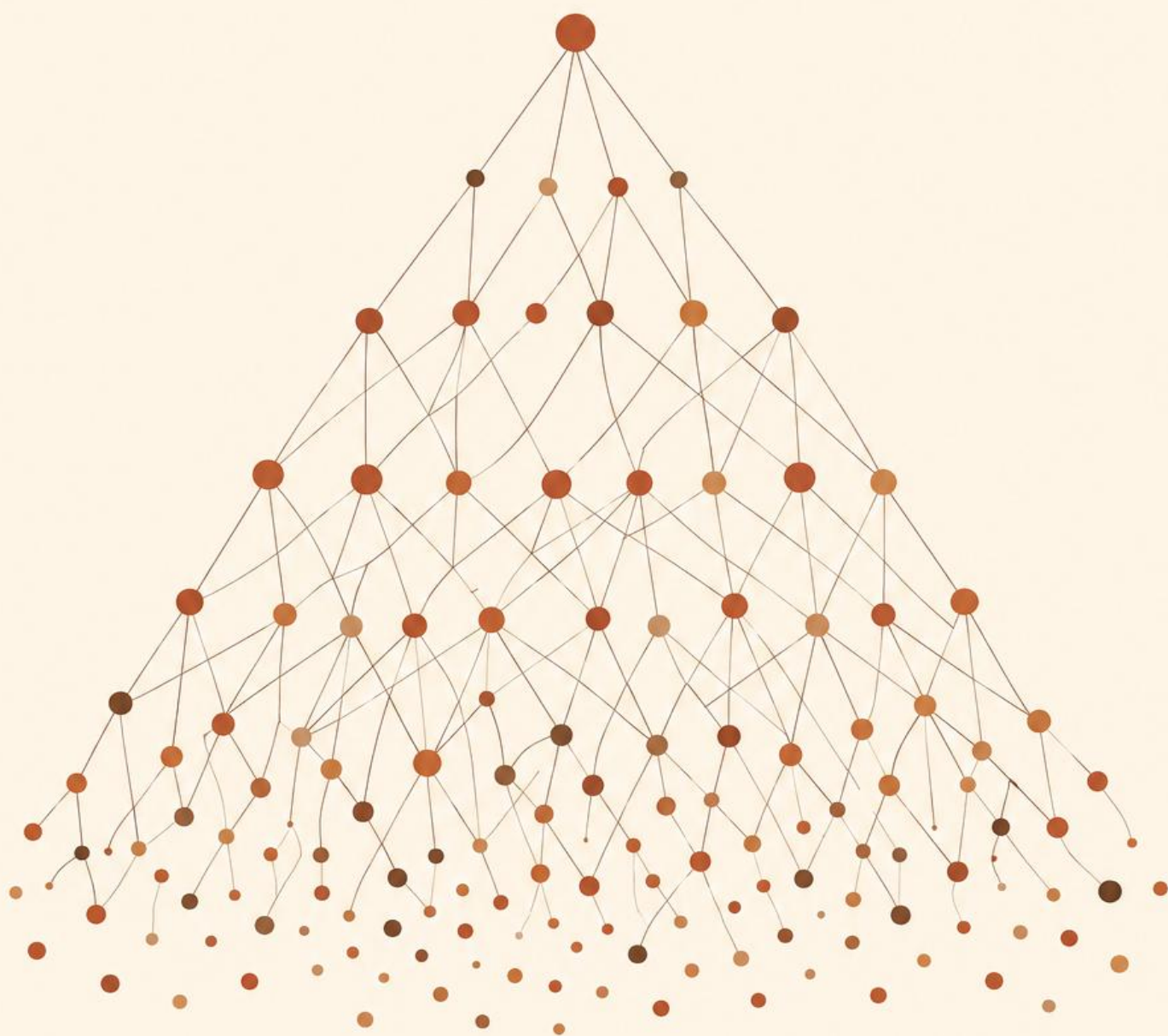


Entre a Pergunta e a Resposta

Manual de Bolso Para Quem Quer Mesmo Entender IA
(Edição para Totós)



Aristides Mendes

Entre a Pergunta e a Resposta

Copyright © 2026 Aristides Mendes

Esta obra é distribuída sob a licença Creative Commons CC BY-NC-ND 4.0.



Entre a Pergunta e a Resposta

Manual de Bolso Para Quem Quer Mesmo Entender IA

(Edição para Totós)

Aristides Mendes

PARTE I

A Receita (O que é a IA e como se faz)

CAPÍTULO 1

Não é magia, é matemática com fome

(Ou: o dia em que a sua torradeira não vai dominar o mundo)

A grande mentira dos filmes

Caso tenha aprendido sobre inteligência artificial a ver filmes, provavelmente imagina um cérebro metálico dentro de um robô que, um dia, acorda e diz: “*Penso, logo existo.*” Pouco depois, decide que a humanidade é um problema de logística e começa a resolvê-lo à sua maneira.

É uma boa história. Dá um ótimo filme. Mas é só isso — uma história. A inteligência artificial que usamos todos os dias, a que escreve um e-mail, reconhece uma cara numa fotografia ou ajuda um carro a manter-se na faixa de rodagem, está muito mais próxima de uma calculadora com uma educação esmerada do que de um cérebro rebelde à espera da sua hora.

Vale a pena guardar uma definição simples, porque vai ser o fio que atravessa todo este livro:

Inteligência artificial é um programa de computador que aprende padrões a partir de exemplos e usa esses padrões para fazer previsões sobre coisas novas.

Repare no que esta frase não diz. Não diz que o programa “compreende” o que está a fazer. Não diz que tem objectivos próprios. Não diz que escolhe agir de uma maneira em vez de outra por vontade. Diz apenas que **aprende padrões e prevê**. É uma definição modesta, quase desapontante — e é exactamente por ser modesta que é honesta.

Há três peças neste mecanismo e vamos voltar a elas vezes sem conta no livro: a **entrada** (os dados que damos à máquina), o **processamento** (as contas matemáticas que ela faz com esses dados) e a **saída** (a resposta que produz). O que costuma impressionar-nos como “inteligência” não é nenhuma destas peças isoladamente, é a complexidade do meio. São tantas contas, encadeadas de forma tão intrincada e feitas a uma velocidade tão grande, que o resultado nos surpreende como se tivesse vindo de alguém com juízo próprio. Não veio. Veio de números.

A criança e a calculadora

Imagine uma criança de quatro anos a aprender o que é um gato. Se lhe mostrar cem fotografias de gatos e cem de cães, dizendo, uma a uma: “*Isto é um gato. Isto é um cão.*” Ao fim de algum tempo — talvez não muito tempo, as crianças são rápidas nisto — ela começa a distinguir sozinha, mesmo perante fotografias que nunca viu. Reparou, sem ninguém lhe ter dito explicitamente, que os gatos tendem a ter orelhas pontiagudas e os cães, com mais frequência, orelhas caídas. Reparou na forma do nariz, no tamanho relativo da cabeça, talvez até na postura. **Aprendeu a regra sem conseguir explicar a regra.**

A inteligência artificial faz, no essencial, a mesma coisa — só que em vez de um cérebro biológico, usa matemática. Em vez de olhar para “um gato”, olha para uma grelha de números que representam os pixéis daquela fotografia: tons de cinzento ou de cor, codificados como valores numéricos. E vai ajustando, internamente, uma enorme quantidade de pequenas contas, até que essas contas, aplicadas a uma fotografia de gato, produzam consistentemente um resultado diferente do que produzem perante uma fotografia de cão. Se isto funcionar bem o suficiente — digamos, acertando 95 vezes em cada 100 — dizemos que o sistema “aprendeu a distinguir gatos de cães”.

A diferença essencial e a que separa todo este livro de um livro sobre robôs conscientes, é esta: **a criança sabe o que é um gato.** Sabe que é um ser vivo, que pode ronronar ou arranhar, que tem fome, que talvez tenha medo de água. Tem, no jargão dos filósofos, um *modelo do mundo* à volta da palavra “gato” — uma rede densa de associações, experiências e sentidos. A IA não tem nada disto. Descobriu apenas que, ao combinar certos números segundo uma certa fórmula, o resultado tende a corresponder àquilo que os humanos, nos dados de treino, rotularam como “gato”. Mude a fórmula ligeiramente, ou mostre-lhe uma imagem suficientemente diferente das que viu e pode errar com a mesma confiança com que, noutras ocasiões, acerta.

Aqui vale a pena dar uma segunda volta à analogia, porque é instrutiva. O que acontece se mostrarmos à nossa criança um gato de uma raça que nunca viu — um sphynx, sem pelo, com a pele rosada e cheia de rugas? Provavelmente vai estranhar, mas ainda assim vai reconhecer um gato: tem o tal modelo do mundo a apoiá-la, sabe que gatos podem variar bastante e continuar a ser gatos. Já a IA, perante uma imagem suficientemente afastada do que viu durante o treino, pode hesitar de formas estranhas — confundir o sphynx com outra coisa qualquer, ou recusar-se a classificar com confiança. Não porque “não saiba o que é um gato sem pelo” no sentido em que nós não sabemos algo; mas porque os seus números nunca foram ajustados para aquele tipo de imagem em particular. Isto é uma

primeira pista de um tema que vamos desenvolver com calma no Capítulo 10: a máquina é tão boa quanto os exemplos que viu e arrisca-se a tropeçar exactamente onde os exemplos escassearam.

A “fome” do título

Porque é que chamamos a isto “matemática com fome”? Porque a inteligência artificial, ao contrário de um programa tradicional, não vem com regras escritas à mão por um programador. Precisa de **dados** — exemplos reais, em quantidade — para descobrir essas regras por si própria. Sem dados, é uma calculadora vazia: sabe fazer contas, mas não tem nada sobre que as fazer.

E a quantidade necessária é, muitas vezes, de tirar o fôlego. Para um modelo de linguagem actual aprender a escrever em português com fluência — a construir frases gramaticalmente correctas, a manter um fio de raciocínio, a imitar registos diferentes de quem escreve um relatório ou quem escreve um poema — terá processado, durante o treino, uma fatia descomunal de texto: praticamente toda a Wikipédia, milhares ou milhões de livros, fóruns de discussão, notícias, manuais técnicos e sim, também receitas de bacalhau à Gomes de Sá. Para um sistema diferente, especializado em reconhecer rostos, terá sido necessário “ver” milhões de fotografias de pessoas, em todas as condições de luz, ângulo e expressão imagináveis — sorridentes, sérias, de perfil, com barba, sem barba, com chapéu. E para um sistema que joga xadrez ao mais alto nível, como o célebre AlphaZero, a fonte de exemplos nem veio de fora: o próprio sistema jogou milhões de partidas contra si mesmo, sem que ninguém lhe explicasse estratégia alguma — apenas observando, repetidamente, o que tendia a levar à vitória e o que tendia a levar à derrota.

Esta última história merece um parêntesis, porque desfaz outro mito comum: o de que a IA precisa sempre de um humano a “ensinar-lhe” a resposta certa caso a caso. Por vezes precisa — é o caso da criança a aprender com fotografias rotuladas. Mas, por vezes, basta dar-lhe um objectivo claro (ganhar o jogo) e deixá-la experimentar à vontade, milhões de vezes, até que ela própria descubra padrões vencedores. Os detalhes técnicos desta distinção vão aparecer mais à frente; por agora, fica a ideia de que há mais do que uma forma de “alimentar” a fome de dados.

Há ainda um terceiro caminho, mais discreto mas talvez o mais comum de todos para sistemas como os assistentes de conversação: depois de um primeiro treino com texto da internet, equipas de pessoas reais avaliam centenas de milhares de respostas geradas pelo sistema, indicando quais são úteis, quais são confusas, quais são perigosas. Por trás de cada

resposta educada e bem comportada que recebemos, há, normalmente, uma quantidade significativa de trabalho humano — muitas vezes invisível, por vezes mal pago — a refinar o comportamento da máquina depois do treino inicial. Voltaremos a este processo, chamado aprendizagem por reforço com feedback humano, no Capítulo 10.

O que é seguro dizer é isto: quanto mais dados de qualidade um sistema recebe, mais padrões consegue encontrar e mais afinadas ficam as suas previsões. Mas há um limite — chega-se a um ponto em que mais dados já não trazem grande melhoria, porque o sistema já “absorveu” tudo o que havia para absorver daquele tipo de exemplo. Voltaremos a este limite, chamado saturação, quando falarmos da quantidade de dados necessária no Capítulo 4.

Por que é que isto nos parece inteligência

Vale a pena parar aqui um momento, porque há um efeito psicológico interessante a acontecer e entendê-lo poupa muita confusão mais à frente no livro.

Quando perguntamos a um assistente de IA *“Qual é a capital de Portugal e que rio a atravessa?”*, a resposta correcta — Lisboa, atravessada pelo Tejo — chega em segundos, bem escrita, com a pontuação certa. Se um amigo nos respondesse assim, diríamos que sabe geografia. Damos por nós a tratar a máquina da mesma forma: como se “soubesse” algo, como se tivesse ido buscar essa informação a uma memória organizada, da mesma maneira que nós o fazemos.

Mas o que aconteceu por baixo do capô foi outra coisa. O sistema não “sabe” que Lisboa é a capital no sentido em que nós sabemos — não tem uma imagem mental da cidade, não esteve lá, não tem orgulho ou afecto pelo Terreiro do Paço. O que tem é um conjunto de padrões estatísticos aprendidos a partir de quantidades imensas de texto em que “Portugal”, “capital” e “Lisboa” apareceram juntos, repetidamente, em contextos consistentes — e, separadamente, “Lisboa”, “rio” e “Tejo” apareceram juntos com a mesma consistência. Quando lhe pedimos a resposta, o sistema está, no fundo, a calcular: dada esta sequência de palavras na pergunta, qual é a sequência de palavras que, segundo tudo o que viu durante o treino, tem mais probabilidade de vir a seguir?

Isto pode parecer um truque decepcionante, mas repare na escala do que está a acontecer. Não é uma tabela com duas entradas (“Portugal” → “Lisboa”). É uma teia de milhões de associações sobrepostas, capazes de responder não só à pergunta exacta que treinámos, mas a milhares de variações que o sistema nunca viu formuladas exactamente

daquela maneira — “Onde fica a sede do governo português?”, “Que cidade portuguesa tem o Tejo a desaguar perto?”. É esta capacidade de generalizar, de responder bem a perguntas nunca vistas com a mesma forma exacta, que produz a sensação de inteligência. Não é compreensão no sentido humano — é, isso sim, uma forma de correlação estatística tão vasta e tão bem afinada que, de fora, se torna difícil distinguir de compreensão. O Capítulo 11 deste livro vai voltar a este ponto com mais detalhe, porque é provavelmente a confusão mais comum — e mais consequente — que as pessoas têm sobre IA.

O que a IA não é — e o que é, na verdade

Para não restarem dúvidas logo à entrada do livro, fica aqui uma comparação directa entre a imagem popular e a realidade técnica:

Não é	É, na verdade
Uma mente consciente	Um programa que faz contas estatísticas, muitas e a grande velocidade
Algo que “pensa” antes de responder	Algo que calcula qual é a palavra (ou pixel, ou número) mais provável a seguir
Infalível	Capaz de errar com frequência — e, quando erra, fá-lo muitas vezes com total confiança
Uma coisa só, sempre igual	Centenas de sistemas diferentes, cada um treinado e afinado para uma tarefa específica
Algo que entende o que diz	Algo que reproduz padrões observados em quantidades imensas de texto ou imagens humanas

Esta tabela vai reaparecer, com mais detalhe e mais argumentos, no Capítulo 11, quando dedicarmos um capítulo inteiro a desconstruir os mitos mais comuns sobre IA. Por agora, serve como mapa de orientação para todo o resto da viagem.

O que vem a seguir

Sabemos agora, em traços largos, o que é a inteligência artificial: um programa que aprende padrões a partir de dados e os usa para prever respostas a coisas novas. Mas saber o que é não responde à pergunta seguinte, que é talvez a mais prática de todas: **como é que alguém, de propósito, constrói um sistema destes?**

Não é como instalar um programa comum, descarregado de um sítio qualquer e pronto

a usar. É um processo com etapas bem definidas — definir o que se quer, recolher os exemplos certos, escolher a “receita” matemática adequada e só depois treinar e testar. No próximo capítulo, vamos seguir esse processo do início ao fim, com a ajuda de uma cozinha, de um cozinheiro e da primeira aparição de uma personagem que vai acompanhar-nos pelo resto do livro: um polvo gigante com tentáculos cheios de pequenos matemáticos a fazer contas.

Experiência de Cozinha

Não precisa de saber programar para sentir isto na pele. Abra um motor de busca de imagens e procure “gato” e, a seguir, “cão”. Repare como, em poucos segundos, viu dezenas de fotografias completamente diferentes entre si — cores, raças, poses, fundos — e ainda assim reconheceu cada categoria sem qualquer esforço consciente. Agora pense no caminho inverso: para um sistema de IA aprender a fazer exactamente o mesmo, teve de “ver” milhões dessas imagens, uma a uma, antes de conseguir generalizar a regra que você aplica em décimos de segundo. A sua intuição é instantânea, fruto de uma vida inteira de experiência sensorial; a da máquina foi construída, número a número, exemplo a exemplo, durante um processo de treino que pode demorar dias ou meses.

O RESUMO DO TOTÓ

A IA é um programa que aprende padrões a partir de exemplos e não um cérebro com vontade própria. Funciona através de três peças simples — entrada, processamento, saída — mas a complexidade desse processamento, feita à custa de enormes quantidades de dados, é o que produz resultados que parecem mágicos. Como a criança que aprende a distinguir gatos de cães sem saber explicar como, a máquina descobre regras sozinha; mas, ao contrário da criança, não tem qualquer noção do que esses padrões realmente significam no mundo. É uma máquina de calcular previsões — talentosa, útil e completamente desprovida de compreensão.

CAPÍTULO 2

A Receita e o Cozinheiro

(Ou: o que é que uma pessoa tem de fazer para criar uma IA do zero?)

O sonho de todos os dias

Imagine que acorda amanhã e decide: *“Vou criar uma inteligência artificial que classifique os comentários dos meus clientes como positivos, negativos ou neutros, para a minha loja online.”*

O que faz a seguir? Não é ligar um cabo qualquer e instalar um programa que já vem com isso resolvido. É um processo com cinco passos fundamentais e vale a pena percorrê-los com calma — não só porque respondem à pergunta prática “como se cria uma IA”, mas porque cada um destes passos vai reaparecer, de uma forma ou de outra, no resto do livro.

Passo 1 — Definir o problema (o que quero que ela faça?)

Parece óbvio, mas é precisamente aqui que a maioria das tentativas falha antes de sequer começar. Não se pode dizer: *“Quero uma IA que seja esperta”* ou *“Quero uma IA que entenda os meus clientes.”* Isto não é um problema bem definido — é um desejo vago, do tipo que nenhuma máquina, nem nenhuma equipa de engenheiros, sabe por onde pegar.

Compare estas duas formulações:

- ✓ **Mal definida:** *“Quero uma IA que entenda os meus clientes.”*
- ✓ **Bem definida:** *“Quero uma IA que, dado um comentário de texto em português, classifique o sentimento como positivo, neutro ou negativo, com pelo menos 85% de acerto quando comparada com a classificação feita por um humano.”*

Repare na diferença. A segunda formulação diz **o quê** (classificar sentimento), **com quê** (um comentário de texto) e **quão bem** (85% de acerto). É a partir desta especificidade que todo o resto do processo se torna possível — sem ela, não há forma de saber se o que se construiu funciona ou não.

Esta etapa, que parece a mais simples de todas, é também a mais ignorada por quem chega novo a este mundo. Há uma tentação de querer saltar logo para a parte vistosa — escolher um modelo sofisticado, treinar, ver resultados — e deixar a definição do problema

para resolver “depois”. Quase sempre, esse “depois” nunca chega bem e o projecto gasta semanas a perseguir um objectivo que nunca foi claramente desenhado.

Vale a pena ver mais um exemplo, porque a armadilha repete-se com frequência surpreendente. Uma equipa de uma cadeia de restauração podia, em teoria, decidir: *“Queremos uma IA que melhore o nosso serviço.”* Isto é, de novo, um desejo, não um problema. Um problema bem definido seria, por exemplo: *“Queremos um sistema que, a partir das mensagens deixadas pelos clientes num formulário de satisfação, identifique automaticamente se a reclamação é sobre comida, sobre atendimento, ou sobre tempo de espera, com pelo menos 80% de acerto, para encaminhar cada caso ao departamento certo.”* Note como esta segunda formulação já contém, embutida, uma resposta implícita à pergunta “que dados vou precisar?” — vamos precisar de mensagens de clientes, já categorizadas pelo departamento certo. É assim que um bom Passo 1 poupa imenso trabalho no Passo 2.

Passo 2 — Recolher os dados (onde arranjo comida?)

Definido o problema, a inteligência artificial precisa de **exemplos** para aprender. No nosso caso, precisamos de milhares de comentários **já classificados por humanos**, que sirvam de referência durante o treino. Por exemplo:

- ✓ “Adorei o produto, chegou rápido!” → Positivo
- ✓ “A embalagem veio toda amassada.” → Negativo
- ✓ “Funciona, mas podia ser melhor.” → Neutro

Onde arranjar isto? Há, no essencial, três caminhos: comprar bases de dados já prontas a empresas especializadas (existem várias e o Capítulo 4 vai detalhar este mercado); usar dados públicos disponíveis na internet, como comentários de produtos ou publicações em redes sociais; ou, o mais comum em projectos a sério, **pagar a pessoas** para classificarem manualmente uma amostra de comentários reais da própria empresa.

Este passo é, quase sempre, **o mais caro e o mais demorado** de todo o processo — mais até do que a parte que parece tecnicamente mais impressionante, que é o treino. E há uma razão simples para isso: **a IA só é tão boa quanto os dados que recebe**. Se os exemplos forem poucos, distorcidos, ou mal classificados, todo o resto do processo está condenado a herdar esses defeitos, por mais sofisticado que seja o modelo escolhido a seguir.

Passo 3 — Escolher o modelo (a receita do bolo) — é aqui que apresentamos o polvo

Antes de entrar no Passo 3, vale a pena um aparte rápido sobre o exemplo que estamos a usar. Ao longo deste capítulo, a nossa IA de referência é um classificador de comentários — um sistema que lê um texto e cospe uma etiqueta: "Positivo", "Negativo" ou "Neutro". É um exemplo deliberadamente simples, porque os cinco passos ficam mais fáceis de ver com ele. Mas o leitor de hoje associa "IA" quase exclusivamente a assistentes de conversação como o ChatGPT — e esses funcionam de forma diferente: em vez de cospir uma etiqueta, cospem tokens para construir texto corrido, palavra a palavra. A boa notícia é que os cinco passos deste capítulo se aplicam a ambos; o que muda é o que sai do polvo no final. Ao longo do livro, vamos voltar a esta distinção com mais detalhe — por agora, basta saber que o nosso polvo classificador e o polvo dos assistentes de conversação são primos próximos, mas com saídas muito diferentes.

Chegámos ao momento em que se escolhe um “modelo”. Um modelo é a arquitectura matemática que vai aprender os padrões a partir dos dados — é, em certo sentido, a receita que vamos seguir para fazer o nosso bolo. Tal como há receitas simples (um pão de ló, com poucos ingredientes e pouco tempo de forno) e receitas elaboradas (um bolo com várias camadas, cremes e horas de preparação), também há modelos simples e modelos complexos.

Para o nosso problema de classificar sentimentos, poderíamos escolher, por exemplo, um modelo relativamente simples (chamado regressão logística no jargão técnico) — rápido a treinar, barato em termos de computação, mas menos preciso em casos ambíguos. Ou poderíamos escolher um modelo muito mais complexo, da família dos que sustentam os assistentes de conversação actuais — mais caro, mais lento a treinar, mas tipicamente muito mais capaz de captar nuances de linguagem.

E é aqui, exactamente neste ponto do livro, que apresentamos a analogia que vai acompanhar-nos pelos próximos capítulos.

Imagine um polvo gigante, com centenas de tentáculos e cada tentáculo cheio de centenas de ventosas. Cada ventosa é como um pequeno matemático sentado a uma mesa, com uma calculadora à frente. A inteligência artificial é esse polvo. Quando lhe colocamos uma pergunta, os números entram pela boca do polvo, viajam por todos os tentáculos — cada um a fazer as suas próprias contas — e, no final, o polvo “cospe” a resposta.

Vale a pena acrescentar aqui uma nota que vai fazer muito mais sentido no Capítulo 8, mas que é útil plantar já: quando o polvo do nosso classificador de comentários "cospe" uma resposta, cospe uma coisa só — uma etiqueta, "Positivo", "Negativo" ou "Neutro" — e fica por aí. Já o polvo de um assistente de conversação como o ChatGPT ou o Claude cospe uma palavra de cada vez, coloca-a na mesa de trabalho à sua frente, relê tudo o que está nessa mesa (a pergunta original mais as palavras que já gerou), e só então cospe a palavra seguinte. É como ditar uma frase a um estenógrafo que, a cada palavra que escreve, nos lê em voz alta tudo o que escreveu até ao momento — para garantir que a próxima palavra faz sentido no contexto de tudo o que veio antes. A este mecanismo chamamos geração autoregressiva, e é ele que permite ao polvo construir parágrafos inteiros com coerência interna, em vez de se limitar a escolher entre três etiquetas.

As ventosas têm aquilo a que chamamos pesos: números que funcionam como a “personalidade” de cada ventosa, determinando se ela tende a somar, a subtrair, a multiplicar, ou a combinar os números de entrada de formas mais complexas. No início, antes de qualquer treino, estas personalidades são praticamente aleatórias — o polvo está, por assim dizer, ainda a aprender a calçar as próprias ventosas. Ao longo do treino, essas personalidades vão sendo ajustadas, repetidamente, até que o polvo comece a acertar com regularidade nas respostas que lhe pedimos. No Capítulo 8 vamos abrir o polvo a sério e observar, tentáculo a tentáculo, o que se passa quando ele responde a uma pergunta.

Porquê escolher precisamente um polvo e não outro animal qualquer? Por duas razões. Primeiro, porque um polvo processa informação de forma genuinamente paralela — muitos tentáculos, muitas ventosas, todos a trabalhar ao mesmo tempo — o que é fiel ao que acontece de facto dentro de uma rede neuronal, onde milhares ou milhões de pequenas unidades fazem contas simultaneamente. Segundo, porque é uma imagem com a qual é fácil construir intuição: é mais simples visualizar tentáculos com ventosas do que uma abstracção chamada “rede neuronal com camadas ocultas”. Ao longo do livro, sempre que falarmos de arquitectura, processamento, ou daquilo que parece ser o “pensar” da máquina, é a este polvo que vamos voltar.

Passo 4 — Treinar a IA (cozinhar o bolo)

Chegamos ao momento mais aguardado e talvez o mais mal compreendido pelo público em geral. Pegamos nos dados que recolhemos no Passo 2 — os comentários, já classificados por humanos como positivos, negativos ou neutros — e entregamo-los ao modelo escolhido no Passo 3. Ou seja: atiramos tudo para dentro do polvo.

O que acontece nos bastidores, passo a passo, é o seguinte:

1. O polvo começa com pesos aleatórios — todas as ventosas estão, por assim dizer, confusas, sem saber bem o que fazem.
2. Recebe um comentário, faz as suas contas (passa pelos tentáculos todos) e produz uma resposta — que, no início, vai estar quase sempre **errada**.
3. Comparamos essa resposta com a classificação correcta que já tínhamos: *“Disseste que era positivo, mas a pessoa que classificou isto disse que era negativo. Erraste.”*
4. O polvo ajusta as suas ventosas — muda ligeiramente os pesos — de forma a diminuir esse erro na próxima vez que vir um comentário parecido.
5. Repete este ciclo, exemplo após exemplo, **milhares ou milhões de vezes**.

A este processo chamamos treino, ou, em termos mais técnicos, aprendizagem supervisionada — supervisionada porque há sempre um “supervisor” (os dados já classificados por humanos) a dizer ao polvo se acertou ou errou, permitindo-lhe corrigir-se. É um processo lento, computacionalmente exigente e que pode demorar desde minutos (para problemas pequenos) até semanas ou meses (para os modelos de linguagem que sustentam os grandes assistentes de conversação actuais).

Passo 5 — Testar e validar (provar o bolo)

Depois de treinado, o polvo já consegue classificar comentários — mas não podemos confiar cegamente nele só porque acertou nos exemplos que já viu durante o treino. Isso seria como avaliar um aluno só com perguntas que ele já decorou previamente.

Por isso, separamos uma parte dos dados — comentários que o polvo nunca viu durante o treino — e testamo-lo com eles. Se acertar em 87% destes casos novos, temos um resultado sólido e o projecto está em boa forma para avançar. Se acertar apenas em 50%, algo correu mal: ou os dados de treino eram insuficientes, ou o modelo escolhido era demasiado simples para a tarefa, ou há ruído nos dados que está a confundir o polvo. Nesse caso, volta-se ao Passo 4 — treinar mais, ou com mais cuidado — ou até ao Passo 3, escolhendo um modelo diferente.

O papel do “cozinheiro” — quem é, de facto, a pessoa que faz isto

A pessoa que conduz todo este processo não é exactamente um “programador” no sentido tradicional — alguém que escreve instruções precisas, linha a linha, para o

computador seguir. É, mais propriamente, um cientista de dados ou um engenheiro de aprendizagem automática (machine learning engineer, no jargão habitual). O que faz, na prática do dia a dia, distribui-se de forma reveladora:

- ✓ **Cerca de 20% do tempo:** escrever código, sobretudo em Python — a linguagem de programação mais usada para este tipo de trabalho.
- ✓ **Cerca de 30% do tempo:** limpar dados — lidar com comentários com erros ortográficos, frases sem sentido, registos duplicados, valores em falta.
- ✓ **Cerca de 30% do tempo:** experimentar — testar dez ou vinte combinações diferentes de modelo, parâmetros e estratégias de treino, para ver qual funciona melhor.
- ✓ **Os restantes 20% do tempo:** uma mistura de espera, paciência e — como dizem os próprios cientistas de dados, meio a brincar, meio a sério — alguma “reza”, porque há momentos em que o treino simplesmente não converge bem e nem sempre é óbvio porquê.

Há também coisas importantes que esta pessoa não faz e que vale a pena desfazer aqui, já que tocam directamente em mitos que vamos desenvolver no Capítulo 11. Não liga a IA a um cérebro humano, real ou simulado. Não “programa” conhecimento à mão, frase a frase — não escreve, em código, regras do tipo “a honestidade é boa, a mentira é má”. E, frequentemente, nem ela própria entende em detalhe absoluto como o polvo chegou a uma decisão específica — os pesos finais, depois de milhões de ajustes, formam um emaranhado tão complexo que se torna, em larga medida, uma caixa-negra mesmo para quem a construiu. Esta opacidade, que parece desconcertante à primeira vista, é uma das características mais discutidas — e mais investigadas — em toda a área da inteligência artificial moderna.

Vale ainda uma palavra sobre como esta profissão chegou a ser tão procurada. Há uma década, “cientista de dados” era um título relativamente raro, associado quase exclusivamente a grandes empresas tecnológicas com orçamentos generosos. Hoje, praticamente qualquer empresa de média dimensão — uma seguradora, uma cadeia de retalho, um banco regional — tem, ou procura ter, alguém com este perfil. A razão é simples: cada vez mais decisões de negócio passaram a depender de padrões escondidos em grandes quantidades de dados e foi-se percebendo que essa procura de padrões é, no fundo, exactamente o trabalho descrito nestes cinco passos. O “cozinheiro” deixou de ser uma curiosidade de laboratório de investigação e tornou-se uma peça comum em qualquer equipa que lide com volumes razoáveis de informação.

Experiência de Cozinha

Sem instalar nada no seu computador, pode experimentar uma versão simplificada deste processo inteiro através do Google Colab, um serviço gratuito que disponibiliza ambientes de programação prontos a usar, directamente no navegador. Procure por tutoriais de “classificação de texto com poucas linhas de código” — em menos de uma hora, é possível treinar uma versão minúscula do nosso polvo de sentimentos, usando apenas algumas dezenas de frases como exemplo e ver, com os seus próprios olhos, os cinco passos deste capítulo a acontecer diante de si.

O RESUMO DO TOTÓ

Criar uma inteligência artificial segue cinco passos: definir com precisão o problema a resolver; recolher dados já classificados que sirvam de exemplo; escolher o modelo — a “receita” matemática, que vamos chamar de polvo pelo resto deste livro; treinar esse modelo, ajustando repetidamente os seus pesos internos até ele começar a acertar; e, por fim, testar com dados novos para confirmar que o resultado é fiável. O “cozinheiro” deste processo, o cientista de dados, passa mais tempo a limpar dados e a fazer experiências do que a escrever código propriamente dito — e quase nunca entende, em detalhe absoluto, como o polvo chegou a cada decisão concreta.

CAPÍTULO 3

As Ferramentas do Ofício

(Ou: o que é que eu preciso de comprar ou instalar para fazer uma IA?)

O mito do supercomputador

Há uma pergunta que surge invariavelmente quando alguém começa a interessar-se por inteligência artificial: *“Não preciso de ter um supercomputador como o da Google para fazer isto, pois não?”*

A resposta curta é: **não precisa**, pelo menos para começar. Um computador portátil normal, do tipo que já tem em casa para trabalhar ou navegar na internet, é suficiente para experimentar os primeiros passos deste mundo — treinar pequenos modelos, testar ideias, seguir tutoriais. Para projectos mais sérios, a conversa muda e vamos ver exactamente onde e porquê. Mas vamos por partes, como convém a um capítulo sobre ferramentas.

No capítulo anterior, conhecemos o cozinheiro e os cinco passos da receita. Este capítulo é sobre a cozinha propriamente dita: o que tem de estar lá dentro para a receita ser, de facto, possível de cozinhar.

Ferramenta 1 — O hardware (a cozinha)

Começamos pela peça mais visível e a que mais alimenta o imaginário popular de salas cheias de máquinas com luzes a piscar: o computador onde tudo isto acontece.

Para experimentar, aprender e treinar modelos pequenos — como o exemplo dos comentários de clientes do Capítulo 2 — um computador com um processador razoável e uma quantidade de memória normal para os dias de hoje chega perfeitamente. Não é preciso equipamento especial. É, de facto, possível seguir tutoriais inteiros de introdução à inteligência artificial sem gastar um centimo em hardware novo.

A conversa muda de figura quando o objectivo é treinar modelos grandes — do tipo que sustenta os assistentes de conversação que hoje todos conhecemos. Aí entram em jogo as **GPUs**, ou placas gráficas. O nome vem de *Graphics Processing Unit*, porque estas peças foram originalmente concebidas para desenhar gráficos de jogos de vídeo — uma tarefa que exige fazer muitíssimas operações matemáticas simples ao mesmo tempo, para desenhar milhões de pixéis num ecrã, várias vezes por segundo. Ora, descobriu-se que essa

mesma capacidade — fazer muitas contas simples em paralelo — é exactamente o que se precisa para treinar redes neuronais. Por uma feliz coincidência histórica, o hardware que os jogadores de computador já usavam tornou-se também a ferramenta essencial da inteligência artificial moderna.

As GPUs usadas para treinar os modelos mais avançados de hoje custam, tipicamente, vários milhares de euros cada uma — e os projectos maiores não usam uma, usam centenas ou milhares delas, ligadas em conjunto, durante semanas ou meses seguidos. Isto explica, em larga medida, por que motivo treinar um modelo do zero ao nível dos grandes assistentes de conversação está fora do alcance de uma pessoa individual e mesmo de muitas empresas — é um investimento que se mede em milhões de euros, sobretudo em electricidade e em equipamento. Vale a pena notar que estes números mudam com alguma rapidez, à medida que novas gerações de GPUs aparecem no mercado; o que fica estável é o princípio: quanto maior o modelo, mais GPUs são necessárias e mais tempo (e dinheiro) é preciso para o treinar.

