

---

# AdANNS: A Framework for Adaptive Semantic Search

---


Anonymous Author(s)

Affiliation

Address

email

## Abstract

1 Web-scale search systems learn an encoder to embed a given query which is then  
2 hooked into an approximate nearest neighbor search (ANNS) pipeline to retrieve  
3 similar data points. To accurately capture tail queries and data points, learned  
4 representations typically are *rigid, high-dimensional* vectors that are generally  
5 used as-is in the entire ANNS pipeline and can lead to computationally expensive  
6 retrieval. In this paper, we argue that instead of rigid representations, different  
7 stages of ANNS can leverage *adaptive representations* of varying capacities to  
8 achieve significantly better accuracy-compute trade-offs, i.e., stages of ANNS that  
9 can get away with more approximate computation should use a lower-capacity rep-  
10 resentation of the same data point. To this end, we introduce AdANNS , a novel  
11 ANNS design framework that explicitly leverages the flexibility of Matryoshka  
12 Representations [30]. We demonstrate state-of-the-art accuracy-compute trade-offs  
13 using novel AdANNS-based key ANNS building blocks like search data structures  
14 (AdANNS-IVF) and quantization (AdANNS-OPQ). For example on ImageNet  
15 retrieval, AdANNS-IVF is up to 1.5% more accurate than the rigid representations-  
16 based IVF [46] at the same compute budget; and matches accuracy while being up  
17 to 90× faster in *wall-clock time*. For Natural Questions, 32-byte AdANNS-OPQ  
18 matches the accuracy of the 64-byte OPQ baseline [12] constructed using rigid  
19 representations – *same accuracy at half the cost!* We further show that the gains  
20 from AdANNS translate to modern-day composite ANNS indices that combine  
21 search structures and quantization. Finally, we demonstrate that AdANNS can  
22 enable inference-time adaptivity for compute-aware search on ANNS indices built  
23 non-adaptively on matryoshka representations. Code will be open-sourced.

## 24 1 Introduction

25 Semantic search [23] on learned representations [38, 39, 48] is a major component in retrieval  
26 pipelines [4, 9]. In its simplest form, semantic search methods learn a neural network to embed  
27 queries as well as a large number ( $N$ ) of data points in a  $d$ -dimensional vector space. For a given query,  
28 the nearest (in embedding space) point is retrieved using either an exact search or using approximate  
29 nearest neighbor search (ANNS) [20] which is now indispensable for real-time large-scale retrieval.

30 Existing semantic search methods learn fixed or *rigid* representations (RRs) which are used as is in  
31 all the stages of ANNS. That is, while ANNS indices allow a variety of parameters for searching  
32 the design space to optimize the accuracy-compute trade-off, the provided data dimensionality is  
33 typically assumed to be an *immutable* parameter. To make it concrete, let us consider inverted file  
34 index (IVF) [46], a popular web-scale ANNS technique [15]. IVF has two stages (Section 3) during  
35 inference: (a) *cluster mapping*: mapping the query to a cluster of data points [35], and (b) *linear*  
36 *scan*: distance computation w.r.t all points in the retrieved cluster to find the nearest neighbor (NN).  
37 Standard IVF utilizes the same high-dimensional RR for both phases, which can be sub-optimal.

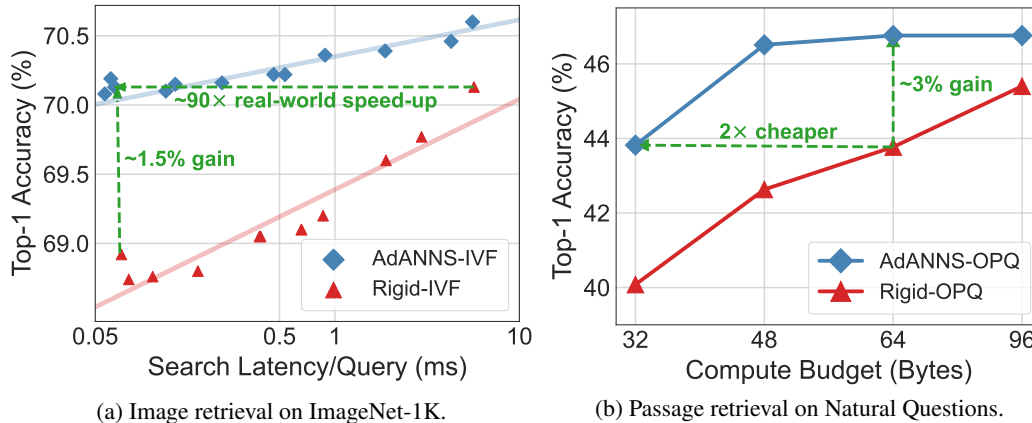



Figure 1: AdANNS helps design search data structures and quantization methods with *better accuracy-compute trade-offs* than the existing solutions. In particular, (a) AdANNS-IVF improves on standard IVF by up to 1.5% in accuracy while being 90 $\times$  faster in deployment and (b) AdANNS-OPQ is as accurate as the baseline OPQ on RRs while being at least 2 $\times$  faster in deployment. Rigid-IVF and Rigid-OPQ are standard techniques that are built on rigid representations (RRs) while AdANNS uses matryoshka representations (MRs) [30].

38 **Why the sub-optimality?** Imagine one needs to partition a dataset into  $k$  clusters for IVF and the  
 39 dimensionality of the data is  $d$  – IVF uses full  $d$  representation to partition into  $k$  clusters. However,  
 40 suppose we have an alternate approach that somehow projects the data in  $d/2$  dimensions and learns  
 41  $2k$  clusters. Note that the storage and computation to find the nearest cluster remains the same in both  
 42 cases, i.e., when we have  $k$  clusters of  $d$  dimensions or  $2k$  clusters of  $d/2$  dimensions.  $2k$  clusters  
 43 can provide significantly more refined partitioning, but the distances computed between queries and  
 44 clusters could be significantly more inaccurate after projection to  $d/2$  dimensions.

45 So, if we can find a mechanism to obtain a  $d/2$ -dimensional representation of points that can accurately  
 46 approximate the topology/distances of  $d$ -dimensional representation, then we can potentially build  
 47 significantly better ANNS structure that utilizes different capacity representations for the cluster  
 48 mapping and linear scan phases of IVF. But how do we find such *adaptive representations*? These  
 49 desired adaptive representations should be cheap to obtain and still ensure distance preservation  
 50 across dimensionality. Post-hoc dimensionality reduction techniques like SVD [13] and random  
 51 projections [24] on high-dimensional RRs are potential candidates, but our experiments indicate that  
 52 in practice they are highly inaccurate and do not preserve distances well enough (Figure 2).

53 Instead, we identify that the recently proposed Matryoshka Representations (MRs) [30] satisfy  
 54 the specifications for adaptive representations. Matryoshka representations pack information in a  
 55 hierarchical nested manner, i.e., the first  $m$ -dimensions of the  $d$ -dimensional MR form an accurate  
 56 low-dimensional representation while being aware of the information in the higher dimensions.  
 57 This allows us to deploy MRs in two major and novel ways as part of ANNS: (a) low-dimensional  
 58 representations for accuracy-compute optimal clustering and quantization, and (b) high-dimensional  
 59 representations for precise re-ranking when feasible.

60 To this effort, we introduce AdANNS , a novel design framework for semantic search that uses  
 61 matryoshka representation-based *adaptive representations* across different stages of ANNS to ensure  
 62 significantly better accuracy-compute trade-off than the state-of-the-art baselines.

63 Typical ANNS systems have two key components: (a) search data structure to store datapoints, (b)  
 64 distance computation to map a given query to points in the data structure. Through AdANNS, we  
 65 address both these components and significantly improve their performance. In particular, we first  
 66 propose AdANNS-IVF (Section 4.1) which tackles the first component of ANNS systems. AdANNS-  
 67 IVF uses standard full-precision computations but uses adaptive representations for different IVF  
 68 stages. On ImageNet 1-NN image retrieval (Figure 1a), AdANNS-IVF is up to 1.5% more accurate  
 69 for the compute budget and 90 $\times$  cheaper in deployment for the same accuracy as IVF.


70 We then propose AdANNS-OPQ (Section 4.2) which addresses the second component by using  
 71 AdANNS-based quantization (OPQ [12]) – here we use exhaustive search overall points. AdANNS-  
 72 OPQ is as accurate as the baseline OPQ on RRs while being at least 2 $\times$  faster on Natural Ques-  
 73 tions [31] 1-NN passage retrieval (Figure 1b). Finally, we combine the two techniques to obtain

74 AdANNS-IVFOPQ (Section 4.3) which is more accurate while being much cheaper – up to  $8\times$   
75 – than the traditional IVFOPQ [23] index. To demonstrate generality of our technique, we adapt  
76 AdANNS to DiskANN [21] which provides interesting accuracy-compute tradeoff; see Table 1.

77 While MR already has multi-granular representations, careful integration with ANNS building blocks  
78 is critical to obtain a practical method and is *our main contribution*. In fact, Kusupati et al. [30]  
79 proposed a simple adaptive retrieval setup that uses smaller-dimensional MR for shortlisting in re-  
80 trieval followed by precise re-ranking with a higher-dimensional MR. Such techniques, unfortunately,  
81 cannot be scaled to industrial systems as they require forming a new index for every shortlisting  
82 provided by low-dimensional MR. Ensuring that the method aligns well with the modern-day  
83 ANNS pipelines is important as they already have mechanisms to handle real-world constraints like  
84 load-balancing [15] and random access from disk [21]. So, AdANNS is a step towards making the  
85 abstraction of adaptive search and retrieval feasible at the web-scale.

86 Through extensive experimentation, we also show that AdANNS generalizes across search data  
87 structures, distance approximations, modalities (text & image), and encoders (CNNs & Transformers)  
88 while still translating the theoretical gains to latency reductions in deployment. While we have mainly  
89 focused on IVF and OPQ-based ANNS in this work, AdANNS also blends well with other ANNS  
90 pipelines. We also show that AdANNS can enable compute-aware elastic search on prebuilt indices  
91 without making any modifications (Section 5.1); note that this is in contrast to AdANNS-IVF that  
92 builds the index explicitly utilizing “adaptivity” in representations. Finally, we provide an extensive  
93 analysis on the alignment of matryoshka representation for better semantic search (Section 5.2).

94 **We make the following key contributions:**

- 95 • We introduce AdANNS , a novel framework for semantic search that leverages matryoshka  
96 representations for designing ANNS systems with better accuracy-compute trade-offs.
- 97 • AdANNS powered search data structure (AdANNS-IVF) and quantization (AdANNS-OPQ)  
98 show a significant improvement in accuracy-compute tradeoff compared to existing solutions.
- 99 • AdANNS generalizes to modern-day composite ANNS indices and can also enable compute-aware  
100 elastic search during inference with no modifications.

## 101 2 Related Work

102 Approximate nearest neighbour search (ANNS) is a paradigm to come as close as possible [7] to  
103 retrieving the “true” nearest neighbor (NN) without the exorbitant search costs associated with  
104 exhaustive search [20, 50]. The “approximate” nature comes from data pruning as well as the cheaper  
105 distance computation that enable real-time web-scale search. In its naive form, NN-search has a  
106 complexity of  $\mathcal{O}(dN)$ ;  $d$  is the data dimensionality used for distance computation and  $N$  is the size  
107 of the database. ANNS employs each of these approximations to reduce the linear dependence on the  
108 dimensionality (cheaper distance computation) and data points visited during search (data pruning).

109 **Cheaper distance computation.** From a bird’s eye view, cheaper distance computation is always  
110 obtained through dimensionality reduction (quantization included). PCA and SVD [13, 25] can  
111 reduce dimensionality and preserve distances only to a limited extent without sacrificing accuracy.  
112 On the other hand, quantization-based techniques [6, 14] like (optimized) product quantization  
113 ((O)PQ) [12, 22] have proved extremely crucial for relatively accurate yet cheap distance computation  
114 and simultaneously reduce the memory overhead significantly. Another naive solution is to indepen-  
115 dently train the representation function with varying low-dimensional information bottlenecks [30]  
116 which is rarely used due to the costs of maintaining multiple models and databases.

117 **Data pruning.** Enabled by various data structures, data pruning reduces the number of data points  
118 visited as part of the search. This is often achieved through hashing [8, 44], trees [3, 11, 15, 46]  
119 and graphs [21, 37]. More recently there have been efforts towards end-to-end learning of the  
120 search data structures [16, 28, 29]. However, web-scale ANNS indices are often constructed on rigid  
121  $d$ -dimensional real vectors using the aforementioned data structures that assist with the real-time  
122 search. For a more comprehensive review of ANNS structures please refer to [5, 33, 49].

123 **Composite indices.** ANNS pipelines often benefit from the complementary nature of various building  
124 blocks [23, 40]. In practice, often the data structures (coarse-quantizer) like IVF [46] and HNSW [36]  
125 are combined with cheaper distance alternatives like PQ [22] (fine-quantizer) for massive speed-ups  
126 in web-scale search. While the data structures are built on  $d$ -dimensional real vectors, past works

127 consistently show that PQ can be safely used for distance computation during search time. As  
128 evident in modern web-scale ANNS systems like DiskANN [21], the data structures are built on  
129  $d$ -dimensional real vectors but work with PQ vectors (32 – 64-byte) for fast distance computations.

130 **ANNS benchmark datasets.** Despite the Herculean advances in representation learning [18, 40],  
131 ANNS progress is often only benchmarked on fixed representation vectors provided for about a  
132 dozen million to billion scale datasets [1, 45] with limited access to the raw data. This resulted in  
133 the improvement of algorithmic design for rigid representations (RRs) that are often not specifically  
134 designed for search. All the existing ANNS methods work with the assumption of using the provided  
135  $d$ -dimensional representation which might not be Pareto-optimal for the accuracy-compute trade-  
136 off in the first place. Note that the lack of raw-image and text-based benchmarks led us to using  
137 ImageNet-1K [43] (1.3M images, 50K queries) and Natural Questions [31] (21M passages, 3.6K  
138 queries) for experimentation. While not billion-scale, the results observed on ImageNet often translate  
139 to real-world progress [27], and Natural Questions is one of the largest question answering datasets  
140 benchmarked for dense passage retrieval [26], making our results generalizable and widely applicable.

141 In this paper, we investigate the utility of adaptive representations – embeddings of different dimen-  
142 sionalities having similar semantic information – in improving the design of ANNS algorithms. This  
143 helps in transitioning out of restricted construction and inference on rigid representations for ANNS.  
144 To this end, we extensively use Matryoshka Representations (MRs) [30] which have desired adaptive  
145 properties in-built. To the best of our knowledge, this is the first work that improves accuracy-compute  
146 trade-off in ANNS by leveraging adaptive representations on different phases of construction and  
147 inference for ANNS data structures.

### 148 3 Problem Setup, Notation, and Preliminaries

149 The problem setup of approximate nearest neighbor search (ANNS) [20] consists of a database of  $N$   
150 data points,  $[x_1, x_2, \dots, x_N]$ , and a query,  $q$ , where the goal is to “approximately” retrieve the nearest  
151 data point to the query. Both the database and query are embedded to  $\mathbb{R}^d$  using a representation  
152 function  $\phi : \mathcal{X} \rightarrow \mathbb{R}^d$ , often a neural network that can be learned through various representation  
153 learning paradigms [2, 18, 19, 38, 40].

154 **Matryoshka Representations (MRs).** The  $d$ -dimensional representations from  $\phi$  can have a nested  
155 structure like Matryoshka Representations (MRs) [30] in-built –  $\phi^{\text{MR}(d)}$ . Matryoshka Representation  
156 Learning (MRL) learns these nested representations with a simple strategy of optimizing the same  
157 training objective at varying dimensionalities. These granularities are ordered such that the lowest  
158 representation size forms a prefix for the higher-dimensional representations. So, high-dimensional  
159 MR inherently contains low-dimensional representations of varying granularities that can be accessed  
160 for free – first  $m$ -dimensions ( $m \in [d]$ ) ie.,  $\phi^{\text{MR}(d)}[1 : m]$  from the  $d$ -dimensional MR form an  
161  $m$ -dimensional representation which is as accurate as its independently trained rigid representation  
162 (RR) counterpart –  $\phi^{\text{RR}(m)}$ . Training an encoder with MRL does not involve any overhead or  
163 hyperparameter tuning and works seamlessly across modalities, training objectives, and architectures.

164 **Inverted File Index (IVF).** IVF [46] is an ANNS data structure used in web-scale search sys-  
165 tems [15] owing to its simplicity, minimal compute overhead, and high accuracy. IVF construction  
166 involves clustering (coarse quantization through k-means) [35] on  $d$ -dimensional representation that  
167 results in an inverted file list [51] of all the data points in each cluster. During search,  $d$ -dimensional  
168 query representation is assigned to the most relevant cluster ( $C_i; i \in [k]$ ) by finding the closest cen-  
169 troid ( $\mu_i$ ) using an appropriate distance metric ( $L_2$  or cosine). This is followed by an exhaustive linear  
170 search across all data points in the cluster which gives the closest NN (see Figure 5 in Appendix A  
171 for IVF overview). Lastly, IVF can scale to web-scale by utilizing a hierarchical IVF structure within  
172 each cluster [15]. Table 2 in Appendix A describes the retrieval formula for multiple variants of IVF.

173 **Optimized Product Quantization (OPQ).** Product Quantization (PQ) [22] works by splitting a  
174  $d$ -dimensional real vector into  $m$  sub-vectors and quantizing each sub-vector with an independent  $2^b$   
175 length codebook across the database. After PQ, each  $d$ -dimensional vector can be represented by a  
176 compact  $m \times b$  bit vector; we make each vector  $m$  bytes long by fixing  $b = 8$ . During search time,  
177 distance computation between the query vector and PQ database is extremely efficient with only  $m$   
178 codebook lookups. The generality of PQ encompasses scalar/vector quantization [14, 35] as special

179 cases. However, PQ can be further improved by rotating the  $d$ -dimensional space appropriately to  
 180 maximize distance preservation after PQ. Optimized Product Quantization (OPQ) [12] achieves this  
 181 by learning an orthonormal projection matrix  $R$  that rotates the  $d$ -dimensional space to be more  
 182 amenable to PQ. OPQ shows consistent gains over PQ across a variety of ANNS tasks and has  
 183 become the default choice in standard composite indices [21, 23].

184 **Datasets.** We evaluate the ANNS algorithms while changing the representations used for the search  
 185 thus making it impossible to evaluate on the usual benchmarks [1]. Hence we experiment with two  
 186 public datasets: (a) ImageNet-1K [43] dataset on the task of image retrieval – where the goal is to  
 187 retrieve images from a database (1.3M image train set) belonging to the same class as the query  
 188 image (50K image validation set) and (b) Natural Questions (NQ) [31] dataset on the task of question  
 189 answering through dense passage retrieval – where the goal is to retrieve the relevant passage from a  
 190 database (21M Wikipedia passages) for a query (3.6K questions).


191 **Metrics** Performance of ANNS is often measured using recall score [21],  $k$ -recall@ $N$  – recall of  
 192 the exact NN across search complexities which denotes the recall of  $k$  “true” NN when  $N$  data points  
 193 are retrieved. However, the presence of labels allows us to compute 1-NN (top-1) accuracy. Top-1  
 194 accuracy is a harder and more fine-grained metric that correlates well with typical retrieval metrics  
 195 like recall and mean average precision (mAP@ $k$ ). Even though we report top-1 accuracy by default  
 196 during experimentation, we discuss other metrics in Appendix C. Finally, we measure the compute  
 197 overhead of ANNS using MFLOPS/query and also provide wall-clock times (see Appendix E.4).

198 **Encoders.** For ImageNet, we encode both the database and query set using a ResNet50 ( $\phi_I$ ) [18]  
 199 trained on ImageNet-1K. For NQ, we encode both the passages in the database and the questions in  
 200 the query set using a BERT-Base ( $\phi_N$ ) [10] model fine-tuned on NQ for dense passage retrieval [26].

201 We use the trained ResNet50 models with varying representation sizes ( $d = [8, 16, \dots, 2048]$ ; default  
 202 being 2048) as suggested by Kusupati et al. [30] alongside the MRL-ResNet50 models trained with  
 203 MRL for the same dimensionalities. The RR and MR models are trained to ensure the supervised  
 204 one-vs-all classification accuracy across all data dimensionalities is nearly the same – 1-NN accuracy  
 205 of 2048- $d$  RR and MR models are 71.19% and 70.97% respectively on ImageNet-1K. Independently  
 206 trained models,  $\phi_I^{\text{RR}(d)}$ , output  $d = [8, 16, \dots, 2048]$  dimensional RRs while a single MRL-ResNet50  
 207 model,  $\phi_I^{\text{MR}(d)}$ , outputs a  $d = 2048$ -dimensional MR that contains all the 9 granularities.

208 We also train BERT-Base models in a similar vein as the aforementioned ResNet50 models. The  
 209 key difference is that we take a pre-trained BERT-Base model and fine-tune on NQ as suggested  
 210 by Karpukhin et al. [26] with varying (5) representation sizes (bottlenecks) ( $d = [48, 96, \dots, 768]$ ;  
 211 default being 768) to obtain  $\phi_N^{\text{RR}(d)}$  that creates RRs for the NQ dataset. To get the MRL-BERT-  
 212 Base model, we fine-tune a pre-trained BERT-Base encoder on the NQ train dataset using the MRL  
 213 objective with the same granularities as RRs to obtain  $\phi_N^{\text{MR}(d)}$  which contains all five granularities.  
 214 Akin to ResNet50 models, the RR and MR BERT-Base models on NQ are built to have similar 1-NN  
 215 accuracy for 768- $d$  of 52.2% and 51.5% respectively. More implementation details can be found in  
 216 Appendix B and additional experiment-specific information is provided at the appropriate places.

## 217 4 AdANNS – Adaptive ANNS

218 In this section, we present our proposed AdANNS  framework that exploits the inherent flexibility  
 219 of matryoshka representations to improve the accuracy-compute trade-off for semantic search com-  
 220 ponents. Standard ANNS pipeline can be split into two key components: (a) search data structure  
 221 that indexes and stores data points, (b) query-point computation method that outputs (approximate)  
 222 distance between a given query and data point. For example, standard IVFOPQ [23] method uses  
 223 an IVF structure to index points on full-precision vectors and then relies on OPQ for more efficient  
 224 distance computation between the query and the data points during the linear scan.

225 Below, we show that AdANNS can be applied to both the above-mentioned ANNS components  
 226 and provides significant gains on the computation-accuracy tradeoff curve. In particular, we present  
 227 AdANNS-IVF which is AdANNS version of the standard IVF index structure [46], and the closely  
 228 related ScaNN structure [15]. We also present AdANNS-OPQ which introduces representation adap-  
 229 tivity in the OPQ, an industry-default quantization. Then, in Section 4.3 we further demonstrate the

230 combination of the two techniques to get AdANNS-IVFOPQ – an AdANNS version of IVFOPQ [23]  
 231 – and AdANNS-DiskANN, a similar variant of DiskANN [21]. Overall, our experiments show that  
 232 AdANNS-IVF is significantly more accuracy-compute optimal compared to the IVF indices built on  
 233 RRs and AdANNS-OPQ is as accurate as the OPQ on RRs while being significantly cheaper.

#### 234 4.1 AdANNS-IVF

235 Recall from Section 1 that IVF has  
 236 a clustering and a linear scan phase,  
 237 where both phase use same dimen-  
 238 sional rigid representation. Now,  
 239 AdANNS-IVF allows the clustering  
 240 phase to use the first  $d_c$  dimensions  
 241 of the given matryoshka represen-  
 242 tation (MR). Similarly, the linear  
 243 scan within each cluster uses  $d_s$   
 244 dimensions, where again  $d_s$  repre-  
 245 sents top  $d_s$  coordinates from MR. Note  
 246 that setting  $d_c = d_s$  results in non-  
 247 adaptive regular IVF. Intuitively, we  
 248 would set  $d_c \ll d_s$ , so that instead  
 249 of clustering with a high-dimensional  
 250 representation, we can approximate it  
 251 accurately with a low-dimensional em-  
 252 bedding of size  $d_c$  followed by a lin-  
 253 ear scan with a higher  $d_s$ -dimensional  
 254 representation. Intuitively, this helps  
 255 in the smooth search of design space  
 256 for state-of-the-art accuracy-compute  
 257 trade-off. Furthermore, this can provide a precise operating point on accuracy-compute tradeoff curve  
 258 which is critical in several practical settings.

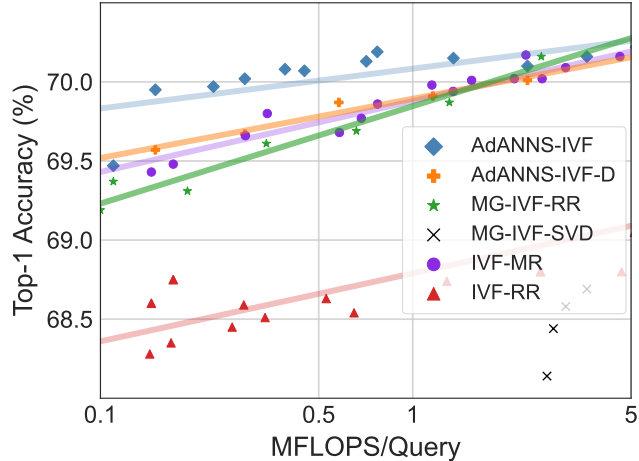


Figure 2: 1-NN accuracy on ImageNet retrieval shows that AdANNS-IVF achieves near-optimal accuracy-compute trade-off compared across various rigid and adaptive base-lines. Both adaptive variants of MR and RR significantly outperform their rigid counterparts (IVF-XX) while post-hoc compression on RR using SVD for adaptivity falls short.

259 Our experiments on regular IVF with MRs and RRs (IVF-MR & IVF-RR) of varying dimensionali-  
 260 ties and IVF configurations (# clusters, # probes) show that (Figure 2) matryoshka representations  
 261 result in a significantly better accuracy-compute trade-off. We further studied and found that learned  
 262 lower-dimensional representations offer better accuracy-compute trade-offs for IVF than higher-  
 263 dimensional embeddings (see Appendix E for more results).

264 AdANNS utilizes  $d$ -dimensional matryoshka representation to get accurate  $d_c$  and  $d_s$  dimensional  
 265 vectors at no extra compute cost. The resulting AdANNS-IVF provides a much better accuracy-  
 266 compute trade-off (Figure 2) on ImageNet-1K retrieval compared to IVF-MR, IVF-RR, and MG-  
 267 IVF-RR – multi-granular IVF with rigid representations (akin to AdANNS without MR) – a strong  
 268 baseline that uses  $d_c$  and  $d_s$  dimensional RRs. Finally, we exhaustively search the design space of  
 269 IVF by varying  $d_c, d_s \in [8, 16, \dots, 2048]$  and the number of clusters  $k \in [8, 16, \dots, 2048]$ . Please  
 270 see Appendix F for more details. For IVF experiments on the NQ dataset, please refer to Appendix H.

