SÉCURITÉ & IA

# Une IA locale qui tient la route

*Matériel, modèles MoE, llama.cpp, Colibri, LoRA : ce que demande vraiment une IA dans ses murs, avec nos mesures, et pourquoi une base documentaire vérifiable compte plus que la carte graphique.*

Publié le 26 novembre 2026 — Imprimé le 26 novembre 2026



Une IA dans ses murs, pour que les documents ne quittent pas l'entreprise: l'idée séduit, et les offres se multiplient. Avant de parler de cartes graphiques, il faut savoir ce qu'on veut lui faire faire, et comment on saura qu'elle ne raconte pas n'importe quoi. Voici ce que nous en avons appris en la faisant tourner chez nous.

## D'abord la mémoire, ensuite la puissance

Un modèle de langage doit tenir entier en mémoire rapide pour répondre vite. C'est la quantité de mémoire, puis sa vitesse, qui décident de ce qu'une machine peut faire. Trois familles se partagent le terrain:

- **Les cartes graphiques RTX.** Très rapides, mais leur mémoire est limitée: 12 à 32 Go sur les cartes courantes, 96 Go sur la RTX PRO 6000 Blackwell, dont le prix a presque doublé en quelques mois avec la pénurie de mémoire. Idéales pour un modèle petit ou moyen, servi à beaucoup de monde.
- **Les machines à mémoire unifiée**: mini-PC AMD Strix Halo, DGX Spark de NVIDIA, Mac à puce Apple. Processeur et carte graphique partagent jusqu'à 128 Go, ce qui permet de charger de gros modèles dans un petit boîtier silencieux. Leur mémoire est plus lente, donc chaque machine sert moins de personnes à la fois.
- **Les serveurs**, quand il faut servir des dizaines d'utilisateurs en même temps. Rarement nécessaires dans une PME.

Les prix ont flambé en 2026. Un mini-PC Strix Halo à 128 Go, qui valait entre 1 500 et 2 200 dollars au printemps, en coûte aujourd'hui 3 500 à 4 300. Il faut donc chiffrer le jour même, et préférer plusieurs machines moyennes à une seule très chère: si l'une tombe, les autres continuent.

## Les modèles MoE changent la donne

Les modèles «à mélange d'experts» (MoE) n'activent qu'une petite partie de leurs paramètres pour chaque mot produit. Un modèle de 80 ou 125 milliards de paramètres n'en sollicite que quelques milliards à la fois: il occupe beaucoup de mémoire, mais il répond presque aussi vite qu'un petit modèle.

C'est exactement ce qui convient aux machines à mémoire unifiée. À l'inverse, un gros modèle «dense», qui mobilise tous ses paramètres à chaque mot, y devient très lent.

## Ce que nous avons mesuré




Notre machine de travail est un mini-PC Strix Halo à 128 Go de mémoire unifiée, sous Fedora, avec llama.cpp comme moteur. Mesures relevées début novembre 2026, en usage réel par des agents de développement:

- **Qwen3-Coder-Next**, modèle MoE spécialisé dans le code: 46 à 47 jetons générés par seconde, 750 à 875 lus par seconde, 49 Go de mémoire. Sur une journée: 773 requêtes et 2,4 millions de jetons lus, sans plantage.
- **Qwen3.8-Flash-Next**, modèle généraliste plus gros: 25 jetons par seconde, environ 500 en lecture, 80 Go de mémoire.

Ces chiffres sont ceux d'une seule machine. Dans une grappe, chaque boîtier sert ses propres demandes: les débits s'additionnent, et quatre à six machines produisent en principe quatre à six fois plus.

Deux leçons pratiques. La longueur de contexte a une limite physique: au-delà de 131 072 jetons, la mémoire a débordé et la machine a dû redémarrer. Et une machine ne fait pas tout à la fois: servir un modèle, faire tourner des agents et calculer des images en parallèle l'a saturée. On répartit, ou on planifie.

## llama.cpp, Colibri et les autres

**llama.cpp** est le moteur de référence pour faire tourner un modèle en local. Il fonctionne sur presque tout (cartes NVIDIA, AMD, mémoire unifiée, Mac), expose une interface compatible avec les outils du marché et se règle finement. C'est celui que nous utilisons.

**Colibri** est un moteur plus récent qui fait circuler les «experts» d'un modèle MoE entre la mémoire graphique, la mémoire vive et le disque. Il promet de faire tourner des modèles bien trop gros pour la mémoire disponible, sur du matériel ordinaire. Il se présente lui-même comme expérimental: nous l'évaluons, nous ne le recommandons pas encore en production.

## Un modèle par besoin

Il n'existe pas de modèle unique qui fasse tout bien. Ce qui fonctionne, c'est un petit catalogue:

- **Rédaction et documents courants**: un modèle généraliste moyen, rapide.
- **Code**: un modèle spécialisé, comme Qwen3-Coder-Next, sur des tâches courtes et bornées.
- **Images et écrans**: un modèle de vision, pour lire une capture, une facture scannée, une interface.
- **Contrôle**: un second modèle qui relit le premier sur ce qui compte: chiffres, dates, montants.

Pour le complexe et le non confidentiel, les grands modèles en ligne restent disponibles. L'essentiel est de savoir, pour chaque tâche, où passe la donnée.

## Une vraie base plutôt que des jetons

Beaucoup d'offres font relire au modèle des centaines de pages à chaque question. C'est coûteux en mémoire, lent dès que plusieurs personnes travaillent, et invérifiable: le modèle peut mélanger, déformer ou inventer.

ClearMyDocs v4, notre moteur documentaire, part de l'autre bout. Les documents sont traités une fois, et chaque fait retenu est cité mot pour mot avec la pièce d'où il vient. Un contrôle automatique refuse tout ce qui ne s'appuie pas sur une source. Le modèle rédige ensuite à partir de ces faits vérifiés, et chaque affirmation du résultat reste reliée à son document.

ClearMyDocs v4 fonctionne depuis plusieurs mois avec un modèle local de taille raisonnable, sur des dossiers juridiques et de facturation réels. C'est la réponse que nous donnons à la question qui compte: que fait-on

quand le modèle se trompe? On ne le croit pas sur parole: on vérifie, automatiquement, contre la source.

## Et l'entraînement: LoRA ou QLoRA?

LoRA permet d'adapter un modèle existant à un domaine en n'entraînant qu'une petite couche supplémentaire, sans toucher au modèle d'origine. QLoRA fait de même sur un modèle compressé, ce qui tient dans beaucoup moins de mémoire, au prix d'un peu de précision.

Nous l'expérimentons pour apprendre à un modèle local les règles d'un projet à partir de travaux déjà jugés bons. Mais pour la plupart des entreprises, ce n'est pas la première étape. Une bonne base documentaire et des contrôles apportent davantage, plus vite, qu'un modèle réentraîné. L'adaptation vient ensuite, quand on sait précisément ce qui manque.

## En résumé

- Choisir la mémoire avant la puissance, et plusieurs machines plutôt qu'une seule.
- Préférer les modèles MoE sur les machines à mémoire unifiée.
- Un modèle par besoin, et un second pour contrôler le premier.
- Une base documentaire vérifiable plutôt que des pages relues à chaque question.
- Mesurer sur ses propres documents avant de décider.

ClearMyDocs et nos logiciels · Nous écrire