To MRL or not to MRL: Text Embeddings are Robust to Truncation Without Matryoshka Learning, Except In Heavy Truncation Scenarios

Sotaro Takeshita[1], Yurina Takeshita[3], Simone Paolo Ponzetto[1], Daniel Ruffinelli[1,2]
[1]Data and Web Science Group, University of Mannheim, Germany
[2]NEC Laboratories Europe, Heidelberg, Germany
[3]Independent Researcher
{sotaro.takeshita, ponzetto, druffinelli}@uni-mannheim.de

Abstract

Matryoshka Representation Learning (MRL) is a widely adopted approach for training text encoders so they provide useful text representations at various sizes, available by simply truncating the resulting vectors at sizes predetermined at training time. Recent works have shown that randomly truncating text embeddings has minimal impact in downstream performance unless vectors are reduced in size by at least 70%, suggesting that embeddings are already robust to truncation without the use of MRL. However, no prior work has compared random truncation to MRL, so it is unclear how the two methods compare as effective embedding reduction methods. In this paper, we study this by applying the same truncation used by MRL to models trained with and without MRL. Our results across several models and downstream tasks show that, unless heavily truncating embeddings (i.e. reducing their size by at least 80%), truncated embeddings of non-MRL models are competitive with, and often outperform models trained with MRL. This suggests that truncation robustness may not necessarily come from MRL, and that the choice of spending the additional training cost of MRL depends on whether heavy truncation is desired. We make our code available for reproduction.[1]

1 Introduction

Text embeddings are widely used in many NLP tasks, from retrieval (Huang et al., 2020) to recommender systems (Wang et al., 2025a) to many others (Zhao et al., 2024). To reduce costs while retaining performance in such tasks, the use of small but well-performing embeddings is desirable. E.g. in text retrieval, bi-encoder models with less than 1B parameters are a common choice for first-stage retrieval over an entire collection of documents (Zhao et al., 2024). To provide more flexibility in this regard, Matryoshka Representation Learning (MRL) (Kusupati et al., 2022) is an approach that adds additional terms to the training objective so that text encoders are able to provide representations that are simultaneously useful at various embedding sizes, available by simply truncating the resulting vectors at sizes pre-determined before training. The use of MRL is widely adopted by most of the latest models, such as Jina-Embeddings-V5 (Akram et al., 2026), Qwen3-Embedding (Zhang et al., 2025c), and EmbeddingGemma (Vera et al., 2025). At the same time, recent works have empirically shown that randomly truncating embeddings is a promising approach to obtaining vectors of reduced size, as the impact in performance is minimal unless vector sizes are reduced by at least 70% (Tsukagoshi and Sasano, 2025; Takeshita et al., 2025; Inkiriwang et al., 2025). However, no prior work has compared random truncation to MRL, the industry standard for truncation, so it is unclear to what extent random truncation is an effective method for

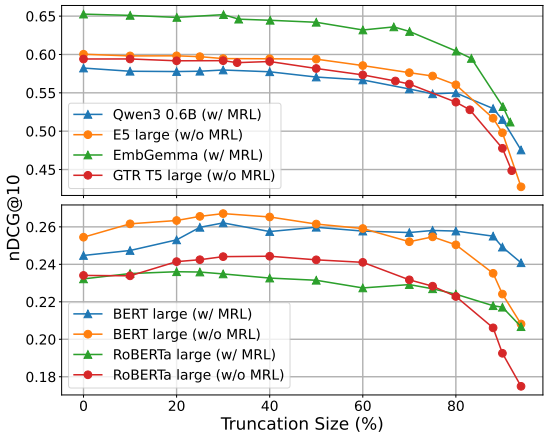


Figure 1: (Top) Robustness of open text encoders as truncation levels increase looks the same whether trained with or without MRL. (Bottom) When models differ only in their use of MRL, truncation on non-MRL models is superior unless heavy truncation is applied.

[1]https://sotaro.io/papers/mrl-or-not

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 52385–52404
October 24-29, 2026 ©2026 Association for Computational Linguistics

trading off performance for runtime and memory costs. In this short paper, we study and quantify this by comparing downstream performance at various vector truncation levels using embeddings from models trained with and without MRL. Our results show that unless heavily truncating embeddings, i.e. reducing their size by about 80%, truncated embeddings of non-MRL models are competitive with, and often superior in performance compared to models trained with MRL. We found this across 8 open models (e.g. Fig. 1, top), 10 text encoders we trained from pre-trained language models (e.g. Fig. 1, bottom) and 24 downstream tasks for classification and retrieval. Upon closer inspection, we found the variance of lower dimensions in Matryoshka embeddings to be much larger than non-MRL models, suggesting MRL does promote better information compression in these dimensions. This might relate to why MRL only brings clear benefits in heavy truncation settings, i.e. in scenarios with potentially more cost reductions, but also higher performance loss, Finally, we found that while MRL requires that truncation dimensions be chosen before training, MRL behaves the same as random truncation when truncating outside of those pre-selected dimensions. Our results suggest that (i) truncation robustness may not necessarily come from MRL but be inherent to the learned representations, (ii) that the additional cost of training with MRL is only beneficial in scenarios where reducing embedding size by about 80% or more is desirable, and (iii) that there may be ways to extend MRL's benefit beyond heavy truncation scenarios.

2 Background and Related Work

Matryoshka Representation Learning (MRL). Following Kusupati et al. (2022), we formalize MRL in a supervised setting. Let $f : T \rightarrow \mathbb{R}^d$ be a text encoder with parameters $\boldsymbol{\theta}$ that maps a sequence of tokens $x \in \mathcal{T}$ to a d-dimensional vector $\boldsymbol{e}$, i.e. $\boldsymbol{e} = f(x|\boldsymbol{\theta})$, where $\mathcal{T}$ is the set of all possible token sequences in a given language. Given labelled dataset $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$ and loss function $\mathcal{L}$, MRL optimizes the following objective:

$$\min_{\boldsymbol{\theta}, \mathcal{W}_\mathcal{M}} \frac{1}{N} \sum_{(x,y)\in\mathcal{D}} \sum_{m\in\mathcal{M}} c_m \cdot \mathcal{L}(\boldsymbol{W}_m \cdot f_m(x|\boldsymbol{\theta}), y)$$

where $\mathcal{M} = \{m_1, m_2, \ldots, m_{|M|}\}$ is a set of vector sizes $m_i \leq d$, $\mathcal{W}_\mathcal{M} = \{\boldsymbol{W}_m\}_{m\in\mathcal{M}}$ the set of learnable classifiers for each $m \in \mathcal{M}$, c_m a scaling factor for the loss term corresponding to vector size m, and $f_m(x|\boldsymbol{\theta}) = f(x|\boldsymbol{\theta})_{1:m}$ the output vector of encoder f truncated to size m. In other words, MRL applies empirical risk minimization over the set of classifiers $\mathcal{W}_\mathcal{M}$, where each uses a different truncation of vector $\boldsymbol{e}$ as input. Weights are typically shared across all classifiers in $\mathcal{W}_\mathcal{M}$ for efficiency, i.e. a nested classifier is trained. Optionally, $\mathcal{W}_\mathcal{M}$ can be dropped so that only $\boldsymbol{\theta}$ is optimized.

Subsequent works have proposed variants of MRL, e.g. to inject MRL properties to already trained models (Yoon et al., 2024), or to reduce the number of layers in a model (Wang et al., 2025b). More recently, Zhang et al. (2025a) compared MRL other forms of size reduction. However, no such studies have used random truncation as a baseline.

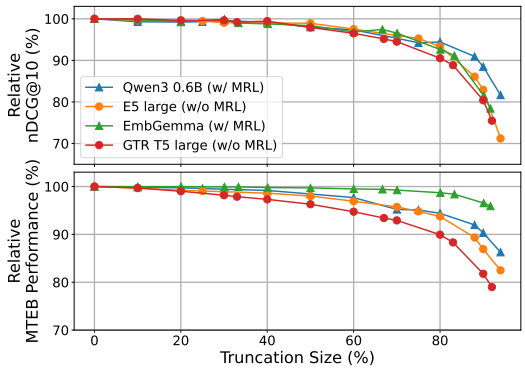
Random truncation. Inspired by the fact that the proof in the seminal Johnson-Lindenstrauss lemma (Johnson and Lindenstrauss, 1984) is based on random projections, many works have since based their embedding-based methods on random projections (Achlioptas, 2003). More recently, Takeshita et al. (2025) found that randomly truncating text embeddings in various ways results in minimal performance loss unless vectors are heavily reduced in size. Similarly, Tsukagoshi and Sasano (2025) also reported the same efficacy of using random truncation as a means of dimensionality reduction.

Closest to our work, Inkiriwang et al. (2025) compared the efficacy of several methods for dimensionality reduction, including random projections, which they found to be the most consistently effective. However, while they included more dimensionality reduction methods, we include more recent models and a larger variety of downstream tasks. More importantly, they do not include MRL in their comparison, despite it being used by some of the best performing text encoders to date (Sturua et al., 2024; Zhang et al., 2025c; Vera et al., 2025).

