

# Le lexique des IA génératives

**Comprendre le vocabulaire, décoder une fiche de modèle,  
choisir un fichier**

## Table des matières

|                                                               |    |
|---------------------------------------------------------------|----|
| Mode d'emploi . . . . .                                       | 2  |
| 1. Décoder une fiche de modèle . . . . .                      | 2  |
| 2. Les mots de base . . . . .                                 | 4  |
| 3. Lire le nom d'un modèle . . . . .                          | 5  |
| 4. Qui fait quoi : les principaux acteurs . . . . .           | 7  |
| 5. Les formats de fichiers . . . . .                          | 10 |
| 6. La quantification . . . . .                                | 11 |
| 7. Faire tourner un modèle chez soi . . . . .                 | 14 |
| 8. Les réglages de génération . . . . .                       | 16 |
| 9. Entraînement et spécialisation . . . . .                   | 18 |
| 10. Architectures et familles . . . . .                       | 20 |
| 11. Mesurer et comparer . . . . .                             | 21 |
| 12. Licences et libertés . . . . .                            | 22 |
| 13. Autour du modèle . . . . .                                | 23 |
| 14. Faux amis et pièges courants . . . . .                    | 25 |
| 15. Par où commencer, concrètement . . . . .                  | 26 |
| 16. Passer à l'action : donner des outils au modèle . . . . . | 26 |
| Index alphabétique . . . . .                                  | 29 |

# Mode d'emploi

Le vocabulaire des modèles de langage a une particularité désagréable : il est presque entièrement anglais, largement composé de sigles, et **personne ne le définit jamais**. Les pages de téléchargement supposent que vous savez déjà. Vous arrivez sur une fiche de modèle et vous lisez :

Qwen3-30B-A3B-Instruct-2507-GGUF · 18,6 GB · Q4\_K\_M · Apache 2.0

Six informations, et pas une seule phrase d'explication.

Ce document sert à lever ce barrage. Il est organisé **par thème** et non par ordre alphabétique, parce qu'on apprend mieux les mots par familles ; un **index alphabétique** en fin de volume permet la consultation ponctuelle. Chaque entrée donne le terme anglais tel que vous le rencontrerez, sa traduction française lorsqu'elle existe et qu'elle est réellement employée, et une explication courte.

Trois conventions :

- Les termes en **gras** sont définis dans le document et figurent à l'index.
- La mention « *ce que vous verrez* » indique la forme concrète sous laquelle le terme apparaît sur une page de téléchargement.
- Les ordres de grandeur sont donnés à titre indicatif : ils dépendent de l'architecture et évoluent vite.

Aucune connaissance préalable n'est nécessaire. Si vous cherchez à comprendre le *fonctionnement* de ces modèles plutôt que leur vocabulaire, ce lexique est le compagnon d'un autre document, « Comprendre les IA génératives », auquel il renvoie parfois.

**Par où commencer.** Si vous n'avez que dix minutes, lisez le chapitre 1, qui décortique une fiche réelle de bout en bout, puis le chapitre 6 sur la quantification — c'est ce qui bloque le plus souvent au moment de choisir un fichier.

---

## 1. Décoder une fiche de modèle

Commençons par le cas concret, dont tout le reste découle. Vous êtes sur **Hugging Face**, la grande plateforme d'hébergement de modèles, et vous tombez sur ceci :

bartowski / Qwen3-30B-A3B-Instruct-2507-GGUF

### Le chemin

| Morceau   | Signification                                                                                                |
|-----------|--------------------------------------------------------------------------------------------------------------|
| bartowski | <b>L'éditeur du dépôt.</b> Ici un particulier réputé pour ses conversions ; ce n'est pas l'auteur du modèle. |
| Qwen3     | <b>La famille.</b> Qwen est la série de modèles d'Alibaba ; le 3 est la génération.                          |
| 30B       | <b>30 milliards de paramètres</b> au total (chapitre 2).                                                     |
| A3B       | <b>3 milliards de paramètres actifs</b> par token : c'est un modèle à mélange d'experts (chapitre 10).       |
| Instruct  | <b>Variante conversationnelle</b> , dressée pour suivre des instructions — par opposition à Base.            |
| 2507      | <b>Date de révision</b> : juillet 2025.<br>Beaucoup d'éditeurs datent ainsi leurs mises à jour.              |
| GGUF      | <b>Le format de fichier</b> (chapitre 5), celui qu'utilisent les logiciels grand public.                     |

Autrement dit : *un modèle Alibaba de troisième génération, 30 milliards de paramètres dont 3 travaillent à chaque token, réglé pour la conversation, révisé en juillet 2025, converti au format GGUF par un tiers nommé bartowski.*

## La liste des fichiers

En dessous, l'onglet « Files » propose une dizaine de variantes du même modèle :

| Fichier                                 | Taille  |
|-----------------------------------------|---------|
| Qwen3-30B-A3B-Instruct-2507-Q8_0.gguf   | 32,5 GB |
| Qwen3-30B-A3B-Instruct-2507-Q6_K.gguf   | 25,1 GB |
| Qwen3-30B-A3B-Instruct-2507-Q5_K_M.gguf | 21,7 GB |
| Qwen3-30B-A3B-Instruct-2507-Q4_K_M.gguf | 18,6 GB |
| Qwen3-30B-A3B-Instruct-2507-IQ3_XS.gguf | 12,8 GB |
| Qwen3-30B-A3B-Instruct-2507-Q2_K.gguf   | 11,3 GB |

**C'est le même modèle, compressé plus ou moins fort.** Le nombre après le Q est le nombre de bits conservés par paramètre : plus il est bas, plus le fichier est petit et plus la qualité se dégrade. Le chapitre 6 détaille ce codage.

Vous n'en téléchargez **qu'un seul** : le plus gros qui tienne confortablement dans la mémoire de votre carte graphique.

## Le reste de la page

- **Model card** : la documentation rédigée par l'auteur. Qualité très variable, du rapport technique de cinquante pages au fichier vide.
  - **License** : ce que vous avez le droit d'en faire (chapitre 12). À lire avant tout usage professionnel.
  - **Downloads last month** : un indicateur de popularité, pas de qualité.
  - **Tags** : text-generation, conversational, fr, transformers... Les langues annoncées sont déclaratives et souvent optimistes.
- 

## 2. Les mots de base

**LLM** — *Large Language Model*, grand modèle de langage. Un système entraîné à prédire la suite d'un texte, dont toutes les capacités apparentes découlent de cette tâche unique. « Grand » renvoie au nombre de paramètres, aujourd'hui des milliards.

**Modèle** — Dans l'usage courant, le fichier contenant les **paramètres**, et par extension le système qui l'exécute. Un modèle n'est pas un programme au sens habituel : c'est un gros tableau de nombres qu'un programme séparé fait tourner.

**Paramètre** (*parameter*), aussi **poids** (*weight*) — Un des nombres réglables dont l'ajustement constitue l'apprentissage. C'est l'unité de mesure de la taille d'un modèle. *Ce que vous verrez* : « 7B », « 70B », « 405B » — B pour billion, milliard.

**Token** — Le morceau de texte élémentaire que manipule le modèle : un mot courant, un fragment de mot rare, un signe de ponctuation. En français, comptez grossièrement **1,3 à 1,5 token par mot** — davantage qu'en anglais, car les vocabulaires ont été construits majoritairement sur des corpus anglophones. C'est l'unité de facturation des services en ligne et l'unité de mesure du contexte.

**Tokenizer** (tokéniseur) — Le programme qui découpe le texte en tokens et les reconvertit ensuite. Chaque famille de modèles a le sien ; c'est pourquoi un même texte ne coûte pas le même nombre de tokens d'un modèle à l'autre. *Ce que vous verrez* : les fichiers `tokenizer.json` et `tokenizer_config.json` dans un dépôt.

**Vocabulaire** (*vocab*) — L'ensemble des tokens qu'un modèle sait manipuler, typiquement de 32 000 à 260 000 entrées. Un vocabulaire large traite mieux les

langues autres que l'anglais mais alourdit le modèle.

**Contexte** (*context*), **fenêtre de contexte** (*context window*) — Tout ce que le modèle a sous les yeux à un instant donné : consignes, historique de la conversation, documents joints. Sa taille se mesure en tokens. *Ce que vous verrez* : « 8K », « 128K », « 1M ». Au-delà, les tokens les plus anciens sont tout simplement oubliés.

**Prompt** — Ce que l'on soumet au modèle. Le mot n'a pas d'équivalent français réellement employé ; « requête » ou « invite » restent rares.

**System prompt** (invite système) — Les consignes générales placées en tête du contexte, qui définissent le rôle et le comportement attendus. L'utilisateur ne les voit généralement pas.

