SPECMRL: REDESIGN MATRYOSHKA REPRESENTATION LEARNING WITHIN PARAMETRIC SPECTRUM

Anonymous authors
Paper under double-blind review

ABSTRACT

Matryoshka Representation Learning (MRL) is designed to support flexible representation sizes by jointly optimizing nested prefixes. While effective, this design lacks an explicit constraint to ensure that the leading portion provides a suitable compact encoding. To address this, we propose SpecMRL, which reorganizes the feature space to enable more effective prefix readout. Specifically, we extend MRL with a parameter-derived spectral basis organized with feature importance. SpecMRL otherwise follows standard MRL, with prefix readout performed in the learned spectral space. Experiments on MTEB show that SpecMRL outperforms MRL at all evaluated reduced dimensions, with larger gains under more aggressive compression. These results suggest that the learned spectral basis better prioritizes informative directions, enabling more effective representation compression. Notably, SpecMRL extends the performance frontier of MRL, surpassing full-dimensional MRL with only half the embedding dimensions.[1]

1 INTRODUCTION

Representation learning has been a well-established research topic in ML community (Chen et al., 2020; Gao et al., 2021b) and underpins a wide range of downstream applications (Reimers & Gurevych, 2019; Karpukhin et al., 2020). The growing applications call for more flexible representation schemes that can adapt to varying latency and storage constraints across use cases. Matryoshka Representation Learning (MRL, Kusupati et al., 2022) was introduced to address this need, offering an approach that can be easily adopted in existing representation learning frameworks.

The central idea of MRL is to extend full-dimensional representation learning to a set of nested prefixes. In practice, each nested readout is constructed from the leading portion of the full representation. This design enables representations at multiple output dimensions to be learned within a single training run. While simple and effective, it does not explicitly ensure that earlier dimensions are better suited for nested readout, motivating a more principled design for MRL.

For such prefix readout to be effective, the leading dimensions should ideally concentrate the information most useful for compact representations. However, standard MRL provides no explicit constraint for establishing this ordering. Earlier studies (LI et al., 2025; Huy et al., 2026) have sought to improve low-dimensional readouts through cross-dimensional supervision. Yet, these approaches keep the original coordinate system unchanged, which may itself be ill-suited to effective prefix truncation. Figure 1 depicts how a more suitable basis can better organize semantic information and improve the quality of low-dimensional representations. This motivates us to rethink how the representation space itself should be organized to better support nested prefix readout in MRL.

To this end, we propose SpecMRL, a parameter-derived spectral redesign of MRL that organizes the representation space by feature importance. Such an organization better aligns with the prefix-based design of MRL and provides a natural foundation for dimension reduction. Starting from a backbone representation, SpecMRL differs from standard MRL only in its choice of basis. Figure 1 presents an comparative overview of MRL and SpecMRL, highlighting different organizations of representation space for nested readout. Specifically, SpecMRL changes only the representation basis, with standard MRL subsequently applied in the resulting space.

[1] Our code is available at `https://anonymous.4open.science/r/SpecMRL`.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

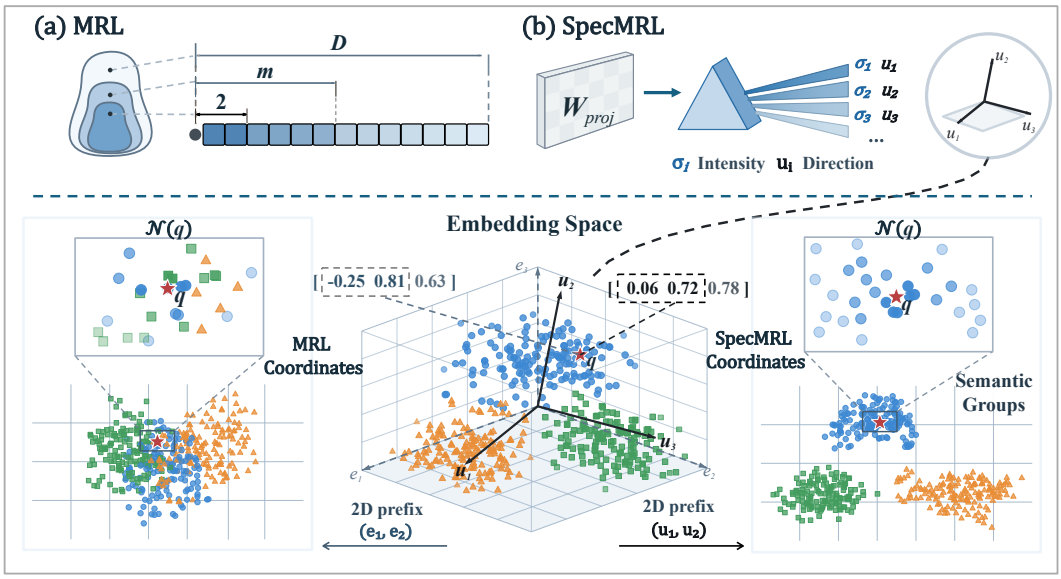


Figure 1: **Schematic comparison of standard MRL and SpecMRL.** Text embeddings are shown as points in the middle, with colors and marker shapes denoting distinct semantic groups. MRL retains coordinate prefixes in the original basis $\{e_i\}$ as compact representations, whereas SpecMRL performs the readout in an ordered spectral basis $\{u_i\}$ derived from the parameter structure of $\boldsymbol{W}_{\text{Proj}}$. The 2D views and enlarged neighborhoods of q (red star) illustrate how this change better organizes the semantic space and supports more effective low-dimensional representations.

Concretely, we introduce a matrix $\boldsymbol{W}_{\text{Proj}}$ and derive from its Gram matrix an ordered spectral basis that serves as the desired coordinate system. Despite this is simple in design, a direct implementation poses two challenges: (1) the computational overhead of standard eigen solvers and (2) the numerical instability of eigen vector gradients. To address these challenges, we introduce several technical refinements and develop a power-iteration-based alternative to approximate the eigen decomposition of the Gram matrix. Our empirical observations further confirm that this design avoids training collapse due to numerical instability while achieving throughput comparable to vanilla MRL.

We conduct experiments to validate the effectiveness of SpecMRL and provide further insights into its design. Specifically, we evaluate SpecMRL on widely used MTEB datasets (Muennighoff et al., 2022) with backbones from multiple model families. The empirical results show that SpecMRL substantially improves performance at reduced dimensions, outperforming MRL by up to 3.80 points under $16\times$ compression. Beyond these gains, SpecMRL further extends the performance frontier of MRL, surpassing its full-capacity performance with only half the dimensions. Further validation shows that these gains persist with larger backbones and longer training, confirming the potential of its further scaling. Our analyses also provide initial insights into why SpecMRL is effective. Taken together, these results motivate a rethink of the design principles of MRL and we hope it can inspire improvements for future representation learning.

2 Background: Matryoshka Representation Learning (MRL)

We briefly review how sentence embeddings are obtained and trained with LLMs before the introduction of MRL (Kusupati et al., 2022). Given an input sequence $\boldsymbol{s}$, the LLM encodes it into contextualized hidden states and aggregates them into a dense representation $\boldsymbol{X}$:

$$\boldsymbol{X} = \mathbf{P}\left(\text{LLM}\left([\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_n]\right)\right).$$

Given a training batch $\mathcal{B}$, $\boldsymbol{X}$ is typically optimized via the following contrastive loss:

$$\mathcal{L}\left(\boldsymbol{X}\right) = -\frac{1}{|\mathcal{B}|}\sum_{\boldsymbol{X}\in\mathcal{B}} \log \frac{\exp\left(\text{sim}(\boldsymbol{X}, \boldsymbol{X}^+; T)\right)}{\exp\left(\text{sim}(\boldsymbol{X}, \boldsymbol{X}^+; T)\right) + \sum_{\boldsymbol{X}^-\in\mathcal{N}} \exp\left(\text{sim}(\boldsymbol{X}, \boldsymbol{X}^-; T)\right)}, \tag{1}$$

where $\boldsymbol{X}^+$ and $\boldsymbol{X}^-$ denote the positive and negative samples of $\boldsymbol{X}$, and T controls the temperature of similarity function $\text{sim}(\cdot, \cdot\,; T)$. MRL extends this contrastive learning over a set of nested di-

mensions $\mathcal{M}$, with loss applied at each dimension. Specifically, for any nested dimension $m \in \mathcal{M}$, MRL retains the first m entries of $\boldsymbol{X}$ as the reduced representation:

$$\boldsymbol{X}_{\text{MRL}}^{(m)} \;=\; \boldsymbol{X}[:, :m], \tag{2}$$

and aggregates the losses across nested dimensions into a weighted objective:

$$\mathcal{L}_{\text{MRL}} \;=\; \sum\nolimits_{m\in\mathcal{M}} w_m \, \mathcal{L}\left(\boldsymbol{X}_{\text{MRL}}^{(m)}\right) \;, \tag{3}$$

where w_m denotes the weight assigned to each dimension m. This nested structure motivates the name, Matryoshka Representation Learning (**MRL**). Here, the "Matryoshka" metaphor is motivated by Russian nesting dolls, reflecting the intuition that more informative features should occupy earlier dimensions. Despite this motivation, MRL does not explicitly ensure such an ordering, leaving the distribution of information across dimensions unspecified. Prior work (LI et al., 2025; Yang & Zhao, 2026; Zhang et al., 2025) has also considered this issue, but mainly introduces additional constraints across nested representations instead of directly ordering the feature dimensions. In this paper, we seek to establish refined design principles toward better dimension reduction. Below, we present SpecMRL as a refined perspective on how the core motivation behind MRL can be realized.

