

Exploring Bottom-Up Clustering for Creating Semantic IDs

Leah Woldemariam

Department of Electrical and Computer Engineering
Cornell Tech, Cornell University
New York, NY, USA
lsw85@cornell.edu

Sudhanshu Garg

PayPal
San Jose, California, USA
sudgarg@paypal.com

Taha Belkhouja

PayPal
San Jose, California, USA
tbelkhouja@paypal.com

Charles Kim-Yip

PayPal
San Jose, California, USA
cyip@paypal.com

Ali Sahami

PayPal
San Jose, California, USA
asahamishirazi@paypal.com

Abstract

The success of generative retrieval has largely been attributed to the use of Semantic IDs, which improve over arbitrary item-level identifiers such as hashes by capturing the semantics of items. The main challenges faced when constructing Semantic IDs, however, is in mapping each identifier to a unique product and capturing information valuable to downstream tasks. Past works have appended additional codewords to de-duplicate item identifiers and utilized residual quantization to create hierarchical clusters. In this work, we present an algorithm for generating Semantic IDs that ensure the identifiers are both unique and preserve the structure of the original embedding. Key to our work is the use of bottom-up clustering to preserve local structure in the embedding space, improving the clustering quality of the resulting Semantic IDs and their utility for downstream generative retrieval.

1 Introduction

Generative retrieval has been established as a method for creating recommendation systems that provides better generalization performance on new items [RMS⁺23]. Instead of matching user and item embeddings, generative retrieval takes user information as input and generates potential items by treating retrieval as a sequential task. User information might include previous item interactions or text queries for the task of search. Its main advantage includes avoiding the error propagation due to multi-stage filtering, as well avoiding popularity bias and cold-start issues [LGC⁺25, RMS⁺23]. Since items in these sequential models must be quantized, a recent line of research has been directed towards approaches to creating these quantized identifiers. Using item semantics to create these identifiers has shown to be the main facilitator in providing better item generalization [RMS⁺23].

Identifiers that encode item semantics, called Semantic IDs (SIDs), are usually a short sequence of integers built in a hierarchical fashion so that top-level codewords represent coarse, categorical information, and lower level codewords increase precision in identifying an item. The item hierarchy allows for a smaller search space at each decoding step [TTD⁺22] and allows for a clear method for recommending diverse items [RMS⁺23]. Among methods used for generating Semantic IDs, Residual-Quantized Variational Autoencoders (RQ-VAEs) and Residual K-Means (R-KMeans) stand out as the most popular due to the use of residual quantization for enforcing the item hierarchy.

The primary issues surrounding the creation of Semantic IDs include codebook collapse and item collision. Codebook collapse occurs when only a few codewords within a codebook at any given level in a hierarchy contain the vast majority of items, under-utilizing the entire codebook space, while collisions occur when a single SID maps to multiple items [LHF⁺25]. Together, these issues can

prevent semantically distinct items from being assigned unique and informative identifiers. Arbitrary assignment methods such as hashing or plain enumeration satisfy the property of uniqueness, but they fail to encode semantic relationships between items. Generating good SIDs therefore requires balancing two objectives: assigning each item a unique identifier while ensuring that semantically similar items are mapped to nearby Semantic IDs.

In this work, we propose an approach to creating Semantic IDs that both satisfies uniqueness and preserves distance in the embedding space. Using bottom-up rather than top-down clustering, we assign items to leaf nodes first, this ensures that each item corresponds to a single leaf node (uniqueness) and similar items in the embeddings space are inherently assigned nearby SIDs. The motivation behind the use of a bottom-up clustering is to preserve this local structure throughout the SID hierarchy: two items with similar embeddings will more likely share the same $(L - 1)$ -th codeword when there are L codewords, thereby preserving fine-grained semantic relationships that can be exploited by downstream models. In contrast to conventional methods, which often introduce uniqueness through an ad hoc deduplication step and prioritize global structure during quantization, our approach preserves local neighborhoods by design. By jointly preserving uniqueness and local embedding structure, our approach provides a more faithful discrete interface between pretrained item representations and downstream retrieval decisions, directly addressing the broader question of how pretrained representations can be made more actionable for downstream tasks.