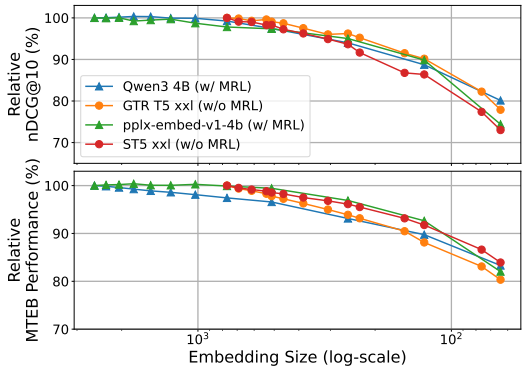
3 MRL vs non-MRL

3.1 Benchmarking Open Encoders

In this section, we compare publicly available text encoders trained with and without MRL. Note that while there are many more differences between these models (e.g. architecture, training recipe), our goal is to study the impact that MRL has on truncation robustness in seminal and state-of-the-art models. It is common to compare text encoders across different architectures, e.g. Wang



(a) Smaller models (∼400M parameters)



(b) Larger models (∼4B parameters)

Figure 2: Performance on NanoBEIR (top) and MTEB (bottom) of text embeddings truncated at various sizes, relative to the performance of the corresponding full-size embeddings.

et al. (2024a); Tsukagoshi and Sasano (2025); Vera et al. (2025), as other aspects typically differentiate new models from prior work, e.g. training recipe (Neelakantan et al., 2022; Sturua et al., 2024). This makes a proper comparison prohibitely expensive. However, we do conduct a more controlled experiment in Sec. 3.2 using a common training recipe. Following MRL, we use prefix truncation as our truncation strategy on all models, that is, we remove the last elements of the embeddings. This means our form of truncation is indeed deterministic. We opt for prefix truncation among other possible strategies because an existing study by Takeshita et al. (2025) show that different truncation strategies do not lead to noticeable performance differences for non-MRL models.

Models. We used eight seminal/state-of-the-art open-weight models, four smaller ones (∼500M params.) and four larger ones (∼4B params.). For non-MRL models, we used E5-Large (Wang et al., 2024a), GTR-T5-Large and -XXL (Ni et al., 2022b), and ST5-XXL (Ni et al., 2022a). For MRL, we used Qwen3-Embedding-0.6B and -4B (Zhang et al., 2025c), EmbeddingGemma-300M (Vera et al., 2025), and pplx-embed-v1-4b (Eslami et al., 2026). For model details, see Appendix A.

Evaluation. For retrieval, we use 13 datasets from NanoBEIR, a smaller variant of BEIR (Thakur et al., 2021), commonly used for research (Conti et al., 2025; Vujanic and Rückstieß, 2026). For classification, we use 11 datasets from MTEB (Muennighoff et al., 2023). We list all datasets in Appendix A and report average performance across tasks (see per-dataset results in Appendix F). We report both relative performance of truncated embeddings compared to the original full-sized embeddings, and absolute performance (in Appendix C). We evaluate at embedding sizes that match common MRL training choices: 64, 128, 256, 512, and 768. We also evaluate at relative positions of 10% to 90% (step 10%) of the original embedding size, which result in sizes outside pre-selected MRL sizes.

Results and discussion. Looking at Fig. 2, if MRL were to bring an advantage, we should see that performance of MRL models drops more slowly than non-MRL models. However, we see no clear influence from MRL, as non-MRL models show similar levels of robustness to truncation, especially in larger models where we see no difference between MRL and non-MRL models at all truncation levels (see Fig. 2b). This is true for both the pre-selected MRL sizes and those outside this choice. The exception is EmbeddingGemma on MTEB, especially in contrast to GTR-T5 which drops faster than other models, but this exception does not hold for NanoBEIR (see Fig. 2a). And performance in absolute terms does not seem to provide an explanation (see Fig. 6 in Appendix C), as EmbeddingGemma consistently outperforms all models, but is more robust to truncation only in MTEB and not NanoBEIR, where Qwen is more robust beyond 80% truncation, despite underperforming in absolute terms. Together, these results suggest that MRL does not provide higher robustness to truncation, except at very high truncation levels and mostly on smaller models. However, while it is important to evaluate the impact of MRL on state-of-the-art models, it is difficult to reason about these results given that any of the various differences between models, and not just MRL, may account for the observed results. So, we report results from a more controlled experiment in the next section.

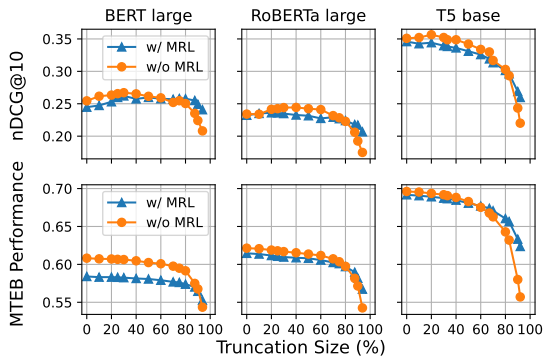


Figure 3: Performance on BEIR and MTEB benchmarks of three pairs of encoders trained with and without MRL. See all models in Fig. 8 in Appendix D.

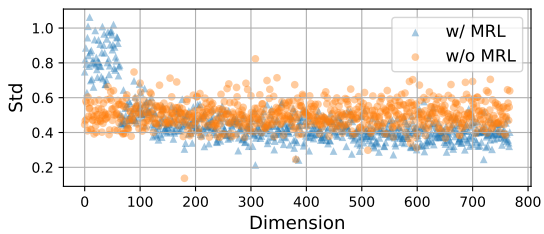


Figure 4: Standard deviation across embedding dimensions from BERT base for different text inputs. MRL increases the variance of the lower dimensions.

3.2 Benchmarking Trained Encoders

To better determine the impact of MRL on embedding truncation, in this section we train pairs of encoders where the only difference is the use of MRL during training. Unless otherwise noted, experimental settings are the same as in Sec. 3.1.

Models. We take five pre-trained language models (PLMs) from different sizes and architectures as starting points for contrastive training. We train two encoder-only models: BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2020), each in two sizes (base and large). We also train the encoder of T5 (Raffel et al., 2020), an encoder-decoder model.

Contrastive training. We train all PLMs with multiple negative ranking loss (Henderson et al., 2017) both for non-MRL and MRL models. For training and validation data, we use the concatenation of SNLI (Bowman et al., 2015) and MultiNLI (Williams et al., 2018). These choices of loss function and training data are common in prior work (Vulić et al., 2023; Ni et al., 2022a; Gao et al., 2021). For MRL models, we uniformly weigh the losses for each of the following nested dimensions: 64, 128, 256, 512 and 768. We train models until convergence (see Fig. 5 in Appendix B) and select the best models on held-out validation samples.

Results and discussion. As before, we evaluate models on MTEB and BEIR.[2] Fig. 3 shows how average downstream performance changes with different levels of truncation for MRL and non-MRL models (see Appendix G for results per dataset). Surprisingly, truncated embeddings from non-MRL models outperform their MRL counterparts almost every time across all models and datasets up until about when 80% truncation is applied, at this crossover point MRL truncation becomes increasingly more beneficial (see Appendix I on how we compute the crossover point). The slight drop in performance in MRL models may be due to the additional constraints in the training objective, but more experiments are needed to understand this. In other words, in a controlled environment, we see that truncated vectors are more effective without MRL than with MRL, unless heavy truncation is applied. As with the results in the previous section, these results show that prefix truncation is often a more effective method of cost reduction than MRL, as it does not rely on any additional training cost and often outperforms truncation done on MRL models. However, MRL does become beneficial at heavy truncation levels, where there is more potential for cost reductions but also more potential for performance loss. To better understand this, we plotted the standard deviation across embedding features in our trained PLMs using queries from NanoBEIR's MS Marco as input. We found that the variance is much higher in lower dimensions with MRL models (see Fig. 4 for BERT, Fig. 24 in Appendix H for all models). This suggests that MRL indeed induces more information storage in the lower dimensions of text embeddings. However, it is unclear why this is not the case for all dimensions, or whether MRL can be improved to achieve this at all levels of truncation.

4 Conclusion

We studied robustness to truncation of text embeddings obtained with and without Matryoshka Representation Learning (MRL), a training method designed for effective truncation. Our test with 8 open text encoders, 10 newly trained encoders and 24 downstream tasks from classification and retrieval benchmarks, show that MRL provides clear

[2] BEIR for smaller models (BERT base and RoBERTa base) and NanoBEIR for the rest.

benefits at very high truncation levels (higher than 80%), but that at lower levels is not more beneficial than applying truncation on models without MRL (the precise point is model- and task-dependent). This suggests future work could improve MRL to provide robustness at all truncation levels.

Limitations

This work has several limitations. First, while the training dataset used for our contrastive training is a common choice in prior work, there are more recent training recipes with larger datasets to train more powerful encoders. We opted for our current choice due to limited computational resources. Second, we only trained small text encoders, again due to limited computational resources. It may be that repeating our experiments from Section 3.2 with larger models produces different results, and that MRL has a better impact in such models, though results in Sec. 3.1 do not suggest this. Third, a larger variety of settings for MRL training could be tested, e.g. exploring the impact that different choices of MRL's pre-determined truncation sizes have on performance. Finally, while our experiments are based on two common application scenarios of text embeddings, namely classification and retrieval, the real world applications are not limited to these two such as clustering or retrieval augmented generation (RAG). Despite these limitations, we believe we have provided enough empirical evidence to suggest that MRL needs further attention so that we may improve embedding robustness to truncation.

References

Dimitris Achlioptas. 2003. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *J. Comput. Syst. Sci.*, 66(4):671–687.

Mohammad Kalim Akram, Saba Sturua, Nastia Havriushenko, Quentin Herreros, Michael Günther, Maximilian Werk, and Han Xiao. 2026. jina-embeddings-v5-text: Task-targeted embedding distillation. *Preprint*, arXiv:2602.15547.

