软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, [doi: 10.13328/j.cnki.jos.007515]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

权重残差向量量化：向量压缩与分层索引结构 *

江宇轩[1]、姚俊杰[1,2]、侯宇轩[1]

[1](华东师范大学，上海 200062)

[2](上海市高可信重点实验室，上海 200062)

通讯作者：姚俊杰, E-mail: junjie.yao@sei.ecnu.edu.cn

摘 要：随着多源异构数据、多模态等在大模型和数据湖等场景的广泛应用，基于向量的数据检索和存储管理显著增长。通过将异构数据映射为高维向量表示，并以向量索引为基础，向量数据库将多种数据类型统一管理和高质量相似性检索，成为生成式检索和 AI 数据库等重要基础。然而，现有向量数据库在存储索引效率、索引构建复杂度及检索准确性方面面临显著瓶颈：一方面，海量高维向量导致索引存储开销和维护成本增加；另一方面，向量索引结构冗长，内存消耗巨大；此外，压缩技术失真引发的检索准确性下降问题仍未有效解决。

本文提出了一种基于权重残差向量量化（Weight Residual Vector Quantization，WRVQ）的新型框架。该方法通过将量化方向与残差长度分离处理，以单位向量形式存储残差方向并附加权重标记，实现了低失真率下的高效压缩与存储。在索引构建方面，本文设计了适配 WRVQ 量化特性的三层倒排索引结构——精确匹配层、模糊匹配层与搜索层，有机结合非对称距离计算（ADC）与近邻搜索技术，实现了高准确度与高效率兼具的近似最近邻检索。在大规模数据集上的实验结果表明，与传统低维嵌入模型及现有量化方法相比，WRVQ 在量化损失、存储压缩比和检索召回率等关键指标上均取得了显著提升，且索引构建与查询性能有显著优势。

关键词：向量数据库；向量量化；近似查询处理

中文引用格式：江宇轩，姚俊杰，侯宇轩. 权重残差向量量化：向量压缩与分层索引结构.软件学报. http://www.jos.org.cn/1000-9825/7515.htm

英文引用格式: Jiang YX, Yao JJ, Hou YX,. Weight Residual Quantization : Vector Compression and Hierarchical Index Structure. Ruan Jian Xue Bao/Journal of Software (in Chinese). http://www.jos.org.cn/1000-9825/7515.htm

Weight Residual Quantization : Vector Compression and Hierarchical Index Structure

JIANG Yu-Xuan[1], YAO Jun-Jie[1,2], HOU Yu-Xuan[1].

[1](East China Normal University, Shanghai, 200062, China)

[2](Shanghai Key Laboratory of Trustworthy Computing, Shanghai, 200062, China)

Abstract: With the widespread application of multi-source, heterogeneous, and multi-modal data in scenarios such as large-scale models and data lakes, there has been a significant growth in vector-based data retrieval and storage management. By mapping heterogeneous data into high-dimensional vector representations and leveraging vector indices, vector databases facilitate the unified management of diverse data types and enable high-quality similarity search, establishing them as a crucial foundation for applications like generative retrieval and AI-native databases. However, existing vector databases face significant bottlenecks in terms of storage and indexing efficiency, index construction complexity, and retrieval accuracy. Specifically, massive high-dimensional vectors lead to increased storage overhead and maintenance costs for indices. Furthermore, vector index structures are often bloated, resulting in substantial memory consumption. Moreover, the degradation of retrieval accuracy caused by distortion from compression techniques remains an unresolved challenge

This paper proposes a novel framework based on Weight Residual Vector Quantization (WRVQ). This method achieves high-efficiency compression and storage with an extremely low distortion rate by decoupling the quantization direction from the residual magnitude. It stores the residual's direction as a unit vector and appends a corresponding weight tag. For indexing, we design a three-layer inverted

基金项目：国家重点研发计划（2024YFC2607402），国家自然科学基金（(61972151)。

收稿时间: 2025-05-06; 修改时间: 2025-06-30, 2025-08-14; 采用时间: 2025-08-20; jos 在线出版时间: 2025-09-02

index structure tailored to the characteristics of WRVQ, comprising an Exact Match Layer, a Fuzzy Match Layer, and a Search Layer. This structure organically integrates Asymmetric Distance Computation (ADC) with nearest neighbor search techniques to realize Approximate Nearest Neighbor (ANN) search that balances both high accuracy and high efficiency. Experimental results on large-scale datasets demonstrate that, compared to traditional low-dimensional embedding models and existing quantization methods, WRVQ achieves significant improvements across key metrics, including quantization loss, storage compression ratio, and retrieval recall. Furthermore, it exhibits considerable advantages in both index construction and query performance.

Key words: Vector Database, Vector Quantization, Approximate Query Processing.

近年来，信息技术的快速发展以及智能终端的广泛普及，使得全球数据量呈现爆炸式增长，显著推动了数据库技术的不断创新和变革。向量数据库通过将各类异质数据转换为向量表示，并利用向量作为数据索引，不仅实现了对异质数据的统一管理，还能够从原始数据中提取特征，以便进行计算和比对。向量数据库在生成式检索领域得到了广泛应用，其对数据特征的表征能力有效提升了数据相似性检索的质量，并借助近邻搜索等技术，为下游模型提供了更具相关性的数据信息。

在向量数据库中，向量的表征与检索是两个核心环节。向量表征直接决定了数据的表示能力，从而影响整体系统的可靠性；而向量检索的准确性则直接关系到系统输出结果的质量。随着数据规模和呈指数级增长，向量数据库在数据存储和索引构建等方面仍然存在诸多性能瓶颈。当前的向量数据库在存储、计算和检索过程中面临以下挑战：（1）随着数据量和多样性的不断增长，存储索引的效率逐步下降，并在大规模异质数据场景下带来较高的维护成本；（2）传统的索引构建方法结构庞大且复杂，导致在大规模数据计算时内存消耗巨大；（3）数据规模的扩张使得检索算法在准确性和效率方面面临更大挑战。

近期，诸多研究提出了多种改进方案以优化向量数据库在大规模数据存储和检索过程中的性能问题。量化技术作为一种实现向量压缩和高效索引的有效手段得到较多探讨。Xiao 等人[1]提出的逐步优化双粒度文档表示方法，通过对比量化生成稀疏嵌入，实现了精确且高效的候选搜索；同时，Zhang 等人[2]开发的集成高维向量索引至关系数据库，利用弛张单调性构建索引，有效支持了复杂的近似相似性查询。尽管上述方法在缓解存储与计算瓶颈方面取得了一定进展，其适用性仍存在一定局限。首先，量化过程的失真往往导致检索准确性降低，使得系统在高精度需求场景下表现不足；其次，索引构建及维护往往较为复杂，内存消耗增多，且响应速度难以满足实时性要求；此外，量化处理的灵活性不足，限制了在多样的向量数据管理场景的适用。

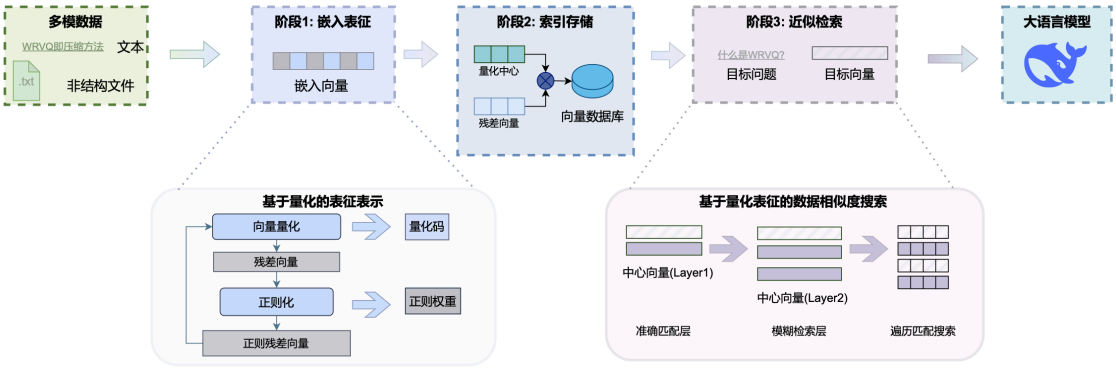


图 1 基于量化表征的向量索引和检索流程图

针对以上难点，本文提出了一种新颖的权重残差向量量化（Weight Residual Vector Quantization，WRVQ）方法，旨在解决大规模异构数据处理中的存储与检索问题。本文的工作总体分为量化表征、存储索引和近似匹配三个部分，如图 1 所示。在索引构建方面，本文设计了适配 WRVQ 量化特性的三层倒排索引结构——精确匹配层、模糊匹配层与搜索层。在搜索匹配方面，有机结合非对称距离计算（ADC）与近邻搜索技术，实现了高准确度与高效率兼具的近似最近邻检索。本文方法通过将向量量化与近似最近邻（ANN）搜索过程分

