

Glosario

01	Los cimientos	
	Modelo	2
	Entrenamiento	2
	Inferencia	2
	LLM	3
	Multimodal	3
	Benchmark	3
02	El idioma de la máquina	
	Token	4
	Prompt	4
	Ventana de contexto	4
	Memoria	5
	Razonamiento	5
03	El error y su remedio	
	Alucinación	6
	RAG	6
04	Del modelo al producto	
	Chatbot	7
	Herramienta	7
	Agente	8
	Skill	8
	Harness	8
05	Quién pone los límites	
	Human-in-the-loop	9
	Guardrails	9
	Cómo encaja todo	10

01 Modelo

el archivo que "sabe" cosas

Un modelo es un archivo enorme de números que aprendió patrones a partir de muchísimos ejemplos. No es un programa con reglas escritas por alguien ("si el cliente dice X, responde Y"); se parece más a una intuición estadística congelada en un archivo.

Una plantilla de Excel tiene fórmulas que alguien escribió a mano. Un modelo es más como un empleado que vio 10 millones de facturas y ya "le suena" cómo se ve una factura, sin que nadie le haya dictado la regla.

Cuando alguien dice "cambié de modelo" está diciendo "cambié de cerebro", no "cambié de programa". Otro modelo significa otras respuestas, otro costo y otra velocidad.

02 Entrenamiento

cómo se fabrica el modelo

Es el proceso, carísimo y lento, en el que el modelo lee cantidades gigantescas de texto y va ajustando sus números hasta que predice bien. Toma meses, cuesta millones de dólares y ocupa centros de datos completos. Lo hacen las grandes empresas de IA, una sola vez, antes de que tú lo uses.

Cuando abres el chat, el entrenamiento ya terminó. El modelo no aprende de ti mientras conversan.

De ahí sale la fecha de corte de conocimiento: si el entrenamiento terminó en cierto mes, el modelo no sabe nada de lo que pasó después, no porque sea tonto, sino porque no estaba en los ejemplos que leyó. Por lo mismo tampoco recuerda tu conversación de ayer, salvo que el producto que uses guarde notas aparte.

03 Inferencia

cómo se usa el modelo

Es lo que pasa cada vez que escribes algo y el modelo responde: toma tu texto y va prediciendo la continuación más probable, pedacito por pedacito. Eso es todo lo que hace.

Entrenar es fabricar el cerebro, una vez; inferir es usarlo, millones de veces. Entrenar es escribir el manual de la empresa e inferir es consultarlo, y consultarlo no lo reescribe.

Cada inferencia se cobra y consume tiempo. Cuando un proceso automatizado se pone caro o va lento, casi siempre es porque hace más llamadas al modelo de las necesarias, no porque el modelo esté roto.

04 LLM

el modelo que trabaja con lenguaje

Large Language Model, o modelo grande de lenguaje. Es un modelo entrenado específicamente para trabajar con texto: continuar, resumir, traducir, clasificar, redactar, responder. ChatGPT, Claude y Gemini son productos contruidos sobre un LLM.

Es un predictor de texto extremadamente bueno. No consulta una base de datos ni busca la respuesta en ninguna parte: la redacta. Eso explica tanto sus aciertos como sus errores.

Por eso es excelente en tareas de lenguaje —redactar, reordenar, extraer, comparar, explicar— y malo en las de exactitud dura: aritmética larga, cifras precisas de memoria, fechas. Para esas se le conectan herramientas: una calculadora, tu base de datos, tu Excel.

05 Multimodal

cuando además del texto entiende imágenes y audio

Un modelo multimodal no solo lee texto: también recibe fotos, capturas de pantalla, PDF escaneados, audio y a veces video, y responde sobre eso.

"Modal" aquí significa tipo de entrada: texto, imagen, sonido.

En una oficina sirve para convertir la foto de una factura en una tabla, leer un plano o un formato escaneado, señalar qué está mal en la captura de un error de Excel, o transcribir y resumir una reunión. La captura de pantalla es un atajo enorme: muchas veces describir el problema con palabras cuesta más trabajo que pegar la imagen y preguntar qué está mal ahí.