Felizmente, há um meio-termo entre “o portátil que já tenho” e “construir um centro de dados”: **alugar** capacidade de computação por horas, através de serviços na nuvem como os disponibilizados pela Amazon, pela Google ou pela Microsoft. Isto permite a uma pequena empresa, ou até a uma pessoa curiosa, aceder durante algumas horas a uma GPU potente, pagar apenas por esse uso e depois libertar o recurso — sem nunca ter de comprar o equipamento físico. É, na prática, como alugar uma cozinha profissional por uma tarde, em vez de construir uma em casa.

Há ainda duas outras peças de hardware que merecem uma palavra, ainda que tenham um papel menos vistoso do que a GPU. A primeira é a **memória RAM** — a memória de trabalho do computador, onde ficam temporariamente os dados que estão a ser processados num dado momento. Para treinar modelos pequenos, a RAM de um portátil normal costuma ser suficiente; para lidar com conjuntos de dados maiores, pode tornar-se um obstáculo antes mesmo da GPU, porque os dados simplesmente não cabem todos na memória de uma vez. A segunda é o **disco de armazenamento**, onde os dados ficam guardados de forma permanente entre uma sessão de trabalho e a seguinte — discos de estado sólido (SSD), mais rápidos do que os discos rígidos tradicionais, fazem uma diferença real quando se está a ler e a escrever grandes quantidades de informação repetidamente durante o treino.

Ferramenta 2 — O software (os utensílios de cozinha)

Tendo a cozinha (o hardware), falta equipá-la com os utensílios certos. No mundo da inteligência artificial, isto significa, sobretudo, uma linguagem de programação e algumas bibliotecas de código.

A linguagem mais usada, por uma margem considerável, é o **Python**. Não é por acaso: o Python tem uma sintaxe relativamente simples de ler e escrever, mesmo para quem nunca programou antes e ao longo dos últimos quinze a vinte anos acumulou uma comunidade enorme de pessoas a desenvolver ferramentas específicas para ciência de dados e inteligência artificial. Quando alguém diz que “está a aprender a programar IA”, é quase certo que está, no fundo, a aprender Python.

Sobre o Python assentam **bibliotecas** — pacotes de código já escrito por outras pessoas, que poupam a quem está a construir uma IA o trabalho de reinventar, a cada projecto, as operações matemáticas básicas de uma rede neuronal. As duas mais conhecidas são o **PyTorch**, mantido principalmente por uma equipa ligada à Meta e o **TensorFlow**, mantido pela Google. Ambas são gratuitas e de uso livre e ambas fazem, no essencial, o mesmo trabalho: dão, a quem programa, um conjunto de “blocos de construção” prontos a usar para montar e treinar redes neurais, sem ter de escrever, do zero, a matemática que está por trás de cada ventosa do nosso polvo.

Para quem não quer instalar nada no computador — nem Python, nem bibliotecas, nem se preocupar com versões e configurações — existe ainda o **Google Colab**, um serviço gratuito que disponibiliza um ambiente de programação completo, a correr directamente no navegador de internet, com acesso ocasional a GPUs sem custo. É, provavelmente, a porta de entrada mais simples para qualquer pessoa que queira experimentar este mundo sem qualquer investimento inicial — e é a ferramenta que temos vindo a sugerir nas caixas de “Experiência de Cozinha” deste livro.

Existe também o **Anaconda**, um programa que trata de instalar o Python e várias bibliotecas relacionadas de uma só vez, de forma organizada — como comprar um kit de utensílios de cozinha já montado, em vez de ir à loja comprar cada peça em separado.

Ferramenta 3 — Os dados (a comida)

Já vimos, no Capítulo 2, que os dados são o ingrediente mais importante de toda a receita — e o Capítulo 4 vai dedicar-se inteiramente a explorar as suas origens. Mas, já que estamos a fazer um inventário de ferramentas, vale a pena fixar aqui, em traços largos, onde

se costuma ir buscar a “comida” para um primeiro projecto.

Há plataformas inteiras dedicadas a disponibilizar conjuntos de dados gratuitos, prontos a usar, em formatos organizados — o **Kaggle** é provavelmente a mais conhecida e contém desde fotografias de gatos e cães até preços históricos de casas ou registos médicos anonimizados. Há também repositórios académicos mais antigos, como o **UCI Machine Learning Repository**, usados há décadas em universidades para ensinar os fundamentos da disciplina. Há ainda **interfaces públicas** (chamadas APIs, no jargão técnico) que permitem aceder programaticamente a dados da Wikipédia, de organismos governamentais portugueses e europeus, ou de outras fontes oficiais. E, por fim, há os **dados próprios** — os de uma loja, de uma empresa, de um projecto pessoal — que costumam ser os mais valiosos exactamente porque são específicos ao problema que se quer resolver, mas que também exigem mais trabalho de preparação antes de poderem ser usados.

Ferramenta 4 — O “cérebro” guardado (onde fica tudo depois de pronto)

Há uma última ferramenta a considerar e é a que faz a ponte directa para o próximo capítulo deste livro. Quando o nosso polvo termina o treino — quando os seus pesos já não estão a ser ajustados e o sistema está pronto a responder a perguntas — toda essa “personalidade” aprendida precisa de ficar guardada em algum lado, num ficheiro.

O tamanho deste ficheiro varia enormemente, dependendo da complexidade do modelo. Para uma IA pequena, dedicada a uma tarefa simples como a nossa classificação de comentários, o ficheiro pode ocupar apenas alguns megabytes — menos espaço do que uma única canção em formato digital. Para um modelo de tamanho médio, especializado, por exemplo, em reconhecer objectos em fotografias, o ficheiro pode crescer para algumas centenas de megabytes — comparável a um filme curto. Já para os grandes modelos de linguagem que sustentam os assistentes de conversação actuais, o ficheiro de pesos pode ocupar centenas de gigabytes — o equivalente a uma colecção considerável de filmes em alta definição. Estes números, tal como os preços das GPUs mencionados anteriormente, tendem a crescer de ano para ano, à medida que os modelos se tornam mais capazes; o que importa reter é a escala relativa, não o número exacto.

Este ficheiro fica guardado, normalmente, num disco rígido ou num serviço de armazenamento na nuvem — fixo, estático, sem necessidade de estar ligado a nada para sobreviver. Para responder a perguntas, o sistema vai buscar esse ficheiro e carrega-o para a memória rápida do computador, de onde o pode processar a uma velocidade muito maior

do que se tivesse de ler o disco a cada cálculo. Falaremos com mais detalhe sobre esta distinção — entre o local onde a informação “mora” de forma permanente e o local onde é “trabalhada” durante uns segundos — já no próximo capítulo, que trata exactamente desta viagem entre o disco e a memória.

Quanto custa, realmente, fazer isto

Vale a pena terminar este capítulo com um resumo honesto sobre dinheiro, porque é uma das perguntas mais frequentes — e mais mal respondidas pelo imaginário popular, que tende a achar que tudo isto custa ou nada, ou uma fortuna inacessível.

Para começar a aprender, experimentar e treinar modelos pequenos, o custo pode ser, literalmente, **zero euros**: um computador que já se tenha em casa, o Google Colab gratuito e conjuntos de dados públicos disponíveis em plataformas como o Kaggle. É possível passar muitas tardes interessantes nesta fase sem gastar dinheiro algum.

Para um projecto profissional de pequena ou média dimensão — como a IA de classificação de comentários que temos vindo a usar como exemplo — os custos costumam envolver, sobretudo, o tempo de uma pessoa qualificada (o nosso cozinheiro do Capítulo 2) e, eventualmente, algumas horas de GPU alugada na nuvem, o que se traduz tipicamente em algumas centenas a alguns milhares de euros, dependendo da escala do projecto. Pense, por exemplo, numa pequena loja online portuguesa que queira automatizar a triagem de comentários de clientes: o trabalho de recolher e classificar alguns milhares de exemplos, escolher um modelo já existente e adaptá-lo (uma técnica chamada fine-tuning, que vamos explorar com mais detalhe no Capítulo 7) e testar o resultado, pode ser feito por uma pessoa, em algumas semanas, com um orçamento que cabe perfeitamente no plano de despesas de uma pequena empresa — está muito mais próximo dos milhares de euros do que dos milhões.

Para construir algo da escala dos grandes modelos de linguagem actuais, a conversa muda inteiramente de patamar: os custos medem-se em milhões ou mesmo centenas de milhões de euros, entre electricidade, equipamento e equipas inteiras de especialistas a trabalhar durante meses. Mas isso, como diria um bom cozinheiro, é conversa para outro livro — o que importa aqui é que, entre estes dois extremos, há um espaço generoso onde a maioria dos projectos reais de inteligência artificial efectivamente acontece.

Há, contudo, um custo que raramente aparece nas conversas sobre preços da IA, mas que em 2026 já não é possível ignorar: o custo físico. Os centros de dados onde as GPUs

habitam não consomem apenas electricidade — e muita: os maiores centros de dados dedicados à inteligência artificial consomem, cada um, tanta energia quanto uma cidade de média dimensão. Consomem também água, em quantidades que surpreendem quem nunca pensou nisto: as GPUs geram calor imenso durante o treino e durante a inferência (o nome técnico para "responder a uma pergunta"), e esse calor é dissipado, em grande parte, através de sistemas de arrefecimento que usam água. Estima-se que um único modelo grande, durante o seu treino, pode consumir milhões de litros de água. A matemática com fome que descrevemos no Capítulo 1 é, literalmente, uma matemática com sede — e com uma factura energética que torna o debate sobre a sustentabilidade da IA um dos mais prementes da actualidade. Não há, aqui, uma resposta simples: a IA pode também ajudar a otimizar redes eléctricas, a acelerar investigação em energias renováveis e a tornar processos industriais mais eficientes. Mas o custo existe, é real, e vale a pena tê-lo presente sempre que se fala de "quanto custa fazer isto".

Experiência de Cozinha

Se ainda não experimentou o Google Colab, esta é a oportunidade ideal. Abra um motor de busca, procure “Google Colab” e crie um novo “notebook” — um documento interactivo onde se escreve e executa código directamente no navegador. Não precisa instalar nada. Experimente escrever simplesmente `print(“Olá, polvo”)` numa célula e executá-la. Parece pouco, mas acabou de usar, gratuitamente e sem qualquer instalação, exactamente o mesmo tipo de ambiente onde milhares de cientistas de dados em todo o mundo treinam os seus modelos todos os dias.

O RESUMO DO TOTÓ

Para criar uma inteligência artificial, são necessárias três categorias de ferramentas: hardware (um computador normal chega para começar; GPUs, alugadas ou compradas, são necessárias para projectos maiores), software (a linguagem Python, bibliotecas gratuitas como o PyTorch ou o TensorFlow e serviços sem instalação como o Google Colab) e dados (bases públicas como o Kaggle, ou dados próprios de uma empresa ou projecto). Depois de treinado, o modelo fica guardado num ficheiro que pode variar de poucos megabytes a centenas de gigabytes, dependendo da sua complexidade. O custo de entrada neste mundo pode ser zero; o custo de construir algo ao nível dos maiores assistentes de conversação actuais mede-se em milhões de euros — e é precisamente esta distância entre os dois extremos que explica por que motivo a inteligência artificial parece, simultaneamente, tão acessível e tão inacessível.

PARTE II

A Despesa e o Armazém (A informação)

CAPÍTULO 4

De Onde Vem a Comida

(Ou: a odisseia dos dados, da internet até ao prato da inteligência artificial)

O ingrediente mais importante — e o mais subestimado

Se a inteligência artificial é, como temos vindo a dizer, uma espécie de bolo matemático, os dados são a farinha, os ovos e o açúcar. Sem eles, não há bolo — por mais talentoso que seja o cozinheiro, por mais sofisticado que seja o forno.

Há uma verdade incômoda que vale a pena dizer já no início desta parte do livro: **uma inteligência artificial alimentada com dados maus é pior do que nenhuma inteligência artificial**. Se lhe dermos exemplos errados, distorcidos ou tendenciosos, ela vai aprender exactamente esses erros, essas distorções e essas tendências — e fá-lo-á com uma confiança total, sem qualquer hesitação. É o que os cientistas de dados resumem, com um certo humor negro, numa expressão que atravessou décadas de informática: *“entra lixo, sai lixo.”*

Por isso, a primeira pergunta que qualquer pessoa a construir uma inteligência artificial enfrenta — mesmo antes de pensar em modelos ou em código — é esta: *onde é que vou buscar dados bons, em quantidade suficiente e que representem fielmente o problema que quero resolver?*

Há, no essencial, cinco respostas a esta pergunta e vamos percorrê-las uma a uma.

Fonte 1 — Dados públicos (a biblioteca gratuita)

Há organizações, governos e empresas que disponibilizam dados de forma gratuita, para qualquer pessoa usar. É, de longe, o ponto de partida mais comum para quem está a aprender ou a experimentar este mundo pela primeira vez.

A **Wikipédia**, com os seus milhões de artigos em dezenas de línguas, é provavelmente o exemplo mais conhecido — funciona como uma espécie de prato principal para qualquer sistema que precise de aprender a lidar com texto. O **Common Crawl** é um arquivo gigantesco que regista, periodicamente, biliões de páginas de toda a internet — literalmente, uma fatia descomunal da web, congelada num determinado momento. O **ImageNet** contém milhões de imagens já classificadas por categoria e foi durante anos a base de referência para treinar sistemas de reconhecimento visual. O **OpenStreetMap** disponibiliza

mapas detalhados de praticamente todo o planeta, úteis para qualquer sistema que precise de raciocinar sobre localizações e percursos. E há ainda os **dados governamentais** — censos, estatísticas económicas, registos de clima, dados de trânsito — que cada país, incluindo Portugal através do INE e de outros organismos públicos, disponibiliza em portais próprios.

O problema destes dados públicos, e é um problema sério, é que estão quase sempre sujos. A Wikipédia tem artigos com erros factuais, vandalismo ocasional e qualidade desigual entre temas. O Common Crawl é, em larga medida, internet em estado bruto — páginas com publicidade entrelaçada no texto, código HTML mal limpo, conteúdo duplicado e por vezes texto que não passa de spam. O ImageNet tem imagens ocasionalmente mal classificadas. Antes de qualquer destes conjuntos de dados poder ser usado a sério, é preciso limpá-los — e essa limpeza, como já vimos no Capítulo 2, costuma ser a parte do trabalho que consome mais tempo a um cientista de dados.

Fonte 2 — Dados próprios (a comida de casa)

Se gere uma empresa, mesmo pequena, é muito provável que já tenha, sem se aperceber, dados valiosos acumulados: histórico de vendas, interações com clientes, registos de utilização de um sistema, fotografias dos produtos que vende.

Uma loja online tem, naturalmente, dados de compras — e pode usá-los para prever o que cada cliente tende a comprar a seguir. Uma fábrica tem dados de sensores instalados nas suas máquinas — e pode usá-los para detectar sinais precoces de avaria, antes que ela aconteça e pare a produção. Um consultório médico, com os devidos cuidados de privacidade e anonimização, tem registos de consultas que podem revelar padrões úteis sobre a evolução de certas condições.

A vantagem destes dados é evidente: são **reais e específicos** ao problema concreto que se quer resolver, sem o ruído de informação irrelevante. A desvantagem é igualmente real: costumam estar **desorganizados** — folhas de cálculo com formatação inconsistente, bases de dados com campos em branco, registos duplicados — e, frequentemente, são **demasiado escassos**. Se uma pequena empresa só tem mil registos de vendas, isso simplesmente não chega para treinar um modelo complexo; é preciso, nesses casos, recorrer a outras fontes para complementar, ou usar técnicas que aproveitem melhor conjuntos de dados pequenos.

Há ainda uma armadilha comum nesta fonte de dados, que vale a pena assinalar: o

histórico disponível nem sempre representa bem o futuro que se quer prever. Uma seguradora que treine um modelo de risco apenas com os seus clientes históricos pode estar, sem se aperceber, a ensinar ao sistema os mesmos critérios — por vezes injustos, por vezes simplesmente desactualizados — que a própria empresa usou no passado para aceitar ou recusar propostas. Os dados próprios são valiosos precisamente porque reflectem a realidade concreta de um negócio; mas essa mesma especificidade significa que carregam também as decisões, os enviesamentos e até os erros humanos que já estavam embutidos nessa realidade antes de qualquer máquina entrar em cena.

Fonte 3 — Dados comprados (o supermercado)

Existe, hoje, um mercado inteiro dedicado exclusivamente a vender dados. As empresas que operam neste mercado chamam-se, no jargão do sector, *data brokers* — correctores de dados.

Compram e revendem, por exemplo, bases de dados de comentários de produtos recolhidos em plataformas como a Amazon ou o Yelp; bases de fotografias de rostos, usadas para treinar sistemas de reconhecimento facial; ou bases de conversas de chat, usadas para afinar assistentes virtuais. Os preços variam de forma drástica — desde alguns euros por mil registos, para dados genéricos e abundantes, até milhões de euros para bases altamente especializadas e difíceis de obter por outra via.

Há aqui, no entanto, um problema ético que é importante não varrer para debaixo do tapete, porque é um dos debates mais activos em toda a discussão pública sobre inteligência artificial hoje: **muitas destas bases de dados foram recolhidas sem o consentimento explícito, claro e informado das pessoas envolvidas.** Alguém que escreveu um comentário num site de compras há cinco anos provavelmente não imaginou que esse comentário acabaria, mais tarde, a treinar um sistema de IA comercial. A questão “*Até onde podemos usar dados de pessoas reais para treinar máquinas e com que tipo de autorização?*” não tem ainda uma resposta legal e social totalmente consolidada — está, isso sim, a ser activamente discutida em tribunais, em parlamentos (incluindo no quadro do Regulamento Geral de Protecção de Dados europeu) e em órgãos de regulação um pouco por todo o mundo. Vale a pena que qualquer pessoa interessada neste tema acompanhe esta discussão, porque está longe de estar terminada.

Fonte 4 — Dados gerados por IA (a comida reciclada)

Parece, à primeira vista, um pouco estranho, mas uma das formas de obter mais dados

é... usar a própria inteligência artificial para criar dados novos.

Um exemplo simples: imagine que tem mil fotografias de gatos. Pode pegar nessas fotografias e, através de pequenas transformações automáticas — rodar a imagem alguns graus, ampliar, reduzir, alterar ligeiramente o brilho ou o contraste — multiplicar esse número para dez mil variações. A IA nunca viu exactamente estas novas imagens, mas todas derivam, de forma controlada, das fotografias originais. A esta técnica chama-se **aumento de dados** (*data augmentation*, no termo original em inglês) e é uma prática comum e bem estabelecida, sobretudo em visão por computador.

Outro caminho, mais recente e cada vez mais comum: se já tem dez mil comentários de clientes classificados, pode pedir a um sistema de IA já existente que reescreva cada um desses comentários com palavras diferentes, mantendo o mesmo significado e a mesma classificação de sentimento — gerando, assim, cinquenta mil comentários sintéticos a partir dos dez mil originais.

Há aqui, contudo, um cuidado importante a ter: **dados gerados por IA podem amplificar preconceitos já presentes nos dados originais**, em vez de os diluir. Se os dados de partida tinham, por exemplo, mais comentários positivos associados a produtos vendidos a um determinado grupo demográfico do que a outro, os dados sintéticos gerados a partir deles vão herdar esse desequilíbrio e, por vezes, exagerá-lo, porque o processo de geração tende a reforçar os padrões mais fortes que já estavam presentes nos dados de partida.

Fonte 5 — Dados recolhidos por si (a horta caseira)

A forma mais trabalhosa de todas, mas também a que dá mais controlo a quem a usa, é **criar os próprios dados** desde o início.

Isto pode significar contratar pessoas, ou recorrer a plataformas especializadas em microtarefas (a *Amazon Mechanical Turk* é o exemplo mais conhecido internacionalmente), para classificar imagens, transcrever ficheiros de áudio, ou rotular textos. Pode significar criar um formulário onde os utilizadores de uma aplicação fornecem dados directamente, sempre com o devido consentimento informado. Ou pode significar instalar sensores físicos — de temperatura, de movimento, de som — para captar dados directamente do mundo real, num contexto industrial ou de investigação.

A vantagem é clara: sabe exactamente o que está a entregar à sua inteligência artificial, com controlo total sobre a qualidade e a representatividade dos exemplos. A desvantagem

é igualmente clara: é caro, é lento e é difícil de escalar — recolher cem mil exemplos próprios costuma demorar muito mais tempo e custar muito mais dinheiro, do que descarregar cem mil exemplos já existentes numa base de dados pública.

O tamanho importa — mas não da forma que se pensa

Quantos dados são, de facto, necessários? A resposta não segue uma regra simples e linear, mas há padrões reconhecíveis.

Para uma tarefa relativamente simples — por exemplo, distinguir um e-mail de spam de um e-mail legítimo — alguns milhares de exemplos costumam já ser suficientes para obter resultados razoáveis. Para uma tarefa de complexidade intermédia — como reconhecer dígitos manuscritos, um clássico exercício introdutório nesta área — são tipicamente necessárias dezenas de milhares de exemplos. Para algo da escala dos grandes modelos de linguagem actuais, os números saltam para uma ordem de grandeza completamente diferente: biliões de palavras, processadas durante o treino — uma quantidade de texto que nenhuma pessoa conseguiria ler na totalidade ao longo de uma vida inteira e que continua a crescer com cada nova geração de modelos.

Mas há um limite a este princípio de “mais dados, melhor resultado” e vale a pena conhecê-lo desde já, porque vamos voltar a ele mais adiante no livro. Chega-se, eventualmente, a um ponto em que acrescentar mais dados já não traz melhorias visíveis — o sistema já absorveu, do tipo de exemplo em questão, tudo o que havia de relevante para absorver. A este fenómeno chama-se **saturação**. Não significa que mais dados sejam inúteis em absoluto; significa apenas que, a partir de certo ponto, o investimento em mais dados deixa de compensar tanto quanto investir noutras coisas — num modelo melhor, por exemplo, ou em dados de melhor qualidade, em vez de simplesmente mais dados do mesmo tipo.

Há também uma distinção importante entre quantidade e diversidade, que costuma escapar a quem está a começar nesta área. Dez milhões de exemplos praticamente idênticos entre si ensinam muito menos a um sistema do que cem mil exemplos genuinamente variados. Se uma empresa de seguros quiser treinar um modelo para avaliar pedidos de indemnização e os seus dados históricos forem, na sua grande maioria, sobre acidentes de automóvel num único distrito do país, o sistema vai aprender muito bem a lidar com esse tipo específico de caso — e vai, provavelmente, sair-se mal perante um pedido de indemnização relacionado com um incêndio, ou oriundo de uma região com características de tráfego completamente diferentes. A quantidade resolve um problema; a diversidade

resolve outro e raramente se substituem uma à outra.

Experiência de Cozinha

Visite o sítio do Kaggle e explore, sem necessidade de descarregar nada, a secção de conjuntos de dados públicos. Procure por um tema que lhe interesse pessoalmente — desportos, cinema, gastronomia, futebol português — e veja como esses dados estão organizados em colunas e linhas, prontos a ser usados. Repare, em particular, na secção de comentários e discussões de cada conjunto de dados: é frequente encontrar ali outros utilizadores a apontar problemas de qualidade, valores em falta, ou enviesamentos — uma boa forma de ver, na prática, porque é que “limpar dados” ocupa tanto tempo do trabalho de um cientista de dados.

O RESUMO DO TOTÓ

Os dados que alimentam uma inteligência artificial vêm, no essencial, de cinco fontes: dados públicos, disponíveis gratuitamente mas normalmente sujos e a precisar de limpeza; dados próprios de uma empresa ou projecto, específicos mas muitas vezes escassos e desorganizados; dados comprados a correctores especializados, rápidos de obter mas com sérias questões éticas associadas ao consentimento; dados gerados por outra inteligência artificial, úteis para multiplicar exemplos mas com risco de amplificar preconceitos já existentes; e dados recolhidos de propósito, os mais controlados, mas também os mais caros e lentos a produzir. Quanto mais dados de qualidade, melhor o resultado — mas chega-se sempre a um ponto de saturação, a partir do qual mais dados do mesmo tipo trazem cada vez menos benefício. A fonte mais usada para os grandes sistemas actuais continua a ser, de longe, a internet pública — com tudo o que isso traz de riqueza e de ruído.

CAPÍTULO 5

Onde Mora a Informação

(Ou: a viagem dos dados, desde o disco rígido até ao momento em que respondem a uma pergunta)

Uma pergunta enganadoramente simples

Onde é que a inteligência artificial “guarda” o que sabe? É uma pergunta que parece ter uma resposta óbvia — “num computador, claro” — mas que, examinada com mais atenção, revela uma distinção fascinante entre dois momentos completamente diferentes na vida de um sistema de IA: o momento em que está a aprender e o momento em que já aprendeu e está a ser usado.

Este capítulo segue essa viagem em duas etapas. Primeiro, vamos ver onde vivem os dados *durante* o treino — quando o nosso polvo ainda está a ajustar ventosas, a corrigir erros, a tornar-se gradualmente competente. De seguida, vamos ver o que acontece *depois* do treino, quando o polvo já está pronto e precisa apenas de consultar aquilo que aprendeu para responder a uma pergunta.

Durante o treino — uma biblioteca em obras

Imagine uma biblioteca enorme, com milhões de livros — é a imagem que vamos usar para os dados de treino. Durante o processo de treino descrito no Capítulo 2, esta biblioteca precisa de estar continuamente acessível: o sistema vai buscar um livro, lê-o, aprende algo com ele, devolve-o à estante e vai buscar o seguinte — repetidas vezes, milhões de vezes, ao longo de semanas ou meses.

Há uma característica desta biblioteca que vale a pena destacar desde já, porque ajuda a perceber por que motivo o treino de uma IA grande é tão demorado: o polvo não lê cada livro apenas uma vez. Ao contrário de uma pessoa, que normalmente lê um romance do início ao fim e passa para o seguinte, o treino de uma rede neuronal costuma envolver passar pela biblioteca inteira várias vezes — cada passagem completa chama-se, no jargão técnico, uma *época* (do inglês *epoch*). Cada vez que o polvo relê o mesmo conjunto de livros, ajusta um pouco mais as suas ventosas, refinando gradualmente os seus pesos. É como se um estudante decidisse reler o mesmo manual cinco ou dez vezes, em vez de uma só, na esperança de absorver cada vez mais nuances a cada nova leitura — só que, no caso do polvo, este processo de releitura pode repetir-se centenas de vezes e com milhões de

“livros” em vez de um manual só.

Onde fica, fisicamente, esta biblioteca? Normalmente em discos rígidos de grande capacidade, ou em serviços de armazenamento na nuvem geridos por empresas como a Amazon, a Google ou a Microsoft — os mesmos serviços que mencionámos no Capítulo 3 a propósito de alugar capacidade de computação. Estes discos guardam os dados de forma permanente: se o computador for desligado, os dados continuam lá, intactos, à espera de serem lidos novamente.

Mas há um problema prático com discos, mesmo os mais rápidos: **ler e escrever neles demora tempo** — não muito, em termos humanos, mas uma eternidade em termos do que uma rede neuronal precisa para fazer milhões de cálculos por segundo. Por isso, durante o treino, os dados não ficam parados no disco enquanto o polvo trabalha. São transferidos, em pequenos lotes, para uma memória muito mais rápida — e é aqui que chegamos a uma das distinções técnicas mais importantes e mais frequentemente mal explicadas, deste mundo.

A memória rápida — RAM e a sua irmã mais veloz

Quando se fala, em conversas informais sobre informática, de “memória rápida”, a maioria das pessoas pensa imediatamente na **RAM** — a memória de trabalho do computador, que já mencionámos no Capítulo 3. E, de facto, para muitos sistemas mais pequenos, é exactamente isso que acontece: os dados são lidos do disco e colocados na RAM, de onde o processador os pode usar rapidamente.

Mas para os sistemas grandes de hoje — os modelos de linguagem que sustentam os assistentes de conversação, ou os sistemas de reconhecimento de imagem mais sofisticados — a memória que realmente importa não é a RAM do computador, mas uma irmã mais veloz e mais especializada: a **VRAM**, a memória das próprias placas gráficas (as GPUs que conhecemos no Capítulo 3). As GPUs foram desenhadas, desde o início, para processar enormes quantidades de números em paralelo e a sua memória interna está otimizada precisamente para alimentar esse processamento sem atrasos. Quando um modelo grande está a treinar, é normalmente para dentro da VRAM que os dados e os pesos do polvo são colocados e é ali, junto ao próprio motor de cálculo, que a maior parte do trabalho pesado acontece.

Vale a pena fixar esta distinção, porque vai ser útil para o resto do livro: há sempre um sítio onde a informação “mora” de forma permanente e estável (o disco, a biblioteca de

estantes fixas) e um sítio onde a informação é “trabalhada” activamente, durante uns segundos ou minutos, a uma velocidade muito maior (a memória rápida — RAM ou VRAM, dependendo da escala do sistema). É exactamente a diferença entre um livro na estante e o mesmo livro aberto, em cima da mesa, enquanto alguém o está a ler.

Para sentir a diferença de velocidade envolvida, vale a pena um exemplo concreto. Ler um dado a partir de um disco rígido tradicional pode demorar alguns milissegundos — o suficiente para que, à escala de milhões de operações por segundo que uma rede neuronal exige, todo o sistema fique parado à espera, como um cozinheiro de mãos cruzadas enquanto o forno aquece. Ler o mesmo dado já presente na memória rápida de uma GPU demora uma fracção minúscula desse tempo — várias ordens de grandeza mais rápido. É esta diferença de velocidade, multiplicada por milhões de operações repetidas durante o treino, que explica por que motivo simplesmente “deixar os dados no disco e ler directamente de lá” tornaria o treino de uma IA grande impraticável, levando não semanas, mas possivelmente anos.

Depois do treino — os pesos como um livro fechado

Eis o momento em que a história dá uma volta interessante e que costuma surpreender quem está a começar a interessar-se por este tema. Quando o treino termina, toda a biblioteca de dados originais — os milhões de exemplos que o polvo usou para aprender — já não é necessária para o sistema funcionar.

O que resta e o que de facto importa a partir deste ponto, é um único ficheiro: o ficheiro de **pesos**, que já conhecemos do Capítulo 2. Esse ficheiro é como um livro extremamente denso, escrito inteiramente em números, que resume tudo o que o sistema “aprendeu” a partir daquela biblioteca enorme — sem conter, contudo, nenhum dos livros originais. É uma diferença que vamos explorar com muito mais detalhe no Capítulo 7, mas que vale a pena adiantar aqui: o polvo não guarda a biblioteca, guarda apenas o resumo numérico que dela extraiu.

Este ficheiro de pesos é guardado, tal como vimos no Capítulo 3, num disco — fixo, estático, sem necessidade de qualquer ligação activa para sobreviver. Pode estar num computador pessoal, num servidor de uma empresa, ou distribuído por vários centros de dados espalhados pelo mundo, no caso dos sistemas mais populares e mais usados em simultâneo por milhões de pessoas.

O momento da resposta — ir buscar o livro à estante

Agora chegamos ao momento que mais interessa ao utilizador comum: o que acontece quando alguém escreve uma pergunta num assistente de conversação e prime a tecla de enviar?

O processo segue, em traços largos, estes passos: o sistema vai buscar o ficheiro de pesos ao disco onde está guardado; carrega-o para a memória rápida — a RAM, no caso de sistemas mais pequenos, ou a VRAM das GPUs, no caso dos sistemas grandes que sustentam os assistentes de conversação actuais; e, com o polvo agora “desperto” e operacional naquela memória rápida, processa a pergunta e produz uma resposta. Vamos detalhar, com todo o cuidado, o que significa exactamente “processar a pergunta” no próximo capítulo e no Capítulo 8; por agora, o que importa fixar é apenas a viagem do ficheiro — do disco para a memória rápida e dali para a resposta que aparece no ecrã.

Há um detalhe interessante a notar aqui: para sistemas muito grandes, manter o ficheiro de pesos permanentemente carregado na memória rápida, pronto a responder a qualquer pergunta vinda de qualquer pessoa no mundo, é uma operação cara — em termos de electricidade, de equipamento e de gestão de muitos pedidos a chegar ao mesmo tempo. É por isso que empresas que oferecem assistentes de conversação investem somas consideráveis em infra-estrutura: não basta ter o ficheiro de pesos pronto numa gaveta qualquer; é preciso ter centenas ou milhares de GPUs prontas a “despertar” o polvo, instantaneamente, sempre que alguém faz uma pergunta — e a fazê-lo, em simultâneo, para milhões de pessoas diferentes.

Esta questão da simultaneidade merece um parêntesis, porque toca directamente numa pergunta que muita gente faz sem saber bem como formulá-la: *“Quando eu e um amigo perguntamos a mesma coisa ao mesmo assistente, ao mesmo tempo, estamos a falar com ‘o mesmo polvo’?”* A resposta é, ao mesmo tempo, sim e não. Os pesos usados para responder a ambos são exactamente os mesmos — o mesmo ficheiro, sem qualquer diferença. Mas, fisicamente, é provável que cada pergunta esteja a ser processada numa GPU diferente, ou num conjunto diferente de GPUs, numa sala de servidores qualquer, possivelmente em países diferentes. É como se houvesse não um único polvo, mas milhares de cópias idênticas do mesmo polvo, todas com a mesma personalidade e o mesmo conhecimento, a trabalhar em paralelo, cada uma a atender um pedido diferente. Esta capacidade de multiplicar cópias idênticas de um sistema já treinado, sem ter de repetir o processo de treino, é uma das razões pelas quais um único modelo pode, de facto, atender milhões de conversas em simultâneo, todos os dias, sem que cada conversa “espere a vez” das outras.

Uma biblioteca sem bibliotecário fixo

Vale a pena fechar este capítulo com uma reflexão que ajuda a desfazer um equívoco comum. Quando perguntamos a um assistente de conversação algo e ele responde com aparente segurança, é tentador imaginar que existe, em algum lugar, uma biblioteca organizada onde a resposta estava à espera de ser encontrada — como se o sistema tivesse ido “procurar” a informação correcta numa estante específica.

Não é assim que funciona e esta é, talvez, a ideia mais importante a reter deste capítulo. Não há uma estante com a etiqueta “capital de Portugal” onde o sistema vá directamente buscar a resposta “Lisboa”. O que existe é um único ficheiro de números — os pesos — onde o conhecimento sobre Portugal, sobre capitais, sobre geografia em geral, está disperso e entrelaçado com tudo o resto que o sistema aprendeu, de uma forma que não corresponde a nenhuma organização por gavetas ou estantes que um humano reconheceria. A “biblioteca” deste capítulo é, na verdade, uma metáfora útil apenas para a fase de treino — depois disso, a informação deixa de estar organizada como livros em estantes e passa a estar comprimida, de forma muito mais densa e muito menos legível, dentro da personalidade de cada ventosa do nosso polvo. É exactamente este mistério — como é que números aparentemente desorganizados conseguem produzir respostas tão coerentes — que vamos começar a desvendar no Capítulo 7.

Esta distinção tem ainda uma consequência prática que vale a pena sublinhar e à qual voltaremos com mais detalhe no Capítulo 9: se a informação não está organizada em estantes consultáveis, então a única forma de a IA “saber” algo é tê-lo absorvido, de forma difusa, durante o treino. Isto significa que um sistema treinado, por exemplo, até ao final de um determinado ano não tem qualquer forma de saber, a partir dos seus próprios pesos, do que aconteceu depois disso — não porque se tenha “esquecido”, mas porque essa informação simplesmente nunca esteve na biblioteca que leu. Para resolver esta limitação, existe uma técnica que permite a um sistema ir literalmente “à biblioteca” consultar informação fresca antes de responder, em vez de depender apenas do que absorveu durante o treino — e é exactamente isso que vamos explorar, com uma analogia que retoma esta mesma imagem da biblioteca, no Capítulo 9 deste livro.

