Embeddings

- Generate embeddings
- Generate a batch of embeddings
- Tips

Embeddings turn text into numeric vectors you can store in a vector database, search with cosine similarity, or use in RAG pipelines. The vector length depends on the model (typically 384–1024 dimensions).

Generate embeddings

Generate embeddings directly from the command line:

```shell
ollama run embeddinggemma "Hello world"
```

You can also pipe text to generate embeddings:

```shell
echo "Hello world" | ollama run embeddinggemma
```

Output is a JSON array.

```shell curl -X POST http://localhost:11434/api/embed \ -H "Content-Type: application/json" \ -d '{ "model": "embeddinggemma", "input": "The quick brown fox jumps over the lazy dog." }' ```
```python import ollama

```
single = ollama.embed(
  model='embeddinggemma',
  input='The quick brown fox jumps over the lazy dog.'
)
print(len(single['embeddings'][0]))  # vector length
```

```
```

```javascript import ollama from 'ollama'
```
const single = await ollama.embed({
  model: 'embeddinggemma',
  input: 'The quick brown fox jumps over the lazy dog.',
})
console.log(single.embeddings[0].length) // vector length
```

The `/api/embed` endpoint returns L2-normalized (unit-length) vectors.

# Generate a batch of embeddings

Pass an array of strings to `input`.

```shell curl -X POST http://localhost:11434/api/embed \ -H "Content-Type: application/json" \ -d '{ "model": "embeddinggemma", "input": [ "First sentence", "Second sentence", "Third sentence" ] }' ``` ```python import ollama
```
batch = ollama.embed(
  model='embeddinggemma',
  input=[
    'The quick brown fox jumps over the lazy dog.',
    'The five boxing wizards jump quickly.',
    'Jackdaws love my big sphinx of quartz.',
  ]
)
print(len(batch['embeddings'])) # number of vectors
```

```javascript import ollama from 'ollama'
```
const batch = await ollama.embed({
  model: 'embeddinggemma',
  input: [
    'The quick brown fox jumps over the lazy dog.',
    'The five boxing wizards jump quickly.',
    'Jackdaws love my big sphinx of quartz.',
  ],
})
```
```

```
console.log(batch.embeddings.length) // number of vectors
```

Tips

- Use cosine similarity for most semantic search use cases.
- Use the same embedding model for both indexing and querying.

□□□ #2
□ □□□ □□□ 2026-10-31 13:16:41 UTC
□ □□□ □□□ 2026-10-31 13:16:42 UTC