

Preserving Multilingual Quality While Tuning Query Encoder on English Only

Oleg Vasilyev, Randy Sawaya, John Bohannon

Primer Technologies Inc.

San Francisco, California

oleg,randy.sawaya,john@primer.ai

Abstract

A query encoder of a dual passage retrieval system can be tuned for specific types of queries or domains, while the precomputed and stored documents representations are kept intact. Switching from one query encoder to another when needed is easily feasible, unlike overhauling the embeddings of a whole knowledge base. In this work we raise a question: Can the generic, original qualities of the encoder be preserved or at least left not too degraded when it is tuned on a narrow domain? We conducted experiments on a high quality multilingual embedding model: Tuning it on a single English-only dataset, we observe that the tuning not only preserves the multilingual qualities, but even improves them. The embedding qualities on distinctly different data are also improved or at least preserved. Drawing on our observations, we suggest a more general hypothesis: Tuning with intentionally low learning rate can preserve or improve a system's properties acquired in training, but not specifically targeted by tuning. We call this *adiabatic tuning* and provide tentative explanations.

1 Introduction

Advances in neural NLP methods have resulted in high quality dense vector text representations (Reimers and Gurevych, 2019; Cer et al., 2018; Conneau et al., 2017). Such representations are often used at the initial stages of an information retrieval system, selecting the most relevant documents, ranked relative to the query (Xiong et al., 2020; Zhan et al., 2020, 2021; Ren et al., 2021b). A dual encoder is successfully used to train the representations (Karpukhin et al., 2020; Ren et al., 2021a; Qu et al., 2021; Hofstätter et al., 2021; Ni et al., 2022; Dong et al., 2022). A dual encoder dense passage retrieval system is efficient for two main reasons: (1) it allows using the simple inner product of query and document representations,

and (2) it allows modifying the query representation for a task or domain, while keeping the stored and precomputed (query-invariant) document representations intact.

If the representation was pretrained in a multilingual setting, tuning on English-only samples may be expected to degrade the multilingual qualities and there may not be enough cross-lingual samples for tuning on a specific domain or types of queries. A multilingual query generator may be employed to overcome a shortage of cross-lingual data (Ren et al., 2022; Zhuang et al., 2023), but, in this work, we follow an arguably simpler strategy. In order to understand the effect of English-only tuning on multilingual qualities of a representation, and to assess a possible degradation, we consider a simple setup: A state of the art multilingual embedding model is taken as the starting point, and fine-tuned by English only samples as the query representation part of a dual encoder.

We assume that our observations of the degradation or preservation of the multilingual qualities may be generalized to other pretrained system qualities that are not directly targeted in tuning. In order to obtain preliminary confirmation of this hypothesis, we also observe the effect of tuning on the embedding quality for queries and text chunks of very different styles, the likes of which could be present in the training of the original encoder, but certainly not targeted in tuning.

Our contribution:

1. We show that fine-tuning a query encoder on an English-only dataset may not only preserve multilingual qualities of query-document embeddings matching, but even improve them.
2. We hypothesize that a tuning regime with intentionally low learning rate (far below of what is necessary to avoid overfitting) preserves or improves the properties acquired in the training, but not targeted by tuning. We call this *adiabatic tuning* and suggest support-

ing observations and conjectural explanations.

3. We add a dataset with graded difficulty, based on ARXIV titles and abstracts.

Although high-resource languages can be used for cross-lingual transfer (Lin et al., 2019), our setting does not have such a goal: the tuning is set to improve the query part of a dual encoder on a certain dataset, with no driving mechanism for preserving or improving the other qualities of the system.

Our starting point is one of the best (for its lean size) multilingual embedding models which differs from starting with a multilingual language model and then aligning the generated embeddings for different languages (Wang et al., 2022).

2 Setup

2.1 Models

In what follows, we use a state-of-the-art multilingual model *intfloat/multilingual-e5-small*¹ (Wang et al., 2024b) which will be referred to here as *E5*. For most of the evaluations, we also consider results using *sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2*² (Reimers and Gurevych, 2019), referred to as *L12*. Finally, we confirm some observations with monolingual *intfloat/e5-small-v2*³ (Wang et al., 2024a), referred to as *E5e*. All these models provide embeddings of a practical small size of 384.

2.2 Datasets

We use MSMARCO (Nguyen et al., 2018) Triplets⁴ for tuning and evaluation. For evaluating the qualities not targeted by tuning, we use the ARXIV dataset with negatives⁵, which we made from arxiv (version 173)^{6,7}, and the test subset of the XNLI multilingual dataset⁸ (Conneau et al., 2018). We also use HOTPOTQA⁹ (Yang et al., 2018) and SQUAD¹⁰ (Rajpurkar et al., 2018, 2016) for con-

firmed some observations (Appendices C, D).

Our test subset of MSMARCO contains 357642 evaluation triplets, made of 7000 samples - all the positives and negatives are used (Appendix A).

Of ARXIV we use titles and abstracts. We made two flavors of evaluation arxiv triplets: (1) *arxiv-title* where a title plays role of the query (anchor), and the corresponding abstract is a positive passage, and (2) *arxiv-first* where the first sentence of abstract is used as the query, and the rest of it is used as a positive (Appendix B). We also use narrow versions of *arxiv-first* in Appendix K.

2.3 Tuning and evaluations

Unless otherwise specified, we freeze the text encoder and proceed to fine-tune only the query encoder (fully or partially unfrozen) by contrastive learning on MSMARCO (or on narrow ARXIV subsets, Appendix K) with a learning rate of 5e-8, batch size of 14 and the triple margin loss with margin 0.1. Other details are in Appendix E. In our experiments we considered different settings of freezing, batch size, learning rate, the margin of triplet loss, the stopping criterion, weight decay, scheduling versions and optimizers.

In most of our evaluations, we compare the similarity (or distance) between the anchor (query) and the positive vs the negative. If the positive does not turn out to be closer than the negative to the anchor, we count this as an error. We thus characterize performance of the encoder on a query by the number of errors divided by the total number of positive-negative pairs. We call this *positive-negative discrepancy* (PND). The measure is easy to interpret, and its range (from 0 to 1) is the same and equally fair for any amounts of positives and negatives, as long as they exist in a selection for a query. On multiple queries we take an averaged PND. We confirm some results also using mean reciprocal rank (MRR), mean average precision (MAP) and precision at top 1 (P@1). The improvement of performance is measured as relative change of a measure M (PND or MRR or other):

$$I = s \frac{\tilde{M} - M}{M} \quad (1)$$

where M is for the original encoder, and \tilde{M} is for the encoder after the tuning. The sign $s = -1$ for PND, because it decreases when improved, and $s = 1$ for the other measures.

For evaluating XNLI we use its pairs of sentences, each sentence is given in 15 languages (Ap-

¹<https://huggingface.co/intfloat/multilingual-e5-small>

²<https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2>

³<https://huggingface.co/intfloat/e5-small-v2>

⁴<https://huggingface.co/datasets/sentence-transformers/embedding-training-data/blob/main/msmarco-triplets.jsonl.gz>

⁵<https://huggingface.co/datasets/primer-ai/arxiv-negatives>

⁶https://huggingface.co/datasets/arxiv-community/arxiv_dataset

⁷<https://www.kaggle.com/datasets/Cornell-University/arxiv>

⁸<https://huggingface.co/datasets/facebook/xnli>

⁹<https://hotpotqa.github.io/>

¹⁰https://huggingface.co/datasets/rajpurkar/squad_v2

pendix F). One sentence is used as a query, another as a passage. All pairs are human-labeled as entailment, neutral or contradiction. Hence, the sentences of an entailment pair should be closer to each other than the sentences of any neutral or contradiction pair. Whenever this does not happen, we count this as an error for PND. In Appendix G we made sure that the amount of errors the original encoder makes on our datasets is large enough to consider how tuning would affect them.

3 Observations

3.1 Tuning partially frozen query model

In Table 1 we show results of tuning the dual encoder, with the text encoder frozen and query model free or partially frozen. Here and throughout the paper we use the easiest version of ARXIV (see Appendix H on performance at other levels). Freezing the embedding block appears to be the best option for preserving the multilingual qualities, and henceforth it is used unless specified otherwise. In Table 2 we confirm the improvement on six other datasets (Appendices A, C, D), and show some other measures.

The multilingual qualities are not only preserved, but even mostly improved, especially on cosine similarity. The PND improvement is shown for each language pair separately in Figure 1. The results for the *L12* model are similar (Appendix J). In Appendix K we also confirm our observations with *E5* tuned on specific categories of ARXIV.

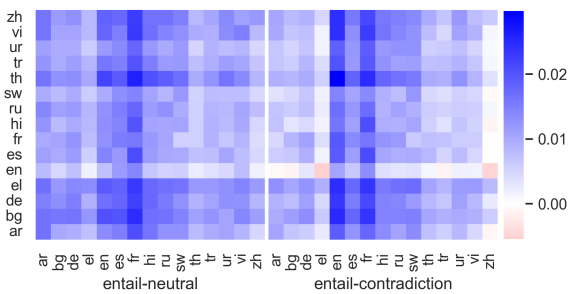


Figure 1: Improvement of *E5* on XNLI assessed by cosine. Query is on axis *Y*; text is on *X*.

3.2 Learning rate and adiabatic tuning

Increasing the tuning learning rate delivers more gains on MSMARCO, while eventually reducing gains on XNLI and even ARXIV. Improvement of PND on MSMARCO and ARXIV is shown in Figure 2(b); the number of language pairs improved

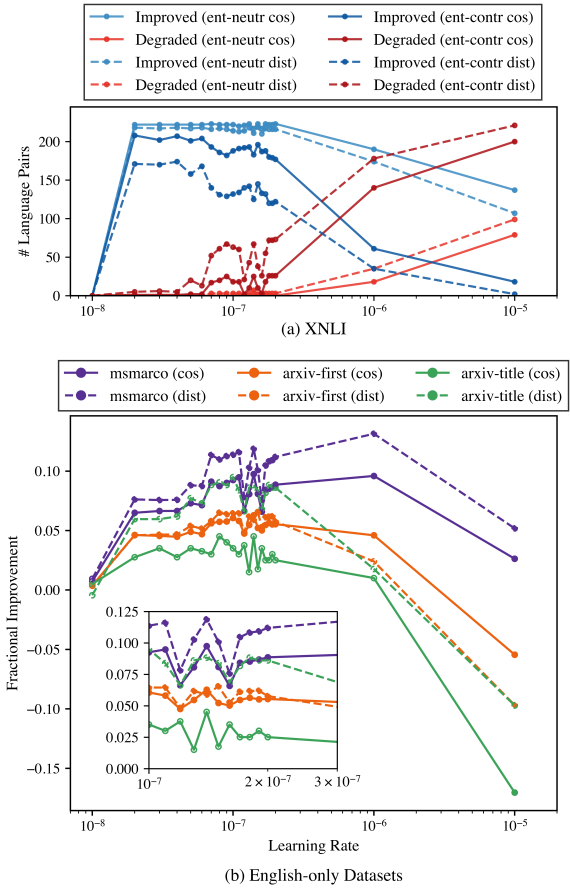


Figure 2: Evaluations on (a) XNLI and (b) the English-only datasets (MSMARCO and ARXIV) of the *E5* query encoder tuned with a frozen embedding block, batch size 14, margin 0.1 using different learning rates. Values that did not pass the two-tailed test are shown with open markers.

and degraded is in Figure 2(a). Appendix L contains the corresponding plots (Figure 11) for the fully tuned *E5* dual encoder, and for the *L12* and *E5e* models. It is interesting that the *E5e* model, not even being multilingual, still improves more than it degrades its rudimentary multilingual qualities. The effects of other tuning parameters are described in Appendix M. For example, the square-root batch size scaling rule works better than linear.