离，构建出基于量化的向量表征，实验结果表明，新的方法不仅在较低失真率下实现了高效压缩和存储，同时支持针对不同数据及特定领域的定制化量化操作，显著提升了系统的检索灵活性与适应性，为向量数据的管理提供了一条新路径[3]。

1 相关工作和基本概念

1.1 向量表征

传统的文本嵌入模型通常被称为词嵌入模型，典型方法包括 word2vec[4] 和 GloVe[5]。这些模型分别采用基于矩阵分解的统计方法和基于浅层窗口的预测方法，但由于缺乏对上下文信息的有效建模，往往难以捕捉深层语义。相比之下，Transformer 通过注意力机制连接编码器和解码器，为当前先进的嵌入模型奠定了理论基础[6]。注意力机制能够有效关注上下文信息，无需通过隐藏层进行信息传递，同时并行计算技术的引入大幅提升了嵌入模型的计算效率。

BERT（Bidirectional Encoder Representations from Transformers）作为广泛使用的文本嵌入模型，采用基于 Transformer 结构的双向编码器表示，并利用大规模自监督预训练策略，在自然语言处理任务上均取得了良好性能[7]。InferSent 将自然语言推理方法引入文本嵌入[8]，而 SBERT（Sentence-BERT）则采用孪生网络和三元网络结构，以生成具有语义意义的句子嵌入，并通过余弦相似性进行比较[9]。此外，近期的一些研究引入对比学习方法，以更有效地指导模型进行正负文本对的微调[10]。

利用大型语言模型的强大语义理解能力，诸多嵌入表征技术将文本、图像等输入映射到高维向量空间中，以支持高效的相似性检索与聚类。Reimers 和 Gurevych 提出的 Sentence-BERT 针对句子级别相似度任务，通过 Siamese 网络结构显著提升检索速度与精度[11]。OpenAI 在 GPT-3 中开放的 Embeddings API，进一步简化了高质量文本嵌入的获取流程，广泛应用于搜索、推荐与对话系统[12]。多模态领域的 CLIP 通过对比学习联合训练图像与文本编码器，使得跨模态检索成为可能[13]。通过直接提示和数据驱动调优两种策略不断优化 LLM 嵌入模型，以应对长文本、多语言及领域定制化需求[14] [15]。

1.2 向量数据管理和查询

为了更高效地管理向量数据，向量数据库近期得到了广泛开发和应用。例如，阿里巴巴研发了 PASE[16] 和 AnalyticDB-V[17]，Facebook 开发了 Faiss[18] 以支持高效的向量计算、查询和检索。此外，数据库初创公司如 Zilliz[19] 和 Chroma[20] 也构建了各自的向量数据库，以满足不同应用场景的需求。

目前，向量数据库大致可分为两类：专用向量数据库和通用向量数据库。专用向量数据库从零开始设计，专门用于管理向量数据，并遵循“非一刀切”（tailored approach）的设计原则[21]，典型代表包括 Faiss、Milvus 和 Chroma。这类数据库通常专注于向量数据的存储、索引与检索，能够通过定制优化达到卓越的性能。相较之下，通用向量数据库则是在现有关系数据库（如 PostgreSQL）的基础上扩展，以支持向量数据管理，并遵循“一刀切”（one-size-fits-all）的设计理念[22]，如 PASE 和 AnalyticDB-V。这类数据库的优势在于能够集成到成熟的关系数据库生态系统中，从而提升系统的可用性，复用已有的数据库功能，消除数据孤岛，并通过统一的系统架构降低运维成本。然而，由于需要兼容关系数据模型，通用向量数据库在向量数据的查询和存储性能方面可能存在一定程度的损失[23]。

在向量数据库的研究工作中，向量检索算法是核心组成部分。其中，大多数近似最近邻（ANN）检索算法属于基于向量表征的检索方法（Embedding-Based Retrieval，EBR）。早期研究主要采用子空间划分技术进行向量近似比较，以提升检索效率，如 KD-Tree[24] 和 K-means Tree[25]。近期研究则更多地关注基于量化的方法（如 IVFADC[26]、IVFPQ[27]），通过优化查询向量与索引向量的距离计算来提升检索性能；以及基于图的方法（如 HNSW[28]），其通过构建图索引结构，将搜索算法的时间复杂度降低至对数级。

随着知识检索需求的不断增长，研究人员针对不同类型和不同领域的数据集进行了大量探索，致力于提升向量检索的时空效率与准确性[29]。其中，Xiao 等人在索引构建过程中通过学习稀疏嵌入，以优化全局判别能力，从而更精准地定位查询向量的潜在区域，该方法为本文的研究提供了重要启发[30]。此外，Tang 等

人提出了一种半有序的高效样本索引方法，以降低索引维护成本与时间开销[31]。这些研究成果为向量数据库的发展提供了有力支撑，并为未来的进一步优化奠定了基础。

1.3 向量量化

向量量化（Vector Quantization）是为了减少表示空间的基数，特别是在输入数据为实值向量时，能够有效降低存储和计算成本。形式上，量化器可被视为一个函数，其作用是将一个 d 维向量映射到一个离散的量化向量空间。

$$x \in R^d \to c_i \in \left\{ c_i \in R^d ; i = 0,1,...,k-1 \right\} \tag{1}$$

设量化向量 c 为簇中心，而簇中心的集合大小 k 被称为码本大小。在实际应用中，由于单一量化器在向量表示能力上的局限性，现有研究通常采用多个量化器对向量进行联合处理，从而生成多个量化向量，以更精确地表示原始向量。

乘积量化（Product Quantization，PQ）是一种典型的向量量化方法，其核心思想是将高维向量空间划分为多个低维子空间，并分别对每个子空间进行量化。最终，每个向量由其低维子空间量化后形成的短码组合表示[27]。在此基础上，优化乘积量化（Optimized Product Quantization，OPQ）通过最小化空间分解误差和量化码本误差，进一步优化 PQ 的空间划分方式，从而降低量化失真[3]。此外，累积量化（Additive Quantization，AQ）关注低维子空间之间的相关性，并优化量化向量与原始向量之间的计算方式，以提高量化器的准确性[32]。

同时，量化器码本的训练过程对于量化器的性能和效率具有重要影响。针对特定数据源，训练出能够精准拟合数据分布的量化簇中心，可有效降低量化误差。一些轻量级量化方法采用基于数据源或残差的线性表示，对码本进行微调，以增强量化效果[33,34]。此外，部分研究结合无监督学习的梯度下降方法，对码本参数进行优化，使得码本更新更加灵活，从而提升向量量化的适应性和泛化能力[35]。

2 一种权重增强的量化表征技术

本节主要介绍在嵌入模型的输出结果上进行针对性量化的设计与实现。在传统的向量量化方法中，影响量化失真的主要因素是码本的大小。由于每个向量被映射为单个码字表示，因此所有向量最终被量化到有限数量的码字集合中。然而，在大多数情况下，码本的大小远小于数据集的规模，导致量化精度较低。这一限制主要源于有限数量的码字难以充分捕捉数据分布的复杂性，从而降低了量化的准确性。

针对上述问题，残差向量量化（Residual Vector Quantization，RVQ）提出了一种改进策略。RVQ 通过引入残差量化器，使原始向量不再仅由单个码字表示，而是通过多个码字的累加近似得到，从而减少量化误差[36]。然而，RVQ 的整体准确性仍然受到码本训练方式的影响，且随着码本层数（即 RVQ 中的迭代次数）的增加，量化器的效率会不可避免地下降。一些研究尝试通过下界排序剪枝（LBS-Pruning）和提前终止（Early Termination, ET）优化码本训练过程，以提高训练效率[37]，但这些方法虽然提升了码本的平均效率，却未能从根本上解决量化损失高的问题。

本文提出了一种新的向量量化方法，旨在针对传统向量量化方法中存在的问题进行优化与改进。该方法在残差向量量化（Residual Vector Quantization, RVQ）的基础上，重点提升量化的准确率，并有效缓解 RVQ 在多个量化轮次后准确率下降的问题。

