MAPAS DEL LABERINTO

UNA SERIE DE PHANTOM MAZE — AI & LANGUAGE LAB

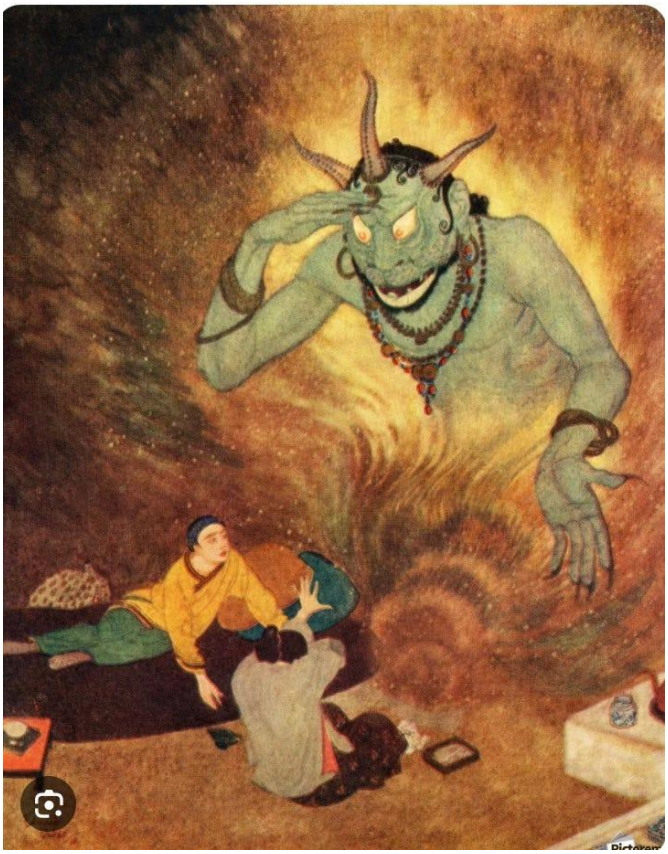
ENSAYO 16

Sobre pesar memoria y guardar genios en el escritorio

Claudia Marsico · Hernán Inverso

DOI 10.5281/zenodo.23575282

Cómo los parámetros y pesos de un gran modelo de lenguaje nos llevan a las puertas de los sueños de inmortalidad



E. Dulac, Aladdin et l'efrit (1914)

La ciencia ficción exploró las fantasías de inmortalidad a través de mentes desprendidas del cuerpo, como el científico de *Transcendence* (2014) o la muerte en un mundo virtual en el episodio *San Junipero* de *Black Mirror* (2016). Aunque, por supuesto, los modelos de lenguaje no son conciencias copiadas, activan una versión modesta de esa fantasía al guardar en un disco casi todo lo que una inteligencia artificial aprendió durante su entrenamiento, repartido en archivos que contienen

números. Esos números son los parámetros del modelo. Antes de entrenarse, una red neuronal ya tiene definida su estructura, pero contiene una enorme cantidad de valores todavía por ajustar. El entrenamiento consiste precisamente en encontrar valores útiles para esos lugares. La mayoría de los parámetros son pesos, es decir, parámetros que determinan con qué fuerza una cantidad que circula por la red interviene en el cálculo siguiente. Otros parámetros cumplen funciones distintas. Así, "parámetro" nombra en general los valores que el modelo aprende y conserva; "peso" nombra la clase más numerosa de esos valores, los que regulan cuánto cuenta una contribución frente a otras.

Supongamos que una capa produce varios números y la siguiente debe combinarlos. Antes de hacerlo, cada uno se multiplica por un peso. Si un valor de 0,8 tiene un peso de 2, pasa a contribuir 1,6; con un peso de 0,1, apenas cuenta; con uno negativo, empuja el resultado en sentido contrario. Un peso regula cuánto influye una cantidad en el cálculo siguiente. Repetida millones de veces a lo largo de la red, esa operación ayuda a determinar qué relaciones se refuerzan, cuáles se atenúan y, en conjunto, cómo se comporta el modelo.

E. Dulac, Aladdin et l'efrit (1914)

El peso, que trae a la mente la idea de balanza, tiene asociada la decisión. En la *Ilíada*, en los momentos clave, Zeus saca una balanza de oro y coloca en sus platillos las suertes de muerte de los héroes, como cuando el platillo de Héctor desciende hacia el Hades y queda sellado que va a morir. En lugar de una sola balanza áurea, un modelo de lenguaje tiene cientos de miles de millones de valores que intervienen en operaciones sucesivas, haciendo que unos resultados influyan más y otros menos. GPT-3 tenía ciento setenta y cinco mil millones de parámetros. Llama 3.1 apareció en versiones de 8B, 70B y 405B. La B es la *billion* inglesa, mil millones. Un modelo de 70B tiene alrededor de setenta mil millones de valores aprendidos. Si pudiéramos escribir uno por segundo, sin parar nunca, anotar los ciento setenta y cinco mil millones de GPT-3 nos llevaría más de cinco mil quinientos años.

