Unsupervised Multi-Modal Representation Learning for High Quality Retrieval of Similar Products at E-commerce Scale

Kushal Kumar
kushlku@amazon.com
Amazon.com

Tarik Arici
aricit@amazon.com
Amazon.com

Tal Neiman
taneiman@amazon.com
Amazon.com

Jinyu Yang
viyjy@amazon.com
Amazon.com

Shioulin Sam
shioulin@amazon.com
Amazon.com

Yi Xu
yxaamzn@amazon.com
Amazon.com

Hakan Ferhatosmanoglu
hakanf@amazon.co.uk
Amazon Web Services

Ismail Tutar
ismailt@amazon.com
Amazon.com

ABSTRACT

Identifying similar products in e-commerce is useful in discovering relationships between products, making recommendations, and increasing diversity in search results. Product representation learning is the first step to define a generalized product similarity metric for search. The second step is to extend similarity search to a large scale (*e.g.*, e-commerce catalog scale) without sacrificing quality. In this work, we present a solution that interweaves both steps, *i.e.*, learn representations suited to high quality retrieval using contrastive learning (CL) and retrieve similar items from a large search space using approximate nearest neighbor search (ANNS) to trade-off quality for speed. We propose a CL training strategy for learning uni-modal encoders suited to multi-modal similarity search for e-commerce. We study ANNS retrieval by generating Pareto Frontiers (PFs) without requiring labels. Our CL training strategy doubles *retrieval@1* metric across categories (*e.g.*, from 36% to 88% in category C). We also demonstrate that ANNS engine optimization using PFs help select configurations appropriately (*e.g.*, we achieve 6.8× search speed with just 2% drop from the maximum retrieval accuracy in medium size datasets).

CCS CONCEPTS

• **Theory of computation** → **Unsupervised learning and clustering**; • **Information systems** → **Nearest-neighbor search**; *Top-k retrieval in databases.*

KEYWORDS

Deep Metric Learning, Unsupervised Learning, Approximate Nearest Neighbor Search, Amazon Opensearch, Clustering

ACM Reference Format:
Kushal Kumar, Tarik Arici, Tal Neiman, Jinyu Yang, Shioulin Sam, Yi Xu, Hakan Ferhatosmanoglu, and Ismail Tutar. 2023. Unsupervised Multi-Modal Representation Learning for High Quality Retrieval of Similar Products at E-commerce Scale. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3583780.3615504

CIKM '23, October 21–25, 2023, Birmingham, UK

ACM ISBN 979-8-4007-0124-5/23/10.
https://doi.org/10.1145/3583780.3615504

1 INTRODUCTION

Identifying similar items in large e-commerce datasets can enable a variety of applications. These include search result post-processing to increase diversity, building relationships between similar items for easy product navigation following a search query, fine-grained multi-modal retrieval and downstream product classification [9]. With advancement in deep learning, we can learn dense product vectors that perform well on product recommendation and classification tasks [4, 13, 31]. Dense vectors are therefore, also a de facto choice for retrieval. For finding similar items in large spaces (order of 10MM or more) we need techniques beyond exact nearest neighbor search (NNS) to meet efficiency and cost constraints. In this paper, we study retrieval in large multi-modal e-commerce datasets by i) learning encoders with no supervision to generate powerful multi-modal product vectors suited to similarity search, and ii) efficiently running NNS at scale using OpenSearch [27].

Contrastive learning (CL), as one type of self-supervised learning (SSL), enables learning encoder models from vision and language signals [6, 32]. In CL, a training instance is created from a positive pair and a number of negative pairs. CL methods aim to learn representations by simultaneously increasing similarities of the positive pair and decreasing similarities of negative pairs in a training instance. Multi-modal datasets naturally present positive pairings between modalities of an example while uni-modal datasets typically require labeling or data augmentation (*e.g.*, creating new views to be used as positive pairs [29]). Easy positives (matching pairs with close representations) and easy negatives (dissimilar pairs with far away representations) reduce CL efficiency (see Section 3.1 for a detailed discussion) by scaling down the gradients via a negative-positive coupling multiplier [12, 34]. Some techniques utilize pairs generated from a larger mini-batch to increase hard negatives [6], others store instances from previous iterations in a queue [14] and thereby increase the number of hard negatives. In this work, we generate hard negatives without any supervision using product

types (or categories). To boost CL efficiency, we increase the likelihood of in-batch hard negatives by sampling negatives from a target category.