Alexander Bondarenko, Maik Fröbe, Meriem Beloucif, Lukas Gienapp, Yamen Ajjour, Alexander Panchenko, Chris Biemann, Benno Stein, Henning Wachsmuth, Martin Potthast, and Matthias Hagen. 2020. Overview of touché 2020: Argument retrieval: Extended abstract. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 11th International Conference of the CLEF Association, CLEF 2020, Thessaloniki, Greece, September 22–25, 2020, Proceedings*, page 384–395, Berlin, Heidelberg. Springer-Verlag.

Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler. 2016. A full-text learning to rank dataset for medical information retrieval. In *Advances in Information Retrieval*, pages 716–722, Cham. Springer International Publishing.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld. 2020. SPECTER: Document-level representation learning using citation-informed transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2270–2282, Online. Association for Computational Linguistics.

Max Conti, Manuel Faysse, Gautier Viaud, Antoine Bosselut, Celine Hudelot, and Pierre Colombo. 2025. Context is gold to find the gold passage: Evaluating and training contextual document embeddings. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 22594–22608, Suzhou, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Sedigheh Eslami, Maksim Gaiduk, Markus Krimmel, Louis Milliken, Bo Wang, and Denis Bykov. 2026. Diffusion-pretrained dense and contextual embeddings. *Preprint*, arXiv:2602.11151.

Jack FitzGerald, Christopher Hench, Charith Peris, Scott Mackie, Kay Rottmann, Ana Sanchez, Aaron Nash, Liam Urbach, Vishesh Kakarala, Richa Singh, Swetha Ranganath, Laurie Crist, Misha Britan, Wouter Leeuwis, Gokhan Tur, and Prem Natarajan. 2023. MASSIVE: A 1M-example multilingual natural language understanding dataset with 51 typologically-diverse languages. In *Proceedings of the 61st Annual Meeting of the Association for*

Computational Linguistics (Volume 1: Long Papers), pages 4277–4302, Toronto, Canada. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Faegheh Hasibi, Fedor Nikolaev, Chenyan Xiong, Krisztian Balog, Svein Erik Bratsberg, Alexander Kotov, and Jamie Callan. 2017. Dbpedia-entity v2: A test collection for entity search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, page 1265–1268, New York, NY, USA. Association for Computing Machinery.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun hsuan Sung, Laszlo Lukacs, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *Preprint*, arXiv:1705.00652.

Jui-Ting Huang, Ashish Sharma, Shuying Sun, Li Xia, David Zhang, Philip Pronin, Janani Padmanabhan, Giuseppe Ottaviano, and Linjun Yang. 2020. Embedding-based retrieval in facebook search. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 2553–2561, New York, NY, USA. Association for Computing Machinery.

Nathan Inkiriwang, Necva Bölücü, Garth Tarr, and Maciej Rybinski. 2025. Do we really need all those dimensions? an intrinsic evaluation framework for compressed embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 13305–13323, Suzhou, China. Association for Computational Linguistics.

William B. Johnson and Joram Lindenstrauss. 1984. Extensions of lipschitz mappings into hilbert space. *Contemporary mathematics*, 26:189–206.

Phillip Keung, Yichao Lu, György Szarvas, and Noah A. Smith. 2020. The multilingual Amazon reviews corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4563–4568, Online. Association for Computational Linguistics.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, and Ali Farhadi. 2022. Matryoshka representation learning. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Markus Leippold and Thomas Diggelmann. 2020. Climate-fever: A dataset for verification of real-world climate claims. In *NeurIPS 2020 Workshop on Tackling Climate Change with Machine Learning*.

Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta, and Yashar Mehdad. 2021. MTOP: A comprehensive multilingual task-oriented semantic parsing benchmark. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2950–2962, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Ro{bert}a: A robustly optimized {bert} pretraining approach.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Maggie, Phil Culliton, and Wei Chen. 2020. Tweet sentiment extraction. `https://kaggle.com/competitions/tweet-sentiment-extraction`. Kaggle.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. 2018. Www'18 open challenge: Financial opinion mining and question answering. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, page 1941–1942, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, page 165–172, New York, NY, USA. Association for Computing Machinery.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia. Association for Computational Linguistics.

Arvind Neelakantan, Tao Xu, Raul Puri, Alec Radford, Jesse Michael Han, Jerry Tworek, Qiming Yuan, Nikolas Tezak, Jong Wook Kim, Chris Hallacy, Johannes Heidecke, Pranav Shyam, Boris Power, Tyna Eloundou Nekoul, Girish Sastry, Gretchen Krueger, David Schnurr, Felipe Petroski Such, Kenny Hsu, and 6 others. 2022. Text and code embeddings by contrastive pre-training. *Preprint*, arXiv:2201.10005.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. 2022a. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874, Dublin, Ireland. Association for Computational Linguistics.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith Hall, Ming-Wei Chang, and Yinfei Yang. 2022b. Large dual encoders are generalizable retrievers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9844–9855, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

James O'Neill, Polina Rozenshtein, Ryuichi Kiryo, Motoko Kubota, and Danushka Bollegala. 2021. I wish I would have loved this one, but I didn't – a multilingual dataset for counterfactual detection in product review. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7092–7108, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1).

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, Brussels, Belgium. Association for Computational Linguistics.

Vaishali Senthil, Ashutosh Hathidara, and Sebastian Schreiber. 2026. Cohyde: Iterative co-training of llm rewriter & dense encoder for tool retrieval. *Preprint*, arXiv:2605.29271.

Saba Sturua, Isabelle Mohr, Mohammad Kalim Akram, Michael Günther, Bo Wang, Markus Krimmel, Feng Wang, Georgios Mastrapas, Andreas Koukounas, Nan Wang, and Han Xiao. 2024. jina-embeddings-v3: Multilingual embeddings with task lora. *Preprint*, arXiv:2409.10173.

Sotaro Takeshita, Yurina Takeshita, Daniel Ruffinelli, and Simone Paolo Ponzetto. 2025. Randomly removing 50% of dimensions in text embeddings has minimal impact on retrieval and classification tasks. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 27705–27726, Suzhou, China. Association for Computational Linguistics.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Hayato Tsukagoshi and Ryohei Sasano. 2025. Redundancy, isotropy, and intrinsic dimensionality of prompt-based text embeddings. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 25915–25930, Vienna, Austria. Association for Computational Linguistics.

Henrique Schechter Vera, Sahil Dua, Biao Zhang, Daniel Salz, Ryan Mullins, Sindhu Raghuram Panyam, Sara Smoot, Iftekhar Naim, Joe Zou, Feiyang Chen, Daniel Cer, Alice Lisak, Min Choi, Lucas Gonzalez, Omar Sanseviero, Glenn Cameron, Ian Ballantyne, Kat Black, Kaifeng Chen, and 70 others. 2025. Embeddinggemma: Powerful and lightweight text representations. *Preprint*, arXiv:2509.20354.

Ellen Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R. Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. 2021. Trec-covid: constructing a pandemic information retrieval test collection. *SIGIR Forum*, 54(1).

Robin Vujanic and Thomas Rückstieß. 2026. LEAF: Knowledge distillation of text embedding models with teacher-aligned representations. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 43362–43383, San Diego, California, United States. Association for Computational Linguistics.

Ivan Vulić, Goran Glavaš, Fangyu Liu, Nigel Collier, Edoardo Maria Ponti, and Anna Korhonen. 2023. Probing cross-lingual lexical knowledge from multilingual sentence encoders. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2089–2105, Dubrovnik, Croatia. Association for Computational Linguistics.

Henning Wachsmuth, Shahbaz Syed, and Benno Stein. 2018. Retrieval of the best counterargument without prior topic knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 241–251, Melbourne, Australia. Association for Computational Linguistics.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550, Online. Association for Computational Linguistics.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024a. Improving text embeddings with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11897–11916, Bangkok, Thailand. Association for Computational Linguistics.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024b. Multilingual e5 text embeddings: A technical report. *Preprint*, arXiv:2402.05672.

Maolin Wang, Xinjian Zhao, Wanyu Wang, Sheng Zhang, Jiansheng Li, Bowen Yu, Binhao Wang, Shucheng Zhou, Dawei Yin, Qing Li, Ruocheng Guo, and Xiangyu Zhao. 2025a. Embedding in recommender systems: A survey. *Preprint*, arXiv:2310.18608.

Shuai Wang, Shengyao Zhuang, Bevan Koopman, and Guido Zuccon. 2025b. 2d matryoshka training for information retrieval. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '25, page 3125–3134, New York, NY, USA. Association for Computing Machinery.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Jinsung Yoon, Rajarishi Sinha, Sercan O Arik, and Tomas Pfister. 2024. Matryoshka-adaptor: Unsupervised and supervised tuning for smaller embedding dimensions. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10318–10336, Miami, Florida, USA. Association for Computational Linguistics.

Biao Zhang, Lixin Chen, Tong Liu, and Bo Zheng. 2025a. SMEC:rethinking matryoshka representation learning for retrieval embedding compression. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 26209–26222, Suzhou, China. Association for Computational Linguistics.

Biao Zhang, Fedor Moiseev, Joshua Ainslie, Paul Suganthan, Min Ma, Surya Bhupatiraju, Fede Lebron, Orhan Firat, Armand Joulin, and Zhe Dong. 2025b. Encoder-decoder gemma: Improving the quality-efficiency trade-off via adaptation. *Preprint*, arXiv:2504.06225.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025c. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *Preprint*, arXiv:2506.05176.

Wayne Xin Zhao, Jing Liu, Ruiyang Ren, and Ji-Rong Wen. 2024. Dense text retrieval based on pretrained language models: A survey. *ACM Trans. Inf. Syst.*, 42(4).