2 Related Work

Semantic IDs. Most work on SIDs use Residual Quantization, with an earlier model using Vector Quantization alone is one exception [VDOV⁺17]. [JCN⁺25] find that RK-Means can out-perform RQ-VAE, noting the added complexity does not necessarily lead to a significant increase in performance. Finally, [LCT⁺25] finds that the structure of SIDs themselves may limit the scaling of GR models, and that combining SID embeddings with collaborative filtering does not improve scaling. [XZL⁺26] introduces orthogonal regularization for multi-modal embeddings in order to ensure the weights learned by each expert do not learn redundant information.

Collision and Codebook Utilization. Many works cite that the primary issue in creating SIDs through the RQ-VAE is the high collision rate and low codebook utilization [KCW⁺24]. [KCW⁺24] finds that not only can codebooks heavily concentrate products into a small number of codes, but that this phenomenon may be isolated into specific levels. [LHF⁺25] similarly note that one pitfall of the Semantic IDs is the lack of uniqueness of the Semantic IDs, which they solve by concatenating the Semantic ID with unique identifiers from the item’s embedding. At Snap, [JZN⁺26] finds the codebook collapse and collision are primary issues in creating SIDs, and mitigate these issues by backpropagating through the codebook and by introducing heuristics, respectively. Alternatively, semantics is usually measured and improved by categorical/tag information [Pen]. [LZW⁺26] adds terms to the loss function that maximize dispersion and improves codebook utilization efficiency.

3 Methods

Overview. As in Algorithm 1 presented in the Appendix, the set of items set can be represented as an $N \times d$ matrix X . Each $x_i \in \mathbb{R}^d$ represents a number of text item attributes used as input together into a pre-trained transformer to create an embedding. Denote the set of labels for each item i at level ℓ as $g^{(\ell)}$, and $(g_i^{(1)}, \dots, g_i^{(L)})$ represents the Semantic ID for item i .

Unlike arbitrary IDs, learned SIDs often struggle to guarantee uniqueness. At the same time, products that are similar to each other should be assigned nearby SIDs in the identifier hierarchy. This is one of the key properties a SID should have and is critical for mitigating the cold-start problem, as it allows unseen or sparsely observed items to share identifier structure with semantically related products. Thus, mapping these embeddings to discrete identifiers can be viewed as a constrained hierarchical clustering problem: the resulting identifiers should uniquely distinguish individual items while preserving the local structure of the original embedding space.

We seek to hierarchically assign items their identifiers through bottom-up clustering, i.e., by first clustering items and successively merging them at each level in the hierarchy. By using bottom-up clustering, we preserve local structure within the embedding space and its fine-grained item relationships. In particular, nearby items are grouped together at the lowest levels of the hierarchy before these local groups are progressively merged into coarser clusters. This is in contrast to top-down approaches, which first capture global information in the embedding space and subsequently partition it into increasingly fine-grained regions, potentially losing local relationships introduced at

deeper levels of the hierarchy. By preserving these local relationships, our approach aims to retain the information in pretrained item representations that is most useful for downstream retrieval decisions, providing a more actionable interface between latent representations and generative models.

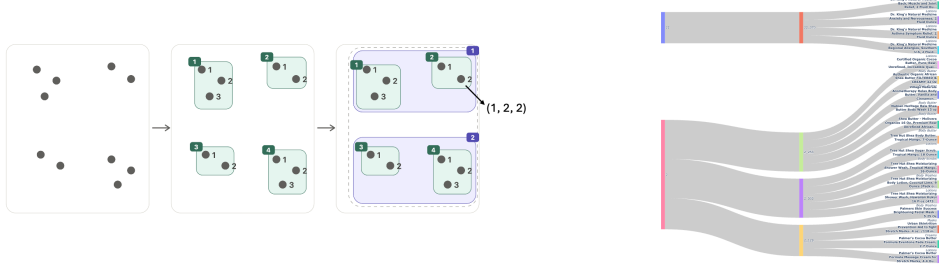


Figure 1: (a) A diagram depicting bottom-up clustering, with each node representing the embedding of an item. In the leftmost panel are the unclustered items, and in the middle panel is the initial clusters (green). The rightmost panel shows the final hierarchical clustering with the top-level groups in purple. (b) Example of a sample items assigned SIDs using the bottom-up approach.

Algorithm. We construct a hierarchical, bottom-up clustering of the product embeddings $X \in \mathbb{R}^{N \times d}$ as described in Algorithm 1. The procedure begins at the finest level by clustering the raw embeddings, streamed in chunks, to partition N products into k_0 fine-grained groups. To ensure that the final codebook uniquely identifies each item, groups containing more than K items are further split until every sub-cluster contains at most K items. We enumerate each item within these fine clusters and assign its index as the L -th codeword, ensuring every product has a unique within-cluster identifier.

For each of the initial groups, we compute a weighted centroid using its member count, yielding a compact set of k_1 representative points that summarizes the full dataset. We then coarsen the hierarchy upward by iteratively merging these centroids. At each level, we cluster the current set of centroids using standard Agglomerative Clustering, using each centroid’s member count as a sample weight so that the merge respects the true population mass. After each merge, we recompute the centroids and weights for the newly formed groups, and propagate the new labels back down to every original product. Repeating this produces a sequence of increasingly coarse cluster assignments.

Each cluster at every level is enumerated $1, \dots, k_\ell$ as the standard format of SIDs. Note that k_ℓ need not be the same at each level. Merging clusters at $g^{(\ell)}$ creates the $\ell - 1$ codewords, thereby creating clusters by moving up the tree. See Algorithm 1 for a more detailed description.

New items are assigned SIDs according to their nearest neighbors. A new item inherits the first $L - 1$ codewords of its nearest neighbor, with a unique L -th codeword assigned within the selected cluster. When a cluster exceeds a predefined threshold, it can be further split into sub-clusters. Cluster assignment may also use multiple nearest neighbors for greater robustness. Importantly, this procedure allows the SID hierarchy learned from pretrained item representations to generalize to previously unseen items without reconstructing the identifier space. By transferring local structure to new items, the resulting SIDs make pretrained representations directly actionable for downstream retrieval under cold-start settings.

4 Experiments

Metrics. We evaluate our approach with a number of metrics to measure codebook quality and clustering quality, and additionally evaluate our method over a next-product prediction task to assess the downstream utility of the resulting Semantic IDs. To measure codebook quality, we measure the average number of items assigned to each SID, which captures the extent to which multiple items share the same identifier and provides an indication of identifier collisions and codebook utilization. To measure clustering quality, we evaluate silhouette score and cosine similarity; we measure these metrics at each depth of the tree created by the SIDs, i.e. for clusters created by the first codebook, second codebook, and so on, while emphasizing the values at the final clusters. These metrics characterize how well the SID hierarchy preserves the structure of the pretrained embedding space.

When a dataset has categorical labels for each item, we additionally measure the average number of categories among items within a codeword, where fewer categories indicate greater semantic coherence within the resulting clusters. As in [RMS⁺23], recall and NDCG is reported for item prediction tasks. These downstream metrics allow us to evaluate whether improvements in the

Metric	Custom Dataset		Beauty		Sports & Outdoors	
	RQ-VAE	Bottom-Up	RQ-VAE	Bottom-Up	RQ-VAE	Bottom-Up
Codebook Quality						
Avg. Rows per ID	1.63	1	1.35	1	1.49	1
Clustering Quality						
Avg. Silhouette Score	0.00	0.29	−0.07	0.07	0.02	0.07
Cosine Similarity	0.80	0.82	0.76	0.80	0.88	0.79
Next Product Prediction						
Recall@10	0.020	0.036	0.0509	0.0567	0.0328	0.0327
NDCG@10	0.001	0.002	0.0255	0.0296	0.0169	0.0166
Other						
Avg. Unique Categories	1.06	14.30	1.16	3.75	1.21	4.54

Table 1: Comparison of RQ-VAE and Bottom-Up clustering methods across the 5.8M Item dataset and Amazon Beauty datasets. Cluster-based metrics are reported at the final depth of the tree.

structure of the Semantic IDs translate into better retrieval decisions, connecting representation quality to downstream actionability.

Baselines. We compare against an RQ-VAE, the a widely used method for constructing SIDs, as our primary baseline. The RQ-VAE constructs codebooks in a top-down, hierarchical fashion starting with a fixed codebook size L .