Exact NNS to find top-k nearest neighbors for every item has quadratic complexity and becomes infeasible for large datasets. We study approximate nearest neighbor search (ANNS) on large e-commerce datasets and incorporate our solutions within Amazon OpenSearch [27]. We study the performance of two ANNS algorithms, IVF [18] and HNSW [23], on cross-modal retrieval as a proxy to recall. IVF uses codebook learning to partition the dataset and quantize vectors for fast distance computation [18]. HNSW uses a graph-based index to map vectors into a multi-layer graph allowing logarithmic scalability [23]. Performance of these ANNS algorithms depends largely on the chosen parameters which must be tuned for the dataset characteristics such as size and vector dimensionality [2]. Using grid search on ANNS algorithm parameters, we generate Pareto frontiers (PFs) [21] for speed and recall using cross-modal retrieval performance as an approximation to true multi-modal recall. This enables us to discover parameter configurations that best exploit the trade-off between speed and recall while satisfying practical constraints such as search-time and throughput.

Finally, we generate multi-modal item vectors from image and text vectors and insert them to an Amazon OpenSearch index configured from ANNS algorithm's PF. Our main contributions are:

- We show that sampling from a target category introduces hard negatives that improves CL performance substantially without requiring additional supervision or auxiliary losses.
- We exploit the trade-off between speed and recall to find the best operating point for the use case using cross-modal retrieval as a proxy for multi-modal recall.
- We provide recommendations to scale ANNS algorithm across multiple datasets. We share useful insights on optimizing OpenSearch kNN indices and maintaining clusters at low cost.

2 RELATED WORK

2.1 Multi-Modal Representation Learning

Contrastive learning (CL) uses InfoNCE (discussed in detail in Section 3.1) to predict the positive sample given K random samples, which includes $K-1$ negative samples. InfoNCE loss is a good fit for learning from multi-modal datasets [6, 11, 25, 33]. In e-commerce, CL is shown to outperform in tasks such as product categorization [5], product matching [1, 24] and classification tasks [9]. CLIP [25] is a popular choice for CL-based multi-modal representations [15, 22]. However, directly applying CLIP training perform poorly on e-commerce datasets due to stark differences between natural and product images [17]. [17] is a very recent work that solves this problem in e-commerce by using pretext tasks. We solve the same problem by still using InfoNCE loss, without the need for additional labels or auxiliary losses; instead we introduce hard negatives using product categories.

2.2 Cross-Modal Retrieval

Cross-modal retrieval is the task of retrieving the same item in other modality using query from a different modality. In e-commerce, there has been studies on cross-modal retrieval and ranking using CLIP-like [25] contrastive pre-training [9, 15, 22]. However, most such studies are on a relatively small sampled datasets where exact nearest neighbor search is feasible. E-commerce datasets are large (of the order of 10MM and more) and oftentimes noisy. Hence, it is critical to study approximate nearest neighbor search (ANNS) algorithms on such datasets to establish ANNS performance.

3 MULTI-MODAL REPRESENTATION LEARNING AND RETRIEVAL

In this section, we first present a CL efficiency perspective of why off-the-shelf CLIP ([25]) training perform poorly on large datasets. We then present our unsupervised algorithm that solves this problem in purview of retrieval in e-commerce and discuss Amazon OpenSearch for retrieval at scale.

3.1 Contrastive Learning Efficiency

Loss function for CL was introduced in [30] and is called InfoNCE. Let $sim(\mathbf{x}, \mathbf{y})$ denote similarity between $\mathbf{x}$ and $\mathbf{y}$ representation vectors, and can be defined as the dot product between normalized $\mathbf{x}$ and $\mathbf{y}$. Let $f(\mathbf{x}, \mathbf{y}) = \exp(\tau^{-1} sim(\mathbf{x}, \mathbf{y}))$ be the score of pair $(\mathbf{x}, \mathbf{y})$, where τ is temperature parameter to control the strength of penalties on hard negative samples. InfoNCE is given as below

$$L_{\text{InfoNCE}}(X; Y|f) = \mathbb{E}\left[\log \frac{f(\mathbf{x_1}, \mathbf{y})}{\sum_{x_i \in \tilde{X}} f(\mathbf{x_i}, \mathbf{y})}\right], \tag{1}$$

where $\tilde{X} = \{\mathbf{x_2}, \ldots \mathbf{x_K}\}$ are negative samples drawn i.i.d. according to $p(\mathbf{x})$ and $(\mathbf{x_1}, \mathbf{y})$ is a positive pair. Let $\mathbf{X} = (\mathbf{x_1}, \ldots \mathbf{x_K})$ be a matrix, then gradient of InfoNCE with respect to $\mathbf{y}$ is given by

$$\nabla L_{\text{InfoNCE}\mathbf{y}} = -\tau^{-1}\mathbf{x_1} + \tau^{-1}\mathbf{X}\text{softmax}(-\tau^{-1}\mathbf{X}^T\mathbf{y}) \tag{2}$$

