Matryoshka-Adaptor: Unsupervised and Supervised Tuning for Smaller Embedding Dimensions

Jinsung Yoon, Rajarishi Sinha, Sercan Ö. Arık, Tomas Pfister
Google Cloud AI
{jinsungyoon, sinharaj, soarik, tpfister}@google.com

Abstract

Embeddings from Large Language Models (LLMs) have emerged as critical components in various applications, particularly for information retrieval. While high-dimensional embeddings generally demonstrate superior performance as they contain more salient information, their practical application is frequently hindered by elevated computational latency and the associated higher cost. To address these challenges, we propose Matryoshka-Adaptor, a novel tuning framework designed for the customization of LLM embeddings. Matryoshka-Adaptor facilitates substantial dimensionality reduction while maintaining comparable performance levels, thereby achieving a significant enhancement in computational efficiency and cost-effectiveness. Our framework directly modifies the embeddings from pre-trained LLMs which is designed to be seamlessly integrated with any LLM architecture, encompassing those accessible exclusively through blackbox APIs. Also, it exhibits efficacy in both unsupervised and supervised learning settings. A rigorous evaluation conducted across a diverse corpus of English, multilingual, and multimodal datasets consistently reveals substantial gains with Matryoshka-Adaptor. Notably, with Google and OpenAI Embedding APIs, Matryoshka-Adaptor achieves a reduction in dimensionality ranging from two- to twelve-fold without compromising performance across multiple BEIR datasets.

1 Introduction

Large language models (LLMs) have showcased remarkable proficiency in handling various text processing tasks, encompassing question answering, summarization, and mathematical reasoning (Brown et al., 2020; Chowdhery et al., 2022; Zhang et al., 2022a). This success can be partially attributed to their ability to transform raw text into semantically enriched representations, with the qual-
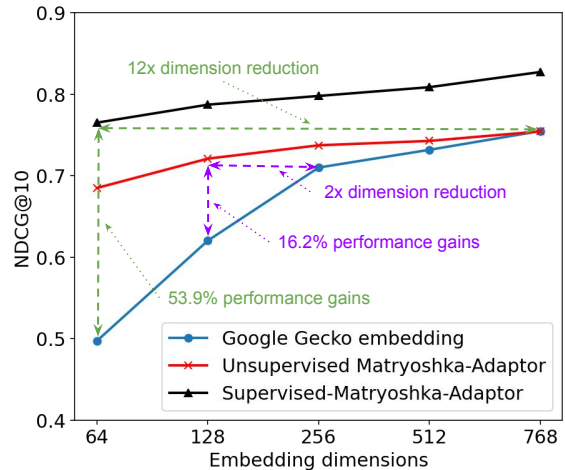

Figure 1: The effectiveness of the Matryoshka Adaptor in dimensionality reduction. In both unsupervised (red line) and supervised (black line) settings, the Matryoshka Adaptor showcases the capability to considerably decrease embedding dimensions while maintaining a negligible impact on nDCG@10 retrieval performance with BEIR SciFact dataset. Notably, at the same embedding dimensionality, the utilization of our approach results in significantly improved performance.

ity of these text-to-embedding mappings being of paramount importance (Ouyang et al., 2022).

Embeddings find extensive utilization in a wide array of downstream tasks, with information retrieval (IR) serving as a prominent one (Wang et al., 2023; Muennighoff et al., 2024; Li et al., 2023). IR involves the process of searching for relevant information within a corpus database using queries, and language modeling plays a pivotal role as both queries and corpus data are frequently textual in nature. In IR systems, text embeddings are commonly employed to rank relevant corpora based on their similarity to queries.

Numerous LLMs have been developed specifically for extracting embeddings from raw text, including Sentence T5 (Ni et al., 2021), OpenAI Embedding APIs (ope), and Google Embedding APIs

10318

(gcp). However, generation of high-dimensional embeddings often entail high latency and computational costs, thereby limiting their practical application in latency-sensitive scenarios such as large-scale recommendation systems.

Matryoshka representation learning (MRL) (Kusupati et al., 2022; Cai et al., 2024) addresses this limitation at pretraining stage, by generating embeddings that retain similar characteristics even when utilizing only a subset of their dimensions (i.e., Matryoshka properties). This enables efficient similarity comparisons with lower-dimensional embeddings and is leveraged by many state-of-the-art models, including those from Google and OpenAI (Lee et al., 2024; ope).

In this study, we present Matryoshka-Adaptor, a novel framework designed to transform arbitrary embeddings into embeddings with Matryoshka properties in both unsupervised and supervised learning setups. In an unsupervised learning setting, the adaptor learns to transform input embeddings into Matryoshka embeddings using only corpus embeddings. We introduced pairwise and top-k similarity loss functions to facilitate this process. In supervised learning setup, the adaptor can be further refined by incorporating relevant (query, corpus) pairs as labeled tuning data. Note that we customize embeddings to better suit specific datasets, using both unsupervised and supervised approaches. This refinement results in improved Matryoshka properties, surpassing even those of embeddings derived from MRL-trained models.

Extensive experiments conducted across 13 BEIR datasets (Thakur et al., 2021), 17 MIRACL datasets (Zhang et al., 2022b), and 5 multimodal Fashion-200K datasets (Han et al., 2017) validate the effectiveness of Matryoshka-Adaptor. We demonstrate significant performance improvements over the latest Google (for English only, multilingual, and multimodal data) (Lee et al., 2024) and OpenAI text embedding models (ope). Matryoshka-Adaptor exhibits broad applicability to diverse embedding types, encompassing text, multimodal, multilingual, and use case-specific embeddings. The contributions of this paper are threefold:

- We propose a novel tuning framework, Matryoshka-Adaptor that achieves substantial dimensionality reduction in embeddings without sacrificing performance achieved by tailoring embeddings to better fit specific datasets.

- Matryoshka-Adaptor is applicable in both un-

supervised and supervised settings, consistently improving retrieval performance across various datasets. Also, Matryoshka-Adaptor demonstrates a roughly two-fold (unsupervised) and six-fold (supervised) reduction in dimensionality, with no loss in performance.

- The benefits of Matryoshka-Adaptor extend to multimodal learning and multilingual scenarios.

2 Related Work

Matryoshka embeddings. (Kusupati et al., 2022) pioneered the development of embedding models whose representations could be substantially reduced in dimensionality without incurring a significant degradation in performance. These Matryoshka embedding models are specifically trained to ensure the utility of such truncated embeddings. The Matryoshka properties inherent in these models enable fine-grained control over the trade-off between latency and accuracy in downstream tasks utilizing the embeddings. Given this advantage, recent embedding models, including those from OpenAI and Google (Lee et al., 2024), have integrated Matryoshka properties by employing Matryoshka Representation Learning (MRL) during pre-training. Our proposed work further enhances the Matryoshka properties through tuning. We introduce modifications to embeddings that are tailored to target datasets in both an unsupervised and supervised tuning setups. The proposed tuning yields enhanced Matryoshka properties compared to the original embeddings, even those derived from MRL-trained models.

Dimensionality reduction. Dimensionality reduction is a well-established framework for reducing the dimensionality of vectors while preserving their inherent properties. Principal Component Analysis (PCA) (Jolliffe and Cadima, 2016), Independent Component Analysis (ICA) (Hyvärinen and Oja, 2000), and Non-negative Matrix Factorization (NMF) (Lee and Seung, 2000) are widely used unsupervised dimensionality reduction techniques. Linear Discriminant Analysis (LDA) (McLachlan, 2005) is a supervised method that leverages labeled data for dimensionality reduction. While ICA is possible, its limitations in this context arise due to the equal importance of all components, requiring separate model training for each reduced dimension. NMF, on the other hand, is not suitable for this application as it is only applicable to non-negative embeddings. Also, PCA's orthogonality

properties may limit its reliability for dimensionality reduction for vectors with highly non-linear relationships, especially at high dimensionality.

Embedding customization. In lieu of employing a single unified model for zero-shot retrieval, embedding customization tailored to individual datasets presents a viable alternative. TART (Asai et al., 2022) constructs a retrieval system that adapts retrieval based on instructions, incorporating different tasks (e.g. for code, question, or answer) to enhance dense embedding retrieval. InstructOR (Su et al., 2022) integrates task and domain descriptions for retrieval while tuning embedding models. Promptagator (Dai et al., 2022) leverages in-context learning to generate synthetic query-corpus pairs using a limited number of original pairs, subsequently fine-tuning pre-trained LLMs with these synthetic pairs. Search-Adaptor (Yoon et al., 2023) customizes embeddings for information retrieval using a small number of query-corpus pairs in supervised learning setup. Unlike these approaches, Matryoshka-Adaptor is applicable to both supervised and unsupervised settings, eliminating the need for labeled query-corpus pairs. Additionally, Matryoshka-Adaptor aims not only to enhance full-dimensional embedding performance but also to improve performance across all reduced-dimensionality embeddings.

3 Unsupervised Matryoshka-Adaptor

3.1 Problem formulation

To facilitate a clear understanding, the proposed framework is formulated within the context of information retrieval. However, it is crucial to emphasize that it can be readily generalized to any application involving embeddings. In unsupervised settings, we assume the availability of a corpus set, denoted as $\mathcal{C} = \{c_1, c_2, ..., c_N\}$ and a pre-trained embedding model (E). The embeddings extracted from the corpus are represented as $\mathcal{C}_E = \{ce_1, ce_2, ..., ce_N\}$, where each embedding vector $ce_i = E(c_i) \in \mathbb{R}^d$. Notably, our framework permits the embedding model, E, to be treated as a black-box model, and Matryoshka-Adaptor can be directly applied to the extracted embeddings, $\mathcal{C}_E$.