**Inférence** (*inference*) — L'exécution du modèle pour produire une réponse, par opposition à l'entraînement. Pendant l'inférence, **les paramètres ne changent pas** : le modèle n'apprend rien de vos échanges.

**Génération** — La production du texte, token par token. Le modèle produit un seul token, l'ajoute au texte, et recommence la traversée complète du réseau pour le suivant.

**Embedding** (plongement, vecteur) — La représentation d'un token ou d'un texte sous forme d'une liste de nombres, où la proximité numérique traduit la proximité de sens. Terme important : il désigne aussi une catégorie entière de modèles (chapitre 10).

**Hallucination** — Une affirmation fausse énoncée avec assurance. Ce n'est pas un bogue mais une conséquence du mécanisme : le système produit du **plausible**, et une référence inventée a exactement la forme d'une vraie.

**Poids ouverts** (*open weights*) — Se dit d'un modèle dont le fichier de paramètres est publiquement téléchargeable. **Ce n'est pas la même chose qu'open source** : les données d'entraînement et le code complet sont rarement publiés, et la licence peut être restrictive (chapitre 12).

---

### 3. Lire le nom d'un modèle

Les noms sont un télégramme dense. Voici les morceaux que vous rencontrerez.

#### La taille

**7B, 13B, 70B, 405B** — Le nombre de paramètres, en milliards. C'est le premier déterminant de la qualité, et surtout de ce que votre machine pourra faire tourner.

**A3B, A17B, A37B** — *Active*. Sur un modèle à **mélange d'experts**, le nombre de paramètres réellement mobilisés pour chaque token. 30B-A3B se lit : *30 milliards au total, 3 milliards actifs*. Conséquence pratique très concrète — **il faut la mémoire des 30 milliards, mais la vitesse est celle d'un modèle de 3 milliards**.

**Mini, Small, Medium, Large, Nano, Flash, Pro** — Des étiquettes commerciales sans définition stable. Elles ne sont comparables qu'à l'intérieur d'une même famille.

## La destination

**Base** (*base, pretrained*) — Le modèle brut, issu du seul pré-entraînement. Excellent compléteur de texte, mais **il ne tient pas une conversation** : posez-lui une question, il peut enchaîner avec dix autres questions. Réservé à ceux qui veulent le spécialiser eux-mêmes.

**Instruct, IT** (*instruction-tuned*), **Chat** — La variante dressée pour dialoguer et suivre des consignes. **C'est celle qu'il vous faut** dans la quasi-totalité des cas.

**Thinking, Reasoning, R** — Entraînée à produire un long brouillon de raisonnement avant de répondre. Plus lente et plus bavarde, nettement meilleure en mathématiques, en logique et en programmation.

**Coder, Code** — Spécialisée dans la programmation.

**VL** (*vision-language*), **Vision, Omni** — Accepte des images en entrée, parfois du son. Omni désigne généralement la prise en charge de plusieurs modalités en entrée comme en sortie.

**Embed, Reranker** — Ce ne sont pas des modèles de conversation mais des outils de recherche documentaire (chapitres 10 et 13). Les télécharger en croyant obtenir un assistant est une erreur de débutant fréquente.

**Guard, Moderation** — Des modèles de filtrage, destinés à classer des contenus comme acceptables ou non.

## Les mentions de version et d'origine

**v0.1, v0.3, 2507, 20250714** — Numéros de révision ou dates. Le format AAMM (2507 = juillet 2025) est courant.

**Distill** — Modèle plus petit entraîné à imiter un plus gros (chapitre 9). *Ce que vous verrez : DeepSeek-R1-Distill-Qwen-7B, c'est-à-dire un Qwen 7B ayant appris à raisonner en imitant R1.*

**Uncensored, Abliterated, Dolphin** — Versions modifiées pour supprimer les refus. Qualité et fiabilité souvent dégradées, et la responsabilité juridique de l'usage vous revient entièrement.

**Merge, Frankenmerge** — Fusion de plusieurs modèles par moyenne de leurs paramètres. Résultats imprévisibles, parfois bons, rarement documentés.

**GGUF, GPTQ, AWQ, MLX, EXL2** — Non pas des variantes du modèle mais des **formats de fichier** (chapitre 5).

**Le piège du dépôt tiers.** TheBloke/..., bartowski/..., unsloth/..., mradermacher/... sont des conversions publiées par des tiers, pas par l’auteur du modèle. Elles sont en général excellentes et souvent disponibles avant les versions officielles, mais vous téléchargez un fichier construit par un inconnu : privilégiez les éditeurs reconnus, et méfiez-vous des dépôts sans historique.

## 4. Qui fait quoi : les principaux acteurs

Vous venez de décoder Qwen3 ou Llama sans savoir qui se cache derrière. Ce chapitre situe les noms que vous croiserez.

**Deux avertissements avant de lire.** D’abord, **c’est le chapitre le plus périssable du document** : les versions sortent au rythme de plusieurs par mois et les positions relatives changent en quelques trimestres. Les *familles* de modèles, elles, sont stables ; ce sont elles qu’il faut retenir, pas les numéros. Ensuite, ce document a été rédigé par un modèle d’Anthropic : les tableaux s’en tiennent donc à des faits vérifiables — qui publie quoi, sous quelle licence, avec quels outils — plutôt qu’à des jugements de valeur. Gardez d’ailleurs à l’esprit, en lisant les communications des laboratoires, que leurs revendications de performance ou de sécurité sont en grande partie déclaratives, **pour tous sans exception**.

### Les modèles fermés

Accessibles par abonnement ou par interface de programmation ; les poids ne sont pas téléchargeables.

| Acteur                                | Modèles                      | Outils                                                                                   |
|---------------------------------------|------------------------------|------------------------------------------------------------------------------------------|
| <b>OpenAI</b><br><i>États-Unis</i>    | GPT                          | ChatGPT (web et mobile), Codex pour la programmation, Sora pour la vidéo, plateforme API |
| <b>Anthropic</b><br><i>États-Unis</i> | Claude — Opus, Sonnet, Haiku | Application Claude, Claude Code, extension pour navigateur, console et API               |

| Acteur                                                                 | Modèles               | Outils                                                                          |
|------------------------------------------------------------------------|-----------------------|---------------------------------------------------------------------------------|
| <b>Google</b><br><b>DeepMind</b><br><i>États-Unis,</i><br><i>R.-U.</i> | Gemini                | Application Gemini, intégration à Workspace, NotebookLM, AI Studio et Vertex AI |
| <b>xAI</b> <i>États-Unis</i>                                           | Grok                  | Application Grok, intégration au réseau X, API                                  |
| <b>Mistral AI</b><br><i>France</i>                                     | Mistral Large, Medium | Le Chat, La Plateforme (API) ; publie aussi des poids ouverts                   |
| <b>Amazon</b><br><i>États-Unis</i>                                     | Nova                  | Bedrock (accès à ses modèles et à ceux de tiers), assistant Amazon Q            |
| <b>Cohere</b><br><i>Canada</i>                                         | Command               | API, plateforme North pour l'entreprise                                         |
| <b>Microsoft</b><br><i>États-Unis</i>                                  | Phi et modèles tiers  | Microsoft 365 Copilot, GitHub Copilot, Azure AI Foundry                         |

## Les modèles à poids ouverts

Ceux que vous téléchargerez réellement. La colonne licence est déterminante : reportez-vous au chapitre 12.

| Éditeur                                   | Famille        | Licence      | Particularité                                                                                                                                                   |
|-------------------------------------------|----------------|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Alibaba</b><br><i>Chine</i>            | Qwen           | Apache 2.0   | <b>La gamme la plus complète</b> , du modèle minuscule au très gros MoE, avec variantes code et vision. Le plus téléchargé et le plus affiné par la communauté. |
| <b>DeepSeek</b><br><i>Chine</i>           | DeepSeek-V, -R | MIT          | Raisonnement et rapport performances/coût. A popularisé l'entraînement par récompense vérifiable.                                                               |
| <b>Moonshot AI</b> <i>Chine</i>           | Kimi           | MIT modifiée | Très gros modèles à experts, orientés agents et outils.                                                                                                         |
| <b>Z.ai</b><br>(ex-Zhipu)<br><i>Chine</i> | GLM            | MIT          | Programmation et tâches longues en autonomie.                                                                                                                   |

| Éditeur                               | Famille                            | Licence                      | Particularité                                                                                                  |
|---------------------------------------|------------------------------------|------------------------------|----------------------------------------------------------------------------------------------------------------|
| <b>MiniMax</b><br><i>Chine</i>        | MiniMax-M                          | Apache 2.0                   | Contextes longs, faible part de paramètres actifs.                                                             |
| <b>Meta</b><br><i>États-Unis</i>      | Llama                              | Licence maison               | A ouvert la voie en 2023 et structuré tout l'écosystème. <b>Sa licence n'est pas libre</b> malgré le discours. |
| <b>Google</b><br><i>États-Unis</i>    | Gemma                              | Conditions propres           | Petits modèles pensés pour l'exécution locale et l'embarqué.                                                   |
| <b>OpenAI</b><br><i>États-Unis</i>    | gpt-oss                            | Apache 2.0                   | Retour d'OpenAI aux poids ouverts, orienté raisonnement et agents.                                             |
| <b>Mistral AI</b><br><i>France</i>    | Mistral,<br>Ministral,<br>Devstral | Apache 2.0<br>(selon modèle) | <b>Le principal éditeur européen.</b> Du modèle embarquable au très gros.                                      |
| <b>Nvidia</b><br><i>États-Unis</i>    | Nemotron                           | Ouverte                      | Souvent accompagnés des données et des recettes d'entraînement. Architectures hybrides.                        |
| <b>Microsoft</b><br><i>États-Unis</i> | Phi                                | MIT                          | Petits modèles très performants pour leur taille, entraînés sur données très filtrées.                         |
| <b>Ai2</b><br><i>États-Unis</i>       | OLMo                               | Pleinement ouverte           | <b>Le seul véritablement open source</b> : poids, données et code publiés. Référence pour la recherche.        |
| <b>TII Émirats</b>                    | Falcon                             | Apache 2.0                   | Un des premiers grands modèles ouverts non américains.                                                         |
| <b>IBM</b><br><i>États-Unis</i>       | Granite                            | Apache 2.0                   | Orientation entreprise, tailles modestes.                                                                      |

## Ce que ces tableaux racontent

**Entraîner un modèle et vendre un outil sont deux métiers distincts.** Certains acteurs font les deux, d'autres se contentent de distribuer : Microsoft vend Copilot en s'appuyant largement sur des modèles tiers, Amazon revend par Bedrock l'accès à des modèles qu'elle n'a pas entraînés. Un outil connu ne signale donc pas un laboratoire de recherche derrière — et à l'inverse, les éditeurs d'outils les plus visibles ne sont pas toujours ceux qui font avancer la technique.

**Les meilleurs modèles ouverts viennent aujourd’hui majoritairement de Chine.** Qwen, DeepSeek, GLM, Kimi et MiniMax occupent le haut des classements ouverts, souvent sous des licences plus permissives que celles de leurs concurrents américains — une inversion complète par rapport à 2023.

**L’écart entre ouvert et fermé s’est considérablement resserré**, sans disparaître. Un bon modèle ouvert de 2026 fait ce que faisaient les meilleurs modèles fermés un an plus tôt.

« **Ouvert** » est un dégradé, pas un état. Aux extrêmes : OLMo publie tout, Llama publie des poids sous conditions restrictives. Entre les deux, la majorité publie les poids sous licence permissive mais garde les données pour elle.

**L’Europe tient par Mistral**, seul acteur européen de premier plan, ce qui explique l’attention qu’on lui porte au-delà de ses parts de marché.

**Où vérifier l’état actuel.** Les classements publics de Hugging Face et les comparateurs par votes humains évoluent en continu ; les trackers de sorties recensent les nouveautés au jour le jour. Mais rappelez-vous le conseil du chapitre 11 : votre propre jeu de questions reste plus instructif que n’importe quel classement.

---

## 5. Les formats de fichiers

Un même modèle circule sous plusieurs emballages, incompatibles entre eux. Choisir le mauvais est la cause la plus fréquente d’échec au premier essai.

**GGUF** — Le format des logiciels grand public (llama.cpp et tout ce qui en dérive : Ollama, LM Studio, KoboldCpp). Un fichier unique, autonome, contenant les paramètres **et** le tokéniseur **et** les métadonnées. Il permet de répartir le modèle entre carte graphique et mémoire centrale, ce qui le rend utilisable sans GPU puissant. **C’est le format à choisir si vous débutez.** Successeur de GGML, qu’on rencontre encore dans de vieux tutoriels et qui n’est plus lu par les versions récentes.

**Safetensors** — Le format de référence de l’écosystème Python (Hugging Face transformers). Conçu pour être **sûr** : contrairement à l’ancien format, il ne peut pas exécuter de code à l’ouverture. C’est la forme sous laquelle les auteurs publient leur modèle d’origine, en pleine précision. *Ce que vous verrez : plusieurs fichiers `model-00001-of-00004.safetensors`.*

**Shards** (fragments) — Les gros modèles sont découpés en morceaux de quelques gigaoctets. Il faut **tous** les télécharger ; un fichier `.index.json` indique leur assemblage.

**.bin / pytorch\_model.bin** — L’ancien format PyTorch, fondé sur le mécanisme *pickle*

de Python, qui peut exécuter du code arbitraire à l'ouverture. À éviter quand une version safetensors existe.

**GPTQ, AWQ** — Deux formats compressés destinés aux cartes graphiques NVIDIA, très utilisés côté serveur. Plus rapides que GGUF sur GPU, mais tout doit tenir dans la mémoire vidéo : pas de répartition possible avec la mémoire centrale.

**EXL2 / EXL3** (ExLlamaV2, ExLlamaV3) — Format orienté performance sur GPU, à taux de compression finement réglable. Public averti.

**MLX** — Le format d'Apple, optimisé pour les puces Apple Silicon (M1 à M5). Sur un Mac récent, souvent plus rapide que GGUF.

**ONNX** — Format d'échange indépendant du cadre logiciel, surtout utilisé pour l'embarqué et les petits modèles.

## Les fichiers annexes d'un dépôt

| Fichier                | Rôle                                                                |
|------------------------|---------------------------------------------------------------------|
| config.json            | L'architecture : nombre de couches, dimensions, taille de contexte. |
| tokenizer.json         | Le vocabulaire et les règles de découpage.                          |
| chat_template          | Le gabarit de mise en forme du dialogue (voir ci-dessous).          |
| generation_config.json | Les réglages de génération conseillés par l'auteur.                 |
| README.md              | La <b>model card</b> , c'est-à-dire la documentation.               |

**Chat template** (gabarit de conversation) — Le modèle n'a pas été entraîné sur des dialogues bruts mais sur un balisage précis, du type `<|im_start|>user ... <|im_end|>`. Employer le mauvais gabarit dégrade fortement les réponses, parfois jusqu'à l'incohérence, sans qu'aucun message d'erreur n'apparaisse. Les logiciels modernes le lisent automatiquement dans le GGUF — c'est l'une des bonnes raisons de préférer ce format.

## 6. La quantification

C'est le chapitre décisif au moment de choisir un fichier, et le plus obscur pour un débutant.

**Quantification** (*quantization*) — Réduire la précision des nombres qui composent le modèle, pour diminuer sa taille. On passe par exemple de 16 bits par paramètre à 4. Le fichier est quatre fois plus petit, il tient dans une carte graphique ordinaire, et la qualité baisse — beaucoup moins qu’on ne l’imaginerait.

**L’analogie du format d’image.** Une photo enregistrée en JPEG de qualité 90 est trois fois plus légère que l’originale et vous ne voyez pas la différence. À qualité 50, elle reste tout à fait regardable. À qualité 10, elle est constellée d’artefacts. La quantification suit exactement cette courbe : très indolore au début, brutale à la fin.

## Les précisions de départ

| Sigle       | Nom                                                | Taille par paramètre |
|-------------|----------------------------------------------------|----------------------|
| <b>FP32</b> | virgule flottante 32 bits                          | 4 octets             |
| <b>FP16</b> | demi-précision                                     | 2 octets             |
| <b>BF16</b> | <i>bfloat16</i> , variante à plus grande amplitude | 2 octets             |
| <b>FP8</b>  | flottant 8 bits, de plus en plus courant           | 1 octet              |

La plupart des modèles sont publiés en BF16. C’est la référence non compressée, celle à laquelle on compare tout le reste.

