YOUR PROMPT SHOULD DO MORE: EFFECTS OF RETRIEVAL INSTRUCTIONS IN EMBEDDING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Prompted embedding models have recently received increasing attention, particularly for retrieval, where detailed retrieval instructions are provided as part of the retrieval prompt. Several new datasets and studies have examined this setting, showing that the current embedding models often struggle to follow such instructions reliably. In this paper, we study the mechanism of how instructions actually affect the representations of retrieval queries in asymmetric retrieval tasks. We show that models can fail to follow even simple task instructions when query-side distractors are included in the evaluation. We hypothesize that this behavior is driven by the training setup of current embedding models and their evaluation, and show that fine-tuning with added query-side distractors leads to substantial improvements, with minimal effect on other tasks.

1 INTRODUCTION

Text embeddings are widely used for semantic matching, where relevance of two texts is determined by their embedding similarity. Modern embedding models are often trained to support multiple tasks through natural language instructions that specify the intended relation between a query and its target. Typically, retrieval tasks are asymmetric, where only the query receives a task instruction while candidate documents are embedded independently. This allows a shared document index for multiple retrieval tasks. Yet, standard evaluation does not match this use. Popular benchmarks, like MTEB (Muennighoff et al., 2023), measure performance on multiple embedding task categories, which suggests that models should adapt query representations toward task-specific targets. However, the tasks are evaluated in isolation: both the queries and candidate sets are constructed for a single, specific task. Consequently, candidate pools rarely contain semantically related texts that do not satisfy the task instruction. We argue that this setting provides limited insight into how instructions actually transform query embeddings and alter retrieval behavior.

In this paper, we study how instructions affect retrieval behavior and embedding representations, uncovering several unintuitive results. For example, task instructions do not always increase the similarity between a question and its answer, and irrelevant task prompts are not consistently distinguishable from relevant ones in terms of performance, suggesting that instructions only weakly affect query representations. We hypothesize that this behavior may reduce robustness when the candidate pool includes examples semantically similar to the queries, which may be the case in practical large-scale retrieval applications. Using a controlled evaluation setting, we show that semantic similarity can dominate retrieval irrespective of the relation required by the task at hand, and that models fail to follow instructions adequately. We find evidence that this behavior may be related to common contrastive training practices that lack query-side negatives, and show that fine-tuning with such negatives substantially improves robustness.

2 RELATED WORK

Text embedding models Modern models rely on bi-encoder architectures trained on contrastive objectives (Reimers & Gurevych, 2019; Gao et al., 2021), a paradigm that continues to produce state-of-the-art embedding models to this day (Zhang et al., 2025). In-batch negatives are often supplemented with hard negatives mined from high-scoring corpus documents (Xiong et al., 2021), sometimes with similarity-margin filtering to avoid false negatives (de Souza P. Moreira et al., 2024;

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

Lee et al., 2025b). Recently, LLMs have also been used to generate synthetic hