

A few vectors for a mixed folder: a pre-registered measurement of pooled directory embeddings

Ali Basheer
Craftpine Inc., Toronto
ali@basheerali.com

October 2026

Abstract

A folder’s metadata is read before its files and holds a few kilobytes. A vector stored there can route a query without opening the folder, and the cheapest is the mean of its files’ embeddings. We ask when the mean is enough and what to store when it is not. Under our main vision-language embedder, the mean does at least as well as a few representatives on GitHub directories that do not mix media, about 95 percent of those eligible. Representatives for mixed folders only are level with the mean over all directories (both post hoc). Where a folder mixes images and texts, the mean loses its minority modality. On Zenodo record folders a held-out minority file finds its folder 0.16 to 0.25 recall@5 less often with the mean than with a few per-modality representatives under two vision-language embedders, and 0.35 to 0.50 less often under two CLIP-family dual towers. There, four representatives per label are not inferior to searching every file, at a 0.02 margin, over all queries and in every cell under all four encoders (post hoc). On queries that a language model wrote as a simulated searcher, the loss under the main embedder is 0.14 (interval 0.02 to 0.28) for pictures in text-heavy folders and 0.42 (0.28 to 0.56) for documents in image-heavy ones. At three per label every folder’s representatives fit in 8 KiB at a further cost of at most 0.012 in any cell. In 2 KiB a per-kind sample of the folder’s files keeps the minority files the mean loses, and on ext4 a summary of 2,048 or 4,000 bytes costs 1.2 blocks per cold directory read. With every creator weighted equally the main embedder’s file-query loss is 0.13 and 0.14, and 0.06 and 0.12 on queries whose names do not give the folder away. Every hypothesis was fixed in a brief before its test. Five briefs are timestamped externally, the rest dated by the working repository’s history. Code, manifests and briefs are released.

1 Introduction

A file system reaches a folder before it reaches the folder’s files. The folder’s own metadata can be read without listing or opening anything inside it, but it holds little. An extended attribute, a named value attached to a file or a directory, holds at most 64 KiB on Linux, and Microsoft’s abstract file system model for the SMB protocols allows one file’s attributes just under 64 KB in total [38, 46]. Common configurations keep them smaller. On ext4 a file’s attributes normally have to fit in its inode and one block of at most 4,096 bytes, and APFS stores a value inside the file-system record up to 3,804 bytes [4, 15, 38]. A vector kept in such a field can be read before a search decides whether to open the folder, and it has to fit in a fixed number of bytes. This is the setting we have in mind. File-level search needs the vector of every file, which a search in this setting has not yet read, so we treat it as the reference and ask how much of it a few kilobytes on the folder keep.

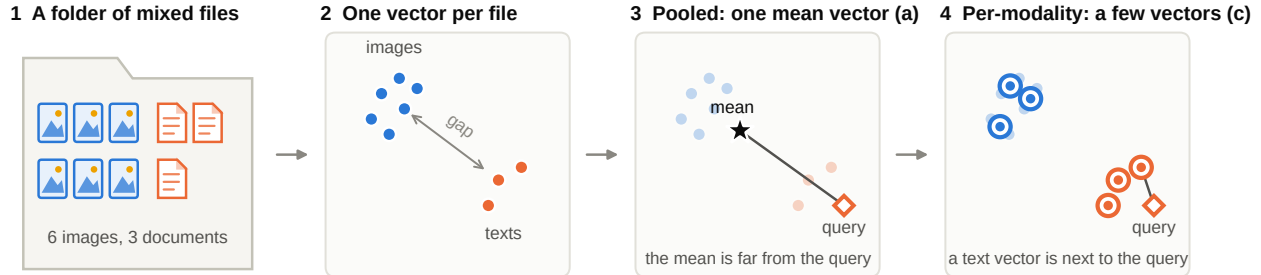


Figure 1: How a folder is represented and scored (a schematic, not data). (1) A folder holds images and documents. (2) Each file becomes a vector in one shared space, and the images and the texts land in two regions. (3) The pooled representation **a** keeps one vector, the mean of the folder’s files, which sits between the two regions. (4) The per-modality representation **c** keeps a few k -means vectors for each modality. A folder’s score for a query is the best cosine between the query and the folder’s vectors. A text query held out from this folder is far from the mean and next to one of the folder’s text vectors.

Folders are becoming units of retrieval again. Retrieval-augmented generation routes a query to a source before it searches inside it, and one router takes the centroid of each source’s document embeddings as an input [14]. Routers over expert embedding models keep centroid pilot embeddings [33]. A context database for agents organises resources, memories and skills as a directory tree whose directories carry a short abstract used for vector retrieval [50]. Vector databases are adding directory-scoped queries [63], and semantic file systems, an idea from 1991 [19], now have embedding-backed descendants [53]. The cheapest representation of a container in an embedding space is the mean of its members’ vectors. It is the dense counterpart of the single bag of words by which early federated search summarised a collection [9, 58].

Containers of files mix media. A record of a field study holds photographs, a scanned permit, a spreadsheet of readings and a README. Several embedders put such files in one space [23, 28, 48, 69]. In contrastive models, image and text embeddings occupy offset regions of that space [36], and in mixed-modality retrieval the offset biases ranking toward the query’s own modality [34]. The authors of the vision-language embedder used for most of this study report the gap dramatically reduced [23], and it still shows one on our files (Section 6). That work studies single items. That one mean misrepresents a group made of two clusters is expected, and textbook material for classifiers [44]. To our knowledge, nobody has measured how much a pooled vector loses on real mixed-media folders, or what a folder’s few kilobytes can keep instead.

We measure it here, and ask when the mean is enough. Most answers below rest on hypotheses whose criteria were fixed before they were tested, and the analyses that a round of review prompted on draws already read are marked post hoc. We answer on three disjoint draws of real Zenodo record folders that followed a pilot draw, on two draws of directories of public GitHub repositories and on whole repository trees. For folders of the last Zenodo draw, a language model also wrote queries as a simulated searcher. Four encoders from three vendors take part, two vision-language embedders and two CLIP-family dual towers. Figure 1 sketches the setup.

1. **When is the mean enough?** Mixed directories are 4 to 5 percent of eligible GitHub directories, and on the others the mean does at least as well as a few representatives under the first of the two vision-language embedders (post hoc). A file system can tell the mixed ones by file type. Keeping representatives only for folders that hold an image and another file keeps most of the minority gain and is level with the mean over all directories at that mix (Section 10, post hoc).

2. **What does the mean lose in a mixed folder?** A pooled centroid loses whichever modality is the folder’s minority, image queries in text-heavy folders and text queries in image-heavy ones. Under a vision-language embedder the loss is 0.17 to 0.25 recall@5 across the three draws. On the last draw it is 0.158 and 0.205 under a second vision-language embedder from another vendor and 0.35 to 0.50 under two CLIP-family dual towers. The hypothesis was fixed before the first of those draws and tested again on the last under each encoder (Section 4). It holds with intervals clustered by the records’ creators and on queries whose folder the file names do not give away, where the image-query loss is about a half to two thirds as large. With every creator weighted equally it is 0.13 and 0.14 under the first embedder, and 0.06 and 0.12 on those clean queries, the most conservative reading of the image-query loss (Section 9). Under the two vision-language embedders, two fifths of it remain when the vectors are fused with a ranking by file names (Section 9, post hoc). On directories of public GitHub repositories, with the owner as the independent unit and at least 100 owners per cell, it is 0.21 and 0.20 under the first embedder on a draw sized for that minimum, confirmed in both cells. With commit subjects as the query it is confirmed on that draw and inconclusive on the first (Section 10).
3. **Does the loss reach a simulated searcher’s queries?** No person wrote queries for this study. A language model that was told nothing about the study wrote them as a simulated searcher, two per folder for 100 folders of the last Zenodo draw, from what a person would see of each folder with names and digits hidden. On those queries the loss under the first embedder is 0.140 recall@5 for pictures in text-heavy folders and 0.420 for documents in image-heavy ones, both with 95 percent intervals above zero and both registered before any query was scored (Section 5). Under E2 the picture cell shows no loss. Per-modality representatives are level with searching every file on these queries under the first embedder.
4. **Why?** A mixed folder is two clusters, its images and its texts, and one vector sits between them. Centering closes most of the global gap and leaves the within-folder separation, and it recovers 8 to 62 percent of the loss depending on the encoder and the cell. A small control with planted minorities shows that a pooled vector also loses a foreign minority of the folder’s own modality, three quarters as much as one of the other modality, and that the modality gap adds 0.18 recall@5 on top (interval 0.02 to 0.35, on ten synthetic hosts; Sections 6 and 7).
5. **What fixes it, and at what budget?** None of the single vectors we tried in the encoder’s space. A few representatives per modality label, found by k -means within the label, do. Four per label, about six vectors per folder, are not inferior to an exhaustive file-level search at a margin of 0.02 recall@5, over all file queries and in every cell under all four encoders. That reading was registered before a study varied the budget from one to eight centroids per folder, on a set earlier sessions had read (post hoc). Three per label, the budget fixed in the brief of session 2, meet the margin over all queries under all four encoders and in every cell under E3, and across three k -means seeds their recall@5 moves by at most 0.0026. By the registered rule they keep that margin at one byte per dimension, where they take 3,993 to 10,648 bytes per folder on average by encoder, and at one bit per dimension under the two vision-language embedders. Two vectors per folder meet a pre-registered margin under the first vision-language embedder, narrowly, and fall short of it under the other three encoders. At 2 KiB, which holds eight one-bit vectors under the first embedder, a sample of the folder’s own files dealt to its kinds keeps the minority files that the folder’s mean loses, and k -means centroids keep the majority of larger folders better. On ext4 such a summary costs 1.2 blocks per directory read as an extended attribute. A folder’s type composition adds little when

the wanted file type is known. At this scale the folder layer itself buys nothing over a flat approximate index, which needs every file’s vector as file-level search does (Sections 12, 14 and 15).

6. **Does a folder summary guide a walk down a tree?** Not yet at 2 KiB. On 261 whole repository trees, a walk guided by summaries at 2 KiB finds the target 0.069 less often than reading every directory’s own summary, though it reads 0.39 as many, and over all queries the mean is the best summary for the walk. Among summaries that merge exactly up a tree, a per-kind sample beats a uniform one on the minority files, and at 8 KiB it beats the mean on them and brings the walk within 0.017 of the flat read (Section 13).
7. **Does a written folder abstract do better?** Not for the folder’s images. An abstract written by a vision-language model from a caption of every image, and asked to cover every kind of file, finds the folder of a described minority image no more often than the pooled vector under two encoders. It loses majority images too. On whole-folder title queries it matches per-modality representatives under one encoder and falls below them under the other, and added to them it gains a little under both (Section 11).

The contribution is a measurement with its controls and the rule it supports, and the remedies we test are existing ones.¹ The core results are leave-one-out directory-retrieval measurements with paired bootstrap intervals over directories, and the searcher queries test the same loss on queries that are not files. Section 9 repeats the headline intervals with resampling over the records’ creators. Each hypothesis was written into a brief, with its criterion, before its test was run. Five of the briefs are timestamped outside the repository, and the rest are dated by the working repository’s history (Section 3). The briefs, the results, the hypotheses that died and one evaluation bug that an audit of this paper found and corrected (Section 12) are in the repository.

2 Related work

The modality gap. Liang et al. [36] showed that multi-modal contrastive models embed images and text in separate regions of their shared space. Each encoder’s outputs lie in a narrow cone at initialisation, and contrastive training keeps the two apart. Chowers et al. [11] trace the gap to the same interaction of loss and initialisation, argue that it is a bug where robustness matters, and close it by a post-processing shift toward the other modality’s mean without loss of clean accuracy. For mixed-modality search, Li et al. [34] show that documents of the query’s modality are ranked higher and remove the bias by subtracting each modality’s mean embedding (GR-CLIP). Each of their documents is a single image and a single text segment, and they name multi-image, multi-text documents as future work. Liu et al. [39] report, for CLIP-style models, that mean-centering removes the offset between the modality centroids and leaves what they call the distribution gap unchanged. Role et al. [54] note that while a gap remains, a query’s embedding tends to be closer to items of its own modality, and Lin et al. [37] report a modality bias in retrievers built on multimodal language models. For one item with several modality streams, Mehrish et al. [45] report that symmetric aggregators often lag the strongest single stream. The closest work to a pooled representation is Grassucci et al. [22], who consider the centroid of a semantic class that spans two modalities and show that closing the gap with a training objective improves clustering while leaving retrieval rankings mostly unaffected. Among embedders built on vision-language models [23, 25, 37, 69], Günther et al. [23] call the gap dramatically reduced and report a cross-modal

¹Code, manifests, briefs and results: <https://github.com/ali-basheer/dirvec-study>.

alignment score of 0.71 on Flickr30K against 0.38 for jina-clip-v2 and 0.15 for OpenAI CLIP. These works study single items, or a class centroid under a training objective. None pools the embeddings of a folder’s files, of different modalities and different content, into one vector of a frozen encoder. What such a vector retrieves is the object of this paper.

Representing a collection. Federated search has two families of collection representation [58]. Lexicon-based methods such as CORI treat a collection as one large bag of words [9], and document-surrogate methods such as ReDDE select collections through sampled documents [59]. Selective search partitions one index into shards and routes a query to a few [2, 29, 30]. Whether one summary or several members should stand for a collection has been asked before, in text. For blog feed search, Elsas et al. [17] found that scoring entries beat scoring the whole feed as one document under their best settings. For blog site search, Seo and Croft [56] found that selection through clusters of highly ranked postings outperformed a global representation of the blog. In expert finding, Balog et al. [6] found that ranking documents first and aggregating them per candidate outperformed one representation per candidate in nearly all settings. Dense routers have returned to the single summary. The router of Dhasade et al. [14] takes the centroid of a source’s document embeddings as an input, and Lee et al. [33] route by centroid pilot embeddings and report that additional centroids per group act as distractors. Their routing averages a query’s similarity over an expert’s pilot embeddings, where we take the maximum over a folder’s representatives. Cluster-based retrieval asks the same question of a cluster of documents. The cluster hypothesis [62] holds that documents in one cluster answer the same queries, and cluster-based retrieval with language models ranks a cluster by one model of its documents, or smooths a document’s model with its cluster’s [31, 40]. A folder is such a cluster, one the file system already provides. All of these collections are text. This paper adds the modality axis.

Multiple representatives. That one centroid fails a class made of several clusters is textbook material. A nearest-centroid classifier misclassifies such a class [44], and prototype methods, several representatives per class, are a standard chapter of statistical learning [24]. The capacity of a single dense vector has also been studied for text retrieval [41]. The loss measured here is therefore expected in kind, and what this paper adds is its size on real mixed-media folders and what fits in the bytes a folder offers. Recommendation has long represented one entity by several vectors. MaxMF embeds a user as several interests and scores by the maximum over them [64]. MIND extracts several interest vectors per user with a capsule routing layer [35], and PinnerSage clusters a user’s actions with Ward’s method and keeps a medoid per cluster [51]. In multi-vector retrieval, late interaction keeps every token vector of a passage and scores a query by the sum over its tokens of the best match [26], and token pooling clusters a document’s token vectors and mean-pools each cluster [12]. The flat walk $\mathbf{d}$ is the file-level analogue of keeping every vector. No late-interaction encoder was measured here. Pooling the members’ vectors before scoring and taking the best member score after it are the early and the late fusion of multimedia analysis [60]. We propose no new multi-representative method. We measure which budget of representatives a mixed-media folder needs, and why.

Documents, hierarchies and file systems. Visual document retrieval embeds a page or a screenshot [18, 42, 67]. One level up, You et al. [66] compose a document from its pages and report that, among independent page scores, the maximum over pages beats their average (37.68 against 32.76 NDCG@5), the same direction as our result for folders. Zhang et al. [68] report a disproportionate visual dominance in the embedding of an interleaved text-image document.

Above the document, hierarchical retrieval has so far meant text summaries. RAPTOR clusters text chunks, summarises each cluster with a language model and embeds the summaries, and finds that searching all nodes at once beats traversing the tree [55]. GraphRAG builds an entity graph over a corpus, summarises its communities with a language model and answers global questions from the community summaries [16]. OpenViking, a context database for AI agents, builds a directory’s abstract bottom-up from summaries of its files, turning images and other media into text summaries first, and uses the abstract for vector retrieval. Its documentation states that retrieval no longer navigates directories recursively [50]. A summary written by a language model is a different folder representation from the pooled vector, and Section 11 measures one built in this way. In file systems, semantic file systems attach attributes to files and directories [19]. VectorVFS stores an embedding in each file’s extended attributes, for images only at its release [53]. The same mechanism on a directory would hold the folder’s own vectors, within the size limits listed in Section 12. The directory-aware vector database of Wang et al. [63] treats a directory as a scope predicate on a vector search and has no directory vector. To our knowledge no published work embeds a mixed-media folder as an object and measures what the embedding retrieves.

Indexes. Our flat baseline is HNSW [43]. Partitioned indexes pay off where the vectors do not fit in memory. SPANN keeps centroids in memory and posting lists on disk and reports twice the speed of DiskANN at the same recall on billion-scale sets [10]. Directory-scoped queries are a filtering problem with indexes of their own [52, 63]. Neither is the regime of Section 15, which asks only whether a folder layer helps an unfiltered search over tens of thousands of in-memory vectors.

3 Study design

Corpora. We use Zenodo records as folders. Each record is a directory assembled by real people, with a filterable licence field. The pool is the Zenodo search for public records of 3 to 60 files that hold both an image type and a document type under a CC0 or CC-BY licence, 2017 to 2026. We keep files of five kinds (image, PDF, text, table, office document), each at most 25 MB and at most 100 MB per record, and only the newest version of a concept. Four draws took part in the study (Table 1). S1 is a 400-directory pilot set stratified by image fraction. S3, for the pre-registered retest, is a 400-directory set drawn the same way from the same pool with S1 excluded. S5, the main set, holds the 2,809 eligible records the pool had left, 2,808 of which could still be downloaded. S11, the confirmation set, holds the 2,597 eligible records of a new pool snapshot (result pages beyond those already taken), with every earlier record and concept excluded. The four sets share no record. Images are mostly scientific figures (plots, maps, micrographs) rather than photographs. Two further draws, S19 and S24, are directories of public GitHub repositories, with owners as the independent unit (Section 10), and session 23 walks whole repository trees (Section 13). Modality labels come from the file: image, pdf_text, pdf_scanned (fewer than 50 extracted characters per page), text, table, other (office documents). A directory’s image fraction is the share of its files labelled image.

Ground truth. The task is directory retrieval. For a query file f in a directory D of at least three files, the one relevant item is D , and every directory representation is built leaving f out. A held-out file is a simulated known-item query. Simulated known-item queries have precedent in IR evaluation [5], as have simulated collections for desktop search [27]. Near-duplicates are found first (exact hash, perceptual hash for images, simhash for text). A file with a near-duplicate anywhere in the set, in its own directory or another, is not a query, because a copy left behind in the

Table 1: The draws of single folders, four of Zenodo records and two of GitHub directories. Directories per image-fraction bucket $[0, .2)$, $[\cdot 2, .5)$, $[\cdot 5, .8)$, $[\cdot 8, 1]$; queries are leave-one-out files after deduplication, counted as scored in the session that reports each set. File size is the sum of the kept files’ bytes.

	S1 (pilot)	S3 (retest)	S5 (main)	S11 (confirmation)	S19 (GitHub)	S24 (GitHub)
Directories	400	400	2,808	2,597	1,600	2,600
Files	4,773	4,891	27,545	25,990	12,656	21,854
Total file size	5.2 GB	4.8 GB	29.0 GB	28.5 GB	1.6 GB	3.5 GB
Directories per bucket	103/100/98/99	103/99/99/99	459/1,005/1,034/310	420/795/978/404	700/378/244/278	943/772/510/375
Queries with a vector	4,071	4,041	23,130	21,225	9,089	16,185
Role	encoder pilot, exploratory	pre-registered retest	main, sessions 5 to 10 and 13	confirmation, four encoders	first draw, sessions 19 and 21	second draw, session 24

directory would make leave-one-out trivial and a copy elsewhere would make it wrong. Recall@ k is the fraction of queries whose directory is in the top k of all directories in the set. We report recall@5, with 95 percent paired bootstrap intervals from 1,000 resamples of directories. Recall@1 and recall@10 are in the repository’s results files. Differences are computed before rounding, so a printed difference can be 0.001 away from the difference of two printed levels.

Cells. Queries are grouped by their modality and by the image fraction of their directory. The two *minority* cells are P1, image queries in directories with image fraction below 0.5, and P2, text-like queries (text, table, pdf_text, other) in directories with image fraction of 0.5 or more. The two *majority* cells M1 and M2 are the complements (image queries in image-heavy directories, text-like queries in text-heavy ones). Scanned PDFs as queries belong to none of the four cells (192 of the 17,140 evaluation queries of S11).

Encoders. E1 is jina-embeddings-v4 [23], a Qwen2.5-VL-based embedder with one space for text and images (2,048 dimensions, single-vector mode, retrieval task, query-side text path), chosen over nomic-embed-multimodal [47] by a pilot on 20 directories (text or table file to own-directory image, median rank 8 of 167 against 19 for the runner-up and 84 at random). E2 is jina-clip-v2 [28], a CLIP-family dual tower with an 8k-token text side (1,024 dimensions); on the same check in 20 directories of the confirmation set its median rank is 51 of 159 against 26 for E1 and 80 at random. That is a weak pass: 15 of the 20 directories are better than random, but the bootstrap interval of the median, 6.5 to 103, includes it. E3 is gme-Qwen2-VL-2B-Instruct [69], a unified embedder from another vendor (Alibaba), built on Qwen2-VL-2B where E1 is built on Qwen2.5-VL-3B (1,536 dimensions, last-token pooling). Session 14 added it so that the vision-language result would not rest on one vendor. It takes no task instruction here beyond the model’s default system prompt, on files or on queries, and on the same check its median rank is 21 of 159. E4 is Nomic Embed v1.5 [48, 49], a second CLIP-family dual tower, from a third vendor (Nomic AI). Its image tower was trained against its frozen text tower (768 dimensions). Session 15 added it so that the dual-tower result would not rest on one model. Its texts carry the prefix the model card gives for text searched against images (search_query), on files and on queries alike, so that a folder’s texts and the queries that reach them share one text path. The choice was written down before any E4 number was read, and the document prefix was not tried. On the same check its median rank is 65.5 of 159, a weaker pass than E2’s. Each encoder reads an image as the image itself, at most 602,112 pixels under E1 and E3 and at least 200,704 under E3, the model’s default. E2 and E4 apply their own shipped preprocessing, E2 the shortest side to 512 pixels and a centre crop, E4 the whole image resized to 224 by 224 pixels without keeping its aspect ratio. Text, table, pdf_text and office documents give the first 2,000 tokens of extracted text, counted by E1’s tokenizer for every encoder. A scanned PDF gives the mean of the vectors of its first eight page renders. The four encoders read the same characters and the same decoded images, apart from a few page renders too small for E1. E4’s own tokenizer also cuts 14 of the 12,757 texts of the confirmation set at its limit of 8,192 tokens. On the

confirmation set all four embed the same 25,637 of its 25,990 files, and on the main set E1 embeds 27,243 of 27,545. The rules were fixed in session 2 and changed three times afterwards, each time for files an encoder rejected (images and page renders under 28 pixels on a side or with an aspect ratio above 200 are skipped; an .xls file that is tab-separated text is read as text). No change alters a vector that had already been computed, and re-scoring S3 after the .xls change moved no cell by more than 0.010.