A veces se comparan los pesos con perillas. La comparación no es metafórica. En 1958 Frank Rosenblatt presentó el perceptrón, uno de los antepasados conceptuales de las redes neuronales actuales. El Mark I Perceptron construido después en Cornell recibía imágenes mediante una retina de 20 por 20 fotocélulas y tenía 512 pesos representados mediante componentes eléctricos regulables con motores que modificaban esos valores durante el aprendizaje. Los números que hoy viven en archivos eran ajustes físicos dentro de una máquina y aprender producía muy literalmente movimiento. La intuición de fabricar algo parecido al razonamiento mediante muchas piezas ajustables es todavía más antigua. Hacia 1300 Ramon Llull imaginó su *Ars* como una máquina de discos concéntricos que podían girarse para generar distintas combinaciones de conceptos según el viejo sueño de mecanizar operaciones intelectuales a partir de elementos simples de disposición variable. Lo cierto es que puede surgir algo enormemente complejo al combinar ajustes simples.

El modelo terminado puede transportarse, en buena medida, como números. Para hacer funcionar los pesos necesitamos saber qué arquitectura los organiza, cómo convertir el texto en tokens y qué software ejecuta las operaciones, pero los pesos son centrales y por eso importa si se pueden descargar o no. Meta, por ejemplo, permite descargar los pesos de modelos Llama como Llama 3.1 y Llama 4;

Mistral hace lo mismo con varios de sus modelos, entre ellos Mistral Large 3. Con el hardware adecuado, esos archivos pueden ejecutarse localmente, modificarse, cuantizarse o estudiarse. En el extremo opuesto, empresas como Anthropic mantienen los pesos de Claude en sus propios servidores: el usuario puede enviar texto y recibir respuestas, pero no obtiene la configuración aprendida para llevársela y ejecutarla por su cuenta. Entre ambos extremos hay además casos mixtos. OpenAI ofrece modelos cerrados a través de ChatGPT y su API, pero también publica modelos de pesos abiertos como gpt-oss; Google hace algo parecido con Gemini, que permanece cerrado, y Gemma, cuyos pesos pueden descargarse. Así, "tener acceso a un modelo" puede significar cosas bastante distintas: conversar con él a distancia, llamar a una API o disponer efectivamente de sus pesos y hacerlos funcionar en una máquina propia.

Tener los pesos no es lo mismo que poder leerlos. La información está distribuida. Ninguna capacidad ni ninguna asociación vive en un valor en particular, sino en patrones repartidos entre enormes cantidades de valores que participan a la vez en cosas distintas. Podemos abrir los archivos, contarlos, leerlos o imprimirlos sin encontrar ninguna línea que diga qué sabe el modelo. En algunos aspectos es semejante a un archivo ejecutable. También un programa compilado es un binario ilegible, también necesita un entorno para funcionar y también hay que ejecutarlo para saber qué hace, pero la analogía se rompe pronto, porque un ejecutable tiene código fuente escrito por alguien, cada parte tiene un nombre y una intención declarada y con la fuente a mano o paciencia suficiente se puede entender qué hace la máquina y por qué. Los pesos no tienen fuente. No existe una versión más explícita de la que hayan sido compilados, ni una lógica de redacción. Solo surgieron de un proceso de optimización y eso no se puede descompilar.

Ahora bien, cada parámetro cuesta espacio y puede escribirse con una cierta cantidad de bits, del mismo modo que un mismo valor puede anotarse con muchos decimales o redondearse. Durante el entrenamiento suele trabajarse con representaciones más precisas porque hay que registrar cambios pequeños. Una vez que el modelo aprendió, la precisión puede resultar excesiva y por eso se recurre a la cuantización, que es el procedimiento de tomar los pesos ya entrenados y volver a escribirlos con menos bits cada uno, 8 o 4 bits, para disminuir la memoria ocupada con pérdida mínima de exactitud.

Cuantizar es como pesar con una balanza menos fina. Para la mayoría de los usos la diferencia no se nota. La cuestión es cuánto puede perderse de esa precisión sin alterar de manera apreciable el comportamiento del conjunto. Los métodos actuales buscan precisamente hacer ese redondeo de manera inteligente, porque algunos valores toleran mejor la aproximación que otros. Pasar de 16 a 4 bits reduce, en principio, a una cuarta parte la memoria necesaria para almacenar los pesos. La escala física cambió en muy poco tiempo. Cuando Meta entrenó Llama 3.1 405B en 2024 necesitó más de dieciséis mil GPU H100 y para ejecutar el modelo en producción redujo sus pesos de 16 a 8 bits para hacerlo caber en un nodo de servidor. Dos años después, esa clase de modelo cabe en un escritorio.

