# Best Local LLMs for Summarization

February 10, 2026 · by Mark Bartlett

Download this guide as PDF

> **Quick Answer:** For summarizing documents locally, Qwen 2.5 14B on 16GB VRAM is the sweet spot — excellent instruction following, respects length constraints, handles 200 pages in a single pass with 128K context. On 8GB, Qwen 2.5 7B or Gemma 2 9B. On 24GB, Qwen 2.5 32B for best quality or Command R 35B for cited summaries. For speed, Gemma 2 9B and Mistral 7B are fastest. For meeting transcripts, Qwen 2.5 14B handles conversational text well. For technical docs, Qwen 2.5 Coder preserves code context. Always specify output length and format in your prompt — 'Summarize in 5 bullet points' beats 'Summarize this' every time.

📚 **Related:** Context Length Explained · VRAM Requirements · Qwen Models Guide · Best LLMs for Data Analysis

You paste a 40-page report into your local model and ask for a summary. You get back eight paragraphs when you asked for three sentences. Half the key findings are missing. There's a statistic on page 2 that doesn't appear anywhere in the original document.

Summarization sounds simple. It's not. A good summarization model needs to follow length instructions precisely, preserve facts without hallucinating new ones, and handle long documents without losing information from the middle. Most local models can do at least one of these well. Few do all three.

Here's which models actually work for summarization, organized by use case, with the prompting techniques that make the difference.

---

## What Makes a Good Summarizer

Four things matter, in order of importance:

**Instruction following.** "Summarize in 3 bullet points" should produce exactly 3 bullet points — not 7, not a paragraph, not 3 bullet points followed by a disclaimer. Qwen 2.5 and Gemma 3 are the strongest instruction followers in the open model space. Smaller models and older architectures tend to ignore length and format constraints.

**Faithfulness.** The model should compress, not invent. A hallucinated statistic in a financial summary or a fabricated clause in a legal summary is worse than no summary at all. Larger models hallucinate less. Lower temperature (0.1-0.3) helps. Explicit prompting — “Only include information stated in the document” — helps more.

**Context window.** You can’t summarize what the model can’t see. At roughly 450 tokens per page:

| Model | Context Window | Usable Context (~70%) | Pages in One Pass |
|---|---|---|---|
| Llama 3.1/3.2 | 128K | ~90K | ~200 pages |
| Qwen 2.5 | 128K | ~90K | ~200 pages |
| Qwen 3 | 128K | ~90K | ~200 pages |
| Gemma 3 | 128K | ~90K | ~200 pages |
| Mistral 7B v0.3 | 32K | ~22K | ~50 pages |
| Gemma 2 9B | 8K | ~5.6K | ~12 pages |

Why 70% usable? System prompts, your instructions, and the output itself consume context. Filling the full window degrades quality — 70% is the practical ceiling. On 8GB VRAM, the KV cache limits you further: a 7B model with 128K advertised context realistically handles ~32-64K tokens of input before VRAM runs out.

**Compression ratio.** Asking a model to compress a 50-page document into 3 sentences is a different task than compressing it into 3 pages. Bigger models handle extreme compression better — they identify the true key points rather than producing vague generalities. 7B models are fine for moderate compression (10:1 or less). For 50:1 or higher, use 14B+.

## Quality Comparison

Tested across general summarization tasks (news articles, reports, meeting notes):

| Model | Accuracy | Length Control | Speed (RTX 3090) | Context | Best For |
|---|---|---|---|---|---|
| **Qwen 2.5 32B** | Excellent | Excellent | ~30 tok/s | 128K | Best overall quality |
| **Qwen 2.5 14B** | Very good | Excellent | ~56 tok/s | 128K | Sweet spot |

| Model | Accuracy | Length Control | Speed (RTX 3090) | Context | Best For |
|---|---|---|---|---|---|
| **Qwen 3 8B** | Very good | Very good | ~60 tok/s | 128K | Matches 2.5 14B quality |
| **Command R 35B** | Very good | Good | ~25 tok/s | 128K | Cited summaries |
| **Gemma 3 27B** | Very good | Very good | ~35 tok/s | 128K | Long docs on 24GB |
| **Llama 3.3 70B** | Excellent | Good | ~15 tok/s | 128K | Max quality (needs 2x GPU) |
| **Llama 3.1 8B** | Good | Moderate | ~50 tok/s | 128K | Budget all-rounder |
| **Gemma 2 9B** | Good | Very good | ~65 tok/s | 8K | Speed priority |
| **Mistral 7B** | Good | Moderate | ~70 tok/s | 32K | Fast, short docs |
| **Gemma 3 4B** | Decent | Good | ~80 tok/s | 128K | Minimal VRAM |

Qwen 2.5’s instruction following is the differentiator. When you say “3 bullet points, 50 words each,” Qwen 2.5 14B delivers. Llama 3.1 8B gives you 5 bullet points of varying length. Mistral 7B gives you a paragraph. For summarization, following the format matters as much as getting the content right.

## Top Picks by Use Case

### General Summarization (Reports, Articles, Briefs)

**Best:** Qwen 2.5 14B (16GB VRAM) or Qwen 3 8B (8GB VRAM)

These handle the widest range of content well. Strong instruction following means your formatting requests are respected. 128K context handles most documents in a single pass. Good faithfulness — they compress rather than invent.

```
ollama pull qwen2.5:14b

# Summarize a report
pdftotext report.pdf - | ollama run qwen2.5:14b "Summarize this report in 5 bullet points. Inclu
```

### Long Document Summarization (100+ Pages)

**Best:** Gemma 3 27B on 24GB VRAM, or Qwen 2.5 7B with chunking on 8GB

For single-pass summarization of long documents, you need a model with 128K context AND enough VRAM for the KV cache. Gemma 3 27B at ~14GB Q4 on a 24GB card leaves 10GB for the KV cache — enough for the full 128K window. That's genuine 200-page single-pass summarization.

On lower VRAM, use chunking (covered below). A 7B model that reads 30-page chunks and summarizes each one produces better coverage than trying to force the entire document into an undersized context window.

### Meeting Notes and Transcripts

**Best:** Qwen 2.5 14B or Qwen 3 8B

Meeting transcripts are messy — partial sentences, interruptions, tangents, repeated points. The model needs to extract decisions and action items from conversational noise. Qwen 2.5 handles this well because it can follow specific extraction prompts:

```
cat meeting_transcript.txt | ollama run qwen2.5:14b "Extract from this meeting transcript:
1. Decisions made
2. Action items (with who is responsible)
3. Open questions
4. Key deadlines mentioned

Use bullet points. Only include items explicitly discussed."
```

Gemma 2 9B is a good alternative when speed matters — fast enough for real-time meeting recap if you're processing transcripts as they come in.

### Technical Documentation

**Best:** Qwen 2.5 Coder 14B or 32B

When summarizing technical docs that include code, architecture decisions, or API specifications, a coding-focused model preserves technical accuracy better. Qwen 2.5 Coder understands code context and won't mangle function names, config values, or technical terms the way a general model sometimes does.

DeepSeek Coder V2 is another option — strong at preserving technical details, though less controllable on output format.

```
ollama pull qwen2.5-coder:14b

cat api_docs.md | ollama run qwen2.5-coder:14b "Summarize this API documentation. List each endp
```

### News and Article Summarization

**Best:** Any 7B+ model works. Mistral 7B for speed.

News articles are short, well-structured, and don't require deep reasoning. Even small models handle them well. If you're processing a feed of articles and want quick summaries, Mistral 7B at ~70 tok/s on a 3090 chews through them fast. Gemma 2 9B is similarly quick.

For batch processing (summarizing 50+ articles), speed matters more than the marginal quality difference between a 7B and 14B model:

```
# Batch summarize articles
for f in articles/*.txt; do
  echo "=== $(basename $f) ===" >> summaries.txt
  cat "$f" | ollama run mistral:7b "Summarize this article in 2 sentences:" >> summaries.txt
  echo "" >> summaries.txt
done
```

---

## VRAM Requirements

Summarization VRAM needs are identical to general inference, plus extra headroom for long context. The KV cache grows with input length — budget 2-4GB extra for 32K+ contexts.

| Model | VRAM (Q4_K_M) | + 32K Context | + 128K Context | Min GPU |
|---|---|---|---|---|
| **Gemma 3 4B** | ~3 GB | ~4 GB | ~6 GB | Any 8GB card |
| **Qwen 2.5 7B** | ~5 GB | ~6.5 GB | ~10 GB | RTX 3060 8GB (32K), 12GB (128K) |

| Model | VRAM (Q4_K_M) | + 32K Context | + 128K Context | Min GPU |
|---|---|---|---|---|
| **Qwen 3 8B** | ~5 GB | ~7 GB | ~11 GB | RTX 3060 12GB |
| **Gemma 2 9B** | ~6 GB | N/A (8K max) | N/A | RTX 3060 8GB |
| **Qwen 2.5 14B** | ~9 GB | ~11 GB | ~16 GB | RTX 4060 Ti 16GB |
| **Gemma 3 27B** | ~14 GB | ~17 GB | ~24 GB | RTX 3090/4090 |
| **Qwen 2.5 32B** | ~20 GB | ~23 GB | Doesn't fit | RTX 3090 (32K limit) |
| **Command R 35B** | ~19 GB | ~22 GB | Doesn't fit | RTX 3090 (32K limit) |

The sweet spots:

- **8GB VRAM:** Qwen 2.5 7B with ~32K usable context (~50 pages)
- **12GB VRAM:** Qwen 3 8B with ~64K usable context (~130 pages)
- **16GB VRAM:** Qwen 2.5 14B with full 128K context (~200 pages) — **best value**
- **24GB VRAM:** Gemma 3 27B with full 128K, or Qwen 2.5 32B with ~32K
- **Speed priority:** Gemma 2 9B or Mistral 7B (8K/32K context but fast)

→ Use our Planning Tool to check exact VRAM for your setup.