Let $(p_1, \ldots, p_K) = \text{softmax}(-\tau^{-1}\mathbf{X}^T\mathbf{y})$. Then p_1 is softmax normalized weight corresponding to the positive example. Eq. (2) can be rewritten as

$$\nabla L_{\text{InfoNCE}\mathbf{y}} = -\tau^{-1}(1 - p_1)\left(\mathbf{x_1} + \tilde{\mathbf{X}}\text{softmax}(-\tau^{-1}\tilde{\mathbf{X}}^T\mathbf{y})\right), \tag{3}$$

where $\tilde{\mathbf{X}}$ is equal to $\mathbf{X}$ except its first column corresponding to the positive example (see [34] and [12] for derivations). We can write out p_1 as follows:

$$p_1 = \frac{f(\mathbf{x_1}, \mathbf{y})}{\sum f(\mathbf{x_i}, \mathbf{y})} = \frac{f(\mathbf{x_1}, \mathbf{y})}{f(\mathbf{x_1}, \mathbf{y}) + \sum_{x_i \in \tilde{X}} f(\mathbf{x_i}, \mathbf{y})}, \tag{4}$$

which is the ratio of similarity score of the positive pair to sum of all similarity scores, which includes the positive-pair similarity score, $f(\mathbf{x_1}, \mathbf{y})$, and $K-1$ negative-pair similarity-scores, $\sum_{x_i \in \tilde{X}} f(\mathbf{x_i}, \mathbf{y})$. p_1 becomes closer to one for large $f(\mathbf{x_1}, \mathbf{y})$ (*i.e.*, easy positives), and small $\sum_{x_i \in \tilde{X}} f(\mathbf{x_i}, \mathbf{y})$ (*i.e.*, easy negatives) effectively reducing the gradient. [34] and [12] propose modifying the InfoNCE to eliminate p_1 from the gradient. While $(1 - p_1)$ multiplier can be seen as reducing CL efficiency, it can also be considered as a necessary learning-rate adaptation where easy training instances are weighted down compared to hard instances. Therefore, our remedy to CL efficiency degradation is to reduce p_1 by sampling harder negatives (while maximizing batch size).