本方法克服了 VQ 和 PQ 方法的关键局限性，即在量化过程中可能导致嵌入特征的丢失。VQ 和 PQ 往往无法很好地保持嵌入向量之间的高余弦相似度，使得量化后的表示在后续向量检索任务中的有效性降低。而本文提出的方法在量化过程中更好地保留了嵌入特征，使其能够为向量检索任务提供更可靠的支持。此外，本方法在计算效率上进行了优化，确保在保持较高量化准确率的同时，兼顾量化比的合理性。

2.1 问题定义

向量量化的目标是将连续输入向量空间划分为有限数量的区域，并为每个区域分配一个代表性的码字，从而实现紧凑的表示形式。其本质在于最小化原始输入向量与其量化表示之间的失真或误差，使得量化后的

表示尽可能保留原始数据的关键信息。向量量化的主要挑战在于在量化比和准确率之间取得平衡。较高的量化比可以有效降低存储和计算成本，但可能会引入较大的信息损失，从而降低检索或计算的准确性；而过于精细的量化则可能导致存储开销增大，削弱量化的实际效益。

该问题可以定义为：在给定的向量数据分布下，寻找一组最优的码字及其对应的分区方案，使整体失真最小化。通常，失真程度通过特定的距离度量（如欧几里得距离或余弦距离）进行衡量，以确保量化后的向量能较好地近似原始向量。如何在压缩效率和检索性能之间合理权衡，是向量量化方法设计中的关键问题。

为了提高量化向量的准确率，码字通常不是单一的。因此，量化过程可以被视为一个将向量转换为长度为 n 的码字串的函数，每个码字代表着码本中相应的向量。

$$q(x) = f(c_{1i}, c_{2j}, ..., c_{nk}), c_{nk} \in C_n \tag{2}$$

上述公式中，x 代表原始向量，而 c_{nk} 代表第 n 个码本的第 k 个向量。而 C_n 则表示第 n 个码本的向量空间，即其所存储的 s 个向量的集合。

$$C_n = \left\{ c_{ni} \mid c_{ni} \in R^d; i = 0,1,...,s-1 \right\} \tag{3}$$

基于残差的量化方法对初始向量及上一个量化器的量化残差进行量化，因此其向量表示是多个量化器的量化结果相加。

$$x_{rvq} = c_{1i} + c_{2j} + ... + c_{nk} \tag{4}$$

对于量化结果的准确率评估，我们使用量化损失（Quantization loss）进行衡量。对于大小为 m 的数据集，其量化损失为每个向量的误差均值：

$$loss = \frac{1}{m} \sum_{i=1}^{m} \| x_i - x_{vqi} \|_2^2 \tag{5}$$

除了准确率，量化比也是评估向量量化的标准之一。我们都希望尽可能地压缩向量，同时保持其准确性。向量量化的量化比通常不考虑码本的大小，因为码本的大小与数据集大小不相关。因此压缩比则表示为原始向量维度 d 和码字串长度 n 的比值：

$$c_{vq} = \frac{d}{n} \tag{6}$$

2.2 优化残差的量化模型

为充分利用数据集的特性，并解决残差向量量化（RVQ）在多轮量化过程中效率逐渐降低的问题，本文思路是提出一种逐步量化的方法，以提升量化准确率，并确保 RVQ 在后续码本训练时维持较高的量化效率。该方法的关键在于对残差向量的长度进行动态调整，以优化量化过程。此外，通过针对残差特性的优化调整码本训练方式，可以有效减少训练后续码本时所引入的“系统性误差”，从而进一步提高整体量化精度。

每个残差量化器生成的量化向量，其长度信息所携带的有效信息量往往低于预期。这一现象可以通过假设模型进行解释：当残差向量的方向确定后，其长度基本上是确定的，即该长度需保证量化向量与残差向量的叠加结果仍位于单位超球面上。这一特性表明，在量化过程中，合理控制残差向量的长度有助于提升 RVQ 的整体表现，并优化最终的量化效果。

$$len(quantized_{i+1}) = len(quantized_i + d_i \cdot residual_i) \approx len(embedding) = 1 \tag{7}$$

根据公式（7），在残差向量量化（RVQ）中，第 i 个量化器的已量化部分 $quantized_i$ 以及量化目标向量 $embedding$ 均已知。因此，在搜索码本中最优的残差量化向量 $residual_i$ 时，我们仅需关注其方向信息，而其长度信息 d_i 在一定程度上是可以计算得到的。尽管从单个量化向量的角度来看，忽略长度信息可能会导致

一定程度的量化损失，但在码本大小固定的情况下，增加可供选择量化方向能够提升整体的全局量化准确率。

为了充分利用数据集的这一特性，并在量化过程中合理调整码本的“方向”与“长度”属性的重要性，我们优化并改进了残差向量量化方法。具体而言，在每一轮量化后，我们对残差向量进行归一化处理，使其标准化为单位向量，从而显著增强其在码本训练中的作用。在此优化方案下，码本仅存储残差向量的方向信息，而不记录其长度信息。

作为一种权衡措施，每个量化码字额外附加一个标记位，以指示该向量的实际权重，即其原始残差长度。这一策略确保了残差向量的标准化，使得码本训练更加稳定，同时通过标记信息保留了向量的权重信息。在该优化方案下，对于目标向量的量化，WRVQ 量化器可以表示为如下形式：

$$q(x)=\sum_{i=0}^{n} quantized_i \cdot \prod_{j=0}^{i} d_j \tag{8}$$

这种方法的优势在于通过对残差进行缩放处理，确保所有残差的长度保持均匀。在码本训练过程中，不再需要区分具有相同方向但不同长度的残差，这显著增强了量化器对残差的量化能力。更为关键的是，该方法有效防止了随着量化轮次的增加，量化后的残差变得越来越稀疏的问题。通过这种方式，量化器的每一层都面对相同类型的量化对象，从而最大化了其区分能力。最后，随着量化轮次的增加，残差的绝对长度逐渐减小，导致量化器的“系统误差”也随着残差的归一化得以解决。

在从码字中重构原始向量的过程中，当从码本中查询到与某个码字对应的向量后，我们根据相关的标记位进一步对该向量进行缩放。最后，所有与每个码字对应的缩放向量将相加，最终得到原始向量。通过这种方法，我们从根本上解决了传统 RVQ 方法中的缺点，即随着码本数量的增加，量化能力下降的问题。此策略显著提升了 RVQ 的量化准确率。

图 2 展示了 WRVQ 量化的总体流程。在对输入的向量表征进行量化之后，WRVQ 会对量化残差进行归一化处理，并记录标记位，以确保下一轮量化器的量化对象为单位向量。因此，对于每一层量化器，WRVQ 会生成两个码字。在图右侧的码字部分，浅色部分表示残差的长度，而深色部分则表示残差的方向，指向码本中的一个单位向量。这种设计能够有效地保留残差的方向信息，同时通过标记位管理长度信息，在这一迭代权重优化过程中，提升量化过程的准确性和效率。

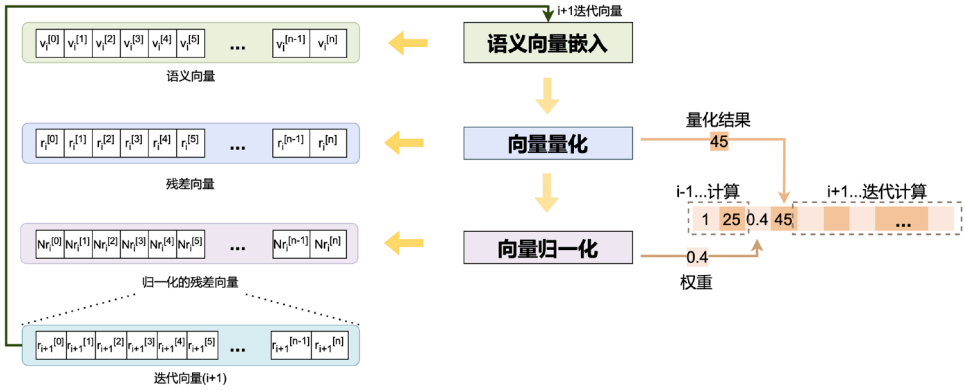


图 2　权重优化的向量量化表征

在 WRVQ 量化管理下，每一个码字均维护了一个半精度浮点数和一个码字表示，当假设原向量维度为 d，码字长度为 n，码本大小为 s。WRVQ 的量化比如公式（9）所示。新方法引入对残差向量的模长（即公式中的“权重”）进行建模，实现了比传统 RVQ 更高的量化保真度。我们不仅考虑了方向（由码字选择决定），也兼顾了长度信息。在相同码字长度下，WRVQ 的量化比约为 RVQ 的 1/3 左右，但仍然可以将高维向

