Embedding Search for GitLab with llama.cpp

I deployed llama.cpp with a Qwen embedding model, picked a simple serving config, and built Wissen Lab to add embedding-based search to GitLab.

---

1. Ollama or Llama.cpp?

I'm new to both Ollama and llama.cpp.

Thanks to ajax for pushing me to look deeper. One thing that caught my attention is that llama.cpp exposes a lot of knobs, so you can tune it to your workload instead of treating it like a black box.

2. Llama Bench

After compiling llama.cpp, the first thing I did was run `llama-bench`. Here are my results:

```
root@Skyworks-GPU ~/llama-build# ./llama-bench -m ../models/Qwen3-
Embedding-8B-Q6_K.gguf -embd 1 -p 8,16,32,64,128,256,512
| model                          |       size |     params | backend    | ngl
|       embd |            test |                  t/s |
| ------------------------------ | ---------: | ---------: | ---------- | --:
| ---------: | -------------: | -------------------: |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
|          1 |             pp8 |         23.67 ± 2.45 |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
|          1 |            pp16 |         41.47 ± 0.53 |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
|          1 |            pp32 |         78.82 ± 3.32 |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
|          1 |            pp64 |        134.01 ± 0.86 |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
|          1 |           pp128 |       289.57 ± 40.16 |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
|          1 |           pp256 |        477.41 ± 5.33 |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
|          1 |           pp512 |       592.39 ± 24.56 |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
|          1 |           tg128 |         19.13 ± 8.08 |

build: c5a778891 (8233)
```

```
root@Skyworks-GPU ~/llama-build# ./llama-bench -m ../models/Qwen3-
Embedding-8B-Q6_K.gguf -embd 1 -p 512 -sm none,layer
| model                          |       size |     params | backend    | ngl
|    sm |       embd |            test |                  t/s |
| ------------------------------ | ---------: | ---------: | ---------- | --:
| ----: | ---------: | -------------: | -------------------: |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
|  none |          1 |           pp512 |        804.24 ± 2.11 |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
|  none |          1 |           tg128 |         35.36 ± 0.07 |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
| layer |          1 |           pp512 |       591.46 ± 26.78 |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
| layer |          1 |           tg128 |        11.65 ± 10.84 |

build: c5a778891 (8233)

root@Skyworks-GPU ~/llama-build# ./llama-bench -m ../models/Qwen3-
Embedding-8B-Q6_K.gguf -embd 1  -b 8192 -sm none -ub 8192,4096
| model                          |       size |     params | backend    | ngl
| n_batch | n_ubatch |    sm |       embd |            test |
t/s |
| ------------------------------ | ---------: | ---------: | ---------- | --:
| ------: | -------: | ----: | ---------: | --------------: |
-------------------: |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
|    8192 |     8192 |  none |          1 |           pp512 |        806.29 ±
4.52 |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
|    8192 |     8192 |  none |          1 |           tg128 |         35.31 ±
0.08 |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
|    8192 |     4096 |  none |          1 |           pp512 |        799.43 ±
1.03 |
| qwen3 8B Q6_K                  |   5.78 GiB |     7.57 B | CUDA       |  99
|    8192 |     4096 |  none |          1 |           tg128 |         35.18 ±
0.04 |
```

What I took away from this:

- For embeddings, I care most about prompt processing throughput (`pp*`). Token generation (`tg*`) is not the bottleneck for my workload.

- `--split-mode none` is consistently faster for me than splitting by layer.
- With `-b 8192`, changing `--ubatch-size` between 8192 and 4096 did not move the needle much, so I keep it simple and use 8192.

For the model, I went with a larger embedding model because my inputs are often messy and long: commit diffs, large merge requests, and long issue comments. I want good recall even if that means larger vectors and more storage.

3. Serve

The final command I chose is:

```
./llama-server --model ../models/Qwen3-Embedding-8B-Q6_K.gguf \
  --embedding --host 127.0.0.1 --port 8081 \
  --split-mode none --main-gpu 0 --batch-size 8192 \
  --ctx-size 8192 --ubatch-size 8192 --pooling last --parallel 1
```

This uses only one GPU, by design. My plan is to run one process per GPU and load-balance across them. For example:

```
upstream qwen {
  least_conn;
  server 127.0.0.1:8081;
  server 127.0.0.1:8082;
  server 127.0.0.1:8083;
  server 127.0.0.1:8084;
  server 127.0.0.1:8085;
  server 127.0.0.1:8086;
  server 127.0.0.1:8087;
  server 127.0.0.1:8088;
}
server {
  listen 8080;
  location / {
    proxy_pass http://qwen;
  }
}
```

4. Wissen Lab

I've wanted embedding-based search for GitLab for a long time. With help from Codex (5.4 high), I managed to finish Wissen Lab in about 12 hours. So far, it seems to work:

```
kubectl logs -n aire wissen-lab-7cf544dcd7-z5p7d -f
Defaulted container "wissen-lab" out of: wissen-lab, wissen-front
INFO:     Started server process [1]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://:8000 (Press CTRL+C to quit)
2026-03-07 22:12:14,357 INFO llama_server embedding success repo_id=2
source_kind=issue source_key=3 chunk_index=0 locator_id=None
content_length=581 failure_count=0
2026-03-07 22:12:15,414 INFO llama_server embedding success repo_id=2
source_kind=issue source_key=3 chunk_index=1 locator_id=52714
content_length=33 failure_count=0
2026-03-07 22:12:16,663 INFO llama_server embedding success repo_id=2
source_kind=issue source_key=3 chunk_index=2 locator_id=52715
content_length=124 failure_count=0
```

It is also reasonably resilient. I asked Codex to add an exponential backoff policy so bursts of messages do not overwhelm the embedding server while the queue is draining.

The only thing I'm concerned about is whether a single Postgres instance with pgvector support can handle tens of thousands of vectors.

5. Follow-up

I realised the in-memory task queue the LLM implemented is not the best choice here. The app already has a Postgres connection, so why not commit tasks to Postgres and retrieve them from there? And of course, Codex (5.4 xhigh) can handle that too.