MEMORY-EFFICIENT MULTILINGUAL EMBEDDINGS WITH A DIFFUSION-LM BACKBONE

Sedigheh Eslami* **Maksim Gaiduk *** **Markus Krimmel***

Louis Milliken* **Bo Wang*** **Denis Bykov**

Perplexity AI

ABSTRACT

Dense textual embeddings are essential for web-scale search and retrieval-augmented generation, but their high memory and storage costs limit deployment across billions of documents. We introduce pplx-embed, a family of multilingual text embedding models that employs native quantization-aware training (QAT) throughout its multi-stage contrastive training pipeline, producing INT8 embeddings by default that achieve competitive retrieval performance compared to full-precision results while offering 4x memory and storage reduction. Furthermore, leveraging diffusion-based language models with bidirectional attention, pplx-embed family captures global document context more comprehensively than causal autoregressive models, enabling superior performance in document-level retrieval tasks. We release two model variants: pplx-embed-v1 for standard retrieval, and pplx-embed-context-v1 for contextualized embeddings that incorporate global document context into passage representations. Our extensive evaluations on public and internal benchmarks demonstrate the effectiveness of our quantization-aware training in achieving memory and storage efficient first-stage retrievers without more than minimal losses in retrieval quality. Our internal evaluation suite focuses on real-world, large-scale search scenarios constructed from 1B production web pages. We publicly release the pretrained pplx-embed-v1 and pplx-embed-context-v1 model weights to facilitate further research on memory-efficient multilingual retrieval.

1 INTRODUCTION

Dense textual embeddings represent texts as points in a vector space where distances reflect semantic similarity, forming the core of modern search systems based on approximate nearest neighbor retrieval. As high-dimensional embeddings are deployed across billions of documents, memory and storage costs become a critical bottleneck for web-scale retrieval. These same constraints arise in retrieval-augmented generation (RAG) where vector databases store document embeddings to retrieve relevant context for LLMs. Quantization addresses this by reducing both raw embedding storage and vector-database index sizes: theoretically, INT8 and binary encodings shrink both memory and persistent storage requirements by factors of 4× and 32×, respectively, allowing larger fractions of the corpus to reside in memory or on disk in compressed form and thereby, improving latency and throughput under realistic cost and hardware constraints (Huerga-Pérez et al., 2025).

This work introduces pplx-embed-v1, a family of multilingual text embedding models that incorporates quantization directly into the contrastive training pipeline and produces intrinsically-optimized low-precision INT8 embeddings by default. This approach achieves 4× storage reduction while obtaining competitive performance compared to the full-precision retrieval results of Qwen3-Embedding and Gemini Embedding, alleviating the accuracy degradation typically associated with post-training quantization (Huerga-Pérez et al., 2025). Our models are trained to encode both passages and full documents, with pplx-embed-context-v1 specifically designed for contextual encoding with respect to the global document context.

*Equal contributions

Additionally, the recent development of large language foundation models has increasingly shifted embedding model training towards employing pre-trained decoder-only LLMs to leverage their pre-existing knowledge and improve embedding quality (Zhang et al., 2025b; Jiang et al., 2024; Lee et al., 2025b). Our work investigates diffusion-based language modelsas an alternative paradigm in which transformer encoders with bidirectional attention are employed, enabling more comprehensive context modeling compared to the causal masking of autoregressive models (Austin et al., 2021; Nie et al., 2025b; Zhang et al., 2025a). This architectural difference is particularly advantageous for retrieval tasks, where capturing the global document context is essential. Notably, our models are not instruction-tuned, eliminating the need for users to maintain instruction prefixes while still delivering strong performance.

On MTEB(Multilingual, v2) (Enevoldsen et al., 2025; Muennighoff et al., 2023), pplx-embed-v1-4B achieves an average nDCG@10 of 69.66% with INT8 quantization, matching or exceeding leading models such as Qwen3-Embedding 4B and Gemini Embedding. Our models also demonstrate competitive performance on MTEB(Code) and MIRACL benchmark (Zhang et al., 2023) datasets. On ConTEB (Conti et al., 2025), pplx-embed-context-v1-4B outperforms state-of-the-art contextual models such as voyage-context-3[1] and Anthropic Contextual[2] with an average nDCG@10 of 81.96%. On the ToolSearch benchmark (Shi et al., 2025), pplx-embed-v1-4B achieves 44.45% average nDCG@10, surpassing larger 7B models including NV-Embed-v1 (Lee et al., 2025a) and GritLM-7B (Muennighoff et al., 2025), offering efficient pre-filtering of relevant tools from large API corpora with reduced context usage. We further demonstrate that pplx-embed-v1 consistently outperforms BGE-M3 (Chen et al., 2024) and Qwen3-Embedding on the large-scale BERGEN (Rau et al., 2024) RAG benchmark, with pplx-embed-v1-0.6B beating the larger Qwen3-Embedding-4B model on three of the five Question Answering tasks. Additionally, we describe internally-curated benchmarks for assessing embedding models as first-stage retrievers at web-scale and show superior performance of pplx-embed-v1 in comparison with Qwen3-Embedding (Zhang et al., 2025b) and BGE-M3 (Chen et al., 2024). Our experiments indicate that pplx-embed-v1 models are well-suited as first-stage retrievers in realistic deployment scenarios, demonstrating that substantial memory and storage savings can be achieved without compromising retrieval quality.

Related Work. Contrastive learning is the dominant paradigm for training text embeddings (Reimers & Gurevych, 2019; Gao et al., 2021; Izacard et al., 2021; Neelakantan et al., 2022; Santhanam et al., 2022). Recent work has integrated quantization into this process, from Contrastive Quant in computer vision (Fu et al., 2022) to EmbeddingGemma's quantization-aware finetuning for weight-quantized variants (Vera et al., 2025). Moreover, Huerga-Pérez et al. (2025) perform systematic evaluations of post-training quantization for RAG-based use cases. Our work, in contrast, targets quantization-aware training of embeddings for web-scale retrieval.

Diffusion language models have been proposed as an alternative to autoregressive LMs (Austin et al., 2021; Nie et al., 2025b; Gong et al., 2025), and a first systematic study of diffusion-based text embeddings highlights the importance of bidirectional attention for long-document context (Zhang et al., 2025a). We build on this work with continued pre-training of diffusion-LM as the backbone for training embeddings, extending masked-LM-style finetuning for retrieval-oriented BERT models (Devlin et al., 2019; Reimers & Gurevych, 2019; Günther et al., 2023; Chen et al., 2024).

Related work on contextual embeddings ranges from training document representations with respect to neighbors (Morris & Rush, 2025), chunk-level in-sequence training in ConTEB (Conti et al., 2025) to training-free late-chunking in (Günther et al., 2024). ConTEB also proposes a benchmark for context-aware evaluation (Conti et al., 2025). Our work builds upon these foundations by employing multi-stage contrastive training with a specialized dual-loss objective.

2 PPLX EMBEDDING

Our pipeline for learning memory-efficient embeddings combines continued diffusion pretraining, pair training, contextual training, and triplet training, maintaining INT8 quantization throughout all contrastive stages (Figure 1). The final pplx-embed-v1 model is obtained via Spherical Linear Interpolation (SLERP) (Shoemake, 1985) of the contextual and triplet checkpoints.

[1] https://blog.voyageai.com/2025/07/23/voyage-context-3/

[2] https://www.anthropic.com/engineering/contextual-retrieval

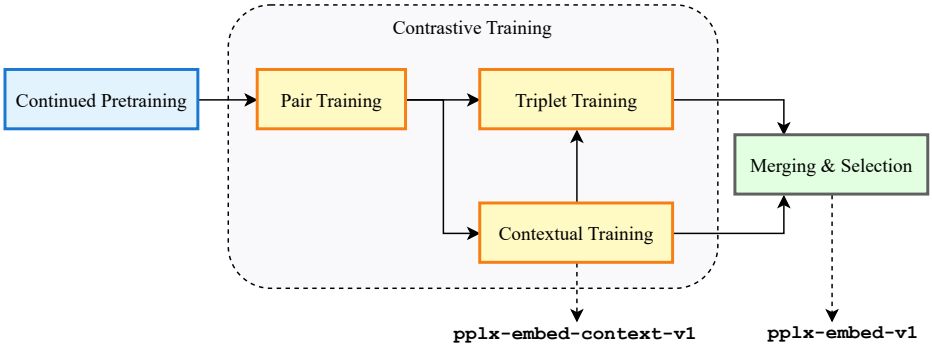

Figure 1: Training pipeline of pplx-embed-context-v1 and pplx-embed-v1.