If we consider XNLI and ARXIV as indicators of how well a model keeps the learned skills while improving on narrow goals (e.g. MSMARCO), then our observation suggests there may be a slow tuning regime, at which the model preserves or even improves the existing skills which are at least a little related to the new goal. We call this *adiabatic tuning*, in analogy to the slow process in quantum mechanics (a system starting in an eigenstate is kept in the same evolving eigenstate). For

frozen	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
	c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
-	7.47	8.46	5.19	5.19	1.75	5.52	222/0	215/2	194/4	147/23
emb.base	7.32	8.82	4.85	5.41	3.51	7.73	222/0	217/1	201/2	159/21
emb	7.30	8.82	4.90	5.39	3.51	7.73	222/0	217/1	201/2	158/20
emb, B0a	7.30	8.76	4.77	5.34	3.26	7.73	222/0	217/1	200/2	159/21
emb, B0a,i	7.48	9.00	5.05	5.36	3.26	7.73	223/0	219/0	199/2	156/25
emb, B0a,i,od	7.31	8.82	4.77	5.19	3.51	7.73	222/0	217/1	200/2	158/21
emb, B0	7.35	8.78	4.77	5.44	3.51	7.73	222/0	217/1	200/2	159/19
emb, B0-5	7.87	9.39	5.79	6.07	3.26	7.51	219/0	213/3	200/5	157/25
emb, B0-10	1.45	2.57	0.89	1.21	0.00	0.44	123/0	112/0	21/0	25/10

Table 1: Evaluations of the *E5* query model tuned on MSMARCO as described in Section 2.3. The rows are in the order of increased freezing (at tuning): from no freezing (top row) to freezing everything up to the last transformer block *B11*. The *emb.base* model has only the first three layers of the embedding block frozen (tokens, positions, token-types). The *emb* model has the full embedding block frozen. For the other notation: *B0* is the full first transformer block; *B0-5* are the first 6 blocks; the extensions *a*, *i*, *od* (for *B0*) denote the layers *attention*, *intermediate* and *output.dense* of the block. The columns *c%* and *d%* show the PND improvement (in percents) relative to the original model, accessed by cosine (*c*) or distance (*d*), grayed if not significant (Appendix I). The columns *c+/-* and *d+/-* show count of language pairs with PND significantly improved (+) or worsened (−).

Dataset	PND		MRR		MAP		P@1	
	c%	d%	c%	d%	c%	d%	c%	d%
MSMARCO 65 negatives	2.41	3.78	0.48	0.55	1.03	1.15	1.92	2.02
SQUAD	1.02	1.12	0.17	0.2	0.17	0.19	0.31	0.33
SQUAD min 5	0.85	1.13	0.16	0.24	0.18	0.26	0.32	0.44
HotpotQA easy	2.52	3.47	0.25	0.34	0.09	0.08	0.16	0.12
HotpotQA medium	2.53	3.57	0.33	0.49	0.07	0.09	0.11	0.13
HotpotQA hard	2.43	3.70	0.30	0.50	0.07	0.11	0.12	0.15

Table 2: Improvements for *E5* tuned with frozen embedding block and learning rate 5e-8.

E5 the learning rates between 2e-8 and 6e-8 may be considered as the best.

Our tentative explanation of adiabatic tuning is as follows: At low learning rates of tuning, the system (the encoder weights) remains in the ‘minimum’ region found at pretraining. This ‘minimum’ region is probably a wide well with uneven ground; the pretraining happened to terminate at some point inside the well. During tuning, the pretraining weight-space of twin encoder becomes just another surface in a family of surfaces, because of the added dimensions (the difference between the weights of the two encoders). We assume that due to continuity, the ‘minimum’ region, even if being reshaped, remains a well as the query encoder weights drift away from the weights of the text encoder. Within this well, improvements of all qualities related to the former, pretraining loss, may be still correlated. But if, at high learning rate, the model is strongly modified at some iteration (i.e.

by backpropagation on a particular batch), then it may move away from the well.

3.3 Extending adiabatic tuning range

From evaluation results in Figure 2 we may consider the learning rate below 7e-8 (but above 1e-8) as safely suitable for adiabatic tuning. But we know this only because we evaluated the tuned models on the out-of-tuning domains ARXIV and XNLI.

Is there any way to know the upper boundary without having extensive data for evaluation? Could there be an empirical recommendation not to exceed certain learning rate? Can we increase the adiabatic tuning range of learning rate?

In attempting to answer these questions, we have considered the largest changes in the layers at different learning rates. One suspect layer, by simple crude measures, is *output.dense.weight*. In Appendix M.3 in Tables 14 and 15 we show the most changing layers and the blocks to which they be-

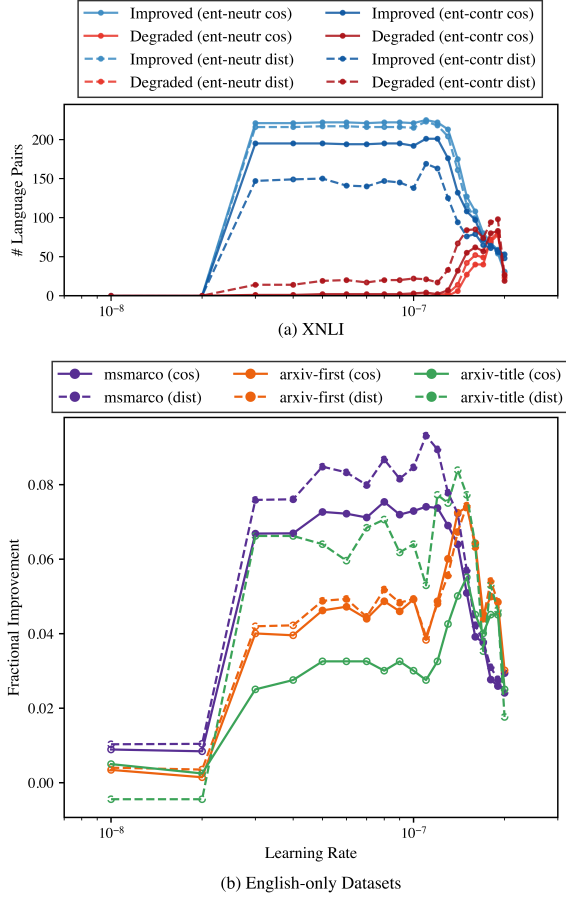


Figure 3: Evaluations of the *E5* query encoder tuned with a frozen embedding block and all layers ‘output.dense.weight’, with batch size 14, margin 0.1 using different learning rates on (a) XNLI and (b) the English-only datasets (MSMARCO and ARXIV). Values that did not pass the two-tailed test are shown with open markers.

long. Our motivation here is based on a simple and crude criteria; more detailed research and understanding may reveal better ways to extend the adiabatic tuning regime.

The gains from the tuning by freezing the layer *output.dense.weight* (in each transformer block) are shown in Figure 3. In comparison to the default tuning (Figure 2) we can see that the adiabatic regime indeed extends from a learning rate of about $6e-8$ (as was in Figure 2) to about $1.3e-7$. Thus, freezing of *output.dense.weight* did help to somewhat extend the adiabatic tuning regime. However, this did not improve the gains, and further increase of the learning rate results in worse deterioration for the version with frozen *output.dense.weight* layer, as can be seen for XNLI starting from the rate $1.4e-7$.

Another way of trying to stay longer in the original ‘minimum’ region during tuning could be by

reducing the inertia of the optimizer. We present a simple attempt in Appendix M.8, but the results are mixed.

4 Conclusion

We considered tuning the query part of a dual encoder starting from a high quality multilingual embedding model, and using English-only samples in the tuning. We found that multilingual qualities are quite stable in many scenarios of the tuning, and can be not only preserved but improved. We explain this by speculating that most of the transformer, except the embedding block, depends weakly on multiple languages. We think of this as a particular case of a general pattern: tuning a certain model quality, if done carefully enough (*adiabatic tuning*), can also retain or even improve the related (but not targeted by tuning) qualities. This allows a resource-light adjustment of multilingual embeddings for a specific query type or domain, even a narrow domain (Appendix K).

Limitations

Our considerations here are limited to starting with a single high quality multilingual embedding model, and tuning it (on English-only samples) as a query encoder. While this setup is good for our understanding and convenient for adjusting an existing model, it would be natural to follow this up by considering a pre-trained multilingual dual encoder which is already asymmetric from the start.

For our illustration we used the state of the art multilingual model *intfloat/multilingual-e5-small*, and also, for comparison, repeated the same observations for the *sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2* model. We also repeated some of observations on monolingual model *intfloat/e5-small-v2* - the tuning improved its rudimentary multilingual properties as well. Still, to gain a better understanding of the observed behaviors, it would be interesting to investigate more multilingual models.

We considered tuning the query encoder on English-only samples, and found that such tuning can “pull up” the quality of other languages too. Choosing another language for tuning would be interesting both for understanding and as a practical scenario.

We used MSMARCO triplets for tuning; we also verified some observations for models tuned on ARXIV-based subsets limited to a category (math,

physics or cs, Appendix K). For evaluation we used a set aside part of MSMARCO triplets, and ARXIV in two variations, and XNLI. The motivation was that the MSMARCO evaluation part must show improvement (after tuning), ARXIV must verify the robustness of the improvement on a very different kind of texts (jargon-heavy), and XNLI must reveal the effect of the English-only driven improvement on multilingual qualities. We also confirmed the tuning gains on SQUAD and HotpotQA (both of which are quite different from MSMARCO). That said, the evaluations can be extended to even more datasets.

More research could be helpful in understanding and identifying the range of adiabatic tuning.

