entiendo que los LLM generan texto a texto. ¿Cómo decide el LLM cuándo detenerse?

Dicho de otro modo, ¿cuándo decide el LLM dar al usuario la respuesta final a una pregunta? La decisión de dejar de generar se determina mediante una combinación de lo que predice el LLM y las reglas establecidas por el sistema de software que lo ejecuta. No es una decisión exclusiva del LLM. Analicemos en detalle cómo funciona esto.

Cuando un LLM responde a una pregunta, produce texto fragmento a fragmento. El nombre técnico de cada fragmento es token.[1] Los tokens pueden ser palabras o partes de palabras. En cada paso, el LLM predice qué token debe seguir según la consigna y lo que ya ha escrito.[2]

Un sistema externo ejecuta el LLM en un bucle de "generar el siguiente token; añadirlo a la entrada; generar el siguiente token" hasta que se activa una *condición de detención* (*stopping condition*). Cuando esto sucede, el sistema deja de solicitar más tokens al LLM y muestra el resultado al usuario.

En la práctica, se utilizan muchas condiciones de parada. Una importante implica un token especial de “fin de secuencia” que (informalmente) significa “fin de respuesta” (*end of answer*). Este token se utiliza en el proceso de entrenamiento para indicar el final de los ejemplos de entrenamiento individuales y, por lo tanto, durante el entrenamiento, el LLM aprende a predecir este token especial en el punto donde se completa la respuesta. Otras condiciones de parada incluyen (entre otras) un límite en el número máximo de tokens generados hasta el momento o la generación de un patrón definido por el usuario denominado *secuencia de detención* (*stop sequence*).

Cuando usamos la versión web de una herramienta como ChatGPT como consumidores, no vemos este proceso, solo el texto final. Sin embargo, cuando su organización comienza a desarrollar sus propias aplicaciones LLM, los desarrolladores pueden ajustar estas reglas de detención y otros parámetros por sí mismos, y estas decisiones pueden afectar la integridad, el costo y el formato de las respuestas.

El punto importante aquí es que la “decisión” de detenerse es una interacción entre las predicciones del token LLM y la lógica de control externo, no una decisión tomada por el LLM.

2. Si el LLM comete un error y lo corrijo, ¿se actualizará inmediatamente?

No, el LLM no se actualizará automáticamente si lo corrige. Si utiliza herramientas como ChatGPT o Claude, su corrección podría ayudar a mejorar futuras versiones del modelo si incluye su historial de chat en una próxima ejecución de entrenamiento, pero esas actualizaciones se realizan durante semanas o meses, no de forma instantánea.

Algunas aplicaciones, como ChatGPT, tienen una función de memoria que se actualiza en tiempo real para recordar información personal como tu nombre, preferencias o ubicación. Sin embargo, esta memoria se utiliza para la personalización y no parece utilizarse para corregir errores de conocimiento factual o razonamiento del modelo.

---

[1] En promedio, un token equivale aproximadamente a tres cuartos de una palabra, y los LLM modernos tienen un vocabulario de decenas de miles a más de 100,000 tokens. Puedes introducir diferentes preguntas en la herramienta Tokenizer de OpenAI y ver cómo se tokeniza una palabra para comprenderla mejor.

[2] Estrictamente hablando, dada una entrada, el LLM genera una probabilidad (es decir, un número entre 0,0 y 1,0) para cada token de su vocabulario. Se puede considerar la probabilidad de un token como una medida de su idoneidad para ser el siguiente. Para todos los tokens del vocabulario, las probabilidades suman 1,0. El siguiente token se selecciona en función de estas probabilidades mediante diversas estrategias controlables por el desarrollador (como elegir el token con la mayor probabilidad o seleccionar un token aleatoriamente en proporción a su probabilidad).

3. Si el LLM genera repetidamente un token a la vez en función de la conversación actual, ¿por qué he visto que utiliza información de una conversación anterior (digamos, de hace una semana) en la respuesta?

Las aplicaciones LLM generan respuestas token por token, según la información recibida en la conversación. Por defecto, no utilizan conversaciones anteriores. Sin embargo, como se indica en la respuesta anterior, algunas aplicaciones LLM tienen una función de memoria que les permite almacenar información de chats anteriores, como tu nombre, intereses, preferencias, proyectos en curso o temas de consulta frecuente.

Al iniciar un nuevo chat, es posible que se añadan automáticamente fragmentos relevantes de esta memoria almacenada al mensaje en segundo plano. Esto significa que el modelo no recuerda chats anteriores en tiempo real, sino que recibe recordatorios de esa información como parte de la entrada. Así es como puede parecer que "recuerda" cosas de hace una semana.

Los detalles sobre qué se almacena y cuándo se utiliza varían según el proveedor y no se han revelado los métodos exactos. Es posible que se esté utilizando una técnica como la *generación aumentada por recuperación* (RAG, *retrieval-augmented generation*) para decidir qué elementos de memoria incluir en un nuevo mensaje. Muchas plataformas permiten a los usuarios ver, editar o desactivar la memoria por completo. En la aplicación ChatGPT, por ejemplo, se puede acceder a esto a través de Ajustes > Personalización.