A Additional Experimental Details

	Name	Domain	Licence
Retrieval	MS MARCO (Nguyen et al., 2016)	Misc.	MIT
	NFCorpus (Boteva et al., 2016)	Bio-Medical	N/A
	FiQA-2018 (Maia et al., 2018)	Finance	N/A
	ArguAna (Wachsmuth et al., 2018)	Misc.	CC BY 4.0
	Touche-2020 (Bondarenko et al., 2020)	Misc.	CC BY 4.0
	Quora	Quora	N/A
	DBPedia (Hasibi et al., 2017)	Wikipedia	CC BY-SA 3.0
	SCIDOCS (Cohan et al., 2020)	Scientific	GNU General Public License v3.0
	FEVER (Thorne et al., 2018)	Wikipedia	CC BY-SA 3.0 l
	Climate-FEVER (Leippold and Diggelmann, 2020)	Wikipedia	N/A
	SciFact (Wadden et al., 2020)	Scientific	CC BY-NC 2.0
	Natural Questions (Kwiatkowski et al., 2019)	Scientific	CC BY-SA 3.0
	HotpotQA (Yang et al., 2018)	Scientific	CC BY-SA 4.0
	TREC-COVID (Voorhees et al., 2021)	Bio-Medical	Dataset License Agreement
Classification	AmazonCounterfactualClassification (O'Neill et al., 2021)	Reviews, Written	CC-by-4.0
	AmazonPolarityClassification (McAuley and Leskovec, 2013)	Reviews, Written	Apache 2.0
	AmazonReviewsClassification (Keung et al., 2020)	Reviews, Written	N/A
	Banking77Classification (Casanueva et al., 2020)	Written	MIT
	EmotionClassification (Saravia et al., 2018)	Social, Written	N/A
	ImdbClassification (Maas et al., 2011)	Reviews, Written	N/A
	MassiveIntentClassification (FitzGerald et al., 2023)	Spoken	Apache 2.0
	MassiveScenarioClassification (FitzGerald et al., 2023)	Spoken	Apache 2.0
	MTOPDomainClassification (Li et al., 2021)	Spoken	N/A
	MTOPIntentClassification (Li et al., 2021)	Spoken	N/A
	TweetSentimentExtractionClassification (Maggie et al., 2020)	Social, Written	N/A

Table 1: A list of datasets used in our evaluation. TREC-COVID is only available in BEIR, not in NanoBEIR (`https://huggingface.co/collections/zeta-alpha-ai/nanobeir`).

Details about non-MRL models. ST5 xxl (Ni et al., 2022a) is an encoder with 5B parameters derived from T5 (Raffel et al., 2020). While the original T5 is built on an encoder-decoder architecture, the ST5 series takes only the encoder and trains it with a contrastive learning objective. GTR xxl (5B parameters) and GTR T5 large (300M parameters) (Ni et al., 2022b) are also derived from encoders of T5. The training recipe focuses more on retrieval tasks, e.g., the training data includes datasets such as MS MARCO (Nguyen et al., 2016) while ST5 is more generic for various embedding tasks. Both ST5 and GTR are trained without MRL objectives. E5 large (300M parameters) (Wang et al., 2024b) is a non-MRL encoder derived from a BERT model. In addition to the mix of public data, the authors used the text pairs synthetically generated by proprietary language models for their contrastive training.

Details about MRL models. EmbeddingGemma (300M parameters) (Vera et al., 2025) is an encoder model derived from Gemma, a decoder-only model. To achieve the architectural conversion, the authors first turned the decoder-only model into an encoder-decoder architecture through a method proposed by Zhang et al. (2025b). Second, similarly to ST5, they took only the encoder and trained it contrastively to obtain the final encoder model. EmbeddingGemma provides the original embedding with a size of 768, and is trained with MRL to produce 512, 256, and 128-dimensional embeddings. Qwen3 Embedding 4B and Qwen3 Embedding 0.6B (Zhang et al., 2025c) are built on the Qwen3 text generation model family through multi-stage training and model merging. The authors trained the models to support more than 100 languages and cover various domains such as code retrieval and bitext mining. The original embedding dimensions are 2560 and 1024, respectively. Both models are trained with MRL with the smallest dimension of 32. The other supported dimensions are not reported in the corresponding documentation and technical report. pplx-embed-v1-4b (Eslami et al., 2026) is also trained from Qwen3 text generation model however by a different organization. While this variant has also gone through multi-stage training and model merging, the authors apply a diffusion training method (Gong et al., 2025) to enable bidirectional text encoding. This model is trained with MRL.

B Validation Loss Curve During Contrastive Trainings

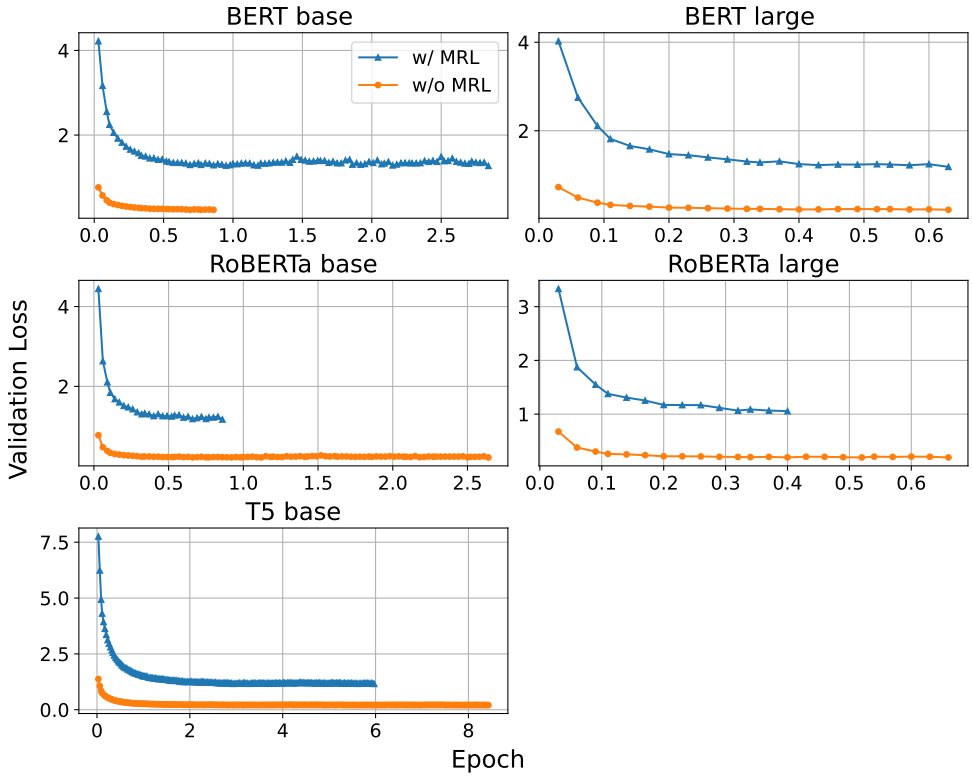


Figure 5: Validation loss curve for contrastive learning with and without MRL for all model pairs. Our training recipe allowed all models to reach convergence.

C Absolute Performance of Open Models

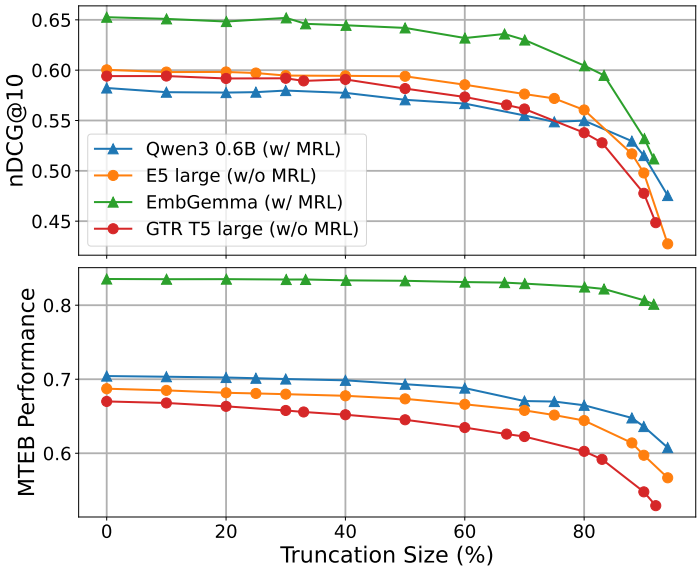


Figure 6: Absolute performance on NanoBEIR (top) and MTEB (bottom) of text embeddings by smaller models (∼500M params.) truncated at various sizes, using four publicly available text encoders with and without MRL. As in Fig. 2a, the slopes indicate no significant difference in robustness to truncation in MRL and non-MRL models. However, we do see that EmbeddingGemma scores significantly higher than all other models in both NanoBEIR and MTEB. Such higher performance may explain EmbeddingGemma's higher robustness to truncation in MTEB, as it may be easier to retain high performance after truncation when performance was high in the first place. But this too does not generally hold, as Qwen is more robust to truncation than EmbeddingGemma in NanoBEIR (see Fig. 2a), despite underperforming in absolute terms.