3 METHODOLOGY

3.1 SPECMRL: PARAMETRIC SPECTRAL DESIGN FOR IMPROVED MRL

In this section, we first present the implementation of SpecMRL and defer its design principles to later sections. By design, MRL relies on a heuristic prior that favors earlier dimensions to construct nested representations. SpecMRL breaks this paradigm and recasts MRL's predefined dimension ordering as a learnable spectral organization. Specifically, it encodes this latent structure in $\boldsymbol{W}_{\text{Proj}} \in \mathbb{R}^{D\times R}$ and recovers an ordered basis from the spectrum of the corresponding Gram matrix:

$$\boldsymbol{U} \;=\; \mathbf{Eigen}\big(\, \boldsymbol{W}_{\text{Proj}}\, \boldsymbol{W}_{\text{Proj}}^{\top} \,\big)\,, \tag{4}$$

where $\mathbf{Eigen}(\cdot)$ returns the basis $\boldsymbol{U}$, with eigen values ordered as $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_D \geq 0$. Keeping the leading m columns $\boldsymbol{U}_{:,1:m} \in \mathbb{R}^{D\times m}$ as the nested readout basis, we define

$$\boldsymbol{X}_{\text{Spec}}^{(m)} = \boldsymbol{X}\boldsymbol{U}_{:,1:m} \in \mathbb{R}^{m}. \tag{5}$$

where $\boldsymbol{X}_{\text{Spec}}^{(m)}$ denotes the projection of $\boldsymbol{X} \in \mathbb{R}^D$ onto the leading m basis vectors. With this definition, SpecMRL is a drop-in modification to MRL, replacing the nested representation $\boldsymbol{X}^{(m)}$ with the spectral prefix $\boldsymbol{X}^{(m)}$Spec in each loss term while leaving the rest of the pipeline unchanged.

Given the constructed spectrum, the training objective of SpecMRL can be formulated as:

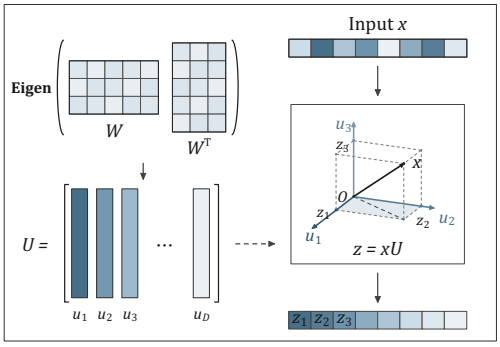
$$\mathcal{L}_{\text{Spec}} \;=\; \sum\nolimits_{m\in\mathcal{M}} w_m \, \mathcal{L}\left(\boldsymbol{X}_{\text{Spec}}^{(m)}\right) \;. \tag{6}$$

This substitution changes only the construction of nested representations while preserving the vanilla MRL framework, and requires no additional loss terms or modifications to the training pipeline. Figure 2 provides an overview of the key stages in the SpecMRL pipeline.

We present a parameter-space view to help interpret our SpecMRL design. Conceptually, the learned parameters of $\boldsymbol{W}_{\text{Proj}}$ can be viewed as an encoding of the embedding-space geometry. From this perspective, SpecMRL may organize feature channels in a more principled manner than vanilla MRL, offering one possible explanation for the observed performance gains. We provide additional empirical observations in Appendix D to further support this interpretation.

3.2 SPECMRL WITH EFFICIENT AND SCALABLE SPECTRAL SOLVER

While SpecMRL is simple in design, its naive implementation can involve several subtle technical challenges that are easy to overlook. A straightforward approach is to directly decompose the Gram matrix using off-the-shelf numerical tools, such as the built-in eigen solvers in PyTorch (Paszke



(a) Spectral readout.

```python
# W: learnable factor [D, R]
# U: cached spectral basis [D, D]
def specmrl(q, p, W, U, y):
    # Warm-started spectral update
    G = W @ W.T
    U, _ = torch.linalg.qr(G @ U.detach())
    # Spectral readout
    q, p = q @ U, p @ U

    # Original Matryoshka loss
    loss = mrl_loss(q, p, y)
    return loss, U.detach()
```

(b) PyTorch-style pseudocode.

Figure 2: Illustration and pseudocode of SpecMRL. (a) Embeddings are rotated into a learned spectral basis before prefix readouts. (b) The basis is updated by one warm-started orthogonal iteration per training step and cached for the next step. `mrl_loss` denotes the original Matryoshka objective.

et al., 2019), to obtain the spectral basis. However, this is (1) computationally expensive and (2) sensitive to ill-conditioned eigenvalue distribution, making it unsuitable in practice.[2]

To address these challenges, we approximate the ordered basis in Eq. 4 with a streaming power-iteration design. Specifically, we first initialize the basis with the ordered eigen vectors of the initial Gram matrix. Starting from this initialization, each subsequent step updates the previous basis through a single matrix multiplication followed by QR orthogonalization (Householder, 1958):

$$\boldsymbol{U}^{(t)} = \mathrm{QR}\Big(\boldsymbol{W}_{\mathrm{Proj}}^{(t)} \boldsymbol{W}_{\mathrm{Proj}}^{(t)\top} \boldsymbol{U}^{(t-1)}\Big), \tag{7}$$

where QR returns the orthogonal factor. Gradients propagate through the Gram matrix and QR factorization, while basis $\boldsymbol{U}^{(t-1)}$ is held fixed. Similar approaches have been explored in optimizer designs to reduce decomposition overhead (Ahn et al., 2025; Vyas et al., 2025). Figure 2 (b) provides PyTorch-style pseudocode for this design. We further validate the convergence of this solver in Appendix D by comparing its solved basis againts the exact eigen decomposition solution.

The benefits of our power iteration design are empirically validated in practice. In particular, our power-iteration design avoids training collapse across all experimental settings with only modest computational overhead. Once training is complete, the spectral basis is fixed and no longer requires recomputation from $\boldsymbol{W}_{\mathrm{Proj}}$. As a result, SpecMRL incurs only the cost of a $D \times D$ linear projection at inference time. We provide a detailed analysis of both training and inference costs in Section 4.1.3 to further assess the practical efficiency of SpecMRL.

4 Experiment

4.1 Method Analysis: Clarifying Design Principles for SpecMRL

We begin with ablation studies to shed light on the research questions regarding SpecMRL, seeking to validate its effectiveness and provide insights into the design principles behind.

4.1.1 How does SpecMRL perform relative to traditional MRL?

We train SpecMRL alongside MRL on backbones from various model families and evaluate their performance on MTEB (Muennighoff et al., 2022). Table 1 summarizes the results, while detailed experimental setups are provided in Appendix A. SpecMRL highlights the following advantages:

(1) SpecMRL outperforms MRL across all reduced dimensions. The observed improvements are consistent across backbones and compression ratios, while SpecMRL preserves the nested structure of MRL. Importantly, the performance gap widens under stronger compression. The larger gains at lower dimensions suggest that SpecMRL more effectively prioritizes informative directions for low-dimensional readouts, which may explain its advantage over MRL.

[2]For readers unfamiliar with this issue, we provide a detailed discussion in Section 4.1.3.

Table 1: Performance of SpecMRL averaged across MTEB tasks, with task-level breakdown provided in Table 13. τ controls the reduction ratio, reducing the output dimension to $1/\tau$ of its original size. Colored entries highlight improvements over vanilla MRL, while bold text marks the best result for each backbone. Parenthetical values denote the gains of SpecMRL over MRL.

	Qwen3-0.6B		**Qwen2-0.5B**		**Llama3.2-1B**	
τ	MRL	SpecMRL	MRL	SpecMRL	MRL	SpecMRL
1	60.97	60.94 (−0.03)	58.96	58.74 (−0.22)	61.92	61.87 (−0.05)
2	60.54	**61.61** (+1.07)	58.67	**59.44** (+0.77)	61.74	**62.42** (+0.68)
4	59.40	61.28 (+1.88)	58.32	59.20 (+0.88)	61.36	62.23 (+0.87)
8	57.88	60.46 (+2.58)	57.26	58.44 (+1.18)	60.62	61.80 (+1.18)
16	54.73	58.53 (+3.80)	55.23	56.61 (+1.38)	59.42	60.91 (+1.49)

(2) SpecMRL extends the frontier of MRL, achieving superior performance using a compact representation. As shown in Table 1, SpecMRL achieves its best performance with only half of the full dimension. This reduction lowers retrieval costs, with negligible overhead and no compromise in downstream performance. We further analyze this dimension–performance trade-off, including the slight degradation observed at the full dimension, in Section 4.3.

We argue that these findings position SpecMRL as a promising alternative to standard MRL, pushing the performance frontier toward more compact representations.

4.1.2 What motivates the use of parametric spectrum for SpecMRL?

The spectral basis in SpecMRL can, in principle, be derived from sources beyond the learned parameter matrix. Below, we introduce several alternatives to clarify the rationale behind our choice.

Parameter-Derived Spectrum Proves Effective. One natural alternative to SpecMRL's parameter-derived spectrum is to construct the spectral basis directly from the input features. Prior work (LI et al., 2025) has also explored exploiting feature-space structure to guide MRL, making feature-derived basis a plausible choice. We further discuss the relation to ESE in Section 4.4.

Based on this motivation, we introduce a feature-derived variant **FeatMRL** for comparison:

$$\boldsymbol{U}_{\text{Feat}}^{(t)} = \mathbf{Eigen}\Big(\boldsymbol{X}^{(t)\top}\boldsymbol{X}^{(t)}\Big), \tag{8}$$

where the basis $\boldsymbol{U}_{\text{Feat}}^{(t)}$ is obtained directly from the Gram matrix of the current input batch $\boldsymbol{X}^{(t)}$. We further consider a momentum-based variant to mitigate potential batch-to-batch variations:

$$\boldsymbol{U}_{\text{Feat}}^{(t)} = \mathbf{Eigen}\,(\boldsymbol{M}^{(t)}), \quad \text{where} \quad \boldsymbol{M}^{(t)} = \beta\,\boldsymbol{M}^{(t-1)} + (1-\beta)\,\boldsymbol{X}^{(t)\top}\boldsymbol{X}^{(t)}. \tag{9}$$

Table 2 reports the results, where $\beta = 0$ denotes vanilla FeatMRL. Detailed results for all evaluated β values are provided in the Table 14. FeatMRL improves over vanilla MRL at several reduced dimensions, but remains below SpecMRL at every evaluated dimension. Momentum smoothing does not close this gap. These results favor the learned parameter-derived basis over the feature-derived alternatives evaluated here.

Table 2: Comparison of SpecMRL and FeatMRL on MTEB. Bold values indicate the best performance, while parenthetical values denote performance drops relative to SpecMRL.

τ	SpecMRL	FeatMRL ($\beta = 0$)	FeatMRL ($\beta = 0.1$)	FeatMRL ($\beta = 0.5$)	FeatMRL ($\beta = 0.9$)
1	**60.94**	60.72 (−0.22)	60.68 (−0.26)	60.69 (−0.25)	60.38 (−0.56)
2	**61.61**	60.45 (−1.16)	60.36 (−1.25)	60.42 (−1.19)	60.01 (−1.60)
4	**61.28**	59.99 (−1.29)	59.84 (−1.44)	59.91 (−1.37)	59.35 (−1.93)
8	**60.46**	58.82 (−1.64)	58.68 (−1.78)	58.75 (−1.71)	58.14 (−2.32)
16	**58.53**	55.83 (−2.70)	55.62 (−2.91)	55.90 (−2.63)	55.15 (−3.38)

Over-Parameterization of $\boldsymbol{W}_{\text{Proj}}$ Brings Additional Gains. In principle, any choice of $R \geq D$ is sufficient for $\boldsymbol{W}_{\text{Proj}}$ to attain full row rank. However, SpecMRL adopts an over-parameterized design with $R \gg D$. This is motivated by our observation that increasing R beyond this threshold can further improve the performance of SpecMRL.

To study the effect of R, we conduct a hyperparameter search over R on Qwen3-0.6B and report the performance on MTEB. As shown in Table 3, we observe that (1) over-parameterizing $\boldsymbol{W}_{\text{Proj}}$ improves performance, and (2) larger values of R continue to improve performance, while the benefit of further scaling decreases. Based on these observations, we use $R =$ 128D throughout this paper.

Table 3: Hyperparameter search for the over-parameterization scale of $\boldsymbol{W}_{\text{Proj}}$. Darker shades indicate better relative performance; the top-two results are shown in **bold** and underline.

τ	D	4D	16D	64D	128D	256D
1	60.77	60.73	<u>60.92</u>	60.72	**60.94**	60.90
2	61.02	61.27	61.34	61.45	**61.61**	<u>61.55</u>
4	60.65	60.84	61.04	61.14	**61.28**	<u>61.26</u>
8	59.64	60.00	60.27	60.43	**60.46**	<u>60.44</u>
16	57.80	58.12	58.30	58.38	**58.53**	<u>58.41</u>
AVG.	59.98	60.19	60.37	60.42	**60.56**	<u>60.51</u>

One possible explanation for this advantage is that SpecMRL makes it feasible to extend MRL to higher-dimensional spaces. $\boldsymbol{W}_{\text{Proj}}$ expands the feature space from $\mathbb{R}^D$ to $\mathbb{R}^R$, while SpecMRL avoids the need to directly perform MRL at dimension R. This enables finer-grained similarity modeling at a manageable computational cost, thereby extending the model capacity beyond that of traditional MRL. Similar benefits of over-parameterized designs have also been observed in prior representation learning studies (Chen et al., 2020).

4.1.3 Efficiency and Numerical Stability

The over-parameterized design, together with the complexity of eigen decomposition, naturally raises concerns about SpecMRL's end-to-end efficiency. In practice, power iteration confines most of the additional computation to standard matrix operations supported by modern hardware, keeping the overhead affordable. Moreover, this overhead is input-independent and remains small relative to the backbone computation. We provide a more comprehensive complexity analysis in Appendix G.

Table 4: Training throughput degradation of SpecMRL with different eigen solvers.

Solver	Qwen3	Qwen2	Llama3
SVD	-4.34%	-3.36%	-8.35%
Eigh	-4.59%	-3.52%	-4.57%
Power	-0.62%	-0.53%	-0.74%

Table 4 reports SpecMRL's training throughput drop relative to MRL for different eigen solvers in the main experiments. Specifically, we benchmark `SVD` and `Eigh` using `torch.linalg.svd` and `torch.linalg.eigh`, respectively. For `Eigh`, we use the MAGMA backend following PyTorch's guidance on numerical stability (Paszke et al., 2019). As shown in Table 4, our design incurs less overhead than established eigen solvers, achieving throughput nearly identical to that of MRL. Beyond efficiency, these eigen solvers are also unstable when the eigen gap $|\lambda_i - \lambda_j|$ vanishes.

Formally, for any eigenvector $\boldsymbol{u}_i$, its gradient with respect to the Gram matrix takes the form:

$$\frac{\partial \boldsymbol{u}_i}{\partial \boldsymbol{A}}[\Delta \boldsymbol{A}] = \sum_{j \neq i} \frac{\boldsymbol{u}_j \boldsymbol{u}_j^\top \Delta \boldsymbol{A} \boldsymbol{u}_i}{\lambda_i - \lambda_j}, \quad \text{where} \quad \boldsymbol{A} = \boldsymbol{W}_{\text{Proj}} \boldsymbol{W}_{\text{Proj}}^\top. \tag{10}$$

Our implementation avoids this gradient dependence, thereby mitigating sensitivity to the ill-conditioned eigenvalues and improving numerical stability of SpecMRL.

At inference time, once the basis is fixed, SpecMRL incurs only the overhead of a $D \times D$ matrix multiplication. Meanwhile, it halves the dimension without compromising performance, leading to faster vector retrieval. Our evaluation confirms that SpecMRL incurs negligible overhead while providing a retrieval speedup of up to $1.86\times$. Appendix F provides the detailed experimental setup.

4.2 Ablation Studies

We conduct ablation studies to further validate the effectiveness of the design choices behind SpecMRL and better understand why they lead to improved performance.

Table 5: Results of ablation studies. Parenthetical values denote drops relative to SpecMRL.

τ	MRL	SpecMRL	(a) MRL *w.* Gram	(b) MRL *w.* Orth
1	60.97	60.94	61.35 (+0.41)	60.59 (−0.35)
2	60.54	**61.61**	60.80 (−0.81)	60.91 (−0.70)
4	59.40	61.28	59.64 (−1.64)	60.24 (−1.04)
8	57.88	60.46	57.70 (−2.76)	59.28 (−1.18)
16	54.73	58.53	54.11 (−4.42)	57.19 (−1.34)

4.2.1 Spectrum Contributes Beyond Gram-matrix Over-Parameterization.

Compared with vanilla MRL, SpecMRL introduces a learnable factor $\boldsymbol{W}_{\mathrm{Proj}} \in \mathbb{R}^{D\times R}$ and derives its ordered spectral basis from the corresponding Gram matrix. To examine whether the improvement can be explained merely by this additional parameterization, we construct an ablated variant that retains the same learnable factor but removes spectral decomposition.

For a nested dimension m, let $\boldsymbol{W}_m = \boldsymbol{W}_{\mathrm{Proj}}[:m, :]$. This variant retains the original coordinate prefixes $\boldsymbol{X}^{(m)}$ and evaluates similarity after the implicit lifting $\boldsymbol{X}^{(m)}\boldsymbol{W}_m$. Equivalently, its similarity is induced by the Gram matrix $\boldsymbol{G}_m = \boldsymbol{W}_m\boldsymbol{W}_m^\top$. We therefore refer to this variant as MRL *w.* Gram. The full formulation and details are provided in Appendix A.3.

As shown in Table 5, the Gram-based variant provides only limited improvements over vanilla MRL and degrades rapidly under stronger compression. In contrast, SpecMRL consistently performs better at reduced dimensions. These results show that introducing the same learnable factor through Gram-based similarity does not reproduce the gains of SpecMRL, supporting the proposed spectral readout over this alternative.

4.2.2 Eigen Decomposition Provides More Than Learnable Orthogonality

Eigen decomposition organizes the learned structure into an ordered set of orthogonal directions. This suggests two possible sources of improvement: learning a favorable orthogonal basis and imposing a spectral ordering. To disentangle their effects, we construct an ablated variant that retains only learnable orthogonality. This allows us to assess whether the learned spectral ordering benefits MRL's nested readouts. Specifically, we replace the orthogonal basis in Eq. 4 as:

$$\boldsymbol{U}_{\mathrm{Orth}} \;=\; \mathbf{QR}\big(\,\boldsymbol{W}_{\mathrm{Proj}}\,\boldsymbol{W}_{\mathrm{Proj}}^{\top}\,\big)\,, \tag{11}$$

where QR is applied to the Gram matrix alone to retain orthogonality. Table 5 (b) reports its results. Compared with vanilla MRL, learned orthogonality provides a clear advantage, but remains inferior to the full SpecMRL design. This further highlights the value of ordering feature directions by learned importance, which is naturally compatible with MRL's nested readout design.

4.3 Validate SpecMRL at Larger Dimensions and Training Scales

We scale both model width and training scale to further evaluate SpecMRL from two perspectives: (1) whether its effectiveness persists in larger representation spaces, and (2) whether its gains truly extend the performance frontier of MRL rather than merely resulting from faster convergence.

Empirical Observations at Larger Representation Dimensions. We first increase the model width and scale the Qwen3 backbone from 0.6B to 4B to extend our evaluation. Table 6 compares SpecMRL with MRL on Qwen3-4B using MTEB. The relative gains of SpecMRL over MRL follow trends similar to those observed on the 0.6B model in the main results reported in Table 1.

Leveraging the larger hidden size of Qwen3-4B, we further conduct a fine-grained sweep over nested dimensions to investigate SpecMRL's behavior under different spectral truncations. Specifically, we partion the spectral basis into 20 bands and report the average MTEB score over a subset of tasks with the first n bands retained. Figure 3 visualizes the resulting trend, where SpecMRL exhibits a pattern similar to that of MRL: enlarging the representation leads to improved performance for most retained spectral bands. Moreover, the learned spectral basis remains effective across unseen dimensions, indicating stability at dimensions unseen during training.

Table 6: Comparison between MRL and SpecMRL on Qwen3-4B. SpecMRL remains effective as the representation space scales up.

Qwen3-4B			
D	**Bands**	MRL	SpecMRL
2560	0 : 20	64.71	64.29 (−0.42)
2432	0 : 19	64.73	64.98 (+0.25)
1024	0 : 8	64.38	**65.06** (+0.68)
512	0 : 4	63.51	64.86 (+1.35)
256	0 : 2	62.40	64.49 (+2.09)
128	0 : 1	60.20	63.51 (+3.31)

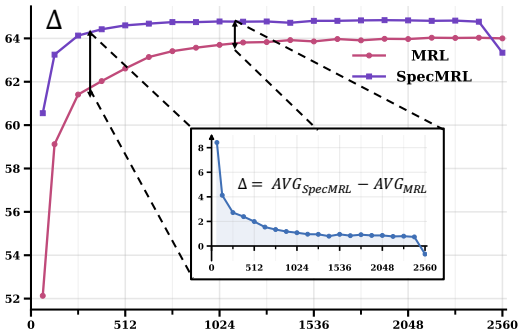


Figure 3: MTEB performance trends of SpecMRL across nested dimensions. The final spectral band introduces an unexpected performance drop.

However, we observe an unexpected degradation when the final spectral bands are retained. We therefore remove these bands in other experimental settings to examine whether the same effect persists. Tables 7 summarizes the results, where $i : j$ denotes retaining bands i through $j - 1$.

Table 7: Comparison of SpecMRL performance with and without the final spectral bands.

	Qwen3-0.6B		Qwen2-0.5B		Llama3.2-1B	
Bands	MRL	SpecMRL	MRL	SpecMRL	MRL	SpecMRL
0 : 20	60.97	60.94 (−0.03)	58.96	58.74 (−0.22)	61.92	61.87 (−0.05)
0 : 19	60.94	61.35 (+0.41)	59.00	59.20 (+0.20)	61.93	62.37 (+0.44)
0 : 10	60.54	**61.61** (+1.07)	58.67	**59.44** (+0.77)	61.74	**62.42** (+0.68)

The degradation in the smallest-singular-value bands remains consistent across different settings. Similar behavior was observed for the spectrum of the LLM unembedding matrix (Wu et al., 2026). Our further analysis attributes this phenomenon to disproportionate projection onto the corresponding spectral subspace, which partially disrupts isotropy. We defer this analysis to Appendix E and hope these findings may motivate more principled designs for embedding training.

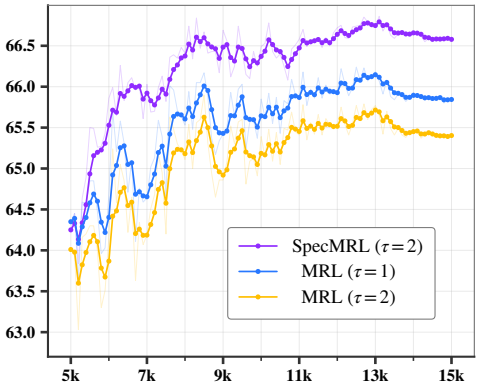


Figure 4: MTEB subset results evaluated per 100-step. The advantage of SpecMRL consistent throughout the training process.

SpecMRL Extends the Performance Frontier Beyond Faster Convergence. We next investigate whether SpecMRL's gains persist beyond faster convergence at larger training scales. Therefore, we extend Qwen3-0.6B training beyond main experiments and monitor its downstream performance relative to MRL in an over-training regime. As shown in Figure 4, SpecMRL not only retains its advantage over MRL throughout training but also attains a higher performance ceiling. Table 15 reports the full MTEB evaluation of SpecMRL and MRL in an over-training regime.

Taken together, these results highlight the potential of SpecMRL toward training with larger models and longer horizons. We hope to further validate this scalability under larger computational budgets.

4.4 Comparison of SpecMRL with Existing Improvements to MRL

We also compare SpecMRL with other MRL variants (ESE, LI et al., 2025; MIPIC, Huy et al., 2026) to further validate its effectiveness. Both ESE and MIPIC are designed to enhance information re-

tention at lower dimensions through additional supervision. Specifically, they augment the standard MRL objective with different loss designs for cross-dimensional supervision. Given that, we restrict our comparison to their reduced-dimension variants, ESE (L2C) and MIPIC (SIA), excluding other design elements. Further discussion of their full designs is provided in Section 5.

We compare SpecMRL with these baselines on Qwen3-0.6B using the same setup as in our main experiments. Table 8 presents the comparative results, where SpecMRL achieves the best overall performance among the methods considered. Notably, ESE (L2C) shows a performance collapse on classification tasks. Therefore, we also report the average over the non-classification subset in Table 16, where the same conclusion holds. Beyond these gains, SpecMRL provides additional advantage in practice, since it only changes the prefix-readout basis. As a consequence, this makes SpecMRL a compatible extension to MRL, requiring no extra hyperparameter tuning while remaining robust across diverse tasks. Further details for baselines implementation and additional evaluation results are provided in Appendix A.2 for clarity.

Table 8: Comparison of SpecMRL with MRL variants on Qwen3-0.6B.

τ	MRL	ESE (L2C)	MIPIC (SIA)	SpecMRL
1	**60.97**	57.97 (−3.00)	60.42 (−0.55)	60.94 (−0.03)
2	60.54	57.26 (−3.28)	59.93 (−0.61)	**61.61** (+1.07)
4	59.40	56.38 (−3.02)	59.08 (−0.32)	**61.28** (+1.88)
8	57.88	55.18 (−2.70)	57.74 (−0.14)	**60.46** (+2.58)
16	54.73	50.26 (−4.47)	55.09 (+0.36)	**58.53** (+3.80)

5 Related Work

Recent studies have explored various approaches to improve MRL. Most methods introduce auxiliary objectives to align representations across dimensions. Following this direction, ESE (LI et al., 2025) constructs PCA-based compression targets from full-dimensional embeddings and uses them to supervise truncated prefixes, a strategy termed learn-to-compress (L2C). ESE further introduces learn-to-express (L2E), which supervises intermediate layers to extend the Matryoshka principle to network depth. Similarly, MIPIC (Huy et al., 2026) uses KL divergence and CKA to align token-level structures across dimensions through its SIA objective. Its PIC component further transfers semantic information from deeper to earlier layers, providing additional cross-layer supervision.

Another line of work learns an additional transformation on top of a frozen encoder to obtain nested representations. For instance, SMEC (Zhang et al., 2025) adopts sequential Matryoshka training with adaptive dimension selection to reduce gradient fluctuations. Meanwhile, AdaWidth (Yang & Zhao, 2026) learns an orthogonal rotation of frozen embeddings, providing a learnable orthogonal basis without explicit spectral ordering. Conceptually, this design resembles the learnable-basis ablation considered in our analysis, which underperforms the proposed spectral basis.

6 Conclusion

In this paper, we propose SpecMRL a parameter-derived spectral redesign of MRL that addresses the limitations of fixed-basis prefix readout. To this end, we derive an ordered orthogonal basis from a learnable parametric spectrum while leaving the original MRL framework unchanged. Experiments show that SpecMRL outperforms MRL at reduced dimensions and further extends the performance frontier of standard MRL. More broadly, these findings highlight the readout basis as an important learning target for nested representations. We hope this perspective motivates further exploration of how information should be organized and inspire more principled MRL designs.

AI use statement

In this work, we used generative AI tools to assist with translation and language editing, refine plotting code for selected figures, review and annotate code, and identify relevant literature. The core implementation of the proposed method was developed by the authors. For literature identified

with AI assistance, the authors independently read the original sources and conducted their own analysis and synthesis. All AI-assisted content was manually reviewed and verified by the authors. The authors take full responsibility for the final content of this paper.

ETHICS STATEMENT

This work studies text representation learning and compression to improve storage and retrieval efficiency. Our experiments use publicly available backbones and datasets in accordance with their respective terms of use. We did not conduct human-subject studies or collect new personal data. The learned representations may inherit biases present in the backbones and training data. These biases and their potential impacts should be assessed in the context of specific downstream applications.

REPRODUCIBILITY STATEMENT

The main text presents the formulation and spectral basis update procedure of SpecMRL, with pseudocode in Figure 2. Appendix A details the pretrained models, training data, hyperparameters, evaluation protocol, and baseline adaptations. Our code is available at `https://anonymous.4open.science/r/SpecMRL`.

REFERENCES

Kwangjun Ahn, Byron Xu, Natalie Abreu, Ying Fan, Gagik Magakyan, Pratyusha Sharma, Zheng Zhan, and John Langford. Dion: Distributed orthonormalized updates, 2025. URL https://arxiv.org/abs/2504.05295.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Luyu Gao, Yunyi Zhang, Jiawei Han, and Jamie Callan. Scaling deep contrastive learning batch size under memory limited setup. In Anna Rogers, Iacer Calixto, Ivan Vulić, Naomi Saphra, Nora Kassner, Oana-Maria Camburu, Trapit Bansal, and Vered Shwartz (eds.), *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pp. 316–321, Online, August 2021a. Association for Computational Linguistics. doi: 10.18653/v1/2021.repl4nlp-1.31. URL https://aclanthology.org/2021.repl4nlp-1.31/.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple contrastive learning of sentence embeddings. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6894–6910, Online and Punta Cana, Dominican Republic, November 2021b. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.552. URL https://aclanthology.org/2021.emnlp-main.552/.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala,

Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Ro-

driguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Alston S. Householder. Unitary triangularization of a nonsymmetric matrix. *J. ACM*, 5(4):339–342, October 1958. ISSN 0004-5411. doi: 10.1145/320941.320947. URL https://doi.org/10.1145/320941.320947.

Xinshuo Hu, Zifei Shan, Xinping Zhao, Zetian Sun, Zhenyu Liu, Dongfang Li, Shaolin Ye, Xinyuan Wei, Qian Chen, Baotian Hu, Haofen Wang, Jun Yu, and Min Zhang. Kalm-embedding: Superior training data brings a stronger embedding model, 2025. URL https://arxiv.org/abs/2501.01028.

Phung Gia Huy, Hai An Vu, Minh-Phuc Truong, Thang Duc Tran, Linh Ngo Van, Thanh Hong Nguyen, and Trung Le. MIPIC: Matryoshka representation learning via self-distilled intra-relational and progressive information chaining. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 13824–13838, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.676. URL https://aclanthology.org/2026.findings-acl.676/.

Jared Kaplan, Sam McCandlish, Thomas Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeff Wu, and Dario Amodei. Scaling laws for neural language models. *ArXiv*, abs/2001.08361, 2020. URL https://api.semanticscholar.org/CorpusID:210861095.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550/.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, et al. Matryoshka representation learning. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, pp. 30233–30249, 2022.

Xianming LI, Zongxi Li, Jing Li, Haoran Xie, and Qing Li. ESE: Espresso sentence embeddings. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=plgLA2YBLH.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*, 2022. doi: 10.48550/ARXIV.2210.07316. URL https://arxiv.org/abs/2210.07316.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. MS MARCO: A human generated machine reading comprehension dataset. *CoRR*, abs/1611.09268, 2016. URL http://arxiv.org/abs/1611.09268.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. *PyTorch: an imperative style, high-performance deep learning library*. Curran Associates Inc., Red Hook, NY, USA, 2019.

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)*, pp. 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410/.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Nikhil Vyas, Depen Morwani, Rosie Zhao, Itai Shapira, David Brandfonbrener, Lucas Janson, and Sham M. Kakade. SOAP: Improving and stabilizing shampoo using adam for language modeling. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=IDxZhXrpNf.

Songhao Wu, Zhongxin Chen, Yuxuan Liu, Heng Cui, Cong Li, and Rui Yan. Your unembedding matrix is secretly a feature lens for text embeddings. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '26, pp. 5454–5463, New York, NY, USA, 2026. Association for Computing Machinery. ISBN 9798400722592. doi: 10.1145/3770855.3818204. URL https://doi.org/10.1145/3770855.3818204.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report, 2024. URL https://arxiv.org/abs/2407.10671.

Shubing Yang and Dongfang Zhao. Adawidth: Query-adaptive embedding width for dense retrieval, 2026. URL https://arxiv.org/abs/2608.23862.

Biao Zhang, Lixin Chen, Tong Liu, and Bo Zheng. SMEC:rethinking matryoshka representation learning for retrieval embedding compression. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 26209–26222, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1332. URL https://aclanthology.org/2025.emnlp-main.1332/.