A Matryoshka embedding, characterized by m dimensions, is defined as the initial m dimensions of the original d-dimensional embedding, where $m < d$. This can be expressed as $\mathcal{C}_E[: m] = \{ce_1[: m], ce_2[: m], ..., ce_N[: m]\}$ where each reduced embedding vector $ce_i[: m] \in \mathbb{R}^m$. A funda-
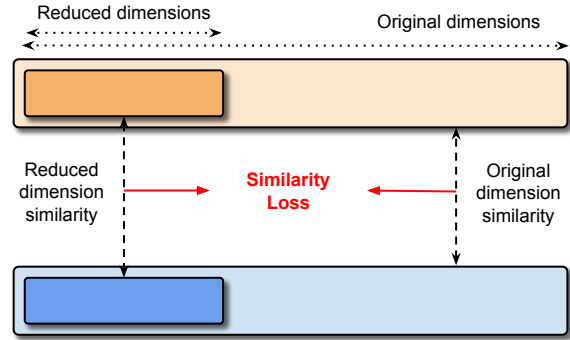

Figure 2: Similarity loss is a measure of the discrepancy between the similarity of two embeddings in their original dimensional space and their similarity in a reduced dimensional space. If the orange and blue embeddings are chosen randomly, this loss is referred to as pairwise similarity loss. If the orange and blue embeddings are selected based on similarity in their original dimensional space (top-k nearest embeddings), this loss is referred to as top-k similarity loss. Note that top-k similarity loss focuses on preserving local similarity relationships among neighboring embeddings.

mental characteristic of Matryoshka embeddings is their capacity to preserve the essential properties of the original embeddings, even within a reduced-dimensional space.

3.2 Tuning objective functions

The proposed Matryoshka-Adaptor, represented by the function $f : \mathbb{R}^d \rightarrow \mathbb{R}^d$, is designed to modify the original embeddings in order to enhance their inherent Matryoshka properties. We define the set of customized corpus embeddings as $\hat{\mathcal{C}}_E = \{\hat{ce}_1, \hat{ce}_2, ..., \hat{ce}_N\}$ and their corresponding Matryoshka embeddings as $\hat{\mathcal{C}}_E[: m] = \{\hat{ce}_1[: m], \hat{ce}_2[: m], ..., \hat{ce}_N[: m]\}$ where $\hat{ce}_i = f(ce_i)$. The primary objective of the function f is to maximize the Matryoshka properties through this customization process. This means ensuring the similarity between any two embeddings remains as consistent as possible, whether they are represented in the original high-dimensional space or the reduced low-dimensional space.

To achieve this objective, we introduce two loss functions. The first loss function, denoted as $\mathcal{L}_{pair}$, is designed to preserve the pairwise similarity between the original embeddings in their reduced-dimension Matryoshka form, expressed as:

$$\mathcal{L}_{pair} = \sum_i \sum_j \sum_m |Sim(ce_i, ce_j) - \quad (1)$$
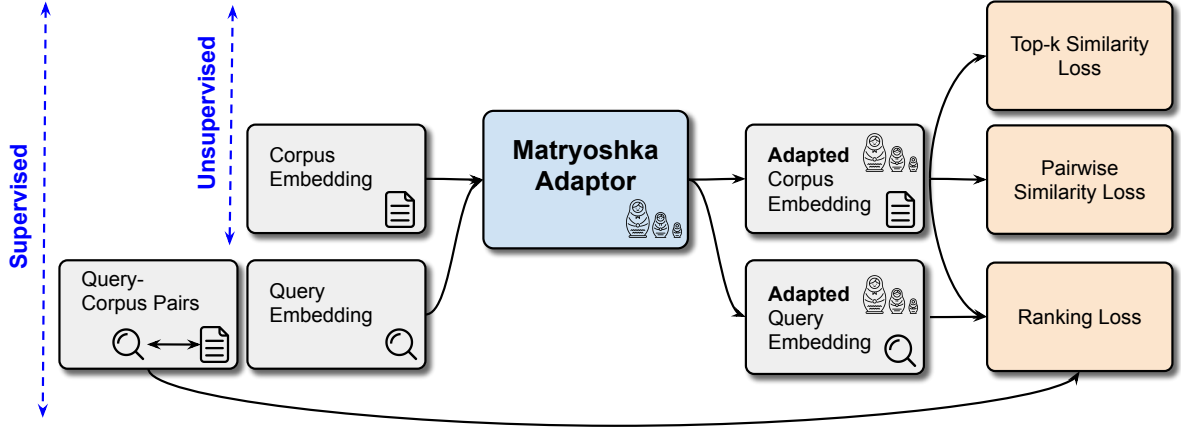$$Sim(f(ce_i)[: m], f(ce_j)[: m])|,$$

Figure 3: Block diagrams illustrating both the unsupervised and supervised Matryoshka-Adaptor frameworks. *Unsupervised Matryoshka-Adaptor*: This variant exclusively utilizes corpus embeddings as input. The training of the adaptor is achieved through a combination of top-k similarity loss and pairwise loss, which are calculated across multiple Matryoshka embeddings with various reduced dimensions. *Supervised Matryoshka-Adaptor*: In this variant, query embeddings and query-corpus pairs are provided as supplementary inputs. A ranking loss is incorporated alongside the top-k and pairwise losses to facilitate the training of the adaptor. Similar to the unsupervised setting, all losses are computed across Matryoshka embeddings with various reduced dimensions.

where Sim represents an arbitrary similarity function, which is chosen to be the cosine similarity.

The second loss function, denoted as $\mathcal{L}_{topk}$, focuses on preserving local similarity relationships among neighboring embeddings:

$$\mathcal{L}_{topk} = \sum_i \sum_{j \in NN_k(i)} \sum_m |Sim(ce_i, ce_j) - \quad (2)$$
$$Sim(f(ce_i)[:m], f(ce_j)[:m])|,$$

where $NN_k(i)$ denotes the set of the top k most similar embeddings to ce_i. A visual representation of these two loss functions, illustrating their application across multiple Matryoshka embeddings of varying dimensions, is provided in Fig. 2.

In order to mitigate any substantial deviation from the original embeddings, we have integrated regularizations into our methodology. Primarily, a skip connection is implemented within the architecture of the learnable function, f, ensuring that this function learns solely the difference from the original embedding, represented as $\hat{ce}_i = ce_i + f(ce_i)$. Furthermore, a reconstruction loss, denoted as $\mathcal{L}_{rec}$, is introduced as an additional regularizer:

$$\mathcal{L}_{rec} = \sum_i |ce_i - f(ce_i)|. \quad (3)$$

The overall objective function, designed to minimize the aggregate loss, is given as:

$$\min_f \mathcal{L}_{topk}(f) + \alpha \mathcal{L}_{pair}(f) + \beta \mathcal{L}_{rec}(f), \quad (4)$$

with $\alpha, \beta > 0$ being hyperparameters. Within the context of the unsupervised tuning setting, we fix their values as $\alpha = 1.0, \beta = 1.0$.

4 Supervised Matryoshka-Adaptor

4.1 Problem formulation

For many information retrieval applications, the availability of pairwise data, which indicates the relevance between specific queries and corpora, can considerably improve retrieval performance (Yoon et al., 2023). In this section, we introduce a method to further refine the Matryoshka-Adaptor by utilizing (a limited number of) such paired samples.

Let $\mathcal{Q} = \{q_1, q_2, ..., q_M\}$ denote the set of queries, and let $R = \{(q_i, c_j, y_{ij})\}_{i=1:M, j=1:N}$ represent the set of query-corpus relevance triplets, where $y_{ij} > 0$ signifies the relevance score between query q_i and corpus c_j. The query embeddings extracted from the model (E) are denoted as $\mathcal{Q}_E = \{qe_1, qe_2, ..., qe_N\}$ with each query embedding vector $qe_i = E(q_i) \in \mathbb{R}^d$. Furthermore, we define Matryoshka query embeddings as $\mathcal{Q}_E[:m] = \{qe_1[:m], qe_2[:m], ..., qe_N[:m]\}$, where $qe_i[:m] \in \mathbb{R}^m$. Note that m can be any integer less than the original embedding dimension.