Directory representations. All are built leave-one-out from the directory’s remaining file vectors and scored by the maximum cosine over the directory’s vectors.

- **a:** the pooled centroid, one vector, the mean of the child vectors.
- **b:** one centroid per modality label (up to six vectors).
- **c:** per-modality k -means with $k = \min(3, n_m)$ representatives for a label with n_m files (up to 18; 5.2 to 5.4 on our sets). The budget of three per label was fixed in the brief of session 2, and session 27 varied it from one to five per label (Section 12). The k -means runs with ten initialisations and one fixed seed, so every run is deterministic. Across three seeds the recall@5 of **c** moves by at most 0.0026 in any cell under any encoder, and that of two blind centered clusters (**tb2c**, below) by at most 0.0084 (session 27, post hoc).
- **d:** every child vector kept, the flat file-level walk. It needs the vector of every file and is the reference the compact representations are measured against.

Centered variants (**ac**, **bc**, **cc**, **dc**) subtract from every file vector, and from the query, the mean vector of its own input group (image inputs: image and pdf_scanned; text inputs: the rest) and renormalise before any pooling, as GR-CLIP does [34]. The group means are computed on a 20 percent calibration split of the set’s directories. On the main set the split’s queries were used to choose parameters and to check for leakage (Section 12). The pilots of E2, E3 and E4 were drawn from the confirmation set’s split. Nothing fitted or chosen on a split is tested on that split’s file queries. Two-vector variants are **bc2**, one centered centroid per input group, and **tb2c**, two centered clusters found by label-blind k -means. Section 12 introduces the single-vector families.

Protocol. The study ran as 27 sessions, each with a brief written before it. A session is one working session in which the author directed an AI coding assistant. “Agent” in this paper means such an assistant. Twenty-two sessions (2, 3, 5 to 12, 14 to 21, 23, 24, 26 and 27) tested a hypothesis whose kill criterion was fixed in the brief before the numbers that decide it were computed. The others built the first corpus (1), measured cost (4), re-ran one evaluation after a bug was fixed (13, Section 12), measured a file system (22, Section 12) and computed from per-query files the estimates that an earlier round of review of a draft asked for (25). Sessions 25 and 26 followed that review and read draws the study had already read, and the paper marks their results post hoc. Session 27 followed a later round of review. Its brief registered queries written as a searcher would write them (Section 5), budgets from one to eight centroids per folder and three k -means seeds (Section 12), before any of their new numbers existed. The last two read the confirmation set again, as earlier sessions had, and the paper marks them post hoc. From session 3 on, the results file copies the hypothesis verbatim and reports the verdict either way. The briefs of sessions 19, 21, 23, 24 and 27 were also timestamped with OpenTimestamps [61], each anchored in a Bitcoin block mined before its first numbers were committed, and the released repository holds the proofs. The amendment of session 27 was stamped before its queries were scored, but its block was mined a few

session	draw	hypotheses and verdicts
1	S1	■ corpus
2	S1	✗ Claim
3	S3	● H
4	S3	■ cost
5	S5	● H5
6	S5	✗ H6
7	S5	✗ H7
8	S5	✗ H8
9	S5	● H9a ● H9b ✗ H9c
10	S5	● H10 (synthetic)
11	S11	● H11a (E1) ✗ H11b (E2)
12	S11	● H12a (E1) ● H12b (E1, E2)
13	S5	● H9a (re-run)
14	S11	● H14a (E3) ● H14b (E3)
15	S11	● H15a (E4) ● H15b (E4) ● H15c (E4 vs E1)
16	S11	● H16a ✗ H16b ✗ H16c
17	S11	● H17a ● H17b (E1, E3) ● H17c (E1)
18	S11	● H18a (E1, E3) ● H18b (E1, E3)
19	S19	○ H19a ○ H19b ● H19c ○ H19d
20	S11	● H20a ● H20b ● H20c
21	S19	● H21a (pooled) ✗ H21b ● H21c (pooled)
22	S19	■ store
23	S23	○ H23a ● H23b ✗ H23c
24	S24	● H24a ● H24b ● H24c ● H24d ● H24e ✗ H24f ● H24g
25	S11	■ estimates (S19, S24)
26	S19	● H26a (pooled) ● H26b (pooled) ✗ H26c (pooled) ● H26d (S11, E1, E3)
27	S11	● H27a (E1) ● H27b (E1) ● H27c (E1) ■ budget ■ seed

● survived ✗ died ● partly (one clause, or one of two readings) ◆ split by encoder ○ inconclusive ■ no hypothesis

Figure 2: The study as it ran. Each row is a session with the draw it scored: Zenodo records (S1 to S11), GitHub directories (S19, S24) or whole repository trees (S23). A marker gives the verdict of one pre-registered hypothesis, stated with its criterion in the appendix. On S11 a label names the encoders a hypothesis was read under, or none when it was read under all four. On the GitHub draws and the trees every hypothesis was read under E1, and H21a and H21c take the pooled reading of both GitHub draws, as the brief of session 24 fixed. Session 26 was registered after an earlier round of review, on draws already read. H26a to H26c pool the two GitHub draws, and H26d is on S11. Session 27 was registered after a later round of review. Its H27a to H27c are read on queries that a model wrote as a simulated searcher, and its budget and seed readings have no kill criterion. Squares mark the sessions, or the parts of a session, that tested no hypothesis.

minutes after the scoring run ended (Section 5). The other briefs, among them those of the Zenodo tests before session 27, are dated by the history of the working repository, which is available on request. Their pre-registration rests on that history. These sessions use the readings of Section 10, with a smallest effect of interest, a minimum number of independent owners and an outcome of inconclusive where the data cannot decide. Where a choice had to be made (an alignment weight, a pooling exponent, the weight of a prior), it was made on the calibration split’s queries, and the hypothesis was read on the evaluation split. Session 13 read H9a there a second time after the fix, with the hypothesis, the grids and the selection rule unchanged. Figure 2 shows the sessions in order with the verdict of each hypothesis, and the appendix lists the hypotheses. The repository holds the briefs, the results files with full tables, the manifests and the ground truth.

4 Pooling loses the minority modality

The exploratory finding. The study began with a different claim, that pooling collapses a directory toward its text and loses image-heavy children. On S1 that claim died under its kill

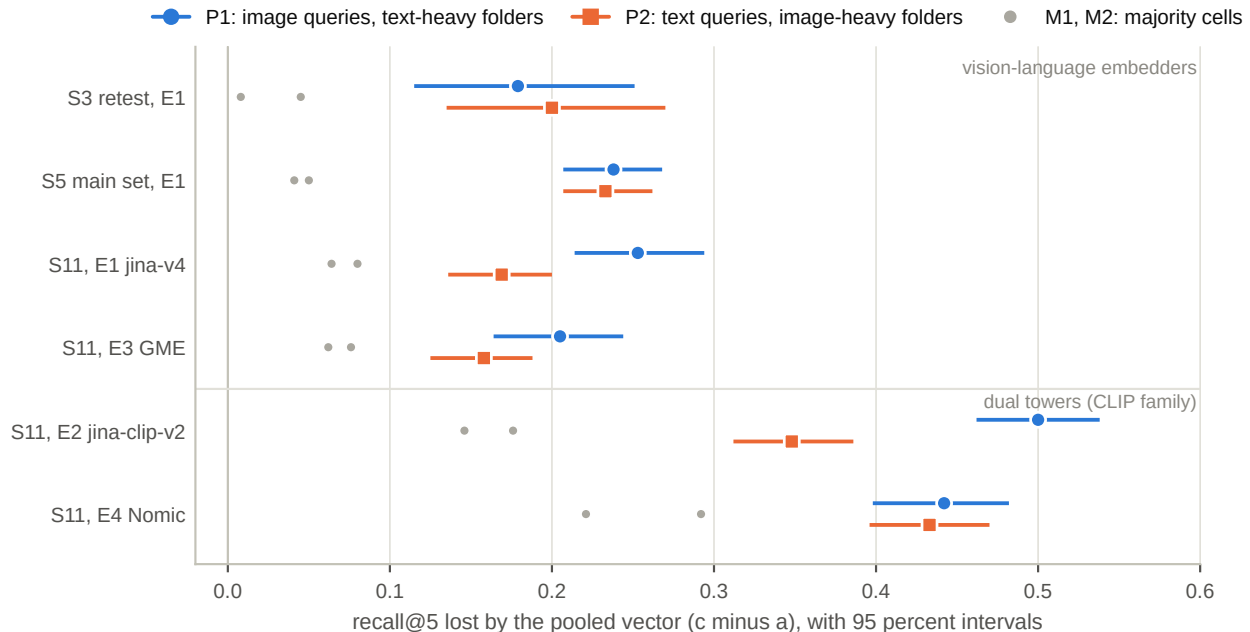


Figure 3: How much the pooled vector loses: recall@5 of c minus a with paired 95 percent intervals, for every measurement in Table 2. Blue: image queries in text-heavy folders (P1). Orange: text queries in image-heavy folders (P2). Gray: the majority cells (M1, M2). The upper rows are vision-language embedders and the lower rows CLIP-family dual towers.

criterion, which needed the pooled centroid beaten on image queries in both image-heavy buckets. It was beaten in $[.5, .8)$ (c minus a $+0.054$) and not in $[.8, 1]$ (-0.001 $[-0.020, +0.016]$). The same tables showed a different pattern. The pooled centroid lost 0.27 recall@5 on image queries in the most text-heavy directories and 0.27 on text-like queries in the most image-heavy ones, and almost nothing where the query’s modality dominated its directory (-0.001 and $+0.021$). What the pooled vector loses is the *minority* modality of each directory, whichever modality that is.

Pre-registered on fresh directories. The pattern was then written down as a hypothesis with its cells and kill rule and tested on S3, 400 directories drawn after S1 was excluded. It survived, with c minus a at $+0.179$ $[+0.115, +0.251]$ in P1 and $+0.200$ $[+0.135, +0.270]$ in P2 ($+0.188$ and $+0.197$ when S3 is re-scored under the later .xls rule). The per-modality centroids b gain the same ($+0.184, +0.195$). In the majority cells c ’s gain is $+0.045$ and $+0.008$. On the main set S5, where these cells were a reported secondary rather than a kill test, the loss is larger and the intervals are tight, at $+0.238$ $[+0.207, +0.268]$ and $+0.233$ $[+0.207, +0.262]$ over all 23,130 queries. On the confirmation set S11 under E1 it is $+0.253$ $[+0.214, +0.294]$ and $+0.169$ $[+0.136, +0.200]$. The kill tests of the loss on S11 are the E2, E3 and E4 measurements, the second calibration seed below and the checks of Section 9. Table 2 and Figure 3 collect the six measurements, and Table 3 gives the cells on S11.

Under CLIP-family dual towers the loss is larger. Under E2 the pooled centroid finds the directory of a minority-modality query in the top five 5.5 percent (P1) and 3.5 percent (P2) of the time, against 55.5 and 38.3 percent for c . That makes c minus a $+0.500$ $[+0.462, +0.538]$ and $+0.348$ $[+0.312, +0.386]$. The majority cells lose too ($+0.146, +0.176$). Session 15 added E4, a second dual tower from another vendor, and tested the loss on S11 by the same rule (H15a).

Table 2: The minority-modality loss, recall@5 of **c** minus **a**, paired bootstrap over directories. P1: image queries in text-heavy directories; P2: text-like queries in image-heavy directories; M1 and M2: the majority cells. Kill-tested rows are marked. The S11 row under E1 was kill-tested under a second calibration seed (+0.255 and +0.193).

Set	Encoder	P1	P2	M1	M2
S3 (kill-tested)	E1	+0.179 [+0.115, +0.251]	+0.200 [+0.135, +0.270]	+0.045	+0.008
S5 (main, 2,808 dirs; reported)	E1	+0.238 [+0.207, +0.268]	+0.233 [+0.207, +0.262]	+0.041	+0.050
S11 (reported; second seed kill-tested)	E1	+0.253 [+0.214, +0.294]	+0.169 [+0.136, +0.200]	+0.080	+0.064
S11 (kill-tested)	E2 (jina-clip)	+0.500 [+0.462, +0.538]	+0.348 [+0.312, +0.386]	+0.146	+0.176
S11 (kill-tested)	E3 (GME)	+0.205 [+0.164, +0.244]	+0.158 [+0.125, +0.188]	+0.076	+0.062
S11 (kill-tested)	E4 (Nomic)	+0.442 [+0.398, +0.482]	+0.433 [+0.396, +0.470]	+0.292	+0.221

It survived, at +0.442 [+0.398, +0.482] in P1 and +0.433 [+0.396, +0.470] in P2. Under E4 the pooled centroid finds the directory of a minority-modality query in the top five 0.6 and 0.5 percent of the time, against 0.2 percent for a random ranking. Its majority cells lose +0.292 and +0.221. The session also pre-registered that on the same queries the loss under E4 exceeds the loss under E1 (H15c). The paired difference is +0.189 [+0.145, +0.231] in P1 and +0.265 [+0.226, +0.305] in P2, so H15c survived. Measured the same way, E2 loses more than E1 by +0.247 [+0.208, +0.286] and +0.179 [+0.142, +0.223]. Between the two towers the order depends on the cell. E4 loses less than E2 in P1 (−0.058 [−0.088, −0.025]) and more in P2 (+0.086 [+0.067, +0.105]). The two towers are also the weakest encoders of the four on these files, by the pilot check and by **c** over all queries (0.683 under E2 and 0.693 under E4, against 0.796 under E1 and 0.811 under E3). In the minority cells **c** reaches 0.555 and 0.383 under E2 and 0.448 and 0.438 under E4, against 0.674 and 0.539 under E1 and 0.741 and 0.544 under E3. Part of the larger loss is therefore the encoders, and not only their wider gap.

Under a second vision-language embedder the loss holds. E1 and E2 come from one vendor. Session 14 added E3, from another, and tested the loss on S11 under the kill rule of session 3, fixed in the brief before any E3 number was computed (H14a). It survived, with **c** minus **a** at +0.205 [+0.164, +0.244] in P1 and +0.158 [+0.125, +0.188] in P2. The majority cells lose +0.076 and +0.062. E3 is the strongest of the four encoders on these files over all queries (**c** 0.811 against 0.796 under E1, 0.693 under E4 and 0.683 under E2). Its pooled centroid still finds the directory of an image query in a text-heavy folder in the top five only 53.6 percent of the time, against 74.1 percent for **c**. The brief expected a loss between E1’s and E2’s if E3’s gap lay between theirs. The gap is about E1’s instead (0.369 against 0.381; Section 6), and so is the loss (0.253 and 0.169 under E1).

Robust to the calibration split. The S11 numbers use one 20 percent calibration split (seed 20261102) for the group means and the cell membership. Session 12 repeated the evaluation under a second seed (20261103; 17,208 queries over 2,034 directories) with the pre-registered expectation that the verdicts do not move. They do not. Under E1 **c** minus **a** is +0.255 [+0.215, +0.292] in P1 and +0.193 [+0.163, +0.228] in P2, and under E2 it is +0.513 [+0.473, +0.548] and +0.361 [+0.325, +0.395], against +0.253, +0.169, +0.500 and +0.348 under the first seed. Every cell of **tb2c** minus **c** under E1 moves by at most 0.003 (−0.035 [−0.058, −0.012] in P1, −0.029 [−0.043, −0.017] in M2), so the two-vector verdict stands as reported, by the rule and by little.

The flat walk is the reference, and **c is close to it.** Keeping every file vector (**d**) is the natural reference for a representation built from the same vectors. It is not a strict ceiling. On the

Table 3: Recall@5 on the S11 evaluation split (17,140 queries over 2,035 directories, 2,597 ranked) by representation, encoder and cell; the cells hold 1,959 (P1), 2,146 (P2), 5,571 (M1) and 7,272 (M2) queries under every encoder. Vision-language embedders above, dual towers below. Vectors per directory in parentheses. **a**: pooled; **ac**: pooled, centered; **bc2**: two labelled centered centroids; **tb2c**: two blind centered clusters; **c**: per-modality k -reps; **cc**: the same, centered; **d**: every file; **dc**: every file, centered.

	E1 jina-embeddings-v4					E3 gme-Qwen2-VL-2B				
	all	P1	P2	M1	M2	all	P1	P2	M1	M2
a (1.00)	0.692	0.421	0.370	0.730	0.830	0.716	0.536	0.386	0.761	0.827
ac (1.00)	0.719	0.513	0.474	0.734	0.835	0.731	0.552	0.471	0.768	0.826
bc2 (1.95)	0.769	0.671	0.591	0.773	0.845	0.782	0.709	0.593	0.812	0.833
tb2c (2.00)	0.781	0.638	0.575	0.805	0.862	0.788	0.682	0.565	0.831	0.849
c (5.20)	0.796	0.674	0.539	0.810	0.894	0.811	0.741	0.544	0.837	0.889
cc (5.20)	0.810	0.713	0.590	0.814	0.899	0.816	0.742	0.568	0.839	0.890
d (9.87)	0.804	0.685	0.551	0.821	0.897	0.815	0.744	0.547	0.843	0.890
dc (9.87)	0.818	0.714	0.597	0.828	0.905	0.820	0.745	0.569	0.847	0.894

	E2 jina-clip-v2					E4 Nomic Embed v1.5				
	all	P1	P2	M1	M2	all	P1	P2	M1	M2
a (1.00)	0.457	0.055	0.035	0.640	0.551	0.397	0.006	0.005	0.397	0.621
ac (1.00)	0.559	0.278	0.229	0.702	0.624	0.544	0.082	0.189	0.572	0.756
bc2 (1.95)	0.617	0.538	0.390	0.734	0.615	0.633	0.421	0.451	0.616	0.762
tb2c (2.00)	0.634	0.450	0.331	0.778	0.664	0.640	0.331	0.398	0.657	0.788
c (5.20)	0.683	0.555	0.383	0.786	0.727	0.693	0.448	0.438	0.689	0.842
cc (5.20)	0.693	0.575	0.404	0.786	0.737	0.703	0.460	0.465	0.693	0.850
d (9.87)	0.694	0.554	0.383	0.796	0.745	0.705	0.460	0.456	0.702	0.850
dc (9.87)	0.706	0.570	0.401	0.798	0.761	0.715	0.470	0.466	0.715	0.858

full S5 set **c** is 0.008 above it in M1 (interval [0.001,0.015]). On S11 **cc** and **dc** are above it in the minority cells under E1 and E2, and in P2 under E3 and E4. In P1 under E3 and E4, **cc** is level with it. Over the five cells (all queries, P1, P2, M1, M2), **c** is within 0.013 of **d** on S3 and S5 and within 0.018 on S11 under all four encoders (Table 3). This was pre-registered under E3 and E4 (H14b, H15b), where the largest gaps are 0.006 and 0.018. Under E4 **c** is below **d** in all five cells, with every interval below zero. In single-bucket cells the gap reaches 0.030. The useful question is therefore how much of file-level search a folder representation keeps.

The number of candidates. The loss depends on how many folders compete. With the own folder and $n - 1$ others drawn at random as the candidates, the expected recall@5 follows from the ranks (session 25, post hoc). Under E1 on S11 the loss in P1 and P2 is +0.014 and +0.032 among 10 candidates, +0.114 [+0.090, +0.141] and +0.140 [+0.119, +0.164] among 100, and +0.232 and +0.152 among 1,000, against +0.253 and +0.169 among all 2,597. The GitHub draws are read the same way in Section 10.

5 Searcher queries

Every query so far is a file of a folder or, in Section 8, the depositor’s own words about a record. Session 27 asked whether the loss reaches the words someone would type to find one file again. Its brief drew the folders and the target files and built a tool for writing queries, and the author was

to write them. The tool showed each folder’s pictures as thumbnails and the first 300 characters of its documents, with file names, the record’s title and description and every digit hidden. The author found the documents of the first folder, raw numeric data files, unreadable with their digits hidden, and no person wrote queries. An amendment, committed before any query was embedded or scored, made a language model the writer. The Bitcoin block that anchors its timestamp was mined a few minutes after the scoring run ended, so that order rests on the repository’s history. The hypotheses and the scoring stayed as the brief fixed them, and the brief’s block precedes every number of the session. Language models have written queries for retrieval before, as training data [8] and as query variants for test collections [1]. We report these as queries that a model wrote as a simulated searcher.

Folders and targets. The folders are evaluation-split folders of the confirmation set outside the four digitization series of Section 9. Each holds at least two images and two text-like files that are evaluation queries under E1 and have a vector under all four encoders. A fixed seed drew 30 folders per image-fraction bucket, at most one per creator family, 120 in all. In each folder one image and one text-like file were drawn as targets, with further files in a drawn order in case the writer could not describe a target. The writer therefore could not pick the files a pooled vector misses. The sample was committed before any query was written.

The writer. Claude Opus 5.5 (Anthropic) wrote the queries, as six subagents of the assistant that ran the session, each given twenty folders. Each subagent saw a prompt that says nothing about the study, its representations, its hypotheses or any result, and the material the tool shows for its folders. That material is a sheet of every picture in the folder with the target framed, the target pictures at the tool’s size, and the folder’s documents as the tool shows them, digits hidden. For each folder a subagent wrote one query to find the target picture (QI) and one to find the target document (QT), as the words someone would type into a search box. A script checked the tool’s rules on every query, no digit and no run of five words copied from the folder’s text, and all 200 queries pass both. Their median length is eight words. One picture target and one document target were skipped as impossible to describe. The subagent given 20 of the folders was refused by an automated permission check before it ran and was not retried. That leaves 100 folders, 25 per bucket, and 50 queries in each of four cells. The writer belongs to none of the encoders’ model families.

Scoring. Each query is embedded on its encoder’s query text path, as the titles of Section 8 are. Every one of the 2,597 confirmation folders is a candidate, represented by all of its files with a vector, since the query is not a file. The minority cells are QI-T, picture queries for folders with image fraction below 0.5, and QT-I, document queries for folders at 0.5 or more. Their majority counterparts are QI-I and QT-T. Intervals are 95 percent from 1,000 resamples of creator families, and with one folder per family a cell holds 50 families. Before any query was scored, the scorer reproduced the ranks of **a**, **c** and **d** on all 2,597 titles of Section 8 under each encoder. On the queries, the flat walk **d** reaches recall@5 0.720 in QI-T and 0.940 in QT-I under E1, above the 0.25 below which the brief would have read a cell’s queries as too vague.