A Implementation Details

A.1 General Setup

Training Setup. Our main experiments use Qwen3-0.6B-Base[3] (Team, 2025), Qwen2-0.5B[4] (Yang et al., 2024), and Llama-3.2-1B[5] (Grattafiori et al., 2024) as backbones, with nested training dimensions $\{1024, 512, 256, 128, 64\}$, $\{896, 512, 256, 128, 64\}$, and $\{2048, 1024, 512, 256, 128\}$, respectively. The scaling experiments additionally use Qwen3-4B-Base[6] (Team, 2025) with nested dimensions $\{2560, 1024, 512, 256, 128\}$. All models use mean pooling to obtain sentence embeddings. We randomly sample 1,152,000 training examples from the publicly available KaLM-embedding-finetuning-data dataset and adopt GradCache (Gao et al., 2021a) to reduce memory usage. For the main experiments, we train for 3,000 steps in BF16 with a per-device batch size of 96 queries, each paired with one positive and seven hard negatives. The learning rate is 10^{-4}, with a 10% warmup followed by linear decay, and the weight decay is 0.01. Queries and passages are truncated to 512 tokens. We use a contrastive temperature of 0.01 and nested loss weights $(1, 0.3, 0.2, 0.1, 0.1)$, ordered from the largest to the smallest dimension. The random seed is fixed to 42. For SpecMRL, we initialize the entries of $\boldsymbol{W}_{\mathrm{Proj}}$ from $\mathcal{N}(0, 0.02^2)$ and use the eigenvectors of the resulting Gram matrix for initialization. More detailed training hyperparameters can be found in our codebase.

Evaluation. For the main experiments, we use the 56-dataset English MTEB benchmark (Muennighoff et al., 2022), covering seven task categories: classification, clustering, pair classification, reranking, retrieval, semantic textual similarity, and summarization. We use the standard metric for each dataset and report the average score across all 56 datasets. Both MRL and SpecMRL follow KaLM's task-specific inference instructions (Hu et al., 2025). For SpecMRL, we perform one orthogonal iteration in FP32 using the final parameter Gram matrix, starting from the spectral basis saved in the checkpoint. The resulting basis remains fixed throughout evaluation. For Qwen2-0.5B, we use the same nested dimensions for training and evaluation, following KaLM (Hu et al., 2025). We use unnormalized embeddings for classification and apply L_2 normalization at each target dimension for all other tasks.

A.2 Implementation of ESE and MIPIC

We implement final-layer adaptations of ESE and MIPIC on Qwen3-0.6B to focus on dimensional compression. Both baselines follow the general training and evaluation setup described above.

ESE (L2C). We retain the learn-to-compress objective and apply it to the final-layer sentence embeddings, omitting intermediate-layer supervision. Contrastive losses are applied to the full 1024-dimensional representation and its 128-dimensional prefix. The latter additionally receives alignment supervision from the compression targets. Representations at other dimensions are obtained by prefix truncation at evaluation time, without additional training losses at those dimensions.

MIPIC (SIA). We retain MRL and Self-Distilled Intra-Relational Alignment (SIA), while omitting Progressive Information Chaining (PIC) and intermediate-layer supervision. MRL follows the general setup, whereas SIA is applied only to final-layer token representations, using mean-pooled sentence embeddings as the anchor.

A.3 Training of the Gram-Matrix Ablation

In this ablated variant, we replacesspectral basis extraction with a learnable linear lifting map. Let $\boldsymbol{W}_{\mathrm{Proj}} \in \mathbb{R}^{D\times R}$, where $R = 128D$. For each nested dimension $m \in \mathcal{M}$, we first truncate the backbone representations to $\mathbf{q}_m, \mathbf{p}_m \in \mathbb{R}^{1\times m}$ and define $\mathbf{W}_m = \boldsymbol{W}_{\mathrm{Proj}}[:m, :]$. The corresponding lifted representations are $\mathbf{q}_m\mathbf{W}_m$ and $\mathbf{p}_m\mathbf{W}_m$. Rather than materializing these R-dimensional

[3] https://huggingface.co/Qwen/Qwen3-0.6B-Base
[4] https://huggingface.co/Qwen/Qwen2-0.5B
[5] https://huggingface.co/meta-llama/Llama-3.2-1B
[6] https://huggingface.co/Qwen/Qwen3-4B-Base

vectors, we compute their cosine similarity implicitly:

$$s_m(\mathbf{q}, \mathbf{p}) = \frac{\mathbf{q}_m \mathbf{G}_m \mathbf{p}_m^\top}{\sqrt{\mathbf{q}_m \mathbf{G}_m \mathbf{q}_m^\top} \sqrt{\mathbf{p}_m \mathbf{G}_m \mathbf{p}_m^\top}}, \qquad \mathbf{G}_m = \mathbf{W}_m \mathbf{W}_m^\top. \tag{12}$$

For nonzero lifted vectors, this expression is algebraically identical to their explicit cosine similarity.

At evaluation, we compute the Cholesky factorization $\mathbf{G}_m = \mathbf{L}_m \mathbf{L}_m^\top$ for each target dimension. For a truncated backbone representation $\mathbf{x}_m$, we use $\mathbf{x}_m \mathbf{L}_m$ as the m-dimensional embedding. This preserves the inner products and norms of $\mathbf{x}_m \mathbf{W}_m$, and hence the cosine similarity.

B Comparison with a Learnable Linear Readout

To assess whether a generic learnable transformation can reproduce the benefits of SpecMRL, we add an unconstrained $D \times D$ linear layer after mean pooling and apply MRL to prefixes of its output. We denote this baseline as MRL *w.* Linear. Using Qwen3-0.6B, we jointly train the linear layer and backbone with the original MRL objective.

As shown in Table 9, the two methods achieve nearly identical full-dimensional scores, but their gap widens as the dimension decreases. Both readouts can combine all backbone channels before truncation. Their different compression behavior supports the value of the spectral construction for preserving information in compact prefixes.

Table 9: MTEB results comparison with learnable linear readout.

Dim.	SpecMRL	MRL *w.* Linear
1024	60.94	60.92 (−0.02)
512	61.61	60.72 (−0.89)
256	61.28	60.14 (−1.14)
128	60.46	59.13 (−1.33)
64	58.53	57.03 (−1.50)

C Robustness to MRL Configurations

To assess the effectiveness of SpecMRL under different MRL training configurations, we replace the default nested loss weights $(1, 0.3, 0.2, 0.1, 0.1)$ with uniform weights of 1, keeping all other training settings unchanged. Table 10 compares MRL and SpecMRL under this setting.

As shown in Table 10, SpecMRL retains its advantage over MRL across all reduced dimensions under uniform loss weighting. The gains remain more pronounced under stronger compression, consistent with the main experiments. These findings show that the benefits of the learned spectral readout extend beyond the default weighting scheme, supporting the applicability of SpecMRL across different loss-weighting configurations.

Table 10: Average scores over 56 MTEB datasets with uniform nested loss weights. Parentheses show gains over MRL.

Dim.	MRL	SpecMRL
1024	60.35	60.37 (+0.02)
512	60.09	61.79 (+1.70)
256	59.31	61.60 (+2.29)
128	58.38	61.04 (+2.66)
64	56.48	59.30 (+2.82)

D Parameter-Space Encoding of Embedding Geometry

In this section, we provide empirical observations that $\boldsymbol{W}_{\mathrm{Proj}}$ indeed captures representation geometry, with its leading spectral directions aligned with the dominant subspace. Let $\boldsymbol{X} \in \mathbb{R}^{N \times D}$ contain sentence representations as rows. We construct the dominant representation subspace basis $\mathbf{E}_k \in \mathbb{R}^{D \times k}$ from the top-k eigenvectors of $\boldsymbol{X}^\top \boldsymbol{X}$, ordered by descending eigenvalues. For an orthonormal readout basis $\mathbf{B}$, let $\mathbf{B}_k$ denote its first k columns. We define the subspace overlap as

$$S_k(\mathbf{B}, \boldsymbol{X}) = \frac{1}{k} \left\| \mathbf{B}_k^\top \mathbf{E}_k \right\|_F^2. \tag{13}$$

Here, k is the retained representation dimension. A higher score indicates greater overlap between the prefix readout subspace and the dominant representation subspace. The score equals 1 when the two subspaces coincide, while an independent, uniformly random k-dimensional subspace has an expected score of k/D.

We randomly sample one million sentences from MS MARCO and compute these statistics using raw backbone representations without normalization or centering. As shown in Table 11, random readout directions yield scores close to the theoretical expectation k/D. MRL achieves higher overlap, consistent with its aggregated prefix losses indirectly encouraging useful information to concentrate in the leading coordinate subspaces. With its parameter-derived spectral basis, SpecMRL achieves the highest overlap across all evaluated dimensions. These observations support the interpretation that the learned parameters encode aspects of embedding-space geometry and suggest that this geometric alignment may facilitate information preservation in low-dimensional prefixes.