## Chunking for Long Documents

When your document exceeds the context window, don't try to force it. Chunk it.

### Map-Reduce (Best for Most Cases)

Split the document, summarize each chunk independently, then summarize the summaries.

```
Document → [Chunk 1] [Chunk 2] [Chunk 3] ... [Chunk N]
              ↓          ↓          ↓              ↓
          [Summary 1] [Summary 2] [Summary 3] ... [Summary N]
              ↓          ↓          ↓              ↓
          ────────── Final Summary Pass ──────────
                           ↓
                   [Final Summary]
```

Each chunk is independent — you can process them in parallel. The tradeoff: connections between distant sections can be missed.

```
# Split document into ~300-line chunks (~2000 tokens each)
split -l 300 large_document.txt chunk_

# Summarize each chunk
for f in chunk_*; do
  cat "$f" | ollama run qwen2.5:7b "Summarize this section concisely:" > "summary_$f"
done

# Produce final summary from chunk summaries
cat summary_chunk_* | ollama run qwen2.5:7b "These are summaries of sections from one document.
```

### Refine (Best for Narrative Documents)

Process sequentially — summarize chunk 1, then update the summary with chunk 2, then with chunk 3. Preserves narrative flow but can't parallelize and may "forget" early details.

```
Chunk 1 → Summary v1
Summary v1 + Chunk 2 → Summary v2
Summary v2 + Chunk 3 → Summary v3
... → Final Summary
```

Best for books, annual reports, meeting transcripts — anything with a chronological structure.

### Hierarchical (Best for 500+ Pages)

Map-reduce with multiple levels. Summarize chunks into section summaries, sections into chapter summaries, chapters into the final summary. Scales to any length.

## Prompting Tips

The difference between a useless summary and a good one is often the prompt, not the model.

### Specify Output Length

Bad: "Summarize this document." Good: "Summarize this document in exactly 5 bullet points, each 1-2 sentences."

Be explicit. If you want 3 sentences, say 3 sentences. If you want 500 words, say 500 words. Models that ignore this (looking at you, Llama 8B) need the instruction repeated in the system prompt and the user message.

### Specify What to Focus On

Bad: “Summarize this meeting transcript.” Good: “Summarize this meeting transcript. Focus on: decisions made, action items with owners, and deadlines. Ignore small talk and tangents.”

The model doesn’t know what matters to you. A legal summary needs different focus than a project management summary of the same document.

### Specify Output Format

```
Summarize this report using this structure:

## Key Findings
- (3-5 bullet points)

## Recommendations
- (numbered list)

## Data Points
- (any specific numbers, dates, or metrics mentioned)

## Open Questions
- (unresolved issues or items needing follow-up)
```

Giving the model a template to fill produces dramatically better results than open-ended summarization. This works with any model 7B and up.

### Include “Only” Constraints

```
Summarize the following document.
- Only include information explicitly stated in the document
- Do not add analysis, interpretation, or external knowledge
- Do not fabricate quotes or statistics
- If something is unclear in the original, note it as unclear
```

These constraints reduce hallucination significantly, especially at 7-14B scale where models are more prone to filling gaps with plausible-sounding fabrications.

### Set Temperature Low

For summarization, set temperature to 0.1-0.3. You want accuracy, not creativity. Higher temperatures increase the chance of hallucinated details and wandering output.

```
# Ollama: set low temperature via API
curl http://localhost:11434/api/generate -d '{
  "model": "qwen2.5:14b",
  "prompt": "Summarize in 3 bullet points: ...",
  "options": {"temperature": 0.2}
}'
```

---

## Tools for Local Summarization

### Ollama + Open WebUI (Easiest)

Open WebUI supports drag-and-drop file upload. Drop a PDF, type “summarize this document,” and it handles text extraction, chunking, and summarization. No coding required.

```
ollama pull qwen2.5:14b

docker run -d -p 3000:8080 \
  --add-host=host.docker.internal:host-gateway \
  -v open-webui:/app/backend/data \
  --name open-webui \
  ghcr.io/open-webui/open-webui:main
```

**Critical:** Increase the context size in Open WebUI’s admin panel (Models > Advanced Parameters). The default 2,048 tokens is far too small for summarization. Set it to 16,000+ — or 32,000-128,000 if your VRAM supports it.

