



Una Introducción a la IA Generativa y los Grandes Modelos de Lenguaje (LLMs)

Jose Orlando Maldonado Bautista

¿Qué es la IA Generativa?

La IA generativa (también conocida como IA gen) es un tipo de inteligencia artificial capaz de **crear contenido original**.

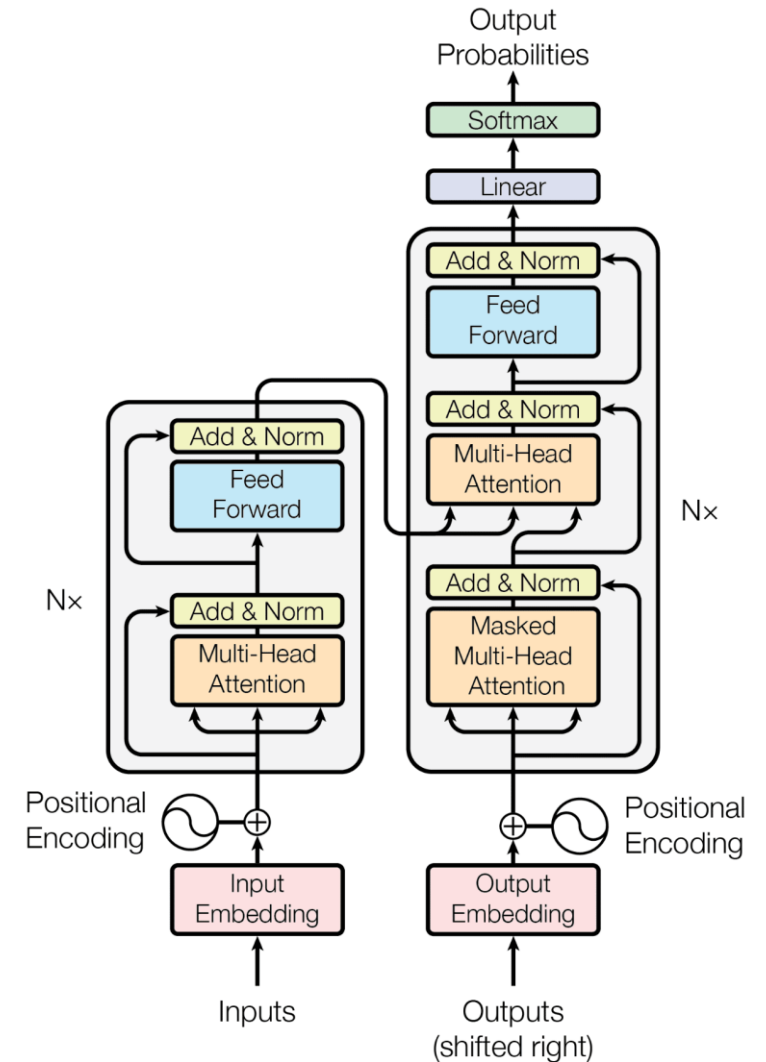
Puede generar:

- Texto (artículos, resúmenes, diálogos)
- Imágenes
- Video
- Audio
- Código de software
- Responde a instrucciones o preguntas escritas por el usuario (prompts).



¿Cómo funciona?

- Se basa en modelos de aprendizaje profundo (deep learning).
- Utiliza redes neuronales que simulan el funcionamiento del cerebro humano.
- Aprende a partir de grandes cantidades de datos para:
- Reconocer patrones y relaciones complejas
- Comprender lenguaje natural
- Generar respuestas relevantes y coherentes



¿Por qué es relevante ahora?

- Aunque la IA ha sido tendencia durante años, la IA generativa se volvió masiva desde 2022 con la aparición de ChatGPT.
- Ha desencadenado una ola de innovación sin precedentes en múltiples sectores.
- Está revolucionando:
 - La productividad personal
 - Los flujos de trabajo empresariales
 - La creación de productos inteligentes



Adopción empresarial

Según McKinsey:

- Un tercio de las organizaciones ya usa IA generativa regularmente en al menos una función empresarial.

Gartner proyecta:

- Para 2026, más del 80 % de las organizaciones habrán adoptado aplicaciones o APIs de IA generativa.





Beneficios y desafíos

Beneficios

- Ahorro de tiempo
- Automatización de tareas repetitivas
- Mejora de la creatividad y generación de ideas

Desafíos

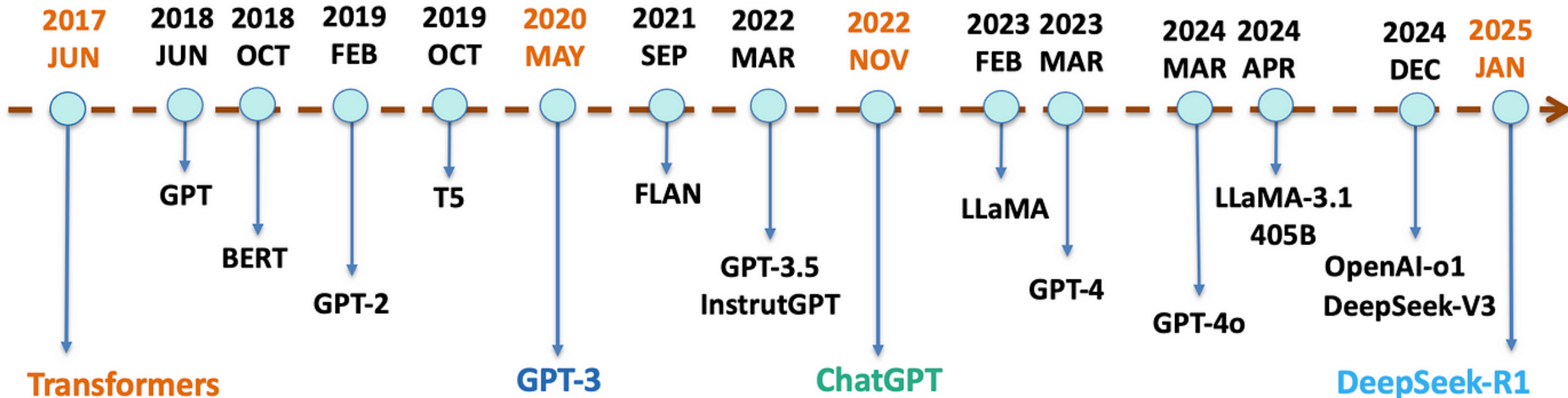
- Alucinaciones (respuestas erróneas)
- Seguridad de datos y privacidad
- Uso responsable y ético

¿Qué es un Modelo de Lenguaje?

- Un **modelo de lenguaje** es un sistema capaz de **procesar, comprender y generar texto** en lenguaje natural. Aprende patrones del lenguaje a partir de grandes cantidades de texto, y puede completar, responder o generar contenido nuevo con sentido.
- Predice la siguiente palabra en una oración (modelo autorregresivo), o completa fragmentos faltantes (modelo de máscara).
- Genera texto coherente, responde preguntas, resume, traduce, y más.
- Se entrena con enormes corpus de texto.
- Aprende representaciones estadísticas de palabras, frases y estructuras.
- Utiliza técnicas de aprendizaje profundo, en especial la arquitectura **Transformer**.



A Brief History of LLMs



LLMs: NEXT PUBLIC RELEASES IN 2025

ESTIMATES AS OF
FEBRUARY 2025

Jan-Mar

Apr-Jun

Jul-Sep

Oct-Dec

OpenAI
GPT-4.5 (Orion)

OpenAI
GPT-5 (GPT + o3)

OpenAI
o4

xAI
Grok-3

xAI
Grok-Video

xAI
Grok-4

Anthropic
Reasoning model

Anthropic
Claude 4

Meta AI
Llama 4

Meta AI
Llama 5

Google DeepMind
Gemini 3

Google DeepMind
Gemma 3

Microsoft
Phi-5

Baidu
ERNIE 5

Estimates only, selected highlights only. Full models table available at <https://lifearchitact.ai/models-table/> Alan D. Thompson, February 2025. <https://lifearchitact.ai/>



Tamaño del modelo:

Cantidad de parámetros entrenables (e.g., 7B, 13B, 175B). Afecta la capacidad de aprendizaje, pero también los recursos requeridos.

Corpus de entrenamiento:

Volumen y diversidad de datos usados para entrenar (libros, código, web). Influye en el conocimiento general y el estilo del modelo (The Pile, Common Crawl, Common Corpus, etc).

Arquitectura:

Tipo de estructura interna (Decoder-encoder, Decoder-only, Mixture of Experts, SSM,). Define cómo procesa la información y escala con el tamaño.

Tipo de entrenamiento:

- **Autoregresivo:** predice la siguiente palabra (GPT)
- **Enmascarado:** rellena huecos en frases (BERT)
- **RLHF:** ajustado con retroalimentación humana (ChatGPT)

Ventana de contexto:

Máximo número de tokens que puede procesar como entrada (e.g., 2048, 8192, 1M). Afecta su capacidad para mantener coherencia en entradas largas.

Multimodalidad:

¿Puede manejar solo texto o también imágenes, audio, video? (e.g., GPT-4V, Gemini)

Capacidades destacadas:

Habilidad para razonar, resumir, traducir, escribir código, seguir instrucciones complejas, etc.

Eficiencia y latencia:

Velocidad de respuesta y uso de recursos. Clave para aplicaciones en tiempo real o en dispositivos con recursos limitados.

Código abierto o cerrado:

¿El modelo está disponible para uso y modificación? Ejemplo: **LLaMA 3 (abierto)** vs **GPT-4 (cerrado)**.

Ajustes adicionales (fine-tuning):

¿Fue adaptado para tareas específicas o entrenado con retroalimentación humana?

Licencia de uso:

Define si el modelo puede ser usado comercialmente, solo para investigación o con restricciones.

Algunos LLM's de gran popularidad



Modelo	Organización	Parámetros	Ventana de Contexto	Multimodal	Código Abierto	Licencia	Precio por 1M tokens (Entrada/Salida)	Capacidades Destacadas
GPT-4o	OpenAI	~200B	128K tokens	Sí	No	Propietaria	\$2.50 / \$10.00	Texto, imagen, audio; razonamiento avanzado
Claude 4 Opus	Anthropic	~300B	200K tokens	Sí	No	Propietaria	\$15.00 / \$75.00	Razonamiento complejo, generación de código
Gemini 2.5 Pro	Google DeepMind	~540B	1M tokens	Sí	No	Propietaria	\$1.25 / \$10.00	Texto, imagen, audio, video; integración con Google Workspace
LLaMA 3 (70B)	Meta	~1.8T	8.2K tokens	No	Sí	Apache 2.0	\$0.73 / \$0.84	Generación de texto, código, multilingüe
Qwen 3 (32B)	Alibaba	32.8B	128K tokens	Sí	Sí	Apache 2.0	N/D	Multilingüe, razonamiento, código abierto
DeepSeek R1	DeepSeek	671B	131K tokens	No	Parcial	MIT	\$0.55 / \$2.19	Razonamiento, eficiencia, código abierto
Grok-3	xAI	~314B	128K tokens	Sí	No	Propietaria	\$3.00 / \$15.00	Razonamiento matemático y científico
Mistral Large 2	Mistral AI	123B	130K tokens	No	Parcial	Investigación / Comercial	\$2.00 / \$6.00	Generación de código, matemáticas, multilingüe
PaLM 2	Google	~340B	32K tokens	Sí	No	Propietaria	N/D	Multilingüe, multimodal, integración con productos de Google
Falcon 180B	TII	180B	4K tokens	No	Sí	Apache 2.0	N/D	Generación de texto, código abierto



Frameworks para ejecución local de LLM's

Son herramientas que permiten ejecutar modelos de lenguaje de gran tamaño directamente en dispositivos personales, sin necesidad de conexión a servicios en la nube.