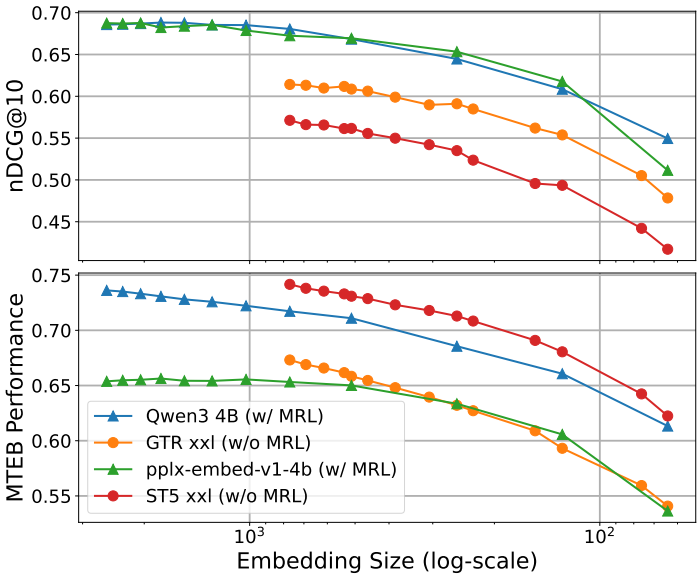


Figure 7: Absolute performance on NanoBEIR (top) and MTEB (bottom) of text embeddings by larger models (∼4B params.) truncated at various sizes, using four publicly available text encoders with and without MRL. Similar to smaller models (see Figs. 2a and 6), the slopes indicate that MRL models are not more robust to truncation compared to non-MRL models.

D Absolute Performance of Trained Models

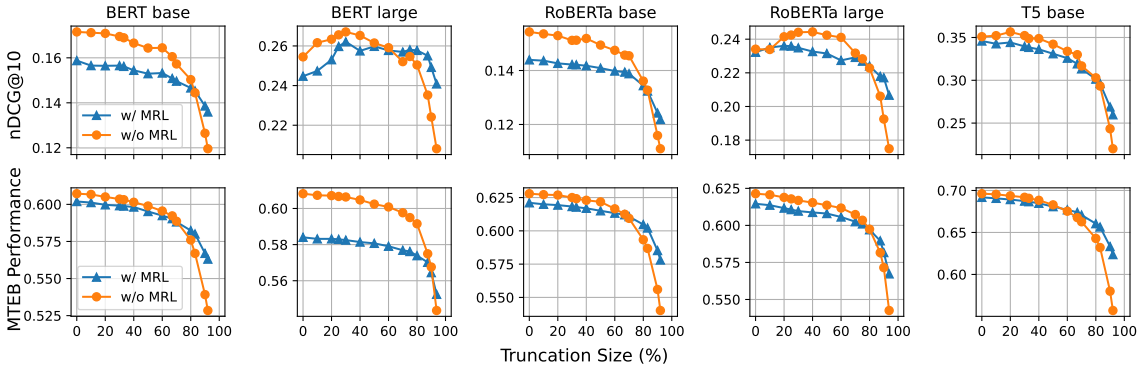


Figure 8: Performance on BEIR and MTEB benchmarks of five pairs of encoders trained with and without MRL. Non-MRL models outperform MRL models up until 80% truncation, where MRL models become more beneficial.

E Relative Performance of Trained Models

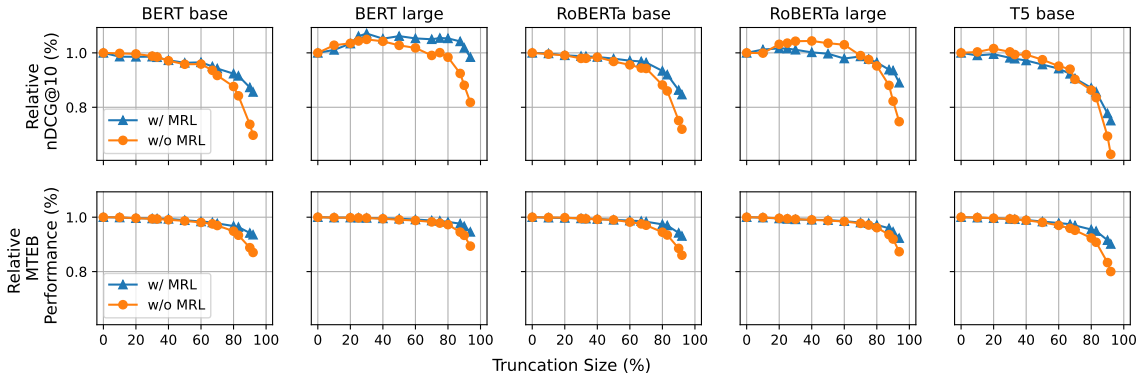


Figure 9: Relative performance on BEIR and MTEB benchmarks of five pairs of encoders trained with and without MRL.

F Performance of Open Models on Individual Datasets

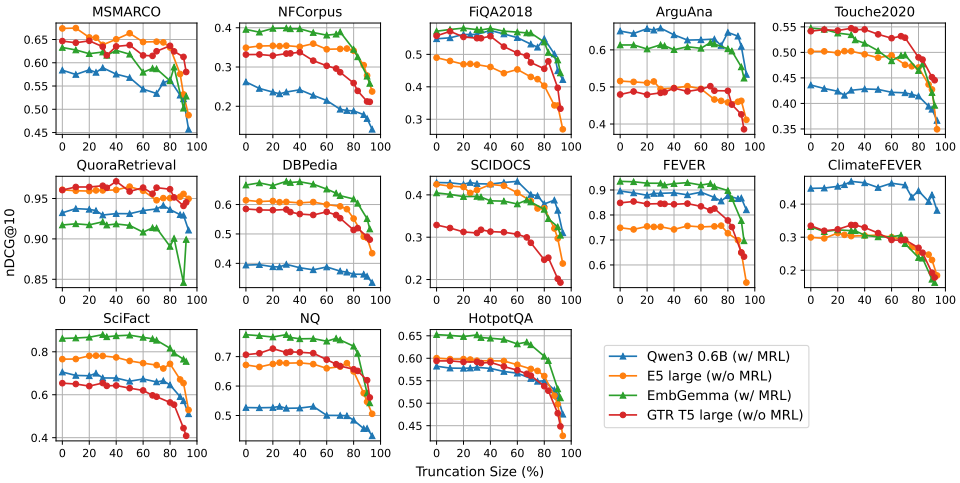


Figure 10: Performance of smaller open text encoders in NanoBEIR datasets.

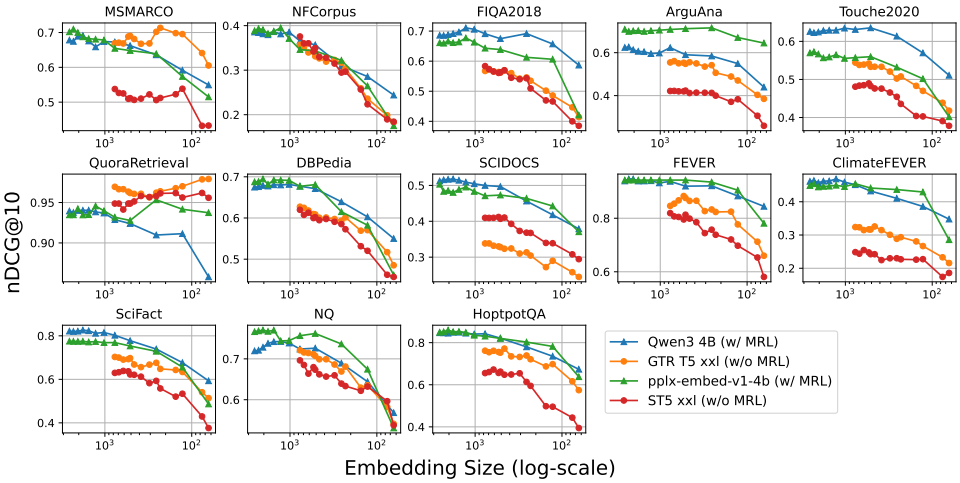


Figure 11: Performance of larger open text encoders in NanoBEIR datasets.

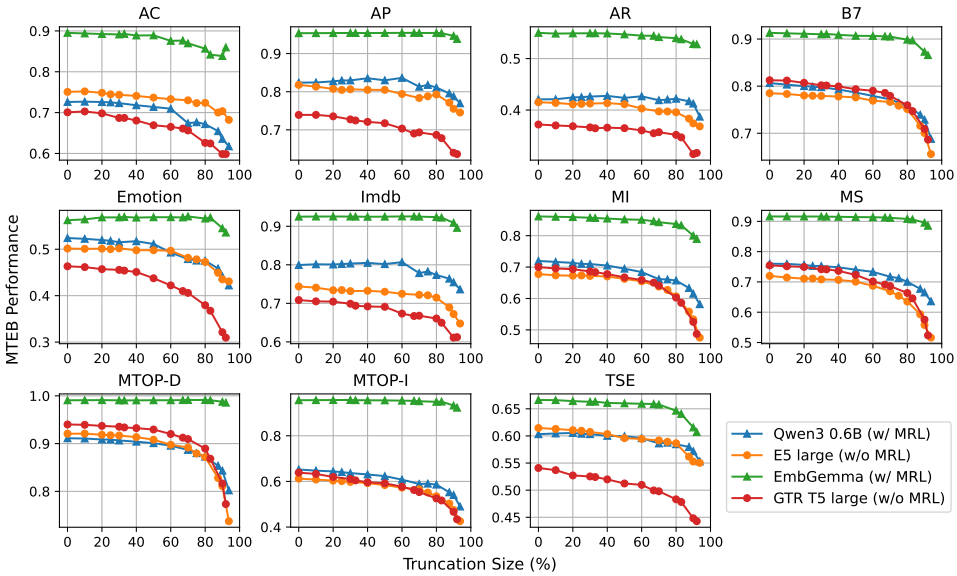


Figure 12: Performance of smaller open text encoders in MTEB datasets.