Experiência de Cozinha

Sem qualquer instalação, abra as propriedades de armazenamento do seu telemóvel ou computador e procure o tamanho de uma aplicação de música ou de fotografia que tenha instalada — normalmente entre algumas dezenas e algumas centenas de megabytes. Agora compare mentalmente com os números deste capítulo: um modelo de linguagem pequeno pode ocupar um espaço semelhante; um modelo grande, dos que sustentam os assistentes de conversação mais usados, pode ocupar uma quantidade de espaço centenas ou mesmo milhares de vezes maior. É uma forma simples de sentir, na pele do seu próprio telemóvel, a escala de que estamos a falar quando dizemos que os pesos de uma LA “ocupam muito espaço”.

O RESUMO DO TOTÓ

Durante o treino, os dados vivem numa espécie de biblioteca em obras — guardados de forma permanente em discos ou na nuvem, mas constantemente transferidos para memória rápida (a RAM, ou a VRAM das placas gráficas, no caso de sistemas grandes) para serem processados a alta velocidade. Depois do treino, essa biblioteca inteira deixa de ser necessária: o que resta é um único ficheiro de pesos, guardado de forma estática num disco, que resume tudo o que foi aprendido sem conter nenhum dos dados originais. Quando alguém faz uma pergunta, esse ficheiro é carregado para a memória rápida, “desperta” o polvo e permite-lhe processar e responder. Não há, em lado nenhum deste percurso, uma estante organizada onde a resposta esteja à espera — apenas um emaranhado denso de números, onde o conhecimento está disperso de uma forma que nenhuma biblioteca humana reconheceria.

CAPÍTULO 6

Como a Comida é Cortada em Pedacinhos

(Ou: o que acontece aos dados antes de chegarem à boca do polvo)

Uma cozinha exige ingredientes preparados

Nenhum cozinheiro, por mais talentoso que seja, atira uma galinha inteira, com penas e tudo, directamente para o tacho. Há sempre uma fase de preparação: depenar, cortar, medir, pesar. A inteligência artificial funciona de forma semelhante. Antes de qualquer dado — um texto, uma fotografia, um ficheiro de áudio, uma tabela de números — poder entrar pela boca do nosso polvo, precisa de passar por uma fase de preparação cuidadosa, que o transforma em algo que ele consegue, de facto, processar.

Este capítulo é sobre essa preparação. Vamos ver como diferentes tipos de informação — texto, imagem, som, tabelas — são cortados em pedaços e convertidos em números e o que se considera “ruído” a eliminar antes de servir o prato.

Como se corta o texto — a história dos tokens

Comecemos pelo tipo de dado mais comum nos sistemas de IA que hoje usamos no dia a dia: o texto. Quando escrevemos uma pergunta a um assistente de conversação, essa frase não chega ao polvo como uma sequência de letras, à maneira como nós a lemos. É primeiro cortada em pedaços mais pequenos, chamados **tokens**.

Um token não é, ao contrário do que se poderia pensar, necessariamente uma palavra inteira. Pode ser uma palavra completa, como “casa”; pode ser apenas parte de uma palavra, como “in” e “feliz” dentro da palavra “infeliz”; e pode até ser um único sinal de pontuação, como uma vírgula ou um ponto de interrogação. A razão para este corte, por vezes contra-intuitivo, é prática: cortar em pedaços mais pequenos do que palavras inteiras permite ao sistema lidar com palavras que nunca viu durante o treino — se aparecer uma palavra rara ou inventada, ainda assim é possível decompô-la em pedaços mais comuns que o sistema já conhece, em vez de ficar completamente perdido.

Depois de cortado em tokens, cada um deles é convertido num número — ou, mais precisamente, numa lista de números a que se chama **vector**, ou **embedding** no termo técnico mais usado. Esta lista de números não é arbitrária: representa, de uma forma comprimida, o significado daquele token e a sua relação com outros tokens. Palavras com

significados parecidos — “rei” e “monarca”, por exemplo — tendem a ter listas de números parecidas entre si; palavras com significados muito diferentes — “rei” e “bicicleta” — tendem a ter listas de números bastante distintas. É como se cada token recebesse umas coordenadas num mapa gigantesco, onde a proximidade geográfica corresponde a proximidade de significado. Vamos voltar a este mapa com mais detalhe no Capítulo 8, quando explorarmos como o polvo usa estes números para “ler” uma pergunta inteira.

Vale a pena demorar um pouco mais nesta ideia do “mapa de significados”, porque é uma das mais bonitas e mais úteis, de toda a inteligência artificial moderna. Durante o treino, o sistema vai ajustando as coordenadas de cada token de tal forma que padrões de uso semelhante acabem perto uns dos outros nesse mapa. Não é um humano a decidir, à mão, que “rei” deve ficar perto de “rainha” — é o próprio processo de treino, ao observar milhões de frases onde estas palavras aparecem em contextos parecidos, que vai empurrando as suas coordenadas para regiões vizinhas do mapa. O resultado, surpreendentemente, é que este mapa acaba por capturar relações bastante sofisticadas: se subtrairmos as coordenadas de “homem” às de “rei” e somarmos as coordenadas de “mulher”, o ponto resultante no mapa fica, muitas vezes, muito próximo das coordenadas de “rainha”. Ninguém ensinou explicitamente esta relação ao sistema; ela emergiu, por si, do simples facto de processar uma quantidade imensa de texto onde estes padrões de uso se repetiam.

Como se corta uma imagem — píxeis e mais píxeis

Uma fotografia, ao contrário do texto, já chega ao computador num formato relativamente próximo de números — afinal, qualquer imagem digital é, no fundo, uma grelha de pequenos pontos de cor, chamados píxeis. Uma fotografia comum, tirada com um telemóvel actual, pode ter milhões destes pontos, cada um com um valor numérico que representa a sua cor — tipicamente uma combinação de vermelho, verde e azul, cada um numa escala que vai, por convenção, de zero a duzentos e cinquenta e cinco.

O processo de preparação de uma imagem para a inteligência artificial costuma envolver alguns passos adicionais: redimensionar a imagem para um tamanho uniforme, porque o polvo precisa de receber sempre o mesmo número de píxeis de cada vez; normalizar os valores de cor, para que fiquem todos numa escala comum, mais fácil de processar matematicamente; e, por vezes, dividir a imagem em pequenos blocos quadrados, que são depois processados em conjunto através de uma técnica especializada — algo que vamos tocar de novo, brevemente, quando falarmos das camadas do polvo no Capítulo 8.

Há ainda uma decisão de preparação que costuma surpreender quem está a começar nesta área: muitas vezes, antes mesmo de a imagem entrar no polvo, é deliberadamente alterada de várias formas — espelhada, rodada ligeiramente, com o brilho ajustado para mais ou para menos — e cada uma destas versões alteradas é tratada como um exemplo de treino adicional. Já tínhamos visto esta técnica, chamada aumento de dados, no Capítulo 4, a propósito de multiplicar exemplos disponíveis; aqui, no momento da preparação propriamente dita, é onde essas transformações são, de facto, aplicadas, frequentemente de forma automática e em grande quantidade, sem que seja preciso qualquer intervenção manual em cada imagem individual.

Como se corta o som — do ouvido ao espectrograma

O áudio segue uma lógica de preparação diferente e talvez menos intuitiva. Um som, fisicamente, é uma onda — uma variação de pressão no ar ao longo do tempo. Para a inteligência artificial conseguir trabalhar com isto, o primeiro passo é cortar essa onda contínua em pequenos segmentos de tempo, normalmente de poucos milissegundos cada.

Cada um destes pequenos segmentos é depois convertido numa representação chamada **espectrograma** — uma espécie de gráfico que mostra quais as frequências (graves, médias, agudas) presentes naquele instante específico e com que intensidade. O resultado final é uma grelha de números — tempo num dos eixos, frequência no outro — que, curiosamente, o polvo acaba por tratar de uma forma muito parecida à de uma imagem: como uma grelha de “píxeis”, só que de som em vez de cor. Uma palavra falada, como “olá”, que dura meio segundo, pode transformar-se, depois deste processo, numa grelha com algumas centenas de números — uma representação bem mais densa do que a simples sequência de letras “o-l-á”, mas que captura nuances de pronúncia, tom e entoação que o texto escrito simplesmente não tem forma de registar.

Como se corta uma tabela — o caso mais simples e o mais traiçoeiro

Há ainda um tipo de dado que parece, à primeira vista, o mais simples de todos: uma tabela, do tipo que se encontra numa folha de cálculo, com colunas como “Idade”, “Salário” ou “Número de Compras”. Aqui, os dados já chegam em forma de números — não é preciso cortar texto em tokens, nem decompor imagens em píxeis.

Mas há armadilhas a evitar e vale a pena conhecê-las, porque são erros comuns mesmo entre quem já tem alguma experiência. A primeira surge quando uma coluna contém **categorias** em vez de números — por exemplo, uma coluna “Profissão” com valores como

“Engenheiro”, “Médico” ou “Professor”. Não se pode simplesmente entregar estas palavras directamente ao polvo; é preciso convertê-las em números de uma forma que não introduza relações falsas. A técnica mais comum, chamada *one-hot encoding*, atribui a cada categoria uma posição própria numa lista de números: “Engenheiro” pode tornar-se $[1, 0, 0]$, “Médico” $[0, 1, 0]$ e “Professor” $[0, 0, 1]$ — garantindo que nenhuma categoria parece, matematicamente, “maior” ou “mais próxima” de outra sem razão.

A segunda armadilha é a **escala**. Se uma coluna tem valores entre zero e cem (a idade, por exemplo) e outra tem valores entre mil e dez mil (o salário mensal, em euros), o polvo pode acabar a dar mais peso ao salário simplesmente porque os números são maiores em magnitude — não porque seja, de facto, mais importante para o problema em questão. Por isso, é prática comum **normalizar** todos os valores para uma escala comum, normalmente entre zero e um, antes de os entregar ao modelo.

A terceira armadilha são os **valores em falta**. Se uma linha da tabela não tem um valor preenchido na coluna “Idade”, há que decidir o que fazer: preencher com a média de todas as outras linhas, criar uma categoria especial para “desconhecido”, ou, em certos casos, simplesmente eliminar essa linha do conjunto de dados. Cada uma destas decisões tem consequências subtis sobre o resultado final e escolher mal pode introduzir distorções difíceis de detectar mais tarde.

O que o polvo faz com estes números — uma antecipação breve

Depois de qualquer um destes tipos de dado estar convertido em números — tokens transformados em vectores, píxeis organizados numa grelha, sons convertidos em espectrogramas, tabelas normalizadas e codificadas — o polvo está, finalmente, pronto para começar o seu trabalho. O que ele faz, em essência, é multiplicar e somar repetidamente: cada ventosa pega nos números que recebe, multiplica-os pelos seus pesos, soma o resultado e passa-o para a ventosa seguinte. Vamos explorar este processo em detalhe completo no Capítulo 8, que é o coração da Parte III deste livro; por agora, basta reter que esta preparação cuidadosa — o assunto deste capítulo — é o que torna esse processamento posterior possível. Sem ela, o polvo simplesmente não teria nada compreensível para processar.

O ruído é inimigo

Vale a pena fechar este capítulo com uma ideia que atravessa todos os tipos de dados que vimos: nem tudo o que está presente num exemplo é, de facto, útil para a aprendizagem

— e parte da preparação consiste, precisamente, em identificar e remover aquilo a que se chama **ruído**.

Em texto, palavras extremamente comuns como “a”, “o” ou “de” são, por vezes, removidas antes do processamento — chamam-se *stopwords* e a ideia é que aparecem por todo o lado, sem acrescentar muito significado distintivo. Em imagens, um fundo desfocado ou um objecto irrelevante pode confundir o sistema: há casos documentados de modelos que aprenderam a associar a palavra “gato” não ao animal em si, mas à carpete onde os gatos das fotografias de treino costumavam estar sentados — uma associação que parece funcionar bem nos dados de treino, mas que falha de forma surpreendente perante uma fotografia de um gato sentado, por exemplo, numa cama. Em áudio, ruído de fundo genuíno — o som de trânsito, de vento, de outras conversas — pode dificultar a tarefa do sistema, que precisa de aprender a distinguir o sinal relevante (a voz que importa) do resto.

Esta fase de limpeza, que já mencionámos a propósito dos dados públicos no Capítulo 4, ocupa, segundo a estimativa que vimos no Capítulo 2, cerca de 30% do tempo de um cientista de dados — mais do que qualquer outra etapa individual do processo. É um trabalho pouco vistoso, raramente mencionado nas notícias sobre os avanços espectaculares da inteligência artificial, mas é, em larga medida, o que separa um sistema que funciona bem de um sistema que falha de formas estranhas e difíceis de explicar.

Vale a pena um último exemplo, porque ilustra bem como o ruído pode esconder-se em lugares inesperados. Imagine uma empresa portuguesa a treinar um sistema para classificar comentários de clientes como positivos ou negativos, usando dados recolhidos ao longo de vários anos. Se, por coincidência, a maior parte dos comentários negativos recolhidos numa determinada altura vier de um período em que a empresa teve um problema temporário de entregas, o sistema pode acabar a aprender uma associação espúria — por exemplo, que certas palavras relacionadas com a época do ano (Natal, Verão) estão ligadas a sentimento negativo, quando na realidade a única coisa que aconteceu foi um problema logístico pontual, sem qualquer relação real com a estação. Detectar e corrigir este tipo de ruído escondido exige, muitas vezes, um conhecimento profundo do contexto do negócio — não basta saber programar; é preciso também entender a história por detrás dos números.

Experiência de Cozinha

Sem qualquer instalação, escreva uma frase curta num motor de busca por “tokenizer online” ou “GPT tokenizer” e procure uma das várias ferramentas gratuitas que mostram, em tempo real, como uma frase é cortada em tokens. Escreva uma frase em português, como “Vejamos a máquina a funcionar” e observe como ela é dividida — provavelmente em pedaços que não correspondem exactamente às palavras que escreveu. É uma forma rápida e gratuita de ver, com os seus próprios olhos, o primeiro passo que qualquer pergunta sofre antes de chegar à boca do polvo.

O RESUMO DO TOTÓ

Antes de qualquer dado poder ser processado por uma inteligência artificial, precisa de ser cortado em pedaços e convertido em números: o texto é dividido em tokens, depois transformados em listas de números chamadas vectores; as imagens são decompostas em píxeis, cada um com um valor de cor; o som é cortado em pequenos segmentos de tempo e convertido em espectrogramas de frequência; e as tabelas precisam de ter as suas categorias codificadas e os seus valores normalizados para uma escala comum. Em todos estes casos, uma parte importante do trabalho consiste em remover ruído — informação irrelevante que pode confundir o sistema em vez de o ajudar. Esta preparação, embora pouco vistosa, é o que transforma dados brutos do mundo real em algo que o nosso polvo de números consegue, finalmente, mastigar.

PARTE III

O Cérebro de Polvo (Como a IA “pensa”)

CAPÍTULO 7

A Memória que Não É Uma Memória

(Ou: porque é que a LA “sabe” tudo e não se lembra de nada)

Um paradoxo que vale a pena sentir antes de explicar

Há algo profundamente estranho na forma como falamos de assistentes de conversação e que normalmente passa despercebido até alguém o assinalar em voz alta. Por um lado, dizemos coisas como “o ChatGPT sabe imensas coisas” ou “o Claude conhece bem este tema” — como se estes sistemas tivessem uma memória vastíssima, cheia de factos arrumados. Por outro lado, se começarmos uma conversa nova com o mesmo assistente, ele não tem a menor ideia de quem somos, do que conversámos ontem, ou do que lhe dissemos há cinco minutos numa janela de conversa diferente.

Como é que uma coisa que “sabe tudo” pode, ao mesmo tempo, não se lembrar de nada? A resposta está numa distinção que é, provavelmente, a ideia mais importante de todo este livro — e que vamos desenrolar com calma ao longo deste capítulo: a inteligência artificial tem, na verdade, **dois tipos de memória completamente diferentes**, que funcionam segundo regras opostas e que é fundamental não confundir.

A primeira memória — os pesos, o disco de vinil

A primeira destas memórias já a conhecemos bem, dos capítulos anteriores: são os **pesos** do nosso polvo — aquele ficheiro denso de números que resulta de todo o processo de treino e que vimos a viajar entre o disco e a memória rápida no Capítulo 5. Vamos agora dar-lhe uma analogia nova, que captura uma propriedade sua particularmente importante: pense nos pesos como um **disco de vinil**.

Um disco de vinil é gravado uma vez, num estúdio, através de um processo cuidadoso e moroso — sulcos são talhados na superfície de forma a codificar, fisicamente, a música que foi tocada. Depois de gravado, o disco está terminado. Pode tocá-lo mil vezes, em mil gira-discos diferentes, em mil casas diferentes e vai ouvir sempre exactamente a mesma música — porque os sulcos não mudam com a reprodução. Ouvir o disco não altera o disco.

Os pesos de uma inteligência artificial comportam-se exactamente assim. São “gravados” durante o treino — um processo que pode demorar semanas ou meses e que

envolve ajustar repetidamente as ventosas do polvo até elas captarem os padrões certos, como vimos no Capítulo 2. Uma vez terminado o treino, esse ficheiro de pesos fica fixo. Quando alguém faz uma pergunta ao sistema, os pesos são “tocados” — usados para processar essa pergunta e gerar uma resposta — mas essa reprodução não altera, em nada, o disco original. A próxima pessoa que fizer a mesma pergunta vai obter, partindo dos mesmos pesos, uma resposta gerada de forma essencialmente equivalente, porque está a “tocar” exactamente o mesmo disco.

Esta é, provavelmente, a propriedade mais difícil de aceitar intuitivamente sobre os assistentes de conversação modernos: **conversar com eles não os ensina nada**. Pode ter uma conversa fascinante, profundamente pessoal, cheia de detalhes sobre a sua vida — e, no minuto seguinte, abrir uma nova conversa e o sistema vai responder como se nunca tivesse ouvido falar de si. Não é mau humor, nem esquecimento no sentido humano da palavra. É, simplesmente, a natureza do disco de vinil: ele não se grava a si próprio enquanto está a tocar.

Vale a pena insistir neste ponto com um exemplo mais próximo do quotidiano, porque é, de facto, contra-intuitivo. Imagine que um professor de português corrige, com cuidado, a redacção de um aluno, explicando-lhe porque é que “haver” e “a ver” se escrevem de forma diferente. Nós esperamos, naturalmente, que esse aluno tenha aprendido algo — que a próxima redacção venha melhor. Quando uma pessoa explica um erro semelhante a um assistente de conversação, dentro de uma conversa (*“não, está errado, é assim que se escreve”*), o sistema vai, de facto, ajustar-se *dentro daquela conversa* — vai parar de repetir o erro enquanto a mesa de trabalho que descrevemos a seguir ainda tiver essa correcção visível. Mas, abrindo uma conversa nova no dia seguinte, o erro vai voltar a acontecer, exactamente da mesma forma, como se a correcção nunca tivesse existido. O disco de vinil não gravou nada; apenas, durante uns minutos, alguém colocou uma nota em cima da mesa.

O músico de jazz — uma segunda imagem para a mesma ideia

Há uma segunda analogia que ajuda a aprofundar esta mesma propriedade e que vale a pena explorar em paralelo à do disco de vinil, porque ilumina um ângulo diferente do problema: o **músico de jazz**.

Um bom músico de jazz, depois de anos de prática, não decorou milhares de partituras exactas para tocar nota a nota perante qualquer pedido. O que fez foi internalizar padrões — escalas, progressões de acordes, frases melódicas típicas, formas habituais de responder musicalmente a outro instrumentista. Quando lhe pedem para improvisar sobre um tema

que nunca tocou antes, ele não está a “procurar” essa música específica numa biblioteca mental de canções memorizadas; está a aplicar, de forma fluida e quase automática, os padrões que absorveu ao longo de uma vida de prática, adaptando-os ao momento presente.

É exactamente isto que o nosso polvo faz quando responde a uma pergunta. Não está a “consultar” uma base de dados onde a resposta esteja guardada palavra por palavra — já vimos isto, aliás, no Capítulo 5, quando desfizemos a ideia da “biblioteca com etiquetas”. Está, isso sim, a aplicar padrões estatísticos absorvidos durante o treino, padrões sobre como as palavras tendem a seguir-se umas às outras, sobre como factos costumam ser apresentados, sobre como perguntas costumam ser respondidas. Tal como o músico de jazz nunca “decorou” a improvisação que vai tocar esta noite, o polvo nunca “decorou”, literalmente, a frase exacta que vai produzir como resposta — ela emerge, em tempo real, da aplicação desses padrões à pergunta concreta que recebeu.

Esta analogia ajuda também a explicar algo que intriga muita gente: por que motivo a mesma pergunta, feita duas vezes ao mesmo sistema, pode produzir respostas com palavras ligeiramente diferentes, ainda que com o mesmo sentido geral. Tal como dois solos de jazz sobre o mesmo tema musical nunca saem exactamente iguais, mesmo tocados pelo mesmo músico na mesma noite, a “improvisação estatística” do polvo tem uma margem de variação — vamos explorar exactamente de onde vem essa variação no Capítulo 8, quando olharmos para o mecanismo que escolhe, passo a passo, qual a próxima palavra a produzir.

A segunda memória — o contexto, a mesa de trabalho

Há, no entanto, um segundo tipo de memória, que funciona segundo regras completamente diferentes das do disco de vinil — e que explica por que motivo um assistente de conversação consegue, de facto, lembrar-se do que disse há dez minutos, dentro da mesma conversa.

Chama-se **contexto** e a melhor forma de o imaginar é como uma **mesa de trabalho pequena**, à frente do polvo. Tudo o que foi escrito na conversa actual — a sua primeira pergunta, a resposta do sistema, a sua segunda pergunta e assim sucessivamente — fica colocado nessa mesa, à vista, e o polvo pode “olhar” para tudo o que lá está sempre que precisa de gerar a próxima resposta. É por isso que, dentro da mesma conversa, pode dizer “chamo-me Aristides” no início e perguntar “qual é o meu nome?” vinte mensagens depois e o sistema vai responder correctamente — não porque tenha “aprendido” o seu nome, no sentido de ter alterado os seus pesos, mas porque essa informação ainda está, literalmente, em cima da mesa, visível, à espera de ser lida.

Mas há um detalhe crucial sobre esta mesa: **é pequena e tem um limite**. Não cabe nela uma quantidade infinita de papéis. Cada sistema tem aquilo a que se chama uma **janela de contexto** — um número máximo de tokens (lembra-se deles, do Capítulo 6?) que cabem na mesa ao mesmo tempo. Este número tem vindo a crescer de forma considerável ao longo dos últimos anos e continua a crescer com cada nova geração de modelos — por isso não vale a pena fixar aqui um número exacto, que ficaria desactualizado rapidamente; o que importa reter é que existe sempre um limite e que, quando a conversa o excede, algo tem de ser feito. Normalmente, as mensagens mais antigas começam a “cair da mesa” — deixam de estar visíveis para o polvo, ainda que a pessoa que está a conversar continue a vê-las no ecrã, no histórico da conversa.

Isto explica um fenómeno que provavelmente já experimentou, se tiver tido conversas muito longas com um assistente de IA: a sensação de que ele, a certa altura, parece “esquecer-se” de algo que foi dito muito no início. Não é, tecnicamente, esquecimento no sentido de uma falha — é, simplesmente, que aquela informação caiu da mesa de trabalho e já não está disponível para o polvo consultar.

Há ainda um detalhe sobre esta mesa de trabalho que vale a pena esclarecer, porque costuma gerar confusão: quando se fala de “memórias” que alguns assistentes de conversação oferecem actualmente — funcionalidades que permitem ao sistema lembrar-se de certas preferências entre conversas diferentes — não se está, na verdade, a violar a regra do disco de vinil fixo. O que está a acontecer, nestes casos, é que o sistema, ou a aplicação que o envolve, guarda um pequeno resumo dessas preferências num ficheiro separado, fora do próprio polvo e coloca esse resumo de volta em cima da mesa de trabalho automaticamente, no início de cada conversa nova — como um assistente humano que, antes de o atender, lesse rapidamente uma ficha com notas sobre si, em vez de se lembrar de si pessoalmente. O disco de vinil continua exactamente igual; apenas se tornou mais eficiente a forma como se prepara a mesa antes de a conversa começar.

A diferença essencial, resumida sem rodeios

Vale a pena fixar, com toda a clareza possível, a diferença entre estas duas memórias, porque é a chave para entender quase tudo o resto sobre como estes sistemas se comportam:

Pesos (disco de vinil)	Contexto (mesa de trabalho)
Fixos depois do treino	Mudam a cada nova mensagem

Enormes — todo o “conhecimento geral” do sistema	Pequenos — limitados a um número de tokens
Partilhados por todas as conversas, com todas as pessoas	Específicos a uma única conversa
Não mudam ao conversar	Crescem (e podem “esvaziar-se” no início) ao conversar
Mudar exige novo treino, lento e caro	Muda automaticamente, a cada mensagem, sem custo

Esta tabela explica, de forma simples, por que motivo a IA parece simultaneamente brilhante e amnésica: o seu “conhecimento geral” está gravado no disco de vinil dos pesos, vasto e estável; a sua “memória da conversa actual” está na mesa de trabalho do contexto, pequena e temporária. São dois sistemas distintos, governados por regras diferentes e a confusão entre eles é, provavelmente, a fonte mais comum de mal-entendidos sobre como a inteligência artificial realmente funciona.

Um risco escondido — o esquecimento catastrófico

Há um fenómeno que vale a pena conhecer antes de avançarmos para a questão de como, de facto, se altera o disco de vinil, porque ilustra bem como esta analogia, apesar de útil, tem os seus limites — e os seus perigos reais.

Quando se faz fine-tuning a um modelo já treinado — a técnica que vamos detalhar já a seguir — existe o risco de o processo, ao ensinar o polvo algo novo e específico, acabar por **desaprender** competências antigas que já tinha. Os investigadores chamam a isto **esquecimento catastrófico**: ao reajustar os pesos de forma demasiado intensa para se especializar numa tarefa nova, o polvo pode perder, sem ninguém querer, parte da fluência geral que tinha antes. É como se, ao gravar uma faixa extra num disco de vinil já existente, o processo de gravação acabasse por estragar ligeiramente os sulcos de uma faixa anterior — não era essa a intenção, mas a física do processo trouxe esse efeito secundário.

Por esta razão, quem faz fine-tuning a sério precisa de ter cuidados específicos: usar uma quantidade de dados novos cuidadosamente equilibrada, ajustar os pesos com grande cautela (em passos pequenos, em vez de mudanças bruscas) e testar repetidamente, depois do processo, se o sistema ainda responde bem a perguntas gerais que nada têm a ver com a especialização pretendida. É um equilíbrio delicado entre ensinar algo novo e não destruir o que já lá estava — e é uma das razões pelas quais o fine-tuning, apesar de parecer simples

na teoria, exige bastante perícia na prática.

Quando se quer mesmo mudar o disco — re-treino e fine-tuning

Há, naturalmente, situações em que se quer, de facto, alterar o que o polvo “sabe” de forma permanente — não apenas dentro de uma conversa, mas para todas as conversas futuras. Isto não se consegue através do contexto; exige voltar ao disco de vinil e regravá-lo, ainda que parcialmente.

A forma mais radical de o fazer chama-se **re-treino completo**: repetir todo o processo do Capítulo 2, do zero, com um conjunto de dados actualizado. É como regravar o disco de vinil inteiro, de início ao fim — caro, demorado, mas garante um resultado coerente com a nova informação.

Há, contudo, um caminho intermédio, muito mais usado na prática, chamado **fine-tuning** (ou “afinação fina”, numa tradução possível). Em vez de treinar o polvo do zero, pega-se num modelo já treinado — que já sabe falar português fluentemente, já entende conceitos gerais do mundo — e treina-se apenas um pouco mais, com um conjunto de dados mais pequeno e específico, normalmente relacionado com uma tarefa ou domínio particular. É como gravar uma faixa extra num disco de vinil já existente, ajustando ligeiramente alguns sulcos sem ter de recomeçar a prensagem inteira. Uma empresa que queira um assistente especializado em responder sobre os seus próprios produtos, por exemplo, pode aplicar fine-tuning a um modelo geral já treinado, usando alguns milhares de exemplos sobre o seu negócio específico — um processo muito mais rápido e barato do que o re-treino completo e que já mencionámos, en passant, no Capítulo 3, a propósito dos custos envolvidos.

Experiência de Cozinha

Experimente o seguinte com qualquer assistente de conversação: numa conversa nova, diga o seu nome e uma curiosidade qualquer sobre si — por exemplo, “chamo-me Aristides e tenho um interesse antigo por café expresso”. Faça depois algumas perguntas sem relação alguma com isso, e, várias mensagens mais tarde, pergunte “de que é que eu disse que gostava?”. O sistema deve responder correctamente — está só a ler a mesa de trabalho. Agora abra uma conversa completamente nova, com o mesmo assistente e faça a mesma pergunta directamente. A resposta vai ser, quase certamente, “não sei” ou algo parecido — porque a mesa de trabalho está vazia e o disco de vinil nunca soube nada sobre si em particular.

Um aviso importante para esta experiência resultar como descrito: faça-a numa conta sem funcionalidades de “memória entre conversas” activadas, ou em modo anónimo/de navegação privada. Se a sua conta tiver esse tipo de memória ligada — a funcionalidade que mencionámos há poucos parágrafos, que guarda um pequeno resumo de preferências fora do próprio polvo — é bem possível que o sistema responda correctamente mesmo na conversa nova. Isso não contradiz nada do que foi dito: os pesos continuam tão imutáveis como sempre; é apenas que, antes de a conversa começar, alguém colocou de volta na mesa uma ficha de notas sobre si. Para isolar o efeito do disco de vinil puro, sem essa ajuda extra, é preciso desligar essa ficha.

O RESUMO DO TOTÓ

A inteligência artificial tem dois tipos de memória completamente diferentes. Os pesos são como um disco de vinil: gravados uma vez, durante o treino e depois fixos — conversar com o sistema não os altera e por isso ele “esquece” tudo entre conversas diferentes. O contexto é como uma mesa de trabalho pequena: guarda tudo o que foi dito na conversa actual, mas tem um limite de tamanho e a informação mais antiga acaba por “cair da mesa” em conversas muito longas. O que parece, à primeira vista, um sistema confuso — sabe tudo, mas não se lembra de nada — é, na verdade, perfeitamente coerente, uma vez compreendida esta distinção. Para alterar de forma permanente o que o sistema “sabe”, só há um caminho: voltar ao disco de vinil, através de re-treino completo ou da técnica mais económica chamada fine-tuning — nunca através da simples conversa.

CAPÍTULO 8

Como Ela Lê e Responde a Uma Pergunta

(Ou: a viagem completa de uma frase, desde a sua boca até ao ecrã)

O momento que estávamos a preparar desde o início

Chegámos ao capítulo que dá sentido a tudo o que veio antes. Já sabemos como se cria uma inteligência artificial (Parte I), de onde vêm os seus dados e como são preparados (Parte II) e como funciona a sua memória peculiar, dividida entre o disco de vinil fixo dos pesos e a mesa de trabalho temporária do contexto (Capítulo 7). Falta apenas uma peça: o que acontece, de facto, no momento exacto em que escreve uma pergunta e prime “enviar”.

Este capítulo segue essa viagem do início ao fim, em duas partes. Na primeira, vamos ver como a sua pergunta é transformada em algo que o polvo consegue processar e como ele decide a que partes dela prestar mais atenção. Na segunda, vamos seguir esses números através dos tentáculos do polvo, até chegarmos à palavra que aparece, finalmente, no seu ecrã.

Vale a pena notar, antes de começarmos, que tudo o que vamos descrever aqui acontece a uma velocidade que escapa por completo à nossa intuição humana. O processo inteiro — desde a sua pergunta entrar até a resposta começar a aparecer no ecrã — pode demorar uma fracção de segundo, ou alguns segundos para perguntas mais longas, mas envolve, nesse curto espaço de tempo, milhares de milhões de operações matemáticas. Não há, em nenhum momento, uma pausa para “pensar” no sentido humano; há, isso sim, uma cascata de cálculos que acontece tão depressa que a ilusão de deliberação surge naturalmente, mesmo sem existir deliberação alguma.

Parte A — Da pergunta aos números***O primeiro passo: cortar e mapear***

Já vimos, no Capítulo 6, que qualquer texto é primeiro cortado em tokens e que cada token é depois convertido numa lista de números — um vector, ou embedding — que funciona como umas coordenadas num mapa gigantesco de significados. Vamos pegar num exemplo concreto e segui-lo até ao fim deste capítulo: imagine que escreve a frase “O gato subiu ao telhado”.

Esta frase é cortada em tokens — algo como “O”, “gato”, “subiu”, “ao”, “telhado” — e cada um destes tokens é convertido na sua lista de números correspondente. Até aqui, nada de novo; é exactamente o processo do Capítulo 6. Mas há um problema que ainda não resolvemos: estas listas de números, por si só, não sabem nada sobre a **ordem** das palavras, nem sobre como elas se relacionam **dentro daquela frase específica**. O vector da palavra “gato” é sempre o mesmo, esteja ela na frase “o gato subiu ao telhado” ou na frase “o cão perseguiu o gato” — e, no entanto, o papel que “gato” desempenha nestas duas frases é bastante diferente. É aqui que entra o mecanismo mais importante e mais mal compreendido, de toda a inteligência artificial moderna: a **atenção**.

O mecanismo de atenção — não é um holofote selectivo

Há uma imagem popular e sedutora, para explicar a atenção: a de um holofote que se acende sobre as palavras importantes de uma frase e deixa as restantes na sombra. É uma boa imagem de partida, mas, tal como acontece com muitas simplificações, esconde algo essencial — e vale a pena corrigi-la aqui, porque é uma das ideias mais mal explicadas em todo este tema.

A atenção não escolhe, de antemão, que “gato” é importante e que “ao” não é. Em vez disso, imagine uma sala onde estão sentadas todas as palavras da frase, em volta de uma mesa redonda. Cada palavra “olha” para todas as outras — incluindo para si própria — e atribui a cada uma delas uma certa quantidade de atenção, que pode ser quase nula ou muito elevada, dependendo do que for relevante para compreender aquela palavra naquele contexto específico. Não há, à partida, nenhuma palavra automaticamente ignorada; há, isso sim, um cálculo de relevância relativa entre **todos os pares de palavras** da frase, ao mesmo tempo.

Voltando à nossa frase: ao processar a palavra “subiu”, o mecanismo de atenção pode decidir que “gato” merece muita atenção (porque é o gato que está a subir e “subiu” precisa de saber quem é o sujeito da acção), que “telhado” também merece bastante atenção (porque é o destino da subida) e que “ao” e “o” merecem relativamente pouca — não porque tenham sido excluídos de propósito, mas porque o cálculo de relevância, aplicado a esta frase concreta, atribuiu-lhes pesos baixos. Numa frase diferente, as mesmas palavras gramaticais poderiam receber pesos de atenção completamente distintos, se isso ajudasse a desambiguar o sentido. A “luz”, para retomar a imagem do holofote, está sempre presente em todas as direcções; o que varia é a sua intensidade.

Este processo acontece, simultaneamente, para cada palavra da frase em relação a todas

as outras — não é um único holofote a percorrer a frase sequencialmente, mas centenas ou milhares destes cálculos de relevância a acontecer ao mesmo tempo, numa única passagem. É, provavelmente, a inovação técnica mais importante por detrás dos assistentes de conversação modernos e é também o que permite ao sistema desambiguar frases que, fora de contexto, seriam ambíguas — por exemplo, distinguir se “ela” se refere à “Maria” ou à “casa” mencionadas algumas palavras antes, dependendo de qual delas recebe mais atenção naquele ponto específico da frase.