271 **Empirical results.** Figure 2 shows that AdANNS-IVF outperforms the baselines across all  
 272 accuracy-compute settings for ImageNet-1K retrieval. AdANNS-IVF results in  $10\times$  lower compute  
 273 for the best accuracy of the extremely expensive MG-IVF-RR and non-adaptive IVF-MR. Specifi-  
 274 cally, as shown in Figure 1a, AdANNS-IVF is up to 1.5% more accurate for the same compute and  
 275 has up to  $100\times$  lesser FLOPS/query ( $90\times$  real-world speed-up!) than the status quo ANNS on rigid  
 276 representations (IVF-RR). We filter out points for the sake of presentation and encourage the reader  
 277 to check out Figure 10 in Appendix F for an expansive plot of all the configurations searched.

278 The advantage of AdANNS for construction of search structures is evident from the improvements  
 279 in IVF (AdANNS-IVF) and can be easily extended to other ANNS structures like ScaNN [15] and  
 280 HNSW [37]. For example, HNSW consists of multiple layers with graphs of NSW graphs [36] of  
 281 increasing complexity. AdANNS can be adopted to HNSW, where the construction of each level can  
 282 be powered by appropriate dimensionalities for an optimal accuracy-compute trade-off. In general,  
 283 AdANNS provides fine-grained control over compute overhead (storage, working memory, inference,  
 284 and construction cost) during construction and inference while providing the best possible accuracy.

285 **4.2 AdANNS-OPQ**

286 Standard Product Quantization (PQ) essentially performs block-wise vector quantization via cluster-  
 287 ing. For example, suppose we need 32-byte PQ compressed vectors from the given 2048 dimensional  
 288 representations. Then, we can chunk the representations in 32 equal blocks/sub-vectors of 64-d each,  
 289 and each sub-vector space is clustered into  $2^8 = 256$  partitions. That is, the representation of each  
 290 point is essentially cluster-id for each block. Optimized PQ (OPQ) [12] further refines this idea, by  
 291 first rotating the representations using a learned orthogonal matrix, and then applying PQ on top of  
 292 the rotated representations. In ANNS, OPQ is used extensively to compress vectors and improves  
 293 approximate distance computation primarily due to significantly lower memory overhead than storing  
 294 full-precision data points IVF.

295 AdANNS-OPQ utilizes MR representations to apply OPQ on lower-dimensional representations.  
 296 That is, for a given quantization budget, AdANNS allows using top  $d_s \ll d$  dimensions from MR and  
 297 then computing clusters with  $d_s/B$ -dimensional blocks where  $B$  is the number of blocks. Depending  
 298 on  $d_s$  and  $B$ , we have further flexibility of trading-off dimensionality/capacity for increasing the  
 299 number of clusters to meet the given quantization budget. AdANNS-OPQ tries multiple  $d_s$ ,  $B$ , and  
 300 number of clusters for a fixed quantization budget to obtain the best performing configuration.

301 We experimented with 8 – 128 byte OPQ budgets for both ImageNet and Natural Questions retrieval  
 302 with an exhaustive search on the quantized vectors. We compare AdANNS-OPQ which uses MRs of  
 303 varying granularities to the baseline OPQ built on the highest dimensional RRs. We also evaluate  
 304 OPQ vectors obtained projection using SVD [13] on top of the highest-dimensional RRs.

305 **Empirical results.** Figures 3 and 1b show that AdANNS-OPQ significantly outperforms – up to  
 306 4% accuracy gain – the baselines (OPQ on RRs) across compute budgets on both ImageNet and NQ.  
 307 In particular, AdANNS-OPQ tends to match the accuracy of a 64-byte (a typical choice in ANNS)  
 308 OPQ baseline with only a 32-byte budget. This results in a  $2\times$  reduction in both storage and compute  
 309 FLOPS which translates to significant gains in real-world web-scale deployment (see Appendix D).

310 We only report the best AdANNS-OPQ for each budget typically obtained through a much lower-  
 311 dimensional MR (128 & 192; much faster to build as well) than the highest-dimensional MR (2048  
 312 & 768) for ImageNet and NQ respectively (see Appendix H for more details). At the same time, we  
 313 note that building compressed OPQ vectors on projected RRs using SVD to the smaller dimensions  
 314 (or using low-dimensional RRs, see Appendix D) as the optimal AdANNS-OPQ does not help in  
 315 improving the accuracy. The significant gains we observe in AdANNS-OPQ are purely due to better  
 316 information packing in MRs – we hypothesize that packing the most important information in the  
 317 initial coordinates results in a better PQ quantization than RRs where the information is uniformly  
 318 distributed across all the dimensions [30, 47]. See Appendix D for more details and experiments.

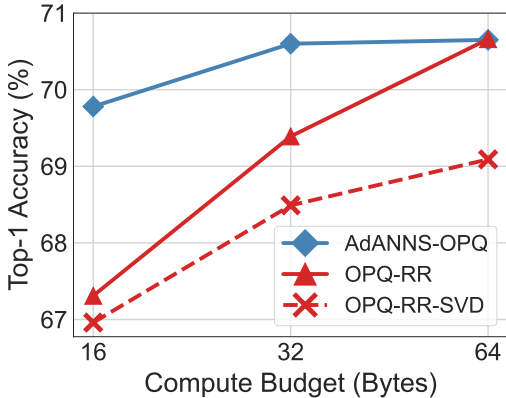


Figure 3: AdANNS-OPQ matches the accuracy of 64-byte OPQ on RR using only 32-bytes for ImageNet retrieval. AdANNS provides large gains at lower compute budgets and saturates to baseline performance for larger budgets.

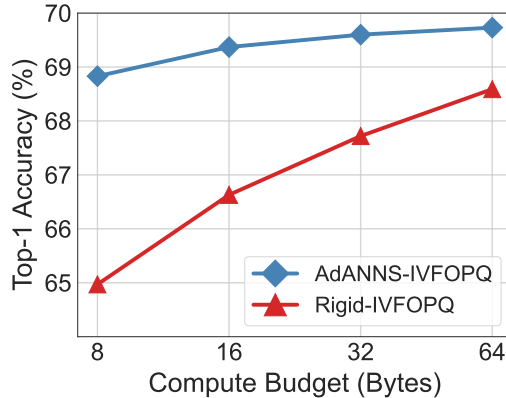


Figure 4: Combining the gains of AdANNS for IVF and OPQ leads to better IVFOPQ composite indices. On ImageNet retrieval, AdANNS-IVFOPQ is  $8\times$  cheaper for the same accuracy and provides 1 - 4% gains over IVFOPQ on RRs.

### 319 4.3 AdANNS for Composite Indices

320 We now extend AdANNS to composite indices [23] which put together two main ANNS building  
321 blocks – search structures and quantization – together to obtain efficient web-scale ANNS indices  
322 used in practice. A simple instantiation of a composite index would be the combination of IVF and  
323 OPQ – IVFOPQ – where the clustering in IVF happens with full-precision real vectors but the linear  
324 scan within each cluster is approximated using OPQ-compressed variants of the representation –  
325 since often the full-precision vectors of the database cannot fit in RAM. Contemporary ANNS indices  
326 like DiskANN [21] make this a default choice where they build the search graph with a full-precision  
327 vector and approximate the distance computations during search with an OPQ-compressed vector to  
328 obtain a very small shortlist of retrieved datapoints. In DiskANN, the shortlist of data points is then  
329 re-ranked to form the final list using their full-precision vectors fetched from the disk. AdANNS is  
330 naturally suited to this shortlist-rerank framework: we use a low- $d$  MR for forming index, where we  
331 could tune AdANNS parameters according to the accuracy-compute trade-off of the graph and OPQ  
332 vectors. We then use a high- $d$  MR for re-ranking.

333 **Empirical results.** Figure 4 shows  
334 that AdANNS-IVFOPQ is 1 – 4%  
335 better than the baseline at all the  
336 PQ compute budgets. Furthermore,  
337 AdANNS-IVFOPQ has the same accu-  
338 racy as the baselines at  $8\times$  lower  
339 overhead. With DiskANN, AdANNS  
340 accelerates shortlist generation by us-  
341 ing low-dimensional representations  
342 and recoups the accuracy by re-  
343 ranking with the highest-dimensional

344 MR at negligible cost. Table 1 shows that AdANNS-DiskANN is more accurate than the baseline for  
345 both 1-NN and ranking performance at only *half* the cost. Using low-dimensional representations  
346 further speeds up inference in AdANNS-DiskANN (see Appendix G).

347 These results show the generality of AdANNS and its broad applicability across a variety of ANNS  
348 indices built on top of the base building blocks. Currently, AdANNS piggybacks on typical ANNS  
349 pipelines for their inherent accounting of the real-world system constraints [15, 21, 24]. However,  
350 we believe that AdANNS’s flexibility and significantly better accuracy-compute trade-off can be  
351 further informed by real-world deployment constraints. We leave this high-potential line of work that  
352 requires extensive study to future research.

## 353 5 Further Analysis and Discussion

### 354 5.1 Compute-aware Elastic Search During Inference

355 AdANNS search structures cater to many specific large-scale use scenarios that need to satisfy precise  
356 resource constraints during construction as well as inference. However, in many cases, construction  
357 and storage of the indices are not the bottlenecks or the user is unable to search the design space.  
358 In these settings, AdANNS-D enables adaptive inference through accurate yet cheaper distance  
359 computation using the low-dimensional prefix of matryoshka representation. Akin to composite  
360 indices (Section 4.3) that use PQ vectors for cheaper distance computation, we can use the low-  
361 dimensional MR for faster distance computation on ANNS structure built *non-adaptively* with a  
362 high-dimensional MR without any modifications to the existing index.

363 **Empirical results.** Figure 2 shows that for a given compute budget using IVF on ImageNet-1K  
364 retrieval, AdANNS-IVF is better than AdANNS-IVF-D due to the explicit control during the building  
365 of the ANNS structure which is expected. However, the interesting observation is that AdANNS-D  
366 *matches or outperforms* the IVF indices built with MRs of varying capacities for ImageNet retrieval.

367 However, these methods are applicable in specific scenarios of deployment. Obtaining optimal  
368 AdANNS search structure (highly accurate) or even the best IVF-MR index relies on a relatively  
369 expensive design search but delivers indices that fit the storage, memory, compute, and accuracy  
370 constraints all at once. On the other hand AdANNS-D does not require a precisely built ANNS index  
371 but can enable compute-aware search during inference. AdANNS-D is a great choice for setups that

Table 1: AdANNS-DiskANN using a 16- $d$  MR + re-ranking with the 2048- $d$  MR outperforms DiskANN built on 2048- $d$  RR at *half* the compute cost on ImageNet retrieval.

|                    | RR-2048 | AdANNS       |
|--------------------|---------|--------------|
| PQ Budget (Bytes)  | 32      | <b>16</b>    |
| Top-1 Accuracy (%) | 70.37   | <b>70.56</b> |
| mAP@10 (%)         | 62.46   | <b>64.70</b> |
| Precision@40 (%)   | 65.65   | <b>68.25</b> |



372 can afford only one single database/index but need to cater to varying deployment constraints, e.g.,  
373 one task requires 70% accuracy while another task has a compute budget of 1 MFLOPS/query.

## 374 5.2 Why MRs over RRs?

375 Quite a few of the gains from AdANNS are owing to the quality and capabilities of matryoshka  
376 representations. So, we conducted extensive analysis to understand why matryoshka representations  
377 seem to be more aligned for semantic search than the status-quo rigid representations.

378 **Difficulty of NN search.** Relative contrast ( $C_r$ ) [17] is inversely proportional to the difficulty of  
379 nearest neighbor search on a given database. On ImageNet-1K, Figure 13 shows that MRs have  
380 better  $C_r$  than RRs across dimensionalities, further supporting that matryoshka representations are  
381 more aligned (easier) for NN search than existing rigid representations for the same accuracy. More  
382 details and analysis about this experiment can be found in Appendix I.2.

383 **Clustering distributions.** We also investigate the potential deviation in clustering distributions  
384 for MRs across dimensionalities compared to RRs. Unlike the RRs where the information is  
385 uniformly diffused across dimensions [47], MRs have hierarchical information packing. Figure 9 in  
386 Appendix E.3 shows that matryoshka representations result in clusters similar (measured by total  
387 variation distance [32]) to that of rigid representations and do not result in any unusual artifacts.

388 **Robustness.** Figure 7 in Appendix E shows that MRs continue to be better than RRs even for out-  
389 of-distribution (OOD) image queries (ImageNetV2 [42]) using ANNS. It also shows that the highest  
390 data dimensionality need not always be the most robust which is further supported by the higher  
391 recall using lower dimensions. Further details about this experiment can be found in Appendix E.1.

392 **Generality across encoders.** IVF-MR consistently has higher accuracy than IVF-RR across dimen-  
393 sionalities despite having similar accuracies with exact NN search (for ResNet50 on ImageNet and  
394 BERT-Base on NQ). We find that our observations on better alignment of MRs for NN search hold  
395 across neural network architectures, ResNet18/34/101 [18] and ConvNeXt-Tiny [34]. Appendix I.3  
396 delves deep into the experimentation done using various neural architectures on ImageNet-1K.