Table 11: Subspace overlap on MS MARCO. Bold values indicate the best scores; parentheses denote absolute gains over MRL.

k	Rand	MRL	SpecMRL
64	0.0628	0.1933	**0.3382** (+0.1449)
128	0.1237	0.2395	**0.4443** (+0.2048)
256	0.2513	0.2995	**0.4709** (+0.1714)
512	0.4999	0.5103	**0.5523** (+0.0419)

We additionally compare the final cached basis with the ordered eigen basis obtained by directly decomposing the same parameter Gram matrix, using the subspace-overlap measure in Eq. 13. The overlap exceeds 0.99999 at $k \in \{64, 128, 256, 512\}$, showing close agreement between the final iterative and directly computed prefix subspaces.

E Energy Concentration in the Tailing Spectral Subspace

We conduct further analyses to investigate why dropping the final spectral bands can improve SpecMRL's downstream performance. Motivated by the analysis in (Wu et al., 2026), we examine the behavior of SpecMRL-derived representations when projected onto different spectral bands.

Specifically, we first partition the spectrum into 20 consecutive bands ordered by eigenvalue. We then construct a calibration set of text embeddings from one million passages sampled from MS MARCO (Nguyen et al., 2016). Let $z_{ij} = \mathbf{u}_j^\top \boldsymbol{X}_{[i]}$ denote the projection of $\mathbf{X}_{[i]}$ onto the j-th spectral direction. The average energy fraction of band B_b is

$$E_b = \frac{1}{N} \sum_{i=1}^{N} \frac{\sum_{j \in B_b} z_{ij}^2}{\sum_{j=1}^{1024} z_{ij}^2}. \tag{14}$$

Figure 5 shows that the final band accounts for a large fraction of energy (21.72%), despite spanning only 5% of the spectral directions. Ideally, this energy should exhibit a smooth decay along the ordered spectrum. The decay pattern reflects the decreasing semantic importance of later spectral bands, while the smoothness helps maintain isotropy by avoiding concentration in a small subset of directions. In turn, the disproportionate concentration in the final band indicates increased anisotropy, suggesting a departure from the expected geometry. Although the cause of this phenomenon remains unclear, we hope this analysis provides an initial explanation for the observed improvement and motivates further investigation.

F Inference and Retrieval Efficiency

Embedding Encoding. We measure the encoding time of MRL and SpecMRL with Qwen3-0.6B backbones on the same set of 100,000 passages. After pooling, SpecMRL applies a full 1024×1024 projection using a precomputed spectral basis. Timing includes the model forward pass, pooling, and projection, but excludes tokenization, input transfer, and model loading. The total encoding time of SpecMRL is only 0.36% higher than that of MRL.

Table 12: GPU retrieval throughput (QPS) of SpecMRL. Parentheses show speedups relative to 1024-dimensional embeddings at the same batch size.

	Batch size				
Dim.	1	64	256	1024	2048
1024	140 (1.00×)	5,207 (1.00×)	7,103 (1.00×)	7,859 (1.00×)	7,961 (1.00×)
512	176 (1.26×)	7,713 (1.48×)	12,558 (1.77×)	14,605 (1.86×)	14,791 (1.86×)
256	205 (1.46×)	9,891 (1.90×)	20,174 (2.84×)	25,508 (3.25×)	25,906 (3.25×)
128	231 (1.65×)	11,414 (2.19×)	28,986 (4.08×)	40,594 (5.17×)	41,282 (5.19×)
64	352 (2.52×)	12,319 (2.37×)	36,712 (5.17×)	56,884 (7.24×)	57,814 (7.26×)

Vector Retrieval. From one million sentences sampled from MS MARCO (Nguyen et al., 2016), we randomly select 8,192 as queries and use the remaining 991,808 as the retrieval database. We perform exact inner-product search using FAISS's `GpuIndexFlatIP` on a single NVIDIA A100 PCIe 40GB GPU. We use FP32 embeddings and retrieve the top 100 results per query. All indices are prebuilt and remain in GPU memory. After warm-up, we measure retrieval time over ten complete passes through the query set, synchronizing the GPU before and after each measurement. Throughput is computed as the number of queries divided by the measured time. As shown in Table 12, reducing the embedding dimension from 1024 to 512 yields a speedup of up to $1.86\times$, with greater gains for larger query batches. Further reducing the dimension provides additional speedups.

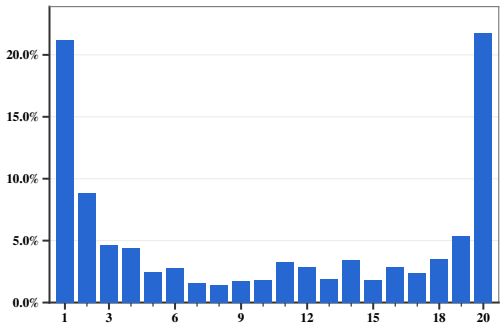


Figure 5: Average projection energy fractions across 20 spectral bands for Qwen3-0.6B. Bands are ordered by decreasing singular value of the learned projection matrix.

G Computational Complexity

This section provides a supplementary analysis of SpecMRL's computational overhead relative to MRL. The additional calculations in SpecMRL consist of only one matrix multiplication, and one QR Householder orthogonalization (Householder, 1958). Counting a multiply-add as two FLOPs, the multiplication requires an approxmiation of $2D^3$ FLOPs, while Householder QR requires another $\frac{8}{3}D^3$ FLOPs. The additional forward cost is therefore given as:

$$F_{\text{extra}} \approx O\left(\underbrace{2RD^2}_{\text{Gram construction}} + \underbrace{\frac{14}{3}D^3}_{\text{basis update}} + \underbrace{2B_{\text{proj}}D^2}_{\text{embedding projection}} \right). \tag{15}$$

For comparison, let P denote the number of linear weights, and T the number of tokens processed per device. Following the approximation in (Kaplan et al., 2020), the backbone forward cost is:

$$F_{\text{backbone}} \approx O\left(2TP\right). \tag{16}$$

Substituting the Qwen3-0.6B model configuration, the additional forward cost remains below 1% of the backbone forward FLOPs, demonstrating the low overhead of our power-iteration design.

Table 13: Task-level breakdown of MTEB results in Table 1. Shaded rows denote SpecMRL, bold values indicate better overall results, parenthetical values report gains over MRL under each setup.

	STS.	**Class.**	**Cluster.**	**PairClass.**	**Rerank.**	**Retr.**	**Sum.**	**Avg**
# Datasets	10	12	11	3	4	15	1	56
	Qwen3-0.6B							
1024 ($\tau = 1$)	80.77	74.25	48.72	72.74	48.93	49.02	30.38	**60.97**
	80.24	74.45	49.06	72.44	47.67	49.30	29.66	60.94 (-0.03)
512 ($\tau = 2$)	80.87	73.99	48.43	72.09	48.25	48.10	29.92	60.54
	80.83	75.11	49.43	74.49	50.98	49.24	30.46	**61.61 (+1.07)**
256 ($\tau = 4$)	80.21	73.45	47.89	71.42	47.11	45.62	29.00	59.40
	81.04	74.98	49.44	74.46	50.75	48.08	30.30	**61.28 (+1.88)**
128 ($\tau = 8$)	79.49	72.99	47.43	68.29	45.47	42.18	29.24	57.88
	80.76	74.56	49.28	73.90	49.95	45.93	30.74	**60.46 (+2.58)**
64 ($\tau = 16$)	78.09	72.11	46.54	63.87	42.52	34.38	29.63	54.73
	80.18	73.89	48.89	72.43	48.07	40.79	29.86	**58.53 (+3.80)**
	Qwen2-0.5B							
896 ($\tau = 1$)	78.57	71.82	46.99	71.52	46.68	47.04	30.59	**58.96**
	78.72	71.80	46.59	71.33	46.24	46.75	27.87	58.74 (-0.22)
512 ($\tau = 2$)	78.49	72.10	46.80	70.91	46.13	46.14	31.13	58.67
	79.25	73.07	47.05	73.22	48.98	46.51	28.31	**59.44 (+0.77)**
256 ($\tau = 4$)	78.41	72.39	46.82	71.23	46.49	44.57	30.18	58.32
	79.40	72.94	47.11	73.07	48.85	45.64	28.37	**59.20 (+0.88)**
128 ($\tau = 8$)	77.71	72.26	46.68	69.69	45.03	41.94	30.67	57.26
	79.29	72.79	46.96	72.25	48.53	43.34	28.52	**58.44 (+1.18)**
64 ($\tau = 16$)	76.41	71.86	46.42	67.69	43.74	36.48	30.47	55.23
	78.33	72.37	46.72	70.00	46.55	38.72	27.59	**56.61 (+1.38)**
	Llama3.2-1B							
2048 ($\tau = 1$)	80.91	75.77	50.76	75.43	47.71	49.60	29.64	**61.92**
	80.71	76.05	50.28	76.02	48.19	49.39	30.18	61.87 (-0.05)
1024 ($\tau = 2$)	80.79	75.84	50.75	75.22	46.97	49.17	30.06	61.74
	81.09	76.61	50.60	76.70	51.27	49.53	30.35	**62.42 (+0.68)**
512 ($\tau = 4$)	80.55	75.73	50.64	74.77	46.63	48.30	29.40	61.36
	81.22	76.59	50.67	76.62	51.06	48.80	30.05	**62.23 (+0.87)**
256 ($\tau = 8$)	80.31	75.40	50.49	74.00	45.30	46.59	29.54	60.62
	81.28	76.52	50.53	76.29	50.57	47.52	30.19	**61.80 (+1.18)**
128 ($\tau = 16$)	79.88	74.79	50.34	73.20	44.88	43.21	30.29	59.42
	81.32	76.28	50.48	75.40	49.69	44.74	30.91	**60.91 (+1.49)**

Table 14: Task-level breakdown of FeatMRL variants. Bold and underlined average scores indicate the best and second-best FeatMRL variants within each group.

