EN

# Practical Guide

## OpenClaw + DeepSeek:
## Local LLM, Open Source and API Costs

Hybrid strategy, real savings and BOTUM field feedback

Mars 2026

## Table of Contents

1. Why DeepSeek?
2. OpenClaw + Ollama + DeepSeek Architecture
3. Hybrid Routing Strategy
4. Real Savings and ROI Calculation
5. Known Limitations
6. BOTUM Field Feedback
7. DeepSeek + OpenClaw Deployment Checklist

*DeepSeek has changed the game: an open-source model that rivals GPT-4 on many benchmarks, running locally via Ollama. Here's how to integrate it with OpenClaw to reduce your API costs by over 88%.*

# 1. Why DeepSeek?

## Benchmark Performance

DeepSeek-R1 and DeepSeek-V3 rank among the best open-source models of 2025. On reasoning, coding, and synthesis tasks, they directly rival GPT-4o and Claude Sonnet while being free for local use.

- MMLU: 88.5% (DeepSeek-V3) vs 88.7% (GPT-4o) — virtually equivalent
- HumanEval (code): 89.1% — above average for frontier models
- Context: up to 128k tokens on DeepSeek-V3
- GPU inference speed: 40-80 tokens/sec on RTX 3090/4090

## Open Source and Ollama-Compatible

DeepSeek is released under the MIT license. Weights are publicly available on Hugging Face. Ollama — the most popular local runtime — distributes quantized versions (Q4, Q5, Q8) ready to use.

```
ollama pull deepseek-r1:7b
ollama pull deepseek-r1:14b
ollama pull deepseek-v3
```

■ *The 7b version runs comfortably on a machine with 16 GB RAM (CPU). The 14b version requires a GPU with 16 GB VRAM for acceptable latency.*

> *“Open source does not mean quality compromise. In 2025, the gap with proprietary models has closed for the majority of enterprise use cases.”*

# 2. OpenClaw + Ollama + DeepSeek Architecture

## Installation

The integration relies on three components: Ollama as the local inference runtime, DeepSeek as the model, and OpenClaw with a provider configured to point to Ollama.

```
# 1. Install Ollama
curl -fsSL https://ollama.ai/install.sh | sh

# 2. Download DeepSeek
ollama pull deepseek-r1:14b

# 3. Configure OpenClaw (config.yml)
default_model: deepseek-r1:14b
providers:
ollama:
base_url: http://localhost:11434
models: [deepseek-r1:7b, deepseek-r1:14b, deepseek-v3]
```

### Observed Latencies

Latencies vary by hardware. Here are BOTUM field measurements on common use cases (time to first token):

| Hardware | Model | First token latency | Tokens/sec |
|---|---|---|---|
| RTX 4090 (24 GB) | DeepSeek-R1 14b | 0.8 sec | 65 t/s |
| RTX 3090 (24 GB) | DeepSeek-R1 14b | 1.2 sec | 42 t/s |
| CPU 32 GB RAM | DeepSeek-R1 7b | 4.5 sec | 12 t/s |
| Cloud (OpenAI API) | GPT-4o | 0.6 sec | ~80 t/s |

■ *For interactive tasks (fast response to a Telegram user), 4.5 sec CPU latency remains acceptable. For batch workflows, it is negligible.*

## 3. Hybrid Routing Strategy

The key is not to replace the cloud API with local — it's to route intelligently based on task type. OpenClaw supports multi-provider: each call can target a different model.

### 5 Decision Criteria

**Reasoning complexity:** Complex logic, advanced code, multi-step analysis → cloud API (GPT-4o, Claude Sonnet)

**Data sensitivity:** Confidential data, PII, business secrets → DeepSeek local (nothing leaves your infrastructure)

**Task volume:** Batch processing, log summarization, repetitive transformations → DeepSeek local

**Required latency:** Real-time interactive response → cloud. Background workflow → local

**Available budget:** Near month-end, shift more tasks to local to stay within budget

### Routing Configuration in OpenClaw

```
# Skill with specific model
# In system prompt or skill config:
model: deepseek-r1:14b # force local for this skill

# Via chat (one-off override):
/model deepseek-r1:14b
```

> *“Hybrid routing is a posture, not a fixed configuration. Review it monthly based on your usage evolution and available models.”*