References

- Daniel Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. [Universal sentence encoder](#). *arXiv*, arXiv:1803.11175.
- Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. [Supervised learning of universal sentence representations from natural language inference data](#). In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.
- Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. [XNLI: Evaluating cross-lingual sentence representations](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.
- Zhe Dong, Jianmo Ni, Dan Bikel, Enrique Alfonseca, Yuan Wang, Chen Qu, and Imed Zitouni. 2022. [Exploring dual encoder architectures for question answering](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9414–9419, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. 2018. [Accurate, large minibatch sgd: Training imagenet in 1 hour](#). *arXiv*, arXiv:1706.02677.
- Elad Hoffer, Itay Hubara, and Daniel Soudry. 2018. [Train longer, generalize better: closing the generalization gap in large batch training of neural networks](#). *arXiv*, arXiv:1705.08741.
- Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. 2021. [Efficiently teaching an effective dense retriever with balanced topic aware sampling](#). In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 113–122, New York, NY, USA. Association for Computing Machinery.
- Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. [Dense passage retrieval for open-domain question answering](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.
- Alex Krizhevsky. 2014. [One weird trick for parallelizing convolutional neural networks](#). *arXiv*, arXiv:1404.5997.
- Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, Xuezhe Ma, Antonios Anastopoulos, Patrick Littell, and Graham Neubig. 2019. [Choosing transfer languages for cross-lingual learning](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3125–3135, Florence, Italy. Association for Computational Linguistics.
- Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2018. [MS MARCO: A human generated Machine Reading Comprehension dataset](#). *arXiv*, arXiv:1611.09268.
- 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. 2022. [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.
- Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. [RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering](#). In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5835–5847, Online. Association for Computational Linguistics.
- Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. [Know what you don’t know: Unanswerable questions for SQuAD](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. [SQuAD: 100,000+ questions for](#)

- machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.
- Nils Reimers and Iryna Gurevych. 2019. [Sentence-BERT: Sentence embeddings using Siamese BERT-networks](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.
- Houxing Ren, Linjun Shou, Ning Wu, Ming Gong, and Daxin Jiang. 2022. [Empowering dual-encoder with query generator for cross-lingual dense retrieval](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3107–3121, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Ruiyang Ren, Shangwen Lv, Yingqi Qu, Jing Liu, Wayne Xin Zhao, QiaoQiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021a. [PAIR: Leveraging passage-centric similarity relation for improving dense passage retrieval](#). In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2173–2183, Online. Association for Computational Linguistics.
- Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, QiaoQiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021b. [RocketQAv2: A joint training method for dense passage retrieval and passage re-ranking](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2825–2835, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2024a. [Text embeddings by weakly-supervised contrastive pre-training](#). *arXiv*, arXiv:2212.03533.
- Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024b. [Multilingual E5 text embeddings: A technical report](#). *arXiv*, arXiv:2402.05672.
- Yau-Shian Wang, Ashley Wu, and Graham Neubig. 2022. [English contrastive learning can learn universal cross-lingual sentence embeddings](#). *arXiv*, arXiv:2211.06127.
- Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. [Approximate nearest neighbor negative contrastive learning for dense text retrieval](#). *arXiv*, arXiv:2007.00808.
- 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.
- Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. [Optimizing dense retrieval model training with hard negatives](#). In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’21, page 1503–1512, New York, NY, USA. Association for Computing Machinery.
- Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Min Zhang, and Shaoping Ma. 2020. [RepBERT: Contextualized text embeddings for first-stage retrieval](#). *arXiv*, arXiv:2006.15498.
- Shengyao Zhuang, Linjun Shou, and Guido Zuccon. 2023. [Augmenting passage representations with query generation for enhanced cross-lingual dense retrieval](#). In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’23, page 1827–1832, New York, NY, USA. Association for Computing Machinery.

A Usage of MSMARCO Triplets

The MSMARCO dataset consists of 499184 samples, with each sample being a tuple given as (query, positives, negatives). The “positives” are the correct answers to the query, and the “negatives” are semantically similar, but incorrect answers. For most samples, there is only one positive, but many negatives. For our tuning we simply select the very first positive and the very first negative. Thus, each sample gives one triplet (anchor, positive, negative) for contrastive learning, where the query is taken as an anchor.

We keep the first 487983 samples (or 34856 batches if each batch is 14 triplets) for tuning, leaving the next 4200 samples (300 batches) for validation, and the last 7000 samples for evaluation. During evaluation we create all possible triplets from the 7000 samples, using all positives and negatives; this makes 357642 evaluation triplets.

Almost half of MSMARCO samples have the maximal number of negatives (65), and for evaluation shown in Table 2 we use a more difficult version ‘MSMARCO 65 negatives’, with all samples with less than 65 negatives filtered out.

B ARXIV Dataset for Triplets

B.1 Dataset arxiv-negatives

Of ARXIV we use titles and abstracts. In order to have a representative subset of a manageable

size for our evaluations, we select all samples that have at least one category with a maximum size of 10K samples. For example, the arxiv category *bayes-an* is the smallest (size 16) in our snapshot (version 173), meaning that there were only 16 arxiv preprints in this category.

We made two flavors of evaluation arxiv triplets from this arxiv subset. In the first version, the anchor is the title, the positive is the corresponding abstract, and the negative is another random abstract. In the second version the anchor is the first sentence of the 'positive' abstract, the positive is the rest of the abstract, and the negative is a similar piece (first sentence excluded) of the 'negative' abstract.

We make use of triplets created from arxiv because this provides our evaluation with a very different kind of text (compared to MSMARCO), and thus allows us to judge the robustness of the improvement. For convenience and reproducibility of creating triplets of different levels of difficulty, we made a dataset *arxiv-negatives*¹¹. The dataset consists of 253140 samples, each sample is a tuple of two elements:

1. An ARXIV paper metadata, including its Id, title and abstract and categories.
2. List of 21 Ids of other ARXIV papers. The first 20 Ids are the papers that are 'closest' to the above paper, and sorted from the most to the least similar; the last 21st Id is an Id of a randomly selected paper (not coinciding with Id of the above paper).

Thus, we have 21 versions of picking up negatives for triplets, from the most difficult to the easiest (the last one, of the random selection).

For example, to create triplets of difficulty 14, for each paper given by the first tuple element, we pick up a paper corresponding to 14th Id given in the second tuple element. From the first paper we can create query and positive, and from the second paper, negative. Through this work we used two flavors:

1. 'Title': The title of the first paper acts as the query and its abstract as the positive; the negative is then the abstract of the second paper.
2. 'First': The query is the first sentence of the abstract of the first paper; the positive is the rest of the abstract; the negative is the abstract of the second paper, with its first sentence

deleted.

B.2 How is it created?

The above dataset is created from the mirror of arxiv (version 173) *arxiv-metadata-oai-snapshot.jsonl* through the following steps:

1. Identified all arxiv categories with a maximum size of 10K papers (i.e. arxiv preprints).
2. Selected all papers that have at least one of the categories identified above. This is the subset of arxiv to deal with: manageably small, yet diverse.
3. For each paper: (1) Sort its categories by size, from smaller to larger. (2) Find all other papers that have the closest match by the categories (the closest match is the longest consecutive list of matched categories, starting from the first one). (3) Of the found papers, select 20 closest by Jensen-Shannon distance between the paragraphs, and sort them by the distance. If there were less than 20 papers, fill to 20 by the last one. (4) Add randomly selected paper as 21st.

Of the total 253140 samples, in 213156 samples (84.2%) all the first 20 negatives are different (which means that not less than 20 papers happen to have the same closest match by categories).

C SQUAD

For using the SQUAD dataset, we identified (for each query) the given paragraph sentences containing an answer to the query as positives, and the rest of the sentences as negatives. We left samples having at least 1 positive and 1 negative. On average there is 1.3 positives and 4.2 negatives per a query. For the evaluation shown in Table 2 we combined train, validation and test subsets. The results are given also for a version called 'SQUAD min 5', in which we have filtered out queries that had less than 5 candidate sentences.

D HotpotQA

For using HotpotQA, we combined its train and dev subsets. For each query ('question') both train and dev subsets contain on average 9.95 passages, of which 2 are always positives. For the evaluation shown in Table 2 we filtered out queries that had less than 10 passages, and split the dataset into 'easy', 'medium' and 'hard' subsets accordingly to the HotpotQA labels of the difficulty of the samples.

¹¹<https://huggingface.co/datasets/primer-ai/arxiv-negatives>

E Tuning

Unless specified otherwise, we tune a dual encoder by contrastive learning in the following simple regime:

1. The text encoder is fully frozen; the frozen parts of the query encoder are specified.
2. The batch size is 14, the learning rate is $5e-8$ and the contrastive learning margin is 0.1. The loss is defined by the triple margin loss.
3. There are 1000 batches per epoch, i.e. 14000 samples per epoch.
4. Stopping occurs after 10 consecutive non-improvement epochs. The improvement is measured on the validation subset after each epoch. The model is considered to be improved if (on the validation subset) both the loss and the count of errors have decreased.
5. The AdamW optimizer is used.

Changing this default regime is considered in Appendixes L, M.

F XNLI

The XNLI dataset consists of pairs of sentences which are human-labeled as entailment, neutral or contradiction. The test subset (which we use) contains 1670 pairs for each of these labels and each sentence is presented in 15 languages: ['ar', 'bg', 'de', 'el', 'en', 'es', 'fr', 'hi', 'ru', 'sw', 'th', 'tr', 'ur', 'vi', 'zh']. We use 225 versions of the pairs, because each sentence of the pair can be in any of the 15 languages. At evaluation the first sentence serves as the query (the embedding is taken by the query model), and the second one as the text. We expect that the sentences of an entailment pair should be closer to each other than the sentences of any neutral pair, or of any contradiction pair. Whenever this does not happen, we count this as an error.

G Performance of Untuned Query Encoder

To establish a baseline before any fine-tuning, and to ensure our evaluation is not too easy, we measure the errors of the original *E5* model on the data described in Section 2.3 and show the results in Table 3. We also measure the errors of *L12* and of *E5e* - a more recent monolingual (English) model.

The count of errors on the triplets (MSMARCO, ARXIV) is straightforward: it is an error when a positive is not closer than a negative to the anchor of the triplet. On XNLI we sum up the error count

data	Evaluation	<i>E5</i>	<i>L12</i>	<i>E5e</i>
MM	N tot	357642		
	PND (cos)	4.7%	15.1%	4.6%
	PND (dist)	4.8%	15.4%	4.5%
ARX-F	N tot	253140		
	PND (cos)	1.6%	4.9%	3.1%
	PND (dist)	1.6%	6.7%	3.5%
ARX-T	N tot	253140		
	PND (cos)	0.2%	1.4%	0.2%
	PND (dist)	0.2%	1.7%	0.2%
XNLI	N total	2788900		
	PND e-n (cos)	10.8%	10.2%	15.9%
	PND e-c (cos)	10.0%	7.2%	15.3%
	PND e-n (dist)	10.5%	10.1%	15.9%
	PND e-c (dist)	9.6%	7.8%	15.4%

Table 3: The count of errors for the original untuned models *E5*, *L12* and *E5e*, on the datasets noted in the first column: *MM* - MSMARCO test 7000 samples (357642 triplets, see Section 2.2 and Appendix A); *ARX-F* - arxiv-first, the arxiv subset with the abstract’s first sentence as an anchor; *ARX-T* - arxiv-title, the arxiv subset with the title as an anchor; *XNLI* - XNLI test subset providing $1670 \times 1670 = 2788900$ comparisons of entailment pairs vs neutral pairs (and the same amount of entailment pairs vs contradiction pairs). For XNLI the errors are averaged over 225 (15x15) language-language versions, and shown as percent of *Ntotal*. The evaluation is done using cosine similarity or euclidean distance similarity (*cos* or *dist* in second column).

over all language-language pairs and divide the sum by the number ($255 = 15 \times 15$) of such pairs. This averaged error is shown as a percentage of the total (2788900) comparisons; each comparison here is either a comparison of an entailment-labeled sample with a neutral-labeled sample (*entail-neutral* in the table) or a comparison of an entailment-labeled sample with a contradiction-labeled sample (*entail-contr* in the table). An error was counted whenever the sentences of an entailment sample happened to be farther from each other than the sentences of a neutral (or contradiction) sample. Separately for each pair of languages PND is shown in Figures 4, 5 for cosine similarity measure. The distance measure gives results visually almost undistinguishable.