## Lire un code de quantification

Prenons Q4\_K\_M :

| Morceau | Sens                                                                                                                                           |
|---------|------------------------------------------------------------------------------------------------------------------------------------------------|
| Q       | <i>Quantized</i> — compressé. Un F ou BF signifierait la pleine précision.                                                                     |
| 4       | Le nombre de <b>bits par paramètre</b> visé.                                                                                                   |
| K       | Méthode dite <i>K-quant</i> : la précision n’est pas uniforme, les parties sensibles du réseau sont mieux préservées.                          |
| M       | <i>Medium</i> . Avec S ( <i>small</i> , plus agressif) et L ( <i>large</i> , plus généreux), désigne le dosage à l’intérieur d’un même niveau. |

**IQ** (*I-quants*) — Une famille plus récente, qui s’appuie sur une **matrice d’importance** calculée à partir de textes réels pour repérer les paramètres à ne surtout pas dégrader. À taille égale, la qualité est meilleure qu’un K-quant ; en contrepartie le calcul est parfois un peu plus lent, en particulier sur processeur. *Ce que vous verrez : IQ4\_XS, IQ3\_M, IQ2\_XXS.*

**imatrix** (*importance matrix*) — Le fichier de mesure qui sert à produire les I-quants, et qui améliore aussi les K-quants. Une mention « imatrix » sur un dépôt est un gage de soin.

**Q4\_0, Q5\_1...** — Les formats historiques, sans le K. Dépassés : à taille égale, un K-quant fait mieux. On les rencontre encore sur d’anciens dépôts.

## Le tableau de référence

Bits réellement consommés par paramètre, et taille obtenue pour quelques modèles courants :

| Quantification | Bits/param. | 7 Md          | 14 Md         | 32 Md        | 70 Md        |
|----------------|-------------|---------------|---------------|--------------|--------------|
| Q2_K           | ≈ 2,8       | 2,4 Go        | 4,9 Go        | 11 Go        | 25 Go        |
| IQ3_XS         | ≈ 3,3       | 2,9 Go        | 5,8 Go        | 13 Go        | 29 Go        |
| Q3_K_M         | ≈ 3,9       | 3,4 Go        | 6,8 Go        | 16 Go        | 34 Go        |
| IQ4_XS         | ≈ 4,3       | 3,8 Go        | 7,5 Go        | 17 Go        | 38 Go        |
| <b>Q4_K_M</b>  | ≈ 4,8       | <b>4,2 Go</b> | <b>8,4 Go</b> | <b>19 Go</b> | <b>42 Go</b> |
| Q5_K_M         | ≈ 5,7       | 5,0 Go        | 10 Go         | 23 Go        | 50 Go        |
| Q6_K           | ≈ 6,6       | 5,8 Go        | 12 Go         | 26 Go        | 58 Go        |
| Q8_0           | ≈ 8,5       | 7,4 Go        | 15 Go         | 34 Go        | 74 Go        |
| BF16           | 16          | 14 Go         | 28 Go         | 64 Go        | 140 Go       |

*Calcul : taille ≈ nombre de milliards de paramètres × bits par paramètre ÷ 8. Les valeurs réelles s’en écartent de quelques pour cent selon l’architecture.*

## Comment choisir

- **Q4\_K\_M est le choix par défaut.** C’est le meilleur compromis connu, et de loin le plus téléchargé.
- **Q5\_K\_M ou Q6\_K** si la mémoire le permet : dégradation à peine mesurable.
- **Q8\_0** n’apporte presque rien par rapport à Q6\_K pour deux fois le volume. Utile surtout comme référence.
- **En dessous de Q4**, la dégradation devient perceptible ; en dessous de Q3, elle est franche.

- **Q2\_K et les IQ1/IQ2** relèvent du dépannage. Ils permettent de faire tourner un très gros modèle sur une petite machine, ce qui reste souvent préférable à un petit modèle en pleine précision.

**La règle qui résume tout** : *prenez le plus gros fichier qui tienne confortablement dans votre mémoire vidéo*. Et à taille de fichier égale, **un gros modèle très compressé bat généralement un petit modèle peu compressé** — un 32 Md en Q3 rend davantage qu'un 7 Md en Q8.

**Perplexité** (*perplexity*, PPL) — La mesure classique de la dégradation : à quel point le modèle est « surpris » par un texte de référence. Plus bas est meilleur. Les tableaux comparatifs de quantification s'appuient dessus, ainsi que sur la **divergence KL**, qui compare directement les réponses du modèle compressé à celles de l'original.

---

## 7. Faire tourner un modèle chez soi

### Les logiciels

**llama.cpp** — Le moteur d'inférence en C++ dont dépend tout l'écosystème grand public. Libre (licence MIT), extrêmement portable, tourne sur processeur seul comme sur carte graphique. La plupart des autres outils ne sont que des habillages autour de lui.

**Ollama** — La façon la plus simple de démarrer : une commande pour installer, une pour récupérer un modèle, une pour l'utiliser. Gère un catalogue de modèles préconfigurés et expose une **API compatible OpenAI**. Convient à la très grande majorité des usages personnels.

**LM Studio** — Interface graphique soignée pour chercher, télécharger et essayer des modèles, avec serveur local intégré. Très accessible, mais le logiciel n'est pas libre.

**Jan** — Alternative libre à LM Studio (licence AGPL), pleinement auditable.

**KoboldCpp, text-generation-webui** — Interfaces plus anciennes, riches en réglages, appréciées pour l'écriture et le jeu de rôle.

**vLLM, SGLang, TGI** — Moteurs destinés au service de plusieurs utilisateurs simultanés. Débit très supérieur, installation nettement plus technique. Inutiles pour un usage individuel.

**API compatible OpenAI** — Une convention d'interface, devenue un standard de fait : la plupart des moteurs locaux l'imitent, si bien qu'un logiciel prévu pour un service en ligne fonctionne avec votre modèle local en changeant une simple adresse.

## La mémoire, qui décide de tout

**VRAM** (mémoire vidéo) — La mémoire de la carte graphique. **C'est la contrainte numéro un.** Ce qui doit y tenir : le fichier du modèle, le **cache KV**, et une marge de fonctionnement d'environ un gigaoctet.

**RAM** (mémoire centrale) — Prend le relais quand la mémoire vidéo est insuffisante, au prix d'un ralentissement considérable : d'un ordre de grandeur, pas de quelques pour cent.

**Offloading** (déport) — Répartir les couches du modèle entre carte graphique et processeur. *Ce que vous verrez : le réglage `n_gpu_layers` ou « GPU layers ». Mettez-en le plus possible ; s'il ne tient pas tout, la vitesse s'effondre rapidement.*

**Cache KV** (*KV cache*) — Les calculs intermédiaires conservés d'un token à l'autre pour éviter de tout recommencer. Il **croît avec la longueur du contexte** et occupe la mémoire vidéo en plus du modèle : de quelques centaines de mégaoctets à plusieurs gigaoctets. C'est la raison pour laquelle un contexte de 100 000 tokens sature une carte bien avant que les poids ne posent problème. On peut le quantifier lui aussi (réglages `q8_0` ou `q4_0` pour le cache) afin de gagner de la place.

**GQA** (*grouped-query attention*) — Un aménagement d'architecture qui réduit fortement la taille du cache KV. Les modèles qui en disposent supportent bien mieux les longs contextes.

**mmap** — Technique de chargement qui permet de lancer un modèle plus gros que la mémoire disponible, en le lisant par morceaux depuis le disque. Pratique, mais lent si le disque n'est pas un SSD rapide.

## La vitesse

**tokens/s** (*tokens per second, t/s*) — La mesure de débit. Repères utiles : **en dessous de 5 t/s, l'usage devient pénible ; au-delà de 15 t/s, la lecture suit confortablement** ; un humain lit autour de 10 t/s.

**Prefill** (ou *prompt processing*) — La phase de lecture du contexte, très rapide car entièrement parallélisable. **Decode** — la phase de production, token par token, beaucoup plus lente.

**TTFT** (*time to first token*) — Le délai avant l'apparition du premier mot. Sur un long contexte, il devient perceptible.

**Flash Attention** — Une implémentation optimisée de l'attention, plus économe en mémoire et plus rapide. À activer si l'option existe.

**Batch** (lot) — Traiter plusieurs requêtes en parallèle. Augmente fortement le débit total, sans effet pour un utilisateur seul.

**CUDA, ROCm, Metal, Vulkan** — Les couches d'accès au matériel, respectivement pour NVIDIA, AMD, Apple et le cas générique. CUDA reste le mieux servi, ce qui explique la domination de NVIDIA sur ce terrain.

## Quel modèle pour quelle machine

| Mémoire vidéo         | Ce qui tourne confortablement                                                             |
|-----------------------|-------------------------------------------------------------------------------------------|
| 8 Go                  | 7-8 Md en Q4_K_M, contexte modéré                                                         |
| 12 Go                 | 12-14 Md en Q4_K_M                                                                        |
| 16 Go                 | 14 Md en Q5/Q6, ou 32 Md en Q3                                                            |
| 24 Go                 | 32 Md en Q4_K_M — le palier le plus intéressant                                           |
| 48 Go                 | 70 Md en Q4_K_M                                                                           |
| Mac à mémoire unifiée | La mémoire est partagée : un Mac de 64 Go fait tourner ce qu'exigerait une carte de 48 Go |

*Un modèle à mélange d'experts change ces repères : il faut la mémoire du total, mais la vitesse obtenue est celle du nombre de paramètres actifs.*

## 8. Les réglages de génération

Ce sont les curseurs que vous trouverez dans l'interface de n'importe quel logiciel local. Ils ne modifient pas le modèle : ils agissent sur la façon dont on **tire au sort** le token suivant parmi ceux qu'il propose.