量在损失极低的情况下压缩成极短的码字表示。

$$c_{wrvq} = \frac{d}{n \cdot (16 + \log_2 s)} \tag{9}$$

3 基于权重量化表征的索引构建与检索

3.1 非对称的向量距离计算

我们在向量嵌入和量化编码的距离计算中，采用了非对称距离计算（Asymmetric Distance Computation，ADC）方法，它与对称距离计算（SDC）有所不同，主要特点是不直接在查询向量上进行量化操作，而是计算查询向量与已量化编码之间的距离。在需要更高准确率的近似最近邻搜索（ANNS）场景中，我们选择采用 ADC 算法，确保在大规模数据集中的检索性能和准确度。

$$SDC: d(x,y) \approx d(q(x), q(y)) \tag{10}$$

$$ADC: d(x,y) \approx d(q(x), y) \tag{11}$$

RVQ 和 PQ 在距离计算中的主要区别体现在内积（ip）和 L2 距离的计算方法上。对于 PQ 的量化结果，可以通过分别计算各个量化向量的距离，并将其简单地组合起来，就可以快速获得量化编码与查询向量之间的距离。由于 PQ 采用了乘积量化的方式，计算过程相对简便且高效。

而在 RVQ 的情况下，虽然可以通过单独计算各量化向量并求和的方法快速获得内积（ip）距离，但计算 L2 距离时，RVQ 表现出一定的不足。这个问题的根源在于，RVQ 在处理多个量化器的情况下，无法像 PQ 那样直接分解和计算 L2 距离。幸运的是，通过引入一个简单的距离计算公式，能够有效地解决这一问题，从而增强 RVQ 在距离计算方面的能力，使其能够在 L2 距离计算中表现得更加准确和高效。这一公式在计算过程中结合了 RVQ 的残差信息，从而弥补了传统方法的不足。

在 RVQ 的背景下，该过程涉及单独计算各个码字的距离，然后将它们相加以确定总距离。尽管 ip 距离的计算很快，但 L2 距离需要额外的步骤。尽管如此，简单的距离计算公式的应用可以解决这个限制。

$$d(x,y) = \| x - y \|^2 = \| x \|^2 + \| y \|^2 - 2x \cdot y \tag{12}$$

对于 RVQ，上述公式变换为

$$\begin{aligned} d(q(x), y) &= \| q(x) \|^2 + \| y \|^2 - 2q(x) \cdot y \\ &= \| q(x) \|^2 + \| y \|^2 - 2\sum_{i=1}^{n} c_i \cdot y \end{aligned} \tag{13}$$

对于 WRVQ，上述公式变换为

$$d(q(x), y) = \| q(x) \|^2 + \| y \|^2 - 2\sum_{i=1}^{n} c_i \cdot y \cdot \prod_{j=0}^{i} d_j \tag{14}$$

在这个公式中，我们可以观察到第一项可以在索引构建过程中预先计算，而第二项在搜索过程中每个查询向量只需要计算一次。这使得我们能够以与计算 ip 距离相同的时间复杂度计算 L2 距离。

在解决了搜索过程中距离计算的基本问题之后，我们还发现，相较于 PQ，RVQ 具有一个显著优势，即其码字并不完全相同。即便在 OPQ 中，由于码字对应的维度不同，导致码字之间并不完全相等，它们仍然保持着一定的平行关系。然而，RVQ 在这方面表现得更为独特。RVQ 的码字生成过程揭示了一个重要特性：每个后续的码字本质上是对前一个码字的“补充”，它们在编码过程中形成了一个递增精度的序列。随着码字数量的增加，量化精度不断提高。这一特点使得 RVQ 在索引构建中具有独特优势，特别是在处理大规模数

据时，能够更有效地提升查询精度。

3.2 量化索引构建

在索引构建过程中，我们假设，对于任何 $k < n$ ，前 k 个码字可以作为簇中心，而剩下的 $n - k$ 个码字则表示相对于该簇中心的残差。这一假设是我们将向量量化步骤与搜索过程分开，并将其视为独立的表示模型的关键原因。

基于这一特性，我们开发了一种基于 WRVQ 的近似最近邻搜索（ANN）算法。该索引将整个向量空间划分为 k 个区域，每个向量由其区域中心向量和残差表示。在 WRVQ 量化编码的表示中，第一个码字表示簇中心，而后续码字则表示残差。与传统的 IVFPQ 算法不同，在 WRVQ 中，表示残差和原始码字的码字本质上是相同的。这一特点使得可以使用倒排索引进一步划分残差，同时，划分后的残差表示也可以继续细分。理想情况下，当码本训练得足够充分时，我们可以基于 n 个码本构建一个 n 层的树结构索引。

然而，这种理想状态在实际的码本训练中难以达到。在实验中，如果将所有的码本都用来构建树结构索引，那么搜索的性能和效率往往无法达到最佳。除了码本的训练可能无法将所有向量正确地划分到应有的子空间外，数据集的规模也会导致在建立第二级或第三级节点时产生许多空节点。这不仅会在索引构建过程中增加不必要的开销，还会使得在第三级之后的分类搜索失去意义。例如，在十个向量中搜索最近邻时，根本没有必要在八个类别内进行分类搜索。

因此，在达到指定的级别后，我们构建的索引会过渡到使用替代搜索算法。在这项工作中，我们直接采用了 topk 搜索算法，以提高搜索的性能和效率。

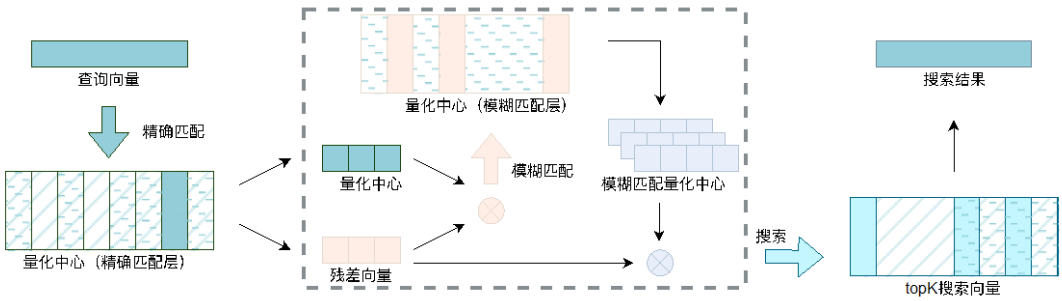


图 3　基于权重向量量化的索引结构

我们将整个索引结构划分为三个层次，如图 3 所示：精确匹配层、模糊匹配层和搜索层。当查询向量进入索引时，它首先经过精确匹配层。该层在本工作中由第一个码字组成。我们采用一个可训练的变换矩阵，它与码本共同优化，并在层间传递时调整残差，以更好地匹配下一层码本，并校正上一层的量化误差。这一设置确保了对于相同的码字，其对应的实际嵌入始终一致。基于第一个码字，我们构建了一个倒排索引，倒排索引根据输入查询向量与簇中心的匹配度来选择最接近的簇中心。这一步骤旨在缩小搜索范围，从而提高后续检索的效率，并确保嵌入计算过程中无失真。

3.3 检索实现

当查询向量通过精确匹配层后，我们获得了倒排索引提供的结果——第一层的簇中心和残差向量。随后，残差向量经过模糊匹配层。与模糊匹配层对应的码字是不确定的，取决于数据集的大小和码本的大小。对于较大的码本或较小的数据集，模糊匹配层可能只对应一个码字，甚至可能没有码字，从而在某些情况下取消这一层的作用。

对于第一层的每个簇中心，模糊匹配层构建一个倒排索引，通过输入的残差向量查询第二层的簇中心。挑战在于，对于 WRVQ，从第二个码字开始，权重是不一致的。这意味着，如果严格计算，倒排索引构建的

索引长度不是码本的长度，而是嵌入数据的长度。因此，倒排索引可能会退化为暴力搜索，尽管这种方法能提高搜索的准确率，但其效率会显著下降。

为了解决这一问题，在索引构建过程中，对于与模糊匹配层相对应的所有码字的权重，我们计算一个近似值来替换原始的码字权重。这种近似计算新的残差向量的方法并非绝对准确，但模糊匹配层的主要目的是进一步缩小搜索范围。从几何意义上讲，经过第一层计算后，查询向量对其残差的方向范围进行了潜在搜索。为了提高准确性，模糊匹配层将若干簇中心传递给搜索层。

