# Text Embedding Efficiency

---

## Text Embedding Efficiency

Compared to last week, efficiency metrics on **Apple M2, M3 Pro, and M4 chips** have been added.

- **M3 Pro** and **M4** were tested on both **CPU** and **MPS** (Apple's GPU-like accelerator).
- **MPS is slightly faster than CPU**.
- **M3 Pro and M4 are significantly faster than M2**, though still **slower than discrete GPUs** like the A6000.
- Embedding times on M-series chips vary significantly due to interference from other running applications. All benchmarking was done with all other apps closed.

---

## Baseline Embedding Model Selection

### 🛰️ Remote Embedding

**Model:** `inf-retriever-v1` (7B)
**MRR:** 0.7738
**Hit@5:** 0.9167

- **GPU (A6000):** 26.22s
- **CPU (AMD EPYC 9654):** 167.84s

---

### 💻 Local Embedding

**Model:** `inf-retriever-v1-1.5b` (1.5B)
**MRR:** 0.7417
**Hit@5:** 0.8800

- **GPU (A6000):** 6.52s
- **CPU (AMD EPYC 9654):** 32.08s
- **Apple M2:** 116.42s
- **Apple M3 Pro:** 33.05s
- **Apple M4:** 21.49s

- **Intel:** Not yet tested

---

**Model:** `gte-multilingual-base` (305M)
**MRR:** 0.6134
**Hit@5:** 0.76

- **GPU (A6000):** 1.22s
- **CPU (AMD EPYC 9654):** 8.26s
- **Apple M2:** 14.63s
- **Apple M3 Pro:** 8.52s
- **Apple M4:** 4.90s

---

## 🧠 Embedding Configuration

| Metric | Description |
|---|---|
| **Embedding Model** | Transformer / Sentence Transformer |
| **Embedding Dimensions** | e.g. 3584, 1536, etc. |
| **Max Token Limit** | Typically 32768 |
| **Model Parameters** | From 305M to 7B |
| **Test Dataset** | Custom-built 50k character texts, 10 chunks each |
| **Token Count** | ~32580 tokens |
| **Disk Usage Metrics** | Raw embedding vs. original .txt/.pdf |
| **Milvus DB Usage** | Vector + original text chunks |
| **Hardware** | GPU: A6000, CPU: AMD EPYC 9654, Apple Silicon (M2/M3) |

---

## 🧪 Disk Usage & Compression Example (inf-retriever-v1)

| Metric | Value |
|---|---|
| **Raw Embedding Size** | 0.1367 MB |
| **Compression vs. .txt** | 0.956 |
| **Compression vs. .pdf** | 0.523 |
| **Milvus DB File Size** | 0.2969 MB |
| **DB Compression vs. .txt** | 2.078 |
| **DB Compression vs. .pdf** | 1.134 |
| | |
| | |
| | |

---

Sure! Below is the **complete English translation** of the remaining sections, combined into a structured and comprehensive format:

---

## 🔍 PDF OCR Evaluation

OCR was performed using GPT APIs and compared with MinerU.

| Model/Tool | Evaluation Dataset | Score | Time per Page |
|---|---|---|---|
| MinerU | Custom dataset with 50 PDFs | 8.5 | 1.47s |
| GPT-4o | Same dataset | 9.2 | 43.5s |
| GPT o4mini | Same dataset | 9.18 | 55.4s |

> 📝 GPT models offer **superior OCR accuracy** but **much slower processing** than traditional tools.

---

# 🖼️ Image Embedding

## Embedding Performance Overview

| Model | Parameters | Dataset (500 images + 500 queries) | MRR | Hit@5 | GPU | CPU |
|---|---|---|---|---|---|---|
| BGE-VL-base | 150M | MS-COCO | 0.781 | 0.932 | 4090 | AMD EPYC 965 |
| BGE-VL-large | 428M | — | 0.808 | 0.938 | 4090 | AMD EPYC 965 |
| nomic-embed-multimodal-3B | 3B | — | 0.775 | 0.916 | 4090 | AMD EPYC 965 |
| nomic-embed-multimodal-7B | 7B | — | 0.785 | 0.924 | 4090 | AMD EPYC 965 |
| jina-clip-v1 | 223M | — | 0.753 | 0.918 | 4090 | AMD EPYC 965 |
| clip-vit-base-patch16 | 150M | — | 0.242 | 0.46 | 4090 | AMD EPYC 965 |
| VisRAG-Ret | 3.43B | — | 0.784 | 0.938 | 4090 | AMD EPYC 965 |