The amount of errors in Table 3 and in Figures 4, 5 is reasonable enough to consider how tuning would affect them. The smallest counts are the counts of positive-negative discrepancies of *E5* and *E5e* on ARX-T (apparently, a title makes an easier ‘query’ than the first sentence of an abstract).

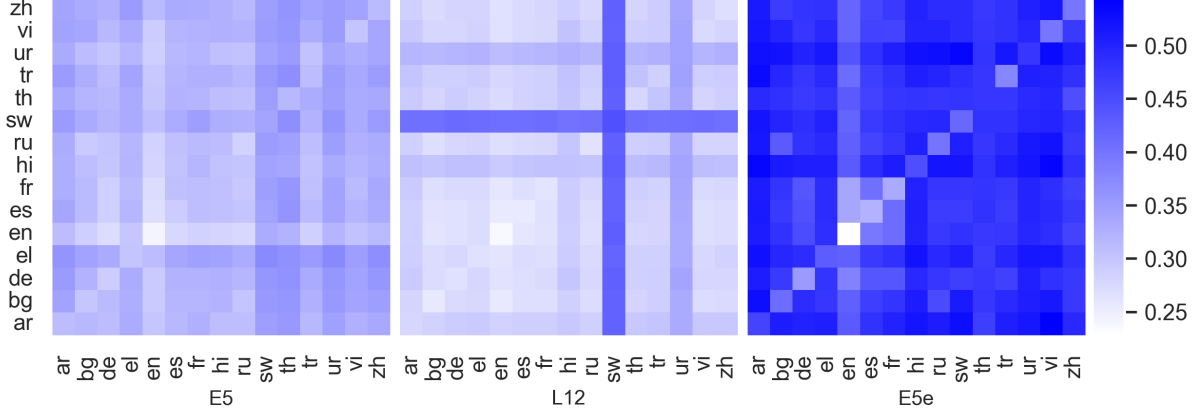


Figure 4: PND of embedding models on XNLI entailment-neutral comparisons assessed by cosine.

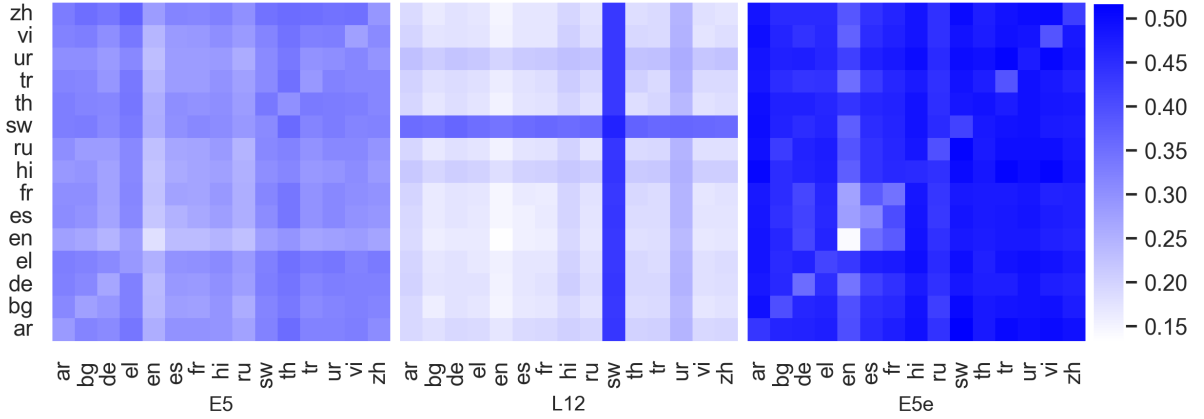


Figure 5: PND of embedding models on XNLI entailment-contradiction comparisons assessed by cosine.

These counts are 309 and 420 for the cosine similarity (the row ARX-T PND (cos)), and 453 and 580 for the distance similarity (the row ARX-T PND (dist)).

Notice that *L12* has far worse PND on English data (MSMARCO and ARXIV). The English-only model *E5e*, as expected, performs worse than multilingual models *E5* and *L12* on multilingual XNLI, but its PND is still far below 50%, because there is much similarity between some of the languages.

H Gains on ARXIV for Different Levels of Difficulty

Throughout the paper we used the easiest version of triplets in the arxiv-negatives dataset, the version that uses randomly selected negatives. Here in Figure 6 we show, for comparison, the fraction of the errors which occur in the original untuned *E5* embeddings using the other levels of difficulty, and also the corresponding improvements (by Equa-

tion 1) after tuning the query encoder on the MS-MARCO with frozen embedding block and our default settings (Section 2.3). The statistical significance of the improvements in Figure 6 is estimated as explained in Appendix I.

The difficulty of intentionally close negatives is much harder, but Figure 6 still shows that performance on ARXIV was mostly improved. We used the easiest triplets version for our evaluations throughout the paper because it more distinctly indicated the trends in the improvements.

I Significance Test

In Table 1, Figure 2 and through the paper we use two-proportion *Z*-test, pooled for $H_0 : p_1 = p_2$. We are comparing the number of errors original n_0 and improved n_1 , having the total N (the totals can be seen in Table 3); a total is the same for original and improved version. We deem the difference to

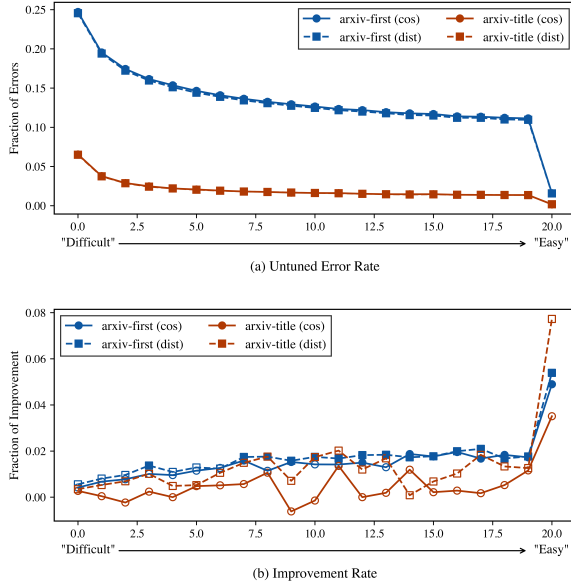


Figure 6: Errors and improvements on arxiv-negatives dataset of different level of difficulty. The “easiest” dataset is a random selection of negatives from the same data used through this work in evaluations. In (a), we show the fraction of errors done by the original $E5$ model (for comparison, see Table 3). In (b), we show the improvement after tuning the query encoder on MSMARCO, with ‘default’ settings, i.e. learning rate $5e-8$, batch size 14, margin 0.1 and frozen embedding block. Values that did not pass the two-tailed test (Appendix I) are shown with open markers.

be significant if $|Z| > Z_c$ where

$$Z = \frac{p_1 - p_0}{\sqrt{\frac{1}{2}P(1-P)N}} \quad (2)$$

with $p_0 = n_0/N$, $p_1 = n_1/N$ and $P = \frac{1}{2}(n_0 + n_1)/N$. We used $Z_c = 1.96$, which is a critical value corresponding to probability 0.975.

Notice that in our examples the values N are typically very large. And the improvements we report, according to Equation 1, are relative, not absolute values.

J Encoder L12 with Frozen Layers

Table 4 shows results of tuning with freezing some of $L12$ layers. It is similar to the Table 1 for $E5$. And, similar to $E5$, freezing everything except the embedding, resulted in negligible changes of the query encoder (not shown in the table).

The changes in cross-lingual qualities corresponding to the third row (*emb*, frozen embedding block) of Table 4 are shown in comparison with $E5$ and $E5e$ embeddings in Figures 7 and 8. Note

that $E5e$ is not a multilingual embedding model. Having a worse start as a multilingual embedding model, $E5e$ also gets much weaker improvements of its multilingual qualities; it is consistent with our understanding of adiabatic tunings (Section 3.2).

K Narrow-Domain Query Encoder

So far we observed that tuning the query encoder on data of a certain style (MSMARCO dataset) could preserve (or even improve) the encoder qualities which are not targeted by the tuning task, especially if we tune with a frozen embedding layer and low learning rate. Here we provide observations using more specialized datasets, based on arxiv-first (arxiv-first is described in Section 2.2 and Appendix B):

1. ARXIV-math: uses only documents with at least one category which has the prefix "math."
2. ARXIV-physics: As above, but with "physics." as the prefix
3. ARXIV-cs: As above, but with "cs." as the prefix

$E5$ tuned on these narrow datasets using our ‘default’ regime (Section 2.3) with frozen embedding block mostly improves the PND (positive-negatives discrepancy fraction) as shown in Table 5. The improvements of these narrow-tuned encoders on individual language pairs, assessed by cosine, are shown in Figures 9 and 10.

L Learning Rate

In Figure 2 we have shown how the improvements of the $E5$ model depend on the learning rate. Here in Figure 11 we compare similar data for $L12$ and $E5e$ as well as a particular instance of $E5$ when both the query and text encoder are subject to tuning (as two independent encoders, with the same starting point) with the embedding block frozen in both encoders. The data confirm that while higher learning rates are not yet overtuning and still give higher gains on the test subset (of MSMARCO), it is the lower learning rates that better preserve and even improve those pretrained qualities which are not the goal of tuning.

M Tuning Regime

M.1 Learning rate and batch size

M.1.1 Scaling rule

The learning rate is usually set with consideration to the batch size; it can be proportional to

frozen	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
	c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
-	6.60	6.93	2.65	-7.86	14.46	-0.12	206/15	57/35	201/15	15/189
emb.base	7.28	8.04	2.46	-9.17	12.38	-1.03	200/15	47/47	206/15	20/142
emb	7.28	8.04	2.51	-9.17	12.4	-1.03	200/15	47/47	206/15	20/143
emb, B0a	7.26	8.03	2.29	-9.22	12.52	-0.96	201/15	46/46	206/15	20/142
emb, B0a,i	7.03	7.75	2.44	-8.6	12.46	-0.63	203/15	51/41	206/15	20/133
emb, B0a,i,od	9.04	10.15	1.80	-17.54	12.49	-8.58	195/19	30/116	207/15	19/167
emb, B0	8.92	9.98	1.78	-16.73	12.88	-8.07	195/16	33/102	209/15	20/163
emb, B0-5	8.54	9.68	2.71	-19.01	12.35	-12.17	209/15	28/129	209/15	19/172
emb, B0-10	0.11	0.15	0.10	-0.12	0.25	-0.02	0/0	0/0	0/0	0/0

Table 4: Evaluations of the $L12$ query model tuned on MSMARCO as described in Section 2.3. The notations are as in Table 1.