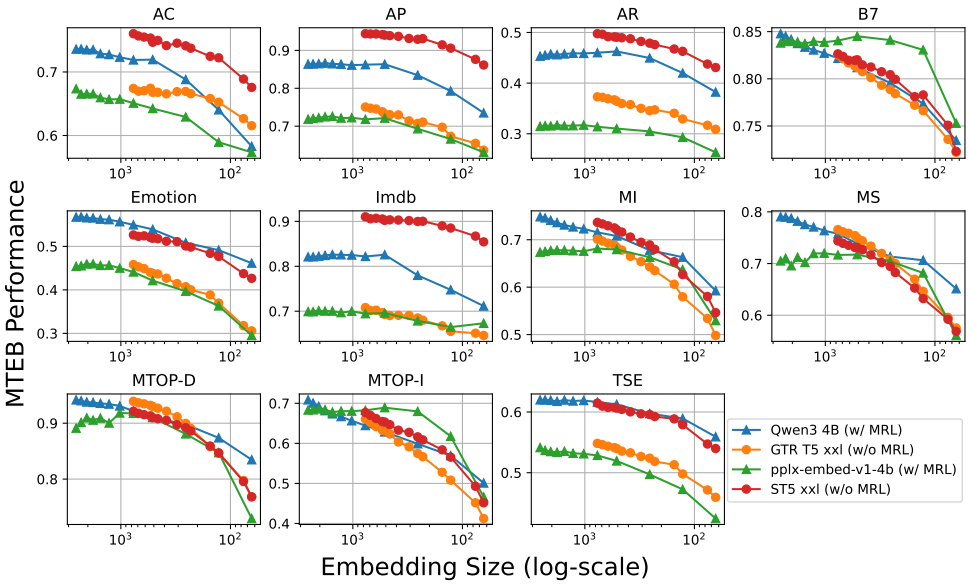


Figure 13: Performance of larger open text encoders in MTEB datasets.

G Performance of Trained Models on Individual Datasets

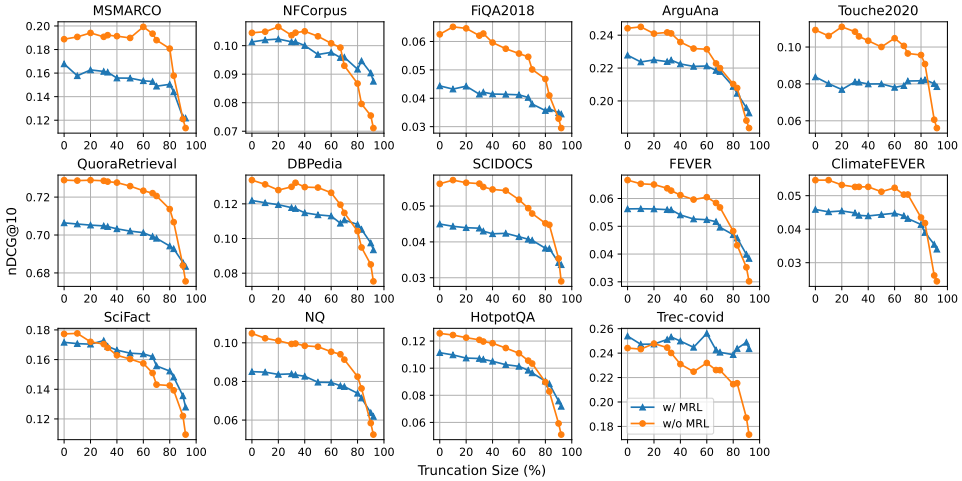


Figure 14: BERT base performance on each of the BEIR datasets.

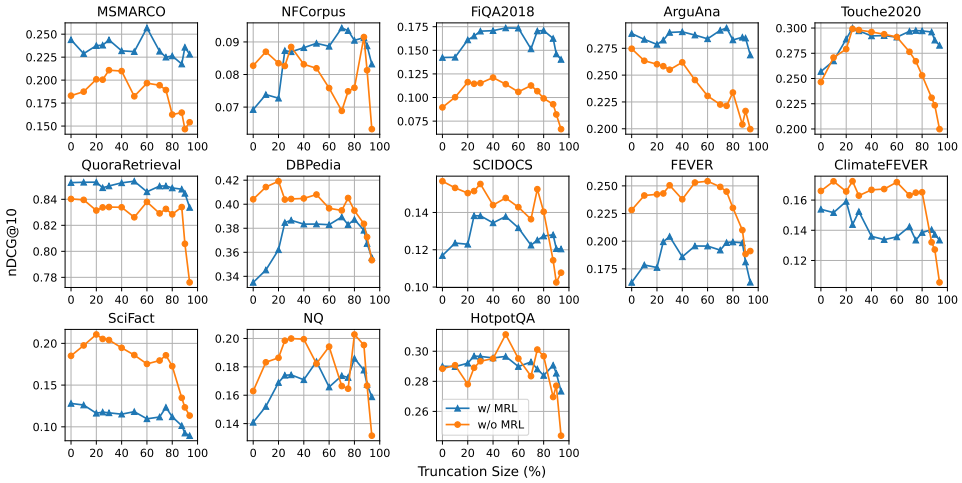


Figure 15: BERT large performance on each of the NanoBEIR datasets.

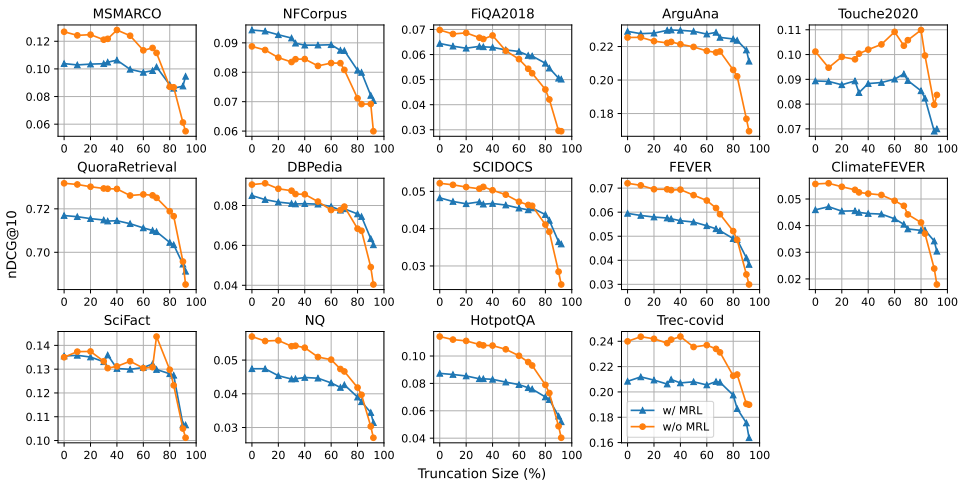


Figure 16: RoBERTa base performance on each of the BEIR datasets.

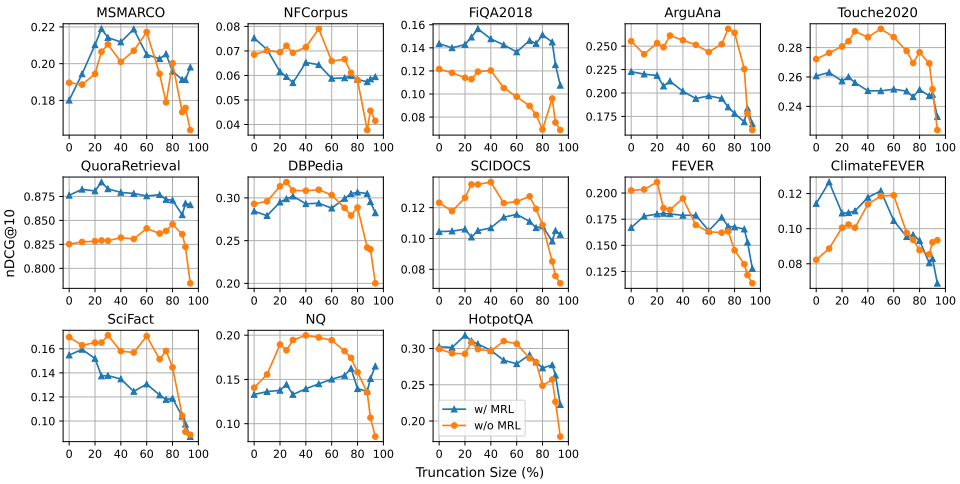


Figure 17: RoBERTa large performance on each of the NanoBEIR datasets.

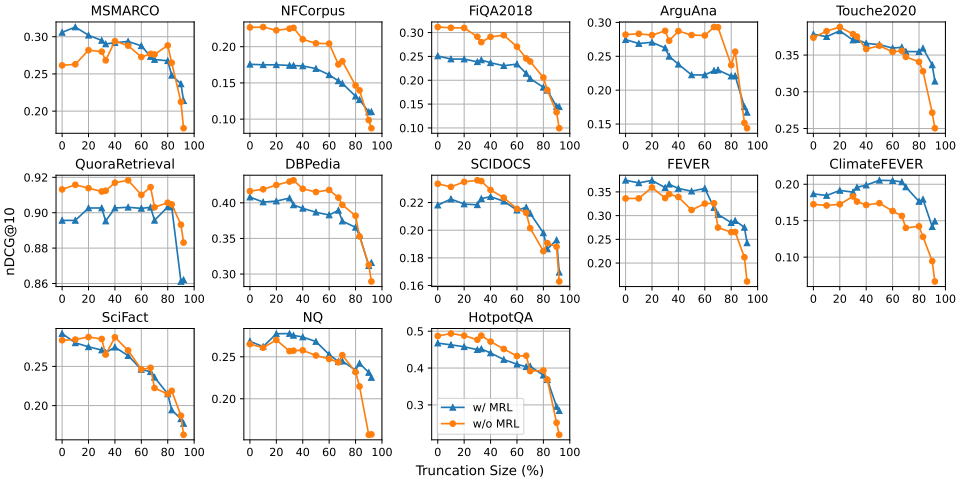


Figure 18: T5 base performance on each of the NanoBEIR datasets.

