## Migration Checklist: BGE‑small‑en‑v1.5 → BGE‑M3

This checklist helps plan an upgrade from **BGE‑small‑en‑v1.5** (English‑only, 384‑dimensional embeddings) to **BGE‑M3** (multilingual, 1024‑dimensional, multi‑function). Use it to evaluate readiness, execute the migration and validate the resulting system.

### Pre‑Migration Assessment

1. **Clarify objectives**  
   • Identify why you need BGE‑M3: multi‑lingual retrieval, long document support up to 8 192 tokens or hybrid retrieval (dense + sparse + multi‑vector)[[1]](https://huggingface.co/BAAI/bge-m3#:~:text=Specs).  
   • Define success metrics (e.g., recall on your dataset, latency, memory footprint).
2. **Compatibility & limitations**  
   • Confirm that your vector database (Pinecone, Milvus, etc.) supports 1 024‑dimension vectors; some services impose dimensionality limits or cost adjustments.  
   • BGE‑M3 outputs a 1 024‑dimensional dense vector by default[[1]](https://huggingface.co/BAAI/bge-m3#:~:text=Specs); this increases per‑vector storage ~2.7× compared with 384 dimensions.  
   • If you plan to use multi‑vector or sparse retrieval, ensure your infrastructure can store additional vectors and lexical weights[[2]](https://huggingface.co/BAAI/bge-m3#:~:text=,g).  
   • Remove hard‑coded query instructions; BGE‑M3 does not require adding prompts like “Represent this query for retrieval.”
3. **Resource estimation**  
   • Recalculate storage costs: each 1 024‑dimensional vector occupies 4 KB (4 bytes × 1 024) plus metadata; multiply by the number of documents to estimate GB consumption.  
   • Estimate index rebuild time: larger vectors and more retrieval modes require longer computation; consider encoding documents offline.  
   • Budget for increased read units (RU) and storage cost in your vector database.

### Migration Steps

1. **Update dependencies**  
   • Upgrade the FlagEmbedding or Hugging Face packages to include BGE‑M3.  
   • Test GPU/CPU compatibility; BGE‑M3 may require more memory.
2. **Re‑encode content**  
   • Recompute embeddings for all existing documents using BGEM3FlagModel('BAAI/bge-m3')[[3]](https://huggingface.co/BAAI/bge-m3#:~:text=Generate%20Embedding%20for%20text).  
   • For hybrid retrieval, also compute sparse weights and optional ColBERT multi‑vectors as needed[[4]](https://huggingface.co/BAAI/bge-m3#:~:text=).  
   • For each document, store the dense vector, optional sparse weights and/or multiple ColBERT vectors in your database.
3. **Rebuild index**  
   • Create a new index/table with dimension = 1 024 and load the new vectors.  
   • If using multi‑vector retrieval, configure the index to store multiple embeddings per document and adjust your similarity search accordingly.  
   • For sparse retrieval, integrate lexical weights into your search engine or use a hybrid engine such as Vespa or Milvus as recommended[[5]](https://huggingface.co/BAAI/bge-m3#:~:text=,refer%20to%20Vespa%20and%20Milvus).
4. **Update query pipeline**  
   • Remove query instruction text; BGE‑M3 queries are processed directly.  
   • Decide whether to use dense, sparse or hybrid retrieval. For hybrid, combine dense similarity with lexical matching scores.  
   • Adjust k (top‑k) and post‑processing as needed to accommodate potential recall differences.
5. **Testing & validation**  
   • Run benchmark queries in English and other target languages to verify improved recall/precision.  
   • Compare results using dense‑only vs hybrid retrieval; measure latency.  
   • Validate long‑document retrieval by querying documents >512 tokens (up to 8 192 tokens[[1]](https://huggingface.co/BAAI/bge-m3#:~:text=Specs)).  
   • Monitor vector database usage (RUs, storage) to identify cost impacts.
6. **Rollout**  
   • Deploy the new index alongside the old one to allow back‑testing.  
   • Update client applications to use the new embedding dimension and retrieval method.  
   • Gradually route traffic to the new system while monitoring performance and quality.
7. **Post‑Migration clean up**  
   • After validating improved performance, deprecate the old index to reduce costs.  
   • Document the migration process and update operational playbooks.  
   • Consider fine‑tuning BGE‑M3 on domain‑specific data if further gains are needed[[6]](https://arxiv.org/html/2402.03216v3#:~:text=training%20of%20M3,The%20model).

[[1]](https://huggingface.co/BAAI/bge-m3#:~:text=Specs) [[2]](https://huggingface.co/BAAI/bge-m3#:~:text=,g) [[3]](https://huggingface.co/BAAI/bge-m3#:~:text=Generate%20Embedding%20for%20text) [[4]](https://huggingface.co/BAAI/bge-m3#:~:text=) [[5]](https://huggingface.co/BAAI/bge-m3#:~:text=,refer%20to%20Vespa%20and%20Milvus) BAAI/bge-m3 · Hugging Face

<https://huggingface.co/BAAI/bge-m3>

[[6]](https://arxiv.org/html/2402.03216v3#:~:text=training%20of%20M3,The%20model) BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation

<https://arxiv.org/html/2402.03216v3>