4.2 Tuning objective functions

In the supervised setting, the Matryoshka-Adaptor (f) undergoes optimization to enhance both its Matryoshka properties and the retrieval performance

(a) OpenAI text-embedding-3-large   (b) OpenAI text-embedding-3-small   (c) Google multimodal embeddings
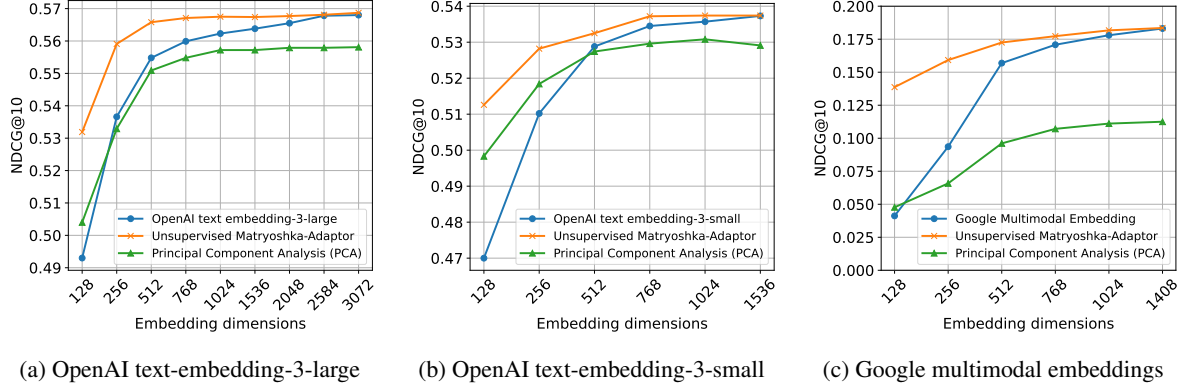
Figure 4: Experimental results of the unsupervised Matryoshka-Adaptor applied to three different embedding models: OpenAI text-embedding-3-large (with 3072 dimensions), OpenAI text-embedding-3-small (with 1536 dimensions), and Google multimodal (with 1408 dimensions). Text embedding results were obtained using 8 BEIR datasets, while multimodal embedding results were obtained using 5 Fashion-200K datasets.

of the Matryoshka embeddings. This is accomplished through the utilization of paired query-corpus samples, in conjunction with the original query and corpus embeddings. Matryoshka ranking loss, denoted as $\mathcal{L}_{rank}$, is introduced to align the ranking between query and corpus considering different Matryoshka embedding dimensions:

$$\mathcal{L}_{rank} = \sum_i \sum_j \sum_k \sum_m I(y_{ij} > y_{ik}) \quad (5)$$

$$(y_{ij} - y_{ik}) \log(1 + \exp(s_{ik}[:m] - s_{ij}[:m])),$$

where $s_{ij}[:m]$ represents the cosine similarity between the adapted query embedding $\hat{q}e_i[:m]$ (where $\hat{q}e_i = qe_i + f(qe_i)$) and the adapted corpus embedding $\hat{c}e_j[:m]$. We use the same adaptor (f) for both query and corpus embeddings. This ranking loss is crucial for effective learning of lower dimensional representations with their information content for the ranking objective being considered.

The supervised Matryoshka-Adaptor is trained using a joint objective function that encompasses the ranking loss as well as the unsupervised Matryoshka losses ($\mathcal{L}_{topk}$, $\mathcal{L}_{pair}$, and $\mathcal{L}_{rec}$). This joint training approach aims to improve the quality of the embeddings while preserving their Matryoshka representations. Query-corpus pairs are employed for the ranking loss, while query and corpus embeddings are utilized for the Matryoshka representation learning. The overall objective function is:

$$\min_f \mathcal{L}_{topk}(f) + \alpha \mathcal{L}_{pair}(f) + \quad (6)$$

$$\beta \mathcal{L}_{rec}(f) + \gamma \mathcal{L}_{rank}(f),$$

with $\alpha, \beta, \gamma \geq 0$ being hyper-parameters with fixed values as $\alpha = 1.0$, $\beta = 1.0$ and $\gamma = 1.0$.

To improve convergence, two-stage training strategy is employed. Initially, the Matryoshka-Adaptor is trained in an unsupervised way using Eq. 4. Subsequently, further tuning is conducted in a supervised way, utilizing Eq. 6.

5 Experiments

5.1 Experimental settings

The Matryoshka-Adaptor is extensively evaluated across a diverse set of 13 BEIR datasets (bei), 17 MIRACL datasets (mir), and 5 Fashion-200K datasets (Han et al., 2017). Such a comprehensive evaluation highlights the data-agnostic nature of the Matryoshka-Adaptor. Query and corpus embeddings are generated using state-of-the-art models, including Google Gecko text embeddings (English and multilingual) (gcp), Google multimodal embeddings, and OpenAI text embeddings (ope). This further highlights the model-agnostic nature of the proposed Matryoshka-Adaptor.

During the evaluation phase, both query and corpus embeddings are transformed using the trained Matryoshka-Adaptor. Cosine similarity is then computed between the transformed query and corpus embeddings across a spectrum of reduced dimensions (e.g., $d = \{8, 16, 32, 64, 128, 256, 512, 768\}$ for Gecko text embeddings). Retrieval performance is evaluated using the normalized discounted cumulative gain at rank 10 (nDCG@10) metric (Järvelin and Kekäläinen, 2002), facilitating a comprehensive assessment of performance across various Matryoshka embedding dimensions. All reported results represent the average value across the datasets. Data
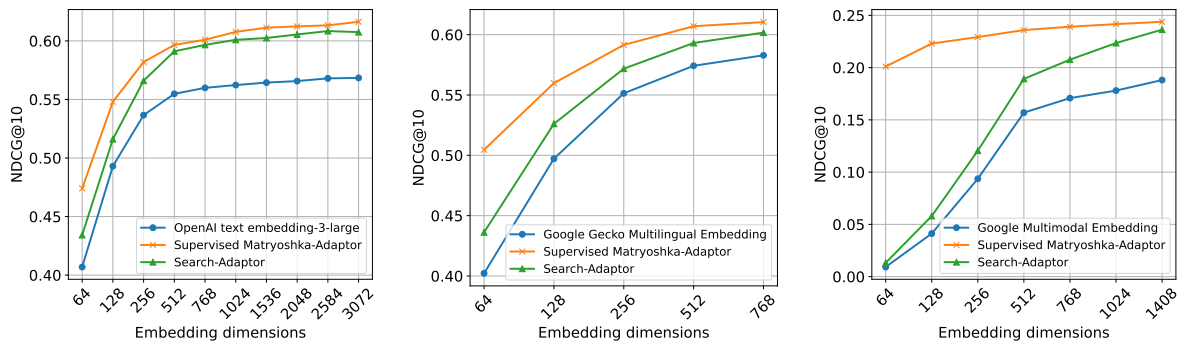
(a) OpenAI text-embedding-3-large   (b) Google gecko multilingual embeddings   (c) Google multimodal embeddings

Figure 5: Experimental results of the supervised Matryoshka-Adaptor on retrieval tasks, utilizing three different embedding models: OpenAI text-embedding-3-large (on 8 BEIR datasets), Google Gecko multilingual (on 17 MIRACL datasets), and Google multimodal (on 5 Fashion-200K datasets). Additional results are in Appendix. D.

statistics and hyper-parameters used in the experiments can be found in Appendix. A and B. Detailed results can be found in Appendix. E and F.

5.2 Unsupervised tuning

The Matryoshka-Adaptor can be applied exclusively with corpus embeddings, referred to as the unsupervised setting, enabling customization of embeddings solely on the corpus side. In this subsection, we present the impact of the unsupervised Matryoshka-Adaptor on two OpenAI text embedding models and compare its performance with the commonly-used unsupervised dimensionality reduction method, Principal Component Analysis (PCA) (Jolliffe and Cadima, 2016).

Fig. 4a and 4b illustrate that the Matryoshka-Adaptor yields significant performance improvements, particularly for lower dimensions, compared to the embeddings of the same dimensionality without it. Furthermore, lower dimensional embeddings processed with Matryoshka-Adaptor achieve comparable performance to original embeddings of high dimensionality. The adaptor achieves a faster saturation in performance with embedding dimensionality, towards the retrieval performance of original embeddings, underscoring its substantial impact in significantly reducing latency and memory requirements for retrieval applications.

Notably, the latest OpenAI embeddings are already trained with Matryoshka Representation Learning (MRL) (ope). The additional performance gains achieved by Matryoshka-Adaptor are attributed to the tuning process. With PCA, some improvements are observed for lower dimensions, however, at higher dimensions, noticeable performance degradation occurs, resulting in performance inferior even to the original embedding. This highlights the superiority of the Matryoshka-Adaptor, designed with the goal of salient information preservation for similarity within lower dimensions compared to PCA, a generic unsupervised dimensionality reduction approach.

5.3 Supervised tuning

The Matryoshka-Adaptor can be applied in supervised learning setup where a limited number of query-corpus pairs are available. In this context, the Supervised Matryoshka-Adaptor is trained utilizing paired query-corpus data. The effectiveness of the Supervised Matryoshka-Adaptor is evaluated on 13 BEIR, 17 MIRACL, and 5 Fashion-200K datasets, with query and corpus embeddings being generated using the latest Google Gecko models, including its multilingual and multimodal versions.

As illustrated in Fig. 5, the Supervised Matryoshka-Adaptor consistently outperforms the alternatives, such as Search-Adaptor (Yoon et al., 2023), particularly for lower embedding dimensions. Additionally, lower dimensional embeddings processed with the Supervised Matryoshka-Adaptor perform comparably to high dimensional embeddings, showcasing its potential to significantly reduce latency and memory requirements for applications like retrieval.