The loss. Three hypotheses were registered under E1 (Figure 4). H27a required **c** minus **a** of at least +0.05 in QI-T with its interval above zero, and it survived at +0.140 [+0.020, +0.280]. H27b, the same in QT-I, survived at +0.420 [+0.280, +0.560]. H27c read **c** minus **d** over all 200 queries as non-inferiority at the margin of 0.02, and at +0.000 [−0.020, +0.020] it is not inferior. Over

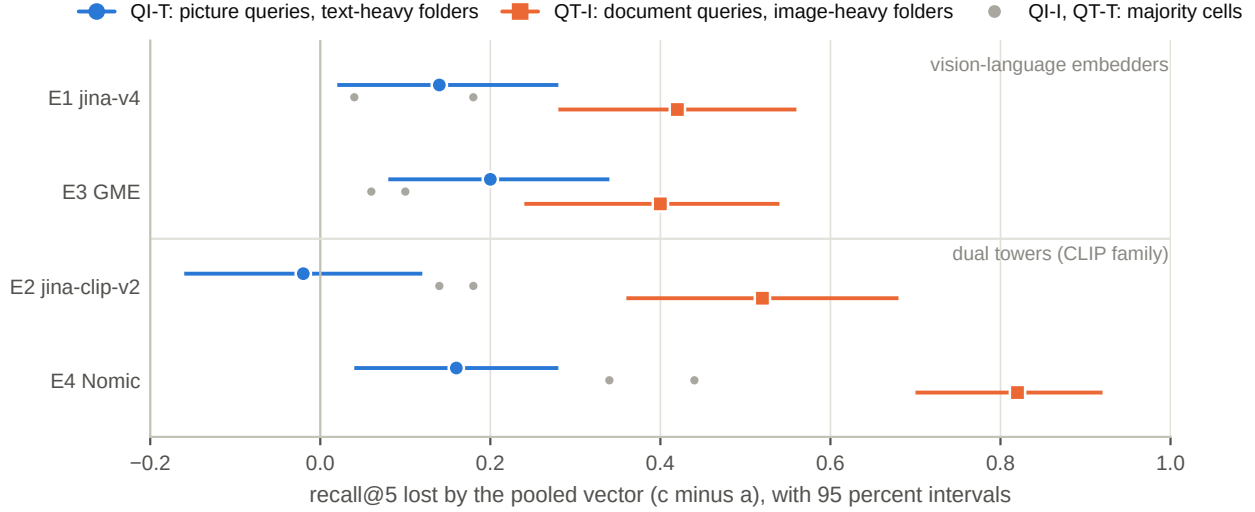


Figure 4: The loss on queries that a model wrote as a simulated searcher (session 27): recall@5 of *c* minus *a* with creator-clustered 95 percent intervals, 50 queries per cell from 100 folders of the confirmation set, all 2,597 folders ranked. Blue: picture queries in text-heavy folders (QI-T). Orange: document queries in image-heavy folders (QT-I). Gray: the majority cells (QI-I, QT-T). The registered tests are the E1 rows.

all queries the pooled vector reaches recall@5 0.675, and *c* and *d* reach 0.870. In QT-I the pooled vector reaches 0.500 against 0.920 for *c*, and in QI-T 0.600 against 0.740. The brief reads H27a and H27b together as an intersection-union test [7], each on its 95 percent interval. Counted instead as one family of three primary tests, H27a’s 98.33 percent interval runs from -0.020 to $+0.314$ and reaches below zero, so that reading would call it dead. H27b’s runs from $+0.260$ to $+0.594$, and H27c’s from -0.025 to $+0.020$, which that reading would call inconclusive.

Other encoders and cells. Under the other three encoders the document cell loses as well, by $+0.520$ under E2, $+0.400$ under E3 and $+0.820$ under E4, with every interval above zero. The picture cell loses under E3 and E4, by $+0.200$ [$+0.080, +0.340$] and $+0.160$ [$+0.040, +0.280$], and shows no loss under E2 (-0.020 [$-0.160, +0.120$]). Under E1 a document query also loses in text-heavy folders, $+0.180$ [$+0.080, +0.300$] in QT-T, as titles do (Section 8), and a picture query in an image-heavy folder loses little ($+0.040$ [$-0.060, +0.140$]). Centering does not help the pooled vector under E1 (-0.010 [$-0.040, +0.020$] over all queries). Two blind clusters lose against *c*, by 0.110 [$0.060, 0.165$] centered and 0.090 [$0.045, 0.140$] uncentered.

File names. The writer saw no file name. A router that scores a folder by the best character *n*-gram cosine between the query and its files’ names, built as in Section 9, puts the target’s folder in the top five for 0.240 of the queries. Fused with the E1 vector rankings by reciprocal rank fusion ($k = 60$), *c* minus *a* is $+0.090$ [$+0.045, +0.135$] over all queries and $+0.220$ [$+0.100, +0.340$] in QT-I.

What these queries can show. A model wrote them, from what a person would see of a folder, and a model’s queries may be more specific than a person’s. In a study of test-collection queries, variants written by a language model found similar sets of relevant documents to people’s variants without capturing their full variety [1]. Fifty queries per cell is a small sample, and the interval of the picture cell reaches down to 0.02. The result shows the loss on queries written to find one file

from what is visible of its folder, in both cells under E1, E3 and E4 and in the document cell under E2. Whether it appears on the queries of people searching their own folders was not measured.

6 The mechanism: a folder is two clusters

The obvious explanation is the modality gap [36]. If a folder’s images and texts sit in different regions, their mean sits in neither, and a query of the smaller group is far from it. We tested the published remedy, per-modality mean-centering before pooling [34], as a pre-registered hypothesis on S5. It said that the centered pooled centroid $\mathbf{ac}$, still one vector per folder, recovers more than half of what $\mathbf{c}$ recovers. From here on, S5 numbers are on its evaluation split (18,742 queries; the calibration split supplies the group means), where $\mathbf{c}$ minus $\mathbf{a}$ is +0.236 and +0.241.

Centering closes most of the global gap and leaves each folder split. The distance between the mean image-input vector and the mean text-input vector over the evaluation split falls from 0.361 to 0.072 after centering. The folders do not follow. Table 4 and Figure 5 give the mean cosine of same-folder file pairs. Before centering, a cross-modality pair in the same folder has cosine 0.524, against 0.770 and 0.722 within a modality. After centering all three fall, the cross pairs most, to 0.205 against 0.568 and 0.462. Centering removes the shared offset between the two groups and leaves the within-folder separation, so the pooled vector still sits between two clusters. Under E2 the gap is wider, 0.787 before centering, and the same-folder cross cosine starts lower, at 0.196 against 0.751 and 0.701 within a modality (0.096 against 0.561 and 0.410 after centering). Under E3 the gap is 0.369, about E1’s, and the folder is again two clusters, at 0.272 across against 0.611 and 0.569 within a modality (0.174 against 0.543 and 0.439 after centering). By these cosines E3 holds a folder’s two groups further apart than E1 does, yet its loss is a little smaller. Under E4 the gap is the widest of the four, 1.095, and a folder’s images and texts are close to orthogonal, at 0.066 across against 0.917 and 0.698 within a modality (0.034 against 0.555 and 0.406 after centering). Yet in P1 E4 loses less than E2 does (Section 4). With four encoders we do not try to say what sets the size of the loss. Two published results on single items sit beside this. Liu et al. [39] report that mean-centering takes the distance between the modality centroids from 0.686 to 0.020 and leaves what they call the distribution gap unchanged at 0.685. The folder-level cosines above show the same pattern inside folders, where the offset goes and the separation stays. Li et al. [34] report that after centering, an interpolation of the image and the text embedding of one document beats either alone. A folder’s files are not two views of one document. For them centering recovers part of the loss, about half on the main set, and no interpolation weight there recovers the rest (Section 12). The alignment score of 0.71 that Günther et al. [23] report for E1 is the average cosine of matched image-text pairs, a different quantity from the distance between the modality means measured here.

Centering recovers part of the loss, and balancing trades the majority for the minority.

The hypothesis died. On S5, $\mathbf{ac}$ recovers 49 percent of $\mathbf{c}$ ’s gain over $\mathbf{a}$ in both minority cells ($(\mathbf{ac} - \mathbf{a}) - 0.5(\mathbf{c} - \mathbf{a}) = -0.002$ $[-0.023, +0.018]$ in P1 and -0.001 $[-0.020, +0.019]$ in P2). On S11 centering recovers 36 and 62 percent under E1, 45 and 56 percent under E2, 8 and 54 percent under E3 and 17 and 42 percent under E4. Under E3 it does almost nothing for image queries in text-heavy folders ($\mathbf{ac}$ minus $\mathbf{a}$ is 0.016 of the 0.205 in P1). On S5 a balanced one-vector folder ($\mathbf{ab}$, the mean of the two group means) recovers 92 to 93 percent of the minority loss, and a centered balanced vector ($\mathbf{acb}$) 107 to 110 percent. Both pay in the majority cells, where $\mathbf{ab}$ changes recall@5 by -0.030 and -0.069 against the pooled centroid and $\mathbf{acb}$ by -0.025 and -0.048 (against $\mathbf{c}$, -0.071 and -0.113 , -0.065 and -0.092). In the single-modality-heavy buckets the cost reaches -0.149 .

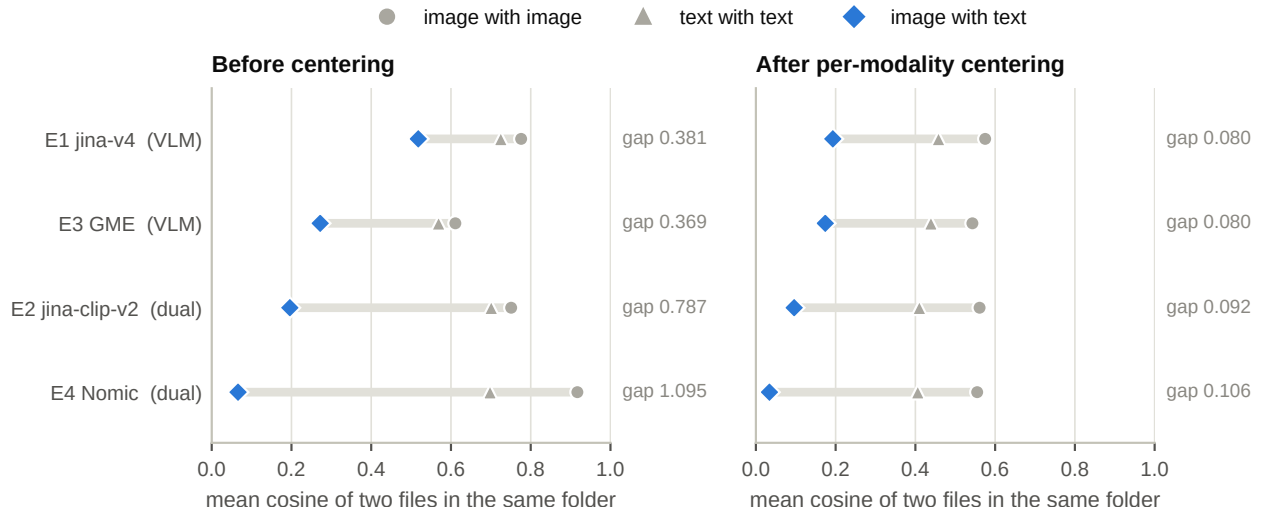


Figure 5: A folder is two clusters under every encoder. Each row gives the mean cosine between two files of the same S11 folder: two images (circle), two texts (triangle), or an image and a text (blue diamond), before and after per-modality centering (Table 4). The image-text pair sits well below both within-modality pairs, also after centering has closed most of the global gap (right-hand labels). In the row labels, VLM marks the vision-language embedders and dual the dual towers.

No single vector we tried held both the minority and the majority. Centering also helps the multi-vector representation and the flat walk itself, so its benefit is not specific to folder vectors. On the S5 evaluation queries cc minus c is $+0.017$ and dc minus d $+0.018$, with $+0.037$ to $+0.047$ in the minority cells. Under E3 on S11 the gains are $+0.004$ and $+0.006$ over all queries, $+0.022$ to $+0.025$ in P2 and nothing in P1. Under E4 both are $+0.010$ over all queries and $+0.010$ to $+0.027$ in the minority cells.

7 Modality, or any minority?

A sharper objection is that nothing above is about modality. A pooled mean dilutes any minority, and a folder’s minority modality is simply a minority. On natural folders the two cannot be separated, because label-blind clustering of a folder’s files recovers the modality partition. On S5 the blind clusters are 0.88 to 0.99 pure by input group. A pre-registered hypothesis, H9c, was that c beats blind k -means at its own budget in both minority cells. It died, at $+0.000$ $[-0.005, +0.006]$ and $+0.001$ $[-0.005, +0.006]$, and the test could not tell modality from dilution. We therefore built the control directly. Hosts are evaluation-split directories of S5 with at least six files, all of one input group. Into each host we planted a guest set of files from one other single-group record, at a fraction drawn uniformly from $[0.2, 0.5]$. We did this once with a record of the same input group (set SAME) and once with a record of the other group (set CROSS), at the same fraction for a host in both sets. Nothing is fitted on the synthetic folders. The pre-registered hypothesis is that the pooled centroid loses more on guest queries in CROSS than in SAME.

The corpus was selected for mixed folders, so the control is small. The evaluation split has 31 single-group directories of host size, donors come from the same pool, and 20 hosts received a guest in SAME, 16 of them also in CROSS. On the 10 hosts with guest queries in both sets, the loss of a against d on guest queries is 0.763 in CROSS and 0.583 in SAME. The difference is $+0.180$ $[+0.021, +0.345]$, so the hypothesis survives, by little. The guest also costs the host, with $+0.118$

Table 4: The modality gap at folder level. Gap: norm of the mean image-input vector minus the mean text-input vector over the evaluation split. Cosines: mean over same-folder file pairs.

Set, encoder	Vectors	Gap	img-img	txt-txt	img-txt
S5, E1	uncentered	0.361	0.770	0.722	0.524
S5, E1	centered	0.072	0.568	0.462	0.205
S11, E1	uncentered	0.381	0.776	0.725	0.518
S11, E1	centered	0.080	0.575	0.458	0.193
S11, E2	uncentered	0.787	0.751	0.701	0.196
S11, E2	centered	0.092	0.561	0.410	0.096
S11, E3	uncentered	0.369	0.611	0.569	0.272
S11, E3	centered	0.080	0.543	0.439	0.174
S11, E4	uncentered	1.095	0.917	0.698	0.066
S11, E4	centered	0.106	0.555	0.406	0.034

[+0.026, +0.238] more loss on the host’s own queries when the guest is of the other modality (16 hosts). Two things in the control matter more than the verdict. First, most of the control’s loss is there with no modality difference at all. On those ten hosts a same-modality minority from another record costs the pooled centroid 0.58 recall@5 on its queries, three quarters of the cross-modality 0.76 (over each set’s own hosts, 0.50 on 19 against 0.71 on 11). A pooled vector loses any foreign minority of a fifth to a half of the folder, and the modality adds 0.18 on top. Second, the labelled two-centroid representation cannot see a same-modality minority. Its SAME loss, 0.52, is no better than the pooled centroid’s 0.50. Two label-blind clusters recover most of it in both sets (losses of 0.11 in SAME and 0.02 in CROSS, and 0.16 and 0.10 when centered). The guests here are topically foreign in a way a natural minority modality is not. Both synthetic losses are therefore two to three times the natural 0.24 to 0.25 (d minus a in P1 and P2 on the S5 evaluation split), and their ratio does not decompose the natural loss. Ten hosts is a small sample, and the interval’s lower end is 0.021. A larger version of this control needs a corpus with single-modality folders of its own.

8 The loss reaches natural-language queries

The file queries of Sections 4, 6 and 7 measure whether a folder vector represents its own files. Session 12 asked, pre-registered, whether the same vectors answer a text query that is not a file. One query is the Zenodo record’s title (Q_{title} , 2,597 queries, one per S11 directory). The other is the title followed by the record’s description from the Zenodo metadata (Q_{desc} , the 2,399 records with a non-empty description, median 842 characters, cut by the study’s 2,000-token input rule like every text input). The relevant folder is the record’s own directory. Every directory is a candidate, and the representations are built from all of a directory’s embedded files, since the query is not one of them. The centered rows subtract the calibration split’s text-input mean from the query, and nothing is fitted on the queries. The hypothesis H12a was that the pooled centroid loses to c on image-heavy directories (image fraction at least 0.5) on Q_{title} under E1. It survived, at +0.182 [+0.158, +0.206] recall@5, with +0.263 [+0.236, +0.290] under E2 and +0.231 [+0.204, +0.256] and +0.359 [+0.333, +0.389] on Q_{desc} . Session 14 ran the same queries under E3, with no instruction on the query as on the files, and found +0.218 [+0.190, +0.244] on Q_{title} and +0.189 [+0.163, +0.216] on Q_{desc} . Session 15 ran them under E4, with the query prefix used on the files, and found +0.548 [+0.522, +0.577] on titles and +0.555 [+0.526, +0.584] on descriptions. Under E4 the pooled centroid finds an image-heavy folder from its title in the top five 0.7 percent of the time. Table 5 gives the levels.

Table 5: Recall@5 with the record’s title (Q_{title} , 2,597 queries) or title and description (Q_{desc} , 2,399 queries, the records with a non-empty description) as the query, at most one query per S11 record, all 2,597 directories ranked. T: text-heavy directories (image fraction below 0.5; 1,215 and 1,156 queries); I: image-heavy (1,382 and 1,243). Random recall@5 is 0.002. Vision-language embedders above, dual towers below; vectors per directory in parentheses. **a**: pooled; **ac**: pooled, centered (files and query); **tb2**: two blind clusters; **tb2c**: the same, centered; **c**: per-modality k -reps; **cc**: the same, centered; **d**: every file.

	E1 jina-embeddings-v4						E3 gme-Qwen2-VL-2B					
	Q_{title}			Q_{desc}			Q_{title}			Q_{desc}		
	all	T	I	all	T	I	all	T	I	all	T	I
a (1.00)	0.658	0.718	0.606	0.708	0.797	0.626	0.582	0.662	0.512	0.662	0.723	0.606
ac (1.00)	0.656	0.682	0.633	0.743	0.781	0.708	0.630	0.648	0.615	0.701	0.698	0.704
tb2 (2.00)	0.742	0.729	0.753	0.813	0.798	0.826	0.674	0.656	0.690	0.741	0.719	0.762
tb2c (2.00)	0.727	0.714	0.737	0.824	0.804	0.843	0.693	0.659	0.722	0.754	0.718	0.788
c (5.20)	0.792	0.796	0.788	0.850	0.843	0.857	0.718	0.705	0.729	0.776	0.755	0.795
cc (5.20)	0.777	0.777	0.776	0.860	0.845	0.874	0.733	0.708	0.755	0.788	0.762	0.813
d (9.87)	0.794	0.796	0.792	0.854	0.845	0.862	0.722	0.710	0.733	0.780	0.762	0.797

	E2 jina-clip-v2						E4 Nomic Embed v1.5					
	Q_{title}			Q_{desc}			Q_{title}			Q_{desc}		
	all	T	I	all	T	I	all	T	I	all	T	I
a (1.00)	0.429	0.549	0.324	0.413	0.600	0.238	0.205	0.431	0.007	0.240	0.492	0.005
ac (1.00)	0.378	0.430	0.331	0.496	0.533	0.463	0.399	0.562	0.255	0.476	0.631	0.331
tb2 (2.00)	0.565	0.523	0.603	0.602	0.574	0.627	0.563	0.554	0.572	0.621	0.623	0.619
tb2c (2.00)	0.455	0.472	0.439	0.586	0.567	0.605	0.566	0.577	0.556	0.627	0.641	0.613
c (5.20)	0.619	0.655	0.588	0.642	0.691	0.597	0.608	0.668	0.555	0.635	0.715	0.560
cc (5.20)	0.527	0.587	0.474	0.634	0.659	0.611	0.631	0.675	0.593	0.683	0.734	0.636
d (9.87)	0.629	0.675	0.588	0.653	0.714	0.597	0.628	0.686	0.577	0.660	0.739	0.587

Three things differ from the file-query picture (Figure 6). First, the loss is on both sides of a text query. With a file as the query, the pooled centroid lost mostly where the query’s modality was the directory’s minority. With a text query it loses on text-heavy directories too, by less. Under E1 on Q_{title} , **c** minus **a** is +0.078 [+0.059, +0.099] on text-heavy and +0.182 on image-heavy directories, and +0.046 and +0.231 on Q_{desc} . Under E2 the pairs are +0.106 and +0.263, +0.091 and +0.359, under E3 +0.044 and +0.218, +0.032 and +0.189, and under E4 +0.237 and +0.548, +0.223 and +0.555. In six of the eight runs the loss is largest in the $[.8, 1]$ bucket (+0.248 to +0.437). Under E4 it is largest in $[.5, .8]$, at +0.564 on both query sets, with +0.510 and +0.530 in $[.8, 1]$. A text query reaches a directory through its text files, and the pooled vector of an image-heavy directory has few of them left. In five of the eight runs the loss is smallest in $[.2, .5]$. In $[0, .2]$, where fewer than one file in five is an image, it is still +0.04 to +0.14 (the smallest, under E3 on titles, with an interval reaching zero). That is the dilution of Section 7, among the directory’s own texts.

Second, centering the query with the file-side text mean is not a dependable fix, and its effect depends on the encoder. On titles it does nothing under E1 (**ac** minus **a** −0.002 [−0.012, +0.008]), hurts under E2 (−0.052, and −0.119 on text-heavy directories) and helps under E3 and E4 (+0.049 [+0.038, +0.060] and +0.193 [+0.177, +0.209]). Where the query must cross the gap, on image-heavy directories, it helps on Q_{desc} under all four encoders (+0.082, +0.224, +0.098 and +0.327) and on titles under E1, E3 and E4 (+0.027, +0.103 and +0.248). On text-heavy directories it does nothing or hurts in the six runs under E1 to E3 (−0.014 to −0.119). Under E4 it helps there too

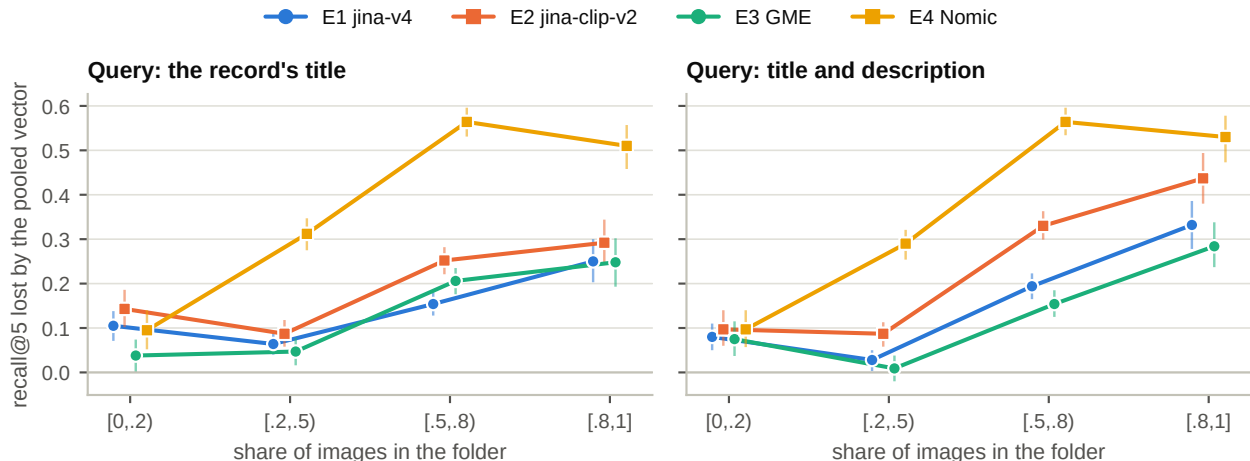


Figure 6: The loss with a text query that is not a file: recall@5 of c minus a , with 95 percent intervals, when the record’s title (left) or its title and description (right) is the query, by the share of images in the folder. The loss is largest in image-heavy folders, since a text query reaches a folder through its text files.