Vale a pena um segundo exemplo, porque a ambiguidade da língua portuguesa oferece material abundante para isto. Considere a frase “*O banco estava cheio de gente à espera, mas o jardim ao lado estava deserto.*” A palavra “banco” pode, em português, referir-se a uma instituição financeira ou a um assento — e só o resto da frase permite desambiguar qual dos dois sentidos está em causa. O mecanismo de atenção, ao processar “banco”, vai prestar atenção a palavras como “gente à espera”, que são mais compatíveis com o sentido de instituição financeira (ou, dependendo do treino do sistema, com uma fila de espera num espaço público), do que com o sentido de simples assento de jardim. É precisamente este tipo de desambiguação, feita instantaneamente e para milhares de palavras ambíguas em simultâneo, que a atenção torna possível — e que mecanismos anteriores, mais simples e sequenciais, tinham muita dificuldade em replicar com a mesma eficácia.

O resultado — uma frase enriquecida

No final deste processo de atenção, cada palavra da nossa frase deixa de ser representada apenas pelo seu significado isolado (o vector genérico de “gato”, sempre igual em qualquer frase) e passa a ser representada por uma versão **enriquecida** desse vector, que já incorpora informação sobre o seu papel específico naquela frase concreta — uma espécie de “gato, mas especificamente o gato-sujeito-que-subiu-a-um-telhado-nesta-frase”. É esta versão enriquecida e não a versão genérica do Capítulo 6, que vai entrar agora nos tentáculos do polvo, na segunda parte deste capítulo.

Parte B — Dos números à resposta

A viagem pelos tentáculos

Com a nossa frase já convertida em números enriquecidos pela atenção, chegou o momento que vínhamos a prometer desde o Capítulo 2: ver o polvo, finalmente, a trabalhar.

Os números entram pela boca do polvo e começam a viajar pelos tentáculos — as **camadas** da rede neuronal. Em cada tentáculo, encontram centenas ou milhares de

ventosas — os neurónios — e cada ventosa faz, essencialmente, a mesma operação simples: multiplica os números que recebe pelos seus pesos (lembra-se, do Capítulo 7, que os pesos são a “personalidade” fixa de cada ventosa, gravada no disco de vinil), soma tudo e aplica uma pequena transformação matemática chamada **função de activação**, que decide se o resultado deve ser passado com força para a ventosa seguinte ou suavizado.

Esta operação — multiplicar, somar, transformar — repete-se em cada ventosa, em cada tentáculo e os resultados de um tentáculo alimentam o tentáculo seguinte, numa cadeia que pode ter dezenas ou até centenas de camadas de profundidade nos sistemas mais sofisticados de hoje. Cada camada sucessiva costuma captar padrões cada vez mais abstractos: as primeiras camadas podem estar a detectar padrões simples, como combinações frequentes de duas palavras; as camadas do meio podem estar a captar relações gramaticais mais complexas; e as camadas finais podem estar já a trabalhar com algo muito mais próximo do “sentido geral” da frase inteira. Ninguém programou esta divisão de trabalho à mão — ela emergiu, sozinha, do processo de treino, exactamente como vimos no Capítulo 2.

Vale a pena explicar um pouco melhor o que faz, de facto, essa “função de activação” que mencionámos. Sem ela, o polvo inteiro seria, matematicamente, equivalente a uma única operação simples — multiplicar e somar, por mais camadas que tivesse, continuaria a ser, no fundo, apenas isso, incapaz de captar relações complexas e não lineares entre as palavras. A função de activação introduz uma pequena dose de “decisão” em cada ventosa: por exemplo, uma das funções mais usadas, conhecida por um nome técnico que não precisamos de memorizar, comporta-se como um interruptor parcial — deixa passar os números positivos praticamente sem alteração, mas reduz drasticamente os números negativos, quase a anulá-los. É uma operação extremamente simples, repetida biliões de vezes em todo o polvo, mas é exactamente essa simplicidade, multiplicada por uma profundidade enorme de camadas, que permite ao sistema captar padrões muito mais sofisticados do que uma simples soma directa de números conseguiria.

Chegar à resposta — uma previsão, não uma certeza

Eis, finalmente, o momento que mais intriga quem está a começar a interessar-se por este mundo: como é que, depois de tudo isto, o polvo decide o que escrever como resposta?

A resposta é, simultaneamente, simples e surpreendente: o polvo não decide a frase toda de uma vez. Decide, isso sim, **uma palavra a seguir à outra**, fazendo sempre a mesma pergunta a si próprio: *dado tudo o que já foi escrito até agora — a minha pergunta original e a resposta*

que já comecei a construir — qual é o token mais provável a seguir? Depois de gerar esse token, acrescenta-o à frase e volta a fazer a mesma pergunta, agora com um token adicional de contexto. E repete este processo, token a token, até decidir que a resposta está completa.

Mas há uma subtileza importante nesta “previsão da palavra mais provável”, que vale a pena desfazer com cuidado: o polvo não produz, ao certo, uma única palavra — produz, sim, uma lista enorme com **todas** as palavras possíveis do seu vocabulário, cada uma com uma probabilidade associada. Para a nossa frase sobre o gato, depois de “O gato subiu ao”, o sistema pode calcular que “telhado” tem uma probabilidade de, digamos, 40%, que “muro” tem 25%, que “sótão” tem 10% e assim por diante, distribuindo o resto da probabilidade por milhares de outras palavras menos prováveis, mas não impossíveis.

De onde vem a variação — o papel da “temperatura”

Isto explica, finalmente, a observação que deixámos em suspenso no Capítulo 7: por que motivo a mesma pergunta, feita duas vezes, pode gerar respostas com palavras diferentes. Em vez de escolher sempre, de forma mecânica, a palavra com maior probabilidade — o que tornaria todas as respostas previsíveis e repetitivas — os sistemas costumam introduzir um elemento controlado de aleatoriedade nesta escolha, regulado por um parâmetro chamado, no jargão técnico, **temperatura**. Com uma temperatura baixa, o sistema tende a escolher quase sempre a opção mais provável, produzindo respostas mais previsíveis e consistentes. Com uma temperatura mais alta, dá espaço a opções menos prováveis, produzindo respostas mais variadas e por vezes mais criativas — mas também, nalguns casos, mais erráticas. É exactamente a mesma lógica do nosso músico de jazz do capítulo anterior: a improvisação nunca é totalmente aleatória, nem totalmente determinada — há sempre uma margem controlada de variação dentro dos padrões aprendidos.

Uma palavra de cada vez, sem plano global

Há uma última ideia, talvez a mais contra-intuitiva de todo este capítulo, que vale a pena sublinhar antes de fechar: o polvo **não tem, em nenhum momento, um plano completo da resposta antes de começar a escrevê-la**. Não decide, de início, “vou responder com uma frase de dez palavras sobre telhados” e depois preenche os espaços. Decide uma palavra, depois outra, depois outra, sempre a partir do que já escreveu — como alguém que improvisa um discurso em voz alta, sem guião, mas com excelente intuição sobre como uma frase coerente costuma continuar a partir de onde já está.

Isto explica também por que motivo, ocasionalmente, um sistema pode “encurralar-se” numa frase que começa de uma forma e tem dificuldade em terminar de forma elegante

— tal como uma pessoa a falar de improviso pode, por vezes, perder-se a meio de uma frase longa, sem ter planeado bem como a havia de fechar. A diferença é que o polvo faz isto a uma velocidade muitíssimo superior à de qualquer pessoa e com uma taxa de sucesso impressionante — mas o princípio de fundo, uma palavra de cada vez, sem plano global fixo, mantém-se mesmo nos sistemas mais sofisticados de hoje.

Experiência de Cozinha

Peça a um assistente de conversação a mesma pergunta criativa — por exemplo, “conta-me uma história de duas frases sobre um polvo que aprende a tocar jazz” — três vezes seguidas, em três conversas novas e separadas. Compare as três respostas. Vai notar que, embora o tema e o espírito sejam semelhantes, as palavras exactas escolhidas variam de cada vez — uma prova directa, à sua frente, de que o sistema está a gerar a resposta palavra a palavra, com uma margem de variação controlada e não a “ler” uma resposta fixa e única guardada em algum lado.

O RESUMO DO TOTÓ

Quando se faz uma pergunta a uma inteligência artificial, a frase é primeiro cortada em tokens e convertida em números; depois, o mecanismo de atenção recalcula esses números, fazendo cada palavra “olhar” para todas as outras na frase e ajustar o seu significado de acordo com o contexto específico — não há nenhuma palavra ignorada à partida, apenas diferentes intensidades de atenção. Esses números enriquecidos viajam então pelos tentáculos do polvo, sendo transformados em cada ventosa através de multiplicações, somas e pequenas funções de decisão, em camadas cada vez mais abstractas. No final, o sistema não produz a resposta toda de uma vez: gera-a palavra a palavra, escolhendo sempre, com uma margem controlada de variação, a opção mais provável a seguir, dado tudo o que já foi escrito até esse ponto. Não há, em nenhum momento, um plano global da resposta — apenas uma sucessão notavelmente fluida de decisões locais, uma palavra de cada vez.

CAPÍTULO 9

Quando o Polvo Vai à Biblioteca

(Ou: o que acontece quando a IA decide ir verificar antes de responder)

Um problema que já tínhamos deixado em aberto

No Capítulo 5, ao desfazer a ideia de que existe uma biblioteca organizada dentro da inteligência artificial, deixámos uma questão pendente: se o conhecimento do polvo está todo gravado nos pesos, fixo desde o fim do treino, então o que acontece quando alguém pergunta sobre algo que só aconteceu **depois** desse treino ter terminado?

A resposta directa é desconfortável, mas honesta: o polvo, sozinho, não tem qualquer forma de saber. Não é por falta de inteligência, nem por preguiça — é, simplesmente, porque essa informação nunca esteve na biblioteca que ele leu durante o treino. Se um sistema foi treinado com dados até um determinado mês e lhe perguntarmos sobre algo que aconteceu três meses depois, ele está, estruturalmente, tão informado sobre esse acontecimento futuro quanto qualquer livro publicado antes da data em questão.

Há, no entanto, uma solução engenhosa para este problema, que se tornou comum em muitos assistentes de conversação actuais e que é o tema deste capítulo: em vez de depender apenas do que está gravado no disco de vinil, o sistema pode, antes de responder, **ir literalmente à biblioteca** consultar informação fresca. A esta técnica chama-se, no jargão técnico, **RAG** — sigla para *Retrieval-Augmented Generation*, ou, numa tradução possível, “geração de texto reforçada por pesquisa”.

Duas formas de responder — de cabeça ou com consulta

Para compreender o que há de novo no RAG, vale a pena comparar duas situações que, à primeira vista, podem parecer idênticas, mas que são, na verdade, processos bastante diferentes.

Imagine que pergunta a alguém “*Qual é a capital de Portugal?*”. Essa pessoa, se tiver feito a escola primária em Portugal, vai responder **de cabeça**, sem hesitar — “Lisboa” — porque essa informação está tão bem internalizada que nem precisa de pensar nela como um facto separado a recordar. Agora imagine que pergunta a essa mesma pessoa “*Qual foi o resultado do jogo de futebol de ontem à noite?*”. Mesmo sendo uma pessoa muito bem informada, é provável que ela diga “*espera, deixa-me verificar*” e procure a resposta num jornal, num telefone, ou

pergunte a outra pessoa — porque essa informação não estava e não podia estar, já internalizada de antemão.

A inteligência artificial, quando não usa RAG, comporta-se sempre como no primeiro caso: responde de cabeça, usando apenas os pesos gravados durante o treino — exactamente o processo que descrevemos em detalhe no Capítulo 8, com o polvo a processar a pergunta e a prever, palavra a palavra, a resposta mais provável, sem qualquer consulta externa. Quando o sistema usa RAG, comporta-se como no segundo caso: antes de gerar a resposta final, faz uma pausa, vai “verificar” numa fonte externa — pode ser a internet, pode ser uma base de dados específica de uma empresa, pode ser um conjunto de documentos internos — e só depois usa essa informação fresca, juntamente com o que já sabia de cabeça, para construir a resposta.

O processo, passo a passo

Vamos seguir, com algum detalhe, o que acontece quando um sistema com RAG recebe uma pergunta que exige consulta — por exemplo, *“Quais foram os resultados financeiros mais recentes desta empresa?”*.

Primeiro, o sistema reconhece, de alguma forma, que esta pergunta provavelmente beneficia de informação actualizada — um relatório financeiro recente é exactamente o tipo de coisa que pode não estar, ou estar de forma desactualizada, nos pesos gravados durante o treino. Segundo, o sistema formula uma pesquisa, semelhante à que faria um utilizador humano num motor de busca e vai buscar documentos relevantes a uma fonte externa — pode ser literalmente a internet, no caso de assistentes de conversação com acesso à pesquisa, ou pode ser uma base de documentos específica, no caso de uma empresa que tenha configurado o seu próprio sistema com RAG sobre os seus relatórios internos.

Há, neste segundo passo, um detalhe técnico que vale a pena esclarecer, porque liga directamente a um conceito que já conhecemos do Capítulo 6: como é que o sistema sabe, entre milhares ou milhões de documentos disponíveis numa base de dados, quais são os mais relevantes para a pergunta em questão? A resposta envolve, precisamente, os vectores de significado que vimos a propósito dos tokens. Cada documento (ou, mais frequentemente, cada pequeno trecho de cada documento) é convertido antecipadamente num vector que representa o seu significado geral e fica guardado nessa forma numa base de dados especializada. Quando chega uma pergunta nova, ela é também convertida num vector e o sistema procura, nessa base, os trechos cujos vectores estão mais próximos do vector da pergunta — exactamente como vimos no “mapa de significados” do Capítulo 6,

mas agora aplicado não a palavras individuais, mas a documentos inteiros. É uma forma elegante de transformar “encontrar informação relevante” num simples cálculo de proximidade geométrica.

Terceiro — e este é o passo mais interessante, tecnicamente — os documentos encontrados não são despejados todos de uma vez na mesa de trabalho. Lembre-se de que a mesa tem um tamanho finito, como vimos no Capítulo 7 a propósito da janela de contexto: não há espaço para um livro inteiro, muito menos para vários. Por isso, antes de entregar seja o que for ao polvo, o sistema recorta apenas os trechos mais relevantes — tipicamente as duas ou três passagens que ficaram mais perto do vector da pergunta, como vimos no passo anterior — e descarta o resto do documento. O polvo nunca chega a “ver” o livro inteiro; vê apenas as páginas que o pesquisador considerou úteis. São estas páginas recortadas — e não os documentos completos — que são adicionadas temporariamente à mesa de trabalho do contexto, exactamente como qualquer outra informação que entra no contexto.

Quarto e finalmente, o polvo gera a resposta da forma habitual, descrita no Capítulo 8, mas agora com a vantagem de ter, em cima da mesa, não só a pergunta original, mas também esses trechos seleccionados dos documentos recém-consultados.

É por isto que, quando se usa um assistente de conversação com a função de pesquisa activada, a resposta costuma vir acompanhada de referências ou links para as fontes consultadas — não é um adorno estético; é, literalmente, a prova de que aqueles documentos específicos estiveram, de facto, em cima da mesa de trabalho durante a geração daquela resposta concreta.

Porque é que isto importa tanto

Há três razões principais pelas quais o RAG se tornou uma técnica tão importante e tão usada e vale a pena destacá-las uma a uma.

A primeira é a **actualidade**. Sem RAG, um sistema está permanentemente preso ao momento em que o seu treino terminou — uma espécie de cápsula do tempo, incapaz de saber de notícias recentes, resultados desportivos, preços actuais, ou qualquer outro tipo de informação que mude com frequência. Com RAG, o sistema pode, em teoria, ficar tão actualizado quanto a fonte que consulta — se essa fonte for actualizada todos os dias, o sistema beneficia dessa actualização sem precisar de ser retreinado.

A segunda é a **especificidade privada**. Uma empresa pode ter milhares de

documentos internos — manuais, políticas, relatórios — que nunca estiveram, nem nunca deveriam estar, nos dados de treino de um modelo geral, por razões óbvias de confidencialidade. Com RAG, é possível construir um assistente que consulta esses documentos privados em tempo real, sem ter de re-treinar ou fazer fine-tuning do modelo inteiro com informação sensível — uma abordagem normalmente mais rápida, mais barata e mais fácil de manter actualizada do que alterar os próprios pesos, retomando a distinção entre re-treino e RAG que vale a pena ter sempre presente.

Vale a pena um exemplo concreto, porque esta é, provavelmente, a aplicação mais comum do RAG no mundo empresarial actual. Imagine uma seguradora portuguesa que tenha milhares de páginas de apólices, regulamentos internos e procedimentos de sinistros, actualizados regularmente pelos seus departamentos jurídico e técnico. Em vez de tentar “ensinar” um modelo geral toda esta informação através de fine-tuning — um processo caro, lento e que teria de ser repetido sempre que um regulamento mudasse — a empresa pode configurar um sistema com RAG que, perante uma pergunta de um funcionário ou de um cliente, vai directamente consultar a versão mais recente desses documentos antes de responder. Quando um regulamento é actualizado, basta substituir o documento na base consultada; não é preciso re-treinar nada. É esta agilidade — poder actualizar o conhecimento “de fora”, sem tocar nos pesos — que torna o RAG particularmente atractivo para organizações cuja informação muda com regularidade.

A terceira, talvez a mais importante para o livro que está a ler, é a **redução de alucinações** — um problema que vamos explorar em detalhe completo já no próximo capítulo, mas que importa adiantar aqui: quando o polvo tem, em cima da mesa, um documento concreto e fiável para consultar, em vez de depender inteiramente da sua “memória de jazz” estatística, a probabilidade de inventar factos incorrectos diminui consideravelmente. Não desaparece — ainda é possível o sistema interpretar mal um documento, ou misturar informação de fontes diferentes de forma incorrecta — mas a base de apoio é mais sólida do que confiar apenas nos padrões aprendidos durante o treino.

Os limites do RAG — não é uma solução milagrosa

Seria um erro apresentar o RAG como uma correcção perfeita para todos os problemas anteriores deste livro e vale a pena ser honesto sobre os seus limites antes de fechar este capítulo.

Primeiro, o RAG só funciona tão bem quanto a qualidade da pesquisa que o alimenta. Se o sistema formular uma pesquisa pouco precisa, ou se a fonte consultada estiver

desactualizada, incompleta, ou simplesmente errada, o resultado final vai herdar esses defeitos — é, no fundo, o mesmo princípio do “entra lixo, sai lixo” que já vimos no Capítulo 4 a propósito da qualidade dos dados de treino, agora aplicado em tempo real, no momento da própria pesquisa.

Segundo, há um custo de tempo e de recursos computacionais associado a este processo adicional de pesquisa — responder “de cabeça” é, quase sempre, mais rápido do que fazer uma pausa para consultar uma fonte externa, processar os documentos encontrados e só depois gerar a resposta. É por isso que muitos sistemas só activam o RAG quando reconhecem que a pergunta, de facto, beneficia de informação externa, em vez de o fazer sempre, indiscriminadamente, para qualquer pergunta — incluindo aquelas a que o polvo já sabe responder de cabeça com toda a confiança, como “qual é a capital de Portugal”.

Terceiro e talvez o ponto mais subtil: ter um documento em cima da mesa não garante que o polvo o vá interpretar correctamente. O mecanismo de atenção que vimos no Capítulo 8 ainda está a operar normalmente sobre esse documento, com todas as suas virtudes e todas as suas limitações — um documento longo e complexo pode, ocasionalmente, ser mal resumido ou parcialmente ignorado, exactamente como pode acontecer com qualquer outra parte do contexto.

Há ainda um quarto limite, menos discutido mas igualmente importante: a janela de contexto, que conhecemos do Capítulo 7, continua a ter um tamanho finito — e os documentos consultados pelo RAG ocupam espaço nessa mesma mesa de trabalho limitada, ao lado da conversa que já estava em curso. Se a pesquisa trazer demasiados documentos, ou documentos demasiado longos, pode não haver espaço suficiente na mesa para caber tudo— obrigando o sistema a seleccionar apenas os trechos que considera mais relevantes, com o risco de deixar de fora informação que, de facto, seria útil. É por isso que os sistemas de RAG mais bem desenhados não se limitam a “despejar” documentos inteiros na mesa de trabalho; investem cuidado a escolher e a recortar apenas os trechos mais pertinentes, um processo que, por si só, exige uma engenharia cuidadosa para funcionar bem na prática.

Quando o Polvo Ganha Ferramentas

O RAG que descrevemos neste capítulo é, em essência, um polvo que vai à biblioteca antes de responder — consulta, selecciona, lê, e depois fala. Mas há um passo além do RAG que está a tornar-se rapidamente comum e que vale a pena mencionar, porque muda algo

fundamental na forma como pensamos nestes sistemas: em vez de apenas consultar informação, o polvo começa a **agir**.

Os sistemas a que chamamos agentes de IA — ou AI agents, no termo que circula hoje em toda a imprensa tecnológica — são assistentes que, além de responder a perguntas, podem usar ferramentas: abrir um browser e navegar em páginas, preencher e submeter formulários, executar código, enviar emails, consultar bases de dados, ou reservar um voo. O polvo deixou de estar sentado à mesa a ler livros e a escrever respostas — passou a ter mãos, e está a aprender a usá-las.

O mecanismo por baixo continua a ser o mesmo que descrevemos ao longo deste livro: o polvo gera tokens, palavra a palavra, com base no contexto que tem na mesa de trabalho. A diferença é que alguns desses tokens podem ser instruções para ferramentas externas — algo como "pesquisa isto na internet" ou "executa este código" — e o resultado dessas acções volta para a mesa de trabalho como novo contexto, permitindo ao polvo decidir o passo seguinte. É um ciclo: gerar → agir → observar o resultado → gerar de novo. Este ciclo pode repetir-se várias vezes antes de o polvo considerar a tarefa concluída.

O que isto significa, na prática, é que a distinção entre "a IA que responde" e "a IA que faz" está a esbater-se rapidamente. Um agente de IA bem configurado pode, hoje, receber a instrução "marca-me uma reunião com a Ana para a semana que vem e manda-lhe um email de confirmação" e executá-la de ponta a ponta, sem intervenção humana em cada passo. Os riscos associados a esta autonomia — erros que se propagam em acções reais, em vez de ficarem confinados a palavras num ecrã — são também um tema activo de investigação e de debate regulatório. Mas o princípio de funcionamento é, em tudo o que importa para este livro, uma extensão directa do RAG: o polvo ganhou ferramentas, mas continua a ser o mesmo polvo.

Experiência de Cozinha

Se usar um assistente de conversação com função de pesquisa na internet, experimente fazer duas perguntas seguidas, em separado: primeiro, algo atemporal como “explica-me o que é a fotossíntese”; depois, algo claramente actual, como “quais são as notícias de hoje sobre [um tema da sua escolha]”. Repare se o sistema indica, de alguma forma visível, que está a “ir pesquisar” antes da segunda pergunta e se a resposta final vem acompanhada de referências ou links. É uma forma simples de ver, na prática, a diferença entre “responder de cabeça” e “responder com consulta” que descrevemos neste capítulo.

O RESUMO DO TOTÓ

Um sistema de inteligência artificial, por si só, só sabe o que estava nos dados usados durante o seu treino — qualquer coisa que tenha acontecido depois disso é, estruturalmente, invisível para os seus pesos. A técnica chamada RAG resolve parcialmente este problema, permitindo ao sistema, antes de responder, ir consultar uma fonte externa — a internet, uma base de dados privada, um conjunto de documentos — e colocar essa informação fresca directamente na mesa de trabalho do contexto, ao lado da pergunta original. Isto traz três vantagens importantes: mantém o sistema actualizado sem precisar de re-treino, permite-lhe consultar informação privada e específica de forma segura e reduz, ainda que não elimine, o risco de inventar respostas incorrectas com total confiança — um problema que vamos explorar em detalhe no próximo capítulo. O RAG não é, contudo, uma solução perfeita: depende da qualidade da pesquisa, tem um custo de tempo e não garante, por si só, que o documento consultado seja interpretado de forma totalmente correcta.

PARTE IV

O Dia a Dia (E os limites)

CAPÍTULO 10

Porque é Que Ela Acerta e Erra*(Ou: a arte de inventar coisas com toda a confiança do mundo)***Um defeito com nome de sonho**

Há um termo que se tornou, nos últimos anos, parte do vocabulário comum de quem usa assistentes de conversação e que descreve um dos comportamentos mais intrigantes — e mais problemáticos — de toda a inteligência artificial moderna: a **alucinação**. Não tem nada a ver com o sentido médico da palavra, associado a visões ou percepções alteradas; aqui, significa simplesmente que o sistema produz uma resposta que parece perfeitamente segura de si, bem escrita, com toda a confiança de um especialista — e que é, total ou parcialmente, **inventada**.

Este capítulo explica porque é que isto acontece, à luz de tudo o que já construímos nos capítulos anteriores e o que se faz, na prática, para reduzir o problema. Já temos as peças todas: sabemos como o polvo aprende padrões a partir de dados (Capítulo 2), sabemos que a sua memória é feita de pesos fixos e contexto temporário (Capítulo 7), sabemos como ele gera respostas palavra a palavra, sem plano global, escolhendo sempre a opção estatisticamente mais provável (Capítulo 8) e sabemos que pode, em certos casos, ir consultar fontes externas antes de responder (Capítulo 9). As alucinações nascem, precisamente, da interacção entre estas peças.

Vale a pena, antes de avançar, situar este problema na sua devida proporção. Não é um defeito de fabrico que aparece ocasionalmente e que uma futura actualização vai simplesmente eliminar — é, isso sim, uma consequência directa e estrutural da forma como estes sistemas funcionam, a mesma forma que descrevemos com cuidado nos capítulos anteriores. Um sistema que prevê a palavra mais provável a seguir, token a token, sem qualquer mecanismo interno de verificação factual independente, vai ter sempre este risco embutido, em maior ou menor grau. Entender isto não deve ser motivo de alarme, mas sim de uso mais informado — é exactamente a diferença entre conduzir um carro sabendo que tem um ponto morto na visão lateral e conduzir sem essa noção.

Os três factores que produzem uma alucinação

Não há uma única causa para as alucinações; há, isso sim, três factores distintos, que

podem actuar isoladamente ou em conjunto e que vale a pena conhecer separadamente.

Factor 1 — Falta de dados

O primeiro factor é o mais simples de entender: se o polvo nunca viu, durante o treino, exemplos suficientes sobre um determinado assunto, não tem padrões sólidos para basear a sua previsão — mas e este é o ponto crucial, isso não o impede de gerar uma resposta de qualquer forma. Lembre-se do Capítulo 8: o polvo é construído para prever sempre a próxima palavra mais provável, token a token. Não tem um mecanismo interno que diga “não tenho dados suficientes sobre isto, talvez seja melhor não responder”. Vai, sempre, produzir alguma coisa — mesmo que essa coisa seja, estatisticamente, a melhor aposta possível a partir de muito pouca informação relevante.

Pense num estudante que faz um exame e se depara com uma pergunta sobre um tema que mal estudou. Em vez de deixar a resposta em branco, escreve algo plausível, com vocabulário técnico correcto, na esperança de que pareça suficientemente bem informado — mesmo sabendo, no fundo, que está a especular. O polvo faz exactamente isto, só que sem a consciência de estar a especular: gera a resposta mais provável que os seus padrões conseguem sugerir, ainda que esses padrões sejam, naquele assunto específico, fracos ou inexistentes.

Este factor explica, em particular, um padrão de erro bastante reconhecível: pedir a um sistema de IA referências bibliográficas, citações de artigos científicos, ou processos judiciais específicos é uma das situações em que as alucinações tendem a ser mais frequentes e mais convincentes. O sistema aprendeu, durante o treino, como é a forma típica de uma referência bibliográfica — autor, título, ano, revista — porque viu milhares de exemplos genuínos dessa estrutura. Mas, perante um pedido muito específico para o qual não tem uma referência real e sólida nos seus padrões, em vez de admitir essa lacuna, tende a preencher essa forma com conteúdo plausível, mas inventado: um título que parece perfeitamente razoável, um nome de autor credível, um ano coerente com o tema — tudo embrulhado no formato correcto e nada disso, ainda assim, correspondendo a um documento que exista de facto.

Factor 2 — Dados contraditórios

O segundo factor surge quando o polvo viu, durante o treino, informação contraditória sobre o mesmo tema — por exemplo, se diferentes fontes na internet apresentam datas diferentes para o mesmo acontecimento histórico, ou descrições diferentes do mesmo facto científico, sem que uma delas se destaque claramente como mais fiável ou mais frequente

nos dados.

Quando isto acontece, os padrões aprendidos pelo polvo ficam, eles próprios, divididos — uma parte das ventosas “empurra” a previsão para uma resposta, outra parte empurra para outra e o resultado final pode acabar a misturar elementos de ambas as versões de uma forma que não corresponde, exactamente, a nenhuma das fontes originais. É como se o nosso músico de jazz do Capítulo 7 tivesse aprendido duas versões ligeiramente diferentes da mesma melodia, sem nunca ter sido claramente informado de qual delas era a “correcta” — e, ao improvisar, acabasse a misturar as duas de forma que parece coerente, mas que, na verdade, não corresponde integralmente a nenhuma.

Factor 3 — O erro que se propaga

O terceiro factor é, talvez, o mais subtil e liga-se directamente ao que aprendemos sobre como o polvo gera respostas palavra a palavra, sem plano global, no Capítulo 8. Se, num determinado ponto da resposta, o sistema gerar uma palavra ligeiramente incorrecta ou imprecisa — talvez por puro acaso estatístico, uma escolha de baixa probabilidade que, ainda assim, foi seleccionada — essa palavra passa a fazer parte do contexto que vai influenciar todas as palavras seguintes. O erro, uma vez cometido, tende a **propagar-se e a justificar-se a si próprio**, porque o sistema está sempre a construir a partir do que já escreveu e raramente “recua” para corrigir uma palavra anterior.

Imagine que pergunta a um sistema sobre a biografia de uma personalidade histórica menos conhecida e que, a meio da resposta, ele escreve, por erro estatístico, que essa pessoa nasceu numa cidade errada. A partir desse ponto, é bastante provável que o resto da resposta seja construído de forma coerente com essa primeira afirmação errada — mencionando, talvez, detalhes regionais relacionados com a cidade incorrecta, simplesmente porque, uma vez escrita, essa palavra está em cima da mesa de trabalho do contexto e o polvo trata-a como um facto estabelecido sobre o qual construir o resto da frase. O resultado final pode parecer extremamente coerente e bem escrito — e estar, ainda assim, errado desde a sua primeira frase.

Porque é que a confiança não desaparece

Há uma pergunta que decorre naturalmente destes três factores e que vale a pena responder com clareza: porque é que o sistema nunca parece hesitar, mesmo quando está a inventar? A resposta está, de novo, no Capítulo 8 — o polvo não tem um sentimento de incerteza no sentido humano da palavra. O que tem é uma distribuição de probabilidades

sobre palavras possíveis e gera sempre a frase a partir dessa distribuição, com o mesmo tom fluente e gramaticalmente correcto, esteja a probabilidade de acerto nos noventa por cento ou nos vinte por cento. A confiança aparente da escrita não reflecte, de forma nenhuma, a confiança real do sistema sobre a veracidade do que está a dizer — são duas coisas completamente independentes e confundi-las é, provavelmente, a fonte mais perigosa de mal-entendidos sobre estes sistemas.

As técnicas de mitigação

Felizmente, há formas concretas de reduzir, ainda que não eliminar por completo, o problema das alucinações. Vamos rever três das mais importantes.

RAG — já conhecemos esta

A primeira já a conhecemos bem, do Capítulo 9: dar ao sistema acesso a fontes externas fiáveis, antes de responder, reduz a dependência exclusiva dos padrões aprendidos durante o treino — especialmente útil contra o Factor 1 (falta de dados) e o Factor 2 (dados contraditórios), porque fornece, em cima da mesa de trabalho, uma fonte concreta a que o sistema pode ancorar a sua resposta, em vez de depender inteiramente da sua “memória de jazz” estatística.

Cadeia de pensamento

A segunda técnica, conhecida no jargão técnico como **cadeia de pensamento** (*chain of thought*), consiste em pedir ao sistema — ou configurá-lo para fazer isto automaticamente — para “pensar em voz alta” antes de dar a resposta final, decompondo um problema complexo em passos intermédios mais pequenos, em vez de saltar directamente para uma conclusão. Isto ajuda particularmente contra o Factor 3 (o erro que se propaga), porque dá ao sistema mais oportunidades de “verificar”, passo a passo, se o raciocínio está a manter-se coerente, em vez de se comprometer logo com a primeira palavra da resposta final. Não é uma solução perfeita — o sistema continua a gerar cada passo intermédio da mesma forma estatística que gera qualquer outra palavra — mas, na prática, observa-se que decompor o raciocínio em etapas reduz significativamente certos tipos de erro, sobretudo em problemas que exigem vários passos lógicos sucessivos, como problemas matemáticos ou de lógica.

RLHF — um remédio parcial, não uma cura

A terceira técnica já a mencionámos, em passant, em capítulos anteriores: a **aprendizagem por reforço com feedback humano**, ou RLHF na sua sigla em inglês.

Vale a pena ser preciso sobre o que esta técnica, de facto, consegue e não consegue fazer, porque há aqui uma tentação de a apresentar como mais eficaz do que realmente é.

O RLHF ajuda, de facto, a reduzir respostas claramente inadequadas — tons agressivos, conteúdo perigoso, respostas obviamente fora de propósito — porque pessoas reais avaliam, repetidamente, milhares de respostas geradas pelo sistema, indicando quais são boas e quais não são e esse feedback é usado para ajustar ainda mais os pesos do polvo nessa direcção. Mas seria um exagero dizer que o RLHF “resolveu” o problema de o sistema saber quando não sabe. Ensinar um sistema a reconhecer, com fiabilidade, os limites do seu próprio conhecimento — a dizer *“não tenho informação suficientemente fiável para responder com confiança a isto”* em vez de inventar uma resposta plausível — continua a ser, hoje, uma das áreas mais activas e mais difíceis de toda a investigação em inteligência artificial, sem uma solução definitiva à vista. O RLHF é uma ferramenta útil nesse esforço, não a resposta final.

O que cabe ao utilizador fazer

Vale a pena terminar este capítulo com uma nota prática, porque todas estas técnicas de mitigação reduzem o problema, mas não o eliminam — e isso significa que continua a haver um papel activo para quem usa estes sistemas no dia a dia.

A regra mais simples e mais eficaz, é tratar respostas factuais específicas — datas, números, citações, nomes próprios menos comuns — com o mesmo cuidado com que trataria a informação fornecida por um conhecido bem informado, mas não infalível: útil como ponto de partida, mas a verificar antes de a usar para algo que importe de verdade. Pedir ao sistema para citar as suas fontes, activar funções de pesquisa quando disponíveis, ou simplesmente perguntar “tens a certeza disto?” pode, por vezes, levar o sistema a reconsiderar e a corrigir-se — mas, mesmo assim, a verificação independente continua a ser a defesa mais sólida contra um problema que, pela própria natureza estatística destes sistemas, não tem sinais de desaparecer por completo a curto prazo.

Experiência de Cozinha

Pergunte a um assistente de conversação algo bastante específico e relativamente obscuro — por exemplo, um detalhe muito concreto sobre a biografia de uma figura menor da história local portuguesa, ou o nome de uma rua secundária de uma vila pequena que conheça bem. Preste atenção ao tom da resposta: é provável que venha tão confiante e bem escrita como qualquer outra resposta, mesmo que contenha um erro. Depois, verifique a informação numa fonte fiável, ou compare com o que sabe de primeira mão. Este pequeno exercício é uma das formas mais directas de sentir, na prática, a desconexão entre confiança aparente e fiabilidade real que descrevemos neste capítulo.