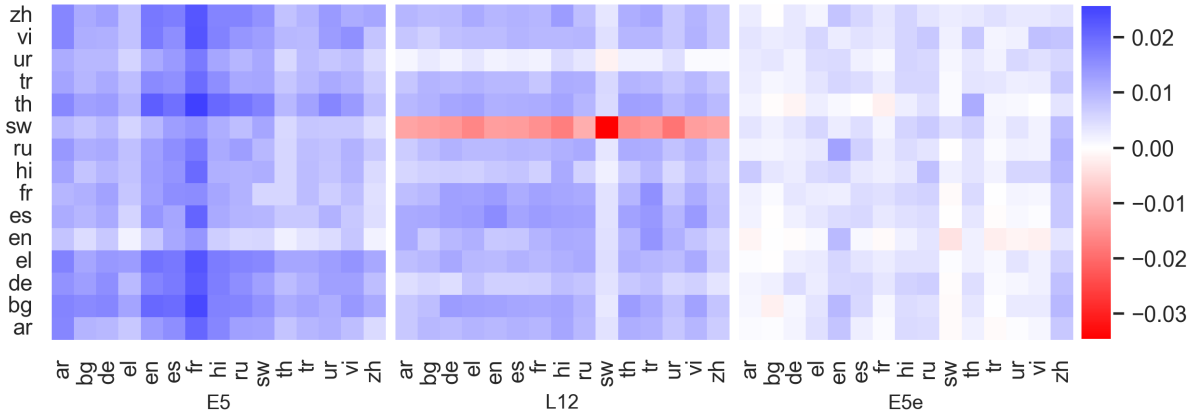


Figure 7: Improvement of $E5$, $L12$ and $E5e$ on XNLI entailment-neutral comparisons assessed by cosine.

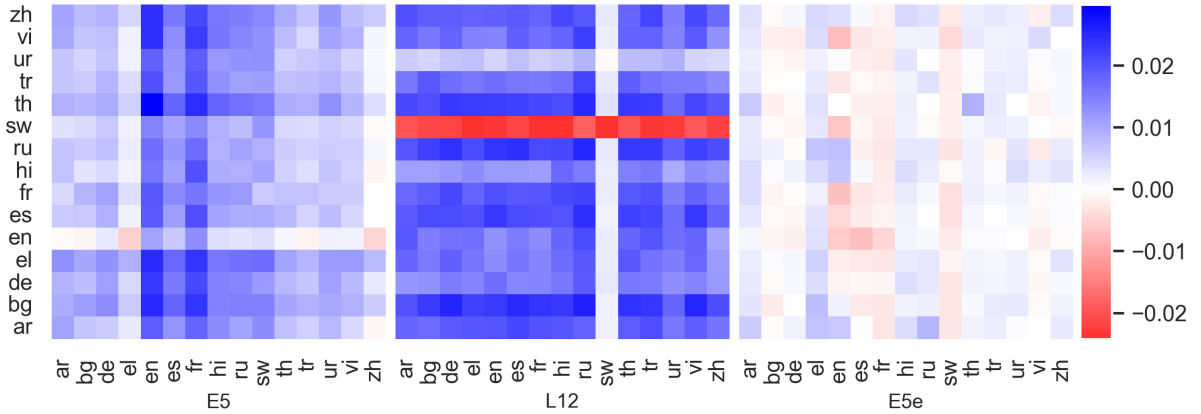


Figure 8: Improvement of $E5$, $L12$ and $E5e$ on XNLI entailment-contradiction comparisons assessed by cosine.

the batch size (linear scaling rule), or proportional to square root of the batch size (square root scaling rule) (Krizhevsky, 2014; Goyal et al., 2018; Hoffer et al., 2018). We show the evaluation results for these scaling rules in Tables 6 and 7. While there is no essential wins in scaling batch size and learning rate up or down, the square root rule seems

more reasonable in keeping the evaluation results approximately the same while increasing the batch size.

Regardless of the overall behavior of scaling the batch size and learning rate together, we have to verify that our default batch size 14 is a good fit for our default learning rate $5e-8$. For this reason,

model	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
	c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
E5-math	0.04	0.45	54.71	52.85	50.38	53.64	218/0	209/0	177/0	133/0
E5-physics	0.16	0.57	19.05	18.64	21.8	20.97	162/0	102/0	31/0	2/0
E5-cs	0.18	0.55	23.32	23.63	25.56	26.49	205/0	136/0	51/0	8/8

Table 5: Evaluations of the *E5* query encoder tuned on ARXIV-math, ARXIV-physics or ARXIV-cs with a frozen embedding block, batch size 14, margin 0.1 and learning rate 5e-8. When evaluated on ARXIV (columns arxiv-first and arxiv-title) the samples with category of the model (the first column) are excluded from the evaluation data.

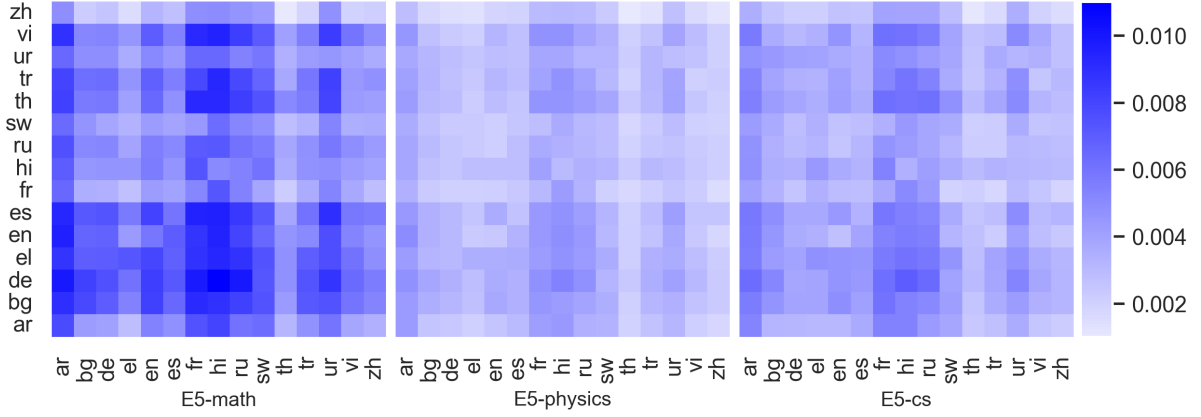


Figure 9: Improvement of narrow-tuned encoders on XNLI entailment-neutral comparisons assessed by cosine.

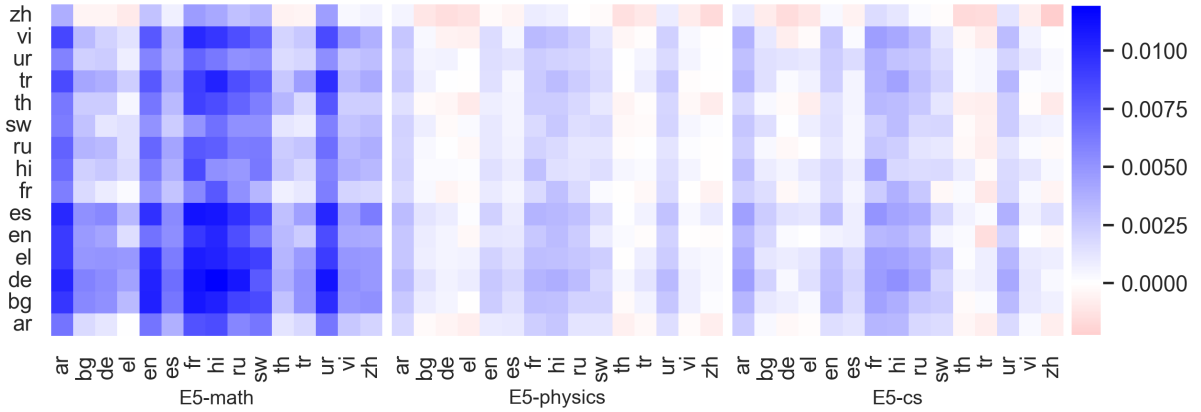


Figure 10: Improvement of narrow-tuned encoders on XNLI entailment-contradiction comparisons assessed by cosine.

batch size	learning rate	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
		c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
7	2.5e-8	6.62	7.67	4.48	5.06	3.26	6.18	221/0	216/0	201/2	160/13
14	5.0e-8	7.30	8.82	4.90	5.39	3.51	7.73	222/0	217/1	201/2	158/20
28	1.0e-7	8.36	10.31	5.71	6.75	3.26	7.95	222/0	218/3	177/20	128/58
56	2.0e-7	8.32	10.54	5.39	6.75	2.76	7.51	222/0	217/3	193/14	141/42
112	4.0e-7	8.46	10.36	5.24	6.00	3.26	8.39	221/0	216/3	197/8	147/35

Table 6: Evaluations of the *E5* query encoder tuned with a frozen embedding block, margin 0.1 and 14000 samples per epoch. Linear scaling rule of learning rate with batch size. Values that did not pass the two-tailed test are shown in gray.