(+0.131 and +0.138), except in the most text-heavy bucket, where it does nothing. In no cell of Table 5 does it reach c . On image-heavy title queries ac reaches 0.615 against 0.729 for c under E3 and 0.255 against 0.555 under E4. The constant was fitted on file-side text inputs, whose mean is not the mean of a short title. A query-side constant would have to be fitted on held-out queries, which this study did not do by design.

Third, over all queries two vectors lose more against a text query than against a file under E1 and E2, about as much under E3 and less under E4. $tb2c$ is 0.065 [0.053, 0.078] below c on Q_{title} under E1 and 0.164 under E2 (0.026 and 0.056 on Q_{desc}), against 0.015 and 0.049 with file queries. Under E3 it is 0.025 [0.012, 0.040] below on titles and 0.022 on descriptions, against 0.023 with files. Under E4 it is 0.042 [0.027, 0.057] below on titles and 0.008 [−0.009, 0.025] on descriptions, against 0.054 with files. The uncentered pair $tb2$ is the better of the two on titles under E1 and E2 and on descriptions under E2. Under E3 and E4 the centered pair is ahead on both text query sets. With file queries it is ahead over all queries under E1 to E3 and level under E4 (0.640 each). Per-modality representatives stay close to the flat walk, less so under E4. Over all queries c is within 0.011 of d in the six runs under E1 to E3 (d minus c +0.002 [−0.002, +0.007] under E1 and +0.004 [−0.001, +0.010] under E3 on Q_{title}), at half the vectors. Under E4 it is 0.020 [0.013, 0.026] behind on titles and 0.025 [0.018, 0.033] on descriptions.

9 Validity checks

Two features of the corpus could inflate the loss. Files of one record often share a naming pattern or come in pairs, such as a figure and its PDF, and such siblings can make a folder easy to find from any one of its files. Some creators also deposited many records, so the directory bootstrap, which treats folders as independent, may give intervals that are too narrow. Session 17 tested both on S11 under all four encoders (Table 6), with the hypotheses fixed before any loss on the new subsets was computed, and checked the evaluation code with a second implementation under E1.

File names alone. A router that sees only file names scores each folder by the best cosine between the name of the query file and the names of the folder’s files, and ranks folders as the

Table 6: The loss under creator clustering and on the clean subset (session 17, S11): recall@5 of $\mathbf{c}$ minus $\mathbf{a}$ in the minority cells, with 95 percent intervals from the directory bootstrap and from resampling creator families. The clean subset (750 P1 and 877 P2 queries) holds the queries whose folder a filename router misses, without a same-stem sibling and outside four digitization series. Its intervals are creator-clustered.

Encoder	Cell	All queries			Clean subset	
		$\mathbf{c} - \mathbf{a}$	Directories	Creators	$\mathbf{c} - \mathbf{a}$	Creators
E1	P1	+0.253	[+0.214, +0.294]	[+0.202, +0.303]	+0.128	[+0.074, +0.180]
	P2	+0.169	[+0.136, +0.200]	[+0.119, +0.225]	+0.172	[+0.126, +0.222]
E2	P1	+0.500	[+0.462, +0.538]	[+0.456, +0.544]	+0.323	[+0.264, +0.383]
	P2	+0.348	[+0.312, +0.386]	[+0.255, +0.446]	+0.298	[+0.247, +0.343]
E3	P1	+0.205	[+0.164, +0.244]	[+0.152, +0.260]	+0.095	[+0.038, +0.150]
	P2	+0.158	[+0.125, +0.188]	[+0.111, +0.209]	+0.163	[+0.115, +0.218]
E4	P1	+0.442	[+0.398, +0.482]	[+0.396, +0.482]	+0.248	[+0.187, +0.304]
	P2	+0.433	[+0.396, +0.470]	[+0.317, +0.549]	+0.391	[+0.331, +0.444]

encoders do. It compares names by TF-IDF over character 3- and 4-grams of the lowercased base name, with the query file left out of its own folder. It puts the query’s own folder in the top five for 0.700 of the evaluation queries, 0.617 in P1 and 0.587 in P2. The pooled E1 vector reaches 0.692. No encoder here reads a file name, but files whose names give their folder away may resemble their siblings in content as well. Session 26 fused the name ranking with the vector ranking by reciprocal rank fusion ($k = 60$, post hoc, registered before it ran). Under the two vision-language embedders the loss keeps about two fifths of its size, and under the dual towers most of it remains. Under E1 $\mathbf{c}$ minus $\mathbf{a}$ is then +0.104 [+0.067, +0.144] in P1 and +0.067 [+0.041, +0.097] in P2 (creator-clustered). Under E3 it is +0.075 [+0.044, +0.109] and +0.057 [+0.035, +0.086], so H26d survived. Equal-weight fusion lifts the pooled vector (0.735 against 0.692 over all queries under E1) and the two dual towers. It lowers the representatives of the two vision-language embedders (0.780 against 0.796 under E1, 0.790 against 0.811 under E3). On GitHub, where the names alone reach 0.271 and 0.265, fusion nearly halves recall (0.706 to 0.387 for $\mathbf{c}$ on the first draw, 0.646 to 0.361 on the second). The fused loss there is +0.083 [+0.051, +0.116] and +0.067 [+0.044, +0.091] in P1s and P2s, owner-weighted.

The clean subset. A query is clean on three conditions. The filename router misses its folder (rank above five). Its folder holds no other file with the same stem and a different extension. Its record lies outside four digitization series, the deposits of four creator families that a count of P2 queries singled out before any loss was computed on them. H17b, pre-registered, required $\mathbf{c}$ minus $\mathbf{a}$ of at least +0.05 on the clean subset, with a creator-clustered interval above zero, in P1 and in P2 under E1 and under E3. It survived, and the same conditions hold under E2 and E4. On clean queries the image-query loss is about half of the loss over all queries under E1, E3 and E4 (+0.128 against +0.253, +0.095 against +0.205 and +0.248 against +0.442) and about two thirds of it under E2 (+0.323 against +0.500). The text-query loss does not shrink under E1 and E3 (+0.172 against +0.169 and +0.163 against +0.158) and shrinks a little under E2 and E4 (+0.298 against +0.348 and +0.391 against +0.433). Removing the four series alone raises it under all four encoders (to +0.212 under E1). The image queries whose folder the file names give away are 1,209 of the 1,959 in P1, and they lose more per query than the clean ones, so most of what the pooled vector loses on image queries is lost on them. That fits a lost match between sibling images, such as images of one series that resemble each other more than anything else. The subsets differ in other ways too, so this reading does not decompose the loss. In the majority cells the loss on clean

queries is smaller as well, and under E1 the image-side one is gone (-0.002 [$-0.029, +0.029$] in M1, against $+0.080$ over all queries).

Creator-clustered intervals. On the Zenodo sets, outside this section and Sections 5 and 11, the intervals in this paper resample directories. Here a record’s family is its first creator, lowercased. The 2,597 S11 records come from 1,814 families, and about 47 effective families, the inverse of the summed squares of the families’ shares of the queries, stand behind the 2,146 P2 queries (204 behind P1). Session 17 resampled each cell’s families instead, 1,000 times, recomputing the difference over all queries of the drawn families. H17a, pre-registered, required the clustered interval of c minus a to lie above zero in P1 and in P2 under each encoder. It survived under all four. Of the two minority cells P2 widens more, to $[+0.317, +0.549]$ under E4. The other verdicts re-read with clustered intervals do not move. The H15c differences stay above zero, as does E2 minus E1 (E4 minus E1: $[+0.142, +0.239]$ in P1 and $[+0.184, +0.346]$ in P2). No interval of c minus d lies entirely below -0.02 under any encoder. The lowest bound is -0.027 , under E4 in P2, with an upper bound of -0.010 . Under E1 $tb2c$ still passes the H11a rule (P1 $[-0.063, -0.014]$, M2 $[-0.064, -0.010]$). On S5 under E1 the clustered intervals of c minus a in the minority cells are $[+0.198, +0.282]$ and $[+0.196, +0.274]$. With the title as the query, c minus a on image-heavy folders is $+0.182$ $[+0.126, +0.242]$ under E1, and $+0.134$ $[+0.101, +0.175]$ on the 744 image-heavy title queries that a filename router misses ($+0.239$, $+0.148$ and $+0.493$ under E2, E3 and E4). Weighting each family equally gives smaller estimates, with clustered intervals: under E1 $+0.133$ $[+0.103, +0.163]$ in P1 and $+0.137$ $[+0.110, +0.164]$ in P2, and on the clean subset $+0.059$ $[+0.025, +0.094]$ and $+0.115$ $[+0.083, +0.148]$; under E3 $+0.105$ $[+0.079, +0.135]$ and $+0.123$ $[+0.093, +0.153]$, and on the clean subset $+0.056$ $[+0.023, +0.090]$ and $+0.096$ $[+0.060, +0.132]$. Creators with many queries contribute larger losses than the average creator. For image queries this is the most conservative reading the study offers, with creators counted once and the queries whose names give the folder away removed. Under it the image-query loss under the two vision-language embedders is about 0.06, with intervals that reach 0.02, below the smallest effect of interest of Section 10. Part of the drop under equal weighting comes from the lone files. With the query’s folder holding a sibling of its input group (1,545 of the 1,959 P1 queries and 1,673 of the 2,146 P2 queries) the query-weighted loss under E1 is $+0.324$ and $+0.218$, and with families weighted equally $+0.266$ and $+0.239$ (Section 10).

A second implementation. The earlier reproductions of the metric shared code with the evaluation script. For H17c an agent wrote a second implementation from a written specification, without reading the first one’s code. On the E1 cache it gives the same rank as the evaluation script for every one of the 17,140 queries in rows a , c and d .

10 A second source: directories of GitHub repositories

Every folder above is a Zenodo record, from one search of one repository, selected for holding both an image type and a document type. Sessions 19 and 24 repeated the test on folders that people built for their own work, the directories of public GitHub repositories, on the conditions an earlier round of review of this study set for such a replication. Days from 2016 to 2025 were drawn at random, and for each day the pool holds the repositories created that day with at least five stars, as the GitHub search API returns them without a token [20]. A repository is usable if it is not a fork, is at most 200 MB and carries one of nine permissive licences (MIT, Apache-2.0, the two BSD licences, ISC, Unlicense, 0BSD, CC0-1.0 and CC-BY-4.0). A folder is a directory of the default branch at any depth with its direct children, read from the repository’s tarball, outside hidden,

vendored and dependency directories. Kept files follow the extension table and the 25 MB cap of the Zenodo sets. A directory with 3 to 100 kept files is eligible, and it is mixed if it keeps a file with an image extension and at least one other kept file. From each repository read, the scored set takes at most two mixed and two other eligible directories at random. Ground truth, input rules and representations are those of Section 3, with the same 20 percent calibration split by directory. The split’s directories remain candidates and give no queries, so the evaluation queries below are fewer than the queries with a vector in Table 1.

The readings were tightened for this source. The minority cells keep only the queries whose folder still holds another file of the query’s input group once the query is left out (P1s and P2s). A lone image among texts has no sibling to be found through under any representation. The owner, the account that holds the repository, takes the place of the creator family. Every registered test is the mean over owners of the owner’s mean, with percentile intervals from 10,000 resamples of owners, and a test with fewer than 100 owners is inconclusive. A loss is confirmed if the lower end of its interval is at least +0.05, a smallest effect of interest fixed in advance [32]. It is below +0.05 if the upper end is under it, present with the size open if the lower end is above zero, and inconclusive otherwise. The three primary tests of a session are read on 98.33 percent intervals (Bonferroni over three) and the secondary ones on 95 percent intervals. Source code is text to the encoders and is reported apart, with the folders that hold none of it as a like-for-like stratum.

Mixed directories are rare. Session 19 read 4,322 repositories of a pool of 5,517 usable ones. Of the 35,970 eligible directories they hold, 1,501 are mixed (4.2 percent), with 11.2 percent at the root of a repository and 3.0 percent at depth two or more. The Zenodo pool was selected for mixed records, and here about one eligible directory in 24 is mixed. The scored set over-samples them on purpose. It holds 1,600 directories (1,000 mixed, 600 other) of 1,079 repositories and 1,053 owners, with 12,656 files and 7,372 evaluation queries from 858 owners under each of E1, E4 and E2.

The first draw. Table 7 gives the registered tests. Under E1, *c* minus *a* is +0.193 in P1s and +0.166 in P2s, with 98.33 percent intervals of [+0.099, +0.295] and [+0.080, +0.260]. Both lower ends are above +0.05, which the rule calls confirmed, but the two cells hold 86 and 98 owners, fewer than the 100 the brief required, so H19a is inconclusive. The rule was not relaxed after the fact. Where *c* is a strict compression of its folder (a label keeps more than three files once the query is left out), *c* minus *d* is −0.002 [−0.014, +0.008] over 512 owners, so *c* is not inferior to opening every file (H19c). Under E4 and E2, read by the same rules without a verdict of their own, the loss is +0.460 and +0.408, and +0.471 and +0.301, every interval above zero and every cell short of the owner minimum. *c* is again not inferior to *d*. On the same queries both towers lose more than E1, as on Zenodo, by +0.126 and +0.281 under E4 and +0.136 and +0.137 under E2 in P1 and P2. The file names do not explain the loss. On the clean subset, where neither filename router of Section 9 finds the folder at rank five and no file shares the query’s stem, it is +0.188 and +0.142 under E1. Both 95 percent intervals lie above zero, and both cells are short of the owner minimum. By kind of query file it is largest for source code among images, +0.298 over 80 queries from 37 owners, against +0.096 for prose, both far short of the owner minimum. On all of P1 and of P2, lone minority files included, it is +0.069 and +0.071 (334 and 231 owners), present with the size open. More than half of the P1 queries there are lone images among texts, which no representation can reach through a sibling.

The second draw. Session 24 drew again under the same rules, from 8,307 repositories that session 19 had not read, with twice the mixed directories: 2,600 directories (2,000 mixed) of 1,885

repositories and 1,826 owners, 21,854 files and 12,951 evaluation queries from 1,473 owners. It was written after session 19’s E1 table had been read, as a replication sized for the owner minimum, and its brief rests the claim on this draw alone. Here **c** minus **a** is +0.210 [+0.144, +0.279] in P1s from 172 owners and +0.196 [+0.133, +0.260] in P2s from 197, so H24a is confirmed in both cells. Where **c** is a strict compression it is not inferior to **d** (−0.002 [−0.010, +0.006], 918 owners; H24c). On the clean subset the loss is +0.180 and +0.209, both confirmed. Pooled over the two draws, with an owner present in both counted once, it is +0.205 [+0.152, +0.263] and +0.186 [+0.135, +0.238], from 258 and 294 owners. That is the size on this source, and session 19’s estimates lie inside the second draw’s intervals. Read the same way, with the query’s folder holding a sibling of its input group and creator families weighted equally, the Zenodo loss under E1 is +0.266 [+0.224, +0.313] and +0.239 [+0.199, +0.276] (session 25, post hoc), so the GitHub loss is somewhat smaller. By kind of query file it is +0.254 for source code among images, +0.236 for configuration files and +0.134 for prose. Mixed directories are 5.0 percent of the eligible directories of the repositories read for this draw. As on Zenodo, a folder is two clusters. Under E1 the same-folder cosine is 0.504 between an image and a text, against 0.717 between two images and 0.736 between two texts.

At the natural mix (session 25, post hoc). The scored sets over-sample mixed directories on purpose. Over all evaluation queries of both draws, owner-weighted, **c** minus **a** is +0.019 [+0.012, +0.026]. It is +0.030 [+0.022, +0.039] on the queries of mixed directories and −0.017 [−0.026, −0.008] on the others. Weighted to the eligible share of mixed directories (4,787 of 101,954), directory-weighted within each stratum and with the scored sets’ folders still as candidates, the net is −0.014 [−0.022, −0.005]. Those candidates over-sample mixed folders. Natural-mix candidates (session 26) take every folder of a draw that is not mixed and add mixed ones in their eligible share, about 630 in all. Against them the net is −0.006 [−0.012, −0.000]. The loss also grows with the number of candidate folders. With the own folder and $n - 1$ others drawn at random, the expected loss in P1s and P2s is +0.013 and +0.049 among 10 folders and +0.138 and +0.161 among 100.

Representatives for mixed folders only (session 26, post hoc). A file system can tell a mixed folder by its file types alone. Session 26 tested that policy, registered before it ran on the same two draws. It keeps **c** for a folder that holds a file labelled image and a file of another label and **a** for the rest. At the natural mix, with the candidates above and 98.33 percent intervals, it keeps most of the minority gain. Against **a** it gains +0.143 [+0.094, +0.197] in P1s (258 owners) and +0.180 [+0.134, +0.228] in P2s (294 owners), so H26a is confirmed. Representatives for every folder gain +0.164 and +0.192 there. Over all directories it is level with the mean, +0.001 [+0.0004, +0.002], not inferior by the registered margin of 0.005 (H26b). Against **c** for every folder it is +0.007 [−0.0001, +0.015] over all directories, so H26c, that the policy is better, failed by its bound, and by that little. In P1s and P2s it gives up 0.021 and 0.012 against **c**. For 98 percent of those queries the own folder is represented alike under both, so that difference comes from competitors kept as means. The verdicts hold for the registered candidate set of about 630 folders. With 100 candidates at the natural mix the policy gains +0.059 and +0.135 in P1s and P2s, against +0.075 and +0.162 for **c**, and it is level with the mean over all directories. Its 98.33 percent lower bound in P1s is then +0.025. With the scored sets’ candidates it gains +0.191 and +0.181 and is −0.004 [−0.009, +0.001] over all directories, just outside the margin. On GitHub two group means in place of **c** for the mixed folders do as well. The mean added to the representatives of every folder is +0.004 [+0.000, +0.009] over all directories and keeps nearly all of the gain, +0.154 and +0.192 (95 percent intervals, no registered test). Under E4 and E2 on G1 the policy is also above the mean over all directories, by +0.008 and +0.004.

Table 7: The registered tests on GitHub directories: owner-weighted c minus a in the minority cells that keep a sibling of the query’s group (P1s, P2s), and c minus d where c is a strict compression of its folder. Point estimate, interval from 10,000 resamples of owners (98.33 percent for the primary tests, 95 percent for the clean subset) and owners. A test with fewer than 100 owners is inconclusive. E4 and E2 are read by the same rules without a verdict of their own.

Draw, encoder	P1s: $c - a$	P2s: $c - a$	$c - d$, c not whole	P1s clean	P2s clean
S19, E1	+0.193 [+0.099, +0.295], 86	+0.166 [+0.080, +0.260], 98	−0.002 [−0.014, +0.008], 512	+0.188 [+0.090, +0.293], 60	+0.142 [+0.063, +0.225], 82
S19, E4	+0.460 [+0.347, +0.574], 86	+0.408 [+0.303, +0.517], 98	−0.002 [−0.011, +0.007], 512	+0.388 [+0.277, +0.504], 60	+0.328 [+0.236, +0.424], 82
S19, E2	+0.471 [+0.357, +0.585], 86	+0.301 [+0.204, +0.401], 98	+0.003 [−0.005, +0.011], 512	+0.389 [+0.276, +0.508], 60	+0.272 [+0.185, +0.364], 82
S24, E1	+0.210 [+0.144, +0.279], 172	+0.196 [+0.133, +0.260], 197	−0.002 [−0.010, +0.006], 918	+0.180 [+0.113, +0.250], 118	+0.209 [+0.147, +0.275], 155
Pooled, E1	+0.205 [+0.152, +0.263], 258	+0.186 [+0.135, +0.238], 294	−0.002 [−0.008, +0.005], 1,425	+0.182 [+0.127, +0.240], 178	+0.186 [+0.137, +0.237], 236

Written queries. A commit subject is the committer’s own words about a change. For each scored directory, session 19 took up to three subjects of commits that changed only files directly in it. It removed every token that holds a path, a file name or an extension of the study’s table, and dropped merges, reverts, stock subjects and subjects of fewer than four words. That gives 1,162 queries over 522 directories and 440 owners. On image-heavy directories, the test of the loss with a text query (H19d), the flat walk passes its gate, reaching recall@5 0.254 against 0.066 for the file names alone. There c minus a is +0.010 [−0.061, +0.078] under E1, from 122 queries and 66 owners, and +0.101 [+0.040, +0.172] under E4 and +0.146 [+0.071, +0.230] under E2, all three inconclusive by the owner minimum. The second draw has 1,827 commit-subject queries from 731 owners. On its image-heavy directories the gate passes again (0.258 against 0.122), and under E1 the loss there is +0.117 [+0.065, +0.169] from 138 owners, confirmed (H24d). Pooled it is +0.083 [+0.041, +0.126] from 203 owners, present with the size open and less than half the pooled loss on held-out files.

11 A written folder abstract

RAPTOR and OpenViking represent a group of items by a summary that a language model writes, and embed the summary (Section 2). Session 18 tested that design in a strong form on S11 under E1 and E3. Qwen3-VL-8B-Instruct wrote a caption of every image and of the first page of every scanned PDF, and the first 500 tokens of every text were extracted. From these and the name words of each file (the runs of two or more letters in its base name), the same model wrote one abstract per folder (L1). It is at most 4,000 characters of prose and was asked to cover every kind of file, images as well as documents. From L1 the model wrote a one-sentence abstract (L0) of at most 256 characters. The model was given no title, description, record identifier or full file name. Every token that contains a digit was removed from the captions and from both abstracts, so names with digits in them, such as marker names, formulas and sample codes, drop out as well. The L1 abstracts have a median length of 1,532 characters. The abstract’s embedding is the folder’s one vector s_1 (s_0 for L0), and s_{1c} adds it to the representatives of c . The queries are not files. One set is the 2,597 record titles. The other is one-sentence descriptions of single images, written by Pixtral-12B, a model from outside the Qwen family, the way someone searching for the image would describe it. Any token that contains a digit or repeats one of the image’s own name words of three or more letters was removed. They cover all 1,959 P1 query images and 2,000 images in image-heavy folders (M1), and each image stays in its folder. Folder representations use all of a folder’s files, and intervals are creator-clustered. The a , c and d ranks on the title queries and on the 2,399 title-and-description queries equal those of Section 8 rank for rank under both encoders.