---

## Image Embedding Efficiency & Compression

- Embedding time and compression were evaluated on M2, M3 Pro, M4, and server GPUs (4090).
- Embedding size vs. original image size was used to calculate **compression ratio**.

### Example (BGE-VL-large, 1024x1024 images):

| Hardware | Time per Image | Embedding Size | Compression Ratio |
|---|---|---|---|
| GPU (4090) | 0.039s | 0.0586 MB | 0.015 |
| M2 (CPU/MPS) | 0.116s / 0.065s | 0.0898 MB | 0.023 |
| M3 Pro (CPU/MPS) | 0.065s / 0.024s | 0.0898 MB | 0.023 |
| M4 (CPU/MPS) | 0.041s / 0.021s | 0.0898 MB | 0.023 |

---

# 🤖 VQA (Visual Question Answering)

## Model Performance

| Model | Params | Dataset | Score | GPU | CPU | Notes |
|---|---|---|---|---|---|---|
| Qwen2.5-Omni-7B | 7B | VQAv2 | 7.36 | 4090 | AMD EPYC 9654 | |
| Qwen2.5-VL-7B | 7B | | 8.67 | 4090 | AMD EPYC 9654 | Fast, supports large batch sizes |
| InternVL3-14B | 14B | | 8.69 | 4090 | AMD EPYC 9654 | |
| InternVL3-8B | 8B | | 8.63 | 4090 | AMD EPYC 9654 | |
| Ovis2-8B | 8B | | 9.45 | 4090 | — | Best performance but CPU unsupported |
| Ovis2-4B | 4B | | 8.98 | 4090 | — | |

> ✅ **Ovis2-8B** achieves **highest score**, but doesn’ t support CPU inference.
> ✅ **Qwen2.5-VL-7B** is well-balanced in **accuracy and inference efficiency**.

---

## VQA Efficiency (Inference Time & Max Batch)

### Inference Speed (Image Size 1024x1024):

| Model | Max Batch Size (GPU) | Time per Image (1/5/10/50) | GPU Memory Use | Notes |
|---|---|---|---|---|
| Qwen2.5-Omni-7B | 5–10 | 4.62s / 9.44s / 9.83s | Up to OOM | First 1–2 batches have high overhead |
| Qwen2.5-VL-7B | 25+ | 1.82s / 5.85s / 15.05s | Efficient use | Best throughput balance |
| InternVL3-14B | 5–10 | 29.29s – 50s+ | High memory | Large model, high overhead |
| InternVL3-8B | 10–15 | 1.44s – 3.16s | Efficient | Good performance scaling |
| Ovis2-8B | 9 | 2.77s – 4.02s | Not on CPU | Best accuracy, no CPU support |

---

## 🧠 Notes on GPU Inference Time Behavior

1. **Initial batches** incur **extra overhead** (model loading, cache warming).
2. **Small batch sizes underutilize GPU**:
   - GPU works best when **parallel threads (SMs)** are saturated.
   - For 7B models + $1024^2$ images, batch size ≈ 4–8 is optimal for Tensor Core usage.
3. Once fully saturated, inference time increases **linearly** with batch size.

---

## ✅ Summary of Default Model Choices

| Use Case | Model | Params | Score | Notes |
|---|---|---|---|---|
| **Remote** | Ovis2-8B | 8B | 9.45 | Best accuracy, GPU only |
| **Local** | Qwen2.5-VL-7B | 7B | 8.67 | Balanced, efficient on CPU/GPU |
| **Embedding** | BGE-VL-large | 428M | 0.808 MRR / 0.938 Hit@5 | Fast + high precision |