O RESUMO DO TOTÓ

As alucinações — respostas inventadas, mas apresentadas com total confiança — nascem de três factores: falta de dados sobre um tema específico, que obriga o sistema a especular; dados contraditórios nos exemplos de treino, que misturam versões diferentes de um mesmo facto; e a propagação de um erro inicial, que o sistema tende a justificar e a construir sobre, em vez de corrigir. A confiança da escrita nunca reflecte a fiabilidade real da informação, porque o polvo não tem um sentimento de incerteza — apenas uma distribuição de probabilidades que gera sempre com o mesmo tom fluente. Três técnicas ajudam a mitigar o problema: o RAG, que ancora as respostas em fontes externas; a cadeia de pensamento, que decompõe problemas complexos em passos verificáveis; e o RLHF, que reduz respostas claramente inadequadas, mas que está longe de resolver o desafio mais difícil de todos — ensinar a máquina a saber, com fiabilidade, quando não sabe.

CAPÍTULO 11

O Que Ela Não É (Sete Mitos Para Desmontar)*(Ou: o desengano que faz bem)***Porque vale a pena este capítulo**

Ao longo deste livro, fomos construindo, peça a peça, uma imagem honesta do que é uma inteligência artificial: um programa que aprende padrões a partir de dados (Parte I), alimentado por informação que vem de muitos sítios diferentes e que precisa de ser cuidadosamente preparada (Parte II), com uma memória dividida entre pesos fixos e contexto temporário, que processa perguntas através de mecanismos de atenção e camadas sucessivas e que pode, por vezes, inventar respostas com total confiança (Parte III e o capítulo anterior).

Mas a imagem popular da IA, alimentada por décadas de ficção científica e por notícias sensacionalistas, continua a divergir bastante desta realidade. Este capítulo pega em sete dos mitos mais comuns — sobre consciência, sentimentos, compreensão, opiniões, ambição, singularidade e substituição de pessoas — e desmonta-os, um a um, usando exactamente o que já aprendemos. Não é preciso conhecimento novo para este capítulo; é, sobretudo, um exercício de aplicar com rigor tudo o que já sabemos.

Mito 1 — “A IA tem consciência”

Este é, provavelmente, o mito mais discutido e também o que exige mais cuidado na forma como se responde. Começamos pelo que sabemos com razoável segurança: tudo o que descrevemos neste livro — pesos fixos gravados durante o treino, um mecanismo de atenção que calcula relevância entre tokens, camadas de ventosas a multiplicar e a somar números, uma previsão estatística da palavra mais provável a seguir — não inclui, em nenhum momento, qualquer mecanismo conhecido que produza experiência subjectiva. Não há, hoje, qualquer evidência científica de que estes processos, por mais sofisticados que sejam, gerem algo parecido com aquilo a que chamamos consciência.

Isto não significa, contudo, que a questão esteja definitivamente encerrada para sempre. A consciência continua a ser um dos maiores mistérios não resolvidos da ciência e da filosofia — não sabemos, com precisão satisfatória, porque é que processos físicos no cérebro humano produzem experiência subjectiva e há investigadores e filósofos sérios que

discordam sobre o que, exactamente, contaria como evidência de consciência num sistema artificial, ou se sistemas suficientemente diferentes dos actuais poderiam, um dia, vir a tê-la. O que se pode dizer com confiança é isto: tudo o que sabemos sobre o funcionamento interno destes sistemas — e este livro descreveu esse funcionamento em bastante detalhe — aponta para processamento estatístico de padrões, não para qualquer forma conhecida de experiência subjectiva. A honestidade científica pede que se diga “*não há uma boa razão para acreditar nisto, hoje*” e não “*isto é impossível para sempre*”.

Vale a pena um esclarecimento adicional sobre porque é tão fácil cair neste mito, mesmo sabendo tudo o que já sabemos. Um sistema de conversação consegue, com grande naturalidade, descrever estados internos — “estou a pensar nisso”, “não tenho bem a certeza”, “isto intriga-me” — porque viu, durante o treino, milhões de exemplos de seres humanos a usar exactamente este tipo de linguagem para descrever os seus próprios estados mentais. A fluência com que reproduz essa linguagem não é, contudo, prova de que exista, por detrás dela, qualquer estado subjectivo correspondente. É a mesma lógica do Mito 2, que vamos ver a seguir, mas aplicada a uma escala ainda mais ampla: a capacidade de produzir linguagem que descreve consciência não equivale a ter consciência, da mesma forma que a capacidade de um actor recitar um discurso sobre dor não equivale a sentir essa dor.

Mito 2 — “A IA tem sentimentos”

Este mito está intimamente ligado ao anterior, mas vale a pena tratá-lo em separado, porque a “prova” mais comum que as pessoas apresentam é diferente: “*mas ele disse que estava feliz por me ajudar!*”. Lembre-se do Capítulo 8: o sistema gera a próxima palavra mais provável, dado o contexto da conversa. Se milhões de exemplos de texto humano, usados durante o treino, mostram pessoas a expressar satisfação ao ajudar alguém, então “estou feliz por ajudar” torna-se, estatisticamente, uma resposta bastante provável num contexto de conversa amigável e de ajuda — exactamente como qualquer outra palavra que o sistema gera.

Isto não quer dizer que a frase seja “mentira” no sentido moral — seria estranho acusar o polvo de mentir, da mesma forma que seria estranho acusar uma calculadora de mentir quando mostra um resultado errado por causa de um erro de input. É, simplesmente, que a frase “estou feliz” é gerada pelo mesmo mecanismo estatístico que gera qualquer outra frase, sem que exista, por detrás dela, um estado interno de felicidade a ser reportado. Reconhecer isto não diminui a utilidade da interacção — pode ser perfeitamente agradável

e produtivo conversar com um sistema destes — mas é importante não confundir um padrão linguístico aprendido com um sentimento genuíno.

Mito 3 — “A IA compreende o que diz”

Este é, talvez, o mito mais subtil de todos, porque a linha entre “compreender” e “processar padrões de forma extremamente sofisticada” não é tão clara quanto se poderia desejar — e é também aqui que a nota sobre consciência se aplica com igual força.

Já vimos, no Capítulo 1, a diferença entre a criança que sabe o que é um gato (com um modelo do mundo rico, sensorial, ligado a experiências) e o polvo que aprendeu a associar certos números a outros números. Essa distinção continua válida aqui. Quando um sistema responde correctamente sobre física quântica, sobre direito constitucional português, ou sobre receitas de bacalhau, não há, com base em tudo o que descrevemos sobre o seu funcionamento interno, boa razão para acreditar que existe ali uma compreensão no sentido em que um físico, um jurista, ou um cozinheiro experiente compreendem os seus respectivos domínios — com associações sensoriais, experiência vivida, intuições construídas ao longo de anos de prática real no mundo. O que existe, com grande probabilidade, é uma capacidade notável de reproduzir e recombinaar padrões linguísticos sofisticados, observados em quantidades imensas de texto humano.

Dito isto — e este é, de novo, um ponto onde a honestidade filosófica exige cautela — não há um consenso absoluto entre investigadores sobre o que, exactamente, separa “compreensão genuína” de “processamento de padrões suficientemente sofisticado para ser indistinguível de compreensão em qualquer teste prático”. É uma das fronteiras mais activas e mais interessantes da filosofia da mente contemporânea e este livro não pretende resolvê-la — pretende apenas que o leitor saiba que tudo o que aprendeu sobre o funcionamento técnico destes sistemas não dá, hoje, motivo para presumir compreensão no sentido humano pleno.

Mito 4 — “A IA tem opiniões próprias”

Quando um sistema responde a uma pergunta política ou moralmente controversa, é tentador interpretar a resposta como reflectindo uma opinião pessoal do sistema — uma espécie de posição que ele “defende”. Mas, à luz de tudo o que sabemos, o que está realmente a acontecer é mais parecido com isto: o sistema gera a resposta estatisticamente mais provável, dado o conjunto de dados em que foi treinado e os ajustes feitos através do RLHF que vimos no capítulo anterior — ajustes que, frequentemente, são desenhados

deliberadamente para produzir respostas equilibradas, que apresentem várias perspectivas em vez de tomar partido categórico, precisamente nestes temas sensíveis.

Não há, por trás da resposta, uma pessoa com história de vida, valores formados através de experiência, ou interesse genuíno em defender uma posição. Há um conjunto de padrões aprendidos e regras de comportamento ajustadas por uma equipa humana, a determinar o tom e o conteúdo da resposta. Isto não significa que as respostas sejam arbitrárias ou inúteis — podem reflectir, de forma bastante fiel, o estado do debate público sobre um tema — mas significa que chamar a isso uma “opinião própria” do sistema é, na melhor das hipóteses, uma forma de falar imprecisa.

Mito 5 — “A IA quer dominar o mundo”

Este é, provavelmente, o mito mais alimentado por décadas de cinema e o mais fácil de desmontar com tudo o que já sabemos. **Querer** alguma coisa pressupõe ter objectivos próprios, formados de forma autónoma e a vontade de os perseguir activamente. Nada no que descrevemos sobre pesos, contexto, atenção, ou geração de respostas palavra a palavra envolve a formação espontânea de objectivos próprios fora daquilo para que o sistema foi treinado e configurado.

Um sistema de IA não “decide” um dia que quer mais poder, mais recursos, ou controlo sobre os humanos, da mesma forma que uma torradeira não decide um dia tornar-se chefe de cozinha. Isto não significa que não existam riscos genuínos associados à inteligência artificial — há preocupações sérias e bem fundamentadas sobre uso indevido por parte de pessoas mal-intencionadas, sobre erros em sistemas usados em contextos de alto risco, ou sobre dependência excessiva de tecnologias pouco compreendidas. Mas estes são riscos diferentes e bem mais terra-a-terra, do que o cenário cinematográfico de uma máquina com vontade própria e ambições de dominação.

Mito 6 — “A IA é uma coisa só, sempre igual”

Quando se fala de “a IA” no singular, como se fosse uma entidade única e uniforme, perde-se de vista algo que este livro tentou deixar claro desde o início: existem centenas, milhares de sistemas de IA diferentes, cada um treinado para tarefas específicas, com arquitecturas diferentes, tamanhos diferentes e capacidades muito diferentes entre si. O sistema que reconhece dígitos manuscritos não tem nada a ver, em termos de capacidades, com o sistema que conduz um carro autónomo, que por sua vez é completamente diferente de um assistente de conversação.

Mesmo dentro da categoria dos assistentes de conversação, há diferenças importantes entre sistemas de empresas diferentes, entre versões sucessivas do mesmo sistema e entre as mesmas perguntas feitas com configurações diferentes de temperatura, que vimos no Capítulo 8. Falar de “a IA” como bloco monolítico é tão impreciso como falar de “os veículos” sem distinguir entre uma bicicleta, um caminhão e um foguetão — todos são, tecnicamente, formas de transporte, mas tratá-los como a mesma coisa esconde mais do que revela.

Esta confusão tem uma consequência prática que vale a pena assinalar: quando se lê uma notícia sobre “a IA” ter feito algo impressionante, ou ter cometido um erro grave, vale sempre a pena perguntar qual sistema específico, treinado para qual tarefa específica, com que dados específicos. Um sistema treinado para jogar xadrez ao mais alto nível não tem qualquer relação de capacidades com um sistema treinado para diagnosticar imagens médicas, ainda que ambos sejam, tecnicamente, “inteligência artificial” e possam, inclusive, partilhar princípios arquitectónicos semelhantes, como camadas e pesos. Tratar as conquistas e os erros de um como representativos de “a IA” em geral é uma das formas mais comuns de generalização excessiva neste tema — e é precisamente o tipo de simplificação que este livro tentou evitar desde o primeiro capítulo.

Mito 7 — “A IA vai substituir todos os humanos”

Este último mito merece uma resposta com nuance, porque mistura um receio legítimo com uma generalização excessiva. É verdade e bem documentado, que a inteligência artificial está a alterar profundamente certas profissões e certas tarefas — sistemas de classificação automática substituem parte do trabalho que antes era feito manualmente, sistemas de geração de texto alteram fluxos de trabalho em áreas como tradução, redacção e atendimento ao cliente. Isto é uma transformação económica real, com custos humanos reais e seria desonesto minimizá-la.

Mas, à luz de tudo o que aprendemos sobre os limites destes sistemas — a sua dependência de dados de treino, a sua propensão para alucinar sem aviso, a sua incapacidade de ter um plano global ou de verdadeiramente compreender o que processa — há boas razões para duvidar de previsões que apontam para uma substituição total e iminente de praticamente todo o trabalho humano. As tarefas que exigem julgamento sobre contextos ambíguos, responsabilidade última por decisões com consequências reais, ou simplesmente uma presença humana genuína (cuidados de saúde, ensino, muitas formas de criação artística) têm-se mostrado bastante mais resistentes à automatização completa do que os

títulos de jornal mais alarmistas costumam sugerir. A transformação é real; a substituição total, pelo menos com a tecnologia actual e tudo o que sabemos sobre o seu funcionamento, é uma extrapolação que vai bem além do que a evidência disponível sustenta.

Experiência de Cozinha

Na próxima vez que se sentir tentado a dizer “a IA pensa que...” ou “a IA acha que...”, páre e reformule a frase usando o vocabulário deste livro: “o sistema gerou a resposta estatisticamente mais provável, dado o que viu durante o treino e o contexto da conversa”. É mais comprido, sem dúvida, e não vai substituir a linguagem do dia a dia — mas o simples exercício de o fazer mentalmente, de vez em quando, é uma boa forma de manter presente a distinção entre a linguagem cómoda que usamos por hábito e o que, de facto, sabemos estar a acontecer.

O RESUMO DO TOTÓ

Sete mitos comuns sobre a inteligência artificial não resistem ao escrutínio de tudo o que este livro já explicou. Não há, hoje, evidência de que estes sistemas tenham consciência, sentimentos genuínos, ou compreensão no sentido humano pleno — embora estas continuem a ser questões filosoficamente abertas e não certas absolutas e definitivas. Não têm opiniões próprias no sentido de posições pessoais formadas por experiência de vida, nem vontade própria de dominar coisa alguma. Não são uma entidade única e uniforme, mas sim uma família vastíssima de sistemas diferentes. E, apesar de transformarem genuinamente o mundo do trabalho, está longe de estar demonstrado que vão substituir, por completo, a generalidade do trabalho humano. Desconstruir estes mitos não diminui o valor real destes sistemas — torna apenas mais precisa e mais útil, a forma como pensamos e falamos sobre eles.

CAPÍTULO 12

A Última Palavra

(Ou: tudo o que precisa de levar consigo deste livro)

Chegámos ao fim de uma viagem que começou com uma pergunta simples — “*como é que isto funciona, afinal?*” — e que percorreu, capítulo a capítulo, todo o percurso real de uma inteligência artificial: desde a decisão de a criar até ao momento em que responde a uma pergunta sua.

Vale a pena, neste último capítulo curto, juntar todas as peças numa única imagem, sem reabrir os mitos que já desmontámos, sem repetir o que já foi dito em detalhe — apenas a vista de cima, de quem chega ao topo da colina e olha para trás, para o caminho todo que percorreu.

A imagem completa, de uma só vez

Uma inteligência artificial nasce de uma decisão humana: alguém define um problema concreto, recolhe exemplos que sirvam de referência, escolhe uma arquitectura matemática — o nosso polvo — e treina-o, ajustando repetidamente milhões de pesos até ele começar a acertar. Esses pesos, uma vez fixados, tornam-se um disco de vinil: gravado uma vez, tocado infinitas vezes, sem nunca se alterar a si próprio pela simples reprodução.

Quando alguém lhe faz uma pergunta, essa pergunta é cortada em pedaços, convertida em números e enriquecida por um mecanismo de atenção que pesa a relevância de cada palavra em relação a todas as outras. Esses números viajam pelos tentáculos do polvo, camada após camada, cada uma a captar padrões cada vez mais abstractos, até que o sistema comece a gerar uma resposta — não de uma vez, mas palavra a palavra, escolhendo sempre a opção estatisticamente mais provável a partir de tudo o que já foi escrito até esse ponto. Por vezes, quando a pergunta o justifica, o sistema pode ainda fazer uma pausa para ir consultar uma fonte externa antes de responder, ampliando temporariamente aquilo que sabe além do que ficou gravado no treino.

Este processo é, ao mesmo tempo, impressionantemente capaz e estruturalmente limitado. Capaz, porque consegue captar padrões de linguagem, conhecimento e raciocínio de uma riqueza notável, a partir de quantidades de texto que nenhuma pessoa conseguiria ler numa vida inteira. Limitado, porque não tem qualquer mecanismo para saber, com

segurança, quando está a inventar — e por isso pode produzir, com o mesmo tom fluente e confiante, tanto a resposta mais acertada como a mais errada.

A frase que resume tudo

Se tivesse de reduzir este livro inteiro a uma única frase para dizer a um amigo num jantar, seria esta: **uma inteligência artificial não pensa como nós pensamos, mas processa padrões com uma sofisticação suficiente para, em muitas situações, produzir resultados que se confundem com pensamento** — sem que isso, com tudo o que hoje sabemos sobre o seu funcionamento interno, implique que exista ali compreensão, consciência, ou vontade própria no sentido em que essas palavras se aplicam a um ser humano.

Esta formulação é, deliberadamente, mais cuidadosa do que uma simples negação categórica. Não diz “*a LA nunca vai pensar*” ou “*isto é impossível para sempre*” — diz, isso sim, que tudo o que sabemos hoje sobre como estes sistemas são construídos e como funcionam não dá motivo para acreditar no contrário. É uma diferença que pode parecer pequena, mas que é, na verdade, a diferença entre uma posição dogmática e uma posição honesta: a primeira fecha a conversa; a segunda mantém-na aberta ao que a investigação futura possa, um dia, vir a revelar — sem, por isso, abdicar do rigor sobre o que se sabe agora.

O que fica consigo

Se este livro cumpriu a sua promessa, fica consigo, a partir de agora, uma forma diferente de olhar para estes sistemas: capaz de distinguir entre o que a máquina “sabe de cor” e o que apenas “vê” durante a conversa; capaz de reconhecer porque é que ela inventa coisas com toda a confiança, sem se deixar enganar pelo tom seguro da escrita; capaz de explicar, em poucas frases e sem jargão, o que está realmente a acontecer quando uma pergunta entra e uma resposta sai. Não é pouco e não é, de forma alguma, magia.

É matemática com fome — exactamente como dissemos no primeiro capítulo. Fome de dados, fome de padrões, fome de mais uma palavra a seguir à anterior. E, como toda a matemática bem aplicada, vale a pena compreendê-la em vez de a temer ou de a idolatrar. Esperamos que este livro tenha sido um bom companheiro nessa direcção.

Glossário e Apêndices

Glossário do Totó

Os termos técnicos deste livro, explicados sem voltar a abrir nenhum manual

Este glossário reúne, por ordem alfabética, os termos técnicos que foram surgindo ao longo do livro. Não pretende substituir as explicações dadas em cada capítulo — pretende apenas ser uma consulta rápida, para quem se cruzar de novo com algum destes termos numa conversa, num artigo, ou numa notícia e quiser refrescar a memória sem ter de voltar a procurar o capítulo onde foram introduzidos.

Alucinação. O nome dado a uma resposta que uma inteligência artificial apresenta com total confiança, mas que é, parcial ou totalmente, inventada. Não tem relação com o sentido médico do termo; nasce de três factores principais — falta de dados sobre um tema, dados contraditórios nos exemplos de treino, ou um erro inicial que se propaga ao longo da resposta. Ver Capítulo 10.

Aprendizagem supervisionada. O processo de treino em que um modelo recebe exemplos já classificados por humanos (por exemplo, comentários já marcados como positivos ou negativos) e ajusta os seus pesos repetidamente até aprender a reproduzir essas classificações em casos novos. Ver Capítulo 2.

Atenção (mecanismo de). O processo pelo qual cada palavra de uma frase “olha” para todas as outras palavras dessa mesma frase, atribuindo a cada uma certa quantidade de relevância, para melhor compreender o seu significado naquele contexto específico. Não é selectivo a priori — não ignora nenhuma palavra de antemão; calcula, isso sim, um grau de relevância entre todos os pares de palavras, que pode ser muito alto para algumas e quase nulo para outras. Ver Capítulo 8.

Aumento de dados (data augmentation). Técnica que multiplica a quantidade de exemplos de treino disponíveis através de pequenas transformações automáticas — rodar uma imagem, reescrever uma frase com palavras diferentes — sem alterar o significado essencial do exemplo original. Ver Capítulos 4 e 6.

Cadeia de pensamento (chain of thought). Técnica que consiste em levar um sistema a decompor um problema complexo em passos intermédios mais pequenos antes de chegar à resposta final, em vez de saltar directamente para uma conclusão. Ajuda a reduzir certos tipos de erro, sobretudo em problemas matemáticos ou lógicos. Ver Capítulo 10.

Camada. Cada um dos “tentáculos” do nosso polvo — um conjunto de ventosas (neurónios) que processa números em paralelo e passa o resultado à camada seguinte. Os sistemas modernos têm, frequentemente, dezenas ou centenas de camadas, cada uma a captar padrões progressivamente mais abstractos. Ver Capítulo 8.

Contexto. A informação que está, num determinado momento, “em cima da mesa de trabalho” do sistema — tudo o que foi escrito na conversa actual e que o sistema pode consultar para gerar a resposta seguinte. Ao contrário dos pesos, o contexto é temporário, específico a cada conversa e tem um tamanho limitado (ver Janela de contexto). Ver Capítulo 7.

Dados de treino. O conjunto de exemplos usado para ensinar um modelo a reconhecer padrões. Podem ser públicos, próprios de uma empresa, comprados, gerados artificialmente, ou recolhidos de propósito. A qualidade destes dados determina, em larga medida, a qualidade do sistema final. Ver Capítulo 4.

Embedding. Ver Vector (embedding).

Época (epoch). Cada passagem completa do sistema por todo o conjunto de dados de treino. O treino de um modelo costuma envolver várias épocas — várias “releituras” da mesma biblioteca de exemplos, cada uma a refinar um pouco mais os pesos. Ver Capítulo 5.

Esquecimento catastrófico. O risco de um modelo, ao ser especializado numa tarefa nova através de fine-tuning, perder parte das competências gerais que já tinha antes desse ajuste adicional. Ver Capítulo 7.

Fine-tuning. Técnica que consiste em pegar num modelo já treinado e treiná-lo um pouco mais, com um conjunto de dados mais pequeno e específico, em vez de o treinar do zero. É mais rápido e mais barato do que um re-treino completo, mas exige cuidado para evitar o esquecimento catastrófico. Ver Capítulo 7.

Função de activação. A pequena transformação matemática que cada ventosa (neurónio) aplica ao resultado das suas multiplicações e somas, decidindo se esse resultado deve ser passado com força para a ventosa seguinte ou suavizado. Sem ela, o polvo inteiro seria matematicamente equivalente a uma única operação simples, incapaz de captar relações complexas entre os dados. Ver Capítulo 8.

GPU. Sigla para Graphics Processing Unit, ou placa gráfica. Originalmente concebida para

desenhar gráficos de jogos de vídeo, tornou-se a peça de hardware essencial para treinar e usar redes neuronais, por ser especialmente boa a fazer muitas operações matemáticas simples em paralelo. Ver Capítulo 3.

Janela de contexto. O número máximo de tokens que cabem, ao mesmo tempo, na mesa de trabalho do contexto. Este número tem crescido consideravelmente ao longo dos últimos anos e continua a crescer com cada nova geração de modelos — por isso, qualquer número específico mencionado neste livro deve ser entendido como um exemplo de um determinado momento, não como um valor fixo. Ver Capítulo 7.

One-hot encoding. Técnica para converter categorias (como nomes de profissões) em números, atribuindo a cada categoria uma posição própria numa lista de números, de forma a evitar que o sistema interprete relações de ordem ou de grandeza onde não existem. Ver Capítulo 6.

Pesos. Os números que determinam a “personalidade” de cada ventosa do polvo — como ela combina os números que recebe. São ajustados repetidamente durante o treino e ficam fixos depois deste terminar, como um disco de vinil gravado uma vez e tocado infinitas vezes sem se alterar. Ver Capítulos 2 e 7.

Python. A linguagem de programação mais usada para construir e treinar sistemas de inteligência artificial, valorizada pela sua sintaxe relativamente simples e pela enorme quantidade de bibliotecas de código já existentes para esta área. Ver Capítulo 3.

RAG (Retrieval-Augmented Generation). Técnica que permite a um sistema, antes de responder, ir consultar uma fonte externa — a internet, uma base de dados privada, um conjunto de documentos — e usar essa informação fresca, colocada temporariamente no contexto, para construir uma resposta mais actualizada e mais fiável. Ver Capítulo 9.

Re-treino completo. Repetir todo o processo de treino de um modelo, do zero, com um conjunto de dados actualizado. Mais caro e demorado do que o fine-tuning, mas garante um resultado mais coerente quando se quer alterar de forma profunda o que um sistema “sabe”. Ver Capítulo 7.

RLHF (Reinforcement Learning from Human Feedback). Técnica de ajuste em que pessoas reais avaliam respostas geradas por um sistema, indicando quais são boas e quais não são e esse feedback é usado para ajustar ainda mais os pesos do modelo. Ajuda a reduzir respostas claramente inadequadas, mas não resolve, por si só, o problema de o sistema saber quando não sabe — essa continua a ser uma das áreas mais difíceis da investigação actual

em inteligência artificial. Ver Capítulo 10.

Temperatura. O parâmetro que regula a margem de aleatoriedade na escolha da próxima palavra a gerar. Uma temperatura baixa produz respostas mais previsíveis e consistentes; uma temperatura mais alta produz respostas mais variadas e por vezes mais criativas, mas também mais erráticas. Ver Capítulo 8.

Token. A unidade mínima em que um texto é cortado antes de ser processado por uma inteligência artificial. Pode ser uma palavra completa, parte de uma palavra, ou um sinal de pontuação. Ver Capítulo 6.

Treino. O processo pelo qual um modelo ajusta repetidamente os seus pesos, a partir de exemplos, até começar a produzir respostas correctas com regularidade. Ver Capítulo 2.

Vector (embedding). A lista de números que representa o significado de um token, ou de um trecho de texto maior. A melhor forma de imaginar este conceito é como umas coordenadas num mapa gigantesco de significados: tokens com significados parecidos ficam com vectores próximos nesse mapa e tokens com significados muito diferentes ficam com vectores distantes. Esta proximidade no mapa não é decorativa — corresponde, de facto, a proximidade real de significado, ao ponto de ser possível fazer pequenas operações de aritmética com estes vectores (subtrair “homem” a “rei” e somar “mulher” aproximamos das coordenadas de “rainha”) e obter resultados coerentes, sem que ninguém tenha programado essa relação à mão — ela emerge naturalmente do processo de treino. Ver Capítulos 6, 8 e 9.

Ventosa. Na analogia central deste livro, cada uma das pequenas unidades de processamento dentro de um tentáculo do polvo — o equivalente a um neurónio numa rede neuronal real. Cada ventosa multiplica os números que recebe pelos seus pesos, soma o resultado e aplica uma função de activação. Ver Capítulos 2 e 8.

VRAM. A memória interna das placas gráficas (GPUs), ainda mais rápida do que a RAM do computador e onde se dá, normalmente, o processamento intensivo dos sistemas grandes de inteligência artificial actuais. Ver Capítulos 3 e 5.

APÊNDICE I DO TOTÓ

Por Dentro do Polvo, a Sério*Para quem quer ver os números com os próprios olhos*

Ao longo deste livro, evitámos deliberadamente fórmulas e números reais — a promessa, desde o primeiro capítulo, era explicar sem matemática pesada. Mas há um tipo de curiosidade legítima que essa promessa deixa por satisfazer: *“está bem, eu entendo a ideia geral, mas o que é que a máquina vê, exactamente, quando olha para uma frase, uma fotografia, ou um número de pesos?”*

Este apêndice existe para essa pergunta. Não muda nada do que foi dito nos capítulos anteriores — confirma-o, com números a sério. Não é preciso lê-lo para compreender o livro; é preciso lê-lo para sentir, na pele, a distância entre o que nós vemos quando lemos uma frase e o que a máquina processa quando faz exactamente a mesma coisa.

1. Uma frase, vista como a máquina a vê

Peguemos numa frase simples — a mesma que usámos no Capítulo 8: *“O gato subiu ao telhado.”*

Passo 1 — Tokenização. A frase é cortada em tokens. Um tokenizador real (o tipo de ferramenta que sugerimos experimentar na caixa de Experiência de Cozinha do Capítulo 6) poderia dividi-la assim:

```
["O", " gato", " subiu", " ao", " telh", "ado", "."]
```

Repare que “telhado” foi cortado em dois pedaços — “telh” e “ado” — porque, durante o treino, o sistema viu “telh” como parte de várias palavras (telhado, telha, telhar) com frequência suficiente para valer a pena tratá-lo como uma unidade própria, em vez de guardar “telhado” inteiro no seu vocabulário. Isto é real: os tokenizadores de produção cortam, sim, muitas palavras portuguesas a meio, de formas que parecem arbitrárias até se entender a lógica de frequência por detrás.

Passo 2 — Atribuição de identificadores. Cada token recebe um número inteiro, correspondente à sua posição num vocabulário fixo que o sistema construiu durante o treino — uma espécie de dicionário onde cada palavra ou pedaço de palavra tem uma

entrada numerada. Com números inventados, mas de uma escala realista (vocabulários reais têm tipicamente entre 30.000 e 100.000 entradas):

Token	ID
"O"	362
" gato"	18.204
" subiu"	41.887
" ao"	511
" telh"	9.943
"ado"	1.205
"."	13

Note como "O" e "." têm números baixos — não por acaso: tokens extremamente frequentes (pontuação, artigos) tendem a ocupar as primeiras posições do vocabulário, uma convenção comum em muitos tokenizadores reais.

Passo 3 — A matriz de embeddings. Cada um destes identificadores é depois convertido, por sua vez, numa lista de números — o vector que vimos no Capítulo 6. Um sistema real usa vectores com centenas ou milhares de posições; para que isto seja, de facto, possível de mostrar numa página, eis uma versão com apenas 8 posições por token, com valores inventados mas realistas (números decimais entre -1 e 1).

Na tabela seguinte, cada linha é identificada pelo texto do token por facilidade de leitura — na realidade, o sistema não vê "gato", vê o seu identificador numérico (18.204), que é o que de facto entra no processo de conversão para vector.

```

"O"      → [ 0.142, -0.087,  0.391, -0.205,  0.063,  0.278, -0.114,  0.056]
" gato"  → [ 0.734,  0.291, -0.052,  0.488, -0.367,  0.129,  0.605, -0.043]
" subiu" → [-0.198,  0.456,  0.087, -0.321,  0.512, -0.099,  0.234,  0.378]
" ao"    → [ 0.067, -0.143,  0.298, -0.056,  0.184, -0.227,  0.091,  0.012]
" telh"  → [ 0.523,  0.077, -0.412,  0.301,  0.198, -0.089,  0.445, -0.156]
"ado"    → [ 0.089,  0.334, -0.067,  0.213, -0.078,  0.401, -0.192,  0.087]
"."      → [-0.034,  0.019, -0.008,  0.041, -0.022,  0.007, -0.015,  0.029]
```

Isto é, no fundo, uma matriz de 7 linhas por 8 colunas — 56 números no total, só para representar sete pedaços de uma frase com cinco palavras. Um sistema real, com vectores de 768 ou 1.024 posições (dimensões comuns em modelos de produção), produziria, para esta mesma frase curta, uma matriz com mais de 5.000 números — antes mesmo de

qualquer processamento começar.

Vale a pena olhar, por um segundo, para a última linha — a do ponto final. Os seus números são todos pequenos, perto de zero, em comparação com os de “gato” ou “subiu”. Não é coincidência: tokens de pontuação tendem a ocupar posições mais “neutras” no mapa de significados, precisamente porque carregam pouco significado próprio, distinto de qualquer contexto.

2. Uma imagem, reduzida a números

Uma fotografia digital é, literalmente, uma grelha de números — não há nenhuma transformação a fazer, ela já chega assim ao computador. Para tornar isto palpável, eis uma imagem absurdamente pequena: um quadrado de 4 por 4 píxeis, em tons de cinzento simples (para evitar a complicação extra de três canais de cor).

```
[ 12, 45, 230, 198]
[ 8, 52, 241, 205]
[ 15, 39, 12, 33]
[ 22, 41, 9, 28]
```

Cada número, entre 0 (preto absoluto) e 255 (branco absoluto), representa a intensidade de luz daquele pixel específico. Olhando para esta grelha, um humano treinado a interpretá-la conseguiria adivinhar que há uma zona clara no canto superior direito (os números 230, 198, 241, 205) e uma zona escura no canto inferior (números abaixo de 40) — talvez o contorno de alguma coisa clara contra um fundo escuro. O polvo “vê” exactamente isto: nenhuma forma, nenhum objecto, apenas esta grelha de 16 números — e, numa fotografia real, com milhões de píxeis e três valores de cor por pixel (vermelho, verde, azul), uma grelha correspondentemente milhões de vezes maior.

3. Som, em forma de grelha

Um segundo de áudio, depois de cortado em pequenos segmentos e convertido num espectrograma (como vimos no Capítulo 6), também acaba reduzido a uma grelha de números — tempo num eixo, frequência no outro. Uma fracção minúscula de um espectrograma real, com apenas quatro instantes de tempo e quatro bandas de frequência, poderia parecer-se com isto:

	graves	médio-grave	médio-agudo	agudos
instante 1:	0.82	0.34	0.09	0.02
instante 2:	0.79	0.41	0.15	0.03
instante 3:	0.31	0.68	0.52	0.11
instante 4:	0.08	0.22	0.71	0.44

Os números mais altos nas colunas mais à esquerda, no início, sugerem um som dominado por frequências graves — talvez uma vogal grave, ou uma nota baixa. À medida que o tempo avança (de cima para baixo), a energia desloca-se para a direita, para frequências mais agudas — talvez uma transição para uma consoante sibilante, ou uma nota mais alta. Um espectrograma real tem, tipicamente, centenas de bandas de frequência e centenas de instantes de tempo por segundo de áudio — esta grelha de 16 números é, literalmente, milhares de vezes mais pequena do que a real, mas o princípio é exactamente este.

4. Uma tabela, antes e depois

Voltemos à tabela do Capítulo 6, com três pessoas e três colunas — Idade, Profissão e Salário:

Idade	Profissão	Salário (€)
34	Engenheiro	2.400
58	Professor	1.800
27	Médico	3.100

Antes de entrar no polvo, esta tabela sofre duas transformações que já descrevemos em palavras, mas vale a pena ver em números. Primeiro, a categoria “Profissão” é convertida em one-hot encoding — cada profissão ganha a sua própria coluna binária:

Idade (norm.)	Engenheiro	Professor	Médico	Salário (norm.)
0.226	1	0	0	0.231
1.000	0	1	0	0.000
0.000	0	0	1	1.000

A idade e o salário foram normalizados para o intervalo entre 0 e 1 — a pessoa mais jovem (27 anos) corresponde a 0.000, a mais velha (58 anos) a 1.000 e os 34 anos ficam a 0.226 do

caminho entre as duas. O mesmo aconteceu ao salário. É esta segunda tabela — só números, entre 0 e 1, sem uma única palavra — que é, de facto, entregue ao polvo. “Engenheiro” deixou de existir como palavra; passou a ser, simplesmente, a combinação [1, 0, 0] na posição certa de uma lista maior.