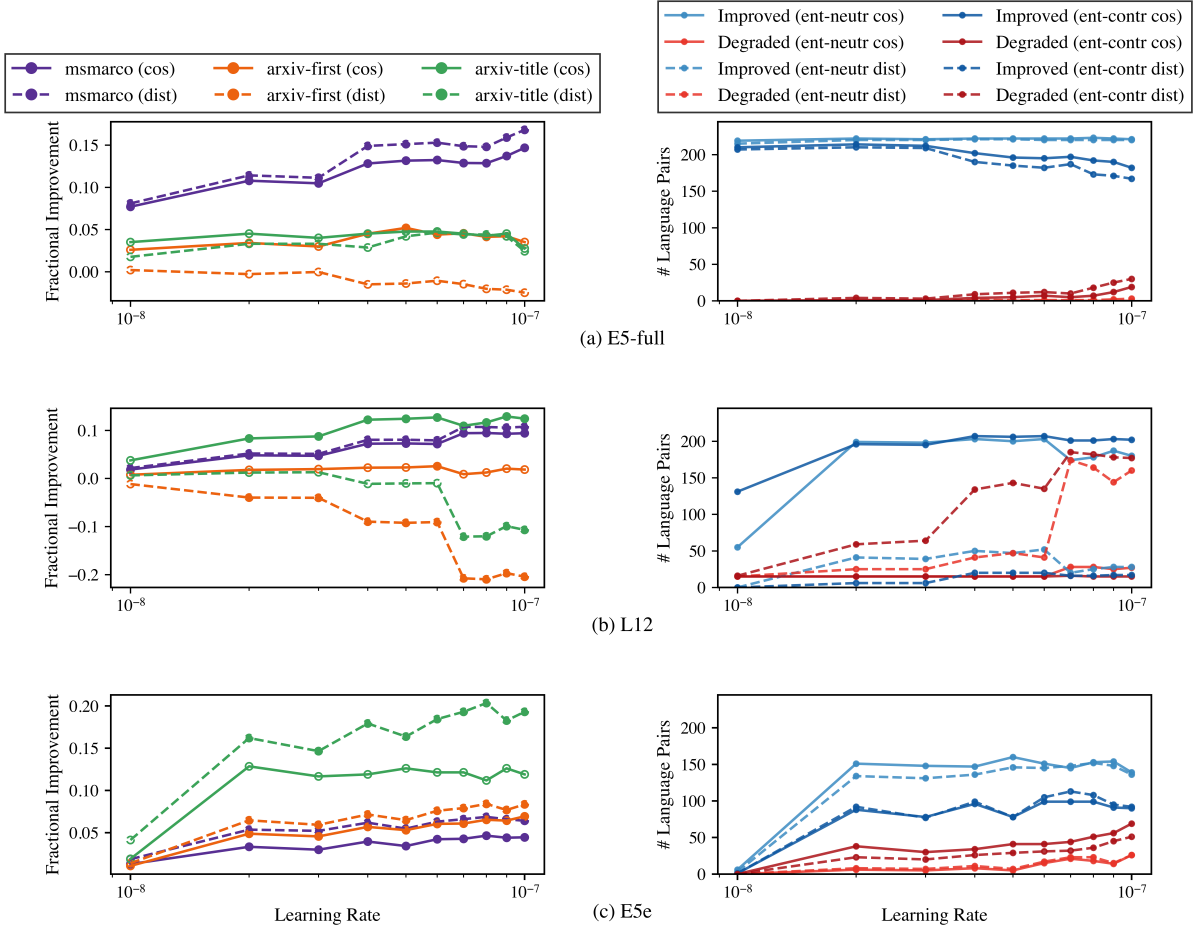


Figure 11: Improvement of various models and tuning configurations on the English-only datasets (MSMARCO and ARXIV) in the left column and XNLI in the right column. Values that did not pass the two-tailed test (Appendix I) are shown with open markers. (a) Evaluations of the *E5*-full dual encoder after both encoders were tuned with a frozen embedding block, batch size 14 and margin 0.1. (b) Evaluations of the *L12* query encoder tuned with a frozen embedding block, batch size 14 and margin 0.1. (c) Evaluations of the *E5e* query encoder tuned with a frozen embedding block, batch size 14 and margin 0.1.

batch size	learning rate	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
		c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
7	3.54e-8	6.80	7.84	4.43	4.76	2.26	6.18	221/0	216/0	192/2	148/17
14	5.00e-8	7.30	8.82	4.90	5.39	3.51	7.73	222/0	217/1	201/2	158/20
28	7.07e-8	7.16	9.10	4.70	5.57	4.01	7.95	222/0	217/1	192/4	140/32
56	1.00e-7	7.32	8.56	4.70	5.16	3.01	6.40	221/0	217/0	204/2	169/13
112	1.41e-7	7.25	8.55	5.24	4.91	4.01	7.51	222/0	217/0	202/2	166/16

Table 7: Evaluations of the *E5* query encoder tuned with a frozen embedding block, margin 0.1 and 14000 samples per epoch. Square root scaling rule of learning rate with batch size. Values that did not pass the two-tailed test are shown in gray.

a simple change of batch size, without altering learning rate, is considered in Appendix M.1.2; the tables 8 and 9 show that our ‘default’ batch size is reasonable. The corresponding data for *L12* are in Appendix M.1.3.

M.1.2 Encoder E5 and the batch size

In Table 8 we show results for batch sizes 7, 14, 28, 56 and 112, while keeping the number of samples per epoch the same (14000). The row with batch 14 here coincides with the values for learning rate 5e-8 in Figure 2, and with the row for the frozen embed-

batch size	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
	c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
7	6.85	7.57	4.55	4.78	3.01	6.62	221/0	217/0	198/2	163/13
14	7.30	8.82	4.90	5.39	3.51	7.73	222/0	217/1	201/2	158/20
28	7.45	9.47	5.14	5.46	4.01	8.39	222/0	219/1	196/6	145/27
56	6.51	7.33	4.16	4.48	2.51	5.74	221/0	217/0	202/1	168/6
112	4.63	4.78	2.55	2.50	2.51	2.21	212/0	203/0	196/0	155/1

Table 8: Evaluations of the *E5* query encoder tuned with a frozen embedding block, learning rate 5e-8, margin 0.1 and different batch sizes (first column); 14000 samples per epoch. Values that did not pass the two-tailed test are shown in gray.

batch size	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
	c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
7	6.76	7.99	4.38	5.06	3.26	6.18	221/0	216/0	177/7	119/31
14	7.30	8.82	4.90	5.39	3.51	7.73	222/0	217/1	201/2	158/20
28	8.50	10.18	5.71	6.45	2.76	8.17	222/0	218/3	197/9	147/36
56	7.42	9.12	4.55	4.81	3.76	8.17	223/0	220/0	200/2	160/21
112	9.50	11.84	-0.82	-4.05	-15.54	-14.13	175/35	146/65	91/117	32/175

Table 9: Evaluations of the *E5* query encoder tuned with a frozen embedding block, learning rate 5e-8, margin 0.1 and different batch sizes (first column); 1000 batches per epoch. Values that did not pass the two-tailed test are shown in gray.

ding block in Table 1. The results for all batch sizes are similar. Tuning with the higher batch size of 112 is a bit ‘safer’ for languages, not degrading any language pair when evaluated by cosine measure, and degrading only one language pair (for entailment vs. contradiction) when evaluated by distance measure. This comes at the price of lower gains on MSMARCO and ARXIV.

Table 9 shows what happens if the number of batches per epoch (1000) is kept the same, rather than the number of samples. In this setting the larger batch size of 112 leads to a less frequent validation (by MSMARCO validation subset) at tuning and, effectively, to later and less reasonable stopping. This results in higher gains on MSMARCO test subset, but in far worse results on ARXIV and XNLI.

M.1.3 Encoder L12 and the batch size

The dependency of tuning *L12* using different batch size is shown in Table 10 (number of samples per epoch is 14000) and in Table 11 (number of batches per epoch is 1000). Observations are somewhat similar to *E5* (Appendix M.1.2), except that generally *L12* does not perform as well as *E5* and a batch size of 7 turns out to be bad for *L12*.

M.2 Weight decay

A weight decay may restrict increase of model weights, but it does not improve the evaluation results. We show some representative results in Tables 12 and 13. While restricting gains on the tuning goal, weight decay does not help to preserve the other qualities: the results on XNLI and ARXIV are no better than without weight decay. If there is any recipe for further improving the gains both on the tuning goal and on the related qualities, it has to be a less crude interference into the tuning.

Since weight decay may be more effective at higher learning rates, the parameters for Table 12 are chosen at higher rate and batch size, compared to our ‘default’ choice, which is used in Table 13. The learning rates and batch sizes of these tables relate by square root scaling rule (see Section M.1.1).

M.3 Candidate layers for freezing

In Section 3.3 we showed how the adiabatic tuning range gets extended when the layer *output.dense.weight* is frozen (in all blocks). The reason for suspecting that this layer is the most responsible for breaking out of the original ‘minimum’ region, is that its maximal weight becomes the highest among all the layers as the learning rate gets closer to the end of the adiabatic range: see Table 14. The maximal relative change of the weights

batch size	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
	c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
7	8.74	9.89	1.59	-16.43	9.59	-9.76	195/18	33/106	205/16	18/168
14	7.28	8.04	2.28	-9.23	12.4	-1.03	200/15	47/47	206/15	20/143
28	5.02	5.49	1.98	-4.05	8.37	1.27	199/15	36/27	196/15	7/77
56	5.05	5.35	1.73	-4.3	8.66	0.63	197/15	35/28	195/15	6/89
112	4.68	5.01	1.69	-3.64	7.47	0.77	193/15	25/25	188/15	4/86

Table 10: Evaluations of the *L12* query encoder tuned with a frozen embedding block, learning rate 5e-8, margin 0.1 and different batch sizes (first column); 14000 samples per epoch. Values that did not pass the two-tailed test are shown in gray.

batch size	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
	c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
7	6.47	8.48	-1.79	-24.53	-1.1	-25.33	12/198	3/212	147/52	9/208
14	7.28	8.04	2.28	-9.23	12.4	-1.03	200/15	47/47	206/15	20/143
28	9.54	10.78	1.11	-20.67	9.16	-13.06	172/29	18/175	201/16	16/185
56	9.51	10.86	1.04	-21.01	8.37	-13.63	173/28	18/175	201/16	17/181
112	9.44	10.88	1.06	-20.86	8.18	-13.77	174/28	21/174	200/17	15/188

Table 11: Evaluations of the *L12* query encoder tuned with a frozen embedding block, learning rate 5e-8, margin 0.1 and different batch sizes (first column); 1000 batches per epoch. Values that did not pass the two-tailed test are shown in gray.

weight decay	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
	c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
100	2.77	1.88	-2.47	-1.06	4.01	1.32	84/104	80/99	90/96	71/92
50	5.39	5.00	0.49	2.12	3.26	2.43	120/51	121/41	140/51	133/46
10	7.08	8.36	3.54	5.11	4.26	6.62	222/0	217/0	201/3	160/18
5	7.88	9.84	4.97	5.87	3.01	7.95	222/0	216/2	189/15	144/36
1	7.26	8.70	4.72	5.11	3.01	7.06	222/0	218/0	202/2	164/16
0.5	7.28	8.78	4.87	5.24	3.01	7.06	221/0	218/0	202/2	163/18
0.1	7.32	8.56	4.70	5.16	3.01	6.40	221/0	217/0	204/2	169/13
0.05	7.32	8.56	4.70	5.16	3.01	6.40	221/0	217/0	204/2	169/13

Table 12: Evaluations of the *E5* query encoder tuned with a frozen embedding block, learning rate 1e-7, batch size 56, margin 0.1 and a range of weight decay (first column). Values that did not pass the two-tailed test are shown in gray.

weight decay	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
	c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
5	6.53	7.26	3.81	4.50	3.51	5.96	222/0	216/0	197/2	150/14
1	7.26	8.73	4.77	5.39	3.76	7.95	222/0	219/0	201/2	160/20
0.5	7.30	8.82	4.90	5.39	3.51	7.73	222/0	217/1	201/2	158/20
0.1	7.30	8.82	4.90	5.39	3.51	7.73	222/0	217/1	201/2	158/20

Table 13: Evaluations of the *E5* query encoder tuned with a frozen embedding block, learning rate 5e-8, batch size 14, margin 0.1 and a range of weight decay (first column). Values that did not pass the two-tailed test are shown in gray.

is also achieved by the layer *output.dense.weight*: see Table 15.

It is a crude adjustment, and freezing this layer in all blocks is probably overkill, but this did help

us in extending the adiabatic range (Section 3.3).

rate	layer
1e-8	1.intermediate.dense.bias
	3.intermediate.dense.bias
2e-8	5.intermediate.dense.weight
	3.attention.output.LayerNorm.weight
3e-8	5.intermediate.dense.weight
	1.attention.output.LayerNorm.weight
4e-8	5.intermediate.dense.weight
	3.attention.output.LayerNorm.weight
5e-8	3.output.dense.weight
	2.output.dense.weight
6e-8	3.output.dense.weight
	2.output.dense.weight
7e-8	3.output.dense.weight
	2.output.dense.weight
8e-8	1.output.dense.weight
	3.output.dense.weight
9e-8	1.output.dense.weight
	4.output.dense.weight
1e-7	1.output.dense.weight
	5.output.dense.weight

Table 14: The ‘most changed’ two layers at each learning rate. The ‘change’ is defined as the maximal weight of the layer *if* it was changed by the tuning. The prefix ‘encoder.layer’ is removed from the layer names here.

M.4 Margin of triple loss

When using the triplet loss for contrastive learning, the margin is an important parameter that can significantly affect model training. In Figure 12 we show the dependency of the evaluation results on the margin during its tuning. We consider our default tuning parameters (Section 2.3), but change the margin. The results are not unexpected: a margin up to 0.15 is reasonable, and at higher margins the disturbance on cross-lingual, and, eventually, on English data evaluation becomes too strong.

The corresponding data for *L12* are given in Figure 13. It shows that a margin of 0.1 works best for *L12*. The results for margin 0.1 are distinctly better. Altogether, *L12* appears to be more sensitive (compared to *E5*) to the tuning parameters if the goal is to preserve performance on multilingual XNLI data and on out-of-domain ARXIV data. Arguably, the margin value of approximately 0.1 is the best both for *L12* and *E5*.

rate	layer
1e-8	5.attention.output.dense.bias
	11.output.dense.bias
2e-8	5.attention.output.dense.bias
	11.output.dense.bias
3e-8	5.attention.output.dense.bias
	11.output.dense.bias
4e-8	5.attention.output.dense.bias
	11.output.dense.bias
5e-8	11.output.dense.weight
	11.output.dense.bias
6e-8	11.output.dense.weight
	11.output.dense.bias
7e-8	11.output.dense.weight
	11.output.dense.bias
8e-8	11.output.dense.weight
	11.output.dense.bias
9e-8	11.output.dense.weight
	11.output.dense.bias
1e-7	11.output.dense.weight
	11.output.dense.bias

Table 15: The ‘most changed’ two layers at each learning rate. The ‘change’ is defined as $(W_t - W_o)/(W_t + W_o)$, where W_t is the maximal weight of the layer in the tuned query encoder, and W_o is the maximal weight of the layer in the original (untuned) encoder. The prefix ‘encoder.layer’ is removed from the layer names here.

M.5 Stopping criterion

In Table 16 we show how the improvement depends on the stopping criterion. The stoppings after 5 or 10 non-improvement epochs give similar results. Stopping after 15 non-improvement epochs continues the trend of increased gain on English data, but with a deterioration on a few language pairs.

M.6 Execution time

There is no essential difference between the execution times for *E5* and *L12*. The tuning time depends on how soon stopping happened. At the settings of interest (Section 2.3, 3.1, 3.2), the tuning on an A100 GPU takes about one hour. For example, tuning 10 times at the default settings (Section 2.3, Appendix E) for rates between 1e-8 and 1e-7 takes 9 hours. At higher rates, stopping occurs earlier; tuning 10 times for rates between 1.1e-7 to 2e-7 takes less than 5 hours. Table 1 (with freezing different parts of the encoder) was obtained in 6 hours.

Evaluation of an encoder on all datasets we

idle epochs to stop	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
	c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
5	6.50	7.60	4.18	4.40	2.26	6.62	222/0	217/0	208/1	174/5
10	7.38	8.95	4.87	4.81	3.26	7.73	222/0	218/0	201/2	163/17
15	8.93	10.98	5.81	6.10	2.51	7.73	222/0	217/3	191/17	140/51

Table 16: Evaluations of the *E5* query encoder tuned with a frozen embedding block, learning rate $5e-8$, batch size 14 and triplet loss margin 0.1, stopped after different number of idle epochs (first column). The epoch is idle if no improvement is made. Values that did not pass the two-tailed test are shown in gray.

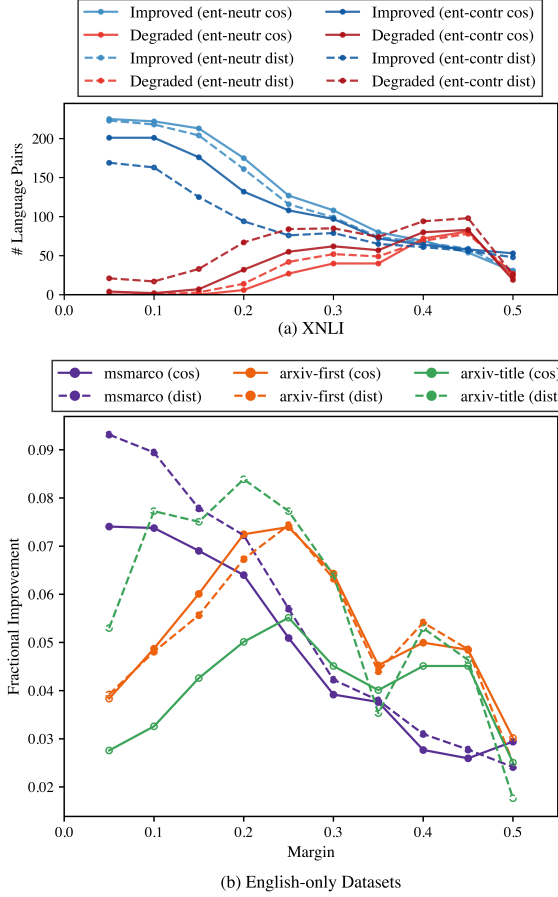


Figure 12: Evaluations of the *E5* query encoder tuned with a frozen embedding block, learning rate $5e-8$, batch size 14 using different triplet loss margins on (a) XNLI and (b) the English-only datasets (MSMARCO and ARXIV). Values that did not pass the two-tailed test are shown with open markers.

used (MSMARCO, ARXIV-first, ARXIV-title and XNLI) takes about 1.2-1.3 hours.

M.7 Effects of learning rate scheduler and weight decay

Using the fine-tuned *E5* model with the frozen embedding block, tuned using a batch size of 14, and a margin of 0.1, we randomly vary the batch size, learning rate scheduler and weight decay in

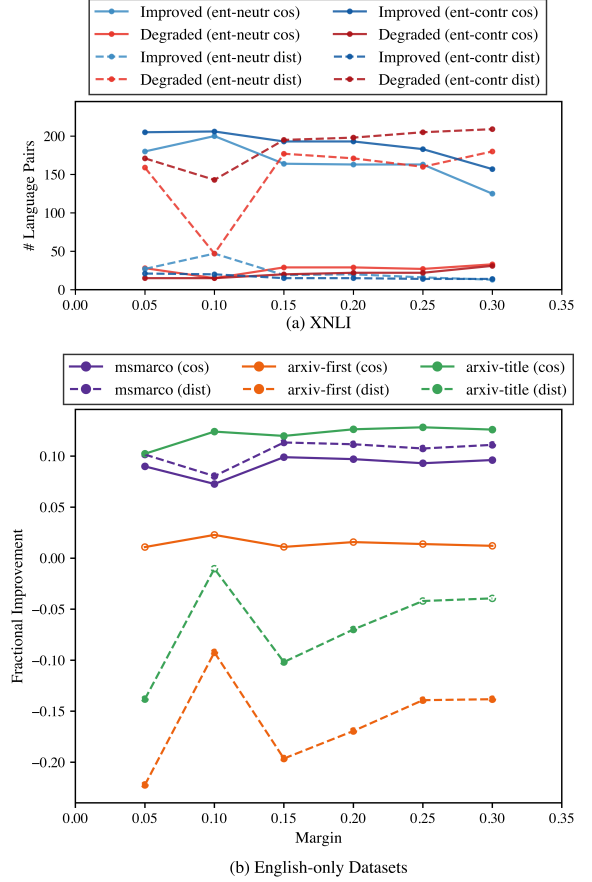


Figure 13: Evaluations of the *L12* query encoder tuned with a frozen embedding block, learning rate $5e-8$, batch size 14 using different triplet loss margins on (a) XNLI and (b) the English-only datasets (MSMARCO and ARXIV). Values that did not pass the two-tailed test are shown with open markers.

order to assess their impact on the model’s final performance. In Table 17 we present the change in performance across these different configurations for a learning rate of 5×10^{-8} , which is our ‘default’ learning rate (Section 2.3). In Table 18 we do the same for a learning rate of 10^{-7} . Table 19 lists the different schedulers we considered. Values in blue indicate the top improvements whereas values in red indicate the worse degradation.

B	Sch	D	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
			c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
100	Q	-	1.83	2.80	0.99	1.23	-0.26	0.96	200/1	172/3	69/71	43/110
	$E_{0.98}$	10^{-6}	-0.41	-1.07	-0.29	-0.83	-0.26	-0.96	0/63	0/39	0/20	1/8
64	-	-	1.58	2.28	0.78	1.07	0.78	1.20	178/1	156/3	72/50	48/87
	Q	-	0.05	0.19	-0.34	-0.59	-0.26	0.00	0/0	0/0	0/0	0/1
	$E_{0.98}$	10^{-6}	0.13	0.17	-0.23	-0.32	-0.26	0.00	0/0	0/0	0/0	0/1
	$E_{0.95}$	10^{-6}	-0.75	-1.47	-0.62	-0.78	-0.52	-0.48	0/67	0/48	0/66	1/42
	$E_{0.95}$	10^{-5}	-0.75	-1.47	-0.62	-0.78	-0.52	-0.48	0/67	0/48	0/66	1/42
	-	10^{-4}	1.58	2.28	0.78	1.07	0.78	1.20	178/1	156/3	72/50	48/87
	L	10^{-4}	0.30	0.55	-0.31	-0.11	-0.78	-0.48	0/0	0/0	0/4	0/9
	Q	10^{-4}	0.12	-0.17	-0.29	-0.43	0.26	0.00	0/0	0/0	0/0	0/0
32	L	10^{-4}	-0.05	-0.21	-0.18	-0.19	0.78	0.00	0/0	0/0	0/0	0/0
	$E_{0.98}$	10^{-4}	0.27	0.46	-0.21	-0.16	0.52	0.24	26/0	39/0	0/0	0/0
	L	-	0.00	-0.15	-0.10	-0.56	0.78	0.24	0/0	0/0	4/0	10/0
16	$E_{0.95}$	-	-0.70	-1.21	-0.65	-0.78	0.26	-0.96	0/53	0/32	7/6	20/0
	$E_{0.95}$	10^{-4}	-0.70	-1.21	-0.65	-0.78	0.26	-0.96	0/53	0/32	7/6	20/0
	Q	10^{-6}	-0.81	-1.39	-0.57	-1.15	-1.30	-2.39	0/167	0/123	0/121	2/81
8	$E_{0.95}$	10^{-6}	-2.98	-4.31	-2.34	-2.73	-2.60	-6.22	0/220	0/208	2/187	15/128
	-	10^{-5}	-0.81	-1.43	-0.78	-1.18	-1.30	-2.87	0/165	0/126	0/115	3/68
	Q	10^{-4}	-0.81	-1.39	-0.57	-1.15	-1.30	-2.39	0/167	0/123	0/121	2/81
	$E_{0.95}$	10^{-4}	-2.98	-4.31	-2.34	-2.73	-2.60	-6.22	0/220	0/208	2/187	15/128

Table 17: Percentage improvement over the fine-tuned E5 model with a frozen embedding block and tuned using a batch size of 14, learning rate $5e-8$ and a margin of 0.1. The blue colors indicate the top improvements whereas the red colors indicate the worse degradation. Three parameters are randomly varied: the batch size (denoted as “B”), the learning rate scheduler (denoted as “Sch”) and the weight decay (denoted as “D”). The learning rate schedulers are defined in Table 19 with an initial learning rate of $5e-8$. c% and d% refer to measuring the similarity of the text pairs using either the cosine similarity or the euclidean distance, respectively. For XNLI, (+) indicates the number of language pairs that were improved while (−) indicates those that have worsened out of a total of 225 language pairs. Note that only the statistically significant (determined by a Z-test) language pairs are retained and hence not all the improved/worsened counts sum to 225. Additionally, (ent-neutr) refers to entailment-entailment similarities compared with entailment-neutral similarities whereas (ent-contr) refers to comparisons against entailment-contradiction similarities.