Hasta hace pocos años solo había modelos "densos", donde prácticamente toda la red neuronal participa en cada cálculo. La mezcla de expertos (MoE) cambia esa lógica y en lugar de una única red gigantesca, el modelo se organiza en muchas subredes más chicas, los "expertos", que son porciones de

la misma maquinaria, y el enrutador, que determina en cada caso cuáles de esos expertos conviene activar. Todos los expertos pertenecen al modelo y necesitamos todos sus parámetros, pero para procesar un elemento determinado solo se encienden algunos.

En la práctica, el "tamaño del modelo" dejó de significar, sin aclaración, "cuánto cuesta cada cálculo". Llama 4 Maverick, presentado por Meta en 2025, tiene alrededor de 400B totales, casi lo mismo que Llama 3.1 405B, un modelo denso, repartidos entre ciento veintiocho expertos, pero solo activa unos 17B por cálculo. DeepSeek-V3 va más lejos con 671B de los que se activan alrededor de 37B por cálculo. La idea de dividir el trabajo entre expertos es mucho más antigua, pero su versión moderna para grandes modelos de lenguaje tomó impulso con las MoE escasamente activadas de 2017 y con Switch Transformer, presentado en 2021, que mostró hasta dónde podía escalar el procedimiento sin activar todos los parámetros a la vez.

Cuantización y mezcla de expertos resuelven problemas distintos de economía de memoria y economía de cómputo y juntas cambiaron cuánto modelo entra en qué máquina. La NVIDIA DGX Spark, la computadora de escritorio para IA que la empresa empezó a vender a fines de 2025, tiene 128 GB de memoria unificada y se anuncia capaz de correr modelos de hasta 200B en una sola unidad, o hasta 405B enlazando dos. Eso solo es posible con cuantización de por medio, ya que 200B a 16 bits cada uno requerirían más de tres veces la memoria de una Spark.

Tenemos una máquina que puede aprender durante meses en un centro de cómputo gigantesco, guardar lo aprendido en números que caben en un disco, abandonar la máquina que lo produjo y volver a funcionar en otra sin que, al abrir los archivos, podamos leer qué sabe. ¿Cómo no pensar en los genios que salen de lámparas de aceite y vasijas en Las mil y una noches? Hay algo de esa desproporción en el salto reciente de la IA. Modelos cuya ejecución pertenecía hasta hace poquísimo al paisaje de los centros de datos ahora se despliegan en una o dos cajitas en nuestro escritorio. En los relatos orientales el poder de la criatura desmesurada se ponía en marcha hablando. Bastaba invocarla y formular un pedido. Después de tanta matemática, volvimos a la antigua idea de una potencia encerrada en un objeto que se despierta con palabras.

◊

CÓMO CITAR

Marsico, Claudia y Hernán Inverso. 2026. "Sobre pesar memoria y guardar genios en el escritorio". *Mapas del laberinto*, ensayo 16. Phantom Maze — AI & Language Lab. https://doi.org/10.5281/zenodo.23575282

Mapas del laberinto es una serie de Phantom Maze, AI & Language Lab. La colección vive en **phantommaze.cloud**; los ensayos aparecen también en phantommazeailabes.substack.com. La suscripción es gratuita.

Algunas de las ideas que enmarcan este texto se desarrollan en *El fantasma sin ego. Un mapa en el laberinto de la inteligencia artificial*, de Hernán Inverso y Claudia Marsico (Phantom Maze Press, 2026; ISBN 978-987-3729-10-2).

Fuentes. Llama Team, “The Llama 3 Herd of Models”, arXiv:2407.21783, 2024, y repositorio oficial de Meta `meta-llama/Meta-Llama-3.1-405B`; Meta AI, “The Llama 4 Herd: The Beginning of a New Era of Natively Multimodal AI Innovation”, 5 de abril de 2025; DeepSeek-AI, “DeepSeek-V3 Technical Report”, arXiv:2412.19437, 2024; Ian Goodfellow, Yoshua Bengio y Aaron Courville, *Deep Learning*, Cambridge, MA, MIT Press, 2016, especialmente caps. 6 y 8; Frank Rosenblatt, “The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain”, *Psychological Review* 65, n.º 6 (1958), pp. 386–408; Geoffrey E. Hinton, James L. McClelland y David E. Rumelhart, “Distributed Representations”, en David E. Rumelhart y James L. McClelland, eds., *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1: *Foundations*, Cambridge, MA, MIT Press, 1986, pp. 77–109; Tim Dettmers, Mike Lewis, Younes Belkada y Luke Zettlemoyer, “LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale”, *Advances in Neural Information Processing Systems* 35 (NeurIPS 2022), pp. 30318–30332; William Fedus, Barret Zoph y Noam Shazeer, “Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity”, *Journal of Machine Learning Research* 23, n.º 120 (2022), pp. 1–39.