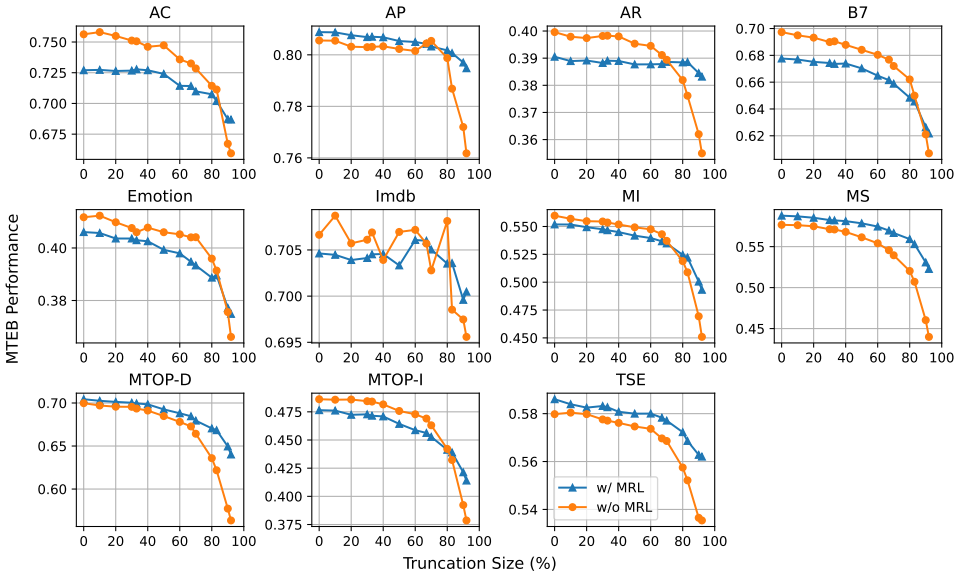


Figure 19: BERT base performance on each of the MTEB datasets.

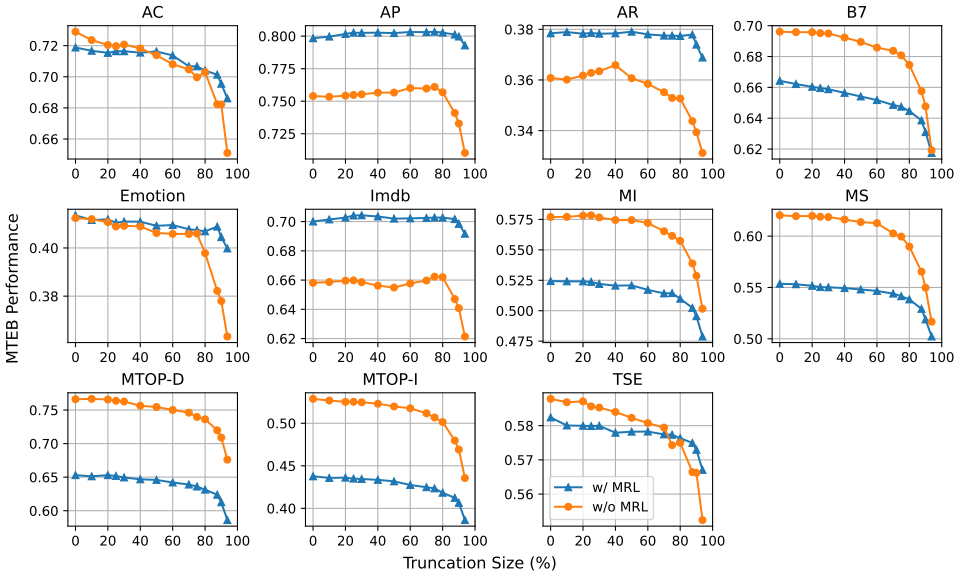


Figure 20: BERT large performance on each of the MTEB datasets.

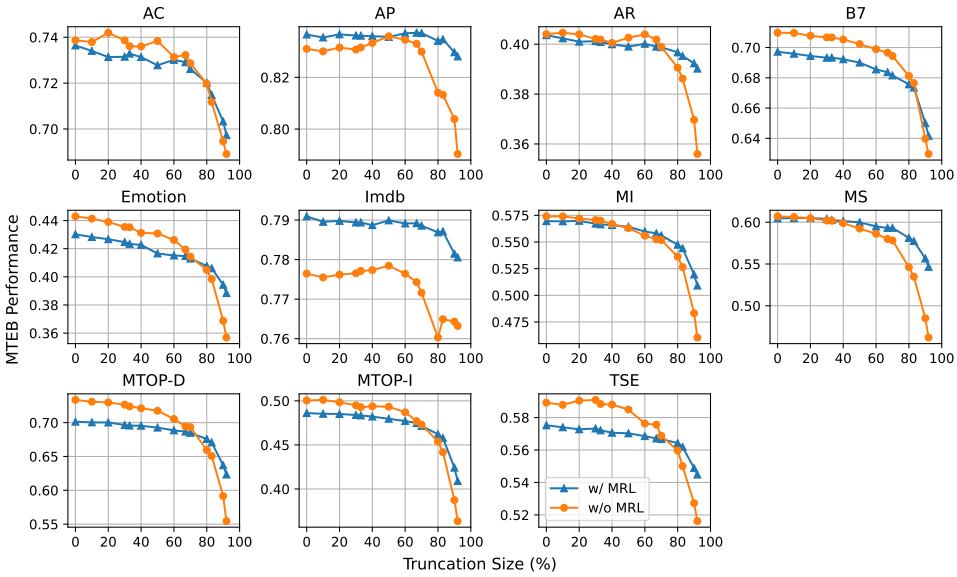


Figure 21: RoBERTa base performance on each of the MTEB datasets.

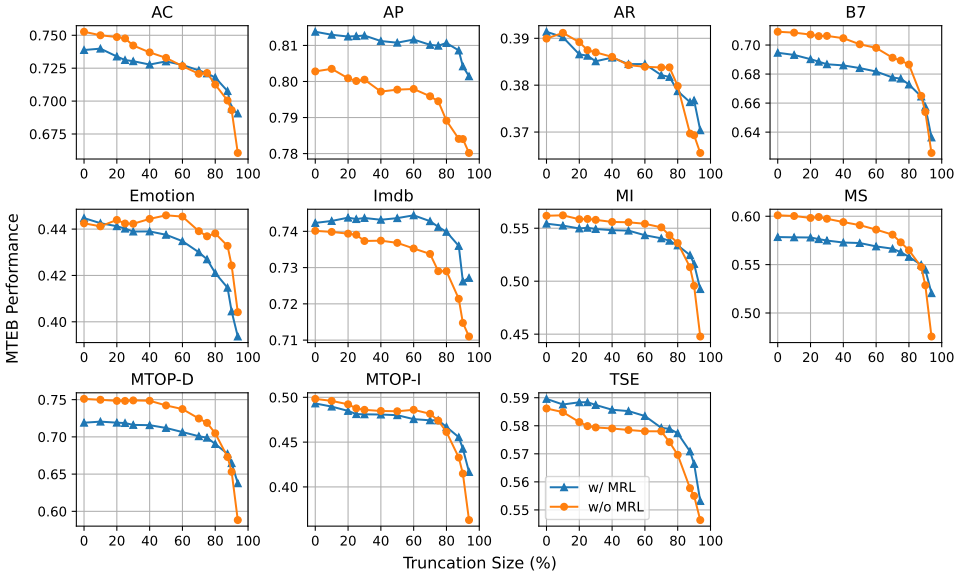


Figure 22: RoBERTa large performance on each of the MTEB datasets.

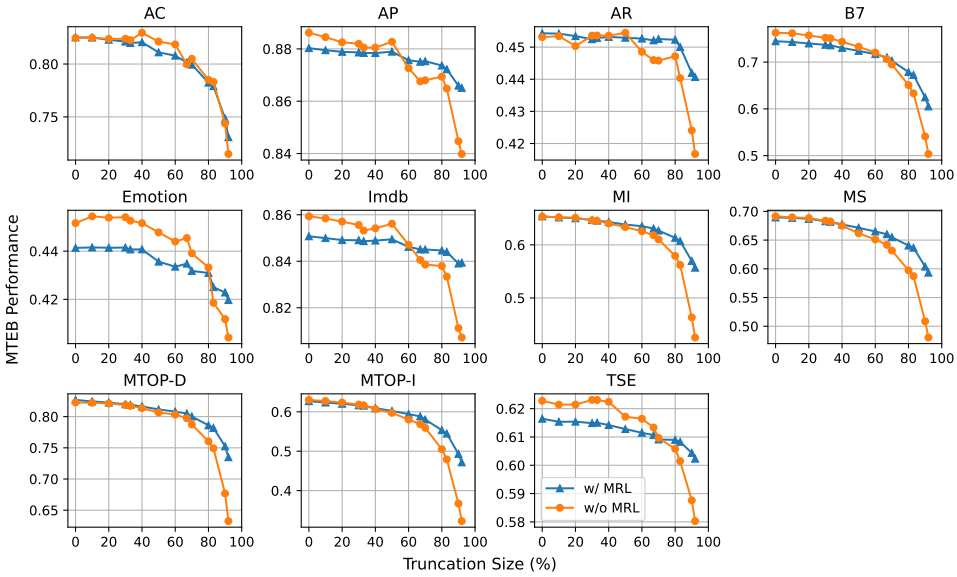


Figure 23: T5 base performance on each of the MTEB datasets.