Si no lo conoce, RAG es una técnica que proporciona al LLM acceso a un conjunto específico de datos confidenciales. Esto le permite proporcionar respuestas útiles.

4. Entiendo que los LLM tienen una fecha límite de formación y no tienen conocimiento de lo ocurrido después de esa fecha. Sin embargo, pueden responder preguntas sobre eventos ocurridos después de esa fecha. ¿Cómo funciona esto?

Al preguntar sobre algo ocurrido después de la fecha límite de formación de un LLM, el modelo no tiene conocimiento del evento a menos que tenga acceso a información actualizada. Algunos sistemas, como ChatGPT con navegación habilitada, pueden realizar búsquedas web en tiempo real para responder a estas preguntas.

En esos casos, el LLM puede generar una consulta de búsqueda basada en su pregunta, y una parte independiente del sistema (externa al propio modelo) realiza la búsqueda. Los resultados se envían al LLM para que este pueda generar una respuesta basada en esa información actualizada. Sin embargo, no todos los LLM ni todas las aplicaciones cuentan con esta capacidad. Sin acceso a datos en tiempo real, un modelo podría generar una respuesta basada en sus datos de entrenamiento, que no reflejan las actualizaciones del mundo real.

5. Si incluyo documentos como parte de una solicitud, ¿puedo asegurarme de que el LLM utilice únicamente los documentos proporcionados al generar la respuesta?

Por ejemplo, si subo un documento de política de gastos corporativos y hago una pregunta, ¿puedo asegurarme de que utilice únicamente este documento y no los documentos de política encontrados en la web y en los que se capacitó?

No. Si bien la estimulación cuidadosa y técnicas como RAG (generación aumentada por recuperación) pueden incentivar a un modelo de IA a priorizar un conjunto de documentos proporcionados, no se puede obligar a los LLM estándar a usar solo ese contenido. El modelo aún tiene acceso a los patrones y datos aprendidos durante el entrenamiento y puede integrar ese conocimiento en su respuesta, especialmente si los datos de entrenamiento incluían contenido similar.

6. Los LLM a veces citan las fuentes utilizadas para generar la respuesta a una pregunta. Si una respuesta incluye citas que la respalden, ¿puedo confiar en ella?

No. Los LLM pueden inventar citas o usar fuentes reales de forma inexacta o engañosa. Algunos sistemas LLM incluyen pasos de posprocesamiento para verificar las citas, pero estas comprobaciones no siempre son fiables ni exhaustivas. Verifique siempre que la fuente citada exista realmente y que su contenido respalde genuinamente la información de la respuesta.

7. Cuando tenemos muchos documentos, usamos RAG, donde primero recopilamos la información relevante de los documentos e incluimos solo la que aparece en la solicitud.

Sin embargo, los LLM modernos tienen ventanas de contexto extensas, y podemos incluir fácilmente todos los documentos. ¿Es realmente necesario RAG?

Los LLM modernos, como GPT-4.1 y Gemini 2.5, ofrecen ventanas de contexto de millones de tokens, suficientes para albergar libros completos. Esto, naturalmente, plantea la pregunta: si podemos abarcarlo todo, ¿para qué molestarnos en usar un subconjunto?

Si bien estas ventanas de contexto extendidas son eficaces, incluir todos los documentos en el mensaje no siempre es recomendable. Hay varias razones por las que RAG (generación aumentada por recuperación) sigue siendo importante.

En primer lugar, RAG no se trata solo de que la consigna sea breve. Se trata de seleccionar las partes más relevantes de los documentos. Sobrecargar el contexto con demasiada información

o información irrelevante puede perjudicar el rendimiento, y mantener el contexto y la consigna relevantes, concisos y precisos suele generar mejores respuestas.

En segundo lugar, aunque los LLM pueden aceptar contextos extensos, no procesan todas las partes con la misma eficacia. Las investigaciones han demostrado que los modelos de IA tienden a centrarse más en el principio y el final de una instrucción y pueden pasar por alto información importante en la parte intermedia.

Finalmente, las solicitudes más largas implican más tokens, lo que aumenta los costos de la API y ralentiza las respuestas. Esto es importante en aplicaciones reales donde el costo y la velocidad son importantes.

En resumen, las ventanas de contexto largas son útiles, pero no invalidan la recuperación. RAG sigue siendo una herramienta importante, especialmente cuando se prioriza la precisión, la eficiencia o el coste. La opción de usar RAG debe evaluarse según las necesidades de su aplicación específica.

8. ¿Se pueden eliminar las alucinaciones LLM?

No, las alucinaciones no pueden eliminarse por completo con la tecnología LLM actual. Surgen de la naturaleza probabilística de los modelos lingüísticos, que generan texto prediciendo secuencias de tokens probables con base en datos de entrenamiento, no verificando hechos con una fuente confiable.