Ver no es leer bien. Una foto torcida, un sello encima de la cifra o una tabla mal escaneada le dejan un hueco de información, y el hueco lo rellena con lo plausible. Para los dígitos que importan —montos, NIT, números de guía— verifica siempre.

06 Benchmark

el examen estandarizado con que se comparan los modelos

Una prueba fija —miles de preguntas de matemáticas, código, comprensión de lectura— que se le pasa a varios modelos para publicar puntajes comparables. De ahí salen las gráficas de barras de los anuncios.

Sirve para descartar: un modelo que aparece muy por debajo en todo probablemente no te sirva. Para elegir entre los que quedan sirve poco, por dos razones. Dos modelos con puntajes casi iguales pueden sentirse muy distintos en tu trabajo real. Y el examen es público, así que los fabricantes optimizan para él, que es estudiar el simulacro en vez de la materia.

El benchmark que decide es el tuyo: junta 20 casos reales de tu operación de los que ya conoces la respuesta correcta, pásalos por los dos candidatos y compara. Media hora de eso vale más que todas las gráficas del lanzamiento.

02 · El idioma de la máquina

07 Token

la unidad en que se mide todo

El modelo no lee letras ni palabras completas: lee **tokens**, que son fragmentos de palabra. "Facturación" puede ser 3 o 4 tokens. Como referencia rápida en español, un token equivale a 3 o 4 caracteres, y una página de texto son unos 500 a 700 tokens.

El token es la unidad de cobro y la unidad de límite. Todo lo que entra —tu pregunta, el documento que pegas, las instrucciones del sistema— y todo lo que sale se cuenta en tokens.

Pegar un PDF de 80 páginas en un chat no es un mensaje: son unos 50.000 tokens. Ahí está la diferencia entre un proceso que cuesta centavos y uno que cuesta cientos de dólares al mes.

08 Prompt

lo que le escribes

Es la instrucción completa que recibe el modelo: tu pregunta, más el contexto que le das, más el ejemplo que le pegas, más el formato que le pides. No es una búsqueda; se parece más al *brief* que le pasas a un proveedor.

Un buen prompt tiene cuatro cosas:

- **Rol** — quién es él para esta tarea ("eres analista comercial").
- **Tarea** — qué quieres, exactamente.
- **Insumos** — los datos, el documento, el ejemplo.
- **Formato** — tabla, viñetas, 200 palabras, tono.

Pedir "hazme un informe de ventas" deja al modelo sin tus ventas y sin saber para quién es, así que redacta un informe genérico. Con las cuatro cosas queda así: *"Eres analista comercial. Con esta tabla que te pego, escribe 5 viñetas para el gerente regional destacando las 3 caídas más grandes frente al mes anterior. Sin recomendaciones."*

09 Ventana de contexto

su memoria de trabajo

Es la cantidad máxima de tokens que el modelo puede tener a la vista al mismo tiempo: tus instrucciones, los documentos pegados, todo lo dicho antes en la conversación y la respuesta que está escribiendo.

Es un escritorio, no un archivador, y cuando se llena algo tiene que salir. Es también una ventana y no un disco duro: al cerrar el chat se borra. Cuando una conversación muy larga se acerca al límite, el inicio empieza a desdibujarse, y de ahí viene la sensación de que se le olvidó lo que le dijiste al principio.

Para una tarea nueva conviene abrir una conversación nueva. Arrastrar 200 mensajes de otro tema no le da más contexto útil: le da más ruido y más costo. Y cuando pegues un documento largo, dile explícitamente qué parte importa.

10 Memoria

las notas que sobreviven al cierre del chat

La ventana de contexto se borra al cerrar, pero tu asistente te muestra un aviso de "memoria actualizada". La memoria no es del modelo: es una función del producto.

El producto guarda aparte unas notas sobre ti —a qué te dedicas, cómo prefieres las respuestas, en qué proyecto andas— y la próxima vez te las vuelve a pegar en el prompt sin que las veas. El modelo no recordó nada: alguien le pasó el resumen antes de que empezara. Es el asistente que le deja un post-it al jefe antes de la reunión, "el cliente se llama Pardo, odia las llamadas los viernes"; el jefe no se acordaba, leyó el post-it.