在第三层，基于精确匹配层的残差向量进行距离计算，并从模糊匹配层获得目标向量潜在的多个范围子空间。通过这些子空间，搜索层能找到查询向量的近似最近邻向量。整体的算法过程如算法 1 所示：

算法 1 权重量化表征的索引搜索算法

```
输入：查询向量 q
输出：近似最近邻向量：v
    //精确匹配层
    for each c in codebook[:1] do
        central[i] = WRVQ(c) //记录每个码字对应量化向量
        distance[i] = l2distance(central[i],q) //记录查询向量与每个量化向量的距离
    end for
    code1 = findmin(distance) //搜索与查询向量距离最短的簇中心码字
    residual = q - WRVQ(code1)//计算第一层残差
    //模糊匹配层
    r = residual * d//将残差乘以权重，该权重是由索引建立时，根据检索精度和空间消耗权衡设定参数
    for each c in codebook[1:n+1] do //从模糊匹配层对应的 n 个码本中遍历所有组合
        central[i] = WRVQ(c)//记录每个组合对应量化向量
        distance[i] = l2distance(central[i],r)//记录残差与每个量化向量的距离
    end for
    code2=findmink(distance,k) //搜索与残差距离最短 k 个量化向量
    code = [ (code1,n) for each n in code2]//计算潜在查询区域
    //搜索层
    for each c in dataindex do //遍历数据索引
        if c[:n+1] in (code1,code2) then
            central[i] = WRVQ(c)//计算真实残差
            distance[i] = l2distance(Central,q)//计算真实距离
        end if
    end for
    code = findmin(distance) //rerank 排序
    v = WRVQ(code)
    return v
```

从结构上我们可以看到，当去除了模糊匹配层时，算法的主体与传统倒排索引相似，只是通过 WRVQ 量化方法提升了准确率，从而带来了更高的准确性。然而，在实验中我们发现，模糊匹配层的加入巧妙地利用了 WRVQ 的序列特性，显著提高了搜索效率。尽管模糊匹配层通过近似的方式缩小了搜索范围，这不可避免地会导致一定的搜索准确性的降低，但在第三层的精确搜索中，仍能获得较高的召回率。

在时间效率上，WRVQ 索引的检索时间复杂度与基于 ivf 的索引结构时间复杂度近似，主要分为三个部分。精确匹配层在第一个码本中查找查询向量的最近邻簇中心，这一过程的时间复杂度是 $O(log\ N)$ ，其中 N 是第一个码本的大小。而模糊匹配层的检索方法类似，时间复杂度为 O(n*log N) 其中 n 代表模糊匹配层对应的码本个数。搜索层进行量化编码的遍历，对于每个搜索簇内的向量，需要与查询向量的量化表示进行比较，时间复杂度为 $O(k*N*d)$ 其中 k 表示搜索层对应的码本个数，d 表示向量维度大小，因此总体的时间复杂度如公式（15）所示：

$$O(\log N + n \cdot \log N + k \cdot N \cdot d) \tag{15}$$

与基于其他类型索引结构（如 HNSW）的方案相比，WRVQ 索引的树结构保留了索引构建时间短、索引添加和删除成本低的优势。通过这一结构，我们能够在保证合理准确率的前提下，显著提升搜索效率。

4 实验分析

4.1 数据集与实验设置

本文使用了通用测评库 MTEB 对我们所提出的量化表征算法进行效率和效果的综合评估。在性能方面，我们选择分类任务、检索任务、 重排序任务、语义相似度任务、文本挖掘任务、聚类任务，以及配对分类任务等七个任务。在效率方面，我们对比了时空效率和增量更新的性能差异。

表 1 表征能力评估数据集及衡量标准

任务	数据集	衡量标准
Classification	AmazonReviews MTOPDomain MTOPIntent	Accuracy
Reranking	AskUbuntuDup MindSmall SciDocs	MRR@10
STS	SICK-R STS12 STSBenchmark	Pearson
BittextMining	Bornholm Tatoeba	Accuracy
Clustering	Arxiv Biorxiv Medrxiv	v_measure
PairClassification	SprintDuplicateQ TwitterSemEval TwitterURLCorpus	Accuracy
Retrieval	ClimateFEVER DBPedia FiQA2018 HotpotQA	MRR@10

在对比方面选取方面，我们将 faiss 索引库中的 pqindex、ivfpqindex、hnswindex、Chroma 向量数据库索引作为参考基准，比较索引的召回率、构建效率与检索效率。对于 pqindex、ivfpqindex 两种基于量化的索引，我们对齐参数以保证索引与 WRVQindex 之间的空间复杂度保持一致。而为了探索与验证 WRVQ 量化表征对检索索引的提升，我们基于线性残差微调以及残差微调与无监督学习结合这两种码本学习方式对 WRVQ 量化表征进行了详细评测。

对于量化表征失真的测试任务，我们主要采用了 ARPVQM 方法组[35]中的 RVQ、PQ、AVQ 作为残差微调与无监督学习结合对比组的基准模型，以及 vector-quantize-pytorch 方法组[33,34]中的 RVQ、PQ 作为线性残差微调对比组的基准模型。在这一任务中，我们使用 Alpaca_data 与 MS MARCO 数据集作为原始数据集，在经过基础嵌入模型 bge-large-en-v1.5 后形成嵌入向量，作为量化模型的输入。之后对量化模型的输出与其输入计算量化损失进行比较。

对于线性残差微调这一码本学习方式，我们总是将全量数据集作为训练集进行训练，而由于残差微调与无监督学习结合这一码本方式，为了检测我们方法对向量学习能力上的优化，我们对数据集进行了不同大小的抽样作为训练集，并总是将训练轮次设置为 1000，这使得码本训练的时间消耗被限制在了一定范围。但与此同时，对于少数收敛较慢的情况，码本并未被训练足够充分，对量化器的性能产生了一定的影响。故而在这一测试任务中，除了量化器的量化性能，其学习效率（即收敛能力）同样也是纳入考量的指标因素。在这一部分，我们使用学习率为 0.001 和默认参数的 Adam 优化器来训练模型以保证学习效率的相对统一，当码本参数达到 4（码本数量）*256（码本存储的向量数）后，我们使用学习率为 0.01 和默认参数的 Adam 优化器的附加实验进行对比。同时，每进行 100 次迭代，模型会通过线性残差微调对码本进行修改。

本文所使用的实验平台为 AMD EPYC 7R32 48-Core Processor，主频为 3.00GHz，内存为 512G。

4.2 检索索引的实验结果与分析

我们首先将基于残差微调与无监督学习结合码本学习方式的 WRVQ（T）与基于线性残差微调码本学习

方式的 WRVQ（L）构建索引体系，与其他索引进行比较，准确度、索引构建时间与检索效率如表 2 所示：

表 2 不同向量索引方法的查询准确率评测

数据集	索引方法	召回率（Recall@1）	召回率（Recall@5）	召回率（Recall@10）
Alpaca	pqindex(milvus)	75.75%	79.94%	80.25%
	ivfpqindex(milvus)	94.30%	94.33%	94.39%
	hnswindex(milvus)	95.71%	96.59%	**97.10%**
	Chroma	94.23%	94.30%	94.40%
	WRVQ(L)	95.65%	95.34%	94.26%
	WRVQ(T)	**96.64%**	**95.87%**	95.03%
MSMARCO	pqindex(milvus)	67.19%	70.13%	71.86%
	ivfpqindex(milvus)	/	/	/
	hnswindex(milvus)	90.23%	**90.98%**	**91.02%**
	Chroma	90.11%	90.39%	90.56%
	WRVQ(L)	84.52%	83.65%	83.02%
	WRVQ(T)	**90.97%**	89.02%	88.87%

在这一测试中，pqindex、ivfpqindex 两种基于量化的索引与 WRVQ 的两种索引均将码本设置为 4（码本数量）*256（码本存储的向量数），其余设置为默认设置。而其他几项索引均设置为默认设置进行比较。

首先，我们观察 Alpaca 数据集上的检索效果，在 Recall@1 和 Recall@5 指标上，基于 WRVQ（T）的效果均优于其他搜索索引，在 Recall@10 中，也仅 hnsw 算法获得了更优的召回率。在表中我们可以发现，基于 WRVQ 的索引在三个召回率指标的得分是接近的，在搜索索引的近似搜索环节中，我们使用了 top-k 算法对候选结果进行重排序，使得检索结果的较为接近。其次我们观察数据量更大的 MSMARCO 数据集的检索效果，各种算法的准确率获得了大幅度下降。在 MSMARCO 数据集上的实验中我们并没有获得 ivfpq 的实验结果，这是因为实验环境制约下，ivfpq 索引并不能完整地被载入内存。相比之下，以量化表征为主体的 WRVQ 虽然也采取了相似的架构，但仍然很好的完成了构建和检索，并获得了较好的准确率，在 Recall@5 指标上的也得到了较好的表现。