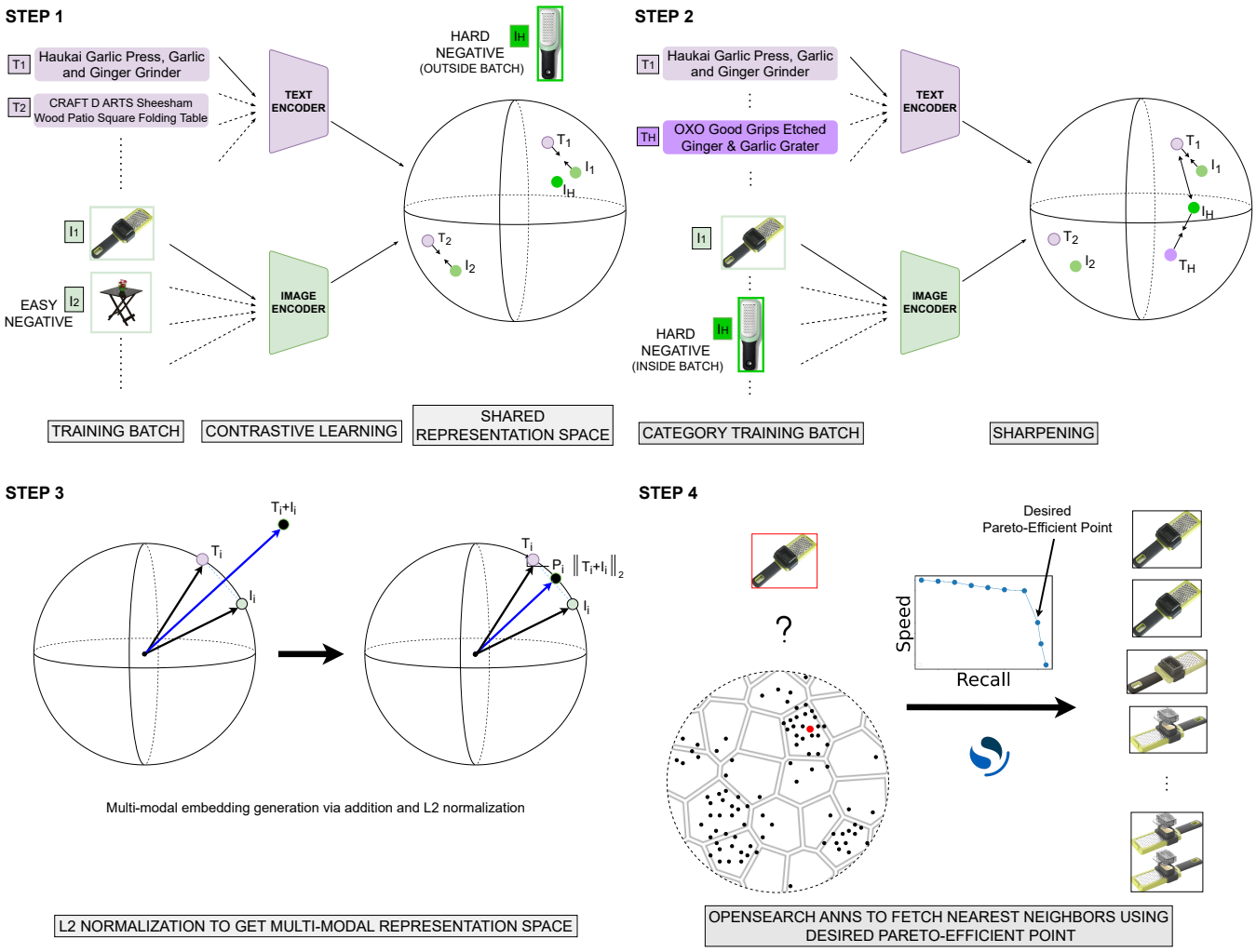


Figure 1: Our unsupervised approach to finding top-k nearest neighbors is divided into 4 steps. Step 1- We train generic image and text encoders from scratch using InfoNCE (*e.g.*, on a large Amazon dataset). Step 2- We "sharpen" representations by introducing harder negatives in the training batch by sampling from a target category. Step 3- We generate multi-modal vectors by adding the uni-modal vectors and L2 normalization. Step 4- We optimize Amazon OpenSearch using Pareto Frontiers to select best operating configuration to find top-k nearest neighbors in the target category.

3.2 Algorithm

We now discuss our unsupervised algorithm to improve CL efficiency for e-commerce dataset suited to retrieval. See Figure 1 for an illustration on our algorithm.

3.2.1 Training Generic Encoders. We start by training encoders for a large subset of Amazon catalog using the InfoNCE loss in Eq. (1) similar to [26]. Image and text encoder outputs are linearly projected to a shared representation space to construct InfoNCE. Given a batch of size K, we produce a $K \times K$ matrix $\mathbf{M}$ where each element m_{pq} is the cosine similarity between p-th text embedding, $\mathbf{t_p}$, and q-th image embedding, $\mathbf{i_q}$, scaled by a temperature parameter. Cross-entropy losses constructed from both row-wise and column-wise softmax normalizations are summed to guide both encoders equally (see Step 1 in Figure 1).

3.2.2 Sharpening Encoders. An interpretation of InfoNCE is that it optimizes retrieval accuracy for top prediction among K items in the batch. Hence, to train encoders suited to retrieval we do not augment auxiliary losses to improve CL efficiency unlike [17]. Instead, we introduce hard negative pairs during training as follows. Amazon e-commerce dataset is organized into hierarchical categories. We define a target category as a set of nodes in the product type hierarchy where we want to find top-k neighbors. We continue training encoders for each target category by generating negative pairs from the category. This increases the probability of hard negatives by $|G|/|T|$ where G is the generic dataset and T is the target category in G, where $|T| \leq |G|$. Replacing a weak negative sample with a hard negative results in higher InfoNCE loss, and minimizing it will "sharpen" the cosine distribution (see

Step 2 in Figure 1). We suggest to study the hierarchy to define target categories [1].

3.2.3 Multi-Modal Vector Generation. Using sharpened image and text encoders, an item can be vectorized to a text vector, $\mathbf{t}$, and an image vector, $\mathbf{i}$. Given that both vectors share the same representation space, we define the multi-modal item embedding vector, $\mathbf{m}$, as $\mathbf{m} = (\mathbf{t} + \mathbf{i})/||\mathbf{t} + \mathbf{i}||$, which is the L2 normalized sum of uni-modal vectors. The multi-modal vector has equal similarity to either uni-modal vectors: $cos(\mathbf{m}, \mathbf{t}) = cos(\mathbf{m}, \mathbf{i})$ (see Step 3 in Figure 1). Hence, angular retrieval efficacy gets propagated to multi-modal vectors.

3.2.4 Optimized ANNS with OpenSearch. Exact Nearest Neighbor Search (NNS) has quadratic complexity ($|query\_space| \times |search\_space|$), hence is oftentimes infeasible [16, 19]. We circumvent the problem using Approximate Nearest Neighbors Search (ANNS). Given a target category $T = \{\mathbf{x}_i\}_{i=1,2,3...n}$ where each data point $\mathbf{x}_i \in \mathbb{R}^d$ and a query item $\mathbf{q} \in \mathbb{R}^d$, ANNS task is to find approximate k nearest neighbors for each $\mathbf{q}$ in T. ANNS builds an index I on the dataset T, finds a subset T_q of T for each q using I and then finds the best k nearest neighbors of $\mathbf{q}$ in T_q using similarity $sim(\mathbf{p}, \mathbf{q})$, where $\mathbf{p} \in T_q$. We build our ANNS framework on top of kNN plugin of Amazon OpenSearch [27]. We choose OpenSearch because it is a fully managed AWS service that is easy to scale and has zero downtime when re-indexing.