Conviene revisarla cada tanto, porque una nota vieja o mal tomada contamina tus respuestas durante meses: sigue creyendo que estás en el cargo anterior. Se puede editar y borrar, entrada por entrada, en la configuración del producto. Y lo que entra a la memoria queda guardado en el servicio, no dentro de una conversación que tú cierras, así que cuida lo confidencial.

11 Razonamiento

cuando el modelo piensa antes de contestar

Algunos modelos, antes de darte la respuesta, escriben para sí mismos un borrador de trabajo: desglosan el problema, prueban un camino, se corrigen. Eso es el razonamiento, que verás anunciado como *pensar*, *thinking* o *razonamiento extendido*. Sigue siendo predicción de texto, solo que primero se habla a sí mismo y después te habla a ti; es la diferencia entre contestar de memoria en plena reunión y salir un minuto a hacer la cuenta en una hoja aparte.

Vale la pena en problemas de varios pasos donde el primer impulso suele fallar: cuadrar cifras que no coinciden, comparar tres condiciones de un contrato, planear un proceso, encontrar por qué una fórmula da error. Reformular un correo, resumir, cambiar el tono o traducir no lo necesitan, y ahí pensar de más solo cuesta tiempo y dinero, porque ese borrador son tokens que se cobran y ocupan lugar en la ventana de contexto aunque casi nunca los veas completos.

Un razonamiento largo y ordenado no garantiza que la respuesta sea correcta. El modelo puede razonar impecablemente hacia una conclusión falsa, y lo que te muestra no es necesariamente el registro fiel de cómo llegó ahí. Sirve para revisar el argumento, no como prueba.

03 · El error y su remedio

12 Alucinación

cuando inventa con total seguridad

Es cuando el modelo produce algo falso pero perfectamente verosímil: una cifra, una fecha, un artículo de ley, un nombre de archivo, una función de Excel que no existe.

No está mintiendo ni fallando. Está haciendo exactamente lo que hace siempre, predecir el texto más plausible, y cuando no tiene el dato, lo plausible ocupa el lugar del dato. El tono de confianza es idéntico cuando acierta y cuando inventa; no hay temblor en la voz. Por eso las alucinaciones peligrosas son las que suenan aburridas y específicas: cifras exactas, citas textuales, normativa, precios, referencias bibliográficas, rutas y nombres de sistemas internos que el modelo nunca vio.

Se controla, en orden de eficacia:

1. Dale el dato en el prompt —pega el documento— en vez de pedirle que lo recuerde.
2. Pídele que cite de dónde sacó cada cifra, dentro del material que le diste.
3. Autorízalo a no saber: "si no está en el documento, escribe *no encontrado*". Suena trivial y baja los inventos de forma notoria.
4. Verifica lo verificable siempre que la respuesta vaya a tener consecuencias.

13 RAG

darle el dato en vez de exigirle que se lo sepa

Retrieval-Augmented Generation, o generación aumentada por recuperación. Son dos pasos: buscar en tus fuentes lo que aplica a la pregunta —tus documentos, tu base de datos, tu intranet— en el momento mismo de responder, y pegar lo encontrado en el prompt, para que el modelo redacte apoyado en ese material y no en su memoria. No es enseñarle el manual de la empresa: es pasarle el manual abierto en la página correcta, medio segundo antes de que conteste.

Resuelve de un golpe las dos carencias más molestas del modelo, que no conoce tus datos internos y que no sabe nada posterior a su fecha de corte, sin tocar el modelo.

Lo que define a RAG es el momento: buscar mientras se responde y pegar lo encontrado. La tecnología de búsqueda es lo de menos. Una base de datos vectorial con búsqueda semántica es la forma más común de hacerlo, pero un buscador por palabra clave en SharePoint, una consulta a la base de datos de la compañía o una revisión de una carpeta de archivos hacen exactamente lo mismo.

Se arma de dos maneras:

- **Tubería fija:** llega la pregunta, el sistema busca siempre, pega los mejores fragmentos y genera. El modelo no decidió nada; ni se enteró de que hubo una búsqueda. Predecible y barato.
- **Agéntica:** buscar es una herramienta que el agente usa cuando quiere. Puede consultar, ver que le falta el año fiscal, reformular y volver a consultar, o cruzar dos fuentes. Más potente, menos predecible.

El dato queda prestado, no aprendido: vive en la ventana de contexto y se evapora al cerrar la conversación. Modificar el modelo es otra cosa, un reentrenamiento parcial que se llama *fine-tuning*, y sale carísimo pagar por entrenar la IA con el catálogo de la empresa cuando lo que hacía falta era poder buscar en el catálogo.

El cuello de botella nunca es el modelo, son tus documentos. Si hay tres versiones del mismo procedimiento y ninguna dice cuál rige, RAG amplifica el desorden y te citará la versión vieja con total seguridad. Un buen RAG empieza por ordenar la carpeta.

04 · Del modelo al producto

14 Chatbot

la forma más simple de usarlo

Una interfaz de conversación sobre un modelo: tú escribes, él responde, tú corriges, él corrige. Cada turno depende de que tú lo pidas, y el resultado es texto que después tú usas.

No ejecuta nada. Te escribe la fórmula, pero tú la pegas en el Excel. Te redacta el correo, pero tú lo envías. El trabajo de brazos sigue siendo tuyo.

Es justo lo que necesitas para redactar, reformular, resumir, entender un tema, preparar un borrador o pensar en voz alta, que es la mayoría de lo que se hace en una oficina.

15 Herramienta

las manos que el modelo no tiene

Una herramienta es una capacidad concreta que alguien le habilitó al modelo para tocar el mundo real: buscar en la web, leer una carpeta, escribir un archivo, consultar una base de datos, hacer una cuenta exacta, enviar un correo.

El modelo no ejecuta nada. Lo único que hace es pedirlo por escrito, algo así como *"ejecuta buscar_factura con número 4471"*. Otro programa lo ejecuta de verdad y le devuelve el resultado como texto, que el modelo lee y usa para seguir. El modelo pide; el programa de alrededor tiene las manos. Si la IA no ve tus archivos es porque nadie le habilitó esa herramienta: no le falta

inteligencia, le faltan manos.

Las herramientas son también donde se ponen los frenos: se puede dar lectura y negar escritura, o exigir tu confirmación antes de enviar o borrar.

Una herramienta bien puesta arregla justo lo que el modelo hace mal por naturaleza. No sabe tus cifras: dale acceso a la base. Se equivoca en aritmética larga: dale la calculadora. No conoce lo de esta semana: dale la búsqueda.

16 Agente

cuando además actúa

Un modelo al que le diste **herramientas** y un objetivo, y que trabaja en ciclo: piensa, actúa, mira el resultado, corrige y sigue, hasta terminar. Al chatbot le pides una respuesta; al agente le encargas un resultado.

Pídele a un chatbot "explícame cómo consolidar estos 12 reportes" y te llega la explicación. Encárgale a un agente "toma los 12 archivos de esta carpeta, consolídalos, márcame las filas duplicadas y déjame el resultado en un Excel nuevo" y te llega el archivo.

Un agente equivocado no entrega una respuesta equivocada: ejecuta acciones equivocadas. Por eso se empieza con permisos de solo lectura, se prueba sobre copias, y las acciones irreversibles —borrar, enviar, pagar— piden confirmación.

17 Skill

el instructivo que le dejas guardado

Un paquete reutilizable de instrucciones y recursos para una tarea concreta, que el agente carga solo cuando hace falta: así se cierra el mes, así se formatea un informe para este cliente, estas son nuestras reglas de nomenclatura, con sus plantillas y ejemplos adjuntos.

Es el procedimiento escrito que le dejas a alguien que entra nuevo. No lo hace más inteligente; hace que haga las cosas como se hacen aquí, sin que tengas que explicarlo cada vez.