397 **Recall score analysis.** Analysis of recall score (see Appendix C) in Appendix I.1 shows that for  
398 a similar top-1 accuracy, lower-dimensional representations have better 1-Recall@1 across search  
399 complexities for IVF and HNSW on ImageNet-1K. Across the board, MRs have higher recall  
400 scores and top-1 accuracy pointing to easier “searchability” and thus suitability of matryoshka  
401 representations for ANNS. Larger-scale experiments and further analysis can be found in Appendix I.

402 Through these analyses, we argue that matryoshka representations are better suited for semantic  
403 search than rigid representations, thus making them an ideal choice for AdANNS.

## 404 5.3 Limitations

405 AdANNS’s core focus is to improve the design of the existing ANNS pipelines. To use AdANNS  
406 on a corpus, we need to back-fill [41] the MRs of the data – a significant yet a one-time overhead.  
407 We also notice that high-dimensional MRs start to degrade in performance when optimizing also for  
408 an extremely low-dimensional granularity (e.g.,  $< 24$ -d for NQ) – otherwise is it quite easy to have  
409 comparable accuracies with both RRs and MRs. Lastly, the existing dense representations can only  
410 in theory be converted to MRs with an auto-encoder-style non-linear transformation. We believe  
411 most of these limitations form excellent future work to improve AdANNS further.

## 412 6 Conclusions

413 We proposed a novel framework, AdANNS , that leverages adaptive representations for different  
414 phases of ANNS pipelines to improve the accuracy-compute tradeoff. AdANNS utilizes the inherent  
415 flexibility of matryoshka representations [30] to design better ANNS building blocks than the standard  
416 ones which use the rigid representation in each phase. AdANNS achieves SOTA accuracy-compute  
417 trade-off for the two main ANNS building blocks: search data structures (AdANNS-IVF) and  
418 quantization (AdANNS-OPQ). Finally, the combination of AdANNS-based building blocks leads to  
419 the construction of better real-world composite ANNS indices – with as much as  $8\times$  reduction in  
420 cost at the same accuracy as strong baselines – while also enabling compute-aware elastic search.

## References

- 421
- 422 [1] M. Aumüller, E. Bernhardsson, and A. Faithfull. Ann-benchmarks: A benchmarking tool for  
423 approximate nearest neighbor algorithms. *Information Systems*, 87:101374, 2020.
- 424 [2] Y. Bengio. Deep learning of representations for unsupervised and transfer learning. In *Proceed-*  
425 *ings of ICML workshop on unsupervised and transfer learning*, pages 17–36. JMLR Workshop  
426 and Conference Proceedings, 2012.
- 427 [3] E. Bernhardsson. *Annoy: Approximate Nearest Neighbors in C++/Python*, 2018. URL  
428 <https://pypi.org/project/annoy/>. Python package version 1.13.0.
- 429 [4] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer*  
430 *networks and ISDN systems*, 30(1-7):107–117, 1998.
- 431 [5] D. Cai. A revisit of hashing algorithms for approximate nearest neighbor search. *IEEE*  
432 *Transactions on Knowledge and Data Engineering*, 33(6):2337–2348, 2021. doi: 10.1109/  
433 TKDE.2019.2953897.
- 434 [6] T. Chen, L. Li, and Y. Sun. Differentiable product quantization for end-to-end embedding  
435 compression. In *International Conference on Machine Learning*, pages 1617–1626. PMLR,  
436 2020.
- 437 [7] K. L. Clarkson. An algorithm for approximate closest-point queries. In *Proceedings of the tenth*  
438 *annual symposium on Computational geometry*, pages 160–164, 1994.
- 439 [8] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. Locality-sensitive hashing scheme based  
440 on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational*  
441 *geometry*, pages 253–262, 2004.
- 442 [9] J. Dean. Challenges in building large-scale information retrieval systems. In *Keynote of the 2nd*  
443 *ACM International Conference on Web Search and Data Mining (WSDM)*, volume 10, 2009.
- 444 [10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional  
445 transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- 446 [11] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in  
447 logarithmic expected time. *ACM Transactions on Mathematical Software (TOMS)*, 3(3):209–  
448 226, 1977.
- 449 [12] T. Ge, K. He, Q. Ke, and J. Sun. Optimized product quantization for approximate nearest  
450 neighbor search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern*  
451 *Recognition*, pages 2946–2953, 2013.
- 452 [13] G. Golub and W. Kahan. Calculating the singular values and pseudo-inverse of a matrix. *Journal*  
453 *of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis*, 2(2):  
454 205–224, 1965.
- 455 [14] R. Gray. Vector quantization. *IEEE Assp Magazine*, 1(2):4–29, 1984.
- 456 [15] R. Guo, P. Sun, E. Lindgren, Q. Geng, D. Simcha, F. Chern, and S. Kumar. Accelerating  
457 large-scale inference with anisotropic vector quantization. In *International Conference on*  
458 *Machine Learning*, pages 3887–3896. PMLR, 2020.
- 459 [16] N. Gupta, P. H. Chen, H.-F. Yu, C.-J. Hsieh, and I. S. Dhillon. End-to-end learning to index and  
460 search in large output spaces. *arXiv preprint arXiv:2210.08410*, 2022.
- 461 [17] J. He, S. Kumar, and S.-F. Chang. On the difficulty of nearest neighbor search. In *International*  
462 *Conference on Machine Learning (ICML)*, 2012.
- 463 [18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In  
464 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–  
465 778, 2016.

- 466 [19] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual  
467 representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and*  
468 *pattern recognition*, pages 9729–9738, 2020.
- 469 [20] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of  
470 dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*,  
471 pages 604–613, 1998.
- 472 [21] S. Jayaram Subramanya, F. Devvrit, H. V. Simhadri, R. Krishnawamy, and R. Kadekodi.  
473 Diskann: Fast accurate billion-point nearest neighbor search on a single node. *Advances in*  
474 *Neural Information Processing Systems*, 32, 2019.
- 475 [22] H. Jegou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE*  
476 *transactions on pattern analysis and machine intelligence*, 33(1):117–128, 2010.
- 477 [23] J. Johnson, M. Douze, and H. Jégou. Billion-scale similarity search with GPUs. *IEEE*  
478 *Transactions on Big Data*, 7(3):535–547, 2019.
- 479 [24] W. B. Johnson. Extensions of lipschitz mappings into a hilbert space. *Contemp. Math.*, 26:  
480 189–206, 1984.
- 481 [25] I. T. Jolliffe and J. Cadima. Principal component analysis: a review and recent developments.  
482 *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering*  
483 *Sciences*, 374(2065):20150202, 2016.
- 484 [26] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih. Dense  
485 passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*, 2020.
- 486 [27] S. Kornblith, J. Shlens, and Q. V. Le. Do better imagenet models transfer better? In *Proceedings*  
487 *of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2661–2671,  
488 2019.
- 489 [28] T. Kraska, A. Beutel, E. H. Chi, J. Dean, and N. Polyzotis. The case for learned index structures.  
490 In *Proceedings of the 2018 international conference on management of data*, pages 489–504,  
491 2018.
- 492 [29] A. Kusupati, M. Wallingford, V. Ramanujan, R. Somani, J. S. Park, K. Pillutla, P. Jain, S. Kakade,  
493 and A. Farhadi. Llc: Accurate, multi-purpose learnt low-dimensional binary codes. *Advances*  
494 *in Neural Information Processing Systems*, 34:23900–23913, 2021.
- 495 [30] A. Kusupati, G. Bhatt, A. Rege, M. Wallingford, A. Sinha, V. Ramanujan, W. Howard-Snyder,  
496 K. Chen, S. Kakade, P. Jain, and A. Farhadi. Matryoshka representation learning. In *Advances*  
497 *in Neural Information Processing Systems*, December 2022.
- 498 [31] T. Kwiakowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein,  
499 I. Polosukhin, J. Devlin, K. Lee, et al. Natural questions: a benchmark for question answering  
500 research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.
- 501 [32] D. A. Levin and Y. Peres. *Markov chains and mixing times*, volume 107. American Mathematical  
502 Soc., 2017.
- 503 [33] W. Li, Y. Zhang, Y. Sun, W. Wang, W. Zhang, and X. Lin. Approximate nearest neighbor search  
504 on high dimensional data—experiments, analyses, and improvement. *IEEE Transactions on*  
505 *Knowledge and Data Engineering*, 2020.
- 506 [34] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. A convnet for the 2020s. In  
507 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages  
508 11976–11986, 2022.
- 509 [35] S. Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):  
510 129–137, 1982.
- 511 [36] Y. Malkov, A. Ponomarenko, A. Logvinov, and V. Krylov. Approximate nearest neighbor  
512 algorithm based on navigable small world graphs. *Information Systems*, 45:61–68, 2014.

- 513 [37] Y. A. Malkov and D. Yashunin. Efficient and robust approximate nearest neighbor search using  
514 hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis & Machine*  
515 *Intelligence*, 42(04):824–836, 2020.
- 516 [38] P. Nayak. Understanding searches better than ever before. *Google AI Blog*, 2019. URL [https://  
517 blog.google/products/search/search-language-understanding-bert/](https://blog.google/products/search/search-language-understanding-bert/).
- 518 [39] A. Neelakantan, T. Xu, R. Puri, A. Radford, J. M. Han, J. Tworek, Q. Yuan, N. Tezak, J. W.  
519 Kim, C. Hallacy, et al. Text and code embeddings by contrastive pre-training. *arXiv preprint*  
520 *arXiv:2201.10005*, 2022.
- 521 [40] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell,  
522 P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision.  
523 In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.
- 524 [41] V. Ramanujan, P. K. A. Vasu, A. Farhadi, O. Tuzel, and H. Pouransari. Forward compatible  
525 training for representation learning. *arXiv preprint arXiv:2112.02805*, 2021.
- 526 [42] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar. Do imagenet classifiers generalize to  
527 imagenet? In *International Conference on Machine Learning*, pages 5389–5400. PMLR, 2019.
- 528 [43] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy,  
529 A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International*  
530 *journal of computer vision*, 115(3):211–252, 2015.
- 531 [44] R. Salakhutdinov and G. Hinton. Semantic hashing. *International Journal of Approximate*  
532 *Reasoning*, 50(7):969–978, 2009.
- 533 [45] H. V. Simhadri, G. Williams, M. Aumüller, M. Douze, A. Babenko, D. Baranchuk, Q. Chen,  
534 L. Hosseini, R. Krishnaswamy, G. Srinivasa, et al. Results of the neurips’21 challenge on  
535 billion-scale approximate nearest neighbor search. *arXiv preprint arXiv:2205.03763*, 2022.
- 536 [46] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos.  
537 In *Computer Vision, IEEE International Conference on*, volume 3, pages 1470–1470. IEEE  
538 Computer Society, 2003.
- 539 [47] D. Soudry, E. Hoffer, M. S. Nacson, S. Gunasekar, and N. Srebro. The implicit bias of gradient  
540 descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878, 2018.
- 541 [48] C. Waldburger. As search needs evolve, microsoft makes ai tools for better search available  
542 to researchers and developers. *Microsoft AI Blog*, 2019. URL [https://blogs.microsoft.  
543 com/ai/bing-vector-search/](https://blogs.microsoft.com/ai/bing-vector-search/).
- 544 [49] M. Wang, X. Xu, Q. Yue, and Y. Wang. A comprehensive survey and experimental comparison  
545 of graph-based approximate nearest neighbor search. *Proceedings of the VLDB Endowment*, 14  
546 (11):1964–1978, 2021.
- 547 [50] R. Weber, H.-J. Schek, and S. Blott. A quantitative analysis and performance study for similarity-  
548 search methods in high-dimensional spaces. In *VLDB*, volume 98, pages 194–205, 1998.
- 549 [51] I. H. Witten, I. H. Witten, A. Moffat, T. C. Bell, T. C. Bell, E. Fox, and T. C. Bell. *Managing*  
550 *gigabytes: compressing and indexing documents and images*. Morgan Kaufmann, 1999.