A described image. H18b, pre-registered, required c minus $s1$ and c minus a of at least $+0.05$, with intervals above zero, on the descriptions of P1 images, gated on the flat walk d reaching recall@5 0.25 (it reaches 0.785 under E1 and 0.751 under E3). It survived under both encoders. Under E1 the abstract finds the folder of a described image as often as the pooled vector to three decimals, 0.490 each, against 0.781 for c . There c minus $s1$ is $+0.291$ [$+0.259, +0.322$] and c minus a is $+0.290$ [$+0.260, +0.324$]. Under E3 $s1$ reaches 0.404, a 0.457 and c 0.747 ($+0.344$ [$+0.310, +0.380$] and $+0.290$ [$+0.259, +0.322$]). With one folder per family the differences are $+0.295$ and $+0.297$ under E1 and $+0.352$ and $+0.301$ under E3. The abstract loses majority images as well. On M1 descriptions it reaches 0.458 under E1 against 0.732 for c , a difference of $+0.274$ [$+0.239, +0.313$], and under E3 the difference is $+0.300$ [$+0.260, +0.341$]. The pooled vector loses little there ($+0.069$ [$+0.043, +0.100$] and $+0.049$ [$+0.022, +0.080$]). The abstracts do not leave the images out. In an exploratory count, not pre-registered, the L1 abstracts of 809 of the 832 P1 folders use a word for a picture (image, photo, photograph, micrograph, figure, picture, illustration, diagram, chart, plot, map or graph). They describe them in general terms, and an abstract that summarises a folder does not keep each member findable. On P1 descriptions L0 loses more ($+0.424$ and $+0.425$ for c minus $s0$), and adding $s1$ to c changes nothing there ($+0.001$ under each encoder). The flat walk adds little over c on P1 descriptions ($+0.005$ under E1) and more on M1 ($+0.035$ [$+0.021, +0.050$]).

Titles. H18a compared $s1$ with c on the titles of image-heavy folders. Three outcomes were fixed in advance for each encoder: above c (a difference of at least $+0.05$ with its interval above zero), below c (at most -0.05 , interval below zero) or between. The hypothesis counts as split if the encoders disagree. They disagree. Under E1 the abstract is level with c , 0.781 against 0.788 (-0.007 [$-0.049, +0.038$] over 1,382 queries from 813 families). Under E3 it falls below, 0.467 against 0.729 (-0.263 [$-0.433, -0.096$]). The four digitization series pull the E1 difference down. On their 396 queries $s1$ minus c is -0.096 [$-0.143, -0.008$]. Without them it is $+0.029$ [$+0.005, +0.055$], and with one folder per family $+0.042$ [$+0.020, +0.064$], so on the other titles the E1 abstract is slightly above c . Under E3 it stays below without the series (-0.100 [$-0.133, -0.067$]). Adding the abstract to c gains a little under both encoders. Over all titles $s1c$ minus c is $+0.025$ [$+0.018, +0.033$] under E1 (0.816 against 0.792) and $+0.010$ [$+0.006, +0.016$] under E3. Over all titles no row is above $s1c$ under E1. Under E3 it is not the best row. BM25 reaches 0.746, the flat walk and the per-modality representatives built with captions in place of images reach 0.741 and 0.735 (next paragraph), and the centered representatives cc of Table 5 reach 0.733, against 0.729. L0 is weaker than L1 under E1 and stronger under E3. Against c over all titles it is -0.043 and -0.068 , where L1 is $+0.004$ and -0.171 . A second abstract allowed to use the full file names gains a little over $s1$ on titles ($+0.020$ under E1 and $+0.074$ under E3) and nothing on P1 descriptions (-0.006 under E1). BM25 over each folder’s file names, captions and text heads, the lexical ceiling on the abstract’s evidence, reaches 0.746 over all titles, -0.050 [$-0.104, +0.018$] against $s1$ under E1 and $+0.199$ [$+0.106, +0.314$] under E3, and 0.560 on P1 descriptions, -0.220 [$-0.255, -0.190$] against c under E1. With the title and the description as the query the picture is the same. Both $s1$ and c reach 0.850 over all queries under E1, and under E3 $s1$ reaches 0.665 against 0.776.

Captions in place of images. Replacing every image vector by the embedding of its caption (rows ta and tc) puts a folder’s images on the text side of the gap. Over all titles and on P1 descriptions it does not help the pooled vector. Over all titles ta minus a is -0.005 [$-0.016, +0.007$] under E1 and $+0.013$ [$-0.011, +0.041$] under E3. On P1 descriptions it is -0.009 [$-0.020, +0.001$] under E1 and -0.045 [$-0.060, -0.027$] under E3, where the captions lower it. Per-group representatives in that all-text space still recover the loss. Over all titles tc minus ta is $+0.136$ [$+0.098, +0.187$]

and +0.141 [+0.105, +0.183], and on P1 descriptions it is +0.247 [+0.220, +0.277] and +0.359 [+0.325, +0.398]. The loss therefore does not need the modality gap, as the control of Section 7 also showed. Neither result apportions the natural loss between the two.

12 What fixes it, and at what budget

No single vector in the encoder’s space. Session 9 searched for the most compact folder representation that keeps the minority without losing the majority. Every parameter was chosen on the calibration split’s 4,388 queries, and the hypotheses were read on the 18,742 evaluation queries of S5. Five families were tried. F1, weighted pooling: the unit sum over the two input groups of $n_g^\alpha \text{mean}_g$, where group g has n_g files with mean vector mean_g and $\alpha \in \{0, 0.25, 0.5, 0.75, 1\}$, uncentered and centered (so **a**, **ab**, **ac** and **acb** are its corners). F2, the same pooling after a linear cross-modal alignment of the image group fitted on the calibration split: orthogonal Procrustes and ridge regression fitted on the pairs of per-folder group means, and CORAL fitted on the covariances of the image and the text files. F3, labelled centroids: one per input group (**b2**, **bc2**) or one per modality label on centered vectors (**bc**). F4, a partitioned vector of twice the encoder’s dimension holding both centered group means, scored by one inner product with a query placed in its own slot and a weighted copy in the other. F5, the label-blind control: k -means over the folder’s files at budgets of two, the number of groups present, and **c**’s budget, uncentered and centered. The selection rule was the configuration maximising the minimum over the four cells of its recall@5 minus **c**’s.

The F1 and F2 rows were scored twice. A number audit of this paper found that the evaluation script had ranked every one of them too low. Their folder vectors were built in double precision, and the query’s own folder then lost a comparison against a single-precision copy of its own score in about half the queries. That halved recall@1 and lowered recall@5 by 0.007 to 0.011 over all queries (0.005 to 0.013 per cell), in the direction that favoured the hypothesis below. Session 13 fixed the script. It checked that the four corners of F1 now rank exactly as **a**, **ab**, **ac** and **acb** and that every other row of session 9 reproduces rank for rank, and it re-ran the selection and the test with the hypothesis and its kill rule unchanged. The selections did not change, and the numbers below are the corrected ones.

The pre-registered hypothesis H9a is that no F1 or F2 configuration selected on the calibration queries is within 0.02 of **c** in both minority cells and both majority cells. It is dead if all four intervals of a selected configuration reach above -0.02 . It survived. The selected F1 vector, centered pooling at $\alpha = 0.5$, is below **c** by 0.048 [0.021, 0.077] in P1, 0.045 [0.019, 0.070] in P2, 0.027 [0.009, 0.047] in M1 and 0.047 [0.036, 0.061] in M2. The selected aligned vector (CORAL at $\alpha = 0.75$, chosen on a two-fold cross-fitted grid) is below by 0.030 [0.000, 0.062] in P1, 0.086 [0.059, 0.110] in P2, 0.001 in M1 and 0.049 [0.038, 0.062] in M2. Equal weighting of the two groups ($\alpha = 0$, centered: **acb**) puts the minority cells above **c** at the point estimate (+0.023 and +0.016; the P2 interval includes zero) and the majority cells 0.065 and 0.092 below it. Raising α moves the loss from M2 to the minority cells, and no α is within 0.02 of **c** in all four cells. Procrustes and CORAL do not raise the same-folder image-text cosine out of sample (0.205 centered, 0.201 after Procrustes, 0.190 after CORAL, against 0.44 to 0.57 within a group). Ridge regression raises it (0.23 to 0.31) by pulling the image vectors toward the text mean, and it loses the image queries with them (P1 recall@5 0.52 at its best setting and 0.005 at its worst). The alignments were also checked for leakage with two-fold cross-fitting. Procrustes at $\alpha = 0.5$ fitted in-sample reached 0.89 recall@5 in P1 on the calibration queries (**c**: 0.71) and 0.54 cross-fitted, so an aligned single vector that looked good in-sample did not hold up out of sample. The partitioned vector F4 lost 0.077 in P1 to **c**. Across

the 40 distinct single vectors scored on the evaluation split (a, ac, ab, acb and every F1 and F2 configuration) none has all four intervals reaching above -0.02 , and the best minimum over the four cells is -0.048 . The binding cell is M2, where every one is at least 0.042 below c on text-like queries in text-heavy folders, with no interval reaching above -0.030 . The closest by count is CORAL at $\alpha = 0.25$, no more than 0.02 below c in three cells and 0.065 below in M2. The five single vectors that are no more than 0.02 below c in both minority cells are 0.055 to 0.113 below it in M2.

Two vectors per folder. Two compact representations came close on S5. Centered per-label centroids **bc** (2.55 vectors per folder) matched c by the kill rule with a 0.023 loss in M1. Two label-blind centered clusters **tb2c** (2.00 vectors) came within 0.013 of c in all five cells and matched it at recall@10 (0.816 against 0.813). One centered centroid per input group (**bc2**, 1.96 vectors) did not, at -0.034 $[-0.045, -0.024]$ against c in M2. **tb2c** was a post hoc pick from a grid, so S11 tested it pre-registered, on folders that took part in no selection, under E1 and E2 (Table 3). Under E1 it survives the kill rule of H11a (no interval of **tb2c** minus c entirely below -0.02), by little, with -0.015 $[-0.022, -0.007]$ over all queries, -0.037 $[-0.058, -0.014]$ in P1, $+0.036$ in P2, -0.005 in M1 and -0.032 $[-0.046, -0.019]$ in M2. Two blind clusters close 85 percent of the pooled centroid’s gap in P1 and half of it in M2. Under E2 the second clause of the pre-registered hypothesis fails, with **tb2c** minus c at -0.105 $[-0.130, -0.080]$ in P1, -0.052 in P2 and -0.063 in M2. Under E3, where session 14 read it by the same rule as a reported secondary, it falls short as well. Its -0.058 $[-0.077, -0.039]$ in P1 and -0.040 $[-0.054, -0.029]$ in M2 lie entirely below -0.02 (-0.023 $[-0.030, -0.016]$ over all queries). Under E4, read by the same rule, it falls short clearly, at -0.117 $[-0.147, -0.090]$ in P1, and four of its five intervals lie entirely below -0.02 . Two vectors meet the margin under one encoder of the four. The labelled pair **bc2** loses at most 0.032 or gains in the minority cells under all four encoders, and it loses 0.049 to 0.112 on the majority text queries (M2; Table 3). One centroid per modality is too coarse for the majority side, and k -means within the modality is what keeps it. Blind clustering does find the modalities in all four spaces (uncentered purity $0.918, 0.982, 0.898$ and 0.998 under E1 to E4, and $0.885, 0.895, 0.879$ and 0.932 after centering). Under E2 and E4, where centering lowers the purity most, the uncentered pair **tb2** is ahead of **tb2c** in both minority cells (in P1, 0.494 against 0.450 under E2 and 0.415 against 0.331 under E4). In M2 the uncentered pair is still below c by more than the margin, by 0.087 under E2 and 0.062 under E4. With the wider gap two vectors are too few, centered or not. Against a text query the two-vector budget is tighter still under E1 and E2, about as tight under E3 and looser under E4 (Section 8).

Five per-modality representatives stay close to the flat walk under all four encoders. c , per-modality k -means with at most three representatives per label, uses 5.2 vectors per folder on S11, a little over half the files. Its point estimates are within 0.018 of the flat walk d over all queries and in the four cells under all four encoders (Table 3 and Figure 7), and within 0.013 on S3 and S5. The rule pre-registered under E3 and E4, that no interval lie entirely below -0.02 , is not a test of non-inferiority. Read as one, with the lower bound at or above -0.02 , c passes over all queries under all four encoders, with the lowest bound at -0.015 , and in every cell under E3. One cell falls just below under E1 (M1, -0.021) and one under E2 (M2, -0.025), and three under E4, down to -0.024 (session 25). On the 77 percent of S11 queries whose folder c strictly compresses, c minus d is -0.006 $[-0.011, -0.002]$ over all queries under E1 (post hoc). Its centered form cc is above d in the minority cells under E1 and E2, and in P2 under E3 and E4. Two blind clusters pass their pre-registered margin under E1, by the rule and with point losses of 0.03 to 0.04 in two cells, and fall short of it under the other three encoders, among them E3, a vision-language embedder

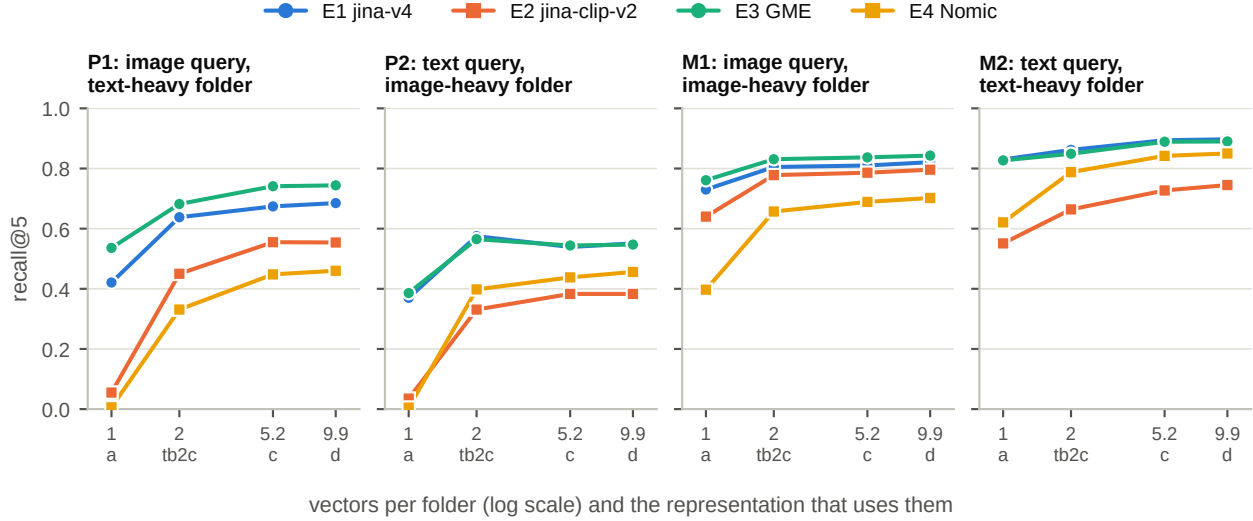


Figure 7: What each budget keeps: recall@5 on S11 by cell and encoder for the pooled centroid **a** (one vector), two blind centered clusters **tb2c**, per-modality k -means **c** and every file **d** (Table 3). Circles are vision-language embedders and squares dual towers. Most of what the flat walk gains over one vector is already there at **c**.

like E1.

Budgets from one to eight centroids. Session 27 varied the budget on the confirmation set under all four encoders, with its reading registered before the new rows were computed (Figure 8). The set had been read before and some rows, among them **c**, were known, so we mark the reading post hoc. Three families were scored, uncentered and centered. The labelled family keeps $\min(k, n_m)$ centroids per label for k from 1 to 5, so that $k = 3$ is **c**. The blind family keeps K centroids over the whole folder for K in 2, 3, 4, 5, 6 and 8, and a third family keeps as many blind centroids in each folder as the labelled one does. A budget qualifies under an encoder when the lower end of the interval of its difference from **d** is at or above -0.02 over all queries and in each of the four cells. Four per label, 6.02 vectors per folder, is the smallest labelled budget that qualifies under all four encoders, uncentered or centered. Three per label qualifies uncentered under E3 only. As the reading of session 25 found, it falls just below the margin in M1 under E1 (-0.021), in M2 under E2 (-0.025) and in three cells under E4, down to -0.024 . Centered, three per label qualifies under E1, E2 and E3. One and two per label qualify under no encoder. Uncentered blind centroids qualify under E4 at no K up to eight. Centered, they need eight (6.13 vectors per folder) under all four encoders and four under E1. With about six vectors per folder the centered rows reach the uncentered flat walk **d** over all queries under every encoder. Four centered centroids per label minus **d** is $+0.010$ $[+0.006, +0.014]$ under E1, $+0.007$ $[+0.003, +0.011]$ under E2, $+0.003$ $[-0.000, +0.005]$ under E3 and $+0.004$ $[+0.000, +0.008]$ under E4. On the second GitHub draw, read under E1 without a verdict, no budget qualifies in the minority cells P1s and P2s, where the intervals resample owners and are wider. There four per label minus **d** is -0.015 $[-0.031, -0.002]$ and -0.018 $[-0.033, -0.005]$. Section 16 reads what the labels buy at the same number of vectors.

The budget in bytes. A metadata field counts bytes. In float32 the representatives of **c** take 42,594, 21,297, 31,945 and 15,973 bytes per folder on average under E1 to E4. Session 20 stored every vector of **a**, **c** and **d** in five formats and scored the S11 evaluation queries again under the

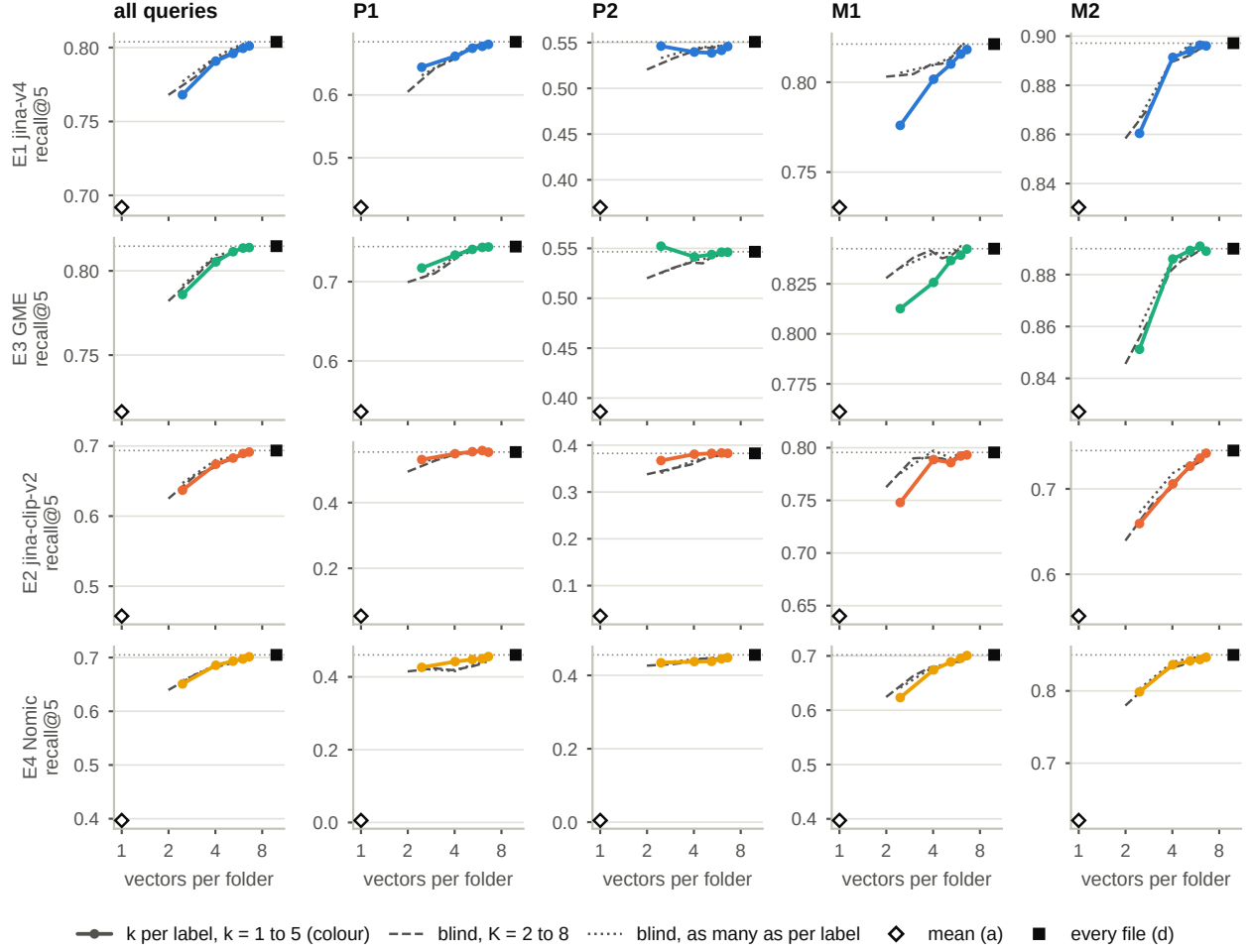


Figure 8: Budgets from one to eight centroids (session 27, confirmation set, post hoc, uncentered rows): recall@5 against the mean number of vectors per folder, by encoder (rows) and cell (columns). Coloured lines with dots keep k centroids per label for k from 1 to 5, the third dot being c . Dashed lines keep K blind centroids for K from 2 to 8, and dotted lines as many blind centroids as the labelled row keeps in each folder. The open diamond is the pooled centroid a , and the filled square, with its level as a dotted line, is every file d .

Table 8: c in a metadata field of fixed size (session 20, S11). For each budget per folder, the most precise format in which every folder’s c fits and its recall@5 over all queries, in P1 and in P2. The formats f32 and f16 are single and half precision, int8 is one byte per dimension, int4 four bits and bin one bit. None means that no format fits. The last two rows give c and a in float32 without a budget. Vision-language embedders above, dual towers below.