5.4 Tuning for Multimodal Embeddings

As previously established, the Matryoshka-Adaptor framework is not confined to text embeddings but can be generalized to multimodal embeddings as well. To illustrate this capability, we applied the Matryoshka-Adaptor to the latest gecko-multimodal embeddings, utilizing the Fashion-
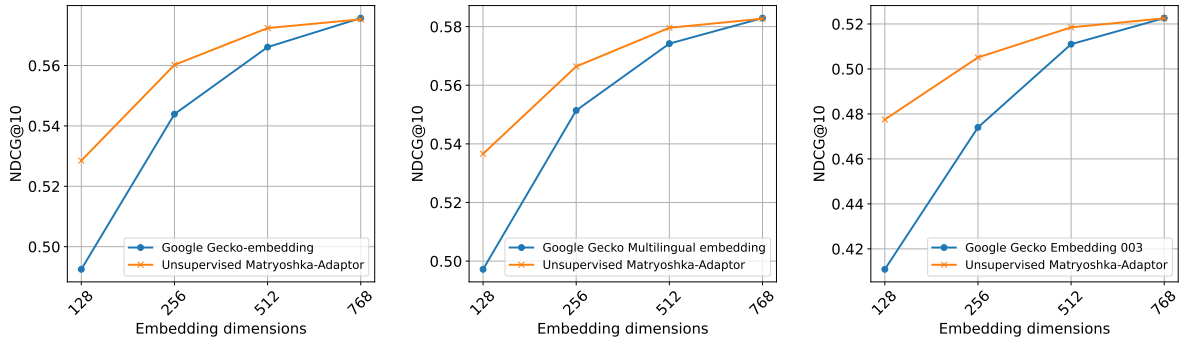
| (a) Google gecko-latest embeddings | (b) Google gecko-multilingual embeddings | (c) Google gecko-003 embeddings |

Figure 6: Experimental results of the unsupervised Matryoshka-Adaptor with three different text embedding models: Google Gecko, Google Gecko multilingual, and Google Gecko-003 (which is not trained with the Matryoshka Representation Learning technique).

200K dataset, which comprises 5 sub-datasets designed for text-to-image retrieval tasks.

Fig. 4c and 5c demonstrate the effectiveness of the Matryoshka-Adaptor in consistently improving the performance of multimodal base embedding models for text-to-image retrieval tasks. These highlight that the Matryoshka-Adaptor significantly outperforms alternative methods such as PCA, in unsupervised learning setups and Search-Adaptor in supervised learning setups, particularly when lower embedding dimensions are considered.

5.5 Tuning for Multilingual Embeddings

Matryoshka-Adaptor is not only model-agnostic but also data-agnostic. Its applicability even extends beyond a single language. To validate this, we evaluate the performance of Matryoshka-Adaptor on MIRACL datasets, which comprise 17 non-English languages.

Fig. 6b and 5b present the results of applying Matryoshka-Adaptor to multilingual retrieval tasks. The findings demonstrate that the performance gains achieved through the proposed tuning method are not limited to English but also extend to non-English language datasets. Furthermore, the improvements are observed to be model-agnostic, as evident from the successful application of Matryoshka-Adaptor to the latest Gecko multilingual embedding models.

6 Discussions

6.1 Models that are not pretrained with MRL

A significant advantage of the Matryoshka-Adaptor framework lies in its broad applicability to a wide array of embedding models. We demonstrate the

efficacy of Matryoshka-Adaptor when applied to embedding models that have not been trained using MRL. Specifically, we utilize earlier versions of the Gecko embedding models (Google Gecko-003), which do not utilize MRL in their pretraining, in conjunction with BEIR datasets to illustrate the impact of the Unsupervised Matryoshka-Adaptor.

Fig. 6c presents evidence of the consistent performance improvements achieved by the Matryoshka-Adaptor when applied to non-MRL trained embedding models. This observation underscores that the performance gains of Matryoshka-Adaptor originate from customizing the embedding to the specific corpus, and this beneficial impact can be extended to any embedding model, even when they had been trained with MRL.

6.2 Ablation studies

To examine the individual contribution of each loss component to the overall performance of Matryoshka-Adaptor, we conduct ablation studies. In the unsupervised setting, each of the top-k loss ($\mathcal{L}_{topk}$), pairwise loss ($\mathcal{L}_{pair}$), and reconstruction loss ($\mathcal{L}_{rec}$) is individually excluded from the original loss summation in Eq. 4, and the corresponding performance degradation is monitored. Specifically, the pairwise loss was excluded by setting $\alpha = 0$, and the reconstruction loss is excluded by setting $\beta = 0$. In the supervised setting, we utilize only the ranking loss ($\mathcal{L}_{rank}$) to assess the impact of three unsupervised losses in Eq. 6.

As evident in Table 1, each loss function contributes uniquely to the overall performance of Matryoshka-Adaptor in the unsupervised setting. Notably, $\mathcal{L}_{topk}$ is particularly beneficial for higher dimensions. In the supervised setting, the incor-

(a) Pairwise and top-k distance metrics   (b) nDCG@10 vs pairwise distance   (c) nDCG@10 vs Top-k distance
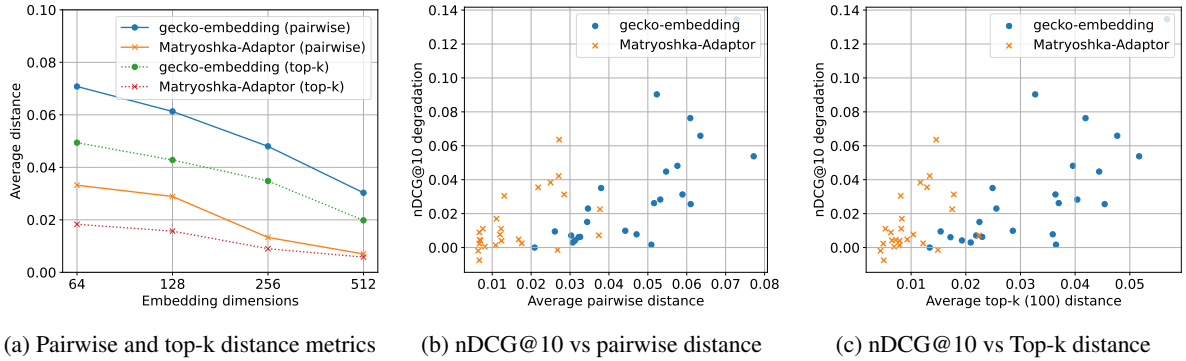
Figure 7: Analysis of distance metrics in unsupervised settings. (a) Average pairwise and top-k distances across varying embedding dimensions, compared to base embeddings. (b) Correlation between nDCG@10 and average pairwise distance. (c) Correlation between nDCG@10 and average top-k distance.

Unsuperivsed Matryoshka-Adaptor (nDCG@10)				
Reduced dims	64	128	256	512
Baseline	0.4332	0.5044	0.5461	0.5590
All three losses	**0.4845**	**0.5380**	**0.5580**	**0.5652**
w/o $\mathcal{L}_{topk}$	0.4798	0.5236	0.5463	0.5630
w/o $\mathcal{L}_{pair}$	0.4745	0.5230	0.5423	0.5598
w/o $\mathcal{L}_{rec}$	0.4824	0.5342	0.5477	0.5621
Supervised Matryoshka-Adaptor (nDCG@10)				
Reduced dims	64	128	256	512
Baseline	0.4332	0.5044	0.5461	0.5590
All four losses	**0.5047**	**0.5473**	**0.5714**	**0.5902**
Only with $\mathcal{L}_{rank}$	0.4767	0.5209	0.5607	0.5787

Table 1: Ablation studies on both supervised and unsupervised Matryoshka-Adaptor across 8 BEIR datasets. The best performance in each scenario is in bold.

poration of the additional unsupervised losses, including $\mathcal{L}_{topk}$ and $\mathcal{L}_{pair}$, is critical for achieving performance improvements. Without these unsupervised losses, the performance gain diminishes by approximately 50%.

6.3 Relationship with distance metrics

In unsupervised learning setup, the absence of labeled data presents a challenge in evaluating the impact of Matryoshka-Adaptor. Towards circumventing this issue and shedding light on the impact of dimensionality reduction, we analyze the average pairwise distances and top-k distances within the corpus embeddings on the disjoint validation corpus. Subsequently, we examine the relationship between these unsupervised metrics and the supervised retrieval metric nDCG@10, utilizing 8 BEIR datasets with Google Gecko embedding models.

Fig. 7a reveals that the unsupervised Matryoshka-Adaptor results in significantly lower

distance metrics compared to the original embeddings. This suggests that the proposed method effectively preserves both pairwise and top-k distances among corpus embeddings when employing a reduced subset of embedding dimensions. These distance metrics can serve as viable unsupervised proxies for assessing the efficacy of the unsupervised Matryoshka-Adaptor. To further elucidate this relationship, we also show the correlation between these distance metrics and nDCG@10 (Fig. 7b and 7c). A strong correlation is observed, indicating that lower distance metrics are associated with smaller performance drop in retrieval tasks, even when using reduced embedding dimensions. This suggests that the unsupervised distance metrics can be indicators of retrieval performance, even in the absence of labeled data.

7 Conclusions

While high-dimensional embeddings often provide a richer representation of the original data, their practical deployment in real-world applications is frequently hampered by computational costs and latency issues. As a result, many applications opt for lower-dimensional embeddings, accepting a compromise in performance. The presented Matryoshka-Adaptor framework offers a solution to this dilemma by enabling substantial dimensionality reduction while minimizing performance degradation, in both supervised and unsupervised scenarios. Furthermore, due to its versatile nature and applicability to any embedding model, regardless of access to model parameters, the Matryoshka-Adaptor constitutes a valuable tool for improving the efficiency and practicality of embedding-based applications.