**Algorithm 1** AdANNS-IVF Psuedocode

```

# Index database to construct clusters and build inverted file system

def adannsConstruction(database, d_cluster, num_clusters):
    # Slice database with cluster construction dim (d_cluster)
    xb = database[:d_cluster]
    cluster_centroids = constructClusters(xb, num_clusters)

    return cluster_centroids

def adannsInference(queries, centroids, d_shortlist, d_search, num_probes,
                    k):
    # Slice queries and centroids with cluster shortlist dim (d_shortlist)
    xq = queries[:d_shortlist]
    xc = centroids[:d_shortlist]

    for q in queries:
        # compute distance of query from each cluster centroid
        candidate_distances = computeDistances(q, xc)
        # sort cluster candidates by distance and choose small number to
        # probe
        cluster_candidates = sortAscending(candidate_distances)[:num_probes]
        database_candidates = getClusterMembers(cluster_candidates)
        # Linear Scan all shortlisted clusters with search dim (d_search)
        k_nearest_neighbors[q] = linearScan(q, database_candidates, d_search,
                                           k)

    return k_nearest_neighbors

```

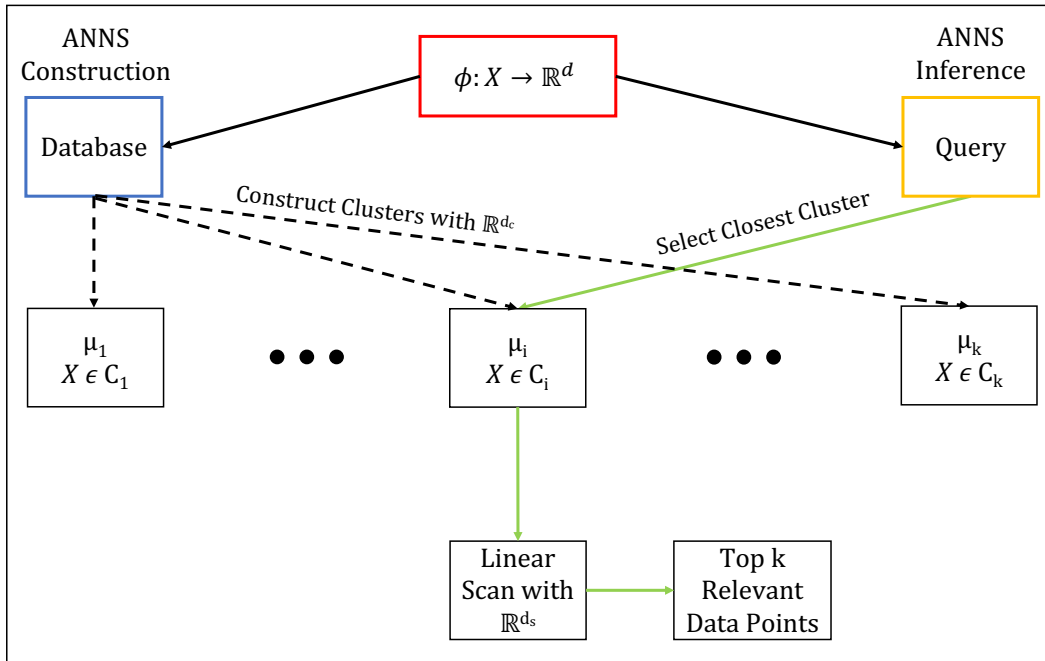


Figure 5: The schematic of inverted file index (IVF) outlaying the construction and inference phases. Adaptive representations can be utilized effectively in the decoupled components of clustering and searching for a better accuracy-compute trade-off (AdANNS-IVF).

Table 2: Mathematical formulae of the retrieval phase across various methods built on IVF. See Section 3 for notations.

| Method       | Retrieval Formula during Inference   |
|--------------|--|
| IVF-RR       | $\arg \min_{j \in C_{h(q)}} \ \phi^{\text{RR}(d)}(q) - \phi^{\text{RR}(d)}(x_j)\ $ , s.t. $h(q) = \arg \min_h \ \phi^{\text{RR}(d)}(q) - \mu_h^{\text{RR}(d)}\ $   |
| IVF-MR       | $\arg \min_{j \in C_{h(q)}} \ \phi^{\text{MR}(d)}(q) - \phi^{\text{MR}(d)}(x_j)\ $ , s.t. $h(q) = \arg \min_h \ \phi^{\text{MR}(d)}(q) - \mu_h^{\text{MR}(d)}\ $   |
| AdANNS-IVF   | $\arg \min_{j \in C_{h(q)}} \ \phi^{\text{MR}(d_s)}(q) - \phi^{\text{MR}(d_s)}(x_j)\ $ , s.t. $h(q) = \arg \min_h \ \phi^{\text{MR}(d_s)}(q) - \mu_h^{\text{MR}(d_s)}\ $   |
| MG-IVF-RR    | $\arg \min_{j \in C_{h(q)}} \ \phi^{\text{RR}(d_s)}(q) - \phi^{\text{RR}(d_s)}(x_j)\ $ , s.t. $h(q) = \arg \min_h \ \phi^{\text{RR}(d_s)}(q) - \mu_h^{\text{RR}(d_s)}\ $   |
| AdANNS-IVF-D | $\arg \min_{j \in C_{h(q)}} \ \phi^{\text{MR}(d)}(q)[1 : \hat{d}] - \phi^{\text{MR}(d)}(x_j)[1 : \hat{d}]\ $ , s.t. $h(q) = \arg \min_h \ \phi^{\text{MR}(d)}(q)[1 : \hat{d}] - \mu_h^{\text{MR}(d)}[1 : \hat{d}]\ $ |
| IVFOPQ       | $\arg \min_{j \in C_{h(q)}} \ \phi^{\text{PQ}(m,b)}(q) - \phi^{\text{PQ}(m,b)}(x_j)\ $ , s.t. $h(q) = \arg \min_h \ \phi(q) - \mu_h\ $   |

## 552 B Training and Compute Costs

553 A bulk of our ANNS experimentation was written with Faiss [23], a library for efficient similarity  
 554 search and clustering. AdANNS was implemented from scratch due to difficulty in decoupling  
 555 clustering and linear scan with Faiss, with code available at [redacted for double blind]. We also  
 556 provide a version of AdANNS with Faiss optimizations with the restriction that  $D_c \geq D_s$  as a  
 557 limitation of the current implementation. All ANNS experiments (AdANNS-IVF, MG-IVF-RR,  
 558 IVF-MR, IVF-RR, HNSW, HNSWOPQ, IVFOPQ) were run on an Intel Xeon 2.20GHz CPU with  
 559 12 cores. Exact Search (Flat L2, PQ, OPQ) and DiskANN experiments were run with CUDA 11.0 on  
 560 a A100-SXM4 NVIDIA GPU with 40G RAM. The wall-clock inference times quoted in Figure 1a  
 561 and Table 3 are reported on CPU with Faiss optimizations, and are averaged over three inference runs  
 562 for ImageNet-1K retrieval.

Table 3: Comparison of AdANNS-IVF and Rigid-IVF wall-clock inference times for ImageNet-1K retrieval. AdANNS-IVF has up to  $\sim 1.5\%$  gain over Rigid-IVF for a fixed search latency per query.

| AdANNS-IVF |                           | Rigid-IVF |                           |
|------------|---------------------------|-----------|---------------------------|
| Top-1      | Search Latency/Query (ms) | Top-1     | Search Latency/Query (ms) |
| 70.02      | 0.03                      | 68.51     | 0.02                      |
| 70.08      | 0.06                      | 68.54     | 0.05                      |
| 70.19      | 0.06                      | 68.74     | 0.08                      |
| 70.36      | 0.88                      | 69.20     | 0.86                      |
| 70.60      | 5.57                      | 70.13     | 5.67                      |

563 **DPR [26] on NQ [31]**. We follow the setup on the DPR repo<sup>1</sup>: the Wikipedia corpus has 21 million  
 564 passages and Natural Questions dataset for open-domain QA settings. The training set contains  
 565 79,168 question and answer pairs, the dev set has 8,757 pairs and the test set has 3,610 pairs.

## 566 C Evaluation Metrics

567 In this work, we primarily use top-1 accuracy (i.e. 1-Nearest Neighbor), recall@k, corrected mean  
 568 average precision (mAP@k) [29] and k-Recall@N (recall score), which are defined over all queries  
 569  $Q$  over indexed database of size  $N_D$  as:

$$\text{top-1} = \frac{\sum_Q \text{correct\_pred@1}}{|Q|}$$

$$\text{Recall@k} = \frac{\sum_Q \text{correct\_pred@k}}{|Q|} * \frac{\text{num\_classes}}{|N_D|}$$

570 where  $\text{correct\_pred@k}$  is the number of k-NN with correctly predicted labels for a given query. As  
 571 noted in Section 3, k-Recall@N is the overlap between  $k$  exact search nearest neighbors (considered

<sup>1</sup><https://github.com/facebookresearch/DPR>

572 as ground truth) and the top N retrieved documents. As Faiss [23] supports a maximum of 2048-  
573 NN while searching the indexed database, we report 40-Recall@2048 in Figure 12. Also note  
574 that for ImageNet-1K, which constitutes a bulk of the experimentation in this work,  $|Q| = 50000$ ,  
575  $|N_D| = 1281167$  and  $\text{num\_classes} = 1000$ . For ImageNetv2 [42],  $|Q| = 10000$  and  $\text{num\_classes}$   
576  $= 1000$ , and for ImageNet-4K [30],  $|Q| = 210100$ ,  $|N_D| = 4202000$  and  $\text{num\_classes} = 4202$ . For  
577 NQ [31],  $|Q| = 3610$  and  $|N_D| = 21015324$ . As NQ consists of question-answer pairs (instance-  
578 level),  $\text{num\_classes} = 3610$  for the test set.

## 579 D AdANNS-OPQ

580 In this section, we take a deeper dive into the quantization characteristics of MR. In this work, we  
581 restrict our focus to optimized product quantization (OPQ) [12], which adds a learned space rotation  
582 and dimensionality permutation to PQ’s sub-vector quantization to learn more optimal PQ codes. We  
583 compare OPQ to vanilla PQ on ImageNet in Table 4, and observe large gains at larger embedding  
dimensionalities, which agrees with the findings of Jayaram Subramanya et al. [21].

Table 4: Comparison of PQ-MR with OPQ-MR for exact search on ImageNet-1K across embedding  
dimensionality  $d \in \{8, 16, \dots, 2048\}$  quantized to  $m \in \{8, 16, 32, 64\}$  bytes. OPQ shows large gains  
over vanilla PQ at larger embedding dimensionalities  $d \geq 128$ . Entries where OPQ outperforms PQ  
on top-1 accuracy are bolded.

| Config |    | PQ           |        |       | OPQ          |        |              |
|--------|----|--------------|--------|-------|--------------|--------|--------------|
| d      | m  | Top-1        | mAP@10 | P@100 | Top-1        | mAP@10 | P@100        |
| 8      | 8  | 62.18        | 56.71  | 61.23 | <b>62.22</b> | 56.70  | 61.23        |
| 16     | 8  | <b>67.91</b> | 62.85  | 67.21 | 67.88        | 62.96  | 67.21        |
|        | 16 | 67.85        | 62.95  | 67.21 | <b>67.96</b> | 62.94  | 67.21        |
| 32     | 8  | 68.80        | 63.62  | 67.86 | <b>68.91</b> | 63.63  | 67.86        |
|        | 16 | <b>69.57</b> | 64.22  | 68.12 | 69.47        | 64.20  | 68.12        |
|        | 32 | 69.44        | 64.20  | 68.12 | <b>69.47</b> | 64.23  | 68.12        |
| 64     | 8  | <b>68.39</b> | 63.40  | 67.47 | 68.38        | 63.42  | 67.60        |
|        | 16 | 69.77        | 64.43  | 68.25 | <b>69.95</b> | 64.55  | 68.38        |
|        | 32 | <b>70.13</b> | 64.67  | 68.38 | 70.05        | 64.65  | 68.38        |
|        | 64 | 70.12        | 64.69  | 68.42 | <b>70.18</b> | 64.70  | 68.38        |
| 128    | 8  | 67.27        | 61.99  | 65.78 | <b>68.40</b> | 63.11  | 67.34        |
|        | 16 | 69.51        | 64.32  | 68.12 | <b>69.78</b> | 64.56  | 68.38        |
|        | 32 | 70.27        | 64.72  | 68.51 | <b>70.60</b> | 64.97  | 68.51        |
|        | 64 | 70.61        | 64.93  | 68.49 | <b>70.65</b> | 64.98  | 68.51        |
| 256    | 8  | 66.06        | 60.44  | 64.09 | <b>67.90</b> | 62.69  | 66.95        |
|        | 16 | 68.56        | 63.33  | 66.95 | <b>69.92</b> | 64.71  | 68.51        |
|        | 32 | 70.08        | 64.83  | 68.38 | <b>70.59</b> | 65.15  | 68.64        |
|        | 64 | 70.48        | 64.98  | 68.55 | <b>70.69</b> | 65.09  | 68.64        |
| 512    | 8  | 65.09        | 59.03  | 62.53 | <b>67.51</b> | 62.12  | 66.56        |
|        | 16 | 67.68        | 62.11  | 65.39 | <b>69.67</b> | 64.53  | 68.38        |
|        | 32 | 69.51        | 64.01  | 67.34 | <b>70.44</b> | 65.11  | 68.64        |
|        | 64 | 70.53        | 65.02  | 68.52 | <b>70.72</b> | 65.17  | 68.64        |
| 1024   | 8  | 64.58        | 58.26  | 61.75 | <b>67.26</b> | 62.07  | 66.56        |
|        | 16 | 66.84        | 61.07  | 64.09 | <b>69.34</b> | 64.23  | 68.12        |
|        | 32 | 68.71        | 62.92  | 66.04 | <b>70.43</b> | 65.03  | 68.64        |
|        | 64 | 69.88        | 64.35  | 67.68 | <b>70.81</b> | 65.19  | 68.64        |
| 2048   | 8  | 62.19        | 56.11  | 59.80 | <b>66.89</b> | 61.69  | 66.30        |
|        | 16 | 65.99        | 60.27  | 63.18 | <b>69.25</b> | 64.09  | 67.99        |
|        | 32 | 67.99        | 62.04  | 64.74 | <b>70.39</b> | 64.97  | 68.51        |
|        | 64 | 69.20        | 63.46  | 66.40 | <b>70.57</b> | 65.15  | <b>68.51</b> |