**Température** (*temperature*) — Le réglage d'audace. À 0, le modèle prend toujours le token le plus probable : réponses reproductibles, mais plates et sujettes aux répétitions. Vers 0,7, équilibre habituel. Au-delà de 1,2, le texte devient inventif puis incohérent. *Pour du code ou de l'extraction de données, descendez vers 0,1 ; pour de la fiction, montez vers 1.*

**Top-p** (*nucleus sampling*) — Ne considérer que les tokens les plus probables dont les probabilités cumulées atteignent p. À 0,9, on écarte la longue traîne des options absurdes. Valeur usuelle : 0,9 à 0,95.

**Top-k** — Ne garder que les k tokens les plus probables. Plus rustique que top-p, souvent laissé à 40 ou désactivé.

**Min-p** — Écarter les tokens dont la probabilité est trop faible **par rapport au meilleur candidat**. Réglage plus récent, souvent supérieur à top-p, notamment

aux fortes températures.

**Repeat penalty** (pénalité de répétition) — Décourage la réutilisation des tokens déjà produits. Indispensable pour éviter les boucles, mais **à manier avec prudence** : trop élevé, il dégrade la grammaire en interdisant les mots-outils, qui se répètent légitimement. Les variantes *frequency penalty* et *presence penalty* agissent de façon comparable.

**Seed** (graine) — Le point de départ du générateur de nombres aléatoires qui effectue le tirage. Deux cas à distinguer, car c'est une source de confusion fréquente :

- **À température nulle, la graine ne sert à rien.** Il n'y a aucun tirage : le modèle prend systématiquement le token le plus probable. Les réponses sont donc déjà identiques d'une exécution à l'autre, graine ou pas.
- **Dès que la température est non nulle,** un tirage a lieu, et c'est là que la graine devient utile : la fixer rend ce tirage reproductible, donc la réponse aussi. On obtient ainsi de la variété d'un essai à l'autre tout en pouvant rejouer un résultat précis.

*En pratique : fixez la graine quand vous voulez comparer deux réglages sans que le hasard brouille la comparaison.*

**Max tokens** — La longueur maximale de la réponse. Une valeur trop basse tronque le texte au milieu d'une phrase.

**Stop sequences** — Des chaînes qui interrompent la génération dès qu'elles apparaissent.

**Streaming** — L'affichage au fil de la production, mot après mot. Ce n'est pas une animation : c'est le reflet direct du mécanisme.

**Décodage spéculatif** (*speculative decoding*) — Un petit modèle propose plusieurs tokens d'avance, le gros se contente de les valider en bloc. Accélère la génération sans changer le résultat.

**Grammar, JSON mode, structured output** — Contraindre la sortie à respecter une forme donnée en interdisant les tokens qui la briseraient. Très utile pour obtenir des données exploitables par un programme.

**Contexte configuré** — La taille de contexte que vous allouez au lancement, souvent inférieure au maximum annoncé du modèle, pour économiser la mémoire. *Ce que vous verrez : `n_ctx`, `num_ctx`, « Context Length ».*

**RoPE scaling, YaRN** — Des techniques d'extension du contexte au-delà de ce pour quoi le modèle a été entraîné. Elles fonctionnent, mais la qualité se dégrade dans les zones étendues.

**Le piège du contexte annoncé.** Un modèle « 128K » n’exploite pas également toute cette longueur. La compréhension se creuse souvent au milieu des textes très longs — phénomène connu sous le nom de *lost in the middle*. Considérez le chiffre annoncé comme un maximum technique, pas comme une garantie de qualité.

---

## 9. Entraînement et spécialisation

Vous croiserez ces termes dans les fiches de modèles, qui décrivent presque toujours la manière dont ils ont été fabriqués.

**Pré-entraînement** (*pre-training*) — La phase longue et coûteuse : apprendre à prédire la suite d’un texte sur des milliers de milliards de tokens. Plusieurs mois de calcul, des dizaines à des centaines de millions d’euros. Elle produit un modèle **Base**.

**Corpus, dataset** (jeu de données) — Les textes utilisés. Leur composition explique une grande part des forces et des angles morts d’un modèle — notamment sa qualité en français, qui dépend directement de la part de français dans le corpus.

**Données synthétiques** (*synthetic data*) — Des textes produits par un autre modèle pour servir à l’entraînement. Pratique désormais massive.

**Fine-tuning** (réglage fin, affinage) — Reprendre un modèle existant et poursuivre son entraînement sur un jeu de données restreint pour le spécialiser. Sans commune mesure avec le pré-entraînement en coût : quelques heures sur une carte graphique unique peuvent suffire.

**SFT** (*supervised fine-tuning*) — L’affinage sur des exemples de bonnes réponses rédigées ou validées par des humains. C’est la première étape qui transforme un compléteur de texte en interlocuteur.

**RLHF** (*reinforcement learning from human feedback*) — Le modèle produit plusieurs réponses, des humains indiquent lesquelles sont préférables, et il est ajusté dans ce sens. C’est là que se travaillent l’honnêteté, le refus de nuire et le ton.

**Reward model** (modèle de récompense) — Un modèle entraîné à imiter les jugements humains, pour éviter d’avoir à les solliciter des millions de fois.

**DPO** (*direct preference optimization*) — Une méthode plus simple et moins coûteuse que le RLHF classique, qui se passe du modèle de récompense. Très répandue dans les affinages communautaires.

**RLVR** (*reinforcement learning with verifiable rewards*) — L’entraînement sur des problèmes dont la réponse se vérifie automatiquement : un théorème se contrôle,

un programme passe ou non ses tests. Le modèle peut s'exercer des millions de fois sans supervision humaine. C'est ce qui a fait basculer les modèles de raisonnement du gadget à l'outil sérieux.

**Alignement** (*alignment*) — L'ensemble des techniques visant à faire correspondre le comportement du modèle aux intentions de ses concepteurs et aux attentes des utilisateurs.

**Complaisance** (*sycophancy*) — La tendance, induite par l'optimisation sur l'approbation humaine, à donner raison à l'utilisateur. Un défaut contre lequel il faut activement lutter : *un modèle qui ne vous contredit jamais n'est pas d'accord avec vous, il est mal réglé.*

**LoRA** (*low-rank adaptation*) — La technique d'affinage économique de référence : plutôt que de modifier les milliards de paramètres, on apprend un petit module de correction que l'on ajoute par-dessus. Quelques dizaines de mégaoctets au lieu de plusieurs gigaoctets, et l'on peut échanger les modules sans recharger le modèle.

**QLoRA** — LoRA appliqué à un modèle déjà quantifié. Permet d'affiner un modèle de 70 milliards de paramètres sur une seule carte grand public.

**PEFT** (*parameter-efficient fine-tuning*) — La famille de méthodes à laquelle appartient LoRA. *Ce que vous verrez : la bibliothèque peft de Hugging Face.*

**Adapter** (adaptateur) — Le petit module appris séparément, LoRA en étant la forme la plus répandue.

**Rank** (rang) — Le paramètre de taille d'un LoRA, typiquement de 8 à 128. Plus il est élevé, plus le module peut apprendre, plus il est lourd et plus il risque d'abîmer le modèle d'origine.

**Distillation** — Entraîner un petit modèle à imiter les sorties d'un grand. C'est ainsi que l'on obtient des modèles compacts étonnamment performants. *Ce que vous verrez : le suffixe Distill.*

**Oubli catastrophique** (*catastrophic forgetting*) — Le risque propre à tout affinage : en apprenant la nouvelle tâche, le modèle perd des capacités générales. Un affinage maladroit produit un modèle excellent sur son domaine et médiocre partout ailleurs.

**Merge** (fusion) — Combiner plusieurs modèles en moyennant leurs paramètres. Sans entraînement, donc gratuit ; les résultats vont de l'excellent au n'importe quoi.

**Abliteration** — Une manipulation qui identifie puis neutralise la direction interne associée au refus, afin de supprimer les garde-fous. Technique courante dans les modèles dits *uncensored*, avec des effets de bord sur la qualité générale.