Datasets. We compare our algorithm to the above baselines with the Amazon Product Reviews dataset, a collection of product reviews for items [NLM19]. We evaluate over the Beauty and Sports & Outdoors splits. We additionally compare our algorithm against the same baselines on a custom dataset containing approximately 6 million items, each with titles, model-generated captions of corresponding images, categorical information, and brand information. For each product, we concatenate the available textual information and encode it using a Qwen model to obtain the pretrained item embedding from which its SID is constructed.

RQ-VAE & Next Product Prediction Model Implementation. We first perform a hyper-parameter search on the batch size, learning rate, weight decay, codebook mode, codebook size, number of codebooks, and VAE hidden dimensions to compare our approach against the best possible RQ-VAE. The RQ-VAE used in these experiments has a codebook size of 256 for 3 codebooks. An extra codeword was appended to the RQ-VAE generated SIDs so that each item corresponding to a single SID, following the style of [RMS⁺23, JCN⁺25].

Results. Table 1 reports the metrics for the finest-grained groups. For the bottom-up approach, this is evaluated at level $L - 1$, corresponding to clusters defined by the tuples $(g^{(1)}, \dots, g^{(L-1)})$, since the L -th level always maps to unique items.

The bottom-up approach achieves higher silhouette scores across all datasets at the finest granularity, indicating that its finest-level clusters are more compact and better separated in the pretrained embedding space. The gap is particularly large on the custom dataset, which contains substantially more items and greater product diversity than the Amazon Reviews Beauty dataset. This suggests that preserving local structure may become increasingly important as the item space grows more heterogeneous. The bottom-up approach also achieves higher Recall@10 and NDCG@10 on the downstream next-item prediction task, showing that improved SID structure translates to better downstream retrieval. The bottom-up approach also yields a higher average number of unique categories within each cluster, which is expected because the optimal RQ-VAE configuration uses a codebook size substantially larger than the number of clusters at each level in the bottom-up hierarchy.

Unlike RQ-VAE, which learns a residual hierarchy from coarse to fine, our bottom-up method constructs the SID hierarchy directly from local relationships in the pretrained embedding space. We show that this structure-preserving approach leads to better clustering quality as well as improved performance on the downstream task of next-item prediction. More broadly, these results suggest that the way pretrained representations are converted into discrete abstractions materially affects their downstream utility: preserving task-relevant local structure can make those representations more actionable for downstream decision-making.

References

- [JCN⁺25] Clark Mingxuan Ju, Liam Collins, Leonardo Neves, Bhuvish Kumar, Louis Yufeng Wang, Tong Zhao, and Neil Shah. Generative recommendation with semantic ids: A practitioner’s handbook. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 6420–6425, 2025.
- [JZN⁺26] Clark Mingxuan Ju, Tong Zhao, Leonardo Neves, Liam Collins, Bhuvish Kumar, Jiwen Ren, Lili Zhang, Wenfeng Zhuo, Vincent Zhang, Xiao Bai, et al. Semantic ids for recommender systems at snapchat: Use cases, technical challenges, and design choices. *arXiv preprint arXiv:2604.03949*, 2026.
- [KCW⁺24] Zhirui Kuai, Zuxu Chen, Huimu Wang, Mingming Li, Dadong Miao, Wang Binbin, Xusong Chen, Li Kuang, Yuxing Han, Jiaying Wang, et al. Breaking the hourglass phenomenon of residual quantization: Enhancing the upper bound of generative retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 677–685, 2024.
- [LCT⁺25] Jingzhe Liu, Liam Collins, Jiliang Tang, Tong Zhao, Neil Shah, and Clark Mingxuan Ju. Understanding generative recommendation with semantic ids from a model-scaling view. *arXiv preprint arXiv:2509.25522*, 2025.
- [LGC⁺25] Siyi Lin, Chongming Gao, Jiawei Chen, Sheng Zhou, Binbin Hu, Yan Feng, Chun Chen, and Can Wang. How do recommendation models amplify popularity bias? an analysis from the spectral perspective. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining*, pages 659–668, 2025.
- [LHF⁺25] Guanyu Lin, Zhigang Hua, Tao Feng, Shuang Yang, Bo Long, and Jiaxuan You. Unified semantic and id representation learning for deep recommenders. *arXiv preprint arXiv:2502.16474*, 2025.
- [LZW⁺26] Xiaoyu Liu, Fuwei Zhang, Yiqing Wu, Xinyu Jia, Zenghua Xia, Fuzhen Zhuang, Zhao Zhang, Fei Jiang, and Wei Lin. Cat-id2: Category-tree integrated document identifier learning for generative retrieval in e-commerce. In *Proceedings of the Nineteenth ACM International Conference on Web Search and Data Mining*, pages 426–435, 2026.
- [NLM19] Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 188–197, 2019.
- [Pen] et al. Penha. Semantic ids for generative search and recommendation.
- [RMS⁺23] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, et al. Recommender systems with generative retrieval. *Advances in Neural Information Processing Systems*, 36:10299–10315, 2023.
- [TTD⁺22] Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, et al. Transformer memory as a differentiable search index. *Advances in neural information processing systems*, 35:21831–21843, 2022.
- [VDOV⁺17] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.
- [XZL⁺26] Yi Xu, Moyu Zhang, Chenxuan Li, Zhihao Liao, Haibo Xing, Hao Deng, Jinxin Hu, Yu Zhang, Xiaoyi Zeng, and Jing Zhang. Mmq: Multimodal mixture-of-quantization tokenization for semantic id generation and user behavioral adaptation. In *Proceedings of the Nineteenth ACM International Conference on Web Search and Data Mining*, pages 788–797, 2026.