Continued Diffusion Pretraining At the first stage, we train two bidirectional diffusion language models via continued pretraining of existing autoregressive decoder-only language models following the methodology of Gong et al. (2025). Considering the state-of-the-art performance of the Qwen3 family (Yang et al., 2025), we choose Qwen3-0.6B[3] and 4B[4] as our base models. We disable causal attention masking and train the resulting transformer encoders to reverse a corrupting noise process. We adopt a continuous-time formulation (Shi et al., 2024) and an absorbing state process in which, at timestep $t \in [0, 1]$, each token has decayed to the absorbing `[MASK]` state independently with probability t. To represent the `[MASK]` state, we re-purpose a rarely used token from the Qwen3 vocabulary. Following previous works (Gong et al., 2025; Ye et al., 2025; Nie et al., 2025b), we preserve the left-shift operation that is applied during autoregressive pretraining. While we also performed experiments with annealing the causal attention mask (Gong et al., 2025), we did not observe substantial performance improvements and abandoned this technique. Details of the continued pre-training stage are described in App. A.1. We also present an ablation study in App. A.2 that evaluates the impact of the diffusion-LM backbone on loss optimization and downstream retrieval performance.

Pooling and Quantization. To produce embeddings, we pool token-level representations extracted from the backbone model into a sequence-level representation. While recent embedding models based on decoder-only transformers (Zhang et al., 2025b; Tang & Yang, 2024) typically employ last-token pooling, our bidirectional architecture allows the application of mean pooling. We propose a pooling method that natively combines mean pooling with INT8 quantization. App. A.2 provides an ablation study comparing the performance of mean pooling and last-token pooling for downstream retrieval tasks.

Given token embeddings $(\mathbf{v}_l)_{l=1}^{L} \in \mathbb{R}^{L \times d}$ for a sequence of length L and embedding dimension d, we define the quantized sequence-level embedding as:

$$\left\lfloor 127 \cdot \tanh\left(\frac{1}{L}\sum_{l=1}^{L}\mathbf{v}_l\right) + \frac{1}{2} \right\rfloor.$$

The resulting vector has integer entries in $\{-127, \ldots, 127\}$ which are representable as signed 8-bit integers. We employ the quantization above not only during inference, but also during all contrastive training stages. We use straight-through gradient estimation (STE) (Bengio et al., 2013) to backpropagate through the non-differentiable rounding operation. The quantized embeddings are compared via their cosine similarity at later stages of the training and at the inference time.

We also support binary quantization to further reduce the size of output embeddings, which rounds each entry of the mean-pooled embedding vector to -1 or 1:

$$\text{bin}(x) = \begin{cases} 1 & \text{if } x > 0, \\ -1 & \text{otherwise.} \end{cases}$$

[3] `https://huggingface.co/Qwen/Qwen3-0.6B-Base`
[4] `https://huggingface.co/Qwen/Qwen3-4B-Base`

While an embedding model could be similarly trained using binary quantization with STE, we found that training-free post-hoc binarization achieves the same performance with minimal loss.

Pair Training. Pair training represents the first contrastive learning stage, establishing foundational semantic alignment between queries and documents. The model learns to maximize the similarity of queries with their corresponding documents and minimize the similarity with unrelated ones. We employ an InfoNCE contrastive loss which contrasts queries simultaneously against in-batch documents and other in-batch queries. Given a set of N query-document pairs, we obtain corresponding INT8-quantized embedding vectors $\mathbf{q}_i \in \mathbb{R}^d$ and $\mathbf{d}_i \in \mathbb{R}^d$ from our encoder for $i = 1, \ldots, N$. For a temperature $\tau > 0$, the loss is defined as:

$$\mathcal{L}^{\text{pair}} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{s(\mathbf{q}_i,\mathbf{d}_i)/\tau}}{e^{s(\mathbf{q}_i,\mathbf{d}_i)/\tau} + \sum_{j\neq i}^{N} m_i(\mathbf{d}_j) e^{s(\mathbf{q}_i,\mathbf{d}_j)/\tau} + \sum_{j\neq i}^{N} m_i(\mathbf{q}_j) e^{s(\mathbf{q}_i,\mathbf{q}_j)/\tau}},$$

with $m_i(\mathbf{x}) = \mathbb{1}_{\{s(\mathbf{q_i},\mathbf{x}) \leq s(\mathbf{q_i},\mathbf{d_i})+0.1\}}$, and $s(\mathbf{q}_i, \mathbf{d}_i) = \frac{\mathbf{q}_i \cdot \mathbf{d}_i}{\|\mathbf{q}_i\|_2 \|\mathbf{d}_i\|_2}$ calculating the cosine similarity. Inspired by Zhang et al. (2025b), the masking function m mitigates the effects of false negative samples contributing in the loss optimization. It evaluates the similarity between the positive pair against each negative sample. When a negative sample exhibits similarity comparable to the positive, indicating potential semantic relevance and thus a likely false negative, the function masks its contribution, thereby preventing distortion of the learned representation space.

Contextual Training. Contextual training is an approach for training embedding models on chunked documents such that the embedding of each chunk retains contextual information from the whole document. Given N query-document pairs where each document contains C chunks, $d_i = \{c_{ik}\}_{k=1}^{C}$, we compute embedding vectors $\mathbf{c}_{ik} \in \mathbb{R}^d$ for the chunk k from the document i. We use a dual-objective loss function in order to capture local chunk-level semantics as well as global-level document representation. Inspired by Conti et al. (2025), we define the local loss as a combination of the in-batch and in-sequence losses for chunks. For in-sequence contrastive loss, the target (gold) chunk from a document is treated as the positive sample, and all remaining chunks from the same document are used as negatives. In contrast, for in-batch loss, the gold chunk remains the positive sample, but the negatives are defined as all other chunks in the batch, including those from the same document. Using mean-pooling and INT8 quantization to obtain the chunk-level embeddings, we define the sequence loss as:

$$\mathcal{L}^{\text{seq}} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{s(\mathbf{q}_i,\mathbf{c}_{i*})/\tau}}{\sum_{k=1}^{C} e^{s(\mathbf{q}_i,\mathbf{c}_{ik})/\tau}},$$

with $\mathbf{c_{i*}}$ representing the embedding of the gold chunk. Furthermore, the in-batch loss is:

$$\mathcal{L}^{\text{batch}} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{s(\mathbf{q}_i,\mathbf{c}_{i*})/\tau}}{\sum_{j=1}^{N}\sum_{k=1}^{C} e^{s(\mathbf{q}_i,\mathbf{c}_{jk})/\tau}}.$$

The final local loss is then calculated by:

$$\mathcal{L}^{\text{local}} = \alpha\mathcal{L}^{\text{seq}} + (1-\alpha)\mathcal{L}^{\text{batch}}.$$

In our experiments, we set $\alpha = 0.2$.

For the global loss, we employ an InfoNCE objective to model query–document similarities. However, multiple queries within a batch may correspond to the same document, which would erroneously treat duplicate documents as negatives and introduce false negatives during training. To mitigate this, we identify and mask duplicate documents in the batch by comparing their hashes. Let $h(d)$ be a hash function mapping documents to identifiers (e.g., MD5 hash of document text). Define a duplicate indicator masking matrix M:

$$M_{ij}^{\text{dup}} = \begin{cases} 0, & \text{if } h(d_i) = h(d_j) \text{ and } i \neq j, \\ 1, & \text{otherwise.} \end{cases}$$

Similar to the pair loss, we apply similarity threshold masking and include query-query negatives. Combining these with duplicate document masking, we define the global loss as:

$$\mathcal{L}^{\text{global}} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{s(\mathbf{q}_i,\mathbf{d}_i)/\tau}}{\sum_{j=1}^{N} M_{ij}^{\text{dup}} m_i(\mathbf{d}_j) e^{s(\mathbf{q}_i,\mathbf{d}_j)/\tau} + \sum_{j\neq i}^{N} m_i(\mathbf{q}_j) e^{s(\mathbf{q}_i,\mathbf{q}_j)/\tau}}.$$