8 Limitations and Future Works

In the unsupervised setting, picking the optimal hyperparameters for Matryoshka-Adaptor poses a challenge due to the absence of validation data. While alternative metrics, such as the proposed distance metrics illustrated in Fig. 7, can be employed, these may exhibit higher levels of noise compared to supervised validation metrics. One major risk associated with Matryoshka-Adaptor lies in the potential for overfitting to the tuning data, during the tuning process. Future research endeavors might include extending the proposed approach to encompass multiple modalities in the tuning objective; exploring semi-supervised variations; and investigating simultaneous utilization of multiple datasets during tuning.

References

Beir data repository. https://github.com/beir-cellar/beir.

Google cloud text embedding. https://cloud.google.com/vertex-ai/docs/generative-ai/embeddings/get-text-embeddings.

Miracl data repository. https://project-miracl.github.io/.

Openai text embedding. https://https://openai.com/index/new-embedding-models-and-api-updates.

Akari Asai, Timo Schick, Patrick Lewis, Xilun Chen, Gautier Izacard, Sebastian Riedel, Hannaneh Hajishirzi, and Wen-tau Yih. 2022. Task-aware retrieval with instructions. *arXiv preprint arXiv:2211.09260*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Mu Cai, Jianwei Yang, Jianfeng Gao, and Yong Jae Lee. 2024. Matryoshka multimodal models. *arXiv preprint arXiv:2405.17430*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Zhuyun Dai, Vincent Y Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith B Hall, and Ming-Wei Chang. 2022. Promptagator:

Few-shot dense retrieval from 8 examples. *arXiv preprint arXiv:2209.11755*.

Xintong Han, Zuxuan Wu, Phoenix X Huang, Xiao Zhang, Menglong Zhu, Yuan Li, Yang Zhao, and Larry S Davis. 2017. Automatic spatially-aware fashion concept discovery. In *Proceedings of the IEEE international conference on computer vision*, pages 1463–1471.

Aapo Hyvärinen and Erkki Oja. 2000. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Ian T Jolliffe and Jorge Cadima. 2016. Principal component analysis: a review and recent developments. *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, et al. 2022. Matryoshka representation learning. *Advances in Neural Information Processing Systems*, 35:30233–30249.

Daniel Lee and H Sebastian Seung. 2000. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems*, 13.

Jinhyuk Lee, Zhuyun Dai, Xiaoqi Ren, Blair Chen, Daniel Cer, Jeremy R Cole, Kai Hui, Michael Boratko, Rajvi Kapadia, Wen Ding, et al. 2024. Gecko: Versatile text embeddings distilled from large language models. *arXiv preprint arXiv:2403.20327*.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Geoffrey J McLachlan. 2005. *Discriminant analysis and statistical pattern recognition*. John Wiley & Sons.

Niklas Muennighoff, Hongjin Su, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. 2024. Generative representational instruction tuning. *arXiv preprint arXiv:2402.09906*.

Jianmo Ni, Gustavo Hernández Ábrego, Noah Constant, Ji Ma, Keith B Hall, Daniel Cer, and Yinfei Yang. 2021. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. *arXiv preprint arXiv:2108.08877*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al.

2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Hongjin Su, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A Smith, Luke Zettlemoyer, Tao Yu, et al. 2022. One embedder, any task: Instruction-finetuned text embeddings. *arXiv preprint arXiv:2212.09741*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2023. Improving text embeddings with large language models. *arXiv preprint arXiv:2401.00368*.

Jinsung Yoon, Sercan O Arik, Yanfei Chen, and Tomas Pfister. 2023. Search-adaptor: Text embedding customization for information retrieval. *arXiv preprint arXiv:2310.08750*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022a. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Xinyu Zhang, Nandan Thakur, Odunayo Ogundepo, Ehsan Kamalloo, David Alfonso-Hermelo, Xiaoguang Li, Qun Liu, Mehdi Rezagholizadeh, and Jimmy Lin. 2022b. Making a miracl: Multilingual information retrieval across a continuum of languages. *arXiv preprint arXiv:2210.09984*.

A Data Statistics

A.1 BEIR datasets

Datasets	Number of train pairs	Number of test pairs	Number of corpus
NFCorpus	110575	12334	3633
SciFact	919	339	5183
Arguana	703	703	8674
SciDocs	14972	14956	25657
FiQA	14166	1706	57638
Trec-Covid	35460	30876	171332
Touche	1077	1137	382545
Quora	7626	15675	522931
NQ	2097	2104	2681468
DBPedia	5673	43515	4635922
HotPotQA	170000	14810	5233329
Fever	140085	7937	5416568
Climate-fever	2299	2382	5416593

Table 2: The statistics of 13 BEIR datasets (sorted by the number of corpus).

A.2 MIRACL datasets

Datasets	Number of train pairs	Number of test pairs	Number of corpus
Yoruba (yo)	959	229	49043
Swahilli (sw)	9359	5092	131924
Bengali (bn)	16754	4206	297265
Hindi (hi)	11668	3494	506264
Telugu (te)	18608	1606	518079
Thai (th)	21293	7573	542166
Indonesian (id)	41358	9668	1446315
Korean (ko)	12767	3057	1486752
Finnish (fi)	20350	12008	1883509
Arabic (ar)	25382	29197	2061414
Persian (fa)	21844	6571	2207172
Chinese (zh)	13113	3928	4934368
Japanese (ja)	34387	8354	6953614
Russian (ru)	33921	13100	9543918
Spanish (es)	21531	6443	10373953
French (fr)	11426	3429	14636953
Germany (de)	2526	628	15866222

Table 3: The statistics of 17 MIRACL datasets (sorted by the number of corpus).

A.3 Fashion-200K datasets

Datasets	Number of train pairs	Number of test pairs	Number of corpus
Dresses	15127	1567	72376
Jackets	8105	1511	71118
Pants	9264	1758	74470
Skirts	6822	1247	47931
Tops	13809	2536	72444

Table 4: The statistics of 5 Fashion-200K datasets.

B Hyper-parameters

We summarize the hyper-parameters used to train Matryoshka-Adaptor. In all experiments (in both unsupervised and supervised settings), we utilize the fixed hyper-parameters that enable applying Matryoshka-Adaptor without extensive hyper-parameter tuning.

Hyper-parameters	Fixed values
Pairwise loss coefficient (α)	1.0
Recovery loss coefficient (β)	1.0
Ranking loss coefficient (γ)	1.0
Batch size for training	128
Batch size for corpus during training	50000
Maximum number of training iterations	5000
Patience for early stopping	500
Learning rates	0.001
Optimizer	Adam

Table 5: Hyper-parameters used to train Matryoshka-Adaptor in all experiments.

C Computational complexity

All experimental procedures were conducted utilizing a single NVIDIA V100 GPU with 16GB of memory. In the unsupervised Matryoshka-Adaptor training regime, processing time did not exceed 10 minutes for datasets ranging from 3,000 to 10 million corpus samples. For supervised Matryoshka-Adaptor training, datasets with fewer than 1 million corpus samples were processed in under 30 minutes, while datasets with up to 10 million corpus samples were processed in less than one hour. It is important to note that the adaptor architecture consists of a shallow multi-layer perceptron, rendering the computational complexity of inference negligible.

D Additional results on supervised Matryoshka-Adaptor

(a) Google gecko-latest embeddings
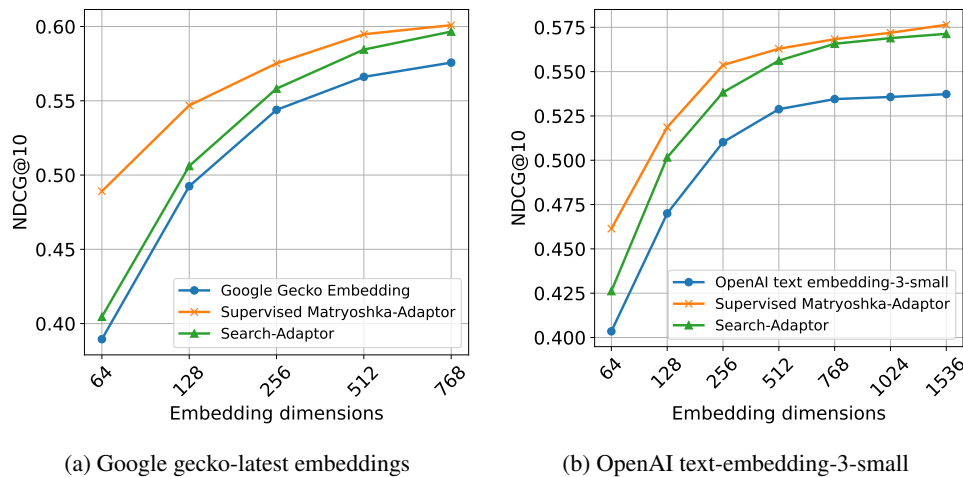
(b) OpenAI text-embedding-3-small

Figure 8: Experimental results of the supervised Matryoshka-Adaptor on retrieval tasks, utilizing two different embedding models: Google Gecko (on 13 BEIR datasets), and OpenAI text-embedding-3-small (on 8 BEIR datasets).

E Detailed Experimental Results on Unsupervised Settings

In this section, we present the experimental results per each dataset in unsupervised settings. In the main manuscript, we report the average values among the entire datasets.