我们进一步评测不同向量索引的构建效率与检索效率，如表 3 所示：

表 3 不同向量索引方法的构建和检索效率评测

数据集	索引方法	构建效率（s）	检索效率（ms/q）
Alpaca	pqindex(milvus)	152	5.31
	ivfpqindex(milvus)	**4**	0.27
	hnswindex(milvus)	42	**0.003**
	Chroma	844	0.84
	WRVQ(L)	5	0.29
	WRVQ(T)	5	0.29
MSMARCO	pqindex(milvus)	193	35.82
	ivfpqindex(milvus)	/	/
	hnswindex(milvus)	102	**0.003**
	Chroma	12507	20.6
	WRVQ(L)	**44**	5.16
	WRVQ(T)	**44**	5.16

首先我们对检索索引的构建效率进行评估，对于 alpaca 数据集，ivf 系列的索引构建效率占据了极大的优势，相比之下 pqindex、hnswindex 以及 chroma 使用了更长的时间构建索引。在实验中我们发现，WRVQ 以及 chroma 索引的构建时间与数据集大小更为相关，这意味着随着数据集大小指数级增大，索引的构建时间将会大幅降低，但与此同时，这种索引对已有知识库的小范围更新更加高效。与之相对的，其他算法的构建效率与数据集大小的相关性较小，这意味着这些索引可能并不适合频繁更新知识库的场景。而在检索效率方面，hnsw 算法获得了较大的优势，但我们的工作相比大多索引结构仍有一定的竞争力。而在检索效率上，由于同样使用了倒排索引作为主结构，我们的方法与 ivf 系列的方法的时间复杂度相同，因此在检索效率的表现上

也相似。

进一步，我们在不同索引方法的空间效率上进行了评测，这里的内存占用指的是峰值消耗。该值指在查询期间，加载完整索引及相关数据结构后，程序达到的最大内存用量。如表 4 所示：

表 4 不同向量索引方法的空间消耗评测

数据集	索引方法	内存峰值消耗（GB）
Alpaca	pqindex(milvus)	8.6
	ivfpqindex(milvus)	6.7
	hnswindex(milvus)	15.2
	Chroma	**5.4**
	WRVQ(L)	5.9
	WRVQ(T)	5.9
MSMARCO	pqindex(milvus)	12.3
	ivfpqindex(milvus)	/
	hnswindex(milvus)	22.7
	Chroma	**7.1**
	WRVQ(L)	9.9
	WRVQ(T)	9.9

可以看出，pqindex 和 WRVQ 方法的内存消耗受数据集规模的影响较大，这是由于这些方法均采用了量化和倒排结构作为核心索引机制。量化能够有效减少存储需求，但在处理更大规模数据集时，仍然会导致一定程度的内存增长；倒排索引的存储开销则会随着数据规模的增加而扩大，因此这些方法在存储成本上对数据集大小更为敏感。

相较之下，hnswindex 和 Chroma 的内存消耗对数据规模的变化相对不敏感。Chroma 在两个数据集上均展现出较低的存储需求，表明其索引结构在高效性与存储优化之间取得了良好平衡。hnswindex 由于基于邻接图结构进行搜索，其索引规模主要取决于邻接关系的复杂性，而非直接受数据集规模的线性增长影响。因此，尽管其存储成本变化较小，但 HNSW 方法本身的索引结构需要大量内存，导致其总体存储开销仍然较高。

然而，在百万级别数据集的场景下，我们提出的方法（WRVQ）仍然表现出较大的相对优势。WRVQ 在存储开销上接近 Chroma，且优于 hnswindex 和 ivfpqindex，表明其在大规模数据场景下具备更好的空间效率。因此，在需要权衡存储开销与检索性能的应用场景中，我们的方法相较于其他方法仍然具有竞争力。

4.3 量化方法的量化能力实验结果与分析

为了探究 WRVQ 对向量表征的量化能力，我们在两个数据集上对各个量化方法的量化损失进行了对比，图 4 展示了线性残差微调对比组在量化损失上的对比结果：

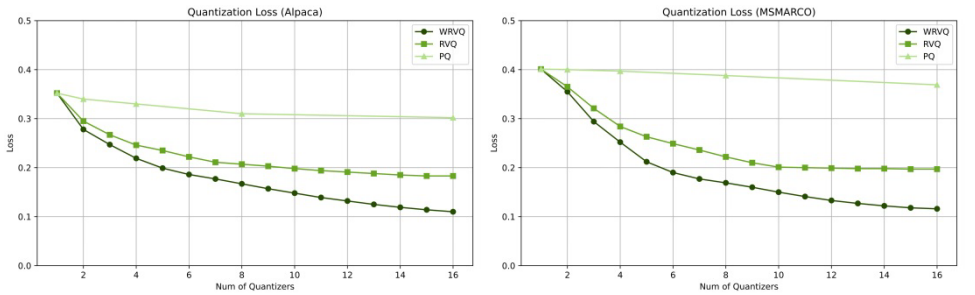


图 4 基于线性残差微调码本的量化器在两个数据集上的量化损失

在实验中我们对不同码本数量情况下的量化器进行测试，可以看出在线性残差微调作为码本训练方法时，pq 的表现并不良好随着码本数量的增加，即使在 8 和 16 两种码本选择的情况下，pq 的量化损失也仅有 0.03 左右的降低。相比之下，两种基于 RVQ 的方法获得了比较大的提升，但仍没有获得较好的结果，这受限于轻量级的码本训练策略。但值得一提的是，随着码本数量的增加，如预想的一样，传统 RVQ 对向量的

量化能力提升逐渐减弱，而 WRVQ 仍保持着不错的量化能力提升，在 15 码本到 16 码本的变化中，RVQ 的性能提升几乎忽略不计，但 WRVQ 仍然获得了 0.05 左右的提升。

轻量化的码本训练策略虽然使码本的训练效率大幅提升，即使在 16 码本数量的情况下，Alpaca 的数据集训练时间仅 17 秒，而 20 倍数据集大小的 MSMARCO 的训练时间也仅 223 秒，这使得这种训练方法在特定场景下具有不错的表现。

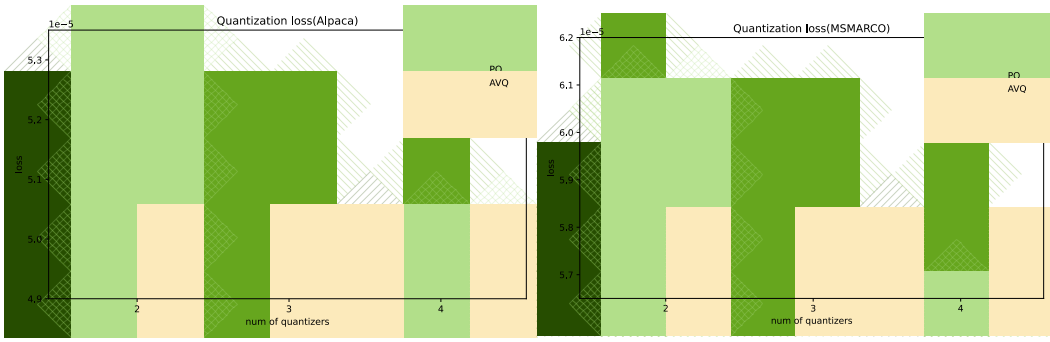
而对于残差微调与无监督学习结合对比组，我们同样在这两个数据集上进行了全面的比较。首先，我们随机生成一千条向量作为模拟向量构成训练集进行训练，之后将训练好的模型在两个数据集上进行评估。在这一部分，单个码本的大小被设置成 64。结果如表 4 所示：