Sin embargo, una ingeniería rápida y cuidadosa y estrategias como RAG, el ajuste fino de datos específicos del dominio y el posprocesamiento con verificaciones basadas en reglas o validación externa pueden reducir las alucinaciones en casos de uso específicos.[3] Si bien estas estrategias no garantizan la eliminación de las alucinaciones, pueden mejorar la confiabilidad de un LLM lo suficiente para muchas aplicaciones prácticas.

9. Dado que las alucinaciones y los errores del LLM no se pueden eliminar, necesitamos verificar las respuestas. ¿Cómo podemos hacerlo eficientemente?

La verificación eficiente de los resultados del LLM depende del tipo de tarea y del nivel de riesgo aceptable. En general, las principales estrategias incluyen la revisión humana y los métodos automatizados.

Para tareas abiertas como resúmenes, ensayos, informes o análisis, la revisión humana proporciona la supervisión más fiable. Sin embargo, esto es costoso y difícil de escalar, especialmente en escenarios que requieren respuestas rápidas o en tiempo real. Una forma de mejorar la eficiencia es revisar solo un subconjunto de resultados (es decir, emplear muestreo) o realizar un triaje según el riesgo, centrando la atención humana en los casos críticos.

Una alternativa cada vez más popular es usar un "juez de IA", que suele ser otro LLM que puede evaluar o verificar los resultados de la primera herramienta. Este enfoque permite una compro-

[3] Para una encuesta reciente de investigación académica sobre este tema, véase Y. Wang, M. Wang, MA Manzoor et al., *Factuality of large language models: A survey*, en Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (Miami: Association for Computational Linguistics, 12-16 de noviembre de 2024), 19519-19529.

bación de precisión escalable y rápida, pero presenta limitaciones: el propio juez puede alucinar o no coincidir con el juicio humano, especialmente en casos complejos. Algunas mejoras incluyen el uso de múltiples jueces para la comparación, la combinación de la retroalimentación de los jueces con la verificación de datos basada en la recuperación, o el diseño de flujos de trabajo donde los resultados con baja fiabilidad se escalan a los humanos.

Las tareas estructuradas, como generar código, clasificar información o producir datos estructurados en formatos como SQL o JSON, se prestan más fácilmente a la automatización. El código generado puede probarse automáticamente con pruebas unitarias o ejecutarse en un entorno de pruebas. Los resultados de la clasificación pueden comprobarse para garantizar que se ajusten a categorías predefinidas. La validez sintáctica de formatos estructurados como JSON, SQL o XML puede comprobarse automáticamente, aunque esto solo garantiza un formato correcto, no la precisión del contenido en sí.

En resumen, las estrategias de verificación más eficientes combinan la automatización y la supervisión humana. Las herramientas automatizadas proporcionan velocidad y escalabilidad, y la supervisión humana proporciona fiabilidad. Al combinar estos métodos y utilizar un triaje con conciencia de riesgos, las organizaciones pueden lograr un equilibrio razonable entre el control de calidad y la eficiencia.

10. Estamos desarrollando un chatbot basado en LLM y queremos garantizar que su respuesta a una pregunta se mantenga inalterada cuando diferentes usuarios la hagan (o cuando un usuario la haga en distintos momentos). ¿Es posible?

Si por “garantía” se refiere a exactamente las mismas palabras cada vez, la respuesta corta es no.

Si se plantea la misma pregunta en diferentes ocasiones con diferentes palabras, es muy probable que las respuestas del LLM cambien. Pero incluso si se utiliza exactamente el mismo enunciado, es casi imposible garantizar que se genere exactamente la misma respuesta cada vez.

Puede reducir la variabilidad configurando ciertos ajustes de LLM (por ejemplo, estableciendo "temperatura" a cero), bloqueando la versión exacta del modelo e incluso autoalojándolo para controlar todo el hardware y el software. Pero incluso así, los factores técnicos dificultan enormemente la eliminación total de la variación en entornos de producción reales.[4] Por lo tanto, ocasionalmente observará pequeños cambios en la redacción o el énfasis que no alteran el significado de la respuesta subyacente. Tenga en cuenta que esto puede ser adecuado si le importa principalmente el significado de las respuestas, más que su redacción exacta.

La única manera de garantizar una redacción idéntica es almacenar (en caché) la respuesta la primera vez que se genera y mostrar ese texto almacenado siempre que se detecte la misma pregunta. Este método funciona bien si la detección de repeticiones es perfecta, pero en la práctica, las preguntas reformuladas o ligeramente modificadas pueden eludir la caché y activar la regeneración de LLM, lo que puede generar una respuesta diferente.

En resumen: puedes lograr respuestas extremadamente consistentes, pero no es posible garantizar una redacción del 100 % con la tecnología actual.

[4] Por nombrar algunos: operaciones de GPU no deterministas, diferencias de redondeo de punto flotante y actualizaciones silenciosas del backend.