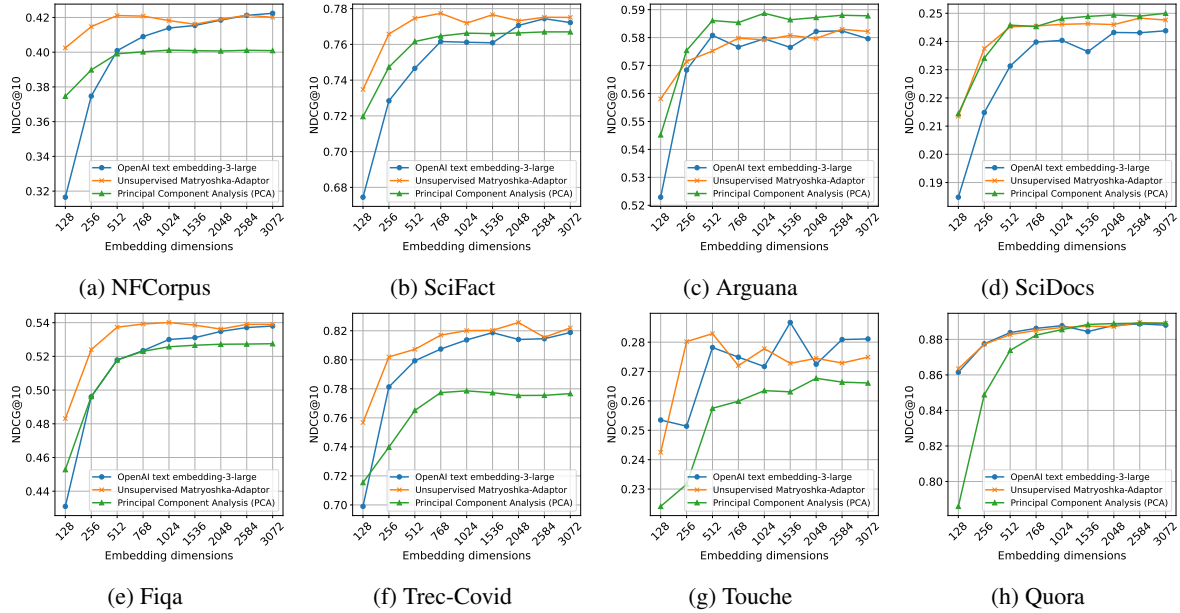## E.1 Unsupervised Matryoshka-Adaptor with OpenAI embedding models

Figure 9: Experimental results of unsupervised Matryoshka-Adaptor with OpenAI-textembedding-3-large on 8 BEIR datasets.
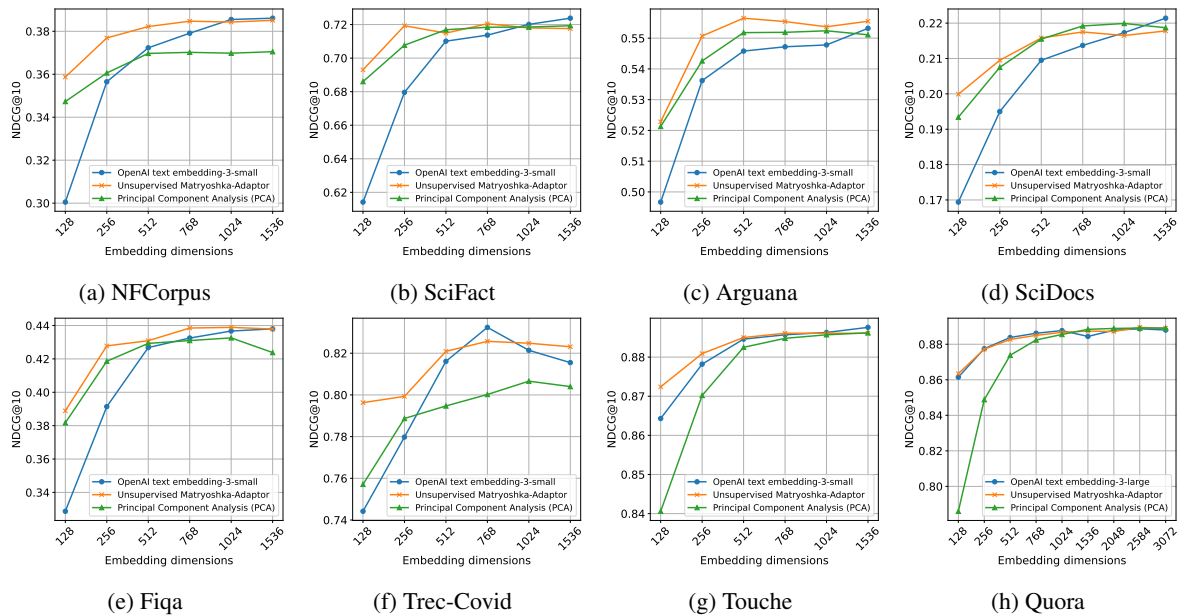

Figure 10: Experimental results of unsupervised Matryoshka-Adaptor with OpenAI-textembedding-3-small on 8 BEIR datasets.

E.2   Unsupervised Matryoshka-Adaptor with Google multimodal embedding models

(a) Dresses (b) Jackets (c) Pants (d) Skirts

(e) Tops
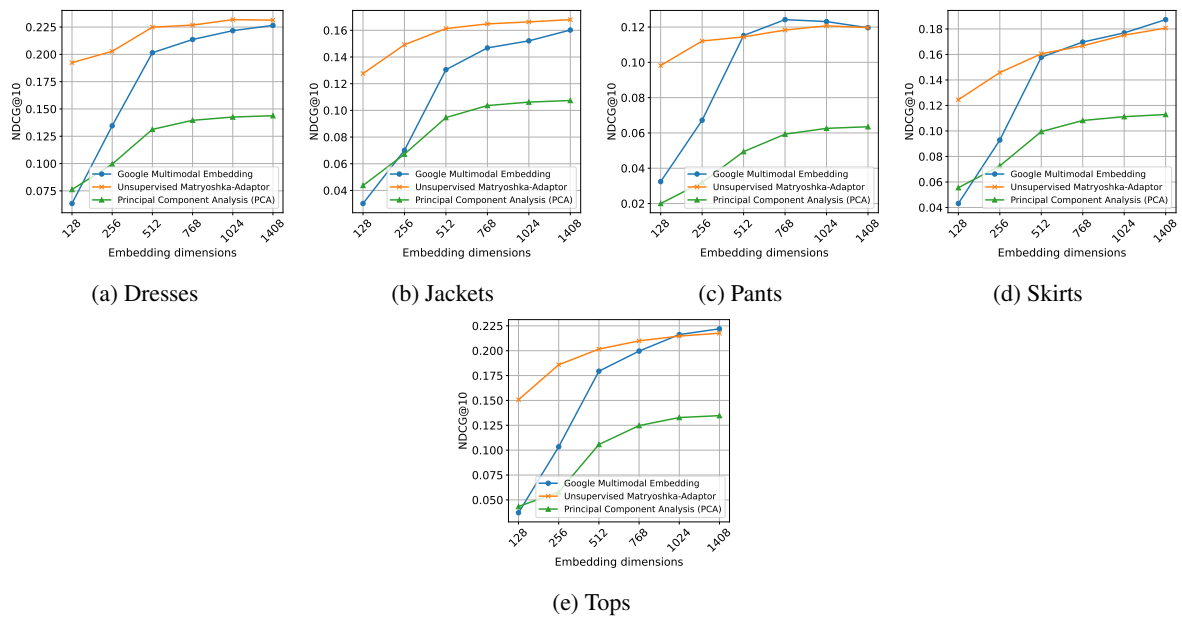
Figure 11: Experimental results of unsupervised Matryoshka-Adaptor with Google multimodal embedding models on 5 Fashion-200K datasets.

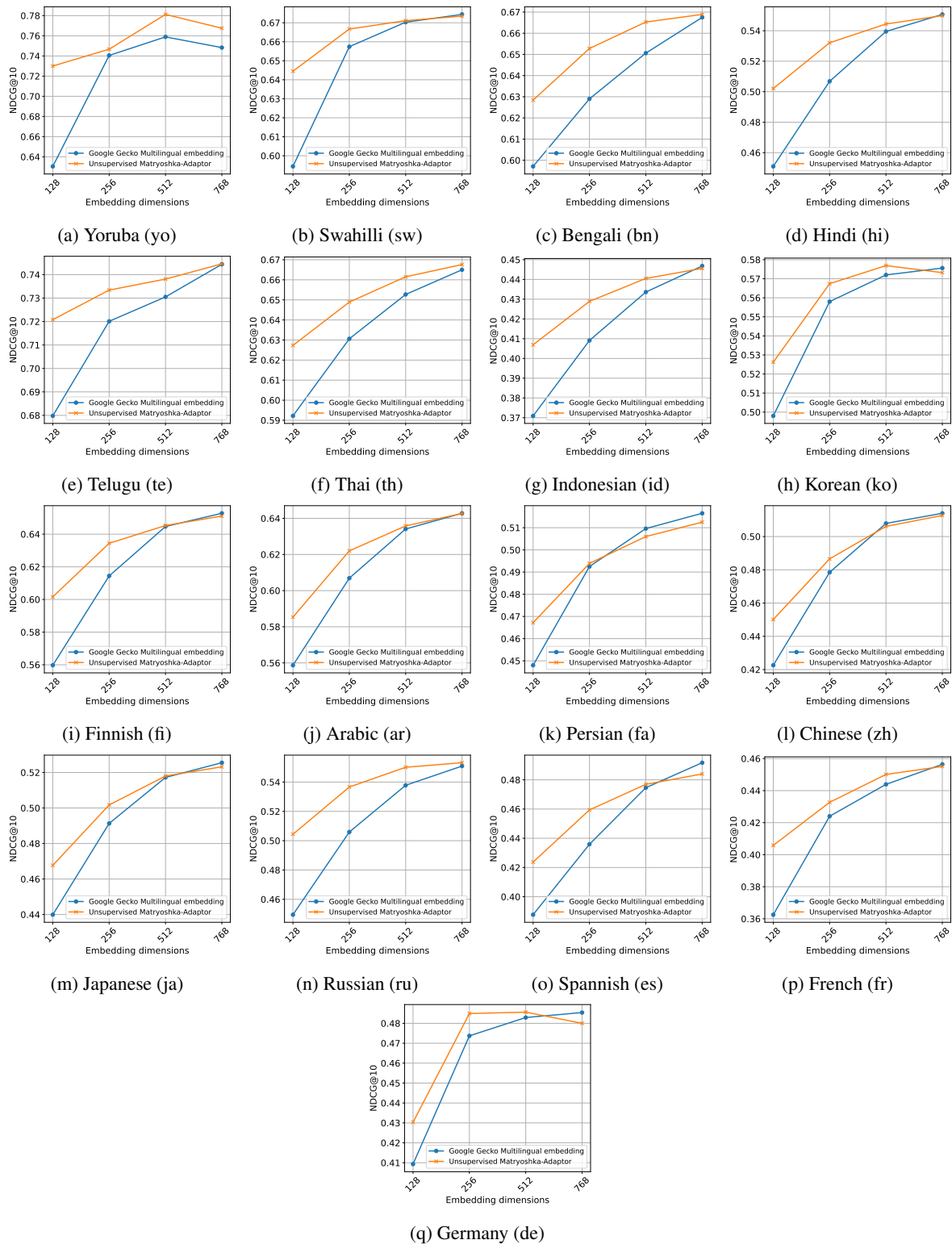## E.3 Unsupervised Matryoshka-Adaptor with Google Gecko multilingual embedding models