Algunas muy populares son:

- **Ollama**
 - Interfaz por línea de comandos
- **LM Studio**
 - Interfaz gráfica (GUI)
- **GPT4All**
 - Interfaz gráfica y por comandos



Comandos habituales en Ollama



Comando	Función	Ejemplo de uso
ollama pull <modelo>	Descarga un modelo desde la librería pública sin ejecutarlo.	ollama pull llama3.2
ollama run <modelo>	Descarga (si no está local) y ejecuta el modelo en modo interactivo.	ollama run llama3.2
ollama list	Muestra los modelos descargados en tu PC.	ollama list
ollama show <modelo>	Muestra detalles meta del modelo (arquitectura, cuantización, etc.).	ollama show llama3.2
ollama ps	Muestra los modelos actualmente en ejecución.	ollama ps
ollama stop <modelo>	Detiene un modelo que esté corriendo en segundo plano.	ollama stop llama3.2
ollama rm <modelo>	Elimina un modelo descargado para liberar espacio.	ollama rm llama3.2
ollama cp <src> <dst>	Duplica un modelo descargado con un nuevo nombre o alias.	ollama cp llama3.2 my-copy
ollama serve	Inicia el servidor REST local (puerto 11434) para usar la API.	ollama serve
ollama --help	Permite ver otras opciones de ayuda	Ollama --help
ollama <comando> --help	Permite ver flags por cada comando	ollama show --help



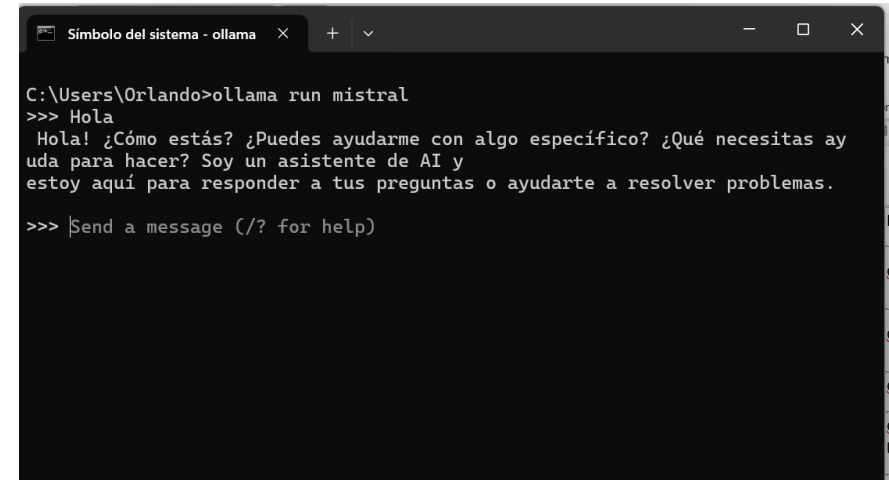
Comandos habituales de Ollama

Una vez en el modelo ejecutándose, algunos comandos útiles:

```
>> /info show // para ver información
>>/? // ver ayuda disponible
>>/? /help // ver ayuda para un comando
>>/bye //para salir
```

Para instalar Ollama:

<https://ollama.com/download>



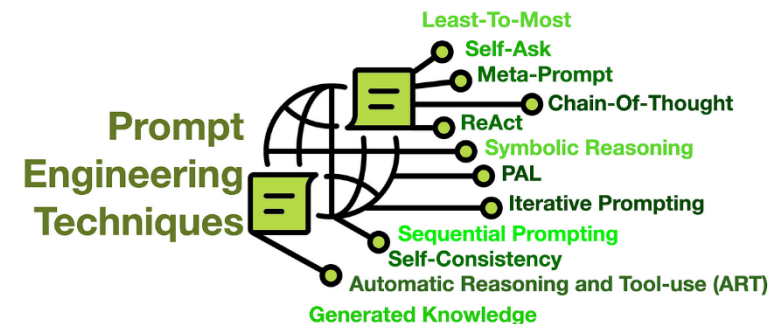
```
Símbolo del sistema - ollama
C:\Users\Orlando>ollama run mistral
>>> Hola
Hola! ¿Cómo estás? ¿Puedes ayudarme con algo específico? ¿Qué necesitas ayuda para hacer? Soy un asistente de AI y estoy aquí para responder a tus preguntas o ayudarte a resolver problemas.
>>> Send a message (/? for help)
```

¿Cómo damos indicaciones a un LLM?

¿Ingeniería de Prompts?

La **Ingeniería de Prompts** es el proceso de diseñar y refinar instrucciones (prompts) para guiar a modelos de inteligencia artificial generativa, como los modelos de lenguaje, hacia la producción de respuestas deseadas. Esto implica estructurar cuidadosamente el texto de entrada para optimizar la calidad y relevancia de las salidas generadas por el modelo.

12 Prompt Engineering Techniques



Tendencia
creciente en
investigación

Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing



Pengfei Liu

Carnegie Mellon University
pliu3@cs.cmu.edu

Weizhe Yuan

Carnegie Mellon University
weizhey@cs.cmu.edu

Jinlan Fu

National University of Singapore
jinlanjonna@gmail.com

Zhengbao Jiang

Carnegie Mellon University
zhengbaj@cs.cmu.edu

Hiroaki Hayashi

Carnegie Mellon University
hiroakih@cs.cmu.edu

Graham Neubig

Carnegie Mellon University
gneubig@cs.cmu.edu

Abstract

This paper surveys and organizes research works in a new paradigm in natural language processing, which we dub “prompt-based learning”. Unlike traditional supervised learning, which trains a model to take in an input x and predict an output y as $P(y|x)$, prompt-based learning is based on language models that model the probability of text directly. To use these models to perform prediction tasks, the original input x is modified using a *template* into a textual string *prompt* x' that has some unfilled slots, and then the language model is used to probabilistically fill the unfilled information to obtain a final string \hat{x} , from which the final output y can be derived. This framework is powerful and attractive for a number of reasons: it allows the language model to be *pre-trained* on massive amounts of raw text, and by defining a new prompting function the model is able to perform *few-shot* or even *zero-shot* learning, adapting to new scenarios with few or no labeled data. In this paper we introduce the basics of this promising paradigm, describe a unified set of mathematical notations that can cover a wide variety of existing work, and organize existing work along several dimensions, e.g. the choice of pre-trained models, prompts, and tuning strategies. To make the field more accessible to interested beginners, we not only make a systematic review of existing works and a highly structured typology of prompt-based concepts, but also release other resources, e.g., a website



Recomendaciones generales para un buen Prompt (1/2)

Claridad en la instrucción

- ✓ Formule el propósito del prompt de manera directa y sin ambigüedades.

Ejemplo: “Resume el siguiente texto en un párrafo de no más de 3 frases.”