For total loss, pplx-embed-context-v1 combines local and global losses with a scheduled weight β. We use the cosine scheduler starting at $\beta = 0.2$ with a final target value of 0.5. The goal is for the model to first focus on learning local chunk-level semantics and gradually include document-level learning in order to mitigate forgetting global coarse document-level semantics. The total loss is defined as:

$$\mathcal{L}^{\text{context}} = \beta\mathcal{L}^{\text{global}} + (1-\beta)\mathcal{L}^{\text{local}}.$$

Triplet Training. Triplet training extends traditional pairwise contrastive learning by incorporating explicit hard negative examples alongside positive documents, enabling models to learn more discriminative embeddings through fine-grained relevance distinctions. Given a set of N query-document triplets, we compute embeddings $\{\mathbf{d}_{ik}^h \in \mathbb{R}^d\}_{k=1}^K$ corresponding to the hard negatives of the query $\mathbf{q}_i \in \mathbb{R}^d$. The triplet contrastive InfoNCE loss is then formulated as:

$$\mathcal{L}^{\text{triplet}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\mathbf{q}_i,\mathbf{d}_i)/\tau}}{\sum_{j=1}^{N} e^{s(\mathbf{q_i},\mathbf{d}_j)/\tau} + \sum_{j=1}^{N}\sum_{k=1}^{K} e^{s\left(\mathbf{q}_i,\mathbf{d}_{jk}^h\right)/\tau}}.$$

2.1 Datasets for Contrastive Learning

For contrastive training, we utilize English, multilingual, and synthetic datasets. The final set contains 65.6% English, 6.7% cross-lingual, 1% code, and 26.7% multilingual samples from 60 different languages. Contextual training is performed on the ConTEB training data, as well as data synthesized from the MLDR training set. Triplet training uses considerably less but higher-quality data, spanning 12 datasets. Of this data, 92% is English, 1% is code, and 7% consists of multilingual text covering 15 different languages.

All synthetic training data are generated using LLM-based synthesis with the Qwen3-30B-A3B-Instruct-2507 model. Inspired by Zhang et al. (2025b), our synthesis pipeline employs a two-stage persona-based approach to create diverse query-document pairs from web-scale corpora based on the top-5 relevant personas. For contextual training, we utilize a similar pipeline that generates synthetic queries for passages in a given document.

3 Evaluations

We conduct extensive evaluations across multiple domains and languages, focusing on retrieval tasks using standard public benchmarks and our internally-curated benchmarks in Sec. 3.1 and Sec. 3.2.

3.1 Public Benchmarks

MTEB. We report the average performance on the 18 retrieval tasks from MTEB(Multilingual v2) in the first column of Table 1. Moreover, results of the MTEB(Code) benchmark comprising 12 retrieval tasks across 15 different programming languages are provided in the second column of Table 1. On MTEB(Multilingual v2), our 4B model, pplx-embed-v1-4B, outperforms gemini-embedding-001 in terms of average score while closely rivaling Qwen3-Embedding-4B. Our 0.6B-parameter model, pplx-embed-v1-0.6B, outperforms its Qwen counterpart. On MTEB(Code), while our 4B model performs slightly worse than Qwen3-Embedding-4B, it outperforms text-embedding-3-large and gemini-embedding-001. Moreover, pplx-embed-v1-0.6B outperforms its Qwen3 counterpart. We provide per-task evaluations and details on the evaluation approach in App. A.4.

To compare the performance-storage trade-offs, we also investigate the storage-efficiency of embedding models in Table 1 by reporting the number of documents that can be stored per gigabyte (GB). Higher values indicate more storage-efficient representations, as more documents can be stored within the same memory or storage budget. As can be seen, the pplx-embed-v1 series demonstrate superior efficiency, with the INT8 and binary variants of pplx-embed-v1-4B achieving comparable retrieval scores at 419.43K and 3.35M documents/GB, respectively. These results highlight pplx-embed-v1's dominance in low-storage regimes, enabling compression with minimal accuracy trade-offs.

Table 1: Storage efficiency (Documents per GB) and average performance (nDCG@10) on multilingual and code retrieval tasks. Upper group: models >1B or unknown parameters; lower group: models <1B parameters. The best score per group is in bold, second-best score underlined.

Model	MTEB (Multilingual, V2)	MTEB (Code)	Documents Per GB
pplx-embed-v1-4B (**INT8**)	**69.66**	78.73	419.43K
pplx-embed-v1-4B (**BIN**)	68.22	78.11	**3.35M**
qwen3-embed-4B	69.60	**80.07**	104.86K
gemini-embedding-001	67.71	76.00	87.38K
text-embedding-3-large	59.27	66.54	87.38K
pplx-embed-v1-0.6B (**INT8**)	**65.41**	**75.85**	1.05M
pplx-embed-v1-0.6B (**BIN**)	61.44	73.91	**8.39M**
qwen3-embed-0.6B	64.65	75.42	262.14K
embed-gemma-0.3B	62.58	68.76	349.52K

Table 2: nDCG@10 on the MIRACLHardNegative task per language.

Model	Docs/MB	Avg	ar	bn	de	en	es	fa	fi	fr	hi	id	ja	ko	ru	sw	te	th	yo	zh
pplx-embed-v1-4B (**INT8**)	390	66.2	74.5	74.7	58.9	55.8	55.2	57.7	76.8	55.8	60.5	51.1	68.7	68.2	68.2	71.4	78.8	74.6	80.2	61.3
pplx-embed-v1-4B (**BIN**)	**3,125**	64.6	73.3	73.1	55.8	54.9	53.1	56.7	74.9	55.1	58.6	49.7	66.2	66.3	65.8	70.6	77.2	73.3	79.4	59.6
gemini-embedding-001	81	**70.4**	**78.8**	**79.3**	60.7	59.0	57.5	61.6	**77.5**	56.3	**65.1**	54.4	**75.5**	**69.2**	**74.0**	**81.3**	80.7	**81.3**	**89.1**	**66.4**
qwen3-embed-4B	97	69.5	78.6	78.3	**63.0**	**59.6**	**58.9**	**62.2**	**77.5**	**60.2**	64.2	**56.8**	72.5	69.1	71.7	68.9	**84.1**	79.9	80.3	65.1
text-embedding-3-large	81	56.9	69.1	52.4	52.6	53.5	51.9	41.7	72.8	51.1	42.8	50.0	60.8	58.2	54.9	67.8	54.2	65.2	68.4	57.4
pplx-embed-v1-0.6B (**INT8**)	976	**68.6**	**77.9**	75.6	**60.7**	57.5	**60.1**	**60.1**	**76.6**	**59.9**	**61.2**	**55.0**	**70.8**	**70.3**	**69.5**	**74.3**	**78.6**	**78.7**	**82.1**	65.2
pplx-embed-v1-0.6B (**BIN**)	**7,812**	64.2	73.9	71.8	54.7	54.1	53.7	55.9	73.2	54.9	56.0	51.2	66.0	67.5	64.1	70.0	75.4	74.5	80.3	58.7
embed-gemma-0.3B	325	62.3	72.8	**76.0**	56.6	**59.2**	55.9	58.8	70.3	58.0	59.7	51.7	68.8	64.1	61.8	62.6	63.1	62.8	48.5	**68.0**
qwen3-embed-0.6B	244	61.2	70.5	66.9	54.2	51.8	55.5	54.3	70.3	55.0	53.2	52.4	63.1	62.0	61.0	49.0	77.7	73.9	72.1	59.2

MIRACL. To provide a more fine-grained illustration of our models' performance on specific languages, we present the evaluation results on the MIRACL benchmark, a subset of MTEB Multilingual, in Table 2. The MIRACL benchmark evaluates retrieval performance on 18 different languages across several scripts. On this task, pplx-embed-v1-0.6B performs particularly well, outperforming Qwen3-Embedding-0.6B on all language subsets. Even the binarized variant outperforms Qwen3-Embedding-0.6B in terms of average score. Our 0.6B model even exceeds the average score of our 4B model. On average, all pplx-embed-v1 models outperform text-embedding-3-large while trailing Qwen3-Embedding-4B and gemini-embedding-001.