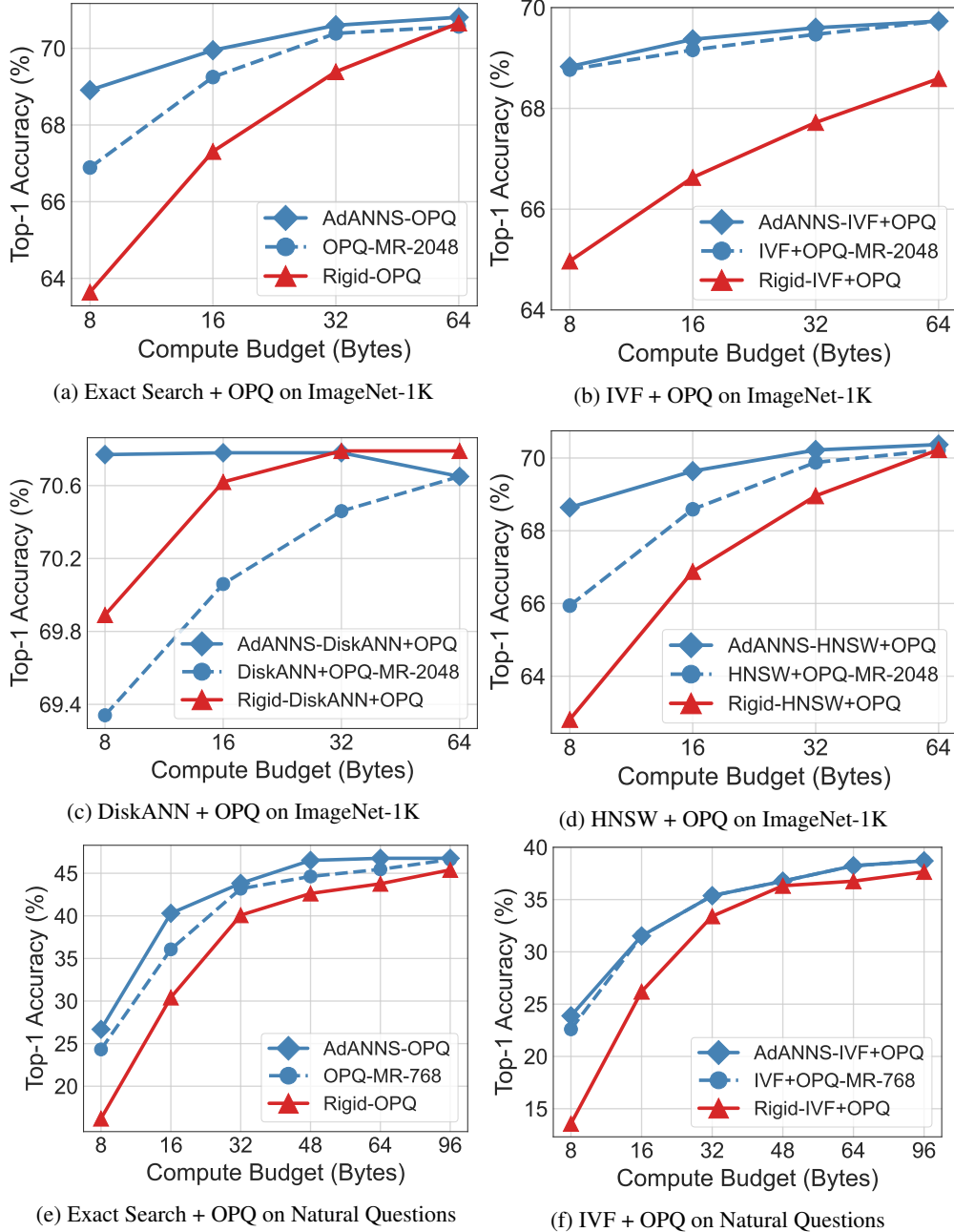


Figure 6: Top-1 Accuracy of AdANNS composite indices with OPQ distance computation compared to MR and Rigid baseline models on ImageNet-1K and Natural Questions.

584 We perform an exhaustive study of compression across embedding dimensionalities  $d$  for composite  
 585 OPQ  $m \times b$  indices on ImageNet-1K (where  $b = 8$ , i.e. 1 Byte), i.e. Exact Search with OPQ,  
 586 IVF+OPQ, HNSW+OPQ, and DiskANN+OPQ, as seen in Figure 6. It is evident from these results  
 587 that:

- 588 1. Learning OPQ codebooks with AdANNS (Figure 6a) provides a 1-5% gain in top-1 accuracy  
 589 over rigid representations at low compute budgets ( $\leq 32$  Bytes). AdANNS-OPQ saturates to  
 590 Rigid-OPQ performance at low compression ( $\geq 64$  Bytes).
- 591 2. For IVF, learning clusters with MRs instead of RRs (Figure 6b) provides substantial gains (1-  
 592 4%). In contrast to Exact-OPQ, using AdANNS for learning OPQ codebooks does not provide



593 substantial top-1 accuracy gains over MR with  $d = 2048$  (highest), though it is still slightly better  
 594 or equal to MR-2048 at all compute budgets. This further supports that IVF performance generally  
 595 scales with embedding dimensionality, which is consistent with our findings on ImageNet across  
 596 robustness variants and encoders (See Figures 7 and 14 respectively).

597 3. Note that in contrast to Exact, IVF, and HNSW coarse quantizers, DiskANN *inherently re-ranks*  
 598 the retrieved shortlist with high-precision embeddings ( $d = 2048$ ), which is reflected in its high  
 599 top-1 accuracy. We find that AdANNS with 8-byte OPQ (Figure 6c) matches the top-1 accuracy  
 600 of rigid representations using 32-byte OPQ, for a  $4\times$  cost reduction for the same accuracy. Also  
 601 note that using AdANNS provides large gains over using MR-2048 at high compression (1.5%),  
 602 highlighting the necessity of AdANNS’s flexibility for high-precision retrieval at low compute  
 603 budgets.

604 4. Our findings on the HNSW-OPQ composite index (Figure 6d) are consistent with all other indices,  
 605 i.e. HNSW graphs constructed with AdANNS OPQ codebooks provide significant gains over RR  
 606 and MR, especially at high compression ( $\leq 32$  Bytes).

### 607 OPQ on NQ dataset

608 Our observations on ImageNet with ResNet-50 MR across search structures also extend to the Natural  
 609 Questions dataset with Dense Passage Retriever (DPR with BERT-Base MR embeddings). We note  
 610 that AdANNS provides gains over RR-768 embeddings for both Exact Search and IVF with OPQ.  
 611 We find that similar to ImageNet (Figure 14) IVF performance on Natural Questions generally scales  
 612 with dimensionality. AdANNS thus reduces to MR-768 performance for  $M \geq 16$ . See Appendix H  
 613 for a more in-depth discussion of AdANNS with DPR on Natural Questions. .

## 614 E IVF

615 Inverted file index (IVF) [46] is a simple yet powerful ANNS data structure used in web-scale search  
 616 systems [15]. IVF construction involves clustering (coarse quantization often through k-means) [35]  
 617 on  $d$ -dimensional representation that results in an inverted file list [51] of all the data points in each  
 618 cluster. During search, the  $d$ -dimensional query representation is first assigned to the closest clusters  
 619 (# probes, typically set to 1) and then an exhaustive linear scan happens within each cluster to obtain  
 620 the nearest neighbors. As seen in Figure 7, IVF top-1 accuracy scales logarithmically with increasing  
 621 representation dimensionality  $d$  on ImageNet-1K/V2/4K. The learned low- $d$  representations thus  
 622 provide better accuracy-compute trade-offs compared to high- $d$  representations, thus furthering the  
 623 case for usage of AdANNS with IVF (Section F).

### 624 E.1 Robustness

625 As shown in Figure 7, we examined the clustering capabilities of MRs on both in-distribution (ID)  
 626 queries via ImageNet-1K and out-of-distribution (OOD) queries via ImageNetV2 [42], as well as  
 627 on larger-scale ImageNet-4K [30]. For ID queries on ImageNet-1K (Figure 7a), IVF-MR is at least  
 628 as accurate as Exact-RR for  $d \leq 256$  with a single search probe, demonstrating the quality of in-  
 629 distribution low- $d$  clustering with MR. On OOD queries (Figure 7b), we observe that IVF-MR is on  
 630 average 2% more robust than IVF-RR across all cluster construction and linear scan dimensionalities  
 631  $d$ . It is also notable that clustering with MRs followed by linear scan with # probes = 1 is more robust

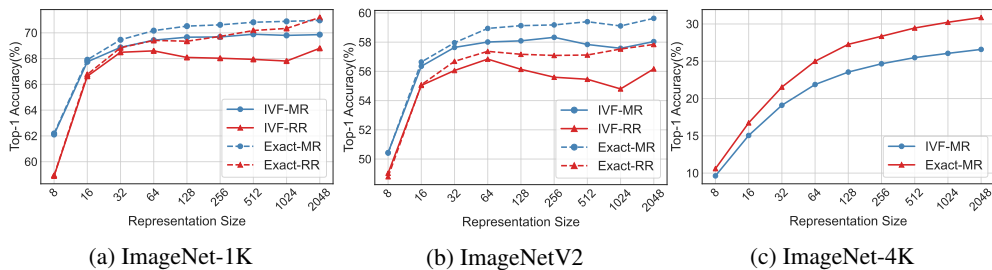


Figure 7: Top-1 Accuracy variation of IVF-MR of ImageNet 1K, ImageNetV2 and ImageNet-4K. RR baselines are omitted on ImageNet-4K due to high compute cost.

Table 5: Top-1 Accuracy of AdANNS-IVF-D on out-of-distribution queries from ImageNetV2 compared to both IVF and Exact Search with MR and RR embeddings. Note that for AdANNS-IVF-D, the dimensionality used to build clusters  $d_c = 2048$ .

| d    | AdANNS-IVF-D | IVF-MR | Exact-MR     | IVF-RR | Exact-RR |
|------|--------------|--------|--------------|--------|----------|
| 8    | <b>53.51</b> | 50.44  | 50.41        | 49.03  | 48.79    |
| 16   | <b>57.32</b> | 56.35  | 56.64        | 55.04  | 55.08    |
| 32   | 57.32        | 57.64  | <b>57.96</b> | 56.06  | 56.69    |
| 64   | 57.85        | 58.01  | <b>58.94</b> | 56.84  | 57.37    |
| 128  | 58.02        | 58.09  | <b>59.13</b> | 56.14  | 57.17    |
| 256  | 58.01        | 58.33  | <b>59.18</b> | 55.60  | 57.09    |
| 512  | 58.03        | 57.84  | <b>59.40</b> | 55.46  | 57.12    |
| 1024 | 57.66        | 57.58  | <b>59.11</b> | 54.80  | 57.53    |
| 2048 | 58.04        | 58.04  | <b>59.63</b> | 56.17  | 57.84    |

632 than exact search with RR embeddings across all  $d \leq 2048$ , indicating the adaptability of MRs to  
633 distribution shifts during inference. As seen in Table 5, on ImageNetV2 AdANNS-IVF-D is the best  
634 configuration for  $d \leq 16$ , and is similarly accurate to IVF-MR at all other  $d$ . AdANNS-IVF-D with  
635  $d = 128$  is able to match its own accuracy with  $d = 2048$ , a  $16\times$  compute gain during inference.  
636 This demonstrates the potential of AdANNS to adaptively search pre-indexed clustering structures.

637 On 4-million scale ImageNet-4K (Figure 7c), we observe similar accuracy trends of IVF-MR  
638 compared to Exact-MR as in ImageNet-1K (Figure 7a) and ImageNetV2 (Figure 7b). We omit  
639 baseline IVF-RR and Exact-RR experiments due to high compute cost at larger scale.

## 640 E.2 Ablations

641 As seen in Figure 8a, IVF-MR can match the accuracy of Exact Search on ImageNet-4K with  
642  $\sim 100\times$  less compute. We also explored the capability of MRs at retrieving cluster centroids with  
643 low-d compared to a ground truth of 2048-d with k-Recall@N, as seen in Figure 8b. MRs were able  
644 to saturate to near-perfect 1-Recall@N for  $d \geq 32$  and  $N \geq 4$ , indicating the potential of AdANNS  
645 at matching exact search performance with less than 10 search probes  $n_p$ .

## 646 E.3 Clustering Distribution

We examined the distribution of learnt clusters across embedding dimensionalities  $d$  for both MR and RR models, as seen in Figure 9. We observe IVF-MR to have less variance than IVF-RR at  $d \in \{8, 16\}$ , and slightly higher variance for  $d \geq 32$ , while IVF-MR outperforms IVF-RR in top-1 across all  $d$  (Figure 7a). This indicates that although MR learns clusters that are less uniformly distributed than RR at high  $d$ , the quality of learnt clustering is superior to RR across all  $d$ . Note that a uniform distribution is  $N/k$  data points per cluster, i.e.  $\sim 1250$  for ImageNet-1K with  $k = 1024$ . We quantitatively evaluate the proximity of the MR and RR clustering distributions with Total Variation

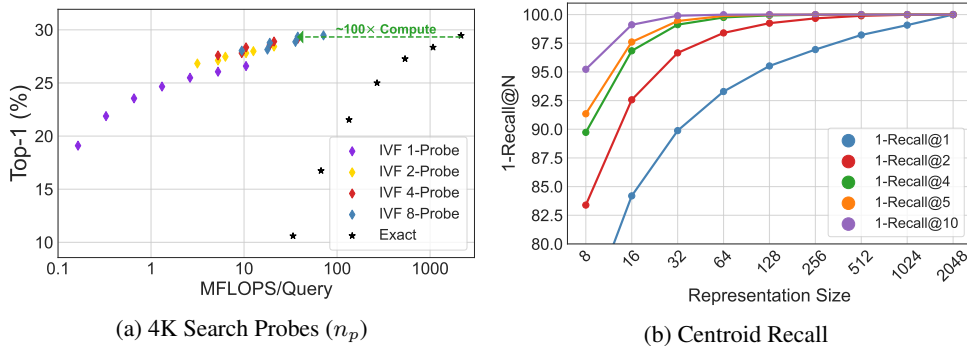


Figure 8: Ablations on IVF-MR Clustering: a) Analysis of accuracy-compute tradeoff with increasing IVF-MR search probes  $n_p$  on ImageNet-4K compared to Exact-MR and b) k-Recall@N on ImageNet-1K cluster centroids across representation sizes  $d$ . Cluster centroids retrieved with highest embedding dim  $d = 2048$  were considered ground-truth centroids.

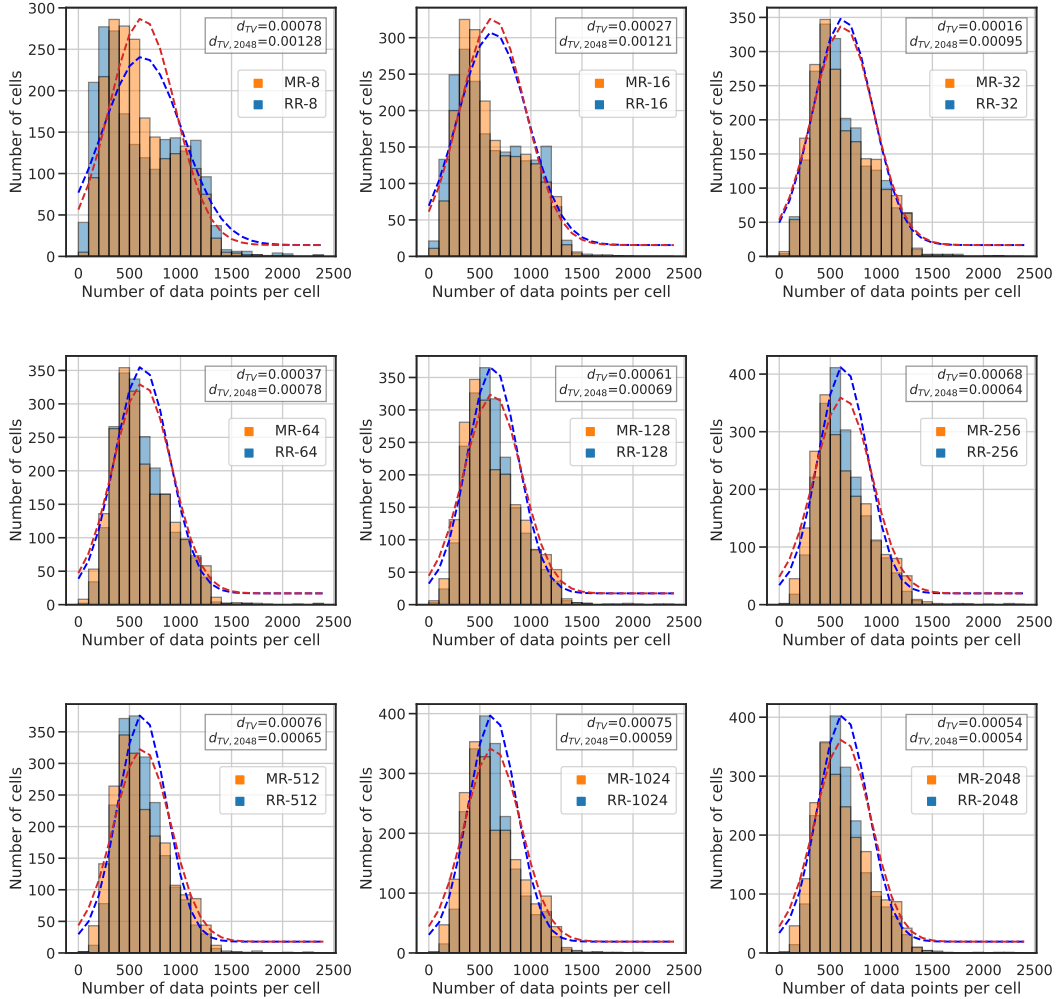


Figure 9: Clustering distributions for IVF-MR and IVF-RR across embedding dimensionality  $d$  on ImageNet-1K. An IVF-MR and IVF-RR clustered with  $d = 16$  embeddings is denoted by MR-16 and RR-16 respectively.

Distance [32], which is defined over two discrete probability distributions  $p, q$  over  $[n]$  as follows:

$$d_{TV}(p, q) = \frac{1}{2} \sum_{i \in [n]} |p_i - q_i|$$

647 We also compute  $d_{TV,2048}(\text{MR-}d) = d_{TV}(\text{MR-}d, \text{RR-}2048)$ , which evaluates the total variation distance of a given low- $d$  MR from high- $d$  RR-2048. We observe a monotonically decreasing  $d_{TV,2048}$   
648 with increasing  $d$ , which demonstrates that MR clustering distributions get closer to RR-2048 as we  
649 increase the embedding dimensionality  $d$ . We observe in Figure 9 that  $d_{TV}(\text{MR-}d, \text{RR-}d) \sim 7e - 4$   
650 for  $d \in \{8, 256, \dots, 2048\}$  and  $\sim 3e - 4$  for  $d \in \{16, 32, 64\}$ . These findings agree with the top-1  
651 improvement of MR over RR as shown in Figure 7a, where there are smaller improvements for  
652  $d \in \{16, 32, 64\}$  (smaller  $d_{TV}$ ) and larger improvements for  $d \in \{8, 256, \dots, 2048\}$  (larger  $d_{TV}$ ).  
653 These results demonstrate a correlation between top-1 performance of IVF-MR and the quality of  
654 clusters learnt with MR.  
655

#### 656 E.4 Inference Compute Cost

We evaluate inference compute costs for IVF in MegaFLOPS per query (MFLOPS/query) as shown in Figures 2, 8a, and 10 as follows:

$$C = d_s k + \frac{n_p d_s N_D}{k}$$

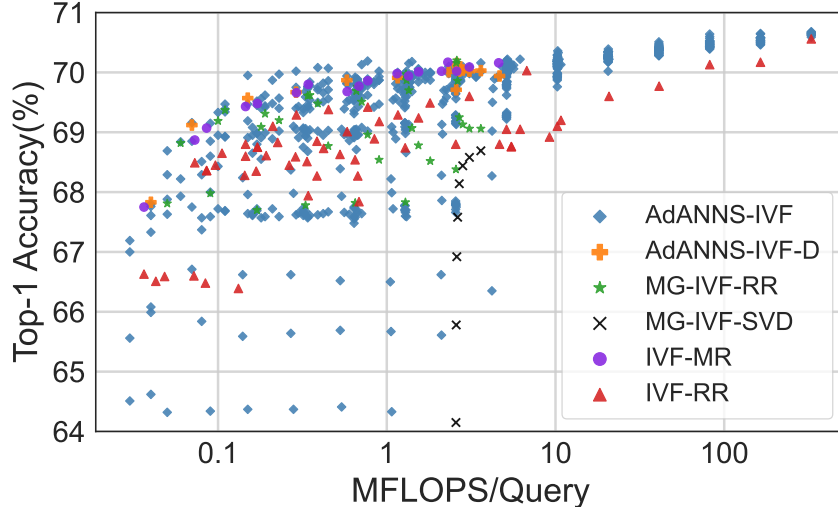


Figure 10: Top-1 accuracy vs. compute cost per query of AdANNS-IVF compared to IVF-MR, IVF-RR and MG-IVF-RR baselines on ImageNet-1K.

657 where  $d_c$  is the **cluster** construction embedding dimensionality,  $d_s$  is the embedding dim used for  
 658 linear **scan** within each **probed** cluster, which is controlled by # of search probes  $n_p$ . Finally,  $k$   
 659 is the number of clusters  $|C_i|$  indexed over database of size  $N_D$ . The default setting in this work,  
 660 unless otherwise stated, is  $n_p = 1$ ,  $k = 1024$ ,  $N_D = 1281167$  (ImageNet-1K trainset). Vanilla IVF  
 661 supports only  $d_c = d_s$ , while AdANNS-IVF provides flexibility via decoupling clustering and search  
 662 (Section 4). AdANNS-IVF-D is a special case of AdANNS-IVF with the flexibility restricted to  
 663 inference, i.e.,  $d_c$  is a fixed high-dimensional MR.

## 664 F AdANNS-IVF

665 Our proposed adaptive variant of IVF, AdANNS-IVF, decouples the clustering, with  $d_c$  dimensions,  
 666 and the linear scan within each cluster, with  $d_s$  dimensions – setting  $d_c = d_s$  results in non-  
 667 adaptive vanilla IVF. This helps in the smooth search of design space for the optimal accuracy-  
 668 compute trade-off. A naive instantiation yet strong baseline would be to use explicitly trained  
 669  $d_c$  and  $d_s$  dimensional rigid representations (called MG-IVF-RR, for multi-granular IVF with  
 670 rigid representations). We also examine the setting of adaptively choosing low-dimensional MR  
 671 to linear scan the shortlisted clusters built with high-dimensional MR, i.e. AdANNS-IVF-D, as  
 672 seen in Table 5. As seen in Figure 10, AdANNS-IVF provides pareto-optimal accuracy-compute  
 673 tradeoff across inference compute. This figure is a more exhaustive indication of AdANNS-IVF  
 674 behavior compared to baselines than Figures 1a and 2. AdANNS-IVF is evaluated for all possible  
 675 tuples of  $d_c, d_s, k = |C| \in \{8, 16, \dots, 2048\}$ . AdANNS-IVF-D is evaluated for a pre-built IVF  
 676 index with  $d_c = 2048$  and  $d_s \in \{8, \dots, 2048\}$ . MG-IVF-RR configurations are evaluated for  
 677  $d_c \in \{8, \dots, d_s\}$ ,  $d_s \in \{32, \dots, 2048\}$  and  $k = 1024$  clusters. A study over additional  $k$  values  
 678 is omitted due to high compute cost. Finally, IVF-MR and IVF-RR configurations are evaluated  
 679 for  $d_c = d_s \in \{8, 16, \dots, 2048\}$  and  $k \in \{256, \dots, 8192\}$ . Note that for a fair comparison, we use  
 680  $n_p = 1$  across all configurations. We discuss the inference compute for these settings in Appendix E.4.

## 681 G AdANNS-DiskANN

682 DiskANN is a state-of-the-art graph-based ANNS index capable of serving queries from both RAM  
 683 and SSD. DiskANN builds a greedy best-first graph with OPQ distance computation, with compressed  
 684 vectors stored in memory. The index and full-precision vectors are stored on the SSD. During search,  
 685 when a query’s neighbor shortlist is fetched from the SSD, its full-precision vector is also fetched  
 686 in a single disk read. This enables efficient and fast distance computation with PQ on a large initial  
 687 shortlist of candidate nearest neighbors in RAM followed by a high-precision re-ranking with full-

Table 6: Wall clock search latency ( $\mu s$ ) of AdANNS-DiskANN across graph construction dimensionality  $d \in \{8, 16, \dots, 2048\}$  and compute budget in terms of OPQ budget  $M \in \{8, 16, 32, 48, 64\}$ . Search latency is fairly consistent across fixed embedding dimensionality  $D$ .

| $d$  | $M=8$ | $M=16$ | $M=32$ | $M=48$ | $M=64$ |
|------|-------|--------|--------|--------|--------|
| 8    | 495   | -      | -      | -      | -      |
| 16   | 555   | 571    | -      | -      | -      |
| 32   | 669   | 655    | 653    | -      | -      |
| 64   | 864   | 855    | 843    | 844    | 848    |
| 128  | 1182  | 1311   | 1156   | 1161   | 2011   |
| 256  | 1923  | 1779   | 1744   | 2849   | 1818   |
| 512  | 2802  | 3272   | 3423   | 2780   | 3171   |
| 1024 | 5127  | 5456   | 5724   | 4683   | 5087   |
| 2048 | 9907  | 9833   | 10205  | 10183  | 9329   |

688 precision vectors fetched from the SSD on a much smaller shortlist. The experiments carried out in  
 689 this work primarily utilize a DiskANN graph index built in-memory<sup>2</sup> with OPQ distance computation.

690 As with IVF, DiskANN is also well  
 691 suited to the flexibility provided by  
 692 AdANNS as we demonstrate on both  
 693 ImageNet and NQ that the optimal PQ  
 694 codebook for a given compute budget  
 695 is learnt with a smaller embedding di-  
 696 mensionality  $d$  (see Figures 6c and  
 697 6e). We demonstrate the capability  
 698 of AdANNS-DiskANN with a com-  
 699 pute budget of  $M \in \{32, 64\}$  in Ta-  
 700 ble 1. We tabulate the search time  
 701 latency of AdANNS-DiskANN in mi-  
 702 croseconds ( $\mu s$ ) in Table 6, which  
 703 grows linearly with graph construc-  
 704 tion dimensionality  $d$ . We also exam-  
 705 ine DiskANN-MR with SSD graph in-  
 706 dices across OPQ budgets for distance  
 707 computation  $M_{dc} \in \{32, 48, 64\}$ , as  
 708 seen in Figure 11. With SSD indices,  
 709 we store PQ-compressed vectors on  
 710 disk with  $M_{disk} = M_{dc}$ , which es-  
 711 sentially disables DiskANN’s implicit  
 712 high-precision re-ranking. We ob-  
 713 serve similar trends to other compos-  
 714 ite ANNS indices on ImageNet, where the *optimal* dim for fixed OPQ budget is not the highest dim  
 715 ( $d = 1024$  with fp32 embeddings is current highest dim supported by DiskANN which stores vectors  
 716 in 4KB sectors on disk). This provides further motivation for AdANNS-DiskANN, which leverages  
 717 MRs to provide flexible access to the optimal dim for quantization and thus enables similar Top-1  
 718 accuracy to Rigid DiskANN for up to 1/4 the cost (Figure 6c).

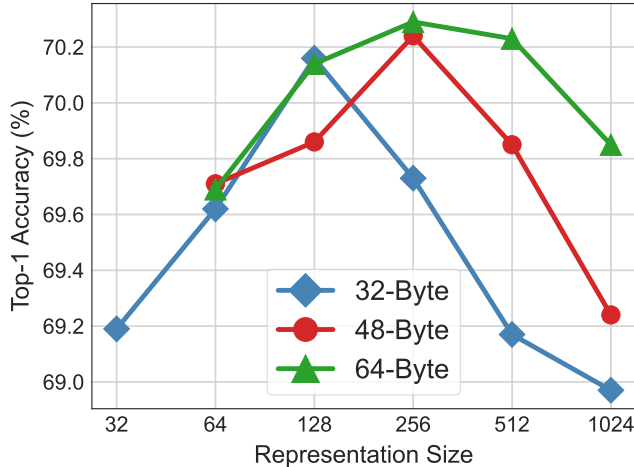


Figure 11: DiskANN-MR with SSD indices for compute budgets  $M_{disk} = M_{dc} \in \{32, 48, 64\}$  across graph construction and OPQ dimensionalities  $d \in \{32, \dots, 1024\}$ . Note that this does not use any re-ranking after obtaining OPQ based shortlist.

## 719 H AdANNS on Natural Questions

720 In addition to image retrieval on ImageNet, we also experiment with dense passage retrieval (DPR) on  
 721 Natural Questions. As shown in Figure 6, MR representations are 1 – 10% more accurate than their  
 722 RR counterparts across PQ compute budgets with Exact Search + OPQ on NQ. We also demonstrate  
 723 that IVF-MR is 1 – 2.5% better than IVF-RR for Precision@ $k$ ,  $k \in \{1, 5, 20, 100, 200\}$ . Note that  
 724 on NQ, IVF loses  $\sim 10\%$  accuracy compared to exact search, even with the RR-768 baseline. We  
 725 hypothesize the weak performance of IVF owing to poor clusterability of the BERT-Base embeddings

<sup>2</sup><https://github.com/microsoft/DiskANN>

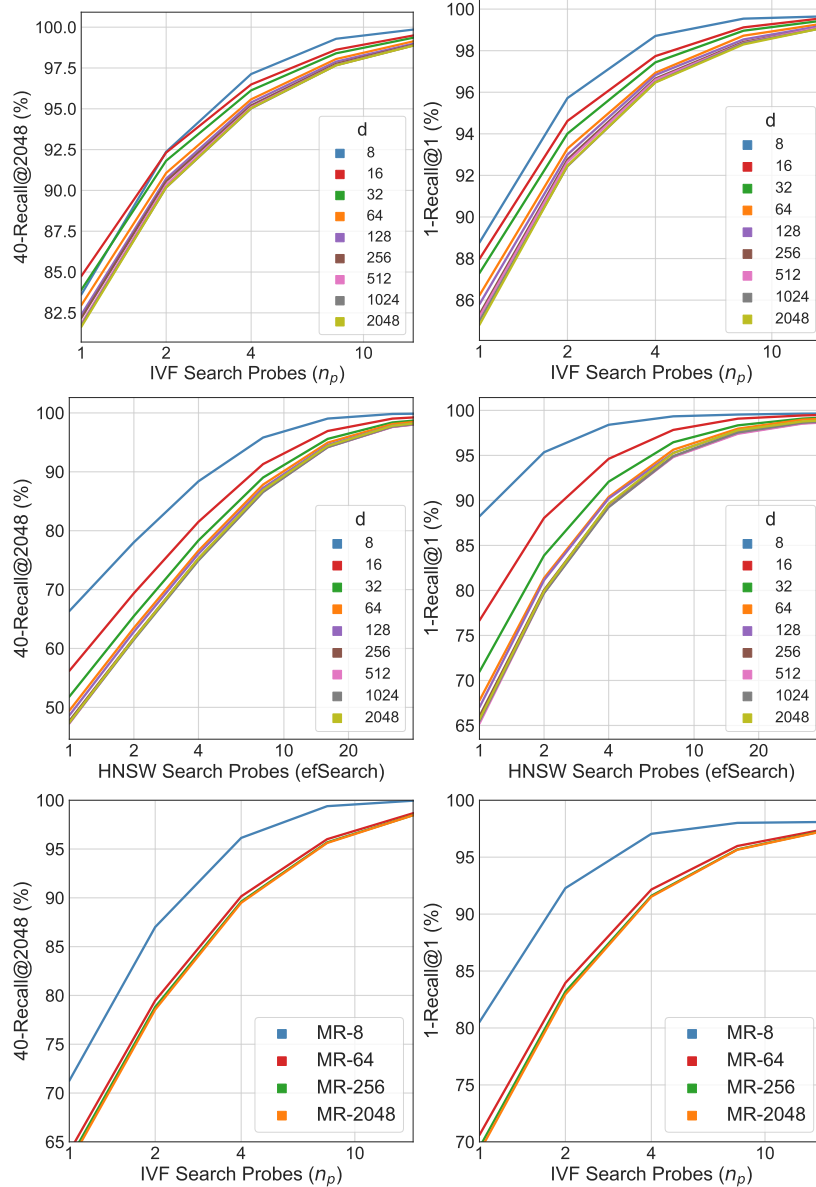


Figure 12: k-Recall@N of  $d$ -dimensional MR for IVF and HNSW with increasing search probes  $n_p$  on ImageNet-1K and ImageNet-4K. On ImageNet-4K, we restrict our study to IVF-MR with  $d \in \{8, 64, 256, 2048\}$ . Other embedding dimensionalities, HNSW-MR and RR baselines are omitted due to high compute cost. We observe that trends from ImageNet-1K with increasing  $d$  and  $n_p$  extend to ImageNet-4K, which is  $4\times$  larger.

726 fine-tuned on the NQ dataset. A more thorough exploration of AdANNS-IVF on NQ is an immediate  
 727 future work and is in progress.

## 728 I Ablations

### 729 I.1 Recall Score Analysis

730 In this section we also examine the variation of k-Recall@N with by probing a larger search space  
 731 with IVF and HNSW indices. For IVF, search probes represent the number of clusters shortlisted for  
 732 linear scan during inference. For HNSW, search quality is controlled by the *efSearch* parameter [37],  
 733 which represents the closest neighbors to query  $q$  at level  $l_c$  of the graph and is analogous to number

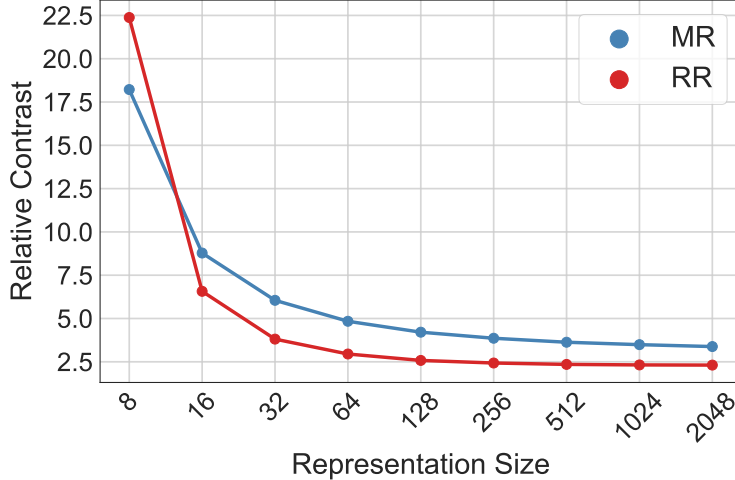


Figure 13: Relative contrast of varying capacity MRs and RRs on ImageNet-1K corroborating the findings of He et al. [17].

734 of search probes in IVF. As seen in Figure 12, general trends show a) an intuitive increase in recall  
 735 with increasing search probes  $n_p$ ) for fixed search probes, b) a decrease in recall with increasing  
 736 search dimensionality  $d$  c) similar trends in ImageNet-1K and  $4\times$  larger ImageNet-4K.

## 737 I.2 Relative Contrast

738 We utilize Relative Contrast [17] to capture the difficulty of nearest neighbors search with IVF-MR  
 739 compared to IVF-RR. For a given database  $X = \{x_i \in \mathbb{R}^d, i = 1, \dots, N_D\}$ , a query  $q \in \mathbb{R}^d$ , and a  
 740 distance metric  $D(\cdot, \cdot)$  we compute relative contrast  $C_r$  as a measure of the difficulty in finding the  
 741 1-nearest neighbor (1-NN) for a query  $q$  in database  $X$  as follows:

- 742 1. Compute  $D_{min}^q = \min_{i=1\dots n} D(q, x_i)$ , i.e. the distance of query  $q$  to its nearest neighbor  $x_{nn}^q \in X$
- 743 2. Compute  $D_{mean}^q = E_x[D(q, x)]$  as the average distance of query  $q$  from all database points  
 744  $x \in X$
- 745 3. Relative Contrast of a given query  $C_r^q = \frac{D_{mean}^q}{D_{min}^q}$ , which is a measure of how *separable* the  
 746 query’s nearest neighbor  $x_{nn}^q$  is from an average point in the database  $x$
4. Compute an expectation over all queries for Relative Contrast over the entire database as

$$C_r = \frac{E_q[D_{mean}^q]}{E_q[D_{min}^q]}$$

747 It is evident that  $C_r$  captures the difficulty of Nearest Neighbor Search in database  $X$ , as a  $C_r \sim 1$   
 748 indicates that for an average query, its nearest neighbor is almost equidistant from a random point  
 749 in the database. As demonstrated in Figure 13, MRs have higher  $R_c$  than RR Embeddings for an Exact  
 750 Search on ImageNet-1K for all  $d \geq 16$ . This result implies that a portion of MR’s improvement  
 751 over RR for 1-NN retrieval across all embedding dimensionalities  $d$  [30] is due to a higher average  
 752 separability of the MR 1-NN from a random database point.

## 753 I.3 Generality across Encoders

754 We perform an ablation over the representation function  $\phi : X \rightarrow \mathbb{R}^d$  learnt via a backbone neural  
 755 network (primarily ResNet50 in this work), as detailed in Section 3. We also train MRL models [30]  
 756  $\phi^{MR(d)}$  on ResNet18/34/101 [18] that are as accurate as their independently trained RR baseline  
 757 models  $\phi^{RR(d)}$ , where  $d$  is the default max representation size of each architecture. We also train  
 758 MRL with a ConvNeXt-Tiny backbone with  $[d] = \{48, 96, 192, 384, 786\}$ . MR-768 has a top-1  
 759 accuracy of 79.45% compared to independently trained publicly available RR-768 baseline with  
 760 top-1 accuracy 82.1% (Code and RR model available on the official repo<sup>3</sup>). We note that this training

<sup>3</sup><https://github.com/facebookresearch/ConvNeXt>

761 had no hyperparameter tuning whatsoever, and this gap can be closed with additional model training  
 762 effort. We then compare clustering the MRs via IVF-MR with  $k = 2048, n_p = 1$  on ImageNet-1K to  
 763 Exact-MR, which is shown in Figure 14. IVF-MR shows similar trends across backbones compared  
 764 to Exact-MR, i.e. a maximum top-1 accuracy drop of  $\sim 1.6\%$  for a single search probe. This suggests  
 765 the clustering capabilities of MR extend beyond an inductive bias of  $\phi^{MR(d)} \in \text{ResNet50}$ , though  
 we leave a detailed exploration for future work.

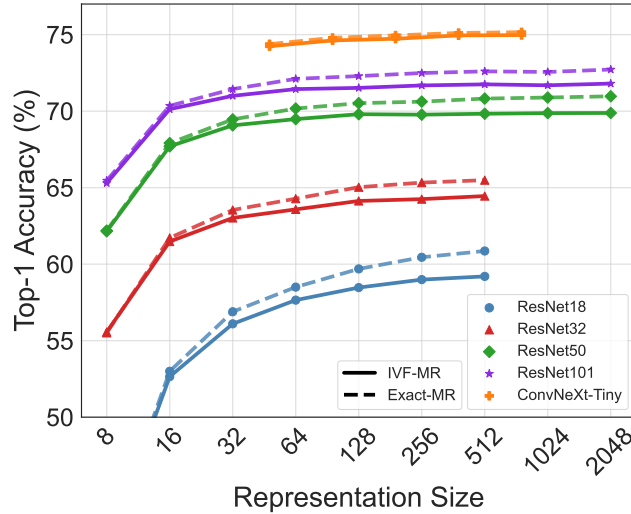


Figure 14: Top-1 Accuracy variation of IVF-MR on ImageNet-1K with different embedding representation function  $\phi^{MR(d)}$  (see Section 3), where  $\phi \in \{\text{ResNet18/34/101}, \text{ConvNeXt-Tiny}\}$ . We observe similar trends between IVF-MR and Exact-MR on ResNet18/34/101 when compared to ResNet50 (Figure 7a) which is the default in all experiments in this work.

766