Budget	E1 jina-embeddings-v4				E3 gme-Qwen2-VL-2B			
	format	all	P1	P2	format	all	P1	P2
2 KiB	none				none			
4 KiB	bin	0.796	0.676	0.543	bin	0.810	0.743	0.558
8 KiB	bin	0.796	0.676	0.543	bin	0.810	0.743	0.558
16 KiB	int4	0.794	0.673	0.540	int4	0.807	0.742	0.547
64 KiB	f16	0.796	0.674	0.539	f16	0.811	0.741	0.544
c , no budget	f32	0.796	0.674	0.539	f32	0.811	0.741	0.544
a , no budget	f32	0.692	0.421	0.370	f32	0.716	0.536	0.386

Budget	E2 jina-clip-v2				E4 Nomic Embed v1.5			
	format	all	P1	P2	format	all	P1	P2
2 KiB	bin	0.675	0.558	0.376	bin	0.648	0.377	0.431
4 KiB	bin	0.675	0.558	0.376	bin	0.648	0.377	0.431
8 KiB	int4	0.676	0.554	0.379	int4	0.692	0.443	0.441
16 KiB	int8	0.683	0.554	0.382	int8	0.693	0.447	0.438
64 KiB	f32	0.683	0.555	0.383	f32	0.693	0.448	0.438
c , no budget	f32	0.683	0.555	0.383	f32	0.693	0.448	0.438
a , no budget	f32	0.457	0.055	0.035	f32	0.397	0.006	0.005

four encoders. The formats are float32, float16, one byte per dimension (int8: the vector divided by its largest absolute entry, multiplied by 127 and rounded), four bits (int4: the same with 7) and one bit (bin: the sign). A stored vector is decoded and normalised before scoring, and the query stays in float32, since it is embedded at search time. Nothing is fitted. Shakir et al. [57] describe such quantisation of finished embeddings. They threshold at zero for one bit, as we do, and map each dimension’s range to one byte with a calibration set, where we scale each vector by its own largest entry. Binary codes can also be learned [65]. In float32 the a , c and d rows reproduce the ranks behind Table 3 query for query under all four encoders.

Three hypotheses were fixed before the run. H20a applies the rule of H15b at one byte per dimension: no interval of c at one byte minus d in float32 may lie entirely below -0.02 over all queries or in P1, P2, M1 or M2. It survived under all four encoders. One byte per dimension costs nothing measurable. Against c in float32 the difference is within 0.001 in every cell under every encoder. c then takes 10,648, 5,324, 7,986 and 3,993 bytes per folder on average under E1 to E4, and at most 28,672. H20b applies the same rule at one bit per dimension. It survived under E1 and E3 and died under the two dual towers, under E2 in M2 (-0.033 [-0.041 , -0.025]) and under E4 over all queries (-0.057 [-0.062 , -0.051]), in P1 and in M1. Under E4 one bit costs every stored vector. Over all queries it changes c by -0.045 and the flat walk by -0.048 [-0.053 , -0.044] against their float32 forms, and at one bit c is 0.009 below the flat walk. H20c required c at one bit per dimension to beat the pooled centroid in float32 in P1 and in P2. It survived under all four encoders, with differences from $+0.172$ to $+0.503$. At one bit c takes 1,331, 666, 998 and 499 bytes per folder on average under E1 to E4, against 8,192, 4,096, 6,144 and 3,072 for the single float32 vector of a .

Table 8 reads the same rows by budget. The brief fixed five budgets per folder, 2, 4, 8, 16 and

64 KiB. It tied 2, 4 and 64 KiB to real limits, and limits near 8 and 16 KiB were looked up after the run. Amazon S3 limits the user-defined metadata of an object to 2 KB, and it also offers a store of another kind, annotations of up to 1 MB each [3]. On ext4 the block that holds a file’s attributes outside its inode is at most 4,096 bytes, and APFS keeps a value inside the file-system record up to 3,804 bytes [4, 15, 38]. Google Cloud Storage allows 8 KiB of custom metadata per object [21]. Btrfs limits an attribute with its name to the node size, 16 KiB by default, and 64 KiB is the ceiling of one attribute value on Linux [38]. At 8 KiB every folder’s *c* fits under every encoder, at one bit per dimension under E1 and E3 and at four bits under E2 and E4. In those formats it is at most 0.012 below *c* in float32 in any of the five cells and 0.007 below over all queries, both under E2. The rule of H15b was registered for one byte and for one bit. Read as a secondary at four bits, it passes under E4 and fails under E2, at -0.030 $[-0.038, -0.022]$ in M2, so at 8 KiB *c* passes it under E1, E3 and E4. At 16 KiB every folder’s *c* fits at four bits under E1 and E3 and at one byte under E2 and E4, and by the same reading no interval lies entirely below -0.02 under any encoder. At 64 KiB it fits in float16 or float32. The flat walk has no fixed size. It grows with the folder, to 491,520 bytes in float32 under E1 for a folder of 60 files, and at 8 KiB it fits every folder only under E2 and E4 at one bit. *c* is bounded by three representatives per modality label and holds at most 14 on S11. These sizes count the stored vectors alone. On most of the systems above the limit also covers the attribute’s name, and on several all the attributes of one file or object together.

What to keep in two kilobytes. Session 20 fixed the summary and varied its format. Session 21 fixed the bytes and asked which summary to put in them, on the cached vectors of both sources. At 2 KiB, the user-metadata allowance of Amazon S3 and half an ext4 block, a folder’s metadata holds eight one-bit vectors under E1, or its mean at one byte per dimension. A folder that fits whole keeps every file at one bit. Otherwise the policies differ in what the eight slots hold. *sample* keeps eight of the folder’s files, dealt to the kinds present and chosen by a hash of each file’s path. *usample* keeps the eight files of smallest hash with the kinds ignored, a bottom-*k* sketch [13] that a parent can compute exactly from its children’s. *kmeans* keeps eight spherical *k*-means centroids. Medoids, farthest-first files and one mean per kind are matched baselines at the same bytes. Three tests were fixed on S19 under E1, read on 98.33 percent intervals, on the queries whose folder does not fit whole (3,333 of the 7,372, and 310 of the 581 minority queries). On the minority queries, sample minus the mean is $+0.269$ $[+0.158, +0.384]$ (H21a) and sample minus the uniform sample $+0.116$ $[+0.054, +0.187]$ (H21c), both from 70 owners and so inconclusive by the owner minimum. Over all queries, sample minus kmeans is -0.042 $[-0.060, -0.026]$ from 207 owners, so the sample is inferior to the centroids at the registered margin (H21b). Session 24 read the three tests again on the second GitHub draw, with 146 owners in the minority cell and 365 over all queries. Sample minus the mean is $+0.252$ $[+0.181, +0.327]$, confirmed, sample minus kmeans -0.052 $[-0.067, -0.038]$, inferior, and sample minus the uniform sample $+0.097$ $[+0.054, +0.145]$, so the kinds matter. For a test whose first-draw cell missed the owner minimum, the brief of session 24 fixed the pooled reading as the one the paper takes. Pooled over both draws, H21a is confirmed at $+0.258$ $[+0.198, +0.321]$ and H21c reads that the kinds matter, at $+0.103$ $[+0.067, +0.142]$. H21b is inferior on both draws and pooled (-0.048 $[-0.060, -0.037]$). The same signs hold on Zenodo and under the other encoders (Table 9). The table prints 95 percent intervals for every row. The registered readings above are on 98.33 percent intervals, which are wider, so the same statistic carries two intervals in this section. On the minority queries of S19 under E1, eight one-bit vectors chosen with the kinds or the clusters in view reach the level of every file in float32. Their owner-weighted recall@5 is 0.624 for sample and 0.626 for kmeans, against 0.628 for every file in float32 and 0.423 for the mean at one byte per dimension. On the folders that do not fit, clustering buys 0.03 to 0.08 of recall@5 over the

Table 9: What to keep in 2 KiB of folder metadata (session 21): recall@5 differences between summaries of equal size, on the queries whose folder does not fit whole. *sample*: files at one bit dealt to the kinds present; *usample*: the files of smallest hash, kinds ignored; *kmeans*: centroids at one bit; *mean*: one vector in the most precise format that fits. 95 percent intervals, from 10,000 resamples of owners on the GitHub draws (owner-weighted) and from 1,000 resamples of directories on S11 (query-weighted). The registered tests are the S19 rows under E1 (Section 12).

Set, encoder	sample – mean, minority	sample – kmeans, all	sample – usample, minority
S19, E1	+0.269 [+0.178, +0.365]	−0.042 [−0.057, −0.029]	+0.116 [+0.064, +0.173]
S19, E4	+0.444 [+0.231, +0.671]	−0.032 [−0.066, −0.008]	+0.067 [+0.000, +0.167]
S19, E2	+0.453 [+0.292, +0.616]	−0.033 [−0.058, −0.013]	+0.049 [−0.026, +0.133]
S24, E1	+0.252 [+0.193, +0.312]	−0.052 [−0.064, −0.040]	+0.097 [+0.061, +0.136]
S11, E1	+0.308 [+0.266, +0.354]	−0.078 [−0.090, −0.065]	+0.107 [+0.086, +0.131]
S11, E3	+0.287 [+0.235, +0.340]	−0.059 [−0.071, −0.047]	+0.064 [+0.048, +0.083]
S11, E2	+0.704 [+0.662, +0.746]	−0.053 [−0.064, −0.043]	+0.078 [+0.059, +0.098]
S11, E4	+0.706 [+0.647, +0.756]	−0.044 [−0.053, −0.034]	+0.055 [+0.033, +0.080]

sample. A summary made of the files’ own vectors, which a directory can keep up to date without clustering, pays that much for it.

Written to a file system. Session 22 wrote summaries of 2,048, 4,000 and 8,192 bytes to ext4 with 4 KiB blocks, for 300 repository trees from the list of session 19 (9,934 directories, 50,906 files), and counted the blocks a reader fetches after a fresh mount. No hypothesis was read on it. On a cold descent with readahead off, an attribute of the directory costs 1.20 blocks per summary at 2,048 and at 4,000 bytes, a sidecar file in the directory 2.04, and one packed index per tree 0.82 and 1.33. Reading the one-bit vectors of a directory’s files instead costs 5.61 blocks a directory on default ext4, where an attribute of 256 bytes does not fit an inode of 256 bytes, and 2.04 with inodes of 1,024 bytes. A summary of 8,192 bytes is an ext4 attribute only with the ea_inode feature, at 2.30 blocks to read and 23.9 blocks written to replace it, against 7.0 written at 2,048 or 4,000 bytes. With the kernel’s readahead as shipped, the packed index reads more than the attribute at every size. A directory’s attribute survives `cp -a`, a tar archive made with extended attributes and a move within the file system. It is lost by a plain `cp -r`, by tar with its default options and by git, which stores no extended attribute. These are block counts on one kernel and one file system. Times were not measured.

13 Walking a repository tree

The idea this study started from is a summary on each folder that says what the folder holds, so that a reader can pick the directory to walk down whatever the folder is called. Session 23 built that walk on whole repository trees. From the pool of session 19, read in an order of its own, it took a repository whole when at least eight of its directories hold a kept file, one of them at depth two or more, with 40 to 300 kept files, at least five of them images and five not. The harvest stopped at its limit of 33,000 files with 261 trees, 11.2 percent of the 2,328 repositories read, and 33,092 files in 8,587 directories. Every directory has a summary of its subtree, and a directory with files also has one of its own files. The held-out query is removed from every summary that held it. The reader starts at the root. At each directory it scores staying by the directory’s own summary and each child by the child’s subtree summary, takes the best and stops when staying wins. A walk succeeds when it stops in the query’s directory. The flat reader reads the own summary of every

Table 10: Walking whole repository trees (session 23, 261 trees, E1). Owner-weighted success of a walk that stops in the query’s directory and of a flat read that ranks every directory’s own summary, over all 20,814 queries and the 2,844 minority queries, and the summaries a walk reads per query (Reads; the flat read reads 32.7). *fsample* keeps a fixed share of its one-bit slots per kind and merges exactly up the tree; *usample* is the bottom-*k* sample by hash; *sample* refills empty slots and does not merge exactly. The registered tests are at 2 KiB.

Budget	Summary	Walk, all	Walk, minority	Flat, all	Flat, minority	Reads
2 KiB	mean	0.630	0.686	0.671	0.782	14.1
	usample	0.542	0.646	0.599	0.745	13.0
	fsample	0.528	0.710	0.597	0.745	12.7
	sample	0.537	0.723	0.599	0.746	12.9
8 KiB	mean	0.630	0.686	0.671	0.782	14.1
	usample	0.614	0.731	0.622	0.753	14.4
	fsample	0.605	0.753	0.622	0.753	14.2
	sample	0.613	0.753	0.622	0.753	14.3
none	every file at one bit	0.625	0.753	0.625	0.753	14.8
	file names	0.477	0.649	0.478	0.655	14.7
	chance	0.069	0.092	0.047	0.049	

directory of the repository and takes the best. The summaries are those of Section 12 at 2 KiB, with one difference. A summary that a parent computes from its children’s has to merge exactly, so the sample here keeps a fixed share of its slots per kind (*fsample*), and a kind with fewer files leaves its slots empty. The uniform sample merges exactly as it is. The minority cell holds the queries of the repository’s smaller input group whose directory keeps a sibling of their group, 2,844 of the 20,814 queries, from 208 of the 259 owners.

Three tests were fixed at 2 KiB under E1, before any tree was read (Table 10). The per-kind sample walks to the minority files $+0.024$ $[-0.022, +0.073]$ more often than the mean, so H23a is inconclusive, and at the registered size a per-kind summary did not measurably help the walk. Among summaries that merge exactly, the kinds matter. *fsample* beats the uniform sample on the minority files by $+0.064$ $[+0.028, +0.104]$ (H23b). The walk reads 0.39 as many summaries as the flat reader (12.7 against 32.7 per query) and finds the target 0.069 $[0.054, 0.086]$ less often, so H23c reads inferior. At 2 KiB the summaries serve a flat read of a tree better than a descent.

Over all queries the mean is the best summary for the walk at 2 KiB, at 0.630 against 0.528 for *fsample*, and it stays the best at 8 KiB, against 0.605 for *fsample*. *fsample* stores 2.86 vectors in an own-files summary and 3.86 in a subtree summary on average, against eight slots, and the subtrees a walk compares near the root hold far more files than that. A plausible reading is that a few sampled files cover a large subtree’s majority worse than its mean does. At 8 KiB, a secondary without a reading, the per-kind sample walks to the minority files $+0.067$ $[+0.031, +0.103]$ more often than the mean, and the walk comes within 0.017 of the flat read. Every vector summary beats a reader of file names, which reaches 0.477 over all queries, by $+0.050$ $[+0.026, +0.074]$ for *fsample* at 2 KiB and $+0.127$ at 8 KiB. For the samples, walk success falls with the depth of the target, from 0.698 at the root to 0.439 at depth three or more for *fsample*, while the mean holds at 0.574 at depth three or more. A walk compounds its errors over the levels.

14 Knowing the target type

A searcher often knows the kind of file wanted, an image or a PDF, and a file system knows each file’s kind for free. Session 16 asked, pre-registered, whether a folder’s type composition helps then. The composition is a few counts and fits beside the vectors in the folder’s metadata. The kind is image, pdf (text or scanned), text, table or other, and a folder’s composition counts its files of each kind, the query file included, since the stored folder holds it. Three uses were tested on the S11 evaluation queries under all four encoders, on top of **a**, **c** and **d**. A filter ranks only the folders that hold the query’s kind. A prior adds τ times the logarithm of the kind’s share in the folder to the score, with τ chosen on the calibration queries. Type-only groups score a folder by its representatives of the query’s kind alone, and for **a** that is one pooled vector per kind (2.42 per folder). Type-only groups are compared on the 15,448 queries whose folder keeps another file of their kind, since under leave-one-out the others have nothing to match.

The filter changes little here. It adds 0.004 to 0.008 recall@5 over all queries. Image queries gain nothing, since 96.8 percent of the folders hold an image, and the rarer kinds gain more, up to 0.083 for scanned PDFs under E1 with **a**. With the filter the pooled centroid still loses the minority, by +0.153 to +0.500 in P1 and P2 across the four encoders, every interval above zero (H16a). On the queries that have another file of their kind in the folder, one pooled vector per kind recovers most of the pooled centroid’s loss, by +0.206 to +0.607 in P1 against the filtered centroid. It falls short of per-type k -means representatives in the majority cells under every encoder, by 0.035 to 0.065 in M1 and 0.035 to 0.068 in M2, so the hypothesis that one vector per type keeps them died (H16b). Within one kind a folder can hold many files, and one mean is too coarse for them. Restricting **c** to the query’s kind costs 0.002 to 0.013 recall@5 against the filter over the same queries, and up to 0.034 in P1. At the weight the calibration queries chose, 0.001 or 0.002, the prior changes recall@5 by at most 0.002 over all queries and by at most 0.010 in any of the five cells. H16c needed it to lower both minority cells, and under each encoder at least one of the two intervals does not lie below zero, so H16c died. Larger weights lower the minority cells and recall overall. Of the majority cells only M1 gains, by 0.009 at most, and M2 falls. Under E1 at $\tau = 0.05$, P1 falls from 0.674 to 0.584 while M1 rises from 0.810 to 0.816. A tree of single-type folders would gain more from the filter, and this corpus cannot show that.

15 A folder layer as an index

A folder layer promises a cheaper search. It ranks folders by a few vectors and then searches the files inside the top few. We measured the promise on S5 against the exhaustive flat walk and against a flat HNSW index [43] over all 27,243 file vectors (hnsplib 0.8.0, $M = 16$, ef_construction 200, one thread, 2,000 timed queries, best of three passes). In the two-stage form, stage one ranks directories by representation x and stage two ranks the files of the top $k = 10$ directories exactly. Hit@10 is whether a sibling of the query is in the top ten files.

With **c** as stage one, the two-stage search is within 0.001 hit@10 of the exact flat walk **d** (-0.001 [$-0.003, +0.001$]), so H5 survives its recall clause. It scores 0.55 of the vectors, though, because **c** keeps up to three representatives per label whatever the group’s size and the set has 9.7 embedded files per directory, so H5’s cost clause fails. A budget that scales with group size, $k = \min(3, \lceil n_m/4 \rceil)$, scores 0.355 of the vectors and loses 0.007 hit@10 (H6 dead on cost, by 0.005). Against flat HNSW the layer buys nothing. We put HNSW over the **c** representatives as stage one and compared it with the flat index at the pre-registered matched setting, the smallest ef_search at which the flat index is not faster than the two-stage search (128, at 462 μ s against 345 μ s).

Hit@10 differs by $+0.0006$ $[-0.0012, +0.0023]$, so H7 is dead. Timing the same indexes a second time moved the matched setting to `ef_search` 192 and the difference to $+0.0003$ $[-0.0015, +0.0020]$. The two-stage index is 1.53 times the flat one in storage. On the same one-thread build, flat HNSW at `ef_search` 64 answers in $200\ \mu\text{s}$ at `recall@10` 0.994 against exact search. At this scale (tens of thousands of in-memory vectors, one thread, frozen embeddings) every indexed method answers in under a millisecond. The exhaustive walk takes 5.2 ms and exact two-stage search with `c` 2.9 ms. At this scale the folder layer matters only through its representation. A flat index holds every file’s vector in one place, which the metadata setting does not have, so it stands beside the flat walk as a reference for that setting. Partitioned designs win at billion scale with posting lists on disk [10], and directory scopes are a filtering problem with indexes of their own [52, 63]. Our result does not reach there. It does not need to, because the loss of Section 4 is present whenever a folder vector is scored at all, whether the folder is a routing unit, a memory or a search result in its own right.

16 Discussion

The rule for a router or an agent’s memory. Consider a system that keeps one vector per folder, collection or source and routes queries on it, such as a source router for retrieval-augmented generation [14], an agent’s directory-shaped memory or a folder-level file search. The results support one rule for it. Where a folder does not mix images with other files, keep the mean. Where it does, keep a few representatives per kind. A file system tells the two apart by file type without embedding anything. On GitHub directories, 4 to 5 percent of eligible ones mix media. On the others a few representatives lose a little to the mean, and keeping them only for the mixed folders keeps most of the minority gain and is level with the mean over all directories (Section 10, post hoc). It was not shown to beat representatives for every folder.

What the mean loses where a folder mixes. On our Zenodo folders, `recall@5` on minority-modality file queries is 16 to 25 points lower with the pooled vector than with per-modality representatives under two vision-language embedders from different vendors, and 35 to 50 points lower under two CLIP-family dual towers. On GitHub directories it is about 20 points lower under the first of those embedders, on two draws. On queries that a model wrote as a simulated searcher from what a person sees of a folder, it is 14 points lower for pictures in text-heavy folders and 42 for documents in image-heavy ones under the first embedder. Under E2 the picture cell shows no loss (Section 5). With fewer collections competing the loss is smaller, 0.01 to 0.05 among 10 and 0.11 to 0.16 among 100 (post hoc), so a router that chooses among a few sources loses little of it.

How many vectors, and in how many bytes. The fix is an old one [51, 64], a few representatives per kind scored by the maximum. On the confirmation set four per label, about six vectors per folder, were not inferior to file-level search over all file queries and in every cell under all four encoders (Section 12, post hoc). Three per label, about five vectors, met the same margin over all queries under all four encoders and in every cell under E3. At three per label a folder stores a little over half as many vectors as it has files. At one byte per dimension that is 3,993 to 10,648 bytes per folder on average by encoder, and every S11 folder fits in 8 KiB at one or four bits per dimension, at a cost of at most 0.012 in any cell. Written to ext4 as an extended attribute of the directory, a summary of 2,048 or 4,000 bytes costs 1.2 blocks per directory read on a cold descent, and one of 8,192 bytes needs the `ea_inode` feature and 2.3 blocks. On whole repository trees at 2 KiB, the summaries guide a flat read better than a descent, and the mean routes a walk best over all queries. A per-kind sample beats it on the minority files only at 8 KiB (Section 13). A summary written by

a language model, which RAPTOR uses for clusters of text chunks and OpenViking for directories, does not replace them. Added to them it gains a little on whole-folder text queries. Alone it does no better than the pooled vector at finding a minority image from its description (Section 11). Per-modality mean-centering, the single-item fix [34], is worth doing on the file side, where over all file queries it helps the pooled vector, the per-modality representatives and file-level search under all four encoders. It does not replace the representatives. Under E3 it recovers 8 percent of the loss on image queries in text-heavy folders. Applied to a text query with the file-side mean, its effect depends on the encoder. Over all title queries it does nothing under E1, hurts under E2 and helps under E3 and E4 (Section 8).

The folder is two clusters, and modality is the usual reason. The dilution control suggests, on ten hosts, that a pooled vector loses any foreign minority, of the folder’s modality or not, and that the modality gap adds to the loss. Blind clustering finds the modality partition because, in this corpus, modality is where the within-folder structure is. In a corpus whose folders split along topic or language instead, we would expect label-blind clusters to help and labelled ones not to. In the control the labelled pair could not see a same-modality guest and the blind pair could (Section 7). We have not measured such a corpus. We present per-label representatives as the practical form. A file system knows a file’s type for free, and four per label met the margin against file-level search under all four encoders, where blind centroids needed eight with centering and never met it without centering under E4 (Section 12). The label does not replace the clustering. *c* still runs *k*-means inside each label, and what the type buys is the split of the budget across the groups. Session 27 measured what that split is worth with the same number of vectors in every folder. At one per label, uncentered, the labels move recall from the majority image files to the minority files under E1 to E3. They gain +0.013, +0.007 and +0.013 in P1 and +0.013, +0.027 and +0.027 in P2, and they lose 0.029, 0.028 and 0.020 in M1, while over all queries the blind split is level or ahead. Under E4 the gain at one per label is +0.004 in P1, against 0.018 lost in M1. From three per label up the two are within 0.015 of each other in every cell under all four encoders, and over all queries the labels are at most 0.007 behind. Under E4, uncentered, the labels stay ahead in P1 at every count by the point estimate, by +0.014 [+0.000, +0.028] at three per label and +0.014 [+0.005, +0.024] at five. There blind centroids at the labelled count never meet the margin against file-level search, while four per label do. At equal bytes under E1 the labels added nothing, on the main set (H9c) and at 2 KiB on S11, where per-label centroids minus blind centroids is −0.002 [−0.004, −0.000] over all queries (session 25, post hoc). From three per label up, blind *k*-means at the same count is about as good a default as the labels, except under E4 without centering.