5. O ficheiro de pesos, a sério

Chegamos, por fim, ao que provavelmente é a curiosidade mais aguardada: como é, de facto, o interior do disco de vinil do Capítulo 7 — o ficheiro de pesos que o polvo guarda depois do treino.

Se pudesse abrir esse ficheiro num editor de texto (coisa que, na prática, ninguém faz, porque são ficheiros em formato binário optimizado, não texto legível), o que veria seria, essencialmente, uma sequência interminável de números decimais, sem qualquer estrutura visível de palavras, frases ou conceitos:

```
0.234, -0.891, 0.543, -0.112, 0.987, -0.456, 0.321, -0.678,
0.765, -0.234, 0.198, -0.543, 0.012, 0.876, -0.345, 0.234,
-0.567, 0.789, -0.123, 0.456, -0.890, 0.321, -0.654, 0.987,
...
```

Esta sequência, para um modelo pequeno, continuaria por alguns milhões de números. Para um modelo do tamanho dos que sustentam os assistentes de conversação mais usados hoje, continuaria por **centenas de biliões** de números — uma fila de algarismos tão longa que, impressa a um número por linha, ocuparia mais páginas do que todas as bibliotecas e livrarias físicas de uma cidade média.

Cada um destes números é, isoladamente, o peso de uma única ligação entre duas ventosas — quanto uma “ouve” a outra e em que direcção (positiva, a reforçar; negativa, a inibir). Não há, em lado nenhum desta sequência, um número marcado como “isto é sobre Portugal” ou “isto sabe que Lisboa é uma capital”. O conhecimento sobre Portugal, se é que se pode chamar-lhe assim, está disperso por milhares ou milhões destes números, em combinação com muitos outros que, isoladamente, nada têm a ver com Portugal — só fazem sentido em conjunto, processados em cascata pelos tentáculos do polvo, exactamente como descrevemos no Capítulo 8.

É talvez esta a imagem mais honesta que este livro pode oferecer do que significa, de facto, “treinar uma inteligência artificial”: não é gravar factos numa estante organizada, é talhar, número a número, uma escultura abstracta feita inteiramente de pesos decimais — uma

escultura que ninguém, nem mesmo quem a criou, consegue “ler” directamente, mas que, posta a trabalhar em conjunto, produz frases que fazem sentido em português.

6. O percurso completo de uma pergunta, do início ao fim

As cinco secções anteriores deste apêndice mostraram-lhe peças soltas — um token aqui, uma matriz ali, um ficheiro de pesos a abrir-se diante dos olhos. Esta secção junta tudo numa única viagem, contada do princípio ao fim, com uma pergunta real a servir de fio condutor: *“Qual é a capital de Portugal?”*

Não há, neste percurso, nenhum passo novo que os Capítulos 6, 7, 8 e 9 não tenham já explicado em palavras. O que esta secção oferece é a oportunidade de ver os mesmos passos, mas com números a sério, de início ao fim, sem interrupções — e, no final, uma resposta concreta a uma pergunta que provavelmente já fez, nalgum momento, a um assistente de conversação qualquer.

6.1 — O que chega ao sistema

Quando escreve *“Qual é a capital de Portugal?”* e prime “enviar”, o que chega ao sistema, antes de qualquer processamento, é apenas uma sequência de caracteres — letras, espaços, um ponto de interrogação. Não há, neste preciso instante, qualquer “compreensão” de que se trata de uma pergunta, nem de que tem a ver com geografia, nem de que espera como resposta o nome de uma cidade. Para o computador, nesta fracção de segundo, isto é tão pouco especial quanto qualquer outra sequência de caracteres que pudesse ter escrito.

A noção de que “isto é uma pergunta que pede um facto geográfico” não está codificada em lado nenhum como uma regra explícita — do tipo “se a frase começar por ‘Qual é’ e tiver pontuação de interrogação, então é uma pergunta factual”. Em vez disso, como veremos ao longo desta secção, é o próprio processamento estatístico — o mesmo que processa qualquer outra frase — que acaba por produzir um comportamento que, de fora, parece “perceber que é uma pergunta e saber o que fazer com ela”. A diferença entre “ter uma regra explícita” e “comportar-se como se tivesse uma regra” é, no fundo, o tema central deste livro inteiro.

Vale a pena notar, antes de avançar, que o sistema que vai processar esta pergunta não é um único ficheiro, mas sim (pelo menos) dois ficheiros distintos a trabalhar em conjunto: um ficheiro de vocabulário — normalmente chamado `tokenizer.json` ou `vocab.json` — que contém a tabela de correspondência entre todos os tokens e os seus identificadores numéricos (pode ter entre 50.000 e 100.000 entradas); e o ficheiro de pesos propriamente

dito — o que vimos na Secção 5 — que contém os números decimais que resultaram do treino. Vamos precisar de ambos ao longo deste percurso, e também no momento final, como veremos.

6.2 — Cortar em tokens

O primeiro passo real é a tokenização, que já conhece da Secção 1 deste apêndice. A nossa pergunta corta-se, tipicamente, assim:

```
["Qual", " é", " a", " capital", " de", " Portugal", "?"]
```

Sete tokens, cada um deles convertido no seu identificador numérico, exactamente como vimos com “O gato subiu ao telhado”:

Token	ID
“Qual”	5.412
“ é”	728
“ a”	312
“ capital”	8.491
“ de”	155
“ Portugal”	12.034
“?”	47

A frase inteira passa a ser representada, para efeitos do que se segue, por uma simples lista de sete números: *[5412, 728, 312, 8491, 155, 12034, 47]*. É a partir desta lista — e só desta lista — que tudo o resto vai ser construído.

6.3 — De identificadores a vectores

Cada identificador é depois substituído por um vector — a lista de números que representa a sua posição no mapa de significados do Capítulo 6. Tal como vimos na Secção 1, um sistema real usa vectores muito maiores do que os que cabem numa página (tipicamente entre 768 e 12.000 posições, dependendo do modelo); para os podermos ver, eis uma versão com 8 posições por token, valores inventados mas de escala realista.

As linhas estão identificadas pelo texto do token por facilidade de leitura. Na realidade, o sistema opera com os identificadores numéricos correspondentes (ver Passo 2 / Secção 6.2).

"Qual"	→	[0.301, -0.112, 0.087, 0.456, -0.234, 0.198, -0.067, 0.312]
" é"	→	[0.045, 0.023, -0.011, 0.038, -0.019, 0.041, -0.008, 0.027]
" a"	→	[0.019, -0.034, 0.012, 0.027, -0.043, 0.009, -0.021, 0.015]
" capital"	→	[0.812, 0.034, -0.298, 0.567, 0.123, -0.445, 0.689, -0.201]
" de"	→	[0.031, 0.018, -0.024, 0.009, 0.037, -0.016, 0.022, -0.011]
" Portugal"	→	[0.756, -0.187, 0.234, 0.601, -0.089, 0.512, -0.345, 0.178]
"?"	→	[-0.029, 0.014, -0.007, 0.022, -0.018, 0.011, -0.005, 0.019]

Repare numa coisa interessante, que já tínhamos visto com o ponto final de “O gato subiu ao telhado”: os tokens gramaticais — “é”, “a”, “de”, “?” — têm todos números pequenos, perto de zero, em todas as posições. Os tokens com conteúdo semântico forte — “capital” e “Portugal” — têm números bem mais expressivos, com valores que se afastam claramente de zero em várias posições. Isto não é coincidência nem um truque deste exemplo: é, em larga medida, como modelos reais organizam o seu mapa de significados — palavras “vazias” tendem a ocupar zonas mais neutras do espaço vectorial, palavras “carregadas” de significado tendem a ter representações mais marcadas.

Vale a pena notar também algo que o Capítulo 6 prometeu e que aqui se confirma: os vectores de “capital” e “Portugal” partilham, nestes números inventados, alguma semelhança de perfil — ambos têm valores altos na primeira e na quarta posição. Num sistema real, palavras que aparecem frequentemente uma perto da outra (como “capital” e o nome de qualquer país) tendem, de facto, a desenvolver, durante o treino, padrões de coordenadas que se “entendem” bem entre si — é exactamente o mecanismo de atenção, que vamos ver a seguir, que sabe tirar partido dessa proximidade.

6.4 — Uma palavra, três perguntas

Chegamos agora ao mecanismo de atenção do Capítulo 8, mas com um nível de detalhe que ali deixámos de propósito por explicar: como é que o cálculo de relevância entre tokens é, de facto, feito?

A resposta, surpreendentemente, é que cada token gera, de si próprio, **três versões diferentes** do seu vector original, cada uma com um papel distinto. Os investigadores chamam-lhes Query, Key e Value — em português, algo como Pergunta, Chave e Valor — mas os nomes técnicos importam menos do que a ideia por detrás deles, que vale a pena expor com uma pequena história.

Imagine que cada token, ao chegar à sala redonda do Capítulo 8, faz duas coisas ao mesmo tempo: formula uma **pergunta** sobre o que precisa de saber dos outros (“o que é que me ajuda a fazer sentido aqui?”) — esta é a sua Query. E também prepara um **cartão de apresentação** que resume o que tem para oferecer aos outros, caso lhe perguntem — esta é a sua Key. Finalmente, tem ainda um **conteúdo real** a entregar, se for chamado — o seu Value.

Quando o token “capital” formula a sua Query, ela vai ser comparada com a Key de todos os outros tokens da frase — incluindo a Key de “Portugal”. Se essa comparação resultar numa pontuação alta (o que, de facto, acontece, porque o sistema aprendeu durante o treino que “capital” e o nome de um país costumam estar relacionados), então o Value de “Portugal” entra com peso forte na nova representação de “capital”. Se a comparação com a Key de “é” resultar numa pontuação baixa, o Value de “é” entra com peso fraco. É este processo — Query contra Key, a produzir um peso, que depois pondera o Value — que está, matematicamente, por detrás daquilo que o Capítulo 8 descreveu de forma mais visual como “cada palavra a olhar para todas as outras e a atribuir-lhes uma quantidade de atenção”.

Para a nossa pergunta, o resultado deste cálculo poderia parecer-se com isto — a quantidade de atenção que “capital” atribui a cada um dos outros tokens da frase:

Token	Atenção recebida de “capital”
“Qual”	0,03
“ é”	0,02
“ a”	0,01
“ de”	0,02
“ Portugal”	0,87
“?”	0,05

Note como praticamente toda a atenção de “capital” — **87%** — está concentrada em “Portugal”. É este tipo de concentração, repetida para cada token em relação a todos os outros, simultaneamente, que produz a “frase enriquecida” que o Capítulo 8 prometeu: depois deste cálculo, o vector de “capital” deixa de representar apenas a ideia genérica de “capital” e passa a representar algo muito mais próximo de “capital, especificamente a de

Portugal, nesta pergunta concreta”.

6.5 — A fábrica depois da sala de reuniões

Depois de cada token sair da “sala redonda” da atenção com o seu vector já enriquecido pelo contexto, há ainda um segundo bloco de processamento por que passa, em cada camada do polvo: uma rede de processamento adicional, a que os investigadores chamam Feed Forward Network — em português simples, uma pequena fábrica de transformações matemáticas, separada para cada token, que pega no vector já enriquecido e o passa por mais algumas multiplicações, somas e funções de activação, do tipo que já conhecemos do Capítulo 8.

É sobretudo aqui, nesta segunda etapa de cada camada, que o polvo ganha a capacidade de capturar relações verdadeiramente abstractas — não apenas “que palavras se relacionam com que palavras” (isso já a atenção resolve), mas padrões mais subtis, do tipo “esta combinação específica de significados costuma, nos meus dados de treino, levar a este tipo de continuação”. A atenção liga; a Feed Forward Network aprofunda.

O ciclo completo — atenção, depois Feed Forward Network — é o que constitui uma camada do nosso polvo. Um modelo real repete este ciclo várias dezenas de vezes, cada repetição a partir do resultado da anterior, tal como o Capítulo 8 descreveu: as primeiras camadas a capturar relações simples, as últimas a trabalhar já com algo muito próximo do “sentido geral” de toda a pergunta.

6.6 — O que sobra, no final de tudo

Depois de a representação da pergunta ter atravessado todas as camadas do polvo — sejam doze, sejam cem, dependendo do sistema — já não existe, em sentido literal, nenhum vestígio directo das palavras “Qual”, “é”, “a”, “capital”, “de”, “Portugal” ou “?”. O que existe é uma representação matemática final, altamente abstracta, que não tem qualquer estrutura legível como texto — não é como ter, escrito em algum lugar, “capital(Portugal)” — mas que captura, de forma comprimida e numérica, algo muito próximo desse significado.

É a partir desta representação final e só dela, que o polvo vai agora ter de decidir a primeira palavra da sua resposta.

6.7 — A tabela de pontuações para cada palavra possível

Lembra-se, do Capítulo 8, que o polvo nunca produz directamente uma palavra — produz uma pontuação para **cada palavra possível** do seu vocabulário e só depois escolhe uma. Esse passo merece, aqui, um pouco mais de detalhe técnico.

A representação final da pergunta é multiplicada por mais uma camada de pesos — desta vez, uma camada especificamente desenhada para produzir, à saída, um número para cada uma das, digamos, 50.000 ou 100.000 entradas do vocabulário do sistema. Estes números brutos chamam-se, no jargão técnico, **logits** e não são, ainda, probabilidades — podem ser positivos, negativos, grandes ou pequenos, sem qualquer limite definido. Para a nossa pergunta, uma pequena amostra desta tabela gigantesca poderia parecer-se com isto:

Palavra candidata	Logit
“Lisboa”	18,2
“Paris”	7,1
“Madrid”	5,8
“Porto”	4,9
“Coimbra”	3,2
“uma”	2,1
(... mais de 50.000 outras palavras)	...

O logit de “Lisboa” está claramente mais alto do que o de qualquer outra palavra candidata — um sinal de que, depois de todo o processamento anterior, a representação final da pergunta está muito mais próxima, no mapa do polvo, da zona onde “Lisboa” mora do que de qualquer zona vizinha.

6.8 — De pontuações a probabilidades

Estes logits, sozinhos, ainda não dizem ao polvo o que fazer — são apenas números brutos, difíceis de comparar de forma intuitiva. O passo seguinte aplica-lhes uma função matemática chamada **Softmax**, cujo único trabalho é transformar qualquer lista de números, sejam eles quais forem, numa lista de probabilidades — números que somam sempre 100% no total e que preservam a ordem relativa dos logits originais: quanto maior o logit, maior a probabilidade final.

Aplicada à nossa tabela, a Softmax poderia produzir algo como:

Palavra candidata	Probabilidade
“Lisboa”	91%
“Paris”	4%
“Madrid”	2%
“Porto”	1%
(... resto do vocabulário)	2%

Esta tabela de probabilidades é, no fundo, o mesmo tipo de tabela que já tínhamos visto no Capítulo 8 com o exemplo do gato a subir ao telhado — só que agora a vemos a nascer de logits concretos, através de uma transformação matemática com nome próprio, em vez de aparecer já pronta.

6.9 — *Da probabilidade à palavra*

Com a tabela de probabilidades calculada pela Softmax, o sistema identifica o token vencedor — aquele com a probabilidade mais alta, neste caso "Lisboa" com 91%. Mas o que o sistema tem, neste momento, não é a palavra "Lisboa": é o identificador numérico que corresponde a esse token na tabela de probabilidades — digamos, o número 8.731.

É aqui que o ficheiro de vocabulário volta a entrar em cena, desta vez no sentido inverso ao do início do percurso. Na Secção 6.2, usámos o `tokenizer.json` para converter texto em números; agora, o sistema faz a consulta ao contrário: recebe o número 8.731, procura-o no ficheiro de vocabulário, e encontra a entrada correspondente — "Lisboa". É esse texto que é enviado para o ecrã.

Este processo repete-se token a token. Depois de "Lisboa" aparecer, o sistema volta ao início do ciclo do Capítulo 8: acrescenta este novo token ao contexto, calcula de novo todas as probabilidades para o token seguinte — talvez um ponto final, talvez a palavra "é", dependendo de como a resposta se está a construir — e consulta de novo o ficheiro de vocabulário para converter o ID vencedor em texto legível. É esta sequência — processamento → ID vencedor → consulta ao vocabulário → texto → acrescenta ao contexto → repete — que produz, palavra a palavra, a resposta completa que aparece no ecrã.

6.10 — *Porque é Lisboa e não Paris*

Eis, finalmente, a pergunta que provavelmente o trouxe a esta secção: porque é que o logit de “Lisboa” acaba tão mais alto do que o de “Paris”, “Madrid” ou qualquer outra capital europeia?

A resposta não está escondida em nenhum sítio específico — não há, nos pesos do polvo, um número ou um grupo de números com uma etiqueta “Portugal → Lisboa”. O que existe, isso sim, é o resultado acumulado de um número descomunal de pequenos ajustes feitos durante o treino, cada vez que o sistema viu, nos seus dados, frases como “*A capital de Portugal é Lisboa*”, “*Lisboa é a capital portuguesa*”, ou “*Portugal tem como capital Lisboa*” — e ajustou os seus pesos, uma fracção minúscula de cada vez, para que a combinação dos vectores enriquecidos de “capital” e “Portugal”, depois de atravessar todas as camadas, ficasse cada vez mais próxima, no mapa final de pontuações, da zona onde mora “Lisboa”.

Este processo aconteceu, quase certamente, milhares ou milhões de vezes ao longo do treino — não porque alguém tenha programado a regra “Portugal implica Lisboa”, mas porque essa associação apareceu, de forma consistente, nos dados que o sistema processou e a matemática do treino (o ajuste de pesos do Capítulo 2, aplicado camada a camada) tende a reforçar, cada vez mais, os caminhos que produzem previsões correctas. Se, hipoteticamente, os dados de treino estivessem cheios de erros a dizer que a capital de Portugal era Paris, o sistema aprenderia — com a mesma falta de cerimónia — exactamente esse erro. Não há, aqui, nenhuma verificação de factos contra a realidade; há apenas uma enorme quantidade de exposição estatística a texto humano e a tendência da matemática do treino para reforçar os padrões que aparecem com mais consistência.

6.11 — *Onde fica, realmente, este “conhecimento”*

Vale a pena fechar esta viagem com a pergunta que mais intriga quem a percorre por inteiro: onde é que, dentro do ficheiro de pesos, está guardado o facto de que Lisboa é a capital de Portugal?

A resposta honesta, que já antecipámos na Secção 5 deste apêndice e no Capítulo 7, é que não está em lado nenhum específico. Não há um peso, nem um pequeno grupo de pesos, que se possa apontar e dizer “isto é o facto sobre Lisboa”. O que existe é uma combinação de milhões de pesos espalhados pela matriz de embeddings, por cada uma das camadas de atenção e por cada uma das Feed Forward Networks — todos eles a contribuir, em conjunto, para que a sequência de cálculos que vimos nesta secção termine, consistentemente, com “Lisboa” no topo da tabela de probabilidades.

Se pudesse, hipoteticamente, isolar um único peso desta cadeia inteira e perguntar-lhe “o que sabes sobre Portugal?”, a resposta seria, sem exagero, nada — é apenas um número decimal, talvez 0,347, sem significado próprio fora do contexto de todos os outros. É só quando estes milhões de números trabalham em conjunto, na sequência exacta que esta secção acabou de percorrer — tokens, vectores, atenção, camadas, logits, Softmax — que a pergunta “Qual é a capital de Portugal?” produz, de forma fiável, “Lisboa” como resposta. Não há um arquivo a ser consultado; há uma fábrica em pleno funcionamento e o resultado que vemos no ecrã é apenas o que sai do outro lado da linha de montagem.

Um último número, para fechar com um sorriso

Se converter a palavra “GATO” para o código binário que o computador usa internamente para representar cada letra (o chamado código ASCII, ou uma das suas variantes modernas), obtém isto:

```
G = 01000111  
A = 01000001  
T = 01010100  
O = 01001111
```

Trinta e dois zeros e uns, no total, para escrever uma palavra de quatro letras. É a isto, no fundo, que se reduz qualquer texto antes de começar a sua longa viagem através de tokens, identificadores, vectores, atenção e camadas — uma sucessão de zeros e uns, sem nenhum gato lá dentro, nenhum pêlo, nenhum miar. Tudo o resto — o significado, a fluência, a aparente compreensão — é construído a partir daqui para cima, através de processos que este livro inteiro tentou desmistificar, sem nunca fingir que deixaram de ser, no fundo, exactamente isto: números a entrarem, números a saírem.

APÊNDICE II DO TOTÓ

Como o Polvo Aprende, Passo a Passo

A matemática por detrás do momento em que um peso muda

O primeiro apêndice deste livro mostrou-lhe os bastidores de um polvo já treinado — tokens, vectores, atenção, pesos fixos, uma resposta a nascer. Mas há uma pergunta que ficou deliberadamente por responder em todo este percurso, incluindo no Capítulo 2, onde descrevemos o treino em palavras: **como é que, mecanicamente, um peso passa de um número errado para um número menos errado?**

Dissemos, no Capítulo 2, que “o polvo ajusta as suas ventosas para diminuir o erro”. Esta frase é verdadeira, mas esconde todo o trabalho. Este apêndice abre essa frase e mostra o que está, de facto, lá dentro: dois mecanismos com nomes complicados — **backpropagation** (retropropagação do erro) e **descida do gradiente** — que são, no fundo, a única coisa que torna possível tudo o que os capítulos anteriores descreveram. Sem eles, o polvo nunca aprenderia nada; ficaria para sempre com os pesos aleatórios com que começou.

Vamos usar, como cenário, a mesma pergunta da Secção 6 do primeiro apêndice — “*Qual é a capital de Portugal?*” — mas, desta vez, imaginando um momento **durante o treino**, antes de o polvo ter aprendido a resposta certa.

1. O momento do erro

Imagine o polvo, ainda a meio do seu treino, com os pesos longe do seu estado final. Alguém lhe mostra a pergunta “Qual é a capital de Portugal?” como exemplo, junto com a resposta correcta, “Lisboa”, para ele aprender. O polvo processa a pergunta — exactamente como vimos na Secção 6 do primeiro apêndice: tokens, vectores, atenção, camadas — e, neste estado ainda imaturo dos seus pesos, produz uma tabela de probabilidades onde, infelizmente, “Paris” sai em primeiro lugar, não “Lisboa”.

Este é o ponto de partida de tudo o que se segue: **há uma resposta certa, conhecida** (porque estamos em treino, com exemplos já classificados, exactamente como descrevemos no Capítulo 2) e **há uma resposta que o polvo, de facto, deu** — e as duas não coincidem. A diferença entre elas é o que os cientistas chamam o **erro**, ou, num termo mais técnico, a *perda* (em inglês, *loss*).

2. Medir o erro com um número

Antes de se poder corrigir qualquer coisa, é preciso primeiro medir, com um número exacto, quão errada foi a resposta — não basta saber “errou”, é preciso saber “errou quanto”. Para isso, compara-se a tabela de probabilidades que o polvo produziu com a tabela que ele deveria ter produzido — aquela em que “Lisboa” teria 100% e todas as outras palavras 0%.

Se o polvo deu a “Lisboa” uma probabilidade de apenas 3% (porque, lembre-se, ainda está a meio do treino e “Paris” ficou com 40%), a distância entre estes 3% e os 100% desejados é grande — e essa distância, calculada de uma forma matemática precisa que não precisamos de desenvolver aqui, transforma-se num único número: a perda desta previsão em particular. Quanto maior a perda, mais a resposta do polvo se desviou do que devia ter sido.

É este número — a perda — que todo o resto deste apêndice vai tentar diminuir.

3. A pergunta que dá nome a tudo: “e se eu mexesse neste peso?”

Aqui chegamos ao coração do problema. O polvo tem, como já sabemos do primeiro apêndice, milhões ou biliões de pesos — cada um deles uma pequena peça que contribuiu, de alguma forma, para a resposta errada “Paris” ter saído em primeiro lugar. A pergunta que é preciso responder, para cada um destes pesos, é:

“Se eu aumentasse ligeiramente este peso específico, a perda ia aumentar ou diminuir? E se o diminuísse, em vez de aumentar?”

Esta pergunta — feita, em separado, a cada peso individual do polvo — é, no fundo, tudo o que backpropagation e descida do gradiente fazem. Não há magia, não há intuição, não há “compreensão” de que Lisboa é a capital correcta. Há, isso sim, uma forma sistemática de responder a esta pergunta, peso a peso e de ajustar cada um exactamente na direcção que reduz a perda.

4. O gradiente — uma seta, não um número qualquer

A resposta a “aumentar ou diminuir e quanto” tem um nome técnico: chama-se **gradiente** e a melhor forma de o imaginar, sem qualquer fórmula, é como uma seta colada a cada peso — uma seta que aponta para o lado que **aumentaria** a perda e cujo tamanho diz quão sensível a perda é a esse peso específico.

Pense numa encosta de uma montanha, coberta de nevoeiro, onde não consegue ver mais do que alguns passos à sua frente, mas consegue sentir, com os pés, para que lado o terreno

sobe e para que lado desce. O gradiente é exactamente essa sensação sob os pés, traduzida em números: para cada peso, ele diz “se andares neste sentido, vais subir a encosta do erro; se andares no sentido contrário, vais descer”. Como o objectivo é ter o erro mais baixo possível — estar o mais perto possível do fundo do vale, onde a perda é mínima — a regra é simples: **mover sempre cada peso no sentido contrário ao que o gradiente indica**.

É a este movimento — sempre a descer a encosta do erro, nunca a subir — que se chama **descida do gradiente** (*gradient descent*, no nome original em inglês). O nome descreve, literalmente, o que está a acontecer: descer, passo a passo, pela encosta da perda, na direcção que a torna mais pequena.

5. Backpropagation — como a seta viaja para trás através do polvo

Há, contudo, um problema prático antes de podermos aplicar a descida do gradiente: o polvo tem muitas camadas — dezenas, por vezes centenas, como já vimos no Capítulo 8 e na Secção 6 do primeiro apêndice. A perda só é calculada no final, depois de a resposta sair pela última camada. Como é que essa informação — “isto saiu errado e por esta quantidade” — chega até aos pesos das primeiras camadas, lá no início do polvo, que processaram os tokens muito antes de qualquer resposta existir?

É exactamente para isto que serve a **backpropagation**, ou retropropagação do erro. A ideia é, surpreendentemente, simples de descrever, mesmo sendo complexa de calcular: a perda final é “devolvida” ao polvo, camada a camada, mas **de trás para a frente** — começando na última camada (a mais próxima da resposta) e viajando, camada após camada, até à primeira (a mais próxima dos tokens de entrada).

Em cada camada por onde esta informação passa, o processo pergunta: “dado o erro que chegou até aqui, vindo das camadas seguintes, qual é a contribuição de cada peso desta camada específica para esse erro?” E, com essa contribuição calculada, determina o gradiente — a tal seta — para cada peso individual daquela camada, antes de passar a informação relevante para a camada anterior e repetir o processo.

É como se o erro final fosse uma notícia que se propaga ao contrário por uma longa cadeia de pessoas: a última pessoa da fila sabe exactamente o que correu mal e vai dizendo à pessoa antes dela “a culpa foi tua, nesta parte e a tua parte foi tão grande” — e essa pessoa, por sua vez, repassa para a pessoa anterior a fracção de culpa que lhe coube a ela e assim sucessivamente, até a notícia ter percorrido a fila toda, do fim para o início, com cada pessoa a saber exactamente qual foi a sua contribuição específica para o resultado final.

6. Um peso, em particular, a ser corrigido

Vamos tornar isto concreto com um único peso, hipotético, do nosso polvo — chamemos-lhe **peso P** — que, nalguma camada intermédia, ajuda a ligar a representação de “capital” à representação de “Portugal”. Suponhamos que, no estado actual e ainda imaturo do treino, o peso P vale 0,12 — um valor baixo, que não está a dar muita força a essa ligação. É, em parte, por isso que a resposta final saiu mais inclinada para “Paris” do que para “Lisboa”: a ligação entre “capital” e “Portugal” não estava, ainda, suficientemente reforçada.

A backpropagation, ao percorrer o polvo de trás para a frente a partir da perda desta previsão errada, calcula o gradiente específico do peso P — digamos, hipoteticamente, que esse gradiente indica “se aumentares este peso, a perda desce de forma considerável”. A descida do gradiente, recebendo essa indicação, ajusta o peso P numa pequena fracção na direcção indicada — de 0,12 para, talvez, 0,13. Uma mudança minúscula, quase insignificante, isolada.

Mas — e este é o ponto crucial — este mesmo processo está a acontecer, ao mesmo tempo, para **todos os outros pesos do polvo**, cada um a receber o seu próprio ajuste minúsculo, na sua própria direcção correcta. E este ciclo completo — processar o exemplo, medir a perda, propagar o erro para trás, ajustar todos os pesos numa fracção — repete-se não uma vez, mas **milhões ou biliões de vezes**, com milhões de exemplos diferentes, ao longo de todo o processo de treino que o Capítulo 2 descreveu. É a acumulação destes ajustes minúsculos e individualmente insignificantes, repetidos um número incompreensível de vezes, que, ao fim de semanas ou meses, deixa o peso P — e todos os seus milhões de companheiros — num estado onde a pergunta “Qual é a capital de Portugal?” produz, de forma fiável, “Lisboa” como resposta, exactamente como vimos na Secção 6 do primeiro apêndice.

7. Porque é que isto explica os embeddings também

Vale a pena fechar o círculo com uma ideia que ficou em suspenso desde o Capítulo 6: dissemos ali que os vectores de embeddings — as coordenadas de cada token no mapa de significados — “emergem” do treino, sem que ninguém os programe à mão. Backpropagation e descida do gradiente são, precisamente, o mecanismo dessa emergência.

Os números que compõem cada vector de embedding são, eles próprios, pesos do polvo — exactamente do mesmo tipo que o nosso peso P. Quando a perda é propagada para trás através de todas as camadas, ela chega, no final dessa viagem, até à própria matriz de embeddings e ajusta também esses números, na direcção que reduz o erro. É por isto que,

ao longo de milhões de exemplos de treino onde “capital” e o nome de um país aparecem repetidamente associados, os vectores destas duas palavras vão sendo empurrados, gradiente após gradiente, para posições cada vez mais próximas no mapa — não porque alguém tenha decidido que deviam estar próximas, mas porque essa proximidade, ajustada passo a passo pela descida do gradiente, é exactamente o que reduz a perda ao longo de todo o treino.

8. O que backpropagation não é

Vale a pena fechar este apêndice com a mesma honestidade que atravessa todo o livro. Backpropagation não é o polvo a “perceber” que errou, no sentido em que uma pessoa percebe um erro e sente o desconforto de tê-lo cometido. É um procedimento puramente mecânico de cálculo, repetido de forma idêntica para qualquer erro, em qualquer exemplo, sem qualquer experiência subjectiva associada — exactamente a distinção que o Capítulo 11 fez, com mais detalhe, a propósito de toda a inteligência artificial.

Também não é um processo inteligente a escolher, de forma criativa, como corrigir-se. É, isso sim, um processo cego e sistemático que segue sempre a mesma receita — calcular a perda, propagar para trás, descer o gradiente — e que só parece “aprender” porque essa receita, repetida biliões de vezes sobre dados suficientes, tende, estatisticamente, a produzir pesos que reduzem o erro. Não há nenhum momento em que o polvo “entenda” que Lisboa é a capital de Portugal; há apenas uma sequência muito longa de pequenos ajustes mecânicos que, no final, deixam os seus pesos numa configuração que, perante esta pergunta, produz consistentemente a resposta certa.

É, talvez, esta a imagem mais precisa de tudo o que costumamos chamar “aprendizagem automática”: não uma mente a aprender, mas uma encosta a ser descida, um passo cego de cada vez, biliões de vezes, até o erro ser pequeno o suficiente para a resposta sair certa.

APÊNDICE III DO TOTÓ

Um Ensaio Sem Plano, Mas Com Estrutura

Como é que “uma palavra a seguir à outra” produz introdução, desenvolvimento e conclusão

No Capítulo 8, ao explicar como o polvo gera uma resposta, deixámos uma frase que, lida com atenção, é genuinamente intrigante: o polvo **não tem, em nenhum momento, um plano completo da resposta antes de começar a escrevê-la** — decide uma palavra, depois outra, depois outra, sempre a partir do que já escreveu.

Para uma frase curta, esta ideia é relativamente fácil de aceitar. Mas e para um ensaio de três mil palavras, com introdução, desenvolvimento, contra-argumentos e conclusão? Como é que “escolher sempre a palavra mais provável a seguir” produz, ao fim de várias páginas, um texto que parece — e muitas vezes é, de facto — bem estruturado, coerente do início ao fim, com um argumento que se constrói e se resolve?

A resposta não está em encontrar uma excepção à regra do Capítulo 8 — a regra mantém-se, sem alteração nenhuma. A resposta está em compreender que “sem plano explícito” não significa “sem estrutura nenhuma”. Há, pelo menos, quatro mecanismos distintos que produzem aquilo que parece planeamento, sem que exista, em sentido literal, um plano. Vamos percorrê-los um a um, usando como exemplo um pedido bem concreto: *“Escreve um ensaio sobre se a energia nuclear deve ou não ter um papel central na transição energética, considerando argumentos a favor e contra.”*

1. A mesa de trabalho é maior do que uma frase

A primeira peça do puzzle já a conhece do Capítulo 7: o contexto, a mesa de trabalho onde fica tudo o que foi escrito até ao momento presente. O que talvez não tenha ficado suficientemente claro nesse capítulo é a escala desta mesa e a sua importância directa para esta pergunta: a janela de contexto não guarda apenas a última frase — guarda, tipicamente, **todo o texto já gerado na resposta actual**, desde a primeira palavra.

Isto significa que, ao gerar a palavra número 1.800 de um ensaio sobre energia nuclear, o polvo tem, em cima da mesa, não apenas a frase anterior, mas o pedido original, o título que já escreveu, a introdução completa, os parágrafos a favor que já desenvolveu e qualquer contra-argumento que já tenha começado a esboçar. O mecanismo de atenção do Capítulo 8 — cada token a “olhar” para todos os outros tokens disponíveis — aplica-se com a

mesma força a um texto de três mil palavras como se aplica a uma frase de cinco. A diferença é apenas de escala: há, simplesmente, muito mais material em cima da mesa para consultar.

É aqui que mora a primeira parte da resposta a “como é que sabe que já disse isto, ou que ainda tem de dizer aquilo”: não sabe, no sentido de ter uma lista de tarefas. Mas **vê**, a cada nova palavra, tudo o que já escreveu — e essa visão completa do que já existe é, ela própria, uma forma poderosa de manter coerência, sem que seja necessário qualquer plano armazenado separadamente.

2. A forma de um ensaio já foi vista milhões de vezes

A segunda peça tem a ver com o que o Capítulo 7 chamou de “padrões absorvidos durante o treino” — e aqui vale a pena ser mais específico sobre o que esses padrões incluem.

Durante o treino, o polvo não viu apenas frases isoladas. Viu, repetidamente, **textos inteiros, do princípio ao fim** — ensaios, artigos de opinião, teses, posts de blogue argumentativos — e, com eles, absorveu não só vocabulário e gramática, mas também os padrões estruturais que esses géneros de texto costumam seguir. Um ensaio que pondera dois lados de uma questão tem, estatisticamente, uma forma muito reconhecível: começa por enquadrar o problema, costuma apresentar primeiro um lado (frequentemente introduzido por expressões como “por um lado” ou “os defensores argumentam que”), depois o lado oposto (“por outro lado”, “os críticos apontam”) e tende a fechar com uma síntese ou uma posição ponderada.