## 4. Real Savings and ROI Calculation

### Baseline: Cloud API Only

Before DeepSeek integration, typical BOTUM usage on OpenClaw generated approximately:

- ~2 million tokens/month (input + output combined)
- Mix GPT-4o (~70%) + Claude Sonnet (~30%)

- Average monthly cost: ~$105 USD/month

### After Hybrid DeepSeek Routing

By shifting ~80% of tasks to DeepSeek local (repetitive tasks, summaries, web search, digests):

- Local tasks (DeepSeek): ~80% of volume → $0
- Remaining cloud tasks: ~20% of volume → ~$12.60/month
- Total reduction: -88% on API costs
- Monthly savings: ~$92.40/month

### GPU Break-Even

If you don't have a GPU and are considering purchasing one:

- Used RTX 3090: ~$600-800 CAD
- Additional electricity: ~$8-12/month
- Break-even: approximately 5 months with realized savings
- After break-even: net savings of ~$80-85/month indefinitely

■ *These figures are based on BOTUM production usage (March 2026). Your ROI will vary based on volume and task mix. On CPU alone (no GPU), the ROI is immediate — just slower.*

## 5. Known Limitations

### Hardware Dependency

Without a dedicated GPU, CPU performance remains limited. DeepSeek-R1 14b on CPU takes 4-8 seconds for the first token — acceptable for batch jobs, challenging for interactive use.

### Quality on Complex Tasks

On complex multi-step reasoning, chained tool calls, or advanced code generation, frontier models (GPT-4o, Claude Sonnet) maintain a measurable advantage. Do not migrate these tasks to local.

### Long Context

DeepSeek-V3 theoretically supports 128k tokens, but in practice on Ollama, quality degradation appears after ~32k tokens depending on hardware and the quantized version used.

### Maintenance

Unlike the cloud API, local requires maintenance: Ollama updates, new model versions, disk space management (models are 8-40 GB depending on size).

> *“Local is not 'set and forget.' It's a tool that requires an operator. If you don't have time to maintain it, keep cloud for critical tasks.”*

## 6. BOTUM Field Feedback

### What We Use in Production

- DeepSeek-R1 14b for: email digests, log summaries, web search, initial drafts
- Claude Sonnet for: complex reasoning, production code, important decisions

- GPT-4o for: multimodal tasks, blog content generation
- Current mix: ~75% local / 25% cloud

### What We Would Do Again

- Start with DeepSeek-R1 7b to test without a GPU — results are already impressive
- Set up explicit routing from day one — no chain migration to do later
- Monitor tokens per provider from day 1 — impossible to optimize without metrics
- Keep a list of 'non-negotiable cloud' tasks — prevents quality regressions

### What We Would Not Do Again

- Migrate all tasks at once — test progressively, section by section
- Ignore CPU latency for interactive tasks — user experience suffers
- Underestimate disk management — plan for 100 GB minimum for multiple models

## 7. DeepSeek + OpenClaw Deployment Checklist

■ Ollama installed and running (`ollama serve` starts at boot)

■ DeepSeek-R1 14b (or 7b for your hardware) downloaded and tested

■ OpenClaw configured with Ollama provider (base_url localhost:11434)

■ Sanity test: `/model deepseek-r1:14b` then a simple question

■ Hybrid routing defined: list of local vs cloud tasks

■ Token monitoring per provider in place

■ Monthly cloud API budget redefined (target -80% minimum)

■ Cloud budget overage alerts configured

■ Ollama update procedure documented

■ Disk space management: cleanup cron if needed

■ Regression tests: verify quality remains acceptable for migrated tasks

■ Internal documentation: who routes what, why, since when

## Conclusion

DeepSeek + Ollama + OpenClaw forms a powerful combination for teams that want to regain control of their AI costs without sacrificing quality. Hybrid routing is the key: not all local, not all cloud, but the right model for the right task.

With -88% on API costs and a 5-month GPU break-even, the investment pays off quickly. And for sensitive data, the benefit of not sending data outside your infrastructure is invaluable.

Post 8: OpenClaw and multi-agent workflows — how to orchestrate multiple specialized agents.

**Full article:** blog.botum.ca/openclaw-deepseek-local-llm-open-source-api-costs

Website: www.botum.ca • contact@botum.ca