表 4 各量化方法的通用量化器在两个数据集上的量化损失

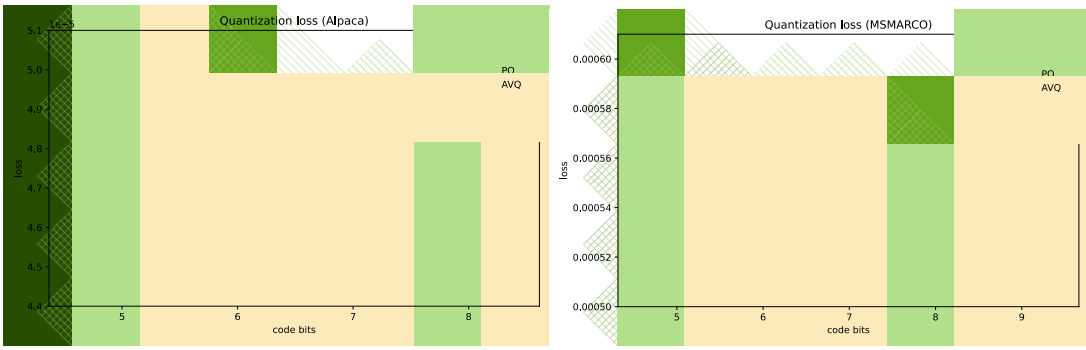
数据集	量化方法	量化损失		
		码本数量：2	码本数量：3	码本数量：4
Alpaca	PQ	0.0171	/	0.0091
	AVQ	0.0163	0.0126	0.0089
	RVQ	0.0169	0.0131	0.0092
	WRVQ	**0.0009**	**0.0009**	**0.0009**
MSMARCO	PQ	0.0172	/	0.0092
	AVQ	0.0164	0.0127	0.0090
	RVQ	0.0170	0.0131	0.0092
	WRVQ	**0.0009**	**0.0009**	**0.0009**

从表格中我们可以观察到，WRVQ 在随机数据训练的情况下，对各个数据集的量化能力达到了非常高的水平，比起结构相同的 RVQ 以及其他类型量化器，其量化损失降低到了十分之一。得益于 WRVQ 对码本的归一化调整，其在低码本的情况下仍能发挥出不错的效果，而相对而言，其他类型量化器都没有达到较好的效果。总体而言可以发现，残差微调与无监督学习结合对比组比起线性残差微调对比组在量化能力上有了极大幅度的提升。

之后，我们分别对两个数据集进行随机抽样，各选取了 1000 条数据作为训练集在不同码本参数的情况下对量化器进行训练，并对两个数据集进行量化。结果如图 5 所示：



（a）



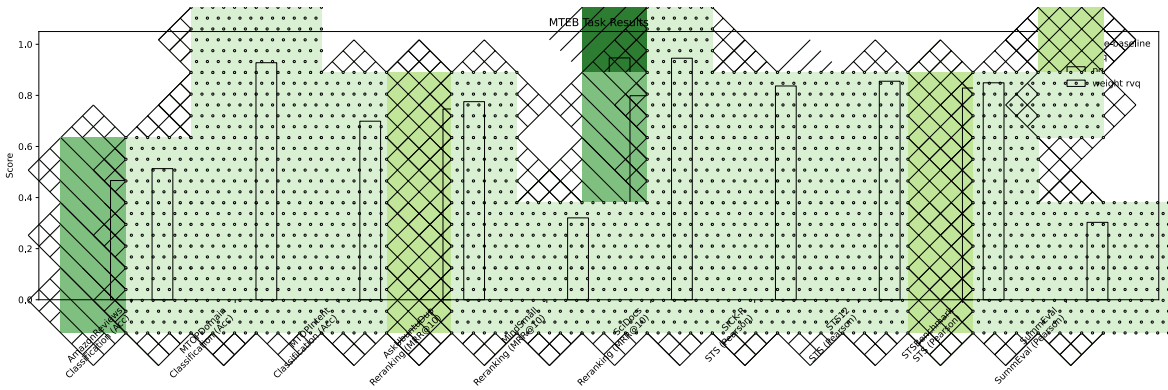
（b）

图 5　不同码本大小的抽样训练量化器在两个数据集上的量化损失

在图 5（a）中，我们对比了四种量化器在相同码本大小（32）下，不同码本数量在两种数据集上的性能表现，发现 wrvq 相对来说获得了更好的表现。但在实验过程中我们发现，码本训练的收敛速度较快，这是因为码本大小过小导致各个量化器的性能没有被完全释放，这导致最终结果差距不大。在图 6（b）的实验中，我们冻结了码本数量参数为 4，通过调整码字的大小对码本大小进行调整，评估了在不同码本大小（2^5-2^9）下各量化器的性能，可以发现 WRVQ 相比同一码本训练方式的其他量化器获得了更多的提升。在 Alpaca 数据集的训练过程中，这种提升在 256 码本大小下出现了消弱，说明训练集已经得到了充分的学习，而在 MSMARCO 数据集中，提升在 512 码本大小下放缓。最终在相同情况下，WRVQ 在充分学习训练集时获得了 10%到 20%的量化损失减少，能够更加准确地表示向量表征。

4.4　量化表征的表征能力实验结果与分析

这一部分实验中，我们使用了外部测评库对几种向量表征进行了测评，评估量化表征的表征能力。



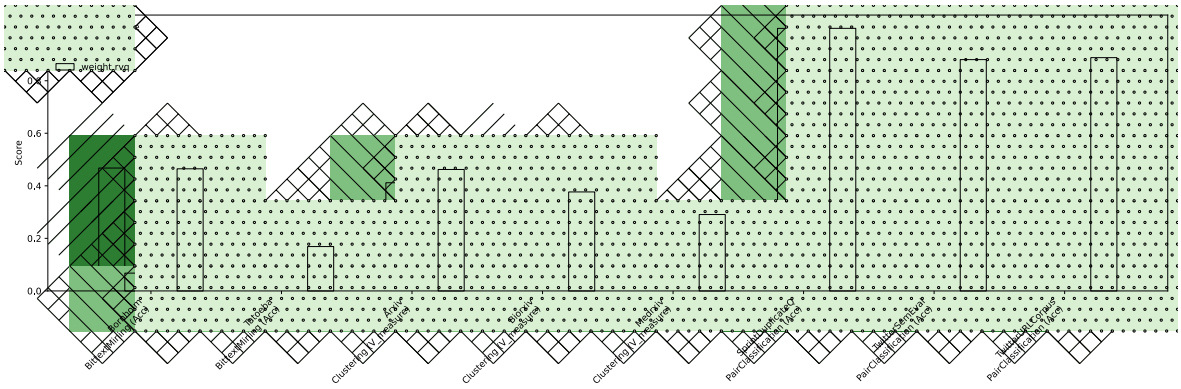


图 6　各量化表征在 MTEB 上的表征能力测试

测试结果的内容在图 6 中进行了展示。从测试中可以明显看出，我们的表示方法在所有七项任务中都取得了接近基准模型的结果。我们观察到，在分类、重新排序、聚类和对分类等任务中，量化表示并未显著影响性能——在某些情况下，量化表示甚至达到或超过原始表征模型的性能。然而，在其余三项任务（检索、STS 和 BittextMining）中，量化原始模型可能会导致表示性能的大幅下降。在这些任务中，我们的量化方法明显优于其他量化方法。

4.5 量化表征增量更新实验结果与分析

为进一步评估 WRVQ 方法在实际应用中的索引构建能力，我们对其在增量索引构建场景下的性能表现进行了系统实验。具体而言，我们采用不同的 batch size 对 WRVQ 索引进行批量增量插入，记录每个批次操作对应的平均构建时间及平均 CUDA 内存消耗。图中横轴为每批插入的向量数，左纵轴为批次构建耗时（秒），右纵轴为显存占用变化（MB）。为便于展示索引增量带来的性能变化，结果以相邻批次增量绝对值形式展示。

如图 7 中实验结果显示，随着批量插入规模的提升，WRVQ 方法的索引增量操作在时间和空间开销上均表现出良好的线性可扩展性，无明显的性能突变或瓶颈，验证了 WRVQ 对增量索引构建的良好适应性。

这进一步证明了该方法不仅适用于全量索引的初始构建，也能够有效支持大规模向量数据的在线扩展与动态更新需求。我们在原型系统中也展示了这一量化表征方法对更大规模数据的处理能力。