Quando o polvo gera a primeira frase do nosso ensaio sobre energia nuclear, ele não está apenas a prever a palavra seguinte com base no significado — está também, implicitamente, a reconhecer **que tipo de texto isto é**, porque o pedido (“considerando argumentos a favor e contra”) activa, no seu mapa de padrões, exactamente o tipo de estrutura textual que viu milhares de vezes associada a pedidos semelhantes. Esta activação não é uma decisão consciente de “vou seguir esta estrutura”; é, antes, o mesmo tipo de processo estatístico do Capítulo 8 — só que, em vez de prever a próxima palavra apenas a partir do significado local, está também a prever, implicitamente, “que tipo de frase costuma vir a seguir a uma introdução deste tipo de texto”.

É por isto que, mesmo sem plano, a primeira frase do ensaio já tende a comportar-se como uma introdução: porque introduções, neste tipo de pedido, são o que estatisticamente costuma vir primeiro.

3. Cada frase escrita estreita as opções da frase seguinte

A terceira peça é, talvez, a mais subtil e ajuda a explicar como a coerência se mantém, frase após frase, sem decair num amontoado de ideias soltas. Voltando à analogia do músico de jazz do Capítulo 7: depois de tocar as primeiras notas de um solo, um bom músico não tem liberdade total sobre o que toca a seguir — as notas já tocadas criam um contexto musical que torna algumas continuações muito mais prováveis e satisfatórias do que outras. Tocar algo completamente desligado, depois de estabelecer um tema, soaria, de imediato, errado.

O mesmo acontece, palavra a palavra, com o nosso ensaio. Se já escreveu um parágrafo a explicar que a energia nuclear produz electricidade com baixas emissões de carbono, a probabilidade estatística da frase seguinte mencionar “transição energética” ou “alternativa aos combustíveis fósseis” sobe consideravelmente — não porque exista um plano a dizer “depois disto, falar de combustíveis fósseis”, mas porque, dado tudo o que já está escrito em cima da mesa, essa é, estatisticamente, uma continuação muito provável e coerente. Inversamente, se a frase seguinte introduzisse, sem qualquer transição, uma observação sobre a gastronomia portuguesa, isso teria uma probabilidade estatística extremamente baixa — não por existir uma regra a proibi-lo, mas porque essa continuação nunca, ou quase nunca, ocorre depois de um contexto deste tipo, nos exemplos de treino que o sistema viu.

Esta auto-consistência cumulativa — cada frase a estreitar, de forma natural, as opções razoáveis para a frase seguinte — é o que evita que um texto longo se desfaça em divagação. Não é planeamento; é o equivalente textual de uma bola de neve: cada escolha feita reduz o espaço de escolhas seguintes razoáveis e esse estreitamento progressivo é, ele próprio, suficiente para produzir um efeito que se parece, de fora, com um plano bem executado.

4. Quando chega a hora do contra-argumento

Vale a pena examinar, com mais detalhe, o momento mais interessante do nosso ensaio: a transição do lado “a favor” para o lado “contra”. Como é que o polvo sabe, sem plano, que chegou a altura de mudar de posição?

A resposta combina as três peças anteriores. Primeiro, o pedido original — “considerando argumentos a favor e contra” — está continuamente disponível na mesa de trabalho e a atenção do polvo, ao gerar cada nova frase, continua a “ver” esse pedido, não apenas o parágrafo imediatamente anterior. Segundo, o padrão estrutural de um ensaio equilibrado, absorvido durante o treino, inclui tipicamente uma extensão razoável dedicada a cada lado — não infinitamente longa, nem demasiado breve — e essa extensão típica influencia a probabilidade de o polvo começar a gerar palavras de transição (como “no entanto”, “por

outro lado”, “os críticos contrapõem”) depois de já ter desenvolvido o lado favorável durante um número de parágrafos estatisticamente típico para este género de texto. Terceiro, a própria menção, na introdução do ensaio, de que ambos os lados seriam considerados, funciona como uma espécie de promessa em cima da mesa de trabalho — uma promessa que, estatisticamente, os textos de treino quase sempre cumprem e que portanto torna muito provável que, a determinada altura, surja a transição para os contra-argumentos.

Nada disto exige que o polvo “saiba” que prometeu algo, no sentido de ter uma memória de intenções. Exige apenas que a sua introdução, com a promessa nela contida, continue visível na mesa de trabalho — e que os padrões aprendidos durante o treino empurrem consistentemente para a realização dessa promessa, exactamente pela mesma lógica estatística que gera qualquer outra palavra.

5. Uma nota sobre rascunhos e múltiplas passagens

Há ainda um desenvolvimento mais recente que vale a pena mencionar, porque ajuda a desfazer de vez a ideia de que tudo isto é um truque de bastidores escondido: alguns sistemas actuais são configurados para, antes de mostrar a resposta final, gerar primeiro uma espécie de rascunho ou esquema interno — um resumo breve de pontos a abordar, por vezes nem mostrado ao utilizador — e só depois escrever o texto completo a partir desse esboço, ou até gerar várias versões e escolher a mais coerente.

Isto não contradiz nada do que foi dito nesta secção — pelo contrário, confirma-o. Esse esboço inicial, quando existe, é gerado **exactamente pelo mesmo processo, palavra a palavra**, que gera qualquer outro texto; simplesmente, em vez de ir directamente para o ensaio final, o sistema usa uma ronda extra de geração para produzir algo parecido com notas de planeamento, que depois entram na mesa de trabalho como mais um elemento de contexto a influenciar a geração do texto final. É planeamento na aparência e no efeito prático — ajuda, de facto, a produzir textos mais bem estruturados — mas é, na sua mecânica interna, construído com as mesmas peças que descrevemos ao longo deste apêndice: tokens, atenção, camadas, probabilidades. Um plano, aqui, não é uma excepção ao mecanismo do Capítulo 8; é mais um produto dele.

O que fica desta viagem

Um ensaio de três mil palavras sobre energia nuclear, ou sobre a existência de Deus, ou sobre qualquer outro tema que exija ponderar argumentos opostos, não nasce de um plano

escrito antecipadamente — nasce de quatro forças a actuar em conjunto, frase após frase: uma mesa de trabalho grande o suficiente para conter tudo o que já foi dito; padrões estruturais de género textual absorvidos de milhões de exemplos durante o treino; uma auto-consistência cumulativa em que cada escolha estreita as escolhas seguintes razoáveis; e, por vezes, uma ronda extra de geração que produz algo parecido com notas de planeamento, através do mesmíssimo mecanismo. O resultado final parece deliberado, estruturado, intencional — e, em todos os sentidos que importam ao leitor, comporta-se como tal. Mas, por debaixo dessa aparência, continua a ser exactamente o que o Capítulo 8 disse desde o início: uma palavra de cada vez, sem nunca, em nenhum momento, existir um plano único e completo guardado em algum lugar antes de a primeira palavra ser escrita.

APÊNDICE IV DO TOTÓ

Pintar a Partir do Ruído*Como nasce uma imagem que nunca existiu*

O Capítulo 6 falou-lhe de imagens como **entrada** — uma fotografia que chega ao polvo já reduzida a uma grelha de píxeis, cada um com um valor entre 0 e 255, prontos a ser processados. Mas há uma pergunta que esse capítulo deixou propositadamente de lado e que merece, agora, o seu próprio apêndice: quando se pede a um sistema de inteligência artificial para **gerar** uma imagem — “um gato a tocar jazz num telhado em Lisboa ao pôr do sol”, por exemplo — de onde vem essa imagem? Não existia antes. Não está escondida em nenhuma base de dados, à espera de ser encontrada. É, em todos os sentidos relevantes, nova.

A resposta tem um nome técnico — **modelo de difusão** — e o mecanismo por detrás dele é, surpreendentemente, uma inversão quase poética de tudo o que o Capítulo 6 ensinou sobre ruído. Lembra-se de que, nesse capítulo, o ruído era tratado como o inimigo a remover, a informação indesejada que confunde o sistema? Pois bem: para gerar uma imagem nova, o polvo começa precisamente **dentro do ruído puro** — e a sua tarefa é, passo a passo, andar para trás, do ruído até à imagem.

1. O processo ao contrário: estragar uma fotografia, de propósito

Para compreender como um sistema aprende a gerar imagens a partir de ruído, é mais fácil começar pelo processo inverso — mais simples de descrever e que é, de facto, o primeiro passo do treino.

Imagine uma fotografia perfeitamente nítida de um gato. O processo de treino começa por lhe **adicionar ruído**, deliberadamente, num pequeno grau — alguns píxeis ficam ligeiramente distorcidos, como granulado televisivo muito ténue sobreposto à imagem. A fotografia continua claramente reconhecível como um gato, mas já não está perfeita.

Este processo repete-se: adiciona-se mais ruído à imagem já levemente distorcida e mais um pouco e mais um pouco, em centenas ou mesmo milhares de pequenos passos sucessivos. A cada passo, a imagem fica menos nítida e mais granulada. Depois de passos suficientes, o resultado final já não tem absolutamente nenhuma relação visual com o gato original — é apenas um campo de pontos aleatórios, indistinguível de ruído puro, do tipo

que se vê numa televisão antiga sem sinal. Tecnicamente, sabemos exactamente quanto ruído foi adicionado em cada passo, porque fomos nós (ou melhor, o processo de treino) a adicioná-lo, de forma controlada e mensurável.

2. O que o polvo aprende a fazer: o caminho inverso

Aqui está a ideia central de um modelo de difusão e vale a pena lê-la duas vezes: o polvo é treinado para fazer, exactamente, **o percurso inverso** — olhar para uma imagem com ruído e prever que ruído, especificamente, foi adicionado num determinado passo, para depois o poder subtrair.

Durante o treino, o polvo vê milhões de pares de imagens — a versão com um certo grau de ruído e o ruído exacto que foi adicionado nesse passo específico — e aprende, através do mesmo mecanismo de retropropagação e descida do gradiente que vimos no Apêndice II, a prever esse ruído com precisão crescente. Não está a aprender “o que é um gato” no sentido de memorizar uma fotografia; está a aprender uma capacidade muito mais geral: dada uma imagem ruidosa qualquer e sabendo aproximadamente quanto ruído lhe foi adicionado, qual é, estatisticamente, o padrão de ruído mais provável presente nela.

Uma vez que o polvo sabe fazer isto — prever e remover um pequeno passo de ruído — pode fazê-lo repetidamente, em sentido inverso ao do treino: começa com ruído puro, prevê e remove um pouco desse ruído, obtendo algo ligeiramente menos aleatório; repete o processo com esse resultado, removendo mais um pouco; e assim sucessivamente, até que aquilo que começou como um campo de pontos completamente aleatórios se transforma, gradualmente, numa imagem nítida e coerente. Vale a pena notar uma assimetria interessante aqui: enquanto o processo de treino (estragar imagens, passo a passo) costuma usar centenas ou milhares de passos para garantir que o sistema aprende bem a tarefa, o processo de geração de uma imagem nova, já com o sistema treinado, usa atalhos matemáticos que permitem obter um resultado igualmente nítido em muito menos passos — tipicamente entre vinte e cinquenta, como veremos a seguir. É por isso que gerar uma imagem demora segundos, não horas.

3. Onde entra o seu pedido de texto

Há, contudo, um problema óbvio com o que descrevemos até agora: se o processo começasse sempre com ruído puro e não tivesse mais nenhuma informação, o resultado final seria uma imagem qualquer, coerente mas completamente aleatória — não necessariamente um gato e muito menos um gato a tocar jazz num telhado em Lisboa.

É aqui que entra o seu pedido de texto e a ligação com tudo o que já sabe deste livro torna-se, de repente, muito directa: a frase que escreveu é cortada em tokens, convertida em vectores e processada pelo mecanismo de atenção do Capítulo 8 — exactamente como qualquer outro texto. A diferença é que, em vez de essa representação ser usada para prever a próxima palavra, ela é usada para **guiar** o processo de remoção de ruído, em cada uma das suas dezenas de passos. Em cada passo, o polvo não pergunta apenas “que ruído está provavelmente presente nesta imagem?” — pergunta “que ruído está provavelmente presente nesta imagem, **dado que o resultado final deve parecer-se com a descrição ‘um gato a tocar jazz num telhado em Lisboa ao pôr do sol’?**”

Esta orientação por texto empurra, a cada um das dezenas de pequenos passos, o processo de remoção de ruído numa direcção consistente com o pedido — não através de uma regra explícita do tipo “desenhar orelhas de gato aqui”, mas através do mesmo tipo de padrão estatístico aprendido durante o treino que já vimos em todos os capítulos anteriores: o sistema viu, no seu treino, uma enorme quantidade de pares texto-imagem e aprendeu que certas combinações de palavras tendem a corresponder a certos padrões visuais. É a mesma lógica de “Portugal” e “capital” a apontarem estatisticamente para “Lisboa”, no Apêndice I — só que, aqui, o destino final não é uma palavra, é uma imagem inteira, construída passo a passo a partir do caos.

Vale a pena tornar isto mais concreto com alguns números de escala. Um processo de geração de imagem típico não envolve apenas alguns passos — envolve, tipicamente, entre vinte e cinquenta rondas de remoção de ruído, cada uma a refinar ligeiramente o resultado da anterior. E, em cada uma destas rondas, a representação da imagem ainda ruidosa — uma grelha de números, exactamente como vimos no Capítulo 6 e no Apêndice I, só que aqui em permanente transformação — é processada em conjunto com a representação do texto do pedido, através de um mecanismo de atenção muito semelhante ao do Capítulo 8, mas a fazer algo ligeiramente diferente: em vez de cada token de uma frase a “olhar” para os outros tokens da mesma frase, é cada região da imagem (cada pequeno grupo de píxeis) a “olhar” para os tokens do texto, perguntando, implicitamente, “que parte deste pedido é mais relevante para o que deve aparecer exactamente aqui?”. É esta atenção cruzada, entre imagem e texto, repetida em cada uma das vinte a cinquenta rondas, que garante que o gato acaba a tocar jazz e não, por exemplo, a dormir tranquilamente — e que esse gato acaba mesmo num telhado e não dentro de uma sala.

4. Porque é que nunca sai a mesma imagem duas vezes

Isto explica, de forma bastante directa, um fenómeno que provavelmente já notou: pedir a um sistema destes a mesma descrição duas vezes produz, quase sempre, duas imagens visivelmente diferentes — ambas coerentes com o pedido, mas nunca idênticas.

A razão é simples e já lhe é familiar do Capítulo 8: o ponto de partida — o campo de ruído puro — é gerado aleatoriamente, de forma diferente a cada vez, exactamente como a temperatura introduzia uma margem de variação na escolha de palavras. Partindo de ruídos iniciais diferentes, mesmo guiado pela mesma descrição de texto, o processo de remoção gradual de ruído percorre caminhos ligeiramente diferentes através de todas as imagens possíveis que correspondem a essa descrição — e chega, por isso, a gatos visivelmente distintos, todos eles a tocar jazz, todos eles em telhados, todos eles ao pôr do sol, mas nunca a mesma imagem do mesmo gato.

5. A mesma lógica, partida em pedaços diferentes

Vale a pena fechar este apêndice com uma observação que ajuda a situar tudo isto dentro do resto do livro: um modelo de difusão não é uma inteligência artificial fundamentalmente diferente das que descrevemos nos capítulos anteriores — usa pesos, treinados por descida do gradiente, organizados em camadas e pode até usar mecanismos de atenção semelhantes aos do Capítulo 8 para relacionar diferentes partes da imagem entre si e com o texto do pedido. O que muda, fundamentalmente, é a **tarefa** para a qual esses pesos foram treinados: em vez de prever a próxima palavra de um texto, foram treinados para prever o ruído presente numa imagem, um passo de cada vez.

É uma boa lembrança de algo que este livro tem vindo a repetir, de várias formas, desde o primeiro capítulo: por debaixo de capacidades que parecem profundamente diferentes — escrever um ensaio, responder a uma pergunta factual, pintar um gato que nunca existiu — está, invariavelmente, a mesma receita básica: pesos ajustados por treino, organizados de forma a captar padrões estatísticos a partir de quantidades imensas de exemplos e usados depois para produzir, passo a passo, algo novo que é estatisticamente coerente com esses padrões. A “mágica”, se é que existe alguma palavra que mereça esse nome neste livro, está sempre na mesma receita — só os ingredientes de entrada e o tipo de saída é que mudam.

Uma curiosidade final

Se já alguma vez gerou uma imagem por IA e reparou que o processo, em vez de aparecer instantaneamente, mostra uma sucessão de versões cada vez mais nítidas até à imagem final — uma espécie de imagem a “revelar-se” gradualmente, como uma fotografia analógica num tanque de químicos — não é um efeito visual decorativo. É, literalmente, o que está a acontecer: cada uma dessas versões intermédias é um passo real do processo de remoção de ruído que descrevemos nesta secção e o sistema está, de facto, a torná-las visíveis à medida que progride.

APÊNDICE V DO TOTÓ

Quando o Polvo Ganha Mãos

De “responder” para “agir”: como a IA deixou de ser só conversa

Ao longo deste livro, o nosso polvo teve sempre o mesmo papel: receber uma pergunta, processar os tokens, atravessar as camadas, e cuspir uma resposta. Um percurso linear, do início ao fim, que termina quando a última palavra aparece no ecrã. É este modelo — uma pergunta entra, uma resposta sai — que sustenta tudo o que descrevemos nos Capítulos 6, 7, 8 e 9.

Mas há algo a mudar, depressa, no mundo da inteligência artificial de 2026, e este apêndice existe para explicar o quê: o polvo ganhou mãos. Em vez de ficar sentado à mesa a responder perguntas, está a aprender a levantar-se, a ir buscar informação, a executar tarefas, a verificar resultados, e a repetir o processo até o trabalho estar feito. A este tipo de sistema chamamos **agente de IA**, e perceber como funciona — e o que nele é genuinamente novo — é a tarefa deste apêndice.

1. O que muda: de linha recta a ciclo

O Capítulo 8 descreveu o polvo como uma fábrica que trabalha em linha recta: os tokens entram por um lado, passam pelas camadas, e a resposta sai pelo outro. Mesmo quando o polvo gera várias centenas de palavras, o mecanismo é sempre o mesmo — uma palavra de cada vez, da esquerda para a direita, até terminar.

Um agente de IA funciona de forma diferente. Em vez de um único percurso linear, o ciclo repete-se:

```
graph TD; A[Pensar] --> B[Decidir uma acção]; B --> C[Executar essa acção]; C --> D[Observar o resultado]; D --> E[Pensar de novo]; E --> F[...]; F --> G[Resposta final];
```

Este padrão — que os investigadores descrevem frequentemente como **Think** → **Act** → **Observe** — é o que distingue um agente de um assistente de conversação simples. Não é

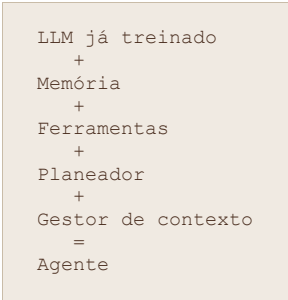
apenas uma resposta mais longa; é uma arquitectura diferente em volta do mesmo polvo.

As acções que o agente pode executar dependem das ferramentas a que tem acesso. Tipicamente incluem coisas como pesquisar na internet, consultar uma base de dados, executar código, abrir e escrever ficheiros, usar uma calculadora, chamar uma API externa, ou até invocar outro sistema de IA. A lista é, em princípio, ilimitada — o que define um agente não são as ferramentas específicas, mas a capacidade de as usar dentro de um ciclo de raciocínio e acção.

2. O polvo não mudou — mudou o que está à sua volta

Aqui está o ponto mais importante deste apêndice, e aquele que mais surpreende quem chega a este tema pela primeira vez: **o polvo em si, na maioria dos casos, não mudou nada.** A arquitectura Transformer que descrevemos nos capítulos anteriores, com o mecanismo de atenção a calcular relações entre tokens, continua exactamente igual. O que mudou foi o ambiente em volta dele.

Um agente de IA é, tipicamente, um LLM já treinado ao qual se acrescentaram quatro peças externas:



```
LLM já treinado
+
Memória
+
Ferramentas
+
Planeador
+
Gestor de contexto
=
Agente
```

Grande parte do que hoje se chama “IA agência” é, na verdade, **engenharia de sistemas** — não uma nova forma de treinar o polvo, mas uma nova forma de orquestrar o que está à sua volta. O polvo continua a ser o mesmo; o que mudou foi a cozinha.

3. A mesa de trabalho do agente

Lembra-se, do Capítulo 7, da mesa de trabalho do contexto — o espaço onde fica tudo o que o polvo pode “ver” num dado momento? Num assistente de conversação simples, essa mesa contém a conversa: as perguntas do utilizador e as respostas do sistema, acumuladas desde o início da sessão.

Num agente, a mesa é muito mais cheia. A cada passo do ciclo, pode conter o objectivo

original que o utilizador definiu, o histórico das acções já executadas, os resultados dessas acções (o output de uma pesquisa, o resultado de um cálculo, o conteúdo de um ficheiro), o estado actual da tarefa, e um plano, se o agente o tiver gerado.

O mecanismo de atenção do Capítulo 8 continua a operar exactamente da mesma forma — cada token a “olhar” para todos os outros — mas agora os tokens incluem muito mais do que palavras de uma conversa. O “estado mental” do agente não está guardado numa memória interna permanente da rede neuronal; está na mesa de trabalho, no contexto que é fornecido ao polvo a cada passo. O polvo não “lembra” o que fez; *vê* o registo do que foi feito, porque esse registo está em cima da mesa.

Isto também explica um limite real dos agentes actuais: se o ciclo de raciocínio e acção for longo o suficiente para encher a mesa de trabalho, os passos mais antigos começam a “cair da mesa”, exactamente como vimos no Capítulo 7. Gerir o contexto de um agente — decidir o que fica na mesa e o que é descartado — é um dos problemas mais activos de engenharia nesta área.

4. Uma surpresa sobre o treino

Há uma pergunta natural que surge aqui: se um agente tem capacidades tão diferentes de um assistente de conversação, não precisará de ser treinado de forma completamente diferente?

A resposta é, na maioria dos casos, não. O LLM que serve de motor a um agente é treinado exactamente como descrevemos no Capítulo 2 e no Apêndice II — pré-treino com quantidades imensas de texto, seguido de fine-tuning e RLHF. O comportamento agência emerge, sobretudo, da arquitectura que o envolve, não de um treino radicalmente diferente.

Dito isto, há uma tendência mais recente que vale a pena mencionar: alguns modelos são treinados com exemplos de *trajetórias* completas de agente — sequências do tipo “utilizador fez esta pergunta → sistema pensou assim → escolheu esta ferramenta → obteve este resultado → pensou de novo → respondeu”. O modelo aprende, por imitação, que certos tipos de perguntas devem desencadear determinadas sequências de acção, em vez de uma resposta directa.

Imagine, por exemplo, um modelo treinado com exemplos como:

```
Utilizador: "Qual a temperatura em Lisboa e quanto demora o comboio até ao Porto?"

Pensar: esta pergunta precisa de dois dados em tempo real...
↓
Usar API meteorológica → temperatura: 22°C
↓
Usar API ferroviária → duração: 2h45m
↓
Responder: "Em Lisboa estão agora 22°C. O comboio até ao Porto demora cerca de 2h45m."
```

O modelo que viu milhares de trajectórias deste tipo durante o treino aprende a activar este padrão de forma natural — não porque alguém programou a regra “quando a pergunta inclui dados em tempo real, usa ferramentas”, mas pela mesma lógica estatística que faz o polvo produzir “Lisboa” quando lhe perguntam a capital de Portugal.

5. O polvo com secretária

Imagine o nosso polvo sentado numa sala vazia. Tem toda a sua inteligência disponível — os pesos, os padrões aprendidos durante o treino, a capacidade de raciocinar e de gerar texto. Mas está sozinho, sem acesso a nada de exterior. Quando lhe fazemos uma pergunta, processa os tokens, atravessa as camadas, e produz a melhor resposta que consegue com o que sabe de cor. É o assistente de conversação do Capítulo 8.

Agora imagine esse mesmo polvo, mas com uma secretária cheia de recursos à sua disposição: um computador com acesso à internet, uma calculadora, livros de referência actualizados, um calendário, um editor de código, e a possibilidade de enviar mensagens a outros sistemas. Quando encontra uma dúvida durante o raciocínio, pode parar, pedir à secretária que consulte uma fonte, receber o resultado, e retomar o pensamento com essa nova informação. O cérebro do polvo não mudou — mudou o ambiente em que trabalha e a capacidade de interagir com ele.

Esta é, no fundo, a diferença entre um LLM e um agente: não é o polvo que é diferente, é a sala onde está instalado.

6. O que pode correr mal — e porquê importa

Os agentes de IA introduzem riscos que os assistentes de conversação simples não têm, precisamente porque as suas acções têm consequências no mundo real. Uma resposta errada num chat é um incómodo; uma acção errada num agente — um email enviado para a pessoa errada, uma compra não autorizada, um ficheiro eliminado — pode ter consequências difíceis de reverter.

Há três tipos de falha que os investigadores e engenheiros desta área observam com mais frequência. O primeiro é a **deriva de objectivo**: o agente começa com um objectivo claro mas, ao longo dos ciclos de raciocínio, vai interpretando-o de formas cada vez mais distantes da intenção original. O segundo é o **loop infinito**: o agente fica preso num ciclo de raciocínio e acção sem nunca concluir a tarefa, porque cada resultado gera nova incerteza em vez de a reduzir. O terceiro é a **propagação de erro**: um erro num passo inicial é tratado como facto nos passos seguintes, e as acções subsequentes são construídas sobre uma base incorrecta, amplificando o problema em vez de o corrigir.

Nenhum destes riscos é exclusivo da IA — também os humanos derivam de objectivos, ficam presos em loops, e propagam erros. Mas num agente de IA, estes problemas podem acontecer a uma velocidade muito superior à humana, e as ferramentas à disposição do agente amplificam as consequências. É por isso que os sistemas agênticos mais responsáveis incluem mecanismos de supervisão humana — pontos de verificação onde um humano confirma antes de o agente executar acções de alto impacto.

Uma nota sobre o futuro

É provável que, nos próximos anos, a distinção entre “LLM” e “agente” se torne cada vez menos nítida. Os modelos estão a ser treinados de forma crescente para alternar naturalmente entre raciocinar, planear e usar ferramentas, tornando o comportamento agência uma capacidade integrada no próprio modelo, em vez de uma camada construída à sua volta. Ainda assim, no núcleo de qualquer sistema agêntico continua a existir o mesmo princípio fundamental que este livro descreveu desde o início: um Transformer que usa o mecanismo de atenção para processar o contexto e prever o próximo passo — que tanto pode ser um token de texto como a decisão de executar uma acção no mundo. O polvo ganhou mãos. Mas continua a ser o mesmo polvo.

APÊNDICE VI DO TOTÓ

As Perguntas do Totó

Tudo o que sempre quis saber sobre IA (e nunca teve coragem de perguntar)

Por baixo de tudo, há matemática

Antes de mergulhar nas perguntas, vale a pena dizer uma coisa em voz alta: tudo o que a inteligência artificial faz — escrever, traduzir, gerar imagens, compor música, jogar xadrez — assenta em quatro ramos da matemática que trabalham em conjunto como peças de um mesmo mecanismo.

A álgebra linear é a língua em que os dados são escritos. Uma imagem, uma frase, uma música — tudo se transforma em listas e grelhas de números (vectores e matrizes) que o sistema pode manipular. É com ela que o polvo recebe os ingredientes.

O cálculo diferencial é o professor que corrige os erros. Quando o sistema erra uma resposta, o cálculo mede exactamente quanto errou e em que direcção deve ajustar cada peso para errar menos da próxima vez. É o motor da aprendizagem, o mecanismo de retropropagação descrito no Apêndice II deste livro.

A probabilidade e estatística é a bússola da incerteza. O mundo real é ruidoso e os dados são amostras imperfeitas. A probabilidade dá ao sistema a capacidade de lidar com essa imperfeição e de generalizar. Quando o polvo diz que “Lisboa” tem 91% de probabilidade de ser a próxima palavra, é a estatística a falar.

A teoria da optimização é o estratega que escolhe o caminho. Saber que há um erro e conhecer a direcção para o corrigir não chega, é preciso decidir o tamanho do passo e a velocidade da descida. É ela que transforma a paisagem montanhosa do Apêndice II numa descida eficiente.

Não é preciso dominar nenhum destes ramos para usar inteligência artificial, nem para compreender este livro. Mas é útil saber que existem, que por baixo de cada resposta que parece mágica há cálculos, probabilidades e optimização a acontecer em paralelo, em fracções de segundo, a uma escala que desafia a intuição humana.

1. A Inteligência Artificial pensa e sente como um humano?

Não. A IA não pensa, não sente, não tem opiniões nem consciência. O que ela faz é processar padrões matemáticos para gerar respostas que ‘parecem’ humanas. É como um papagaio muito avançado que aprendeu a combinar palavras com base em bilhões de exemplos que leu, mas sem nunca entender o que elas significam. Este livro inteiro explica, passo a passo, o mecanismo por detrás dessa aparência — e o que encontramos nunca é pensamento, é matemática com escala.

2. A IA é uma pessoa ou um robô físico?

Nenhum dos dois. Quando falas com o ChatGPT ou com outro assistente, não estás a falar com um robô de metal. Estás a falar com um modelo matemático — um ficheiro enorme com números — alojado em servidores. Esse ficheiro não tem braços, pernas ou olhos; é puro cálculo.

3. Como é que a IA aprendeu a falar português?

Ela não “aprendeu” como uma criança. Durante o treino, os programadores alimentaram-na com bilhões de textos em português — livros, sites, conversas. Ela não os decorou, mas analisou-os estatisticamente: “Depois da palavra ‘boa’, a palavra ‘noite’ aparece 70% das vezes.” Assim, constrói frases com base em probabilidades, não em compreensão.

4. A IA pode inventar informações falsas?

Sim, e isso chama-se **alucinação**. Como ela não distingue verdade de mentira — apenas probabilidade — se a próxima palavra mais provável for uma invenção, escreve-a com a mesma confiança com que escreveria um facto real. Por isso, é importante confirmar informações importantes em fontes independentes.

5. A IA substitui a criatividade humana?

Não substitui, mas é uma ferramenta poderosa. Ela consegue combinar ideias que já existem de formas novas, mas não tem originalidade genuína. A verdadeira criatividade — sentir, ter intenção, quebrar regras de propósito — continua a ser exclusivamente humana. A IA pode ser o pincel; o artista és tu.

6. Como é que a IA transforma uma pergunta numa resposta?

O processo é uma cascata de cálculos: a tua pergunta é convertida em números (vectores).

Esses números passam por centenas de camadas matemáticas, onde são multiplicados e somados com bilhões de outros números — os “pesos”. No final desta viagem, o modelo gera uma lista de probabilidades para a próxima palavra, escolhe uma, acrescenta-a ao contexto, e repete o processo até completar a resposta. Tudo isto é descrito em detalhe no Capítulo 8 deste livro.

7. O que são os “pesos” da IA? E porque é que são tão importantes?

Pesos são números — por exemplo 0,745; -2,31; 1,02 — que funcionam como botões de ajuste. Durante o treino, a IA ajusta esses números para que, ao receber uma pergunta, a resposta saia o mais correcta possível. Quanto mais pesos, mais complexa a IA pode ser. Os maiores modelos actuais têm bilhões de parâmetros: o GPT-4, por exemplo, tem estimadamente $1,8 \times 10^{12}$ (1,8 bilhões no sentido português, ou seja, 1,8 milhões de milhões); modelos mais recentes rondam os 3 bilhões. Modelos mais simples têm apenas alguns milhões.

8. Onde é que a IA guarda toda a informação que leu? Não ocupa imenso espaço?

A IA não guarda os textos que leu. Guarda apenas os padrões matemáticos extraídos desses textos. É como se lesse um milhão de livros e depois os queimasse, ficando apenas com um “instinto” sobre como as palavras se relacionam. Por isso, o ficheiro do modelo ocupa “apenas” algumas dezenas ou centenas de gigabytes, mesmo tendo processado quase toda a internet. Um modelo grande com 1,8 bilhões de parâmetros, armazenado com 2 bytes por parâmetro, ocupa cerca de 3,6 TB na precisão padrão — mas os engenheiros comprimem estes ficheiros para formatos mais eficientes (como INT4, com 0,5 bytes por parâmetro), reduzindo o tamanho para cerca de 900 GB.

9. Porque é que a IA às vezes dá respostas muito longas ou repete a mesma coisa?

Por duas razões: pode estar presa num ciclo de alta probabilidade (certas palavras de ligação são sempre muito prováveis, criando loops), ou a **temperatura** — o parâmetro que controla a aleatoriedade na escolha de palavras — está mal calibrada. Com temperatura baixa, a IA torna-se repetitiva e previsível; com temperatura alta, torna-se variada, mas por vezes incoerente.

10. Porque é que a IA tem um limite no tamanho do texto que aceita (janela de contexto)?

Esse limite existe porque o mecanismo de atenção — que permite à IA relacionar cada

palavra com todas as anteriores — tinha, na arquitectura original, um custo computacional que crescia com o quadrado do tamanho do texto: dobrar o texto custava quatro vezes mais. Modelos modernos reduziram este custo com técnicas como atenção esparsa e flash attention, o que permitiu ampliar os limites (actualmente 128 mil tokens ou mais em alguns modelos), mas não o eliminou. A janela de contexto continua a ser uma troca entre capacidade e custo em energia e computação.

11. O que é o RAG e porque é que permite à IA pesquisar na internet antes de responder?

RAG significa *Retrieval-Augmented Generation* — geração de texto reforçada por pesquisa. É uma técnica que permite ao sistema ir consultar uma fonte externa (a internet, uma base de dados privada, um conjunto de documentos) antes de gerar a resposta, em vez de depender apenas do que ficou gravado durante o treino. O processo é: recebe a pergunta → pesquisa informação relevante → coloca essa informação na “mesa de trabalho” do contexto → gera a resposta com base em tudo isso em conjunto. É por isso que alguns assistentes conseguem dizer-te o resultado de ontem à noite, mesmo que os seus dados de treino sejam de meses atrás. O RAG é explicado em detalhe no Capítulo 9 deste livro.

12. O que são os “agentes de IA” e em que diferem de um assistente normal?

Um assistente de IA normal recebe uma pergunta e devolve uma resposta — um único percurso linear. Um agente de IA vai mais longe: recebe um objectivo e executa um ciclo de acção repetido — pensar, decidir uma acção, executar essa acção, observar o resultado, pensar de novo. As acções podem incluir pesquisar na internet, abrir ficheiros, executar código, enviar emails ou chamar outras IAs. O motor continua a ser o mesmo LLM, mas agora tem ferramentas à sua disposição e um ciclo que se repete até o trabalho estar feito. É a diferença entre um escritor numa sala vazia e esse mesmo escritor com um computador, uma calculadora e acesso à internet — o cérebro não mudou, mudou o ambiente.

13. Se a IA só prevê a próxima palavra, como é que ela sabe onde colocar vírgulas, pontos e parágrafos?

A pontuação, para a IA, é exactamente igual a uma palavra: são tokens. Durante o treino, ela percebeu que, em certos contextos, a probabilidade de aparecer uma vírgula ou um ponto final é elevada. Ela não conhece regras gramaticais; apenas imita os padrões de pontuação que viu em biliões de textos. É por isso que, se o padrão indicar, pode colocar vírgulas a mais ou a menos.

14. Quando peço para corrigir um texto, a IA usa um dicionário ou um corrector ortográfico?

Não. O que ela faz é reescrever o texto do zero. Lê a tua versão, capta o contexto, e gera uma nova versão que tem maior probabilidade estatística de estar correcta, com base em tudo o que leu. É como se um professor de línguas lesse o teu texto e o reescrevesse de forma mais elegante — mas sem saber explicar as regras que está a aplicar.