(a) Yoruba (yo)

(b) Swahilli (sw)

(c) Bengali (bn)

(d) Hindi (hi)

(e) Telugu (te)

(f) Thai (th)

(g) Indonesian (id)

(h) Korean (ko)

(i) Finnish (fi)

(j) Arabic (ar)

(k) Persian (fa)

(l) Chinese (zh)

(m) Japanese (ja)

(n) Russian (ru)

(o) Spannish (es)

(p) French (fr)

(q) Germany (de)

Figure 12: Experimental results of unsupervised Matryoshka-Adaptor with Google Gecko multilingual embedding models on 17 MIRACL datasets.

F   Detailed Experimental Results on Supervised Settings

In this section, we present the experimental results per each dataset in supervised settings. In the main manuscript, we report the average values among the entire datasets.

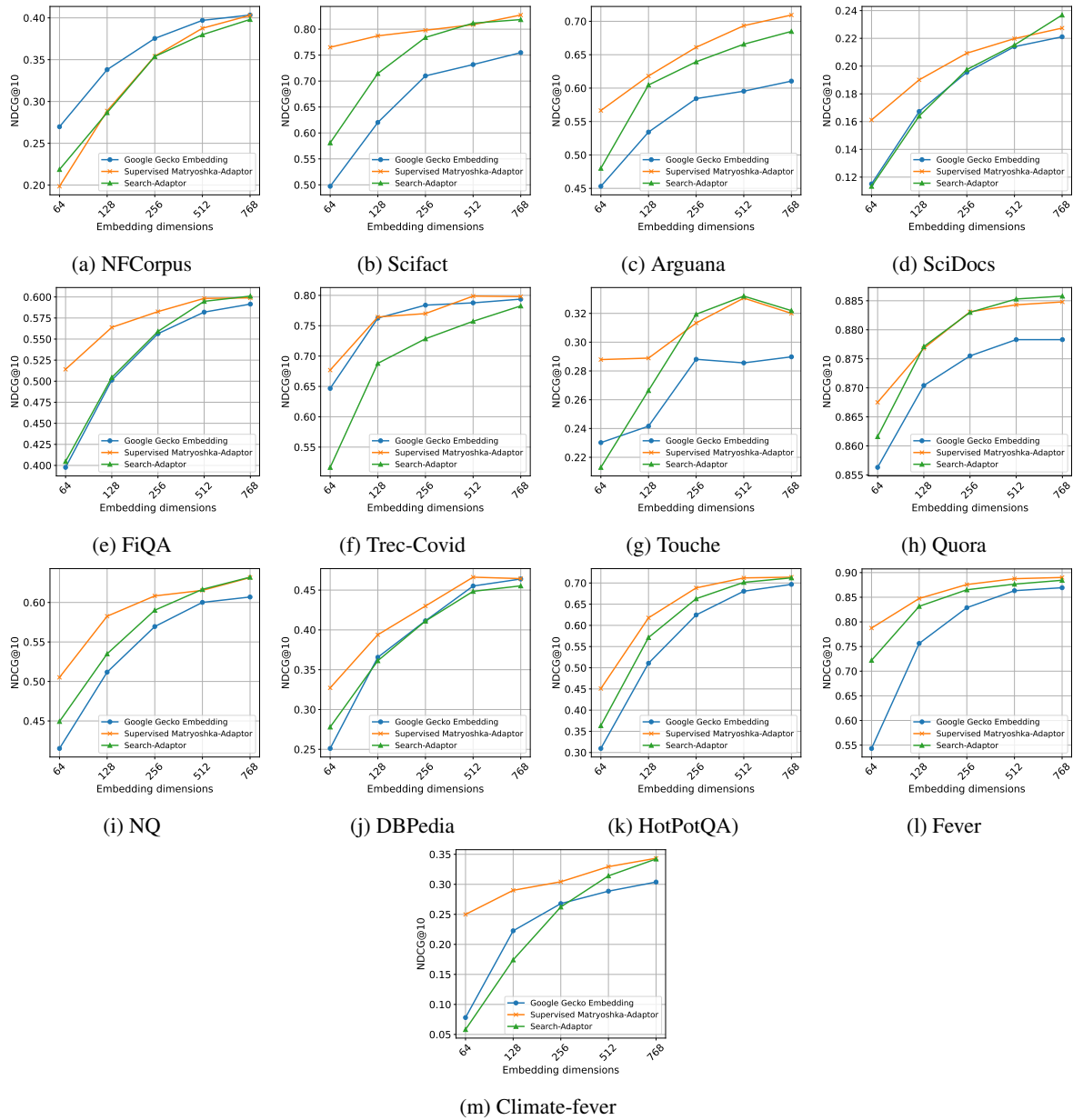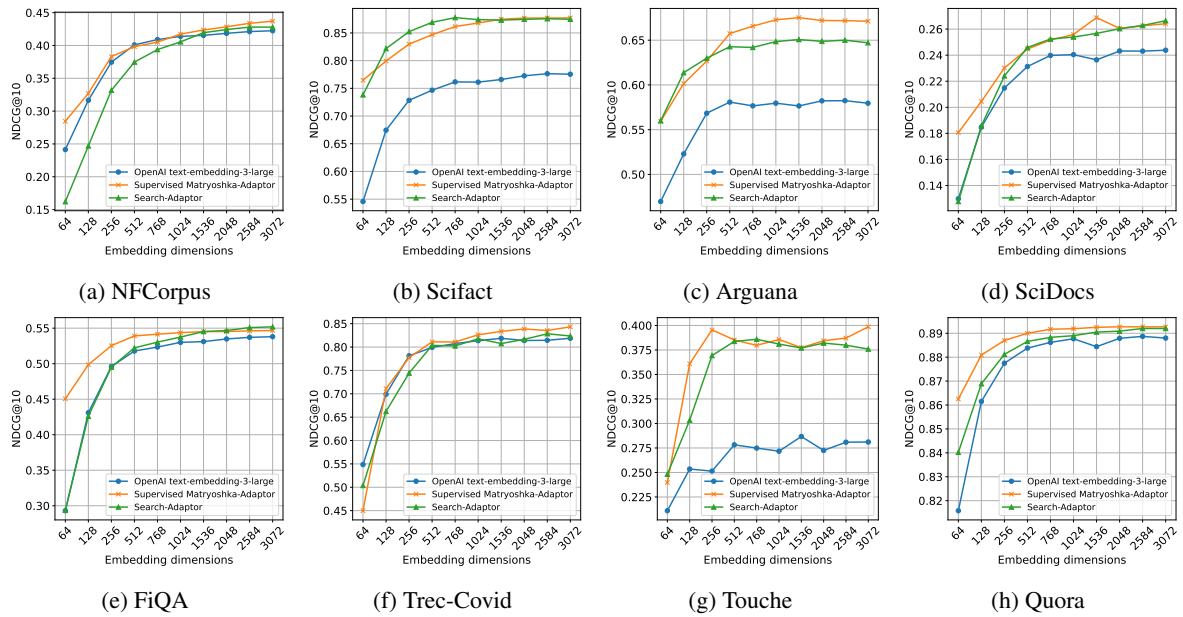## F.1   Supervised Matryoshka-Adaptor with Google Gecko embedding models

Figure 13: Experimental results of supervised Matryoshka-Adaptor with Google latest Gecko embedding models on 13 BEIR datasets.

F.2 Supervised Matryoshka-Adaptor with OpenAI text-embedding-3-large models

Figure 14: Experimental results of supervised Matryoshka-Adaptor with OpenAI text-embedding-3-large models on 8 BEIR datasets.