**Checkpoint** — Un instantané des paramètres à un moment de l'entraînement.

Certains dépôts en publient plusieurs.

---

## 10. Architectures et familles

**Transformer** — L'architecture qui domine le domaine depuis 2017. Le « T » de GPT. Sa particularité : tous les tokens sont traités en parallèle et dialoguent entre eux par le mécanisme d'attention.

**Attention** — Le mécanisme par lequel chaque token va chercher l'information pertinente chez les autres. Son coût augmente avec le **carré** de la longueur du texte, ce qui explique le prix des longs contextes.

**Tête d'attention** (*attention head*) — Une des dizaines d'unités d'attention qui travaillent en parallèle dans chaque couche, chacune s'occupant d'un type de relation.

**Couche** (*layer*) — Un étage de traitement. Un modèle en compte typiquement de 30 à 120. L'information s'abstrait en montant : syntaxe en bas, intentions en haut.

**Dense** — Se dit d'un modèle classique, où tous les paramètres travaillent à chaque token. Par opposition au mélange d'experts.

**MoE** (*mixture of experts*, mélange d'experts) — Une architecture où le réseau est découpé en sous-réseaux spécialisés, dont seuls quelques-uns s'activent pour chaque token. Permet de très gros modèles à coût de calcul réduit. **La contrepartie est mémoire** : il faut tout charger, même ce qui ne sert pas.

**Expert** — Un des sous-réseaux d'un MoE. **Router** (routeur) — le petit réseau qui décide, pour chaque token, quels experts solliciter.

**Paramètres actifs** (*active parameters*) — Ceux réellement mobilisés par token dans un MoE. *Ce que vous verrez : la notation A3B.*

**SSM** (*state space model*), **Mamba** — Une famille d'alternatives à l'attention, dont le coût croît linéairement et non quadratiquement avec la longueur. Beaucoup plus économe sur les longs textes.

**Architecture hybride** — L'alternance de couches d'attention classiques et de couches à espace d'états, devenue courante en 2026 pour concilier qualité et contextes longs.

**Multimodal** — Se dit d'un modèle qui accepte plusieurs types d'entrée : texte, images, parfois son ou vidéo.

**VLM** (*vision-language model*) — Un modèle qui lit des images en plus du texte. Les images sont découpées en carrés puis converties en vecteurs insérés dans la suite des tokens, ce qui permet de réutiliser exactement la même machinerie.

**Modèle de diffusion** — Un mécanisme entièrement différent, employé pour **produire** des images : on part d'un brouillard aléatoire et on le débruite progressivement. À ne pas confondre avec les modèles de langage, qui écrivent token par token.

**Modèle d'embedding** — Un modèle qui ne produit pas de texte mais convertit un document en un vecteur unique, afin de comparer des textes par leur sens. Brique de base de la recherche documentaire.

**Reranker** — Un modèle qui reclasse des résultats de recherche par pertinence. Plus lent qu'un modèle d'embedding, mais nettement plus fin.

---

## 11. Mesurer et comparer

**Benchmark** — Un jeu d'épreuves standardisé permettant de comparer des modèles. À prendre avec réserve : ce sont des examens, et l'on peut réviser pour un examen.

| Sigle                  | Ce qu'il mesure                                                                                                                                                                             |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>MMLU</b>            | Culture générale et raisonnement, questionnaire à choix multiples couvrant 57 disciplines. Longtemps la référence, aujourd'hui saturé ; sa variante <b>MMLU-Pro</b> est plus discriminante. |
| <b>GPQA</b>            | Questions de niveau doctoral en sciences, conçues pour résister à la simple recherche en ligne.                                                                                             |
| <b>HumanEval, MBPP</b> | Programmation : produire une fonction qui passe des tests.                                                                                                                                  |
| <b>SWE-bench</b>       | Résoudre de vrais tickets de bogues sur des dépôts réels. Beaucoup plus exigeant et plus représentatif.                                                                                     |
| <b>AIME, MATH</b>      | Mathématiques de compétition.                                                                                                                                                               |
| <b>IFEval</b>          | Capacité à suivre des consignes de forme précises.                                                                                                                                          |

**Leaderboard** (classement) — Un tableau comparatif public. Le plus consulté pour les modèles ouverts est hébergé sur Hugging Face.

**LMarena** (anciennement *Chatbot Arena*) — Un classement fondé sur des votes humains à l’aveugle : deux réponses anonymes, l’utilisateur choisit la meilleure. Produit un score **Elo**, comme aux échecs. Moins manipulable que les examens automatiques, mais favorise le style agréable autant que la justesse.

**Contamination** — La présence, dans les données d’entraînement, des questions du test lui-même. Le modèle ne raisonne pas, il restitue. C’est le problème structurel des benchmarks, et la raison de la course aux épreuves inédites.

**Eval** — Toute procédure d’évaluation, y compris celles que l’on construit soi-même pour son propre usage — souvent la seule vraiment informative.

**Le conseil qui vaut pour tous les classements.** Constituez-vous **une dizaine de questions représentatives de votre usage réel** et posez-les à chaque modèle que vous essayez. Ce test maison vous renseignera mieux que n’importe quel tableau : un modèle brillant en mathématiques de compétition peut se révéler médiocre pour rédiger vos courriers en français.

---

## 12. Licences et libertés

Le vocabulaire est ici trompeur, et les conséquences juridiques réelles.

**Poids ouverts** (*open weights*) — Le fichier de paramètres est téléchargeable. **Cela ne dit rien des droits d’usage.**

**Open source** — Au sens strict, suppose la publication du code **et** des données permettant de reconstruire le modèle, sous une licence libre. Très peu de modèles y satisfont réellement. L’emploi du terme dans la communication des éditeurs est le plus souvent abusif.

**Apache 2.0, MIT** — De véritables licences libres, permissives : usage commercial autorisé, modification autorisée, redistribution autorisée. *Ce sont les licences les plus confortables, et celles de la plupart des modèles chinois ouverts.*

**Licences maison** — Plusieurs grands éditeurs imposent leurs propres conditions, qui ne sont pas des licences libres. On y trouve typiquement des seuils d’utilisateurs au-delà desquels une autorisation devient nécessaire, des obligations d’attribution, ou des restrictions sur l’entraînement d’autres modèles à partir des sorties. **Lisez le texte** plutôt que l’étiquette.

**Non-commercial** (*CC-BY-NC, research only*) — Usage personnel et recherche uniquement. Fréquent sur les modèles expérimentaux.

**Acceptable use policy** — Une liste d’usages interdits, annexée à la licence et opposable.

**Gated model** (modèle sous condition) — Un dépôt qui exige d'accepter des conditions, voire une validation manuelle, avant le téléchargement. *Ce que vous verrez : un bouton « Agree and access repository ».*

**Provenance des données** — Question distincte de la licence du modèle, et juridiquement instable : la légalité de l'entraînement sur des œuvres protégées fait l'objet de contentieux en cours dans plusieurs pays. Une licence permissive sur les poids ne règle pas cette question.

**En pratique.** Pour un usage personnel, ces distinctions ont peu de portée. Dès qu'un usage professionnel ou une redistribution est envisagé, lisez le fichier de licence du dépôt — pas le résumé affiché, le texte intégral.

---

## 13. Autour du modèle

Un modèle seul ne fait pas grand-chose. Voici le vocabulaire de ce qu'on construit autour.

### Formuler une demande

**Prompt engineering** — L'art de formuler une demande. Moins ésotérique qu'on ne le dit : être précis, donner des exemples, annoncer le format attendu, et demander explicitement de raisonner étape par étape suffisent à couvrir l'essentiel.

**Zero-shot** — Demander sans fournir d'exemple. **Few-shot** — en fournir deux ou trois dans la demande, ce qui améliore considérablement la régularité du résultat.

**Chain of thought** (chaîne de pensée, CoT) — Demander au modèle de dérouler son raisonnement avant de conclure. Ce n'est pas cosmétique : le texte produit sert au modèle de mémoire de travail, et lui donne réellement davantage de moyens de calcul.

**Context engineering** — L'organisation de ce qu'on place dans le contexte : consignes, documents, historique, résultats d'outils. Devenu l'enjeu central dès qu'on dépasse le simple dialogue.

### Faire travailler le modèle sur vos documents

**RAG** (*retrieval-augmented generation*, génération augmentée par la recherche) — La méthode de référence pour interroger vos propres documents. On découpe les documents, on les indexe, on retrouve les passages pertinents à chaque question, et on les glisse dans le contexte. **Le modèle n'apprend rien : on lui met les bons extraits sous les yeux.**

**Chunking** (découpage) — Fractionner les documents en passages de taille exploitable. Étape déterminante pour la qualité : un découpage qui sépare une question de sa réponse rend l'ensemble inefficace.