OpenSearch supports two ANNS algorithms: IVF [18] and HNSW [23] that have performed well [2]. However, since representation learning is decoupled with retrieval, our end-to-end method is generalizable across ANNS algorithms. There is no algorithm that outperforms others on all datasets and recall points. We need to find the best performing algorithm for our recall use case. To study this behavior, we run grid search on ANNS parameters to generate Pareto Frontiers (PFs) of speed and recall[2]. Grid search is slow and cannot be performed for every target category. Hence, we divide our target categories into brackets and show that datasets in same bracket perform similarly.

4 EXPERIMENTS

4.1 Datasets

We train our generic CL encoders on large Amazon dataset of size 2B spanning across several product categories and countries. We use product title and the main image as the two input modalities for CL. For sharpened CL encoders, we randomly select 24 target categories - Category A, Category B and Category C across eight countries filtering out low search impression products. Test queries are randomly sampled and used for evaluating both CL models and ANNS algorithms on cross-modal retrieval. We also experiment on a 10MM sample of uni-modal Deep1b [3] by [2] and $100M$ sample of multi-modal Yandex Text-to-Image [28]. See Table 1 for statistics on datasets used for different experiments.

[1]Large or very small categories may result in poor performance due to easy or very hard negatives respectively

[2]PF of speed & recall is a set of (speed, recall) points $s.t.$ no other point improves both speed & recall.

Table 1: Dataset statistics for CL and ANNS experiments. Amazon datasets are aggregated across countries. Deep1b and Yandex Text-to-Image provide test queries unlike Amazon datasets.

Dataset Name	Items (MM)	Vector Dim	Train	Test	Metric
Deep1b	9.9	96	–	$10k$	$recall@k$
Yandex Text-to-Image	10 & 100	200	–	$100k$	$recall@k$
Category A	547	512	$54MM$	$10k$	$retrieval@k$
Category B	280	512	$39MM$	$10k$	$retrieval@k$
Category C	262	512	$43MM$	$10k$	$retrieval@k$

4.2 Implementation Details

All of our experiments are performed on NVIDIA A100 GPUs with PyTorch framework. The image encoder is a ViT-B/16 [10] with 12 layers and 85.8M parameters. The text encoder is implemented by a 12-layer BERT model [8] with maximum sequence length of 77. Our model is pre-trained on the 2B Amazon dataset for 540K steps with batch size of 65,536. We use AdamW optimizer [20] with a weight decay of 0.2, beta=(0.9, 0.98), and eps=$1e-6$. The learning rate is initialized as 0 and is warmed up to $5e-4$ after 2,000 training steps. We then decrease it by the cosine decay strategy to 0. For data augmentation, a 224×224-pixel crop is taken from a randomly resized image. We use xlm-roberta-large [7] as our tokenizer, which is trained on 2.5TB of filtered CommonCrawl data covering 100 languages. This pre-trained model is used as the initialization of the sharpening stage on target categories. Sharpening follows the same setting except that (i) batch size is 16,384, (ii) total training step is 400K, and (iii) learning rate is initialized as $5e-4$ followed by the cosine decay.

4.3 Evaluation Metrics

We evaluate CL models on both image-to-text (I2T) retrieval and text-to-image (T2I) retrieval metrics ($retrieval@k$), defined as follows: $retrieval@k = \sum \mathbb{1}[x_i \in \{retrieved\_neighbors(y_i, k)\}] \; / \; |Y|$, where Y is the set of all queries, x_i corresponds to y_i from other modality, and $retrieved\_neighbors(y_i, k)$ are the top k neighbors of y_i from other modality. We also evaluate ANNS algorithm on Deep1b [3] and Yandex Text-to-Image [28] datasets using $recall@k = \sum |\{retrieved\_neighbors(y, k)\} \cap \{true\_neighbors(y)\}| \; / \; \sum |true\_neighbors(y)|$. Note that we may use retrieval and recall interchangeably for ease of presentation.

4.4 Effect of Sharpening

We compare sharpened CL models with generic CL model on $retrieval@k$. $retrieval@k$ is calculated using exact nearest neighbors on the test datasets of size $10k$. Category performance is macro-averaged across countries in Table 2. We first observe that T2I and I2T are comparable. In Category A, generic CL model achieves $retrieval@100$ of 0.935 but drops sharply with $retrieval@1$ of 0.468. This indicates that the generic representations struggle to distinguish similar products. Sharpened CL model improves significantly over the generic CL with $retrieval@1$ of 0.862. Similar observations can be drawn from other target categories.

We also demonstrate the effect of sharpening by visualizing cosine similarities with respect to product neighbors. See Figure

(a) Product Neighbors (b) Query Product (c) Product Neighbors

Figure 2: Visualization of cosines with respect to product neighbors for a recliner chair query. Note that only product images are shown for ease of illustration and visually similar products may differ in their titles.