Especificación del contexto o rol

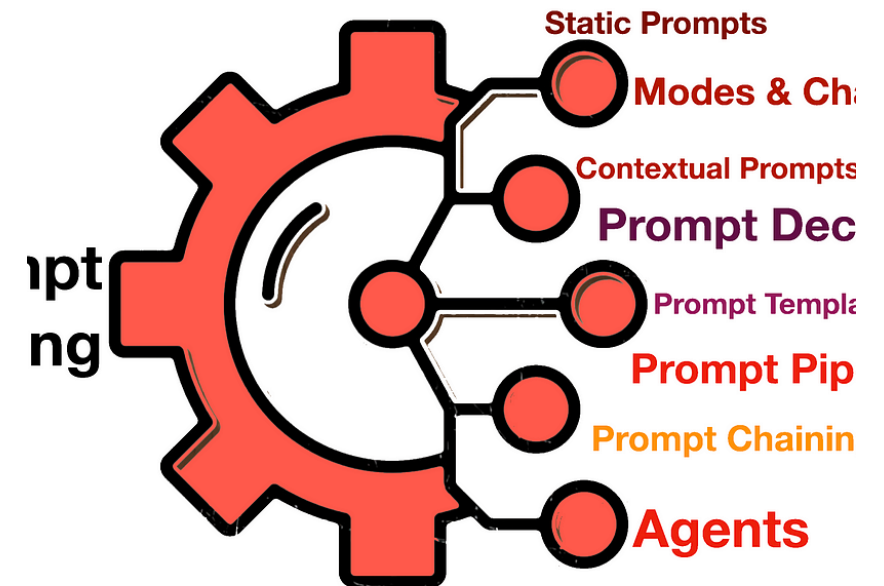
- ✓ Indique explícitamente el rol que debe asumir el modelo si es relevante para la tarea.

Ejemplo: “Actúa como un profesor universitario de filosofía y responde...”

Definición del formato de salida

- ✓ Determine el tipo de respuesta esperada: lista, tabla, párrafo, código, etc.

Ejemplo: “Devuelve los datos en formato de tabla con columnas: término, definición, ejemplo.”



Recomendaciones generales para un buen Prompt (2/2)

Incorporación de ejemplos (opcional pero recomendable)

- ✓ Cuando sea posible, proporcione ejemplos que ilustren el patrón deseado.

Ejemplo:

Entrada: “Madrid” → Salida: “España”

Entrada: “Lima” → Salida: ?

Descomposición en pasos

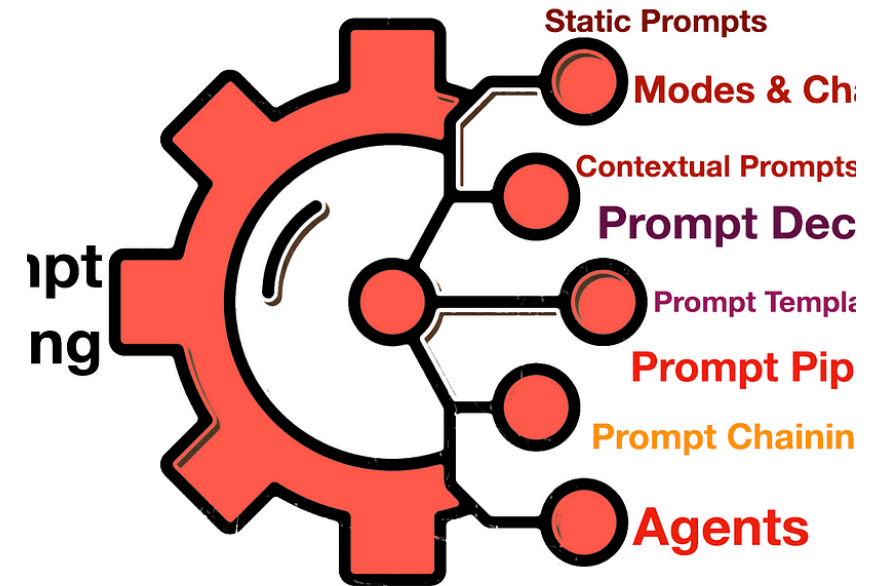
- ✓ Para tareas complejas, solicite razonamiento paso a paso antes de emitir una respuesta final.

Ejemplo: “Expón tu razonamiento paso a paso antes de llegar a una conclusión.”

Eliminación de ambigüedad

- ✓ Evite expresiones vagas o subjetivas. Prefiera términos medibles u observables.