Limitations. *Two sources.* The Zenodo folders are deposits by researchers under CC0 or CC-BY of 3 to 60 files, whose images are mostly figures. The three later Zenodo draws are disjoint but come from one search of one repository, and 527 of the 2,597 S11 records share their first creator with a record of S5. The GitHub directories come from public repositories with at least five stars and a permissive licence, and they are read under E1 alone on the second draw and under E1, E4 and E2 on the first. Personal working directories, mailboxes and agent memories were not measured. Within S11 the records are not independent either. Some creators deposited many records, and about 47 effective creator families stand behind the 2,146 text queries in image-heavy folders. Section 9 resamples creators and sets aside the queries whose file names give their folder away, and the verdicts re-read there do not move. The other Zenodo intervals, among them those of the hypotheses of sessions 3, 5 to 10, 16 and 20, resample directories only. The GitHub tests of sessions 19, 21 and 24 resample owners. The Zenodo pool was selected for records that hold both

an image type and a document type, and the GitHub scored sets over-sample mixed directories, so the study measures how much a mixed folder loses. How common mixed folders are was counted only among the eligible directories of the GitHub repositories read, 4.2 percent on the first draw and 5.0 on the second. The natural-mix estimates reweight strata that take at most two directories per stratum and repository. The mixed-only policy was tested on draws the study had already read. *Ground truth.* Relevance is structural. The one relevant folder of a query file is its own, and a record’s title or description is relevant to that record only. This measures whether a folder vector represents its files. It does not measure topical relevance across folders, and the queries of Section 8 are the depositor’s own words, which may also occur inside the record’s files. No person wrote queries for this study. The searcher queries of Section 5 and the image descriptions of Section 11 were written by models, and the commit subjects of Section 10 by the people who made a change, to describe it. The searcher queries cover 100 folders of one set, with their tests read under E1, and a model’s queries may be more specific than a person’s. *Encoders.* There are four, from three vendors. The two vision-language embedders are both built on Qwen vision-language models (Qwen2.5-VL-3B and Qwen2-VL-2B), so they share a lineage, and the second was measured on the confirmation set only, under one calibration seed. The dual towers were measured on the confirmation set only, and the second under one seed. Both towers passed the pilot check weakly and are the two weakest encoders here, so their larger loss is partly the encoders and not only the wider gap. E4’s tokenizer cut 14 of the 12,757 texts of the confirmation set at 8,192 tokens, so E4 read less of those texts than the other encoders did. The two input groups are images (with page renders of scanned PDFs) and extracted text. Audio and video were not included. *Scale.* Each Zenodo set ranks 400 to 2,808 folders of ten to twelve files on average, as one flat set. The trees of Section 13 are 261 repositories admitted with 40 to 300 kept files, walked under E1 alone with held-out files as queries. Larger folders, deeper trees and more candidates were not measured, and the conclusion of Section 15 holds for tens of thousands of in-memory vectors only. *The metadata setting.* The byte budgets of Section 12 count stored vectors and nothing else. Summaries were written to one file system, ext4 on one kernel, as random bytes of the right sizes, and the blocks read were counted. Other file systems, a real summary maintained on writes, and the time to read a folder’s attribute against listing and opening its files were not measured. The formats are plain rounding with no learned codebook and no calibration. Folders here hold at most 60 files, and larger ones, where the flat walk grows and c does not, were not measured. *The controls.* The dilution control rests on the ten hosts with guest queries of both kinds, its interval runs from 0.02 to 0.35, and its guests are topically foreign. It shows that a same-modality minority is lost too, and it does not apportion the natural loss between dilution and modality. Two vectors passed their pre-registered test under E1 by the margin rule, with point estimates outside the margin in two cells, and fell short under E2 to E4. They are not a budget the study supports. The budget was varied on the confirmation set, which earlier sessions had read, and on the second GitHub draw no budget of the sweep meets the margin in the minority cells, where the intervals resample owners (Section 12). Three k -means seeds were compared, on the confirmation set only. The margin rules of H9b, H11a, H14b, H15b, H20a and H20b kill a claim only when an interval lies entirely below -0.02 . Passing one shows that no loss beyond the margin was detected, not that two rows are equivalent. The margins themselves, 0.02 for the flat walk, 0.05 as the smallest loss of interest on GitHub and 0.005 for the mixed-only policy, were fixed in the briefs without an argument from the setting. The non-inferiority readings on the Zenodo sets are query-weighted. *What was not tried.* Names and vectors were fused with equal weights only. Trained pooling networks, corrections of the gap that retrain the encoder and late-interaction encoders were not measured. A folder’s representative vectors were combined by their maximum cosine throughout. The only other combinations tried are the single inner product of the partitioned vector F4 and the prior of Section 14. The written abstracts of Section 11 come

from one model and one set of prompts fixed in advance, and were embedded under E1 and E3 only. The single-vector result is a statement about the 40 vectors we scored on the main set under E1, and the query-side centering constant was not fitted on queries. *The evaluation code.* The audit of this paper found one bug in it, which was corrected (Section 12). Sessions 9, 10, 13, 16, 18 and 20 re-ran earlier rows and required identical ranks, which guards against drift and not against an error common to all runs. A second implementation, written from a specification without the first one’s code, gives identical ranks for a, c and d under E1 (Section 9). The other rows rest on the first implementation.

Reproducibility. The repository holds the data manifests (Zenodo record identifiers, file paths, licences, hashes), the ground truth and the embedding and evaluation scripts. It also holds every session brief with its pre-registered hypothesis and every results file with full tables and the deviations of its session. The embedding caches rebuild from the manifests, and the model revisions are pinned.

17 Conclusion

Under the first embedder, on GitHub directories that do not mix media, which are most of them, the mean of a folder’s embeddings did at least as well as a few representatives (post hoc). Where a folder mixes images and texts, its files form two clusters in a shared embedding space, and the mean sits between them and loses the folder’s minority modality. On Zenodo record folders the loss is 0.16 to 0.25 recall@5 under two vision-language embedders and 0.35 to 0.50 under two CLIP-family dual towers, and on GitHub directories it is about 0.2 under the first embedder for files with a sibling of their kind. On image queries it holds at about half its size with creators weighted equally or on queries whose file names do not give the folder away, and at about a quarter with both. On queries that a model wrote as a simulated searcher it is 0.14 for pictures in text-heavy folders and 0.42 for documents in image-heavy ones under the first embedder. A few representatives per kind recover it, and none of the single vectors we tried, centering included, does so without giving up the majority. Four per label were not inferior to exhaustive file-level search at a margin of 0.02, over all queries and in every cell, under all four encoders on the confirmation set (post hoc). Three per label fit every folder’s metadata in 8 KiB at a further cost of at most 0.012 in any cell. A folder abstract written by a vision-language model does not replace them. Keeping representatives only for the folders that mix, chosen by file type, was level with the mean over all GitHub directories at their natural mix under the first embedder (post hoc). It was not shown to beat representatives for every folder. Each registered hypothesis was tested against a criterion fixed in advance, the analyses added after review are marked post hoc, and the hypotheses that died or stayed inconclusive are reported beside the ones that survived.

Use of AI

I used AI assistants (Claude, Anthropic; Gemini, Google) throughout this work, for writing, coding, analysis and review. The sessions of Section 3 were working sessions in which I directed such an assistant, the second implementation of Section 9 was written by one, and “agent” in the text means such an assistant. The rounds of review that the text mentions were done by such assistants. Models also produced study materials. Qwen3-VL-8B-Instruct wrote the image captions and folder abstracts of Section 11, and Pixtral-12B wrote the image descriptions used there as queries. Claude Opus 5.5 wrote the searcher queries of Section 5, as a simulated searcher that was told nothing

about the study. No person wrote queries for this study. I conceived and directed the study and take full responsibility for its content.

References

- [1] Marwah Alaofi, Luke Gallagher, Mark Sanderson, Falk Scholer, and Paul Thomas. Can Generative LLMs Create Query Variants for Test Collections? An Exploratory Study. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23)*, pages 1869–1873, 2023. doi: 10.1145/3539618.3591960.
- [2] Robin Aly, Djoerd Hiemstra, and Thomas Demeester. Taily: shard selection using the tail of score distributions. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '13)*, pages 673–682, 2013. doi: 10.1145/2484028.2484033.
- [3] Amazon Web Services. Working with object metadata. Amazon Simple Storage Service User Guide, accessed 4 October 2026, 2026. URL <https://docs.aws.amazon.com/AmazonS3/latest/userguide/UsingMetadata.html>.
- [4] Apple Inc. Apple File System Reference. Developer documentation, revision of 22 June 2020, 2020. URL <https://developer.apple.com/support/downloads/Apple-File-System-Reference.pdf>.
- [5] Leif Azzopardi, Maarten de Rijke, and Krisztian Balog. Building Simulated Queries for Known-Item Topics: An Analysis Using Six European Languages. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 455–462, 2007. doi: 10.1145/1277741.1277820.
- [6] Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. Formal models for expert finding in enterprise corpora. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '06)*, pages 43–50, 2006. doi: 10.1145/1148170.1148181.
- [7] Roger L. Berger. Multiparameter Hypothesis Testing and Acceptance Sampling. *Technometrics*, 24(4):295–300, 1982. doi: 10.1080/00401706.1982.10487790.
- [8] Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. InPars: Unsupervised Dataset Generation for Information Retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*, pages 2387–2392, 2022. doi: 10.1145/3477495.3531863.
- [9] James P. Callan, Zhihong Lu, and W. Bruce Croft. Searching distributed collections with inference networks. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '95)*, pages 21–28, 1995. doi: 10.1145/215206.215328.
- [10] Qi Chen, Bing Zhao, Haidong Wang, Mingqin Li, Chuanjie Liu, Zengzhong Li, Mao Yang, and Jingdong Wang. SPANN: Highly-efficient Billion-scale Approximate Nearest Neighborhood Search. In *Advances in Neural Information Processing Systems*, volume 34, pages 5199–5212, 2021. URL https://papers.nips.cc/paper_files/paper/2021/hash/299dc35e747eb77177d9cea10a802da2-Abstract.html.

- [11] Rhea Chowers, Oshri Naparstek, Udi Barzelay, and Yair Weiss. Is the Modality Gap a Bug or a Feature? A Robustness Perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 30288–30298, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Chowers_Is_the_Modality_Gap_a_Bug_or_a_Feature_A_CVPR_2026_paper.html.
- [12] Benjamin Clavié, Antoine Chaffin, and Griffin Adams. Reducing the Footprint of Multi-Vector Retrieval with Minimal Performance Impact via Token Pooling. arXiv preprint, 2024. URL <https://arxiv.org/abs/2409.14683>.
- [13] Edith Cohen and Haim Kaplan. Summarizing Data Using Bottom-k Sketches. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Principles of Distributed Computing*, pages 225–234, 2007. doi: 10.1145/1281100.1281133.
- [14] Akash Dhasade, Rachid Guerraoui, Anne-Marie Kermarrec, Diana Petrescu, Rafael Pires, Mathis Randl, and Martijn de Vos. Efficient Federated Search for Retrieval-Augmented Generation Using Lightweight Routing. In *Distributed Applications and Interoperable Systems (DAIS 2026)*, volume 16591 of *Lecture Notes in Computer Science*, pages 3–20, Cham, 2026. Springer. doi: 10.1007/978-3-032-27358-1_1.
- [15] e2fsprogs project. ext4(5): the second, third and fourth extended file systems. e2fsprogs 1.47.4, manual page dated March 2025, 2025. URL <https://man7.org/linux/man-pages/man5/ext4.5.html>.
- [16] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From Local to Global: A Graph RAG Approach to Query-Focused Summarization. arXiv preprint, 2024. URL <https://arxiv.org/abs/2404.16130>.
- [17] Jonathan L. Elsas, Jaime Arguello, Jamie Callan, and Jaime G. Carbonell. Retrieval and feedback models for blog feed search. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '08)*, pages 347–354, 2008. doi: 10.1145/1390334.1390394.
- [18] Manuel Faysse, Hugues Sibille, Tony Wu, Bilel Omrani, Gautier Viaud, Céline Hudelot, and Pierre Colombo. ColPali: Efficient Document Retrieval with Vision Language Models. In *International Conference on Learning Representations (ICLR)*, pages 61424–61449, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/99e9e141aafc314f76b0ca3dd66898b3-Abstract-Conference.html.
- [19] David K. Gifford, Pierre Jouvelot, Mark A. Sheldon, and James W. O’Toole, Jr. Semantic file systems. In *Proceedings of the Thirteenth ACM Symposium on Operating Systems Principles (SOSP '91)*, pages 16–25, 1991. doi: 10.1145/121132.121138.
- [20] GitHub. REST API Endpoints for Search. GitHub Docs, REST API version 2022-11-28, 2026. URL <https://docs.github.com/en/rest/search/search>.
- [21] Google Cloud. Cloud Storage: Quotas and limits. Online documentation, last updated 30 September 2026, 2026. URL <https://cloud.google.com/storage/quotas>.
- [22] Eleonora Grassucci, Giordano Cicchetti, Emanuele Frasca, Aurelio Uncini, and Danilo Comminiello. Closing the Modality Gap Aligns Group-Wise Semantics.

- In *International Conference on Learning Representations (ICLR)*, pages 106315–106334, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/ad7e42e7b1f638e991d822724969be45-Abstract-Conference.html.
- [23] Michael Günther, Saba Sturua, Mohammad Kalim Akram, Isabelle Mohr, Andrei Ungureanu, Bo Wang, Sedigheh Eslami, Scott Martens, Maximilian Werk, Nan Wang, and Han Xiao. jina-embeddings-v4: Universal Embeddings for Multimodal Multilingual Retrieval. In *Proceedings of the 5th Workshop on Multilingual Representation Learning (MRL 2025)*, pages 531–550, Suzhou, China, November 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.mrl-main.36. URL <https://aclanthology.org/2025.mrl-main.36/>.
 - [24] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2 edition, 2009. doi: 10.1007/978-0-387-84858-7.
 - [25] Ziyang Jiang, Rui Meng, Xinyi Yang, Semih Yavuz, Yingbo Zhou, and Wenhui Chen. VLM2Vec: Training Vision-Language Models for Massive Multimodal Embedding Tasks. In *International Conference on Learning Representations (ICLR)*, pages 1255–1279, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/04261fce1705c4f02f062866717d592a-Abstract-Conference.html.
 - [26] Omar Khattab and Matei Zaharia. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*, pages 39–48, 2020. doi: 10.1145/3397271.3401075.
 - [27] Jinyoung Kim and W. Bruce Croft. Retrieval Experiments Using Pseudo-Desktop Collections. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, pages 1297–1306, 2009.
 - [28] Andreas Koukounas, Georgios Mastrapas, Sedigheh Eslami, Bo Wang, Mohammad Kalim Akram, Michael Günther, Isabelle Mohr, Saba Sturua, Nan Wang, and Han Xiao. jina-clip-v2: Multilingual Multimodal Embeddings for Text and Images. arXiv preprint, 2024. URL <https://arxiv.org/abs/2412.08802>.
 - [29] Anagha Kulkarni and Jamie Callan. Document allocation policies for selective searching of distributed indexes. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM '10)*, pages 449–458, 2010. doi: 10.1145/1871437.1871497.
 - [30] Anagha Kulkarni, Almer S. Tigelaar, Djoerd Hiemstra, and Jamie Callan. Shard ranking and cutoff estimation for topically partitioned collections. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM '12)*, pages 555–564, 2012. doi: 10.1145/2396761.2396833.
 - [31] Oren Kurland and Lillian Lee. Corpus structure, language models, and ad hoc information retrieval. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '04)*, pages 194–201, 2004. doi: 10.1145/1008992.1009027.
 - [32] Daniël Lakens, Anne M. Scheel, and Peder M. Isager. Equivalence Testing for Psychological Research: A Tutorial. *Advances in Methods and Practices in Psychological Science*, 1(2): 259–269, 2018. doi: 10.1177/2515245918770963.

- [33] Hyunji Lee, Luca Soldaini, Arman Cohan, Minjoon Seo, and Kyle Lo. RouterRetriever: Routing over a Mixture of Expert Embedding Models. arXiv preprint, 2025. URL <https://arxiv.org/abs/2409.02685>.
- [34] Binxu Li, Yuhui Zhang, Xiaohan Wang, Weixin Liang, Ludwig Schmidt, and Serena Yeung-Levy. Closing the Modality Gap for Mixed Modality Search. In *NeurIPS 2025 Workshop on Socially Responsible and Trustworthy Foundation Models*, 2025. URL <https://neurips.cc/virtual/2025/loc/mexico-city/133248>.
- [35] Chao Li, Zhiyuan Liu, Mengmeng Wu, Yuchi Xu, Huan Zhao, Pipei Huang, Guoliang Kang, Qiwei Chen, Wei Li, and Dik Lun Lee. Multi-Interest Network with Dynamic Routing for Recommendation at Tmall. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM '19)*, pages 2615–2623, 2019. doi: 10.1145/3357384.3357814.
- [36] Victor Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Zou. Mind the Gap: Understanding the Modality Gap in Multi-modal Contrastive Representation Learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 17612–17625, 2022. doi: 10.52202/068431-1280. URL https://papers.nips.cc/paper_files/paper/2022/hash/702f4db7543a7432431df588d57bc7c9-Abstract-Conference.html.
- [37] Sheng-Chieh Lin, Chankyu Lee, Mohammad Shoeybi, Jimmy Lin, Bryan Catanzaro, and Wei Ping. MM-Embed: Universal Multimodal Retrieval with Multimodal LLMs. In *International Conference on Learning Representations (ICLR)*, pages 44215–44234, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/6d5d6afa9957cfc9142ba60e78a467e9-Abstract-Conference.html.
- [38] Linux man-pages project. xattr(7): Extended attributes. Linux man-pages 6.19, manual page dated 8 February 2026, 2026. URL <https://man7.org/linux/man-pages/man7/xattr.7.html>.
- [39] Hongyuan Liu, Qinli Yang, Wen Li, Zhong Zhang, Jiaming Liu, Wei Han, Zhili Qin, Jinxia Guo, and Junming Shao. The Geometry of Compromise: Unlocking Generative Capabilities via Controllable Modality Alignment. arXiv preprint, 2026. URL <https://arxiv.org/abs/2604.00279>.
- [40] Xiaoyong Liu and W. Bruce Croft. Cluster-based retrieval using language models. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '04)*, pages 186–193, 2004. doi: 10.1145/1008992.1009026.
- [41] Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. Sparse, Dense, and Attentional Representations for Text Retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345, 2021. doi: 10.1162/tacl_a.00369. URL <https://aclanthology.org/2021.tacl-1.20/>.
- [42] Xueguang Ma, Sheng-Chieh Lin, Minghan Li, Wenhui Chen, and Jimmy Lin. Unifying Multimodal Retrieval via Document Screenshot Embedding. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6492–6505, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.373. URL <https://aclanthology.org/2024.emnlp-main.373/>.

- [43] Yu. A. Malkov and D. A. Yashunin. Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836, 2020. doi: 10.1109/TPAMI.2018.2889473.
- [44] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. URL <https://nlp.stanford.edu/IR-book/>.
- [45] Ambuj Mehrish, Anindya Nag, and Sebastiano Vascon. Query-Conditioned Spherical Centroid Aggregation for Multimodal Retrieval. arXiv preprint, 2026. URL <https://arxiv.org/abs/2609.15335>.
- [46] Microsoft Corporation. [MS-FSA]: File System Algorithms, section 2.1.5.15.6, FileFullEaInformation. Microsoft Open Specifications, Windows Protocols, revision 42.0 of 9 September 2025, 2025. URL https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-fsa/be0bb27a-4954-4786-80a6-947df0e82a11.
- [47] Nomic AI. Nomic Embed Multimodal: Open Source Multimodal Embedding Models for Text, Images, PDFs, and Charts. Nomic news post and Hugging Face model cards (nomic-ai/nomic-embed-multimodal-3b and nomic-ai/nomic-embed-multimodal-7b), April 2025. URL <https://www.nomic.ai/news/nomic-embed-multimodal>.
- [48] Zach Nussbaum, Brandon Duderstadt, and Andriy Mulyar. Nomic Embed Vision: Expanding the Latent Space, 2024. URL <https://arxiv.org/abs/2406.18587>.
- [49] Zach Nussbaum, John X. Morris, Brandon Duderstadt, and Andriy Mulyar. Nomic Embed: Training a Reproducible Long Context Text Embedder, 2024. URL <https://arxiv.org/abs/2402.01613>.
- [50] OpenViking. OpenViking documentation: Context Layers (L0/L1/L2) and Retrieval Mechanism. Online documentation, accessed 1 October 2026, 2026. URL <https://docs.openviking.ai/en/concepts/03-context-layers>.
- [51] Aditya Pal, Chantat Eksombatchai, Yitong Zhou, Bo Zhao, Charles Rosenberg, and Jure Leskovec. PinnerSage: Multi-Modal User Embedding Framework for Recommendations at Pinterest. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20)*, pages 2311–2320, 2020. doi: 10.1145/3394486.3403280.
- [52] Liana Patel, Peter Kraft, Carlos Guestrin, and Matei Zaharia. ACORN: Performant and Predicate-Agnostic Search Over Vector Embeddings and Structured Data. *Proceedings of the ACM on Management of Data*, 2(3):1–27, 2024. doi: 10.1145/3654923.
- [53] Christian S. Perone. VectorVFS: Your filesystem as a vector database. Software: GitHub repository and PyPI package, 2025. URL <https://github.com/perone/vectorvfs>.
- [54] François Role, Sébastien Meyer, and Victor Amblard. Fill the Gap: Quantifying and Reducing the Modality Gap in Image-Text Representation Learning. arXiv preprint, 2025. URL <https://arxiv.org/abs/2505.03703>.
- [55] Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval. In *International Conference on Learning Representations (ICLR)*, pages

- 32628–32649, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/8a2acd174940dbca361a6398a4f9df91-Abstract-Conference.html.
- [56] Jangwon Seo and W. Bruce Croft. Blog site search using resource selection. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM '08)*, pages 1053–1062, 2008. doi: 10.1145/1458082.1458222.
 - [57] Aamir Shakir, Tom Aarsen, and Sean Lee. Binary and Scalar Embedding Quantization for Significantly Faster & Cheaper Retrieval. Hugging Face Blog, 22 March 2024, 2024. URL <https://huggingface.co/blog/embedding-quantization>.
 - [58] Milad Shokouhi and Luo Si. Federated Search. *Foundations and Trends in Information Retrieval*, 5(1):1–102, 2011. doi: 10.1561/15000000010.
 - [59] Luo Si and Jamie Callan. Relevant document distribution estimation method for resource selection. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '03)*, pages 298–305, 2003. doi: 10.1145/860435.860490.
 - [60] Cees G. M. Snoek, Marcel Worring, and Arnold W. M. Smeulders. Early versus late fusion in semantic video analysis. In *Proceedings of the 13th Annual ACM International Conference on Multimedia (MM '05)*, pages 399–402, 2005. doi: 10.1145/1101149.1101236.
 - [61] Peter Todd. OpenTimestamps: Scalable, Trust-Minimized, Distributed Timestamping with Bitcoin. Blog post, 2016. URL <https://petertodd.org/2016/opentimestamps-announcement>.
 - [62] Ellen M. Voorhees. The cluster hypothesis revisited. In *Proceedings of the 8th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '85)*, pages 188–196, 1985. doi: 10.1145/253495.253524.
 - [63] Mengzhao Wang, Zheng Gong, Jingpei Hu, Jiajie Fu, Maojia Sheng, Junwen Chen, and Yifan Zhu. Directory-Aware Query and Maintenance in Vector Databases. arXiv preprint, 2026. URL <https://arxiv.org/abs/2606.16903>.
 - [64] Jason Weston, Ron J. Weiss, and Hector Yee. Nonlinear latent factorization by embedding multiple user interests. In *Proceedings of the 7th ACM Conference on Recommender Systems (RecSys '13)*, pages 65–68, 2013. doi: 10.1145/2507157.2507209.
 - [65] Ikuya Yamada, Akari Asai, and Hannaneh Hajishirzi. Efficient Passage Retrieval with Hashing for Open-domain Question Answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 979–986. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.acl-short.123. URL <https://aclanthology.org/2021.acl-short.123/>.
 - [66] Chengsong You, Qiyi Jiang, Junwei Zhou, Xiaoyu Cao, Weiyao Wang, Yiwei Xu, Ziyang Zhao, Zhen Sun, Qicheng Zhu, Xuanyi Fu, Yufan Chen, Yilun Li, Rongkang Xiong, Yunhai Hu, and Nan Du. DocPC: Document-Level Visual Retrieval via Representative Page Composition. arXiv preprint, 2026. URL <https://arxiv.org/abs/2608.25434>.