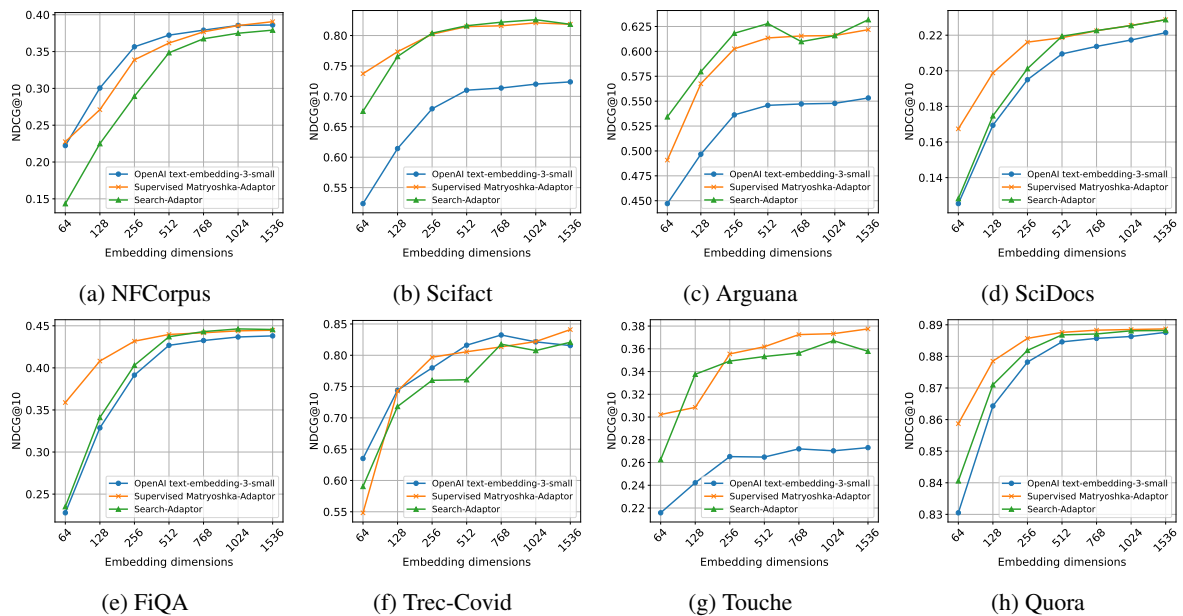## F.3 Supervised Matryoshka-Adaptor with OpenAI text-embedding-3-small models

Figure 15: Experimental results of supervised Matryoshka-Adaptor with OpenAI text-embedding-3-small models on 8 BEIR datasets.

F.4 Supervised Matryoshka-Adaptor with Google Gecko multilingual embedding models
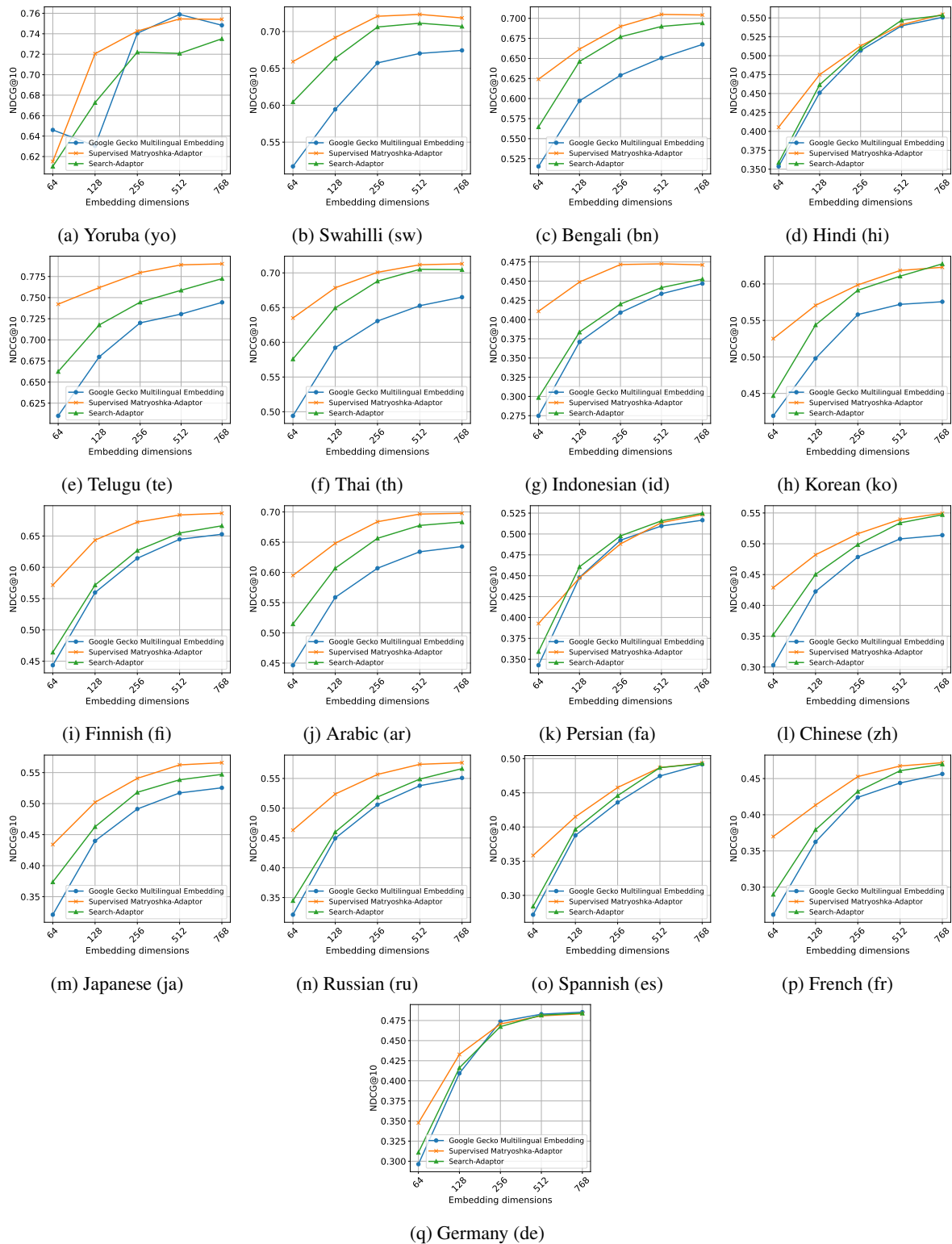

Figure 16: Experimental results of supervised Matryoshka-Adaptor with Google Gecko multilingual embedding models on 17 MIRACL datasets.

F.5 Supervised Matryoshka-Adaptor with Google multimodal embedding models

(a) Dresses

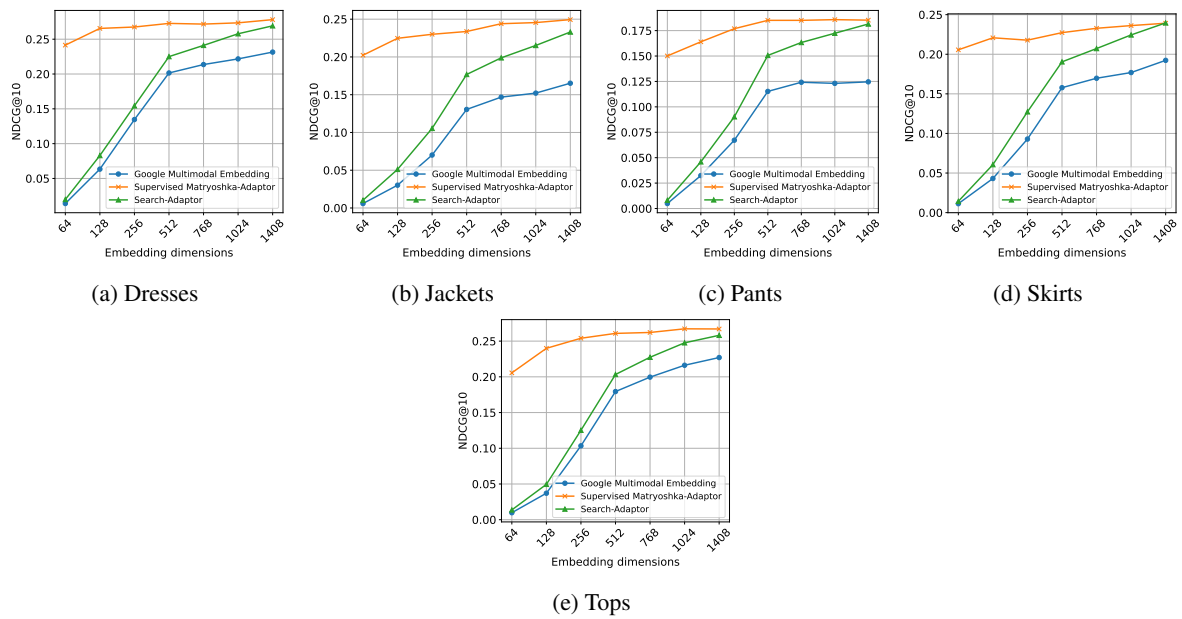(b) Jackets

(c) Pants

(d) Skirts

(e) Tops

Figure 17: Experimental results of supervised Matryoshka-Adaptor with Google multimodal embedding models on 5 Fashion-200K datasets.