**Base vectorielle** (*vector database*) — Le stockage qui permet de retrouver les passages par proximité de sens plutôt que par mots-clés.

**RAG, affinage ou simple invite ?** La question revient constamment. Pour **ajouter des connaissances** — vos documents, vos procédures — c'est le RAG, presque toujours. Pour **changer un comportement ou un style**, c'est l'affinage. Pour tout le reste, une bonne invite système suffit. L'erreur classique consiste à vouloir affiner un modèle pour lui « apprendre » des documents : coûteux, fragile, et moins efficace qu'un RAG.

## Faire agir le modèle

**Tool calling, function calling** (appel d'outils) — Le modèle n'exécute rien lui-même : il émet une demande structurée — « *appelle la fonction météo avec le paramètre Marseille* » — qu'un programme extérieur exécute avant de lui renvoyer le résultat. C'est ainsi qu'il calcule juste, consulte le web ou lit un fichier.

**Agent** — Un modèle placé dans une boucle : observer, décider, agir, vérifier, recommencer. C'est ce qui permet d'enchaîner une tâche longue sans intervention. La difficulté n'est pas la décision mais la **fiabilité** : une tâche en cinquante étapes réussies à 98 % chacune n'aboutit en entier qu'une fois sur trois environ.

**MCP** (*Model Context Protocol*) — Un protocole standardisé de connexion entre un modèle et des outils ou des sources de données. Devenu la norme du secteur : il évite de réécrire une intégration sur mesure pour chaque logiciel.

## Sécurité

**Jailbreak** — Une formulation destinée à contourner les garde-fous d'un modèle.

**Prompt injection** (injection d'invite) — Une attaque plus sérieuse : des instructions malveillantes dissimulées dans un document ou une page web que le modèle va lire, et qu'il risque de suivre comme si elles venaient de vous. **C'est le problème de sécurité majeur des agents**, et il n'a pas de solution complète à ce jour. Règle de prudence : tout contenu extérieur est une donnée, jamais une consigne.

**Guardrails** (garde-fous) — Les filtres et contraintes ajoutés autour du modèle pour encadrer ses sorties.

## Utiliser un service en ligne

**API** — L'interface programmatique d'un service. **Token pricing** — la facturation, comptée séparément pour les tokens envoyés et produits, les seconds coûtant généralement trois à cinq fois plus cher. **Rate limit** — le plafond de requêtes autorisé par minute.

**Inference provider** (hébergeur d'inférence) — Une société qui fait tourner des modèles ouverts et en vend l'accès. Utile pour essayer un modèle de 400 milliards de paramètres sans l'installer.

---

## 14. Faux amis et pièges courants

Quelques confusions qui coûtent du temps à tout le monde au début.

« **Open source** » ne veut pas dire « **librement utilisable** ». Vérifiez la licence, pas le discours.

**Un gros modèle compressé bat souvent un petit modèle intact.** À taille de fichier égale, préférez le plus grand nombre de paramètres.

**Les paramètres d'un MoE ne sont pas ceux qu'on croit.** 30B-A3B exige la mémoire d'un 30 milliards et offre la vitesse d'un 3 milliards. C'est un compromis de vitesse, pas de mémoire.

**Le contexte annoncé est un maximum, pas une garantie.** Un modèle « 1M » peut perdre le fil bien avant.

**Un modèle Base n'est pas un assistant.** Si vos réponses partent en digressions ou enchaînent les questions, vous avez probablement téléchargé la mauvaise variante.

**Un modèle d'embedding ne discute pas.** Il ne produit pas de texte du tout.

**Le nombre de téléchargements ne mesure pas la qualité** mais l'ancienneté et la visibilité.

**Les benchmarks sont des examens, et on peut réviser pour un examen.** Construisez votre propre test.

**La quantification du cache KV existe aussi.** Si vous manquez de mémoire sur un long contexte, c'est souvent le premier réglage à ajuster, avant de descendre en qualité de modèle.

**Le mauvais gabarit de conversation dégrade tout, silencieusement.** Aucun message d'erreur ne vous préviendra.

**Un modèle local ne se met pas à jour.** Ses connaissances sont figées à sa date d'entraînement, et il n'apprend rien de vos conversations.

---

## 15. Par où commencer, concrètement

Si vous voulez essayer dès ce soir, voici le chemin le plus court.

1. **Installez Ollama** (ou LM Studio si vous préférez une interface graphique). Les deux fonctionnent sous Linux, macOS et Windows.
2. **Commencez petit** : un modèle de 7 à 8 milliards de paramètres en Q4\_K\_M. Il tiendra sur presque toutes les machines et vous donnera une base de comparaison.
3. **Vérifiez votre débit**. En dessous de 5 tokens par seconde, descendez d'un cran en taille ou en quantification.
4. **Montez ensuite** vers le plus gros modèle que votre mémoire vidéo accepte, toujours en Q4\_K\_M.
5. **Testez avec vos propres questions**, pas avec celles des classements.

Trois pièges à éviter pour un premier essai : télécharger un modèle Base au lieu d'Instruct, prendre un fichier plus gros que la mémoire disponible, et juger un modèle sur une seule question.

À ce stade, vous avez un modèle qui répond. Le chapitre suivant traite de l'étape d'après — celle qui justifie vraiment de l'avoir installé chez vous.

---

## 16. Passer à l'action : donner des outils au modèle

Une question s'impose : **pourquoi avoir installé tout cela, alors que des assistants gratuits et plus performants sont accessibles depuis n'importe quel navigateur ?**

La confidentialité est une réponse, mais elle ne suffit pas à convaincre grand monde. La vraie réponse est ailleurs : **un modèle installé chez vous peut recevoir vos propres outils**, et donc accéder à vos données et agir sur votre machine. Il cesse d'être un conseiller à qui l'on décrit sa situation pour devenir un exécutant qui la consulte directement.

La différence est considérable. Comparez :

---

| Sans outils                                                       | Avec outils                                                                                  |
|-------------------------------------------------------------------|----------------------------------------------------------------------------------------------|
| « Comment trier mes courriels ? » — il vous explique une méthode. | « Trie mes courriels de la semaine et résume ceux qui attendent une réponse. » — il le fait. |

| Sans outils                                                              | Avec outils                                                              |
|--------------------------------------------------------------------------|--------------------------------------------------------------------------|
| « Comment renommer des photos par date ? » — il vous donne une commande. | « Renomme et classe ces photos par date de prise de vue. » — c’est fait. |
| « Quel est mon planning ? » — il n’en sait rien.                         | Il lit votre agenda et répond.                                           |

C’est le **MCP** (chapitre 13) qui rend cela possible : un protocole standardisé par lequel un logiciel expose des capacités — lire une boîte aux lettres, consulter un agenda, parcourir un dossier, piloter une installation domotique — que le modèle peut appeler.

**Que brancher en premier.** Les connexions les plus immédiatement utiles à un particulier sont, dans l’ordre : l’accès à un **dossier de fichiers**, l’**agenda**, la **messagerie**, puis les services qu’on héberge déjà chez soi. Le RAG, lui, vient plus tard : il ne se justifie que si vous avez un vrai corpus de documents à interroger.

## Les harnais : les logiciels qui font agir le modèle

Entre votre modèle et vos outils, il faut un intermédiaire : un **harnais d’agent**, qui tient la boucle observer → décider → agir → vérifier, gère les connexions MCP et vous demande votre accord avant les actions sensibles. Tous fonctionnent avec un modèle local ; tous sont libres.

| Outil               | Éditeur               | Licence    | Pour quoi faire                                                                                                                                                                                                     |
|---------------------|-----------------------|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Goose</b>        | Agentic AI Foundation | Apache 2.0 | <b>Le plus accessible pour un usage non spécialisé.</b> Application de bureau ou ligne de commande, large répertoire d’extensions MCP, automatisation bien au-delà du code. Projet hébergé par la Linux Foundation. |
| <b>Hermes Agent</b> | Nous Research         | Libre      | Agent personnel persistant : mémoire d’une session à l’autre, nombreuses compétences, passerelles vers les messageries instantanées.                                                                                |
| <b>OpenClaw</b>     | Communauté            | Libre      | Assistant piloté depuis une messagerie, pour qui préfère écrire à son agent plutôt qu’ouvrir une application.                                                                                                       |

| Outil            | Éditeur    | Licence    | Pour quoi faire                                                                          |
|------------------|------------|------------|------------------------------------------------------------------------------------------|
| <b>Cline</b>     | Cline      | Apache 2.0 | Intégré à l'éditeur VS Code, avec validation fichier par fichier. Orienté programmation. |
| <b>Aider</b>     | Aider AI   | Apache 2.0 | Programmation en binôme, adossé à Git : chaque modification devient un commit.           |
| <b>OpenHands</b> | Communauté | Libre      | Tâches longues exécutées dans un conteneur isolé. Plus lourd, plus sûr.                  |

Les interfaces graphiques évoquées au chapitre 7 — LM Studio, Jan, Open WebUI — intègrent également le MCP de façon croissante : selon votre besoin, vous n'aurez peut-être aucun logiciel supplémentaire à installer.