- [67] Shi Yu, Chaoyue Tang, Bokai Xu, Junbo Cui, Junhao Ran, Yukun Yan, Zhenghao Liu, Shuo Wang, Xu Han, Zhiyuan Liu, and Maosong Sun. VisRAG: Vision-based Retrieval-augmented Generation on Multi-modality Documents. In *International Conference on Learning Representations (ICLR)*, pages 21074–21098, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/3640a1997a4c9571cea9db2c82e1fc35-Abstract-Conference.html.
- [68] Xin Zhang, Ziqi Dai, Yongqi Li, Yanzhao Zhang, Dingkun Long, Pengjun Xie, Meishan Zhang, Jun Yu, Wenjie Li, and Min Zhang. Towards Text-Image Interleaved Retrieval. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4254–4269, Vienna, Austria, July 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.214. URL <https://aclanthology.org/2025.acl-long.214/>.
- [69] Xin Zhang, Yanzhao Zhang, Wen Xie, Mingxin Li, Ziqi Dai, Dingkun Long, Pengjun Xie, Meishan Zhang, Wenjie Li, and Min Zhang. Bridging Modalities: Improving Universal Multimodal Retrieval by Multimodal Large Language Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9274–9285, 2025. URL <https://openaccess.thecvf.com/content/CVPR2025/html/Zhang-Bridging-Modalities-Improving-Universal-Multimodal-Retrieval-by-Multimodal-Large-Language-CVPR-2025-paper.html>.

A Pre-registered hypotheses and verdicts

Each hypothesis was fixed in the study brief before its test was run. From session 3 on the results files copy it verbatim. Intervals are 95 percent paired bootstrap over directories, 1,000 resamples, unless stated. In sessions 17 and 18, in H26d and in H27a to H27c they resample the records’ first-creator families instead (Section 9). On the GitHub draws and trees (sessions 19, 21, 23, 24 and 26) they resample owners 10,000 times, and the estimate is the mean over owners of the owner’s mean. There a test with fewer than 100 owners is inconclusive, a loss is read as in Section 10 (confirmed, present with the size open, below +0.05 or inconclusive), and the primary tests of a session are read on 98.33 percent intervals. Session 27 reads its two loss tests together, each on its 95 percent interval (Section 5).

Claim (session 2, S1). A pooled centroid loses image queries in image-heavy directories and per-modality representations do not. **Kill:** **b** and **c** do not beat **a** on image queries in both [.5, .8) and [.8, 1] by more than the bootstrap interval. **Dead:** in [.8, 1] **c** minus **a** is $-0.001 [-0.020, +0.016]$ and **b** minus **a** $-0.018 [-0.029, -0.007]$.

H (session 3, S3). A pooled centroid loses queries whose modality is the minority in their directory; per-modality k -reps recover them. **Kill:** the interval of **c** minus **a** does not exclude zero in both minority cells. **Survived:** $+0.179 [+0.115, +0.251]$, $+0.200 [+0.135, +0.270]$.

H5 (session 5, S5). Two-stage retrieval with **c** at $k = 10$ loses at most 0.01 hit@10 against exact **d** and scores at most half the vectors. **Kill:** the interval of the hit@10 difference lies entirely below -0.01 . The cost clause was not part of the kill criterion. **Recall clause survived** $(-0.001 [-0.003, +0.001])$; **cost clause failed** (share 0.55).

- H6 (session 6, S5).** A size-scaled budget **c4**, with $k = \min(3, \lceil n_m/4 \rceil)$ representatives per label, loses at most 0.01 hit@10 and scores at most 0.35 of the vectors. **Dead on cost:** share 0.355; $-0.007 [-0.009, -0.004]$.
- H7 (session 7, S5).** HNSW over **c** as stage one beats flat HNSW on hit@10 at the matched setting (the smallest ef_search at which the flat index is not faster). **Dead:** $+0.0006 [-0.0012, +0.0023]$.
- H8 (session 8, S5).** The centered pooled centroid recovers more than half of **c**'s gain over **a** in both minority cells. **Dead:** $-0.002 [-0.023, +0.018]$ and $-0.001 [-0.020, +0.019]$; 49 percent recovered.
- H9a (session 9, S5; re-run in session 13 after the ranking fix).** No single vector in the encoder's space (families F1, F2) selected on the calibration queries is within 0.02 of **c** in both minority cells and both majority cells; dead if all four intervals of the selected configuration reach above -0.02 . **Survived** on the corrected rows: centered pooling at $\alpha = 0.5$ is $-0.048 [-0.077, -0.021]$ in P1 and $-0.047 [-0.061, -0.036]$ in M2; CORAL at $\alpha = 0.75$ is $-0.086 [-0.110, -0.059]$ in P2 and $-0.049 [-0.062, -0.038]$ in M2. As first printed, with the bug, the same four cells read $-0.061, -0.058, -0.094$ and -0.056 .
- H9b (session 9, S5).** The selected F3 or F4 configuration is within 0.02 of **c** over all queries and in the four cells; dead if any interval lies entirely below -0.02 . **Survived by the rule** for the selected F3 configuration, centered per-label centroids **bc** at 2.55 vectors per folder (M1 $-0.023 [-0.039, -0.008]$); **dead** for its two-vector reading **bc2** (M2 $-0.034 [-0.045, -0.024]$) and for the selected F4 configuration (P1 $-0.077 [-0.098, -0.060]$).
- H9c (session 9, S5).** **c** beats label-blind k -means at its budget in both minority cells. **Dead:** $+0.000 [-0.005, +0.006]$ and $+0.001 [-0.005, +0.006]$; at that budget the blind clusters recover the modality partition (purity 0.987).
- H10 (session 10, synthetic folders built on S5).** On guest queries the pooled centroid loses more against **d** when the guest is of the other input group than of the same group at the same fraction. **Survived:** $+0.180 [+0.021, +0.345]$ over 10 hosts.
- H11a (session 11, S11, E1).** Two blind centered clusters are within 0.02 recall@5 of **c** over all queries and in P1, P2, M1, M2; dead if any interval of **tb2c** minus **c** lies entirely below -0.02 . **Survived by the rule:** $-0.015 [-0.022, -0.007]$, $-0.037 [-0.058, -0.014]$, $+0.036 [+0.018, +0.053]$, $-0.005 [-0.016, +0.004]$, $-0.032 [-0.046, -0.019]$.
- H11b (session 11, S11, E2).** Under E2 the pooled centroid loses to **c** in both minority cells, and **tb2c** is within 0.02 of **c** by the H11a rule. **First clause survived** ($+0.500 [+0.462, +0.538]$, $+0.348 [+0.312, +0.386]$); **second clause dead** (P1 $-0.105 [-0.130, -0.080]$); H11b dead.
- H12a (session 12, S11, E1).** With the record title as the query, the pooled centroid loses to **c** on image-heavy directories: the interval of **c** minus **a** lies above zero. **Survived:** $+0.182 [+0.158, +0.206]$.
- H12b (session 12, S11, E1 and E2).** Under a second calibration seed the S11 verdicts stand: **c** minus **a** excludes zero in P1 and P2 under both encoders, and no interval of **tb2c** minus **c** under E1 lies entirely below -0.02 . **Survived:** $+0.255, +0.193$ (E1), $+0.513, +0.361$ (E2); **tb2c** minus **c** at worst $-0.035 [-0.058, -0.012]$.

- H14a (session 14, S11, E3).** Under a third encoder from another vendor, the interval of c minus a lies above zero in P1 and in P2. **Survived:** $+0.205$ [$+0.164, +0.244$], $+0.158$ [$+0.125, +0.188$].
- H14b (session 14, S11, E3).** No interval of c minus d over all queries, P1, P2, M1 or M2 lies entirely below -0.02 . **Survived:** the point differences run from -0.001 to -0.006 , and the lowest bound is -0.013 (M1, -0.006 [$-0.013, -0.000$]).
- H15a (session 15, S11, E4).** Under a second CLIP-family dual tower, from a third vendor, the interval of c minus a lies above zero in P1 and in P2. **Survived:** $+0.442$ [$+0.398, +0.482$], $+0.433$ [$+0.396, +0.470$].
- H15b (session 15, S11, E4).** No interval of c minus d over all queries, P1, P2, M1 or M2 lies entirely below -0.02 . **Survived:** the point differences run from -0.008 to -0.018 , every interval lies below zero, and the lowest bound is -0.024 (P2, -0.018 [$-0.024, -0.010$]; M1, -0.013 [$-0.024, -0.004$]).
- H15c (session 15, S11, E1 and E4).** On the same queries the loss is larger under E4 than under E1: the interval of (c minus a under E4) minus (c minus a under E1), from 1,000 paired resamples of the cell’s directories, lies above zero in P1 and in P2. **Survived:** $+0.189$ [$+0.145, +0.231$], $+0.265$ [$+0.226, +0.305$].
- H16a (session 16, S11, E1 to E4).** With a filter that ranks only the folders holding the query’s kind of file, the interval of c minus a lies above zero in P1 and in P2 under each encoder. **Survived** under all four: the differences run from $+0.153$ [$+0.123, +0.182$] (E3, P2) to $+0.500$ [$+0.462, +0.538$] (E2, P1).
- H16b (session 16, S11, E1 to E4).** On the queries whose folder keeps another file of their kind, no interval of one pooled vector per kind minus per-kind k -means representatives lies entirely below -0.02 over all queries, P1, P2, M1 or M2. **Dead** under all four, over all queries, in M1 and in M2: the M1 differences run from -0.035 to -0.065 and the M2 differences from -0.035 to -0.068 .
- H16c (session 16, S11, E1 to E4).** A prior on the kind’s share in the folder, with its weight chosen on the calibration queries, lowers recall@5 against the filter alone in P1 and in P2, both intervals below zero. **Dead** under all four: the chosen weight is 0.001 or 0.002, and under each encoder at least one of the two intervals includes zero or lies above it (under E4, -0.010 [$-0.015, -0.005$] in P1 and -0.000 [$-0.002, +0.001$] in P2).
- H17a (session 17, S11, E1 to E4).** Under each encoder the creator-clustered interval of c minus a lies above zero in P1 and in P2. **Survived** under all four (Table 6); the lowest bound is $+0.111$ (E3, P2).
- H17b (session 17, S11, E1 and E3).** On the clean subset (Section 9), c minus a is at least $+0.05$ and its creator-clustered interval lies above zero, in P1 and in P2. **Survived:** $+0.128$ [$+0.074, +0.180$] and $+0.172$ [$+0.126, +0.222$] under E1, $+0.095$ [$+0.038, +0.150$] and $+0.163$ [$+0.115, +0.218$] under E3; the same conditions hold under E2 and E4.
- H17c (session 17, S11, E1).** A second implementation written from a specification gives, for a , c and d , the same recall@5 at three decimals over all queries and in P1, P2, M1 and M2, and identical ranks for at least 99 percent of the queries of each row. **Survived:** the same rank for every one of the 17,140 queries in rows a , c and d .

- H18a (session 18, S11, E1 and E3).** On the titles of image-heavy folders, s_1 minus c is read per encoder as above c (at least $+0.05$, interval above zero), below c (at most -0.05 , interval below zero) or between; a split if the encoders disagree. **Split:** between under E1 (-0.007 [$-0.049, +0.038$]), below under E3 (-0.263 [$-0.433, -0.096$]).
- H18b (session 18, S11, E1 and E3).** On descriptions of P1 images, gated on d reaching $\text{recall@5 } 0.25$, c minus s_1 and c minus a are each at least $+0.05$ with intervals above zero. **Survived:** $+0.291$ [$+0.259, +0.322$] and $+0.290$ [$+0.260, +0.324$] under E1, $+0.344$ [$+0.310, +0.380$] and $+0.290$ [$+0.259, +0.322$] under E3.
- H20a (session 20, S11, E1 to E4).** With every representative stored at one byte per dimension, no interval of c minus d in float32 over all queries, P1, P2, M1 or M2 lies entirely below -0.02 . **Survived** under all four; the lowest bound is -0.025 (E2, M2, -0.018 [$-0.025, -0.012$]).
- H20b (session 20, S11, E1 to E4).** The same at one bit per dimension. **Survived** under E1 and E3, where the lowest bounds are -0.019 and -0.017 , both in M1; **dead** under E2 (M2, -0.033 [$-0.041, -0.025$]) and under E4 (all queries -0.057 [$-0.062, -0.051$], P1 -0.084 [$-0.101, -0.066$], M1 -0.101 [$-0.114, -0.088$]).
- H20c (session 20, S11, E1 to E4).** The interval of c at one bit per dimension minus a in float32 lies above zero in P1 and in P2. **Survived** under all four: $+0.255$ [$+0.216, +0.298$] and $+0.173$ [$+0.141, +0.205$] under E1, $+0.503$ [$+0.465, +0.541$] and $+0.341$ [$+0.306, +0.379$] under E2, $+0.207$ [$+0.167, +0.246$] and $+0.172$ [$+0.141, +0.203$] under E3, $+0.371$ [$+0.327, +0.409$] and $+0.426$ [$+0.390, +0.464$] under E4.
- H19a (session 19, S19, E1).** On GitHub directories, c minus a in P1s and in P2s, read as a loss: confirmed if both cells are confirmed, replicated with the size open if both are confirmed or present, not replicated at the registered size if either is below $+0.05$, inconclusive otherwise. **Inconclusive:** $+0.193$ [$+0.099, +0.295$] from 86 owners and $+0.166$ [$+0.080, +0.260$] from 98, both short of the owner minimum.
- H19b (session 19, S19, E1).** The H19a statistic on the clean subset, read as a loss on 95 percent intervals. **Inconclusive:** $+0.188$ [$+0.090, +0.293$] from 60 owners and $+0.142$ [$+0.063, +0.225$] from 82.
- H19c (session 19, S19, E1).** Where c does not hold its folder whole, c minus d is not inferior if the lower end of its interval is at least -0.02 , inferior if the upper end is under -0.02 . **Not inferior:** -0.002 [$-0.014, +0.008$], 512 owners.
- H19d (session 19, S19, E1).** With commit subjects as the query on image-heavy directories, c minus a , read as a loss on its 95 percent interval, gated on d reaching $\text{recall@5 } 0.10$ and more than the file names alone. **Inconclusive:** the gate passed (0.254 against 0.066); $+0.010$ [$-0.061, +0.078$] from 66 owners.
- H21a (session 21, S19, E1).** At 2 KiB, on the minority queries whose folder does not fit whole, the sample minus the mean, read as a loss. **Inconclusive** on S19: $+0.269$ [$+0.158, +0.384$] from 70 owners. By the rule of session 24 the pooled reading stands: **confirmed**, $+0.258$ [$+0.198, +0.321$] from 216 owners.
- H21b (session 21, S19, E1).** At 2 KiB, over all queries whose folder does not fit whole, the sample minus kmeans is not inferior if the lower end of its interval is at least -0.02 , inferior if the upper end is under -0.02 . **Inferior:** -0.042 [$-0.060, -0.026$] from 207 owners.

- H21c (session 21, S19, E1).** At 2 KiB, on the minority queries whose folder does not fit whole, the sample minus the uniform sample: the kinds matter if the lower end of the interval is above zero, the uniform sample is better if the upper end is below zero, no difference at the margin if the interval lies within -0.02 and $+0.02$. **Inconclusive** on S19: $+0.116$ $[+0.054, +0.187]$ from 70 owners. By the rule of session 24 the pooled reading stands: **the kinds matter**, $+0.103$ $[+0.067, +0.142]$ from 216 owners.
- H23a (session 23, S23, E1).** On whole repository trees at 2 KiB, walk success with the per-kind sample fsample minus walk success with the mean on the minority queries, read as a loss. **Inconclusive:** $+0.024$ $[-0.022, +0.073]$ from 208 owners.
- H23b (session 23, S23, E1).** At 2 KiB, walk success with fsample minus with the uniform sample on the minority queries: the kinds matter if the lower end of the interval is above zero. **The kinds matter:** $+0.064$ $[+0.028, +0.104]$ from 208 owners.
- H23c (session 23, S23, E1).** At 2 KiB, walk success minus flat success, both with fsample, over all queries: not inferior if the lower end of the interval is at least -0.02 , inferior if the upper end is under -0.02 . **Inferior:** -0.069 $[-0.086, -0.054]$ from 259 owners, the walk reading 0.39 of the summaries the flat read reads.
- H24a (session 24, S24, E1).** H19a’s statistic and readings on a second draw of GitHub directories, sized for the owner minimum after session 19’s E1 table was read. **Confirmed** in both cells: $+0.210$ $[+0.144, +0.279]$ from 172 owners and $+0.196$ $[+0.133, +0.260]$ from 197.
- H24b (session 24, S24, E1).** H19b’s on the second draw. **Confirmed** in both cells: $+0.180$ $[+0.113, +0.250]$ from 118 owners and $+0.209$ $[+0.147, +0.275]$ from 155.
- H24c (session 24, S24, E1).** H19c’s statistic and readings on the second draw. **Not inferior**, at -0.002 $[-0.010, +0.006]$ from 918 owners.
- H24d (session 24, S24, E1).** H19d’s on the second draw. **Confirmed:** the gate passed, at 0.258 against 0.122 for the file names, and the difference is $+0.117$ $[+0.065, +0.169]$ from 138 owners.
- H24e (session 24, S24, E1).** H21a’s on the second draw. **Confirmed:** $+0.252$ $[+0.181, +0.327]$ from 146 owners.
- H24f (session 24, S24, E1).** H21b’s on the second draw. **Inferior:** -0.052 $[-0.067, -0.038]$ from 365 owners.
- H24g (session 24, S24, E1).** H21c’s on the second draw. **The kinds matter:** $+0.097$ $[+0.054, +0.145]$ from 146 owners.
- H26a (session 26, S19 and S24, E1, post hoc).** The mixed-only policy (c for a folder that holds an image and another file, a for the rest) minus a in P1s and P2s at the natural mix, with natural-mix candidates, confirmed if both lower ends of the 98.33 percent intervals are at least $+0.05$. **Confirmed:** $+0.143$ $[+0.094, +0.197]$ from 258 owners and $+0.180$ $[+0.134, +0.228]$ from 294.
- H26b (session 26, S19 and S24, E1, post hoc).** The same policy minus a over all queries at the natural mix, not inferior if the lower end of its 98.33 percent interval is at least -0.005 . **Not inferior:** $+0.001$ $[+0.0004, +0.002]$.

H26c (session 26, S19 and S24, E1, post hoc). The same policy minus **c** over all queries at the natural mix, above zero if the lower end of its 98.33 percent interval is above zero. **Dead:** +0.007 [−0.0001, +0.015].

H26d (session 26, S11, E1 and E3, post hoc). With the vector ranking fused with the filename ranking (reciprocal rank fusion, $k = 60$), **c** minus **a** is at least +0.05 and its creator-clustered interval lies above zero, in P1 and in P2. **Survived:** +0.104 [+0.067, +0.144] and +0.067 [+0.041, +0.097] under E1, +0.075 [+0.044, +0.109] and +0.057 [+0.035, +0.086] under E3.

H27a (session 27, S11, E1). On queries written to find a picture in a folder with image fraction below 0.5 (QI-T), **c** minus **a** is at least +0.05 and its creator-clustered interval lies above zero, gated on **d** reaching recall@5 0.25. The queries were written by a model as a simulated searcher, by an amendment made before any was scored. **Survived:** +0.140 [+0.020, +0.280] over 50 queries; the gate passed at 0.720.

H27b (session 27, S11, E1). The same on queries written to find a document in a folder with image fraction of 0.5 or more (QT-I). **Survived:** +0.420 [+0.280, +0.560] over 50 queries; the gate passed at 0.940.

H27c (session 27, S11, E1). Over all 200 of these queries, **c** minus **d** is not inferior if the lower end of its interval is at or above −0.02, inferior if the upper end is below −0.02. **Not inferior:** +0.000 [−0.020, +0.020].

Reading H27d (session 27, S11, E1 to E4). Per family of budgets, the smallest whose difference from **d** has a lower bound at or above −0.02 over all queries and in P1, P2, M1 and M2, under each encoder and under all four. It is a reading without a kill criterion. **Four per label** under all four encoders, uncentered or centered. Three per label qualify uncentered under E3 only and centered under E1 to E3. Blind centroids need eight, centered, to qualify under all four, and uncentered ones qualify at no count under E4.

Reading H27e (session 27, S11, E1 to E4). The largest spread of recall@5 across three k -means seeds over the five cells and four encoders, expected below 0.005. **0.0026** for **c** (E2, P1) and **0.0084** for **tb2c** (E4, M2).