Así el conocimiento de tu operación deja de vivir en el prompt que reescribes de memoria cada mañana y pasa a estar escrito, con versiones y compartido con el equipo.

18 Harness

el arnés: toda la maquinaria alrededor del modelo

Es todo lo que rodea al modelo para que sea útil: qué herramientas tiene disponibles, con qué permisos, qué instrucciones fijas recibe siempre, qué se le manda en cada turno, qué pasa cuando algo falla, cuándo tiene que pedirte permiso, cuándo se corta.

El modelo es el motor y el harness es el resto del vehículo: chasis, frenos,

tablero y cinturón. Un motor excelente sin frenos no es un buen carro.

Entre modelos parecidos, la diferencia entre una herramienta de IA que sirve y una que frustra está casi siempre aquí. Si tu asistente no ve tus archivos, no recuerda las reglas de la empresa o hace algo peligroso sin avisar, lo que hay que arreglar es el harness.

05 · Quién pone los límites

19 Human-in-the-loop

el humano en el circuito, justo donde importa

Diseñar el proceso para que una persona apruebe antes de ciertos pasos, en vez de dejar que el agente corra de punta a punta. Es poner el freno exactamente donde el error sería caro o irreversible.

El asunto no es cuánto revisas, es dónde. Si hay que revisar todo, en dos semanas nadie revisa nada y el proceso se abandona. La palanca está en clasificar las acciones:

- **Reversibles** —generar un borrador, producir un reporte, proponer una clasificación— van solas.
- **Irreversibles o hacia afuera** —enviar al cliente, borrar, pagar, publicar, escribir en el sistema de producción— piden firma.

En cualquier reunión de automatización, la pregunta que ordena todo es qué es lo peor que esto puede hacer sin que nadie lo apruebe. Si la respuesta incomoda, ahí va el humano.

20 Guardrails

las barandas: los límites que no dependen de su buena voluntad

Reglas y controles duros, puestos por fuera del modelo, que acotan lo que puede llegar a pasar: temas que no toca, datos que no puede sacar, formatos que debe cumplir, montos que no puede superar, sistemas a los que simplemente no tiene acceso.

Escribir en el prompt "no reveles precios de costo" es una instrucción, y una instrucción se puede olvidar, malinterpretar o rodear con una pregunta astuta. Un guardrail no se pide: se impone. Si el sistema no tiene acceso a la tabla de costos, no hay pregunta astuta que la saque. "Por favor no hagas X" es un deseo y va en el prompt; "no puedes hacer X" es un guardrail y va en el harness.

Hay dos capas, de menos a más confiable. Los filtros de entrada y salida revisan o bloquean lo que entra y lo que sale. Los permisos hacen que la herramienta ni exista para ese agente, y son siempre más confiables, porque no hay nada que burlar.

Cómo encaja todo

Un modelo es un archivo de números que salió de un entrenamiento carísimo y ya terminado. Cada vez que lo usas hace inferencia: predecir texto, pedazo por pedazo. Si trabaja con lenguaje es un LLM; si además recibe imágenes y audio es multimodal; y los puntajes con que te lo venden salen de algún benchmark. Todo lo que entra y sale se mide en tokens; lo que le escribes es el prompt; y lo que puede tener a la vista de una sola vez es la ventana de contexto, que se borra al cerrar salvo las notas que el producto guarde como memoria. Algunos modelos escriben un borrador para sí mismos antes de contestarte: eso es razonamiento. Cuando no tiene un dato no se calla — lo rellena con lo plausible, y eso es una alucinación; la forma seria de evitarlo es RAG: buscarle el dato en tus fuentes y pegárselo en el prompt. Encima de todo esto se construyen productos. El chatbot solo conversa; dale herramientas y un objetivo y tienes un agente; escríbele tus procedimientos y tienes una skill; y toda la maquinaria que lo rodea es el harness. Ahí mismo viven los dos controles que separan un experimento de algo que puedes poner a trabajar: el human-in-the-loop, que exige tu firma donde el error sería caro, y los guardrails, que impiden por diseño lo que no debe pasar nunca.