**Tip:** Click the uploaded file and select “Using Full Document” instead of chunked retrieval when you want to summarize the entire document, not search it.

### Ollama CLI (Fastest for Single Files)

```
# Text file
cat meeting_notes.txt | ollama run qwen2.5:7b "Summarize in bullet points:"

# PDF (requires pdftotext from poppler-utils)
pdftotext report.pdf - | ollama run qwen2.5:14b "Executive summary in 3 paragraphs:"

# Focused extraction
cat contract.txt | ollama run qwen2.5:14b "Extract: parties, key obligations, payment terms, te
```

### LM Studio

Paste text into the chat. LM Studio doesn't have file upload, but you can adjust temperature and context size per session — useful for dialing in summarization quality.

---

## Common Failures and Fixes

### Hallucinated Details

The model adds facts not in the original. A fabricated statistic in a financial summary causes real harm.

**Fix:** "Only include information explicitly stated in the document." Larger models (14B+) hallucinate less. Temperature 0.1-0.3. Verify any numbers in the summary against the source.

### Lost in the Middle

Key information from the middle of long documents gets dropped. Models attend more to the beginning and end of context.

**Fix:** Use chunked summarization even when the document fits in context. Chunks of 20-30 pages give more uniform coverage. Or use two passes: first pass summarizes, second pass checks for gaps.

### Over-Compression

“The document discusses various aspects of the project.” That describes every document ever written.

**Fix:** Give the model a structure to fill: “Include: specific findings, numerical data, names, dates, and action items.” A template prompt prevents generic output.

### Ignoring Length Instructions

You asked for 3 sentences and got 3 paragraphs.

**Fix:** Use Qwen 2.5 — it’s the strongest instruction follower for length control. For stubborn models, add: “Your response MUST be exactly 3 sentences. Count them.” Or use structured output to constrain the format.

## The Privacy Argument

The documents worth summarizing are exactly the ones you shouldn’t upload to a cloud API.

| Use Case | Why Local |
|---|---|
| **Legal briefs** | Attorney-client privilege. Cloud upload may waive privilege. |
| **Medical records** | HIPAA. Patient data can’t hit third-party servers without compliance work. |
| **Financial docs** | SEC/FINRA requirements. Client data has strict handling rules. |
| **Proprietary code** | Trade secrets. See our local coding models guide. |
| **Internal memos** | M&A, investigations, HR matters. Cloud APIs create discovery risk. |

A local 14B model processes these documents in minutes, on hardware you control, with no data leaving your machine. The summaries aren’t GPT-4 quality. They’re private, fast, and good enough — which, for sensitive documents, is what matters.

## The Recommendation

| Your Setup | Model | Context Strategy |
|---|---|---|
| 8GB VRAM, short docs | Qwen 2.5 7B (Q4) | Single pass, ~50 pages |

| Your Setup | Model | Context Strategy |
|---|---|---|
| 8GB VRAM, long docs | Qwen 2.5 7B (Q4) | Map-reduce chunking |
| 8GB VRAM, speed priority | Gemma 2 9B (Q4) | Single pass, ~12 pages |
| 12GB VRAM | Qwen 3 8B (Q4) | Single pass, ~130 pages |
| 16GB VRAM | Qwen 2.5 14B (Q4) | Single pass, ~200 pages **(sweet spot)** |
| 24GB, need citations | Command R 35B (Q4) | Single pass, inline citations |
| 24GB, long docs | Gemma 3 27B (Q4) | Single pass, full 128K |
| 24GB, best quality | Qwen 2.5 32B (Q4) | Single pass, ~50 pages (32K usable) |
| Dual 24GB GPUs | Llama 3.3 70B (Q4) | Best quality, single pass |
| Any VRAM, 500+ pages | Any of the above | Hierarchical map-reduce |

Start with `ollama pull qwen2.5:14b` if you have 16GB VRAM. It's the summarization sweet spot — strong instruction following, 128K context, fast enough at ~56 tok/s on a 3090. Upload a document to Open WebUI, specify your format, and check if the quality meets your needs. If you need more, move up to 32B or 27B. If you need less, drop to 7B.

The best summarization model is the largest one that fits your VRAM with room for context. Context window matters more than raw model size — a 7B model that reads the whole document produces better summaries than a 70B model that only sees the first 15 pages.

---

## Related Guides

- Context Length Explained
- VRAM Requirements for Local LLMs
- Qwen Models Guide
- Best Local LLMs for Data Analysis
- Local AI Privacy Guide
- Open WebUI Setup Guide
- Best Local LLMs for RAG
- Structured Output from Local LLMs

Get notified when we publish new guides.

Subscribe — free, no spam

Source: https://insiderllm.com/guides/best-local-llms-summarization/

Free guides for running AI locally