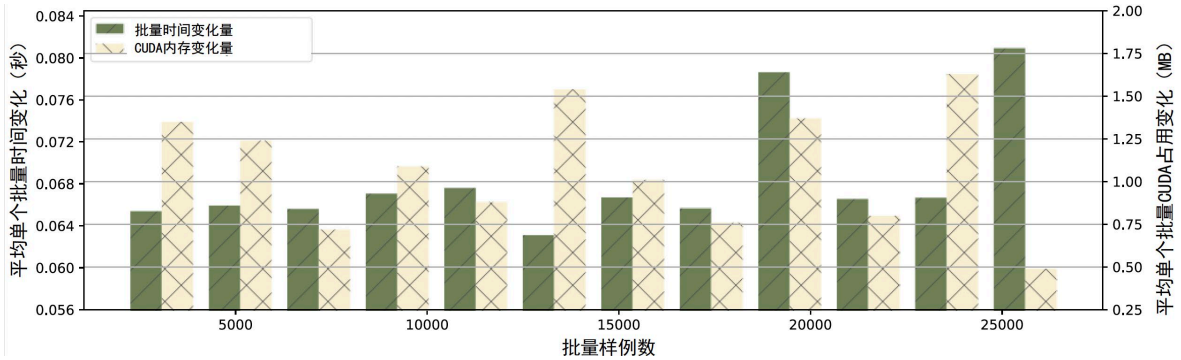


图 7　不同量化向量表征方法在增量索引的时空开销

5 总　结

本研究旨在解决大规模向量数据的索引和查询挑战时所面临的维度和表征问题[38]。本文提出一种新颖的量化表征方法——权重残差向量量化（WRVQ），我们平衡了检索的判别能力与存储与计算的时空效率。相较于近期的量化向量表征方式，新方法具有更高的存储效率和检索性能。

我们的研究结果表明，在向量数据库的数据存取和知识查询等场景中，量化表征是一种有效的解决方案，有望为处理大规模嵌入表示的向量数据提供更可靠的表示、存储和检索途径。

我们通过构建面向生成式场景的多模态音视频实时交互等应用场景[39,40]，进一步验证了量化表征的性能优势，并在大规模文本和多模态等数据集上验证了量化表征的有效性。

References:

[1] Shitao Xiao, Zheng Liu, Weihao Han, Jianjin Zhang, Yingxia Shao, etc. Progressively Optimized Bi-Granular Document Representation for Scalable Embedding Based Retrieval. Proc. of WWW. 2022. 286 – 296.

[2] Zhang, Qianxi, et al. VBASE: Unifying Online Vector Similarity Search and Relational Queries via Relaxed Monotonicity. Proc. of OSDI. 2023.

[3] Tiezheng Ge, Kaiming He, Qifa Ke, and Jian Sun. Optimized Product Quantization. PAMI. 2014. 744 – 755.

[4] Mikolov, Tomas, Kai Chen, and etc. Efficient Estimation of Word Representations in Vector Space. Proc. of ICLR. 2013.

[5] Jeffrey Pennington, Richard Socher, Christopher Manning. GloVe: Global Vectors for Word Representation. Proc. of EMNLP. 2014. 1532 – 1543.

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, and etc. Attention is All You Need. Proc. of NeurIPS. 2017. 6000 – 6010.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Proc. of NAACL. 2019. 4171 – 4186.

[8] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. Proc. of EMNLP. 2017. 670 – 680.

[9] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. Proc. of EMNLP. 2019. 3982 – 3992.

[10] Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. MTEB: Massive Text Embedding Benchmark. arXiv:2210.07316, 2022.

[11] Nils Reimers, Iryna Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. Proc. of EMNLP, 2019, pp. 3982 – 3992.

[12] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, et al. Language Models are Few-Shot Learners. Proc. of NeurIPS, Vol. 33, 2020, pp. 1877 – 1901.

[13] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, et al. Learning Transferable Visual Models From Natural Language Supervision. Proc. of ICML, 2021.

[14] Jie Tao, Lei Zhang, Ying Li, Zhi Chen. Optimizing LLM Embeddings via Prompting and Data-Driven Tuning. Proc. of ACL, 2024, pp. 1234 – 1245.

[15] Shitao Xiao, Zheng Liu, Peitian Zhang and Niklas Muennighoff. C-Pack: Packaged Resources To Advance General Chinese Embedding. arXiv:2309.07597, 2023.

[16] Yang, Wen, Tao Li, Gai Fang and Hong Wei. PASE: PostgreSQL Ultra-High-Dimensional Approximate Nearest Neighbor Search Extension. Proc. of SIGMOD. 2020.

[17] Chuangxian Wei, Bin Wu, Sheng Wang, Renjie Lou, Chaoqun Zhan, Feifei Li, and Yuanzhe Cai. AnalyticDB-V: a Hybrid Analytical Engine towards Query Fusion for Structured and Unstructured Data. Proc. of VLDB. 2020, 13, 12. 3152 – 3165.

[18] Jeff Johnson, Matthijs Douze, and Herve Jegou. Billion-scale Similarity Search with GPUs. Trans.on Big Data. 2019. 535 – 547

[19] Jianguo Wang, Xiaomeng Yi, Rentong Guo, Hai Jin, Peng Xu,etc. Milvus: A Purpose-Built Vector Data Management System. Proc. of SIGMOD. 2021. 2614 – 2627.

[20] chroma. [Online]. Available: https://www.trychroma.com

[21] M. Stonebraker and U. Cetintemel. "One size fits all": An Idea whose Time has Come and Gone. Proc. of ICDE. 2005. 2 – 11.

[22] Dittrich J. and Jindal A. Towards a One Size Fits All Database Architecture. Proc. of CIDR. 2011.

[23] Zhang, Yunan et al. Are There Fundamental Limitations in Supporting Vector Data Management in Relational Databases? A Case Study of PostgreSQL. Proc. of ICDE. 2024.

[24] J. S. Beis and D. G. Lowe. Shape Indexing using Approximate Nearest-neighbour Search in High-dimensional Spaces. Proc. of CVPR. 1997. 1000 – 1006.

[25] Muja M, Lowe D G. Fast approximate nearest neighbors with automatic algorithm configuration. VISAPP (1). 2009. 2(331-340): 2.

[26] Jégou H, Tavenard R, Douze M, et al. Searching in one billion vectors: re-rank with source coding. Proc. of ICASSP. 2011.

[27] Herve J´egou, Matthijs Douze, and Cordelia Schmid. Product Quantization for Nearest Neighbor Search. PAMI. 2011. 117－128.

[28] Malkov Y A, Yashunin D A. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. PAMI. 2018, 42(4). 824－836.

[29] Zhang H, Zhao P, Miao X, et al. Experimental analysis of large-scale learnable vector storage compression. Proc. of VLDB, Vol. 17, No. 4. 2024.

[30] Shitao Xiao, Zheng Liu, Weihao Han, Jianjin Zhang, Yingxia Shao, etc. Progressively Optimized Bi-Granular Document Representation for Scalable Embedding Based Retrieval. Proc. of WWW. 2022. 286－296.

[31] Haowen Dong, Chao Zhang, Guoliang Li, Jianhua Feng. A Survey on Cloud-Native Databases. Journal of Software. 2024, 35(2): 899–926.

[32] A. Babenko and V. Lempitsky. Additive Quantization for Extreme Vector Compression. Proc. of CVPR. 2014. 931－938.

[33] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund and Marco Tagliasacchi. SoundStream: An End-to-End Neural Audio Codec. arXiv:2107.03312, 2021.

[34] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho and Wook-Shin Han. Autoregressive Image Generation Using Residual Quantization. Proc. of CVPR. 2022. 11523－11532.

[35] Mohammad Hassan Vali and Tom Bäckström. Stochastic Optimization of Vector Quantization Methods in Application to Speech and Image Processing. Proc. of ICASSP. 2023.

[36] Yongjian Chen, Tao Guan, and Cheng Wang. Approximate Nearest Neighbor Search by Residual Vector Quantization. Sensors, 10(12). 2010. 11259－11273.

[37] Bolong Zheng, Ziyang Yue, Qi Hu, Xiaomeng Yi, Xiaofan Luan,etc. Learned Probing Cardinality Estimation for High-Dimensional Approximate NN Search. Proc. of ICDE. 2023. 3209－3221.

[38] Bo Tang, Jianbin Qin, Rui Mao. Vector Databases: Key Technologies, System Architecture, and Future Challenges. Communications of the China Computer Federation. 2023, 19(11).

[39] Yongkang Zhou, Muyang Yan, Junjie Yao, Gang Xu. ProRAG: Towards Reliable and Proficient AIGC-Based Digital Avatar, In Proc. of DASFAA 2025.

[40] Xiu Tang, Sai Wu, Jie Hou, Gang Chen. Efficient Sample Retrieval Techniques for Multimodal Model Training. Journal of Software. 2024, 35(3): 1125–1139.

附中文参考文献:

[31] 董昊文, 张超, 李国良, 冯建华.云原生数据库综述[J].软件学报,2024,35(2): 899－926.

[38] 唐秀, 伍赛, 侯捷, 陈刚. 面向多模态模型训练的高效样本检索技术. 软件学报, 2024, 35(3): 1125－1139.

[40] 唐博，秦建斌，毛睿. 向量数据库：关键技术、系统架构与未来挑战. 中国计算机学会通讯. 2023, 19(11).