H Standard Deviations of Each Dimension

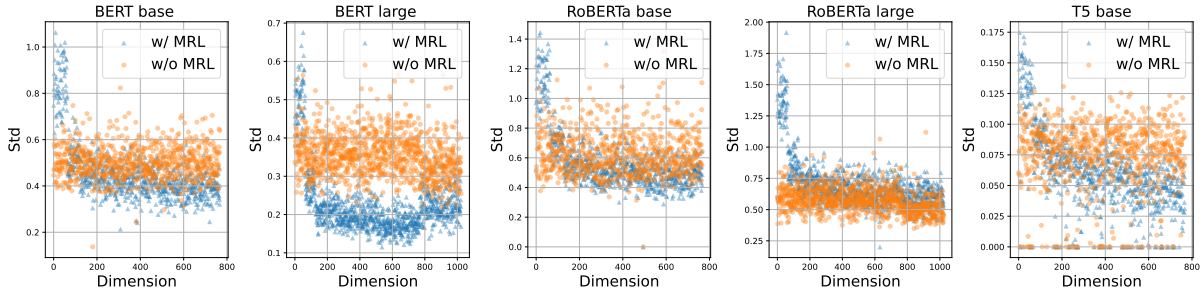


Figure 24: Standard deviations of values taken by each dimension when encoding different texts. We observe that MRL models seem to store more information in the lower dimensions compared to non-MRL models, which may be related to its better performance at heavy truncation levels.

I Computation of crossover point

We use linear interpolation to estimate the crossover point between each MRL and non-MRL model pair based on the results in Sec. 3.2 (summarized in Fig. 8). Through this analysis, we observe that the crossovers occur on average at approximately 75% truncation across the two benchmarks and five model pairs. Since all experiments use a truncation step size of 10%, we conservatively summarize this finding by stating that non-MRL models remain comparable to MRL models until roughly 80% of the embedding dimensions are truncated.

J Confidence Intervals

For precision, and following Senthil et al. (2026), we report in Tables 2 and 3, for each truncation level, the per-query 95% confidence interval (CI) for NDCG@10, computed via percentile bootstrap using 10000 samples with replacement over a fixed seed. In addition, to determine the statistical significance of the observed performance difference at each truncation level, we tested whether the 95% CIs of the per-query difference excludes zero, the null hypothesis of no difference ($p < 0.05$). Each underlined entry in Tables 2 and 3 shows when the difference is statistically significant. For RoBERTa and T5, the difference is only statistically significant at heavy truncation, such as 64 remaining dimensions for RoBERTa-base (8% of the original embedding size). The exception is BERT, where no MRL seems more favourable most of the time.

		64	**76**	**128**	**153**	**256**	**307**	**384**	**460**	**512**	**537**	**614**	**691**	**768**
B-b	**w/**	$0.22^{0.25}_{0.20}$	$0.23^{0.25}_{0.20}$	$0.24^{0.26}_{0.21}$	$0.24^{0.26}_{0.21}$	$0.25^{0.27}_{0.22}$	$0.25^{0.28}_{0.23}$	$0.26^{0.28}_{0.23}$	$0.26^{0.28}_{0.23}$	$0.26^{0.28}_{0.23}$	$0.26^{0.28}_{0.23}$	$0.26^{0.28}_{0.23}$	$0.26^{0.28}_{0.23}$	$0.26^{0.28}_{0.23}$
	w/o	$0.22^{0.25}_{0.20}$	$0.23^{0.25}_{0.20}$	$0.25^{0.28}_{0.23}$	$0.26^{0.28}_{0.23}$	$\underline{0.27}^{0.30}_{0.25}$	$\underline{0.28}^{0.31}_{0.25}$	$\underline{0.28}^{0.30}_{0.25}$	$\underline{0.28}^{0.31}_{0.25}$	$\underline{0.28}^{0.31}_{0.25}$	$\underline{0.28}^{0.31}_{0.25}$	$\underline{0.28}^{0.31}_{0.26}$	$\underline{0.28}^{0.31}_{0.26}$	$\underline{0.28}^{0.31}_{0.26}$
R-b	**w/**	$\underline{0.23}^{0.25}_{0.20}$	$0.23^{0.25}_{0.21}$	$0.24^{0.27}_{0.22}$	$0.24^{0.27}_{0.22}$	$0.25^{0.27}_{0.22}$	$0.25^{0.28}_{0.23}$	$0.25^{0.28}_{0.23}$	$0.26^{0.28}_{0.23}$	$0.26^{0.28}_{0.23}$	$0.26^{0.28}_{0.23}$	$0.26^{0.29}_{0.23}$	$0.26^{0.28}_{0.23}$	$0.26^{0.28}_{0.23}$
	w/o	$0.21^{0.23}_{0.18}$	$0.22^{0.24}_{0.19}$	$0.24^{0.26}_{0.21}$	$0.24^{0.27}_{0.22}$	$0.26^{0.29}_{0.24}$	$0.26^{0.29}_{0.24}$	$0.26^{0.29}_{0.24}$	$0.27^{0.29}_{0.24}$	$0.27^{0.29}_{0.24}$	$0.27^{0.30}_{0.24}$	$0.27^{0.30}_{0.24}$	$0.27^{0.30}_{0.25}$	$0.27^{0.30}_{0.25}$
T-b	**w/**	$\underline{0.26}^{0.29}_{0.24}$	$\underline{0.27}^{0.30}_{0.24}$	$0.30^{0.32}_{0.27}$	$0.30^{0.33}_{0.28}$	$0.32^{0.35}_{0.29}$	$0.33^{0.35}_{0.30}$	$0.33^{0.36}_{0.30}$	$0.34^{0.36}_{0.31}$	$0.34^{0.37}_{0.31}$	$0.34^{0.37}_{0.31}$	$0.34^{0.37}_{0.32}$	$0.34^{0.37}_{0.32}$	$0.35^{0.37}_{0.32}$
	w/o	$0.22^{0.25}_{0.20}$	$0.24^{0.27}_{0.22}$	$0.29^{0.32}_{0.27}$	$0.30^{0.33}_{0.28}$	$0.33^{0.36}_{0.30}$	$0.33^{0.36}_{0.31}$	$0.34^{0.37}_{0.32}$	$0.35^{0.38}_{0.32}$	$0.35^{0.38}_{0.32}$	$0.35^{0.38}_{0.33}$	$0.36^{0.38}_{0.33}$	$0.35^{0.38}_{0.33}$	$0.35^{0.38}_{0.32}$

Table 2: 95% Confidence intervals (CIs) via per-query bootstrap for three base-sized model pairs: BERT-base (B-b), RoBERTa-base (R-b), T5-base (T-b) on NanoBEIR datasets. Each column shows the average performance and CIs for the indicated number of remaining dimensions after truncation. A score is underlined when it significantly outperforms its counterpart by two-sided paired t-test ($p < 0.05$).

		64	**102**	**128**	**204**	**256**	**409**	**512**	**614**	**716**	**819**	**921**	**1024**
B-l	**w/**	$0.22^{0.24}_{0.19}$	$0.22^{0.24}_{0.20}$	$0.23^{0.25}_{0.20}$	$0.23^{0.25}_{0.21}$	$0.23^{0.26}_{0.21}$	$0.24^{0.26}_{0.21}$	$0.24^{0.25}_{0.22}$	$0.24^{0.26}_{0.21}$	$0.24^{0.26}_{0.22}$	$0.24^{0.26}_{0.22}$	$0.24^{0.27}_{0.22}$	$0.24^{0.27}_{0.22}$
	w/o	$0.21^{0.23}_{0.19}$	$0.22^{0.25}_{0.20}$	$0.24^{0.26}_{0.21}$	$\underline{0.25}^{0.28}_{0.23}$	$\underline{0.25}^{0.28}_{0.23}$	$\underline{0.26}^{0.29}_{0.24}$	$\underline{0.26}^{0.28}_{0.24}$	$\underline{0.27}^{0.29}_{0.24}$	$\underline{0.27}^{0.29}_{0.24}$	$\underline{0.26}^{0.29}_{0.24}$	$0.26^{0.29}_{0.24}$	$0.25^{0.28}_{0.23}$
R-l	**w/**	$\underline{0.20}^{0.22}_{0.17}$	$\underline{0.21}^{0.23}_{0.18}$	$0.22^{0.24}_{0.19}$	$0.22^{0.24}_{0.20}$	$0.22^{0.25}_{0.20}$	$0.23^{0.25}_{0.20}$	$0.23^{0.24}_{0.21}$	$0.23^{0.25}_{0.20}$	$0.23^{0.25}_{0.21}$	$0.23^{0.25}_{0.20}$	$0.23^{0.25}_{0.20}$	$0.23^{0.25}_{0.21}$
	w/o	$0.17^{0.20}_{0.15}$	$0.19^{0.22}_{0.17}$	$0.21^{0.23}_{0.18}$	$0.22^{0.25}_{0.20}$	$0.23^{0.25}_{0.21}$	$0.24^{0.27}_{0.22}$	$0.24^{0.26}_{0.23}$	$0.24^{0.27}_{0.22}$	$0.24^{0.27}_{0.22}$	$0.24^{0.27}_{0.22}$	$0.23^{0.26}_{0.21}$	$0.23^{0.26}_{0.21}$

Table 3: 95% Confidence intervals (CIs) via per-query bootstrap for two base-sized model pairs: BERT-large (B-l), RoBERTa-large (R-l) on NanoBEIR datasets. Each column shows the average performance and CIs for the indicated number of remaining dimensions after truncation. A score is underlined when it significantly outperforms its counterpart by two-sided paired t-test ($p < 0.05$).