ConTEB. We evaluate our models on the ConTEB (Conti et al., 2025) benchmark using the authors' evaluation framework `cde_benchmark`[5], a standardized toolkit for assessing chunk-level retrieval performance. When a contextual model is provided, the toolkit employs late-chunking (Günther et al., 2024) for encoding chunks. In our evaluations, we perform quantization on the pooled chunk embeddings and compute cosine similarity between query embeddings and all document chunk embeddings, ranking chunks by relevance score. Our evaluations are conducted on eight diverse datasets from the ConTEB suite: SQuAD, MLDR, NarrativeQA, Football, COVID-QA, Geography, ESG Reports, and Insurance. App. A.5 provides more details of the evaluation process. In Table 3, we provide a comparison of the nDCG@10 for contextual and non-contextual models. Our results show that pplx-embed-context-v1-4B yields the best performance while pplx-embed-context-v1-0.6B ranks third, outperforming contextually trained ModernBERT-Large and Anthropic Contextual Retrieval, but trailing the voyage-context-3 model.

BERGEN. To demonstrate the effectiveness of pplx-embed-v1 in large-scale RAG pipelines, we present evaluations of the BERGEN benchmark (Rau et al., 2024). We index the KILT Wikipedia dump (Petroni et al., 2021), consisting of 24.8 million non-overlapping 100-word passages, using pplx-embed-v1, Qwen3-Embedding, and BGE-M3. Following Rau et al. (2024), we evaluate the five Question Answering tasks that benefit most from retrieval augmentation and report the results in Table 4. For each question, the top-5 retrieved passages are presented to Qwen2.5-32B-Instruct[6], which generates an answer. In Table 4, we report the match metric, measuring the proportion of

[5] `https://github.com/illuin-tech/conteb`

[6] `https://huggingface.co/Qwen/Qwen2.5-32B-Instruct`

Table 3: Comparison of performance on ConTEB. Models above the line are non-contextualised; models below are contextualised. Dataset abbreviations: Covid (covid-qa), ESG (esg-reports), FB (football), Geo (geography), Ins (insurance), MLDR (mldr), NQA (narrative-qa), SQ (squad). * indicates numbers taken from Conti et al. (2025).

Model	Avg	Covid	ESG	FB	Geo	Ins	MLDR	NQA	SQ
pplx-embed-v1-0.6B (**INT8**)	55.32	63.44	42.01	29.29	63.60	10.13	79.84	80.71	73.5
pplx-embed-v1-0.6B (**BIN**)	53.29	59.59	40.13	25.96	60.83	11.09	78.95	78.63	71.12
pplx-embed-v1-4B (**INT8**)	58.83	63.81	47.23	34.26	73.57	14.96	79.76	81.66	75.38
pplx-embed-v1-4B (**BIN**)	57.91	62.29	46.83	31.92	72.16	15.75	79.10	80.82	74.38
qwen3-embed-0.6B	49.36	49.10	34.72	23.15	58.11	12.65	76.27	74.02	66.85
qwen3-embed-4B	54.81	54.76	39.65	31.64	71.81	14.18	76.15	77.62	72.68
pplx-embed-context-v1-0.6B (**INT8**)	76.53	56.21	46.92	71.43	89.38	99.28	85.99	84.13	78.91
pplx-embed-context-v1-0.6B (**BIN**)	71.69	50.28	33.85	66.35	86.99	96.23	82.84	80.54	76.40
pplx-embed-context-v1-4B (**INT8**)	**81.96**	**62.16**	**62.40**	78.13	**93.04**	**100**	**89.50**	**86.71**	**83.73**
pplx-embed-context-v1-4B (**BIN**)	80.46	59.60	58.14	76.49	92.56	99.69	88.90	85.87	82.67
modernBERT-Large*	75.6	56	43.1	63.9	90.7	**100**	88.7	81.3	80.9
anthropic contextual*	72.4	60.7	34.8	53.9	89.4	**100**	85.4	77.7	77.1
voyage-context-3	79.45	55.43	54.00	**79.56**	92.85	**100**	89.24	81.79	82.70

Table 4: Match metric on the BERGEN benchmark with `Qwen/Qwen2.5-32B-Instruct` as a generator. No reranking is performed and answers are generated based on the top-5 retrieved passages. The standard retrieval prompt is prepended to queries for Qwen3-Embedding. See App. A.6 for details.

Model	KILT-NQ	KILT-HotpotQA	KILT-TriviaQA	ASQA	PopQA
pplx-embed-v1-4B (**INT8**)	**67.7**	**51.9**	**91.9**	71.5	68.7
pplx-embed-v1-0.6B (**INT8**)	67.2	51.6	91.0	72.6	**70.0**
qwen3-embed-4B	67.1	50.2	91.5	**72.7**	66.4
qwen3-embed-0.6B	63.0	47.2	88.3	66.9	63.9
bge-m3	66.8	49.3	89.4	69.4	68.5

generated answers that contain the corresponding ground-truth label. We observe that pplx-embed-v1 performs strongly across all tasks. The pplx-embed-v1-4B model achieves the best results in three of the five tasks, outperforming Qwen3-Embedding-4B in four tasks. We also note that pplx-embed-v1-0.6B outperforms Qwen3-Embedding-4B in three of the five tasks, highlighting its strong performance despite its small size.

Tool Search. We evaluate our models on the ToolRet benchmark (Shi et al., 2025), a comprehensive tool retrieval dataset consisting of 35 tasks across three categories: Web (19 tasks), Code (7 tasks), and Custom (9 tasks). The benchmark contains diverse queries requiring models to retrieve relevant tools from a corpus of API documentation, with evaluation metrics including nDCG@10, Precision@10, Recall@10, and Comprehensiveness@10. As shown in Table 5, our pplx-embed-v1-4B model achieves an average nDCG@10 of 44.45, ranking second overall among all evaluated models and demonstrating particularly strong performance on the Web category (42.07 nDCG@10). Despite using INT8 quantization, our models remain competitive with larger full-precision baselines, with pplx-embed-v1-0.6B achieving 43.05 average nDCG@10 while being significantly more efficient. These semantic retrieval approaches significantly reduce context explosion by identifying relevant tools from large API corpora, enabling more efficient context management.

3.2 Internal Benchmarks

Public benchmarks are limited in fully capturing web-scale retrieval challenges such as noisy documents and distribution shifts in production. To benchmark performance in realistic deployment scenarios, we built PPLXQuery2Query and PPLXQuery2Doc, web-scale benchmarks with up to 115K real-world queries spanning over easy-to-hard difficulty, evaluated against more than 30 million documents pooled from over 1 billion web pages. This setup gives us a clearer signal on recall and ranking performance under realistic corpus size and noise.

Table 5: Results on ToolRet benchmark. All metrics are @10. N=nDCG, P=Precision, R=Recall, C=Comprehensiveness.

Model	Web				Code				Custom				Avg	
	N	P	R	C	N	P	R	C	N	P	R	C	N	C
bm25	26.33	6.10	34.22	22.79	41.90	6.20	56.49	55.39	41.16	8.39	48.60	38.90	36.46	39.03
gtr-t5-large	24.37	5.27	31.64	21.26	36.76	5.33	47.42	45.92	42.04	8.48	50.84	40.00	34.39	35.73
gte-base	30.75	7.00	39.44	25.88	41.68	6.20	53.96	51.64	37.95	6.96	46.57	38.10	36.79	38.54
bge-large	30.03	7.01	39.28	25.63	41.53	6.00	52.76	51.18	43.90	8.31	51.79	42.24	38.49	39.68
e5-mistral-7B	31.07	7.65	41.30	27.04	44.97	6.66	58.95	56.79	40.88	7.91	49.35	38.35	38.97	40.73
nv-embed-v1	31.51	7.74	40.52	26.74	**47.92**	7.10	**62.07**	**59.60**	48.70	10.07	57.69	43.88	42.71	43.41
gte-qwen2-1.5B	37.53	9.31	48.31	30.95	47.38	**7.29**	61.12	59.55	**52.98**	**10.63**	**59.47**	45.68	**45.96**	**45.39**
gritlm-7B	36.58	9.34	46.01	27.65	41.26	6.17	53.81	52.07	45.55	9.74	54.01	41.40	41.13	40.37
pplx-embed-v1-0.6B (**INT8**)	35.84	8.19	45.26	30.31	45.52	6.62	59.07	57.32	47.79	9.24	55.33	45.24	43.05	44.29
pplx-embed-v1-4B (**INT8**)	**42.07**	**9.89**	**52.55**	**36.42**	41.61	6.20	54.96	53.12	49.68	9.53	57.77	**45.98**	44.45	45.17

Table 6: Query-to-Query Retrieval Performance on PPLXQuery2Query Benchmark. Models are grouped by size (separator line at 1B parameters). Qwen models evaluated using their default prompt as specified [7].