**Faites-vous aider pour l'installation.** C'est le conseil le moins intuitif et le plus utile de ce chapitre. Brancher un serveur MCP suppose d'éditer un fichier de configuration, de renseigner des chemins, parfois de déboguer une erreur obscure — exactement le genre de tâche pour laquelle ces modèles sont excellents. **Décrivez votre système et demandez la marche à suivre pas à pas** ; collez-lui les messages d'erreur. C'est aussi un excellent premier test : un modèle qui vous accompagne correctement dans son propre outillage est un modèle qui vous servira ensuite.

## Les précautions, et elles sont sérieuses

Donner des outils à un modèle change la nature du risque. Jusque-là, une erreur produisait une phrase fausse ; désormais elle peut supprimer un fichier ou envoyer un courriel.

- **L'injection d'invite est le danger principal** (chapitre 13). Un courriel reçu, une page web consultée peuvent contenir des instructions dissimulées que le modèle risque de suivre comme si elles venaient de vous. Ce n'est pas une hypothèse d'école : c'est le problème de sécurité non résolu des agents, et il devient concret dès que vous branchez une messagerie.
- **Commencez en lecture seule.** Laissez-le consulter avant de l'autoriser à écrire, envoyer ou supprimer.
- **Gardez la validation manuelle active** au début. Tous ces outils demandent confirmation avant d'agir ; ne désactivez cette option qu'une fois en confiance, et jamais pour des actions irréversibles.
- **Isolez ce qui peut l'être.** Un dossier de travail dédié plutôt que votre répertoire personnel entier ; un conteneur si l'outil le propose.

- **N'accordez jamais à un agent un accès dont vous ne pourriez pas assumer le mauvais usage** — messagerie professionnelle, services bancaires, sauvegardes.

## Et ensuite

Une fois l'outillage en place, les usages viennent d'eux-mêmes : tri et résumé de courriels, gestion de fichiers, préparation d'une revue de presse, rappels agenda, pilotage d'une installation domotique. **C'est à ce moment seulement qu'un modèle local devient préférable à un service en ligne** — non parce qu'il répond mieux, mais parce qu'il travaille chez vous, sur vos données, avec vos outils, sans rien en envoyer ailleurs.

---

## Index alphabétique

Le nombre renvoie au chapitre.

- **Abliteration** — 3, 9
- **Acceptable use policy** — 12
- **Adapter** (adaptateur) — 9
- **Agent** — 13, 16
- **Ai2, OLMo** — 4
- **Aider** — 16
- **AIME** — 11
- **Alibaba** — 4
- **Alignement** — 9
- **Anthropic** — 4
- **Apache 2.0** — 12
- **API** — 13
- **API compatible OpenAI** — 7
- **Architecture hybride** — 10
- **Attention** — 10
- **AWQ** — 5
- **Base** (variante) — 3
- **Base vectorielle** — 13
- **Batch** — 7
- **Benchmark** — 11
- **BF16** — 6
- **Cache KV** — 7
- **Chain of thought** — 13
- **Chat template** — 5
- **Checkpoint** — 9
- **Chunking** — 13
- **Cline** — 16
- **Coder** (variante) — 3
- **Cohere** — 4
- **Complaisance** — 9
- **Contamination** — 11
- **Context engineering** — 13
- **Contexte** — 2, 8
- **Corpus** — 9
- **Couche** — 10
- **CUDA** — 7
- **Dataset** — 9
- **Décodage spéculatif** — 8
- **DeepSeek** — 4
- **Dense** — 10
- **Diffusion** (modèle de) — 10
- **Distillation, Distill** — 3, 9
- **Divergence KL** — 6
- **Données synthétiques** — 9
- **DPO** — 9
- **Elo** — 11
- **Embedding** — 2, 10
- **Eval** — 11
- **EXL2** — 5
- **Expert** — 10
- **Falcon** — 4
- **Few-shot** — 13

- **Fine-tuning** — 9
- **Flash Attention** — 7
- **FP8 / FP16 / FP32** — 6
- **Function calling** — 13
- **Gated model** — 12
- **Gemini** — 4
- **Gemma** — 4
- **Génération** — 2
- **GGML** — 5
- **GGUF** — 5
- **GLM** — 4
- **Google DeepMind** — 4
- **Goose** — 16
- **GPQA** — 11
- **gpt-oss** — 4
- **GPTQ** — 5
- **GQA** — 7
- **Grammar** — 8
- **Granite** — 4
- **Grok** — 4
- **Guardrails** — 13
- **Hallucination** — 2
- **Harnais d’agent** — 16
- **Hermes Agent** — 16
- **Hugging Face** — 1
- **HumanEval** — 11
- **IFEval** — 11
- **imatrix** — 6
- **Inférence** — 2
- **Inference provider** — 13
- **Instruct** — 3
- **IQ (I-quants)** — 6
- **Jailbreak** — 13
- **Jan** — 7
- **JSON mode** — 8
- **K-quant** — 6
- **Kimi** — 4
- **KoboldCpp** — 7
- **Leaderboard** — 11
- **Llama** — 4
- **llama.cpp** — 7
- **LLM** — 2
- **LM Studio** — 7
- **LMarena** — 11
- **LoRA** — 9
- **Lost in the middle** — 8
- **Mamba** — 10
- **Max tokens** — 8
- **MCP** — 13, 16
- **Merge** — 3, 9
- **Meta** — 4
- **Metal** — 7
- **Min-p** — 8
- **MiniMax** — 4
- **Mistral AI** — 4
- **MIT (licence)** — 12
- **MLX** — 5
- **mmap** — 7
- **MMLU** — 11
- **Model card** — 1, 5
- **MoE** — 3, 10
- **Moonshot AI** — 4
- **Multimodal** — 10
- **Nemotron** — 4
- **Nvidia** — 4
- **Offloading** — 7
- **Ollama** — 7
- **Omni** — 3
- **ONNX** — 5
- **Open source** — 2, 12
- **OpenAI** — 4
- **OpenClaw** — 16
- **OpenHands** — 16
- **Oubli catastrophique** — 9
- **Paramètre** — 2
- **Paramètres actifs** — 3, 10
- **PEFT** — 9
- **Perplexité** — 6
- **Phi** — 4
- **Poids** — 2
- **Poids ouverts** — 2, 12
- **Pré-entraînement** — 9
- **Prefill** — 7
- **Prompt** — 2
- **Prompt engineering** — 13
- **Prompt injection** — 13, 16
- **Q4\_K\_M** — 6
- **QLoRA** — 9
- **Quantification** — 6
- **Qwen** — 4

- **RAG** — 13, 16
- **RAM** — 7
- **Rank** — 9
- **Rate limit** — 13
- **Reasoning** — 3
- **Repeat penalty** — 8
- **Reranker** — 3, 10
- **Reward model** — 9
- **RLHF** — 9
- **RLVR** — 9
- **ROCm** — 7
- **RoPE scaling** — 8
- **Router** — 10
- **Safetensors** — 5
- **Seed** — 8
- **SFT** — 9
- **Shards** — 5
- **SSM** — 10
- **Stop sequences** — 8
- **Streaming** — 8
- **SWE-bench** — 11
- **System prompt** — 2
- **Température** — 8
- **Tête d'attention** — 10
- **TGI** — 7
- **Thinking** — 3
- **Token** — 2
- **Tokenizer** — 2
- **Tokens/s** — 7
- **Tool calling** — 13
- **Top-k, Top-p** — 8
- **Transformer** — 10
- **TTFT** — 7
- **Uncensored** — 3
- **VL / Vision** — 3
- **vLLM** — 7
- **VLM** — 10
- **Vocabulaire** — 2
- **VRAM** — 7
- **Vulkan** — 7
- **xAI** — 4
- **YaRN** — 8
- **Z.ai** — 4
- **Zero-shot** — 13

---

*Lexique accessible sans aucun prérequis. Les ordres de grandeur, les tailles de fichiers et les noms de logiciels reflètent l'état du domaine en septembre 2026 — c'est la partie qui se périmera le plus vite.*

*Document compagnon : « Comprendre les IA génératives », qui explique le fonctionnement plutôt que le vocabulaire.*