	STS.	**Class.**	**Cluster.**	**PairClass.**	**Rerank.**	**Retr.**	**Sum.**	**Avg**
# Datasets	10	12	11	3	4	15	1	56
$\tau = \mathbf{1}$	80.24	74.45	49.06	72.44	47.67	49.30	29.66	60.94
$\beta = 0$	80.70	73.08	48.86	72.31	49.35	48.99	29.80	**60.72** (−0.22)
$\beta = 0.1$	80.73	72.54	48.77	72.27	49.82	49.10	31.06	60.68 (−0.26)
$\beta = 0.5$	80.65	72.30	48.79	72.87	49.54	49.29	31.65	<u>60.69</u> (−0.25)
$\beta = 0.9$	80.78	71.97	48.85	72.37	48.71	48.72	29.69	60.38 (−0.56)
$\tau = \mathbf{2}$	80.83	75.11	49.43	74.49	50.98	49.24	30.46	61.61
$\beta = 0$	80.85	72.86	48.79	71.78	48.62	48.43	29.41	**60.45** (−1.16)
$\beta = 0.1$	80.86	72.37	48.68	71.72	49.22	48.31	30.96	60.36 (−1.25)
$\beta = 0.5$	80.79	71.98	48.56	72.45	48.96	48.85	31.85	<u>60.42</u> (−1.19)
$\beta = 0.9$	80.97	71.69	48.69	71.77	47.89	47.92	29.47	60.01 (−1.60)
$\tau = \mathbf{4}$	81.04	74.98	49.44	74.46	50.75	48.08	30.30	61.28
$\beta = 0$	80.98	72.73	48.66	70.87	47.60	47.28	29.31	**59.99** (−1.29)
$\beta = 0.1$	80.98	71.98	48.46	70.98	48.32	47.20	29.82	59.84 (−1.44)
$\beta = 0.5$	80.79	71.69	48.51	71.68	48.02	47.66	31.25	<u>59.91</u> (−1.37)
$\beta = 0.9$	80.99	71.32	48.48	70.76	46.82	46.34	30.29	59.35 (−1.93)
$\tau = \mathbf{8}$	80.76	74.56	49.28	73.90	49.95	45.93	30.74	60.46
$\beta = 0$	80.54	72.27	48.47	68.55	45.75	44.56	30.96	**58.82** (−1.64)
$\beta = 0.1$	80.49	71.67	48.17	68.77	46.44	44.58	30.41	58.68 (−1.78)
$\beta = 0.5$	80.39	71.21	48.22	69.37	46.24	45.12	31.54	<u>58.75</u> (−1.71)
$\beta = 0.9$	80.45	70.89	48.24	68.63	44.96	43.54	31.35	58.14 (−2.32)
$\tau = \mathbf{16}$	80.18	73.89	48.89	72.43	48.07	40.79	29.86	58.53
$\beta = 0$	78.85	71.60	47.86	63.11	42.51	37.47	30.86	<u>55.83</u> (−2.70)
$\beta = 0.1$	78.73	70.95	47.58	63.54	43.23	37.26	29.82	55.62 (−2.91)
$\beta = 0.5$	78.71	70.78	47.76	63.82	43.15	38.35	29.59	**55.90** (−2.63)
$\beta = 0.9$	78.82	70.28	47.79	63.63	41.91	36.18	30.12	55.15 (−3.38)

Table 15: Task-level breakdown of MTEB in an over-training regime on Qwen3-0.6B, with training steps extended from 3K to 15K. The improvement of SpecMRL remains consistent with Table 1, suggesting that the gains of SpecMRL persist beyond faster convergence.

	STS.	**Class.**	**Cluster.**	**PairClass.**	**Rerank.**	**Retr.**	**Sum.**	**Avg**
# Datasets	10	12	11	3	4	15	1	56
1024 ($\tau = 1$)	80.52	81.40	52.79	70.19	48.99	50.69	28.91	**63.54**
	80.74	81.15	52.36	68.12	47.86	50.59	30.38	63.25 (-0.29)
512 ($\tau = 2$)	80.53	81.34	52.74	69.44	48.43	49.80	28.28	63.19
	81.35	81.50	52.64	71.39	50.95	50.59	29.75	**63.88 (+0.69)**
256 ($\tau = 4$)	80.20	81.12	52.49	68.01	46.83	47.52	27.82	62.23
	81.55	81.41	52.60	71.44	50.85	49.69	29.50	**63.64 (+1.41)**
128 ($\tau = 8$)	79.50	80.91	52.28	64.48	45.29	44.91	28.53	61.03
	81.23	81.09	52.52	70.75	50.42	47.33	28.13	**62.77 (+1.74)**
64 ($\tau = 16$)	78.21	80.29	51.88	61.40	43.08	39.03	28.71	58.69
	80.45	80.23	52.28	69.22	49.26	42.12	28.85	**60.85 (+2.16)**

Table 16: Task-level breakdown of MTEB results for SpecMRL and existing MRL variants. Column $\mathrm{Avg}_{\mathrm{w/o,Cls}}$ denotes the average MTEB results over non-classification MTEB datasets.

	Method	**STS.**	**Class.**	**Cluster.**	**PairClass.**	**Rerank.**	**Retr.**	**Sum.**	$\mathrm{Avg}_{\mathrm{w/o\ Cls}}$
1024 ($\tau = 1$)	MRL	80.77	74.25	48.72	72.74	48.93	49.02	30.38	57.35
	ESE	81.12	59.88	48.61	70.96	49.12	49.49	29.93	**57.44** (+0.10)
	MIPIC	79.88	73.05	48.75	69.69	48.51	49.21	30.77	56.98 (−0.37)
	SpecMRL	80.24	74.45	49.06	72.44	47.67	49.30	29.66	57.26 (−0.09)
512 ($\tau = 2$)	MRL	80.87	73.99	48.43	72.09	48.25	48.10	29.92	56.87
	ESE	80.92	59.12	48.47	69.47	48.49	48.23	28.85	56.75 (−0.12)
	MIPIC	80.02	72.82	48.56	68.49	47.94	48.03	30.12	56.41 (−0.45)
	SpecMRL	80.83	75.11	49.43	74.49	50.98	49.24	30.46	**57.92** (+1.05)
256 ($\tau = 4$)	MRL	80.21	73.45	47.89	71.42	47.11	45.62	29.00	55.57
	ESE	80.40	58.10	48.17	71.04	48.07	46.06	30.02	55.91 (+0.35)
	MIPIC	79.87	72.47	48.23	67.35	46.39	46.03	31.57	55.43 (−0.14)
	SpecMRL	81.04	74.98	49.44	74.46	50.75	48.08	30.30	**57.55** (+1.98)
128 ($\tau = 8$)	MRL	79.49	72.99	47.43	68.29	45.47	42.18	29.24	53.76
	ESE	79.79	57.19	47.81	68.43	47.33	43.76	29.12	54.63 (+0.88)
	MIPIC	79.36	72.07	47.96	63.52	44.77	43.13	30.51	53.82 (+0.07)
	SpecMRL	80.76	74.56	49.28	73.90	49.95	45.93	30.74	**56.61** (+2.85)
64 ($\tau = 16$)	MRL	78.09	72.11	46.54	63.87	42.52	34.38	29.63	50.00
	ESE	77.59	54.63	46.45	61.87	43.13	32.43	27.75	49.07 (−0.92)
	MIPIC	78.08	71.20	47.20	59.58	42.22	36.88	29.96	50.70 (+0.70)
	SpecMRL	80.18	73.89	48.89	72.43	48.07	40.79	29.86	**54.34** (+4.34)