Model	Small (240K)			Medium (1.2M)			Large (2.4M)		
	R@10	R@20	R@100	R@10	R@20	R@100	R@10	R@20	R@100
pplx-embed-v1-4B (**INT8**)	**85.04**	**88.15**	**92.75**	**77.47**	**81.87**	**88.55**	**73.46**	**78.26**	**86.17**
pplx-embed-v1-4B (**BIN**)	84.02	87.28	91.98	76.44	80.77	87.69	72.41	77.23	85.21
qwen3-embed-4B	80.90	84.57	90.21	72.36	76.96	84.90	67.90	73.02	81.96
pplx-embed-v1-0.6B (**INT8**)	**82.68**	**85.97**	**91.01**	**75.05**	**79.31**	**86.40**	**71.05**	**75.75**	**83.86**
pplx-embed-v1-0.6B (**BIN**)	80.25	83.69	89.01	72.58	76.83	84.11	68.60	73.24	81.42
bge-m3	73.70	77.39	84.08	65.63	69.85	77.76	61.78	66.15	74.69
qwen3-embed-0.6B	68.71	73.15	81.54	59.31	64.12	73.64	55.07	59.86	69.82

PPLXQuery2Query. We construct the PPLXQuery2Query benchmark from real search logs over five consecutive days from our production search system. Details of the construction process are provided in App. A.7. This benchmark contains a query set of 100K instances evaluated against document corpora of increasing size (240K, 1.2M, 2.4M), enabling analysis of scale-dependent retrieval performance. Evaluation uses Recall@K for K $\in \{10, 20, 100\}$.

Results in Table 6 show that on the PPLXQuery2Query benchmark, pplx-embed-v1 models achieve state-of-the-art performance in all size categories. On the Large corpus, pplx-embed-v1-4B achieves 73.46% R@10 and 86.17% R@100 with INT8 quantization, surpassing Qwen3-Embedding-4B by 5.56% and 4.21%, respectively. The binary quantized variant maintains strong performance with minimal degradation of only 0.96–1.05% while still outperforming all competitors. Similarly, pplx-embed-v1-0.6B achieves 71.05% R@10 and 83.86% R@100, establishing substantial margins over BGE-M3 and Qwen3-Embedding 0.6B.

PPLXQuery2Doc. We designed PPLXQuery2Doc with the goal of evaluating the retrieval performance on the web-scale search in the real-world settings. Details of the construction process are provided in App. A.8. The resulting benchmark comprises 15,000 queries, with 9,380 in English and 5,620 in other languages. We focus on Recall@K to better reflect real-world cascade search systems where embedding models serve as the first-stage retrieval component. We evaluate across three benchmark sizes of small, medium, and large, and report Recall@10, Recall@20, and Recall@100 across all benchmarks, with additional Recall@1000 reported for the large benchmark.

Reported results in Tables 8 and 7 demonstrate that pplx-embed-v1 achieves strong performance across both English and Multilingual variants. On the Large corpus, pplx-embed-v1-4B achieves 88.23% Recall@1000 on English and 91.66% on Multilingual, surpassing Qwen3-Embedding-4B. The binary quantized variants maintain competitive performance with minimal degradation of 1-2%, demonstrating the effectiveness of our quantization-aware training approach. Similarly, pplx-

[7] https://huggingface.co/Qwen/Qwen3-Embedding-0.6B/blob/main/config_sentence_transformers.json

Table 7: Query-to-Document Retrieval Performance on PPLXQuery2Doc Benchmark (English). Models are grouped by size (separator line at 1B parameters).

Model	Small (7.5M)			Medium (15M)			Large (30M)			
	R@10	R@20	R@100	R@10	R@20	R@100	R@10	R@20	R@100	R@1000
pplx-embed-v1-4B (**INT8**)	**16.29**	**26.00**	**61.38**	**13.76**	**21.84**	**51.05**	**12.18**	**19.22**	**44.26**	**88.23**
pplx-embed-v1-4B (**BIN**)	15.73	25.05	59.97	13.30	20.92	49.52	11.74	18.40	42.82	87.13
qwen3-embed-4B	11.93	19.73	52.72	9.82	16.00	42.31	8.46	13.73	35.53	83.13
pplx-embed-v1-0.6B (**INT8**)	**14.82**	**23.38**	**56.89**	**12.54**	**19.60**	**46.59**	**11.14**	**17.17**	**40.17**	**84.43**
pplx-embed-v1-0.6B (**BIN**)	13.34	21.31	53.33	11.13	17.62	42.94	9.72	15.27	36.42	80.71
bge-m3	11.37	18.69	49.69	9.42	15.27	39.27	8.12	13.08	32.76	78.23
qwen3-embed-0.6B	10.50	17.42	48.02	8.59	14.04	37.55	7.42	11.97	31.10	77.90

Table 8: Query-to-Document Retrieval Performance on PPLXQuery2Doc Benchmark (Multilingual). Models are grouped by size (separator line at 1B parameters).

Model	Small (7.5M)			Medium (15M)			Large (30M)			
	R@10	R@20	R@100	R@10	R@20	R@100	R@10	R@20	R@100	R@1000
pplx-embed-v1-4B (**INT8**)	**21.05**	**32.35**	**69.70**	**17.87**	**27.14**	**59.19**	**15.58**	**23.80**	**51.84**	**91.66**
pplx-embed-v1-4B (**BIN**)	20.42	31.65	68.56	17.17	26.45	57.73	14.97	23.05	50.42	90.67
qwen3-embed-4B	17.73	27.80	64.08	14.93	23.18	53.36	13.06	20.19	46.47	88.58
pplx-embed-v1-0.6B (**INT8**)	**19.51**	**29.92**	**66.30**	**16.29**	**24.97**	**55.58**	**14.33**	**21.90**	**48.40**	**89.05**
pplx-embed-v1-0.6B (**BIN**)	17.66	27.44	62.56	14.78	22.69	51.59	12.96	19.73	44.38	85.61
bge-m3	16.57	26.16	60.14	13.60	21.42	49.18	11.80	18.46	42.14	83.33
qwen3-embed-0.6B	16.23	25.31	59.40	13.45	20.78	48.45	11.68	17.97	41.55	84.27

embed-v1-0.6B achieves 84.43% (English) and 89.05% (Multilingual) with INT8 quantization, outperforming all sub-1B competitors by substantial margins. These high recall rates at $K = 1000$ validate the models' effectiveness as first-stage retrievers in multi-stage ranking pipelines, where maximizing recall is critical for downstream reranking performance.

3.3 Effect of Binary Quantization

Across all benchmarks, the performance loss from binary quantization is considerably higher for pplx-embed-v1-0.6B, whose performance drops by roughly 2–4.4 percentage points, compared to pplx-embed-v1-4B, which only loses up to 1.6 percentage points. The same trend is observed when comparing pplx-embed-context-v1-4B and pplx-embed-context-v1-0.6B, with the 0.6B model losing up to 5 percentage points and the 4B model losing about 1 percentage point when using binary quantization. Besides the difference in the number of parameters, our 4B models may be more resilient to binarization due to their output dimension of 2560, compared to 1024 for pplx-embed-v1-0.6B. This allows our 4B models to preserve more information in their compressed representations, making them less susceptible to the information loss inherent in quantization.

4 Conclusion

This work presented pplx-embed, a family of multilingual embedding models trained with native INT8 quantization to address memory and storage constraints in large-scale retrieval. By integrating INT8 quantization into multi-stage contrastive learning, pplx-embed-v1 achieves substantial storage reductions—4× for INT8 and up to 32× for binary—while maintaining performance comparable to full-precision retrieval across benchmarks including MTEB Multilingual v2, BERGEN, ToolSearch, and our internal web-scale evaluations. Furthermore, pplx-embed-context-v1 achieves state-of-the-art performance on the ConTEB benchmark by incorporating global context into chunk embeddings. pplx-embed-v1 model family builds on diffusion-based language models with bidirectional attention with the goal of training embedding models that better capture global document context and, in turn, improve retrieval performance.

REFERENCES

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 17981–17993, 2021.

Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics, ACL*, pp. 2318–2335, 2024.

Max Conti, Manuel Faysse, Gautier Viaud, Antoine Bosselut, Céline Hudelot, and Pierre Colombo. Context is gold to find the gold passage: Evaluating and training contextual document embeddings. *arXiv preprint arXiv:2505.24782*, 2025.