“Escríbelo mejor” → “Reescribe el texto en un lenguaje más técnico y formal”



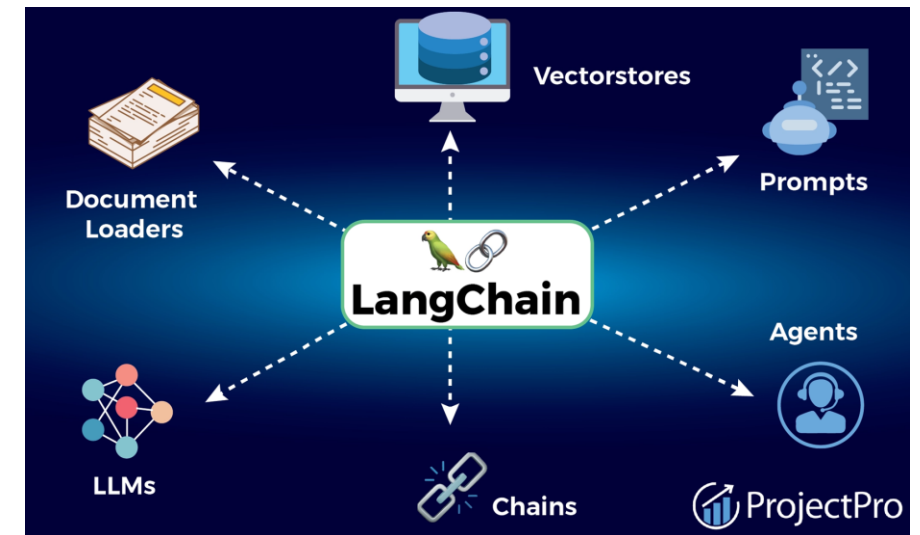
Herramientas clave para desarrollar aplicaciones con IA generativa



- **LangChain**
Framework para encadenar LLMs con lógica, herramientas y datos externos. Ideal para agentes conversacionales, búsqueda aumentada y automatización.
- **LlamaIndex**
Conecta modelos con tus propios datos (documentos, bases, APIs). Util para construir sistemas de búsqueda semántica y Q&A personalizados.
- **Haystack**
Enfocado en búsqueda y generación de respuestas (RAG). Permite pipelines modulares usando modelos locales o en la nube.
- **AutoGen**
Marco de agentes colaborativos impulsados por LLMs. Permite crear flujos donde varios agentes (o humanos) interactúan para resolver tareas complejas.
- **Hugging Face Transformers**
Biblioteca líder para acceder y desplegar modelos preentrenados. Soporta múltiples tareas: texto, imagen, código, traducción, entre otros.

¿Qué es LangChain?

- **Framework de código abierto** para desarrollar aplicaciones con modelos de lenguaje (LLMs), en Python y JavaScript.
- **Orquesta LLMs, bases de datos y APIs**, facilitando la creación de chatbots, agentes y sistemas generativos.
- **Modular y flexible**: permite combinar múltiples modelos y flujos sin reescribir el código.
- Funciona como **interfaz genérica para distintos LLMs**, integrándolos con datos y herramientas externas.



¿Qué es HuggingFace?

Plataforma líder en IA de código abierto, enfocada en modelos preentrenados, NLP y herramientas accesibles para desarrollo de aplicaciones de inteligencia artificial.

Beneficios clave

- **Acceso a modelos de última generación.** Miles de modelos listos para usar desde el Model Hub, para tareas como generación de texto, clasificación, resumen, QA, imágenes, etc.
- **Flujos de trabajo simplificados.** Bibliotecas fáciles de usar, bien documentadas. Permiten ajuste fino, tokenización, entrenamiento y evaluación con pocas líneas de código.
- **Implementación sencilla.** Herramientas para desplegar modelos en web, apps o sistemas internos, sin requerir conocimientos avanzados de infraestructura.
- **Comunidad activa y colaborativa.** Red global de desarrolladores y científicos compartiendo modelos, soluciones y buenas prácticas.
- **Compromiso con una IA responsable.** Modelos documentados con advertencias sobre sesgos y usos adecuados; impulsan debates éticos y gobernanza abierta.



Hugging Face



“Formando nuevas generaciones con sello de excelencia comprometidos
con la transformación social de las regiones y un país en paz”