Across these parameters, on average, the batch size appears to have the most significant impact, generally leading to poorer performance as the batch size is decreased. Within each batch size group, we see that using an exponential learning rate scheduler ($E_{0.95}$ or $E_{0.98}$) is generally worse than using any of the other schedulers or no scheduler at all. A specific exception exists when using a batch size of 100 where the exponential scheduler outperforms the quadratic one when the learning rate is set to 10^{-7} . Across all the configurations considered, the most impact seems to be the one shown in the first row of Table 17, where we see

good improvement over MSMARCO and ARXIV-first while simultaneously showing improvement over XNLI ent-neutr.

M.8 Varying the optimizer and learning rate

Table 20 shows the effects of choosing a different optimizer with a small and large learning rate. In addition to Adamax, we tried Adadelta and Stochastic Gradient Descent (SGD), both of which did not change the model weights in a significant enough way to affect the overall performance and hence, are not presented. For higher learning rates, SGD without momentum did elicit a change as shown

B	Sch	D	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
			c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
100	Q	-	-1.12	-1.17	-0.82	-0.14	-0.52	-1.71	0/138	3/117	15/65	61/40
	$E_{0.98}$	10^{-6}	-0.31	-0.18	-0.39	-0.05	-0.52	-1.22	0/0	0/0	0/11	0/7
64	-	-	-1.20	-1.05	-0.79	-0.11	-0.26	-0.73	0/57	5/34	1/80	9/47
	Q	-	-0.72	-0.89	-0.66	-0.14	-0.26	-1.71	0/65	4/43	1/62	21/33
	$E_{0.98}$	10^{-6}	-0.96	-1.13	-0.66	-0.49	-0.52	-1.22	0/80	6/56	3/65	29/35
	$E_{0.95}$	10^{-6}	-1.78	-2.36	-0.97	-1.35	0.78	-1.46	1/160	9/131	22/103	73/68
	$E_{0.95}$	10^{-5}	-1.78	-2.36	-0.97	-1.35	0.78	-1.46	1/160	9/131	22/103	73/68
	-	10^{-4}	-1.20	-1.05	-0.79	-0.11	-0.26	-0.73	0/57	5/34	1/80	9/47
	L	10^{-4}	-0.76	-0.80	-0.58	-0.16	-0.26	-1.46	0/61	3/32	0/58	14/32
	Q	10^{-4}	-2.12	-3.35	-1.47	-1.60	0.00	-2.68	2/190	8/162	41/107	93/69
32	L	10^{-4}	-2.05	-3.36	-1.45	-1.33	0.00	-2.68	2/189	8/158	41/102	94/68
	$E_{0.98}$	10^{-4}	-2.12	-3.54	-1.42	-1.84	0.00	-2.68	2/192	8/163	41/110	93/71
	L	-	-0.02	0.13	-0.58	-0.65	-1.04	-1.22	0/0	0/0	11/0	43/0
16	$E_{0.95}$	-	-2.52	-3.32	-1.32	-1.38	-0.26	-2.20	3/186	8/164	49/86	101/55
	$E_{0.95}$	10^{-4}	-2.52	-3.32	-1.32	-1.38	-0.26	-2.20	3/186	8/164	49/86	101/55
	Q	10^{-6}	-2.05	-3.29	-1.11	-1.38	-0.26	-3.66	0/212	3/182	21/140	70/105
8	$E_{0.95}$	10^{-6}	-2.44	-3.81	-1.55	-1.78	-0.52	-3.41	0/213	4/182	24/135	76/97
	-	10^{-5}	-2.03	-3.19	-0.74	-1.57	-0.26	-3.41	0/209	3/182	20/139	71/103
	Q	10^{-4}	-2.05	-3.29	-1.11	-1.38	-0.26	-3.66	0/212	3/182	21/140	70/105
	$E_{0.95}$	10^{-4}	-2.44	-3.81	-1.55	-1.78	-0.52	-3.41	0/213	4/182	24/135	76/97

Table 18: Percentage improvement over the fine-tuned E5 model with a frozen embedding block and tuned using a batch size of 14, learning rate 10^{-7} and a margin of 0.1. The blue colors indicate the top improvements whereas the red colors indicate the worse degradation. Three parameters are randomly varied: the batch size (denoted as “B”), the learning rate scheduler (denoted as “Sch”) and the weight decay (denoted as “D”). The learning rate schedulers are defined in Table 19 with an initial learning rate of 10^{-7} . c% and d% refer to measuring the similarity of the text pairs using either the cosine similarity or the euclidean distance, respectively. For XNLI, (+) indicates the number of language pairs that were improved while (−) indicates those that have worsened out of a total of 225 language pairs. Note that only the statistically significant (determined by a Z-test) language pairs are retained and hence not all the improved/worsened counts sum to 225. Additionally, (ent-neutr) refers to entailment-entailment similarities compared with entailment-neutral similarities whereas (ent-contr) refers to comparisons against entailment-contradiction similarities.

Scheduler	Definition
L	$\alpha(t) = \alpha_0 \left(1 - \frac{t}{T}\right)$
Q	$\alpha(t) = \alpha_0 \left(1 - \left(\frac{t}{T}\right)^2\right)$
$E_{0.95}$	$\alpha(t) = 0.95^t \alpha_0$
$E_{0.98}$	$\alpha(t) = 0.98^t \alpha_0$

Table 19: The definitions of the various learning rate schedulers used in Table 18 where t is the current training step, T , the total number of training steps and α_0 , the initial learning rate.

in Fig. 14, but the trend in performance is similar to what is presented in Fig. 2 with higher resolution near the transition point between improved and degraded multilingual performance. At around 9×10^{-7} , we see a sharp increase in the number of degraded language pairs while the model maintains constant improvement on MSMARCO. With a high enough learning rate, it seems that the gradients are able to overcome a barrier in the loss landscape that confined the weights to a region in which multilingual characteristics were preserved.

From the table, the default version of Adamax (Adamax with momentum) has a nearly negligible

O	M	LR	msmarco		arxiv-first		arxiv-title		xnli ent-neutr		xnli ent-contr	
			c%	d%	c%	d%	c%	d%	c+/-	d+/-	c+/-	d+/-
AdamW	Yes	2e-8	6.50	7.62	4.63	4.60	2.76	5.96	222/0	218/0	208/1	171/5
AdamW	Yes	1e-7	9.26	11.38	6.06	6.45	3.51	9.49	222/0	214/3	188/18	132/63
Adamax	Yes	2e-8	0.81	1.14	0.62	0.63	0.75	-0.22	0/0	0/0	1/0	1/0
	No	2e-8	6.40	7.50	4.25	4.63	3.01	6.40	224/0	220/0	214/1	178/4
	Yes	1e-7	6.67	7.65	4.67	4.68	3.51	6.40	222/0	217/0	205/2	171/7
	No	1e-7	6.46	7.60	4.67	5.06	3.51	6.40	222/0	217/0	198/2	157/12

Table 20: Percentage improvement over the untuned *E5* model. O, M and LR represent the choice of optimizer, whether or not momentum was used and the learning rate, respectively. All the models here are tuned with a batch size of 14, margin 0.1, and a frozen embedding block. Adamax with no momentum corresponds to choosing $\beta_1 = \beta_2 = 0$ for the optimizer parameters.

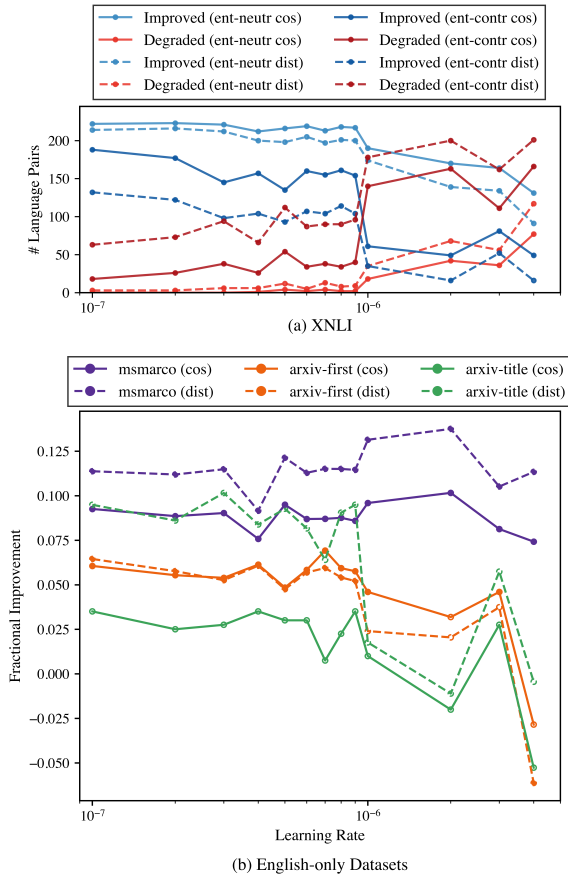


Figure 14: Evaluations on (a) XNLI and (b) the English-only datasets (MSMARCO and ARXIV) of the *E5* query encoder tuned with a frozen embedding block, batch size 14, margin 0.1 using different learning rates. Here we tune using SGD without momentum. Values that did not pass the two-tailed test are shown with open markers.

effect on the model when used with a small learning rate, suggesting that this particular configuration for the optimizer forces the model weights to

change very slowly. When momentum is switched off, the model weights change enough to improve the overall performance in both English and other languages. Continuing down to the bottom row, if we turn up the learning rate to a higher value, the model weights begin to change more significantly which brings about less improvement in the model’s multilingual capacity (still an improvement nonetheless), but maintains the same improvement on English. Overall, going from the first row to the last row (for Adamax), we transition from a point in model weight space where performance on all languages can be enhanced or preserved to a point which is better suited for the English-only task defined in tuning.