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186. Association for Computational Linguistics, 2019.

Kenneth C. Enevoldsen, Isaac Chung, Imene Kerboua, Márton Kardos, Ashwin Mathur, David Stap, Jay Gala, Wissam Siblini, Dominik Krzeminski, Genta Indra Winata, Saba Sturua, Saiteja Utpala, Mathieu Ciancone, Marion Schaeffer, Diganta Misra, Shreeya Dhakal, Jonathan Rystrøm, Roman Solomatin, Ömer Veysel Çagatan, Akash Kundu, and et al. MMTEB: massive multilingual text embedding benchmark. In *International Conference on Learning Representations (ICLR)*, 2025.

Yonggan Fu, Qixuan Yu, Meng Li, Xu Ouyang, Vikas Chandra, and Yingyan Lin. Contrastive quant: quantization makes stronger contrastive learning. In *Proceedings of the 59th ACM/IEEE Design Automation Conference*, pp. 205–210, 2022.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple contrastive learning of sentence embeddings. pp. 6894–6910, 2021.

Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, Hao Peng, and Lingpeng Kong. Scaling diffusion language models via adaptation from autoregressive models. In *International Conference on Learning Representations (ICLR)*, 2025.

Michael Günther, Jackmin Ong, Isabelle Mohr, Alaeddine Abdessalem, Tanguy Abel, Mohammad Kalim Akram, Susana Guzman, Georgios Mastrapas, Saba Sturua, Bo Wang, Maximilian Werk, Nan Wang, and Han Xiao. Jina embeddings 2: 8192-token general-purpose text embeddings for long documents. *arXiv preprint arXiv:2310.19923*, 2023.

Michael Günther, Isabelle Mohr, Bo Wang, and Han Xiao. Late chunking: Contextual chunk embeddings using long-context embedding models. *arXiv preprint arXiv:2409.04701*, 2024.

Naamán Huerga-Pérez, Rubén Álvarez, Rubén Ferrero-Guillén, Alberto Martínez-Gutiérrez, and Javier Díez-González. Optimization of embeddings storage for rag systems using quantization and dimensionality reduction techniques. *arXiv preprint arXiv:2505.00105*, 2025.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*, 2021.

Albert Q Jiang, Alicja Ziarko, Bartosz Piotrowski, Wenda Li, Mateja Jamnik, and Piotr Miłoś. Repurposing language models into embedding models: Finding the compute-optimal recipe. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 61106–61137, 2024.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. Nv-embed: Improved techniques for training llms as generalist embedding models. In *International Conference on Learning Representations (ICLR)*, 2025a.

Jinhyuk Lee, Feiyang Chen, Sahil Dua, Daniel Cer, Madhuri Shanbhogue, Iftekhar Naim, Gustavo Hernández Ábrego, Zhe Li, Kaifeng Chen, Henrique Schechter Vera, et al. Gemini embedding: Generalizable embeddings from gemini. *arXiv preprint arXiv:2503.07891*, 2025b.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Bettina Messmer, Vinko Sabolčec, and Martin Jaggi. Enhancing multilingual llm pretraining with model-based data selection. *arXiv preprint arXiv:2502.10361*, 2025.

John Xavier Morris and Alexander M Rush. Contextual document embeddings. In *International Conference on Learning Representations (ICLR)*, 2025.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. MTEB: massive text embedding benchmark. In *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2023.

Niklas Muennighoff, Hongjin SU, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. Generative representational instruction tuning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=BC4lIvfSzv.

Arvind Neelakantan, Tao Xu, Raul Puri, Alec Radford, Jesse Michael Han, Jerry Tworek, Qiming Yuan, Nikolas Tezak, Jong Wook Kim, Chris Hallacy, et al. Text and code embeddings by contrastive pre-training. *arXiv preprint arXiv:2201.10005*, 2022.

Shen Nie, Fengqi Zhu, Chao Du, Tianyu Pang, Qian Liu, Guangtao Zeng, Min Lin, and Chongxuan Li. Scaling up masked diffusion models on text. In *International Conference on Learning Representations (ICLR)*, 2025a.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025b.

Guilherme Penedo, Hynek Kydlícek, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin A. Raffel, Leandro von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Guilherme Penedo, Hynek Kydlícek, Vinko Sabolcec, Bettina Messmer, Negar Foroutan, Amir Hossein Kargaran, Colin Raffel, Martin Jaggi, Leandro von Werra, and Thomas Wolf. Fineweb2: One pipeline to scale them all - adapting pre-training data processing to every language. *arXiv preprint arXiv:2506.20920*, 2025.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021.

David Rau, Hervé Déjean, Nadezhda Chirkova, Thibault Formal, Shuai Wang, Stéphane Clinchant, and Vassilina Nikoulina. BERGEN: A benchmarking library for retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, 2024.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*, pp. 3980–3990. Association for Computational Linguistics, 2019.

Stephen Robertson, Hugo Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. *Foundations and trends® in information retrieval*, 3(4):333–389, 2009.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. Colbertv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3715–3734, 2022.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Zhengliang Shi, Yuhan Wang, Lingyong Yan, Pengjie Ren, Shuaiqiang Wang, Dawei Yin, and Zhaochun Ren. Retrieval models aren't tool-savvy: Benchmarking tool retrieval for large language models. In *Proceedings of the 63nd Annual Meeting of the Association for Computational Linguistics*, 2025.

Ken Shoemake. Animating rotation with quaternion curves. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*, 1985.

Yixuan Tang and Yi Yang. Pooling and attention: What are effective designs for llm-based embedding models? *arXiv preprint arXiv:2409.02727*, 2024.

Henrique Schechter Vera, Sahil Dua, Biao Zhang, Daniel Salz, Ryan Mullins, Sindhu Raghuram Panyam, Sara Smoot, Iftekhar Naim, Joe Zou, Feiyang Chen, et al. Embeddinggemma: Powerful and lightweight text representations. *arXiv preprint arXiv:2509.20354*, 2025.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jian Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025.

Siyue Zhang, Yilun Zhao, Liyuan Geng, Arman Cohan, Luu Anh Tuan, and Chen Zhao. Diffusion vs. autoregressive language models: A text embedding perspective. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 4273–4303, 2025a.

Xinyu Zhang, Nandan Thakur, Odunayo Ogundepo, Ehsan Kamalloo, David Alfonso-Hermelo, Xiaoguang Li, Qun Liu, Mehdi Rezagholizadeh, and Jimmy Lin. Miracl: A multilingual retrieval dataset covering 18 diverse languages. *Transactions of the Association for Computational Linguistics*, 11:1114–1131, 2023.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, et al. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025b.

A APPENDIX

A.1 DETAILS ON CONTINUED PRETRAINING

Training Details. During training, we sample $t \sim \mathcal{U}(0.001, 1)$ for each input sequence independently and mask each token in the input sequence with probability t. We train our models via the standard evidence lower bound which is given by the sum of token-wise cross entropies at masked positions, scaled by $1/t$. Half of our training data consists of English educational web pages from FineWeb-Edu (Penedo et al., 2024) while the other half covers 29 other languages with data sourced from FineWeb2 (Penedo et al., 2025) and FineWeb2-HQ (Messmer et al., 2025). We train our models for 60,000 steps with a global batch size of 1024 and sequence length 4096. Thus, we perform pretraining on approximately 250 billion tokens of multilingual text data. Following Nie et al. (2025a), we truncate 1% of the training sequences to a randomly chosen length to ensure that the models are exposed to varying sequence lengths. We use an AdamW optimizer (Loshchilov & Hutter, 2019) with a warmup-stable-decay schedule and peak learning rates of 5×10^{-4} and 3.16×10^{-4} for the 0.6B and 4B models, respectively. We leverage the resulting generative models exclusively for contrastive learning, evaluating them solely on embedding quality rather than generative performance.

Hyperparameters. We perform pretraining using automatic mixed bfloat16 precision and the FlashAttention-2 (Dao, 2023) implementation. The AdamW optimizer uses weight decay of 0.01 and we set the exponential decay rates to $\beta_1 = 0.9$ and $\beta_2 = 0.98$. We apply gradient clipping based on the ℓ^2 norm of the gradients. Since the appropriate clipping threshold depends on implementational details, we do not state it here. The learning rate is warmed linearly over the course of the first 6,000 steps and decayed in a cosine schedule over the course of the last 12,000 steps.

Data. In Table 9, we list the composition of the pretraining data. We fix the prevalence of non-English languages according to their relative word count in the FineWeb2 corpus (Penedo et al., 2025).

A.2 DIFFUSION VS AUTO-REGRESSIVE PRETRAINING

In this section, we present an ablation study to demonstrate the effectiveness of diffusion pre-training and bidirectional attention. Starting from pre-trained base models, we perform a small number of contrastive pair training steps and evaluate the performance of the resulting embedding models. We evaluate four configurations derived from two base models and two pooling strategies. We compare the causally masked Qwen3 base model (denoted as Qwen3) against a bidirectional backbone pre-trained with a diffusion objective (denoted as Diffusion). For each backbone, we apply either mean pooling or last-token pooling. The Qwen3 base model remains causally masked during contrastive training while the diffusion base model uses bidirectional attention throughout training. We perform pair training on English data for less than one epoch. The training loss, presented in Figure 2, serves as an initial indicator of model performance. We find that the model configurations using the bidirectional diffusion backbone achieve substantially lower loss values in comparison to those initialized with the causally masked Qwen3 model.

In Table 10, we further compare the performance of the four variants on english retrieval tasks consisting of the MTEB(En, V2) benchmark and the English subset of MIRACLRetrievalHard-Negatives. We observe that the combination of mean pooling and diffusion pre-training provides improvements to a range of retrieval tasks, resulting in an increase of $\sim$1% point on average. In addition to modestly improving benchmark performance, mean pooling is crucial for our contextual embeddings training, as it enables computing many chunk-level representations from a single document.

A.3 DETAILS ON CONTRASTIVE TRAINING

The data used for each of our training stages differ in format, size and language distribution. In order to improve the quality of in-batch negatives, a single batch is made up of data from a single dataset; the target dataset is selected randomly with probability proportional to the size of the dataset.

Table 9: Composition of pretraining data.

Language	Script	Source	Prevalence
eng	Latn	FineWeb-Edu	50.00%
rus	Cyrl	FineWeb2-HQ	9.22%
cmn	Hani	FineWeb2-HQ	8.51%
jpn	Jpan	FineWeb2-HQ	5.19%
deu	Latn	FineWeb2-HQ	4.11%
spa	Latn	FineWeb2-HQ	4.10%
fra	Latn	FineWeb2-HQ	3.46%
ita	Latn	FineWeb2-HQ	2.18%
por	Latn	FineWeb2-HQ	1.72%
nld	Latn	FineWeb2-HQ	1.17%
pol	Latn	FineWeb2-HQ	1.15%
ind	Latn	FineWeb2-HQ	0.94%
vie	Latn	FineWeb2-HQ	0.80%
kor	Hang	FineWeb2	0.76%
tur	Latn	FineWeb2-HQ	0.66%
fas	Arab	FineWeb2-HQ	0.62%
ces	Latn	FineWeb2-HQ	0.56%
swe	Latn	FineWeb2-HQ	0.56%
ron	Latn	FineWeb2	0.55%
arb	Arab	FineWeb2-HQ	0.51%
nob	Latn	FineWeb2	0.50%
hun	Latn	FineWeb2-HQ	0.48%
dan	Latn	FineWeb2-HQ	0.44%
ukr	Cyrl	FineWeb2	0.40%
tha	Thai	FineWeb2	0.39%
ell	Grek	FineWeb2-HQ	0.36%
fin	Latn	FineWeb2	0.32%
hin	Deva	FineWeb2	0.18%
ben	Beng	FineWeb2	0.10%
zsm	Latn	FineWeb2	0.09%

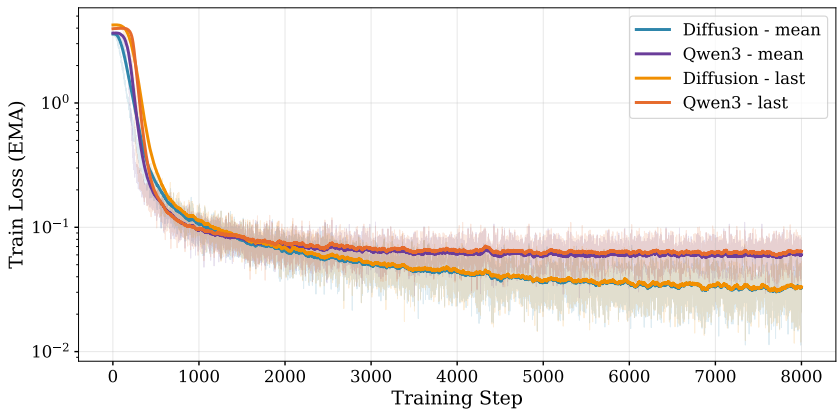

Figure 2: Smoothed training loss curves using exponential moving average with $\alpha = 0.02$, span=100.

Table 10: Effect of continued pretraining on selected tasks. Qwen3 refers to the causally-masked Qwen/Qwen3-0.6B-Base model, mean and last indicate mean pooling and last-token pooling, respectively. Dataset abbreviations: Game (CQADupstackGamingRetrieval), Unix (CQADupstackUnixRetrieval), DBP (DBpedia), FEV (FEVER), FiQA (FiQA2018), HotQ (HotpotQA), MIR (MIRACLRetrieval), MSM (MSMARCO), NFC (NFCorpus), NQ (NaturalQuestions), SciD (SCIDOCS), SciF (SciFact).

Base	Pooling	Game	Unix	DBP	FEV	FiQA	HotQ	MIR	MSM	NFC	NQ	SciD	SciF	Avg
Qwen3	last	47.7	33.2	**31.2**	**72.1**	26.6	51.7	39.3	28.3	**30.9**	38.6	**17.1**	61.5	39.9
	mean	47.2	31.4	30.3	68.1	27.0	50.5	37.8	27.3	30.2	35.5	16.8	63.0	38.8
Diffusion	last	48.6	32.0	30.8	67.1	28.8	51.8	41.4	31.1	29.1	38.2	15.7	61.4	39.7
	mean	**49.9**	**33.4**	30.2	66.5	**31.1**	**54.2**	**41.5**	**31.3**	29.6	**39.0**	16.5	**64.6**	**40.6**

Pair Training. Our models are trained using contrastive learning with InfoNCE loss, where each query uses all in-batch negative documents and all other queries as negatives, with a temperature of 0.02. Critically, we apply INT8 tanh quantization from the beginning of training, enabling the models to learn representations optimized for reduced precision. Both pplx-embed-v1-0.6B and pplx-embed-v1-4B are trained with a global batch size of 16,384 for 50,000 steps at a sequence length of 256 tokens. We use learning rates of 2e-4 and 5e-5 for the 0.6B and 4B models respectively, selected to balance training stability and convergence speed across different model scales.

Contextual Training. We train both models initialized from the checkpoint from pair training with quantization applied during both training and inference. The training corpus comprises four contextual retrieval datasets: synthetic MLDR (LLM-generated queries), MLDR, NarrativeQA, and SQuAD, all formatted with chunk-level annotations for contextual embedding learning. Additionally, we synthesize queries for MLDR chunks lacking associated queries. During synthesis, we incorporate neighborhood information into our prompts to prevent generating queries relevant to multiple chunks. Documents are partitioned into chunks of 256 tokens with a maximum of 16 chunks per document.

We employ a hybrid Matryoshka dual-objective loss that combines global document-level InfoNCE with local loss, training in six embedding dimensions [128, 256, 512, 1024, 2048, 2560] with equal weighting. The dual objective weight β is scheduled from 0.2 to 0.5 during training using a cosine annealing schedule. To mitigate false negatives, we implement three masking strategies: relative similarity threshold masking to mask negatives within 0.1 of the positive similarity, duplicate document masking to exclude same-document chunks, and query-query negative mining where additional query-query similarities serve as hard negatives for the global loss. Training runs for 7,000 steps with AdamW optimization ($lr = 10^{-5}$, weight decay=0.1) and gradient clipping at 1.0. The 0.6B and 4B models were trained with batch sizes of 128 and 16, respectively. The temperature value is set to 0.2.

Triplet Training. We perform triplet tuning using an InfoNCE loss. In this stage, each query uses in-batch negative documents augmented with 3 mined hard negatives. We increase the sequence length to 512 tokens and use a global batch size of 512 with gradient accumulation of 4. The temperature is adjusted from 0.02 to 0.03. Both models are fine-tuned for 2,000 steps, with learning rates of 5e-5 and 1e-5 for the 0.6B and 4B models respectively.

A.4 Details on MTEB Evaluation

We provide per-task scores for the MMTEB Multilingual v2 retrieval benchmark in Table 11 and per-task scores for the MTEB code benchmark in Table 12. We evaluate pplx-embed-v1 with sequence length 1024 on all tasks except for LEMBPasskeyRetrieval, where we use a sequence length of 16,384. We observed that the SyntheticText2SQL task contains duplicate documents in its corpus. To break the symmetry between these duplicates, we inject small random noise into our embeddings.

Furthermore, to calculate the number of documents that can be stored per GB, we first derive the bytes required to store a single embedding (embedding dimension × bytes per value) and then divide

Table 11: nDCG@10 on MMTEB(Multilingual, V2) retrieval tasks

Task	qwen3-embed-4B	text-embedding-3-large	gemini-embedding-001	pplx-embed-v1-4B **(INT8)**	pplx-embed-v1-4B **(BIN)**	qwen3-embed-0.6B	pplx-embed-v1-0.6B **(INT8)**	pplx-embed-v1-0.6B **(BIN)**
Average	69.60	59.27	67.71	**69.66**	68.22	64.65	**65.41**	61.44
AILAStatutes	**81.19**	41.85	48.77	61.00	60.26	**79.02**	59.84	53.77
ArguAna	75.64	57.99	**86.44**	69.99	67.81	**70.96**	65.94	61.07
BelebeleRetrieval	81.16	68.79	**90.73**	77.88	76.74	68.74	**72.86**	68.99
CovidRetrieval	**87.37**	68.43	79.13	77.50	76.98	**84.76**	77.37	74.35
HagridRetrieval	98.77	99.04	**99.31**	98.77	98.77	98.76	**98.77**	**98.77**
LEMBPasskeyRetrieval	84.25	69.75	38.50	**89.50**	79.25	**84.75**	76.75	60.25
LegalBenchCorporateLobbying	95.42	95.22	**95.98**	94.85	93.78	**94.52**	94.36	93.01
MIRACLRetrievalHardNegatives	69.49	56.94	**70.42**	66.24	64.64	61.23	**68.56**	64.21
MLQARetrieval	81.92	73.25	**84.16**	83.34	81.77	72.79	**78.98**	74.83
SCIDOCS	**31.44**	23.07	25.15	23.87	23.33	**24.41**	22.83	21.70
SpartQA	20.15	7.44	10.30	**23.30**	22.04	**10.58**	9.17	7.60
StackOverflowQA	94.32	92.44	**96.71**	93.61	93.08	89.99	**90.65**	88.97
StatcanDialogueDatasetRetrieval	42.28	31.10	51.11	**56.53**	55.56	33.63	**41.14**	34.79
TRECCOVID	**92.92**	79.56	86.32	85.61	83.94	**90.52**	85.90	81.33
TempReasonL1	1.23	2.13	**2.96**	1.19	1.16	1.02	**1.43**	1.40
TwitterHjerneRetrieval	72.58	81.44	**98.02**	75.67	75.24	60.04	**70.04**	65.20
WikipediaRetrievalMultilingual	91.23	89.24	**94.20**	92.68	92.30	87.13	**90.98**	89.74
WinoGrande	51.51	29.11	60.52	**82.40**	81.28	50.79	**71.85**	65.92

Table 12: nDCG@10 on MTEB(Code) retrieval tasks

Task	qwen3-embed-4B	text-embedding-3-large	gemini-embedding-001	pplx-embed-v1-4B **(INT8)**	pplx-embed-v1-4B **(BIN)**	qwen3-embed-0.6B	pplx-embed-v1-0.6B **(INT8)**	pplx-embed-v1-0.6B **(BIN)**
Average	**80.07**	66.54	76.00	78.73	78.11	75.42	**75.85**	73.91
AppsRetrieval	89.18	28.46	**93.75**	87.81	86.65	75.34	**78.66**	71.70
COIRCodeSearchNetRetrieval	**87.93**	75.54	81.06	84.39	83.70	**84.69**	82.38	80.15
CodeEditSearchRetrieval	76.49	71.11	**81.61**	81.26	80.23	64.42	**76.36**	72.88
CodeFeedbackMT	**93.21**	68.92	56.28	86.17	85.67	**90.82**	82.96	81.79
CodeFeedbackST	**89.51**	80.42	85.33	86.95	86.52	**86.39**	84.73	83.26
CodeSearchNetCCRetrieval	**95.59**	73.18	84.69	92.27	91.67	**91.72**	88.10	86.04
CodeSearchNetRetrieval	**92.34**	90.50	91.33	90.81	90.54	**91.01**	89.86	88.73
CodeTransOceanContest	**90.99**	84.25	89.53	88.38	88.61	**86.05**	85.03	84.86
CodeTransOceanDL	**35.04**	34.23	31.47	33.91	33.11	31.36	35.16	**35.36**
CosQA	37.98	31.00	**50.24**	42.34	41.01	36.48	**43.32**	41.04
StackOverflowQA	94.32	92.44	**96.71**	93.61	93.08	89.99	**90.65**	88.97
SyntheticText2SQL	**78.21**	68.45	69.96	76.80	76.52	**76.74**	72.98	72.18

1 GB by this per-embedding size, under the assumption that each stored embedding represents one document.

A.5 Details on ConTEB Evaluation

Our ConTEB evaluation process employs two distinct contextual embedding strategies depending on document characteristics. For standard-length documents, we use the `ContextualEmbedder`, which generates contextual embeddings by incorporating surrounding chunk information during encoding. For exceptionally long documents (notably the ESG Reports dataset, which contains documents exceeding 30,000 tokens), we implement the `FixedContextualEmbedder` with a fixed partitioning strategy that divides documents into overlapping segments of configurable size (default: 10,000 tokens with 35-chunk overlap).

A.6 Details on BERGEN Evaluation

Embedding Models. Within the BERGEN benchmark, we implement a custom retriever class that uses the official sentence-transformers implementations of pplx-embed-v1, Qwen3-Embedding, and BGE-M3, ensuring fair evaluations. When encoding queries, we set `prompt_name="query"` for the Qwen3-Embedding models.

RAG Configuration. Below, we provide the command we use for performing the end-to-end RAG evaluations. While the top-100 passages are retrieved from the KILT dump, only the top-5 passages are presented to the generator.

```
python3 -u bergen.py retriever="${retriever}"
↪  generator="vllm_qwen-25-32b-instruct" dataset="${dataset}"
↪  retrieve_top_k=100 generator.init_args.max_length=32768
```

A.7 Details of Curating PPLXQuery2Query

Our key insight for creating PPLXQuery2Query is that queries leading to the same destination URL exhibit semantic similarity, providing natural supervision for query-to-query retrieval without manual annotation. We therefore employ the following procedure for creating the PPLXQuery2Query:

(1) apply Personally Identifiable Information (PII) detection to exclude any queries that could reveal user identity; (2) collect query-URL pairs from search logs; (3) group queries by destination URL, creating clusters of semantically related queries; (4) filter clusters to retain only those with $\geq$2 queries and remove exact string duplicates; (5) within each cluster, designate the temporally first query as the evaluation query and remaining queries as pseudo documents.

A.8 Details of Curating PPLXQuery2Doc

We construct the PPLXQuery2Doc benchmark through the following methodology: (1) We create a high-quality query set using stratified sampling across four dimensions: query intent (Informational, Navigational, Transactional, Factual lookup, Exploratory), query form (Keyword-based, Natural language question, Telegraphic, Long-form verbose), query length (Short: 1–2 tokens, Medium: 3–11 tokens, Long: 12+ tokens), and language distribution to ensure multilingual coverage. (2) For each query, we retrieve documents using four retrieval systems—BM25 (Robertson et al., 2009), BGE-M3 (Chen et al., 2024), Multilingual-e5-large-instruct (Wang et al., 2022), and Qwen3-embed-0.6B (Zhang et al., 2025b)—over a corpus of 1 billion real-world web pages. (3) The union of the results from all four systems forms a candidate pool of documents per query. (4) We assign Boolean relevance labels by thresholding Reciprocal Rank Fusion (RRF) scores that aggregate ranking signals from the four retrieval systems and our production search system (which is independent of pplx-embed), ensuring robust relevance judgments through multi-system consensus.