Table 2: Retrieval accuracies (R@k) of CL models. We report both I2T and T2T accuracies on target categories macro-averaged over countries. Sharpening improves retrieval accuracies consistently.

CL Model	Target Category	Text to Image			Image to Text		
		R@100	R@10	R@1	R@100	R@10	R@1
Generic	Category A	0.935	0.784	0.468	0.935	0.762	0.437
Sharpened		0.993	0.971	0.862	0.993	0.969	0.857
Generic	Category B	0.962	0.811	0.477	0.959	0.796	0.465
Sharpened		0.999	0.996	0.924	0.999	0.995	0.927
Generic	Category C	0.919	0.709	0.356	0.922	0.699	0.356
Sharpened		0.998	0.989	0.877	0.996	0.982	0.871

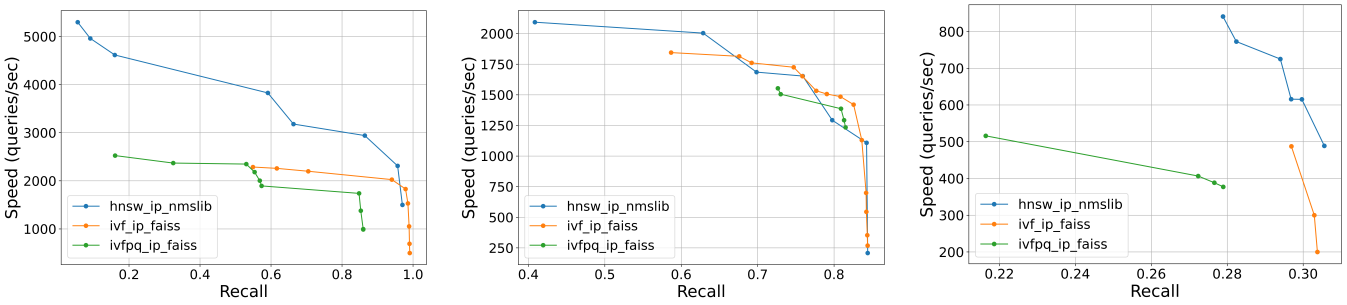
2 for comparison between generic and sharpened CL neighbors for a recliner query product. Generic CL model retrieves good quality neighbors only at a high cosine above 0.9 while quality drops quickly as cosine decreases. This explains the low retrieval accuracy at small k. Sharpened model retrieves more similar neighbors and its performance does not drop as significantly with decreasing cosine. However, this is not just the matter of scaling, as generic model also retrieves low quality neighbors in high cosine buckets (*e.g.* it confuses office chair to be a close neighbor of recliner).

4.5 Optimizing ANNS using PFs

ANNS algorithms implicitly trade-off between search speed and recall. However, the trade-off only holds at the Pareto Frontiers (PFs) [21] of speed and recall. To study the trade-off, we perform grid search on ANNS parameters to generate speed and recall grid and select Pareto-efficient ones [2]. Since grid search is infeasible on all target categories, we divide our datasets into three brackets based on ANNS index size, with small as < 100 GB, medium as $100 - 500$ GB and large as > 500 GB. For small dataset we choose a 10MM sample of Deep1B [3] and run three algorithms - HNSW [23] (hnsw_ip_nmslib), IVF [18] with and without product quantization (ivfpq_ip_faiss & ivf_ip_faiss respectively[3]). See Figure 3 for speed and recall grid and corresponding PFs. We observe that 1) IVF without quantization outperforms HNSW on best recall, 2) HNSW generally achieves higher speed than IVF, and 3) IVF with quantization results in low recall and speed. For medium dataset we choose C8 Category B train dataset that contains 18MM vectors of 512 dimensions. See Figure 3 for speed and recall grid and PFs. We observe that 1) IVF and HNSW are equally good for high recall, 2) IVF is a better choice for slightly less recall and higher speed, and 3) HNSW achieves best speed at low recalls. For large dataset we choose all C8 Category A dataset that contains 137MM vectors of 512 dimensions. See Figure 3 for speed and recall grid and PFs. We observe that 1) neither models achieve high recall, and 2) HNSW achieves the best speed. We believe the reason for low recall on large dataset is that our CL encoders were sharpened only on the Category A train dataset, hence the alignment is poor on low SI products. We will explore sharpening on low SI target datasets as part of future work.

4.5.1 Evaluating Optimized ANNS. PFs for the dataset can guide the choice of ANNS algorithm and configuration based on speed and recall demands. We propose choosing the configuration from PFs of a representative dataset in the same size bracket to avoid re-generating PF for each new dataset. To justify this hypothesis we perform following evaluations. We compare PFs between Category B and Category C train dataset that belong to the medium size bracket for HNSW. We find that the PFs are comparable and similar configurations achieve high recall (see Figure 4). We also evaluate the optimal configurations from different size brackets on Yandex Text-to-Image datasets. We find that both small and medium configurations perform well in terms of recall and speed (see Table 3). We also use the highest recall configurations to find retrieval accuracy on Category B datasets excluding C8. We find that the accuracies are high that suggest ANNS configurations translate to different datasets of similar sizes. We also evaluate ANNS retrieval against a baseline token-overlap based retrieval method. We find that 1) ANNS retrieves more than 40% novel neighbors than token-overlap, 2) ANNS achieves 8% higher textual embeddings similarities than token-overlap using OpenAI text encoders, and 3) ANNS neighbors have better scores on multiple downstream tasks such as duplicate detection and near-duplicate detection tasks. See Table 4 for the summary on the best performing ANNS algorithms.

[3]The algorithm identifiers contain algorithm name, metric (inner product) and engine.



(a) Small - Deep1B (10MM) (b) Medium - C8 Category B Train (18MM) (c) Large - C8 Category A (137MM)

Figure 3: PFs for the three dataset brackets and three algorithms. We observe that different ANNS algorithms perform differently across dataset sizes and speed/recall trade-offs. See Table 4 for summary on best performing algorithm.

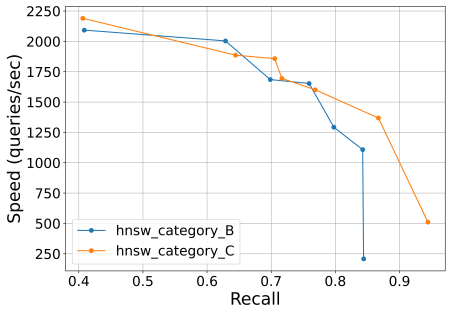


Figure 4: Comparing HNSW PFs for similar sized datasets Category B and C train. Pareto efficient configurations perform consistently across datasets with similar sizes.

Table 3: Evaluating optimized ANNS configurations on Yandex multi-modal dataset and Category B datasets.

Dataset	Search Space	T2I@100	Speed
Yandex	10MM	0.88	1622
Yandex	100MM	0.8	1355
C1 Category B	4MM	0.952	2350
C2 Category B	1MM	0.989	3562
C3 Category B	3MM	0.984	3320
C4 Category B	3MM	0.837	3112
C5 Category B	3MM	0.956	3215
C6 Category B	4MM	0.884	2267
C7 Category B	4MM	0.957	2743

4.5.2 Recommendations on Deploying OpenSearch. We use the latest AWS Graviton2 R6g data nodes and C6g master nodes for all our experiments. To deploy ANNS with OpenSearch, it is important to tune shards and replicas for optimal search speed. Shard is like an index partition while replica is a copy of the primary index. We perform experiments on shards and replicas on Yandex 10MM and 100MM datasets. We found that fewer shards result in slower indexing but higher search speed and vise-versa. The average shard size should be as big as would fit in memory (even 50 GB or more). More replicas improve search speed at same recall. We achieve similar speed on 10× larger Yandex sample using 10X cluster size, 5X number of shards and 5X number of replicas. It is also important to understand indexing and search requirements for deployment. Indexing is a slow step, for *e.g.*,, Yandex 100MM takes around 2.5 hours to index while only 77 seconds to fetch 100 neighbors for 100k queries. Hence, complete re-indexing should be avoided especially for online use cases. However, keeping a large index running can be expensive. Instead, we suggest to delete replicas and store primary index in t3.small instances with enough storage. When a query needs to be executed, replicas and data nodes can be updated which is much quicker than full indexing.

Table 4: Overview of best performing ANNS algorithms.

Dataset (#vecs, dim)	Index Size	Cluster Configuration	Shards / Replicas	Use Case	ANNS Algorithm
SMALL Deep1b (10MM, 96)	37GB	4 r6g.12xlarge	1 / 3	High Recall	IVF (Flat)
				High Speed	HNSW
MEDIUM Category B (18MM, 512)	143GB	18 r6g.12xlarge	3 / 6	High Recall	IVF (Flat) / HNSW
				High Speed	IVF (Flat)
LARGE Category A (137MM, 512)	1TB	40 r6g.12xlarge	20 / 1	High Recall	-
				High Speed	HNSW

5 CONCLUSION

We studied generating multi-modal embedding vectors via contrastive learning and finding top-k neighbors for every item in multi-modal e-commerce datasets at desired speed-recall operating points using OpenSearch ANNS engines. We show that sharpening on target categories improves recall significantly due to sampling harder negatives. We demonstrate that ANNS engines running with Pareto point configurations achieves higher queries per second. We propose methods to scale ANNS without generating PFs for each dataset as well as insights to maintain clusters at low cost.

REFERENCES

[1] Mario Almagro, David Jiménez, Diego Ortego, Emilio Almazán, and Eva Martínez. Block-scl: Blocking matters for supervised contrastive learning in product matching, 2022.

[2] Martin Aumüller, Erik Bernhardsson, and Alexander Faithfull. Ann-benchmarks: A benchmarking tool for approximate nearest neighbor algorithms, 2018.

[3] Artem Babenko and Victor Lempitsky. Efficient indexing of billion-scale datasets of deep descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2055–2063, 2016.

[4] Oren Barkan and Noam Koenigstein. Item2vec: Neural item embedding for collaborative filtering, 2016.

[5] Lei Chen and Hou Wei Chou. Utilizing cross-modal contrastive learning to improve item categorization BERT model. In *Proceedings of the Fifth Workshop on e-Commerce and NLP (ECNLP 5)*, pages 217–223, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *CoRR*, abs/2002.05709, 2020.

[7] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale, 2019.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[9] Xiao Dong, Xunlin Zhan, Yangxin Wu, Yunchao Wei, Michael C. Kampffmeyer, Xiaoyong Wei, Minlong Lu, Yaowei Wang, and Xiaodan Liang. M5product: Self-harmonized contrastive learning for e-commercial multi-modal pretraining, 2021.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020.

[11] Jiali Duan, Liqun Chen, Son Tran, Jinyu Yang, Yi Xu, Belinda Zeng, and Trishul Chilimbi. Multi-modal alignment using representation codebook, 2022.

[12] Andreas Fürst, Elisabeth Rumetshofer, Viet Tran, Hubert Ramsauer, Fei Tang, Johannes Lehner, David Kreil, Michael Kopp, Günter Klambauer, Angela Bitto-Nemling, and Sepp Hochreiter. Cloob: Modern hopfield networks with infoloob outperform clip, 2021.

[13] Mihajlo Grbovic, Vladan Radosavljevic, Nemanja Djuric, Narayan Bhamidipati, Jaikit Savla, Varun Bhagwan, and Doug Sharp. E-commerce in your inbox: Product recommendations at scale. 06 2016.

[14] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. *CoRR*, abs/1911.05722, 2019.

[15] Mariya Hendriksen, Maurits Bleeker, Svitlana Vakulenko, Nanne van Noord, Ernst Kuiper, and Maarten de Rijke. Extending clip for category-to-image retrieval in e-commerce, 2021.

[16] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, page 604–613, New York, NY, USA, 1998. Association for Computing Machinery.

[17] Yang Jin, Yongzhi Li, Zehuan Yuan, and Yadong Mu. Learning instance-level representation for large-scale multi-modal pretraining in e-commerce, 2023.

[18] Herve Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011.

[19] Wen Li, Ying Zhang, Yifang Sun, Wei Wang, Wenjie Zhang, and Xuemin Lin. Approximate nearest neighbor search on high dimensional data — experiments, analyses, and improvement (v1.0), 2016.

[20] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2017.

[21] Dinh The Luc. *Pareto Optimality*, pages 481–515. Springer New York, New York, NY, 2008.

[22] Haoyu Ma, Handong Zhao, Zhe Lin, Ajinkya Kale, Zhangyang Wang, Tong Yu, Jiuxiang Gu, Sunav Choudhary, and Xiaohui Xie. Ei-clip: Entity-aware interventional contrastive learning for e-commerce cross-modal retrieval. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18030–18040, 2022.

[23] Yu. A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs, 2016.

[24] Ralph Peeters and Christian Bizer. Supervised contrastive learning for product matching. In *Companion Proceedings of the Web Conference 2022*. ACM, apr 2022.

[25] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 18–24 Jul 2021.

[26] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021.

[27] Amazon Web Services. Opensearch, 2021.

[28] Harsha Vardhan Simhadri, George Williams, Martin Aumüller, Matthijs Douze, Artem Babenko, Dmitry Baranchuk, Qi Chen, Lucas Hosseini, Ravishankar Krishnaswamy, Gopal Srinivasa, Suhas Jayaram Subramanya, and Jingdong Wang. Results of the neurips'21 challenge on billion-scale approximate nearest neighbor search, 2022.

[29] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning?, 2020.

[30] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

[31] Jizhe Wang, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. Billion-scale commodity embedding for e-commerce recommendation in alibaba, 2018.

[32] Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance-level discrimination. *CoRR*, abs/1805.01978, 2018.

[33] Jinyu Yang, Jiali Duan, Son Tran, Yi Xu, Sampath Chanda, Liqun Chen, Belinda Zeng, Trishul Chilimbi, and Junzhou Huang. Vision-language pre-training with triple contrastive learning, 2022.

[34] Chun-Hsiao Yeh, Cheng-Yao Hong, Yen-Chi Hsu, Tyng-Luh Liu, Yubei Chen, and Yann LeCun. Decoupled contrastive learning, 2021.