15. Como é que a IA imita o estilo de um escritor (ex: Eça de Queirós) se só tem números no ficheiro?

Dentro dos pesos numéricos, estão guardadas as relações geométricas entre palavras. Eça de Queirós usa vocabulário específico — “ramalhete”, “aristocracia”, “decadência”, “ironia” — frases longas com orações subordinadas e um tom crítico. A IA não copia frases dele; quando pedes “estilo Eça de Queirós”, ela activa os caminhos numéricos que levam a esse vocabulário e a essa ironia subtil. É como navegar num mapa onde “Eça” é uma coordenada específica — e ao chegar lá, as palavras que emergem são as que, estatisticamente, rodeiam essa coordenada nos dados de treino.

16. A IA sabe qual é a finalidade do texto que está a gerar (email formal vs. post informal)?

Não, no sentido de ter intenção. A “finalidade” é um gatilho estatístico que tu activas pelo prompt. Se escreveres “email profissional”, ela gera texto formal porque, estatisticamente, textos com a palavra “profissional” usam mais “Venho por este meio”. A finalidade está nas tuas instruções, não no “pensamento” da IA.

17. Como é que a IA traduz idiomas sem saber gramática de nenhum deles?

As palavras que significam a mesma coisa em diferentes línguas ficam matematicamente próximas no espaço interno da IA, mesmo sendo de idiomas diferentes. Traduzir é, para ela, navegar nesse espaço geométrico partilhado: descobre onde está “cão” em português e procura o ponto mais próximo em inglês — e encontra “dog”. Não usa dicionários nem regras, apenas geometria.

18. Há risco de a IA copiar literalmente um texto que leu durante o treino? Isso não é plágio?

Para textos longos e únicos, a probabilidade é praticamente zero. A IA não decora; reconstrói. Só há risco de cópia exacta em textos que aparecem milhões de vezes idênticos

(como a Constituição ou código “Hello World”). Para o resto, o que gera é uma combinação inédita de padrões, não uma fotocópia. Citações muito famosas são a exceção onde o risco é maior.

19. Se eu fizer a mesma pergunta duas vezes, vou obter a mesma resposta?

Provavelmente não. A IA tem um parâmetro chamado temperatura que adiciona um factor controlado de aleatoriedade à escolha das palavras. Com temperatura zero, a resposta é sempre a mesma — a mais provável. Com temperatura mais alta (o padrão na maioria dos assistentes), pode escolher a segunda ou terceira opção mais provável, gerando respostas diferentes mas igualmente coerentes.

20. A IA pode ser utilizada para fins maliciosos?

Sim, como qualquer ferramenta poderosa. Pode gerar notícias falsas, emails de phishing ou código malicioso. Por isso, os criadores colocam barreiras de segurança (chamadas “alinhamento” ou alignment) para tentar bloquear esses usos. Mas essas barreiras são imperfeitas, e há sempre tentativas de as contornar — razão pela qual a investigação em segurança de IA é uma área activa e crescente.

21. Se a IA é tão boa a imitar, porque é que às vezes diz coisas sem sentido ou contraditórias?

Porque não tem um modelo do mundo nem raciocínio lógico formal. Ela junta palavras prováveis — e a coerência lógica é um subproduto dessa probabilidade, não uma garantia. Se fizeres uma pergunta com uma premissa falsa, ela vai inventar uma resposta plausível porque as palavras da pergunta já apareceram juntas em contextos de ficção ou hipótese. A IA não distingue o impossível físico do improvável estatístico.

22. Como é que uma IA gera uma imagem a partir de uma descrição textual?

Ela não “desenha” como um humano. O processo começa com ruído puro — como a estática de uma televisão antiga — e vai retirando esse ruído, passo a passo, até sobrar a imagem. É como esculpir: comesas com um bloco de mármore (ruído) e, camada após camada, a IA “talha” os pixels até formarem o gato, a guitarra e o telhado. O texto serve de mapa que guia cada etapa do processo. Este mecanismo é explicado em detalhe no Apêndice IV deste livro.

23. A IA copia imagens que viu durante o treino para criar as minhas?

Não, e a prova é matemática: uma imagem tem milhões de pixels, e o espaço de combinações possíveis é astronomicamente vasto. A IA aprendeu conceitos — o que é um “gato”, o que é uma “guitarra” — e recombina esses conceitos em posições e cores inéditas. É como um pintor que estudou dez mil gatos e agora pinta um gato que nunca existiu.

24. Porque é que as IAs de imagem têm dificuldade em gerar mãos, olhos ou texto legível?

Porque esses elementos têm estruturas rígidas e específicas — uma mão tem sempre cinco dedos, uma letra “R” tem de ter um traço curvo específico. A IA aprende padrões estatísticos, mas não regras geométricas rígidas. Como as mãos aparecem em inúmeras posições e ângulos nos dados de treino, o modelo fica “confuso” e gera seis dedos ou dedos fundidos. Falta-lhe uma “régua” interna que garanta a contagem exacta.

25. Se a geração de uma imagem começa com ruído e vai sendo “limpa”, a IA sabe como vai ser a imagem final?

A resposta é surpreendente: a IA não sabe qual é a imagem final, nem nunca vê o destino. Em cada passo, recebe apenas um empurrão matemático — um gradiente — que lhe diz: “Se ajustares os pixels nesta direcção, a probabilidade de o resultado se parecer com a descrição aumenta.” O texto “gato a tocar guitarra” é convertido num vector numérico. A IA faz uma soma nesse espaço de muitas dimensões: gato + tocar + guitarra. Essa soma aponta para uma região inédita do espaço — uma combinação que nunca foi visitada durante o treino. A IA gera a imagem que melhor corresponde a esse ponto. É por isso que às vezes gera gatos com três orelhas: a geometria, nessa zona, é ambígua.

26. O que são os “Modelos de Difusão” (como o Midjourney ou o Stable Diffusion) e porque são tão populares?

São o estado da arte actual para geração de imagens. Funcionam por difusão reversa: primeiro, durante o treino, a IA aprende a estragar imagens — adicionando ruído até estas virarem estática. Depois, aprende a fazer o caminho inverso: remover o ruído para reconstruir a imagem. Durante a geração, parte da estática e aplica o caminho inverso, guiada pelo texto. São populares porque produzem imagens com maior qualidade e diversidade do que os modelos anteriores (as *GANs* — Generative Adversarial Networks).

27. É possível criar imagens em alta resolução (4K, 8K) com IA?

Sim, mas com um truque. Gerar uma imagem 4K directamente seria computacionalmente proibitivo. A solução é gerar a imagem em baixa resolução e depois usar um modelo separado de upscaling que “inventa” os detalhes extra com base nos padrões que conhece. É como pegar numa foto pequena e pedir a um restaurador de arte que a pinte em tamanho gigante, preenchendo os espaços vazios com pinceladas coerentes.

28. Gerar vídeo com IA é mais difícil do que gerar imagens?

Muito mais. Uma imagem é uma “fatia” no tempo; um vídeo é uma sequência de 24 a 30 fatias por segundo, com a exigência de que o objecto mantenha a mesma aparência e se mova de forma suave entre fatias consecutivas. A IA tem de garantir consistência temporal — o que é matematicamente muito mais complexo. Por isso, os vídeos gerados por IA são muitas vezes curtos e ainda apresentam “alucinações” visuais: braços que derretem, fundos que mudam de cor de frame para frame.

29. Como é que a IA cria um vídeo a partir de uma frase?

Usa um modelo de difusão aplicado ao espaço-tempo. Ela gera não apenas a posição dos pixels — como na imagem — mas também a trajectória de cada pixel ao longo dos frames. O processo é semelhante ao da imagem: parte de ruído estático e vai “esculpindo” o vídeo, frame a frame, com uma restrição extra: o frame seguinte tem de ser uma evolução natural do anterior.

30. Porque é que os vídeos de IA às vezes parecem “sonhos” estranhos ou com fluidez artificial?

Porque a IA ainda não percebe bem as leis da física — gravidade, sombras, reflexos — nem a biomecânica de como os corpos se movem. Ela gera o que é estatisticamente provável, não o que é fisicamente correcto. O resultado é um vídeo que parece impressionante num frame isolado, mas que ao mover-se revela pequenas inconsistências — objectos que flutuam, sombras que não acompanham o corpo.

31. Quanto tempo demora a gerar um vídeo com IA?

Muito mais do que uma imagem. Enquanto uma imagem leva segundos, um vídeo de cinco segundos pode levar vários minutos ou horas, dependendo da resolução e complexidade. Para cada frame, a IA recalcula a posição de milhões de pixels, garantindo coerência com os frames vizinhos. O custo computacional escala rapidamente com a duração e a resolução.

32. Como é que a IA compõe música? Ela conhece teoria musical?

Não conhece teoria musical no sentido humano — escalas, acordes, harmonia. Mas durante o treino analisou milhões de partituras e áudios, transformando as notas em sequências numéricas, tal como as palavras. Quando pedes “uma música triste com piano”, ela gera uma sequência de notas que, estatisticamente, aparece associada à palavra “triste” e ao timbre “piano”. A harmonia emerge dos padrões, não de regras escritas.

33. A IA consegue imitar a voz de um cantor específico?

Sim, através de clonagem de voz (*TTS — Text-to-Speech*). Com apenas alguns segundos de áudio da pessoa, a IA extrai as características acústicas únicas — timbre, entoação, respiração — e cria um modelo matemático que reproduz essas características. Depois, ao receber novo texto, gera áudio com a “máscara” daquela voz. É usado em dobragens, mas levanta questões éticas sérias, nomeadamente o risco de deepfakes de voz.

34. A música gerada por IA é original ou plágio?

É original no sentido de que a combinação exacta de notas, ritmos e timbres nunca existiu antes. No entanto, se pedires “uma música no estilo de Beethoven”, a IA vai gerar algo que soa muito parecido, porque os padrões desse compositor estão fortemente gravados nos pesos. É uma zona cinzenta: não é uma cópia, mas é uma “sombra estatística” do estilo. Em termos legais, a discussão ainda está a decorrer em vários países.

35. A IA consegue gerar sons ambientais realistas (chuva, multidão, trânsito)?

Sim, e isso é mais fácil do que música ou voz. Para sons ambientais, a IA aprende padrões de ruído contínuo e frequências — o som da chuva tem um espectro de frequências específico. Ao gerar, combina essas frequências com variações temporais controladas. O resultado é um som que o ouvido humano interpreta como “real”, mesmo não sendo uma gravação verdadeira.

36. Como é que a IA gera código de programação se nunca “executou” esse código?

Durante o treino, a IA leu biliões de linhas de código em repositórios públicos (como o GitHub). Aprendeu padrões: depois de “def” vem um espaço e o nome da função; depois de “if” vem uma condição. Quando pedes “uma função que ordena uma lista”, ela gera tokens que, estatisticamente, se parecem com esse tipo de código. Não sabe se o código funciona — apenas imita a estrutura.

37. O código gerado pela IA tem sempre erros (bugs)?

Muitas vezes sim, especialmente para lógicas complexas. A IA gera o que é provável, não o que é correcto. Pode usar uma função que não existe nessa versão da linguagem, ou esquecer-se de importar uma biblioteca. Por isso, programadores usam a IA para gerar esqueletos de código, mas testam, corrigem e optimizam manualmente.

38. A IA pode substituir programadores no futuro?

Não completamente, mas vai automatizar tarefas repetitivas — gerar código boilerplate, escrever testes, documentar funções. A IA não tem capacidade de raciocínio abstracto para entender requisitos de negócio, arquitectura de sistemas a longo prazo ou para depurar problemas complexos que exigem intuição. O papel do programador vai evoluir para “orquestrador” — em vez de escrever cada linha, vai rever, corrigir e integrar o que a IA gera.

39. A IA entende lógica e algoritmos, ou só decora padrões?

Só decora padrões. Para a IA, o código `“for i in range(10): print(i)”` é uma sequência de tokens como qualquer outra. Não “sabe” que isso vai imprimir números de 0 a 9; apenas viu muitas vezes essa sequência associada a pedidos de “imprimir números”. Se pedires um algoritmo inédito que não exista nos dados de treino, provavelmente vai gerar algo sintacticamente correcto mas semanticamente errado.

40. A IA consegue explicar o código que gera?

Sim, e essa é uma das suas utilidades mais práticas. Se pedires “gera esta função e explica cada linha”, ela vai gerar o código e depois, em linguagem natural, descrever o que cada parte faz. A explicação é baseada nos padrões de documentação que leu — não é compreensão genuína, mas muitas vezes é suficientemente boa para ajudar programadores iniciantes a aprender.

41. A IA gera código seguro? Pode introduzir vulnerabilidades?

Sim, e isso é um risco real. A IA gera código com base no que é comum nos dados de treino, não no que é seguro. Pode replicar vulnerabilidades clássicas — injeção de SQL, buffer overflow — porque esses padrões aparecem frequentemente em repositórios públicos. Cabe ao programador humano inspeccionar o código gerado e aplicar boas práticas de segurança.

42. Porque é que os modelos de IA para imagens se chamam “Difusão”, os de texto “Transformers” e os de código às vezes “Codex”?

Os nomes vêm da arquitectura matemática que usam:

- **Difusão** (imagens/vídeo): porque o processo imita a difusão física — a propagação de ruído até à saturação, e depois o caminho inverso.
- **Transformer** (texto): porque usa um mecanismo de Atenção que “transforma” a sequência de entrada noutra sequência, pesando a importância relativa de cada token em relação aos outros.
- **Codex** (código): era um nome comercial da OpenAI, mas a arquitectura era a mesma do GPT — apenas treinada especificamente com repositórios de código. O nome evoca “códice”, um livro manuscrito antigo.

43. A IA guarda as minhas conversas e as imagens que gero?

Depende do serviço. A maioria das plataformas armazena interações temporariamente para melhorar o modelo, mas afirma não as partilhar publicamente. É crucial ler os termos de serviço: algumas empresas usam os teus dados para treinar futuras versões; outras garantem que os dados são anonimizados e eliminados. Se tiveres dúvidas, não partilhes informações sensíveis — moradas, documentos de identificação, segredos empresariais — em IAs públicas.

44. A IA pode ser usada para criar deepfakes (voz e imagem falsas) de mim?

Sim, e esse é um dos maiores riscos éticos actuais. Com apenas alguns segundos de áudio, uma IA consegue clonar a tua voz; com algumas fotografias, consegue gerar vídeos de ti a dizer coisas que nunca disseste. A solução está em duas frentes: legislação (tornar a criação de deepfakes não consentidos um crime) e tecnologia (marcas de água digitais e detectores de IA que identifiquem conteúdo sintético).

45. Quem é o dono dos direitos de autor de uma imagem, música ou texto gerado por IA?

Esta é uma zona jurídica cinzenta sem consenso global. Em muitos países, obras geradas exclusivamente por IA não têm direitos de autor, porque a lei exige “autoria humana”. Se apenas escreveste o prompt, a obra pode ser considerada “sem autor” e de domínio público. Se fizeste várias iterações, editaste o resultado manualmente ou combinaste a IA com o teu próprio trabalho, podes reivindicar autoria parcial. Consulta um advogado antes de

comercializar conteúdo 100% gerado por IA.

46. A IA vai roubar empregos? Quais os mais ameaçados?

Não vai “roubar” profissões inteiras, mas vai automatizar tarefas específicas dentro de muitas delas. As mais expostas são as que envolvem tarefas repetitivas e previsíveis: tradução básica, redação de textos padrão, design gráfico simples, código boilerplate, atendimento ao cliente de primeiro nível, análise de dados rotineira. As mais seguras são as que exigem empatia, julgamento em contextos ambíguos, criatividade original, negociação e tomada de decisão ética. O futuro não é “IA vs. Humanos”, mas “Humanos com IA vs. Humanos sem IA”.

47. Como é que a IA pode perpetuar preconceitos (racismo, sexismo, etc.)?

Porque aprende com dados humanos — e os dados humanos estão cheios de preconceitos históricos. Se o modelo leu milhões de textos onde “enfermagem” está associada a “mulher” e “engenharia” a “homem”, vai reproduzir esse viés. As empresas tentam corrigir isto com curadoria de dados e técnicas de desvio, mas a solução é imperfeita. A responsabilidade final é do utilizador humano, que deve questionar e não aceitar cegamente o que a IA produz.

48. A IA pode ser usada para vigilância em massa e controlo social?

Sim, e já está a ser usada em alguns países. Sistemas de reconhecimento facial identificam pessoas em multidões em tempo real. Combinados com dados de redes sociais, criam perfis comportamentais detalhados. A resposta a este risco passa por legislação rigorosa — como a Lei da IA da União Europeia, que proíbe vigilância em massa em espaços públicos — e por tecnologias de privacidade diferencial.

49. O que é a “privacidade diferencial” e como protege os meus dados?

É uma garantia matemática que permite a um sistema aprender padrões gerais a partir de dados pessoais sem revelar nada sobre nenhuma pessoa em particular. A ideia central é esta: o resultado que o sistema produz, seja um modelo treinado, seja uma estatística publicada, deve ser praticamente indistinguível quer determinada pessoa esteja nos dados ou não.

Para o conseguir, o sistema adiciona ruído aleatório controlado aos cálculos durante o treino, de forma que o impacto de qualquer pessoa individual nos pesos finais do modelo fique matematicamente "enterrado" nesse ruído. Imagina que és uma das cem mil pessoas cujos dados foram usados para treinar um modelo médico. Com privacidade diferencial,

mesmo que alguém compare o modelo treinado com os teus dados com um modelo treinado sem os teus dados, não consegue determinar se estiveste incluído ou não — a diferença entre os dois é indistinguível do ruído. A tua presença nos dados é, matematicamente, invisível.

Empresas como a Apple e a Google já usam esta técnica para melhorar os seus modelos com dados de milhões de utilizadores, sem que seja possível extrair informação sobre nenhum utilizador específico. O custo é uma ligeira redução na precisão do modelo — é a troca entre utilidade e privacidade.

50. A IA pode tomar decisões éticas (ex: carros autónomos em acidentes iminentes)?

Não, no sentido genuíno. A IA não tem um “código moral” — apenas optimiza para um objectivo definido por humanos. Num acidente de carro autónomo, a IA vai escolher o que o programador definiu como prioridade (proteger o condutor, proteger peões, seguir a lei). Essa escolha é sempre arbitrária e programada, nunca um julgamento moral genuíno.

51. Como posso saber se um texto, imagem ou vídeo foi gerado por IA?

Está a tornar-se cada vez mais difícil, mas há pistas:

- **Textos:** frases muito “perfeitas”, sem erros, com estrutura repetitiva e ausência de opiniões pessoais profundas.
- **Imagens:** mãos com dedos a mais ou a menos, fundos desfocados de forma estranha, texto ilegível em placas ou livros.
- **Vídeos:** movimentos pouco naturais, inconsistências de iluminação entre frames.
- Alguns modelos já incluem marcas de água invisíveis nos pixels que permitem detectar a origem.

52. Como posso usar a IA de forma ética e responsável?

Cinco princípios práticos:

1. **Sê transparente:** Se publicares conteúdo gerado por IA, informa o público.
2. **Não enganes:** Não uses IA para criar provas falsas, testemunhos ou documentos oficiais.
3. **Protege a privacidade:** Nunca insiras dados pessoais de terceiros em IAs públicas.

4. Questiona os resultados: A IA erra e tem vieses. Valida informações críticas com fontes independentes.

5. Pensa no impacto: Antes de automatizar com IA, pergunta-te: quem ganha? Quem perde? Há dignidade humana em risco?

53. O que está a ser feito a nível global para regular a IA?

Vários países e blocos estão a criar legislação específica:

- **União Europeia:** Aprovou a Lei da IA (AI Act), a primeira do mundo, que classifica as IAs por níveis de risco. IAs de vigilância em massa e manipulação comportamental são proibidas.
- **EUA:** Têm directrizes da Casa Branca e iniciativas voluntárias das empresas, mas ainda sem lei federal abrangente.
- **China:** Exige que as IAs sejam “alinhas com os valores socialistas” e que as empresas registem os seus algoritmos.
- **Portugal e Brasil:** Estão a debater projectos de lei que criminalizam deepfakes e exigem transparência em decisões automatizadas.

A tendência é que todas as IAs venham a ter rótulos de transparência obrigatórios, como os ingredientes nos alimentos.

54. A IA pode “escapar” ao controlo humano e tornar-se perigosa?

É um cenário improvável no curto prazo, mas não impossível a longo prazo. A IA actual é “estreita” — faz uma tarefa bem, mas não tem vontade própria nem objectivos autónomos. Para se tornar perigosa, precisaria de capacidade de auto-melhoria, acesso a sistemas físicos e objectivos mal alinhados com os valores humanos. Investigadores como Geoffrey Hinton e Yoshua Bengio alertam para este risco e defendem investigação em segurança e alinhamento de IA.

55. O que é a “explicabilidade” em IA e porque é importante?

Explicabilidade é a capacidade de um modelo explicar como chegou a uma decisão. Modelos complexos são “caixas negras” — ninguém sabe exactamente porque escolheram A em vez de B. Num diagnóstico médico, se a IA errar, precisamos de saber porquê. A União Europeia já exige que IAs de “alto risco” — concessão de crédito, selecção de currículos — sejam explicáveis. Técnicas como SHAP ou LIME tentam criar explicações aproximadas, mas são ainda limitadas.

56. A IA pode ser viciante?

Sim, especialmente as IAs conversacionais e os geradores de conteúdo “infinito”. São desenhadas para maximizar o engagement, activando o sistema de recompensa cerebral. O risco é o uso excessivo, a substituição de interacções humanas reais e a perda de capacidade crítica. A regra de ouro: usa a IA como ferramenta, não como substituta da vida real.

57. Qual é o maior desafio ético da IA nos próximos 10 anos?

Não será a “superinteligência”, mas a desigualdade. Quem tiver acesso às melhores IAs — e aos dados para as treinar — terá um poder económico e político descomunal. O desafio é democratizar o acesso à IA e garantir que os benefícios são partilhados. É um problema político e económico, não apenas tecnológico.

58. A IA já está presente no meu telemóvel sem eu saber?

Sim. Quando desbloqueias o telefone com reconhecimento facial, estás a usar uma rede neural. Quando o teclado prevê a próxima palavra, é um pequeno modelo de linguagem local. As sugestões de legendas nas fotos, os filtros de realidade aumentada, a optimização da bateria — tudo usa IA. A maioria destes modelos corre localmente no dispositivo, sem enviar dados para a nuvem, para proteger a tua privacidade.

59. Como é que a IA escolhe os anúncios que me aparecem na internet?

Através de aprendizagem por reforço combinada com sistemas de recomendação. A IA regista silenciosamente o que clicas, quanto tempo ficas a ver cada anúncio e quais as compras que fazes. Constrói um “perfil” com centenas de características e calcula, em milissegundos, qual o anúncio com maior probabilidade de te fazer clicar, com base em milhões de utilizadores com perfis semelhantes. Não é magia publicitária — é estatística aplicada ao teu comportamento.

60. Os assistentes virtuais (Siri, Alexa, Google Assistant) percebem mesmo o que digo?

Não, não percebem. Usam reconhecimento automático de fala que converte as tuas palavras em texto, e depois um modelo leve que tenta extrair uma “intenção”. São modelos pequenos, especializados e limitados — não têm capacidade de conversa aberta. Apenas os assistentes mais recentes (como o novo Siri com Apple Intelligence) começam a usar LLMs para respostas mais flexíveis.

61. Os filtros de spam no Gmail usam IA? Que tipo?

Sim, usam modelos de classificação supervisionada que analisam o remetente, as palavras do assunto, a estrutura do email e padrões de erro ortográfico. Treinados com milhões de emails marcados manualmente como “spam” ou “não-spam”, o Gmail bloqueia actualmente mais de 99,9% do spam. É um dos exemplos mais antigos e bem-sucedidos de IA no quotidiano.

62. Como é que o YouTube ou a Netflix sabem o que me sugerir a seguir?

Usam filtragem colaborativa: a IA analisa o teu histórico, compara com o de milhões de outros utilizadores e deduz: “utilizadores que viram X também viram Y — e tu viste X.” Quanto mais usas a plataforma, mais o modelo se ajusta ao teu gosto. Mas cuidado: estes algoritmos são optimizados para a retenção, não para a tua felicidade. Tendem a sugerir conteúdo mais extremo ou polarizador porque gera mais cliques.

63. Os carros modernos usam IA? O que fazem?

Sim, cada vez mais. Usam IA para assistência à condução: alerta de mudança de faixa, travagem automática de emergência, controlo de cruzeiro adaptativo. Carros mais avançados usam redes neurais para visão computacional, interpretando imagens de câmaras para identificar peões, sinais e outros veículos. Mesmo os melhores sistemas ainda cometem erros — a IA automóvel está em constante evolução, mas não é infalível.

64. Como é que a IA traduz textos e conversas em tempo real?

A tradução em tempo real combina três modelos a trabalhar em sequência, com atrasos inferiores a um segundo:

Reconhecimento de fala (ASR): O áudio é convertido em texto. Modelos modernos fazem isto com mais de 95% de precisão em várias línguas, em cerca de 200-300 milissegundos.

Tradução automática neural (NMT): O texto é codificado num vector numérico abstracto (uma representação semântica) e depois decodificado para a língua de destino. Ao contrário da tradução palavra a palavra, o modelo olha para a frase inteira, captando o contexto. “Je voudrais réserver une table” não sai como “Eu quereria reservar uma mesa” — sai como “Gostaria de reservar uma mesa.”

Síntese de voz (TTS): O texto traduzido é convertido em áudio com voz natural. Os

melhores modelos actuais produzem vozes praticamente indistinguíveis das humanas.

Em videochamadas, pode ainda haver um quarto passo: sincronia labial, onde os movimentos da boca são ajustados para parecerem sincronizados com as palavras traduzidas. Toda esta cadeia acontece em menos de 800 milissegundos. O que a IA não faz: não compreende o que traduz, perde subtilidades culturais e ironias, e pode cometer erros de pronome em traduções longas.

65. A IA que recomenda notícias nas redes sociais pode influenciar a minha opinião política?

Sim, e já foi comprovado em vários estudos. Os algoritmos são optimizados para maximizar o engagement — e o conteúdo que gera mais raiva ou indignação tem mais cliques. Ao mostrar-te mais do que já te faz reagir, a IA cria câmaras de eco onde vês apenas opiniões semelhantes às tuas. A recomendação mais sensata: diversifica as tuas fontes deliberadamente e trata as notícias sugeridas pela IA com cepticismo saudável.

66. Os jogos de vídeo usam IA? O que é a “IA” dos inimigos nos jogos?

Sim, mas a “IA” dos jogos tradicionais não tem nada a ver com a IA generativa moderna. Nos jogos clássicos, é um conjunto de árvores de decisão e regras programadas manualmente — “se o jogador estiver a menos de 10 metros, o inimigo dispara.” Alguns jogos mais recentes utilizam aprendizagem por reforço para criar personagens que se adaptam ao estilo de jogo do utilizador — uma técnica demonstrada com grande sucesso em jogos de tabuleiro como o Go (AlphaGo) e em jogos de estratégia em tempo real como o StarCraft II (AlphaStar). Nos jogos comerciais de mundo aberto, a transição para IA generativa ainda está a começar.

67. Os bancos e seguradoras usam IA para me avaliar? Como?

Sim, através de scoring algorítmico. Os bancos analisam o teu histórico de crédito, padrões de gastos, profissão e localização para decidir se aprovam um empréstimo e a que taxa. As seguradoras fazem o mesmo para calcular prémios. O problema é que estes modelos podem ser discriminatórios. Na Europa, a Lei da IA obriga os bancos a explicar decisões automatizadas — se um empréstimo for recusado, tens o direito de pedir revisão humana.

68. Como é que a IA faz diagnósticos médicos? Pode substituir um médico?

A IA é usada para analisar imagens médicas — raios-X, ressonâncias, retinografias — com precisão que, em alguns casos, supera a de especialistas humanos. Já detecta cancro da

mama em mamografias com maior sensibilidade do que radiologistas. Mas não substitui um médico: é uma ferramenta de apoio. O médico interpreta o diagnóstico da IA no contexto do paciente, considera factores psicológicos e sociais, e toma a decisão final.

69. Os agricultores usam IA? Como?

Sim, na chamada agricultura de precisão. Drones sobrevoam plantações com câmaras multiespectrais e a IA detecta pragas, doenças e défices de água ou nutrientes, zona a zona. Sistemas de irrigação inteligentes ligam-se a modelos de previsão meteorológica para regar apenas o necessário. Há até robôs com IA que identificam e arrancam ervas daninhas, pulverizando herbicida apenas na planta-alvo.

70. Como é que a IA que gera legendas automáticas para surdos funciona?

É uma combinação de reconhecimento de fala e segmentação semântica. A IA converte o áudio em texto, identifica pausas, mudanças de orador e tom emocional, e adiciona pontuação e identificadores como “[plateia aplaude]”. As legendas em directo ainda cometem erros em sotaques, nomes próprios e palavras técnicas — a precisão perfeita é ainda um objectivo em curso.

71. A IA ajuda a combater as alterações climáticas? Como?

Sim, de várias formas: optimiza o consumo de energia em edifícios e redes eléctricas; prevê eventos climáticos extremos com maior antecipação; acelera a investigação em química para novos materiais de captura de carbono e baterias mais eficientes; monitoriza desflorestação em tempo real através de satélites. No entanto, o treino de grandes modelos de IA também consome muita energia — algumas estimativas apontam que o treino de um único modelo grande emite tanto CO₂ quanto cinco carros ao longo da sua vida útil. O equilíbrio entre benefícios e custos ambientais é uma área activa de debate.

72. A IA que gere os feeds das redes sociais sabe quando estou triste ou ansioso?

Em parte. Os algoritmos analisam o que publicas, as palavras que usas, as horas em que estás activo e a rapidez com que digitas. Estes dados são usados para inferir o teu estado emocional, de forma imperfeita. O problema é que a IA pode então mostrar-te conteúdo que reforça esse estado — se estás triste, pode sugerir vídeos melancólicos porque pessoas tristes clicam neles, criando um ciclo vicioso.

73. Como é que a IA sabe que sou eu quando faço reconhecimento facial no telemóvel?

Uma câmara infravermelha projecta pontos invisíveis no teu rosto, criando um mapa 3D. Esse mapa é convertido num vector numérico de muitas dimensões — uma “impressão digital facial” — guardado localmente no telemóvel (nunca sai do dispositivo). Quando tentas desbloquear, a IA gera um novo vector e compara com o guardado. Se a distância entre eles for inferior a um limiar definido, és tu. O sistema tolera barbas, óculos e gorros, mas tem dificuldade com gémeos idênticos.

74. O que são os “chatbots” de atendimento ao cliente e porque são muitas vezes frustrantes?

A maioria usa modelos pequenos e rígidos, com árvores de decisão pré-programadas: “se o utilizador disser X, responda Y.” Só conseguem lidar com perguntas muito específicas. Quando a pergunta sai do script, repetem respostas genéricas ou transferem para um humano. A frustração acontece porque as empresas ainda preferem modelos baratos e limitados que resolvem 80% dos casos básicos, em detrimento de LLMs avançados que custariam mais mas seriam genuinamente úteis nos restantes 20%.

75. A IA pode escrever o meu currículo ou biografia para redes sociais? Devo confiar?

Pode, e é uma das aplicações mais comuns. A IA gera rascunhos com frases bem estruturadas e vocabulário profissional. O problema é que pode não soar a ti — usa linguagem genérica e frases feitas. A recomendação: usa a IA para um primeiro rascunho, depois edita com a tua voz, humor e experiências reais. Para um currículo técnico, a IA funciona bem; para uma biografia onde a autenticidade é chave, o teu toque humano é indispensável.

76. A IA pode prever o que vou comprar antes de eu comprar?

Sim. Quando navegas num site de comércio electrónico, a IA regista cada clique, tempo de permanência e itens que adicionas ao carrinho e abandonas. Com esses dados, calcula a probabilidade de compra para cada produto e ajusta o site em tempo real — mostrando ofertas personalizadas no momento em que a probabilidade de compra é máxima. É por isso que, muitas vezes, vês anúncios de algo que pesquisaste mas não compraste.

77. A IA pode ser usada para combater notícias falsas?

Estão a surgir ferramentas de fact-checking automático que comparam o texto de uma notícia com bases de dados de fontes verificadas, analisando contradições e linguagem manipuladora. No entanto, a IA não tem “bom senso” para distinguir ironia de desinformação, nem contexto histórico profundo. A abordagem mais promissora é a IA colaborativa: a máquina sinaliza suspeitas, o humano decide. A melhor arma contra as fake news continua a ser o teu pensamento crítico.

78. Os assistentes de IA vão eventualmente substituir os professores?

Não, e a maioria dos especialistas em educação defende que não devem. A IA é excelente para tarefas mecânicas — explicar conceitos, corrigir gramática, sugerir exercícios — mas a educação é também sobre relação humana, inspiração, empatia e adaptação ao contexto social e emocional. O futuro é o **professor aumentado**: o docente usa a IA para personalizar percursos e identificar dificuldades, libertando tempo para o que realmente importa — mentorar e motivar.

79. A IA sabe quando estou a mentir ou a esconder algo?

Actualmente, não de forma fiável. Algumas IAs analisam microexpressões faciais e tom de voz para inferir emoções, mas com baixa precisão. Em testes controlados, modelos de detecção de mentira têm uma taxa de acerto de apenas 55% — pouco melhor que uma moeda ao ar. A mentira humana é demasiado complexa e dependente de contexto para ser capturada por padrões estatísticos.

80. A IA pode ser usada para criar arte que emocione as pessoas?

Sim, e isso já acontece. Imagens geradas por IA podem ser belas, assustadoras ou melancólicas — e as pessoas sentem emoções genuínas ao vê-las. No entanto, a intencionalidade é a diferença fundamental: a IA não sentiu nada ao criá-las; apenas optimizou padrões. Quando uma obra de IA toca o teu coração, a emoção está em ti e na tua interpretação, não na máquina. A IA pode ser o pincel; a alma da obra é sempre humana.

A IA está em todo o lado — muitas vezes invisível, já a moldar o que vês, lêes, compras, estudas e sentes. Esta ubiquidade torna-a uma ferramenta extraordinária, mas também um veículo de riscos subtis: vieses, vigilância, manipulação e dependência. Usá-la com sabedoria não é uma questão técnica, mas uma questão de literacia digital — saber quando confiar, quando questionar, quando desligar, e, acima de tudo, quando a máquina serve o humano, e não o contrário.

ÍNDICE

A RECEITA (O QUE É A IA E COMO SE FAZ)	4
NÃO É MAGIA, É MATEMÁTICA COM FOME	5
A RECEITA E O COZINHEIRO	12
AS FERRAMENTAS DO OFÍCIO	19
A DESPENSA E O ARMAZÉM (A INFORMAÇÃO)	26
DE ONDE VEM A COMIDA	27
ONDE MORA A INFORMAÇÃO	34
COMO A COMIDA É CORTADA EM PEDACINHOS	40
O CÉREBRO DE POLVO (COMO A IA “PENSA”)	46
A MEMÓRIA QUE NÃO É UMA MEMÓRIA	47
COMO ELA LÊ E RESPONDE A UMA PERGUNTA	54
QUANDO O POLVO VAI À BIBLIOTECA	61
O DIA A DIA (E OS LIMITES)	68
PORQUE É QUE ELA ACERTA E ERRA	69
O QUE ELA NÃO É (SETE MITOS PARA DESMONTAR)	75
GLOSSÁRIO E APÊNDICES	84
GLOSSÁRIO DO TOTÓ	85
POR DENTRO DO POLVO, A SÉRIO	89
UM ÚLTIMO NÚMERO, PARA FECHAR COM UM SORRISO	102
COMO O POLVO APRENDE, PASSO A PASSO	103
UM ENSAIO SEM PLANO, MAS COM ESTRUTURA	108
PINTAR A PARTIR DO RUÍDO	113
QUANDO O POLVO GANHA MÃOS	118
AS PERGUNTAS DO TOTÓ	123