A Appendix

Below is the algorithm describing our bottom-up approach. In our experiments, the bottom-up clustering approach used 2000 and 64 clusters at level $L - 1$ and $L - 2$, respectively for the custom dataset and otherwise matches the codebook size used by the RQ-VAE for the Amazon datasets.

Algorithm 1 Bottom-Up Hierarchical Clustering

Require: Embedding matrix $X \in \mathbb{R}^{N \times d}$; strictly decreasing coarsening schedule $K = [k_1, k_2, \dots, k_L]$; optional per-cluster size cap $c_{\max}$.

Ensure: Per-item level labels $\{g^{(0)}, g^{(1)}, \dots, g^{(L)}\}$, each $g^{(\ell)} \in \{0, 1, \dots\}^N$ giving every item i a label at level ℓ ; tuple $(g_i^{(1)}, g_i^{(2)}, \dots, g_i^{(L)})$ is unique per item.

Step 1: Finest clustering

- 1: $g^{(L-1)} \leftarrow \text{MINIBATCHKMEANS}(X, k_1) \triangleright$ returns a label $g_i^{(L-1)} \in \{0, \dots, k_1 - 1\}$ for every item i
- 2: $(C, w) \leftarrow \text{GROUPCENTROIDS}(X, g^{(L-1)}) \triangleright C \in \mathbb{R}^{k_{\ell} \times d}$ centroids; w : its item count $\triangleright a$ maps cluster labels at $\ell + 1$ to labels at level ℓ
- 3: **for all** groups s in $g^{(L-1)}$ **do**
- 4: assign the members of s distinct ids $0, 1, 2, \dots$ (running count within s) as their label $g^{(L)}$
- 5: **end for**
- 6: $g^{(L)} \leftarrow$ these per-item ids $\triangleright$ (uniqueness)

Step 2: Recursive Merging

- 7: **for** $\ell = L - 2$ **to** 1 **do**
 - 8: $g^{(\ell)} \leftarrow \text{MINIBATCHKMEANS}(C, k_{\ell}, \text{sample_weight} = w, \text{parent} = g^{(\ell+1)})$
 - 9: **if** $c_{\max} \neq \emptyset$ **then** $g^{(\ell)} \leftarrow \text{ENFORCEMAXCLUSTER SIZE}(X, g^{(\ell)}, c_{\max})$ $\triangleright$ splits any level- ℓ group with more than $c_{\max}$ items
 - 10: **end if**
 - 11: relabel $g^{(\ell)}$ to dense ids $\{0, \dots, n_{\ell} - 1\}$
 - 12: $(C, w) \leftarrow \text{GROUPCENTROIDS}(X, g^{(\ell)})$
 - 13: **end for**
 - 14: **return** $(g^{(1)}, g^{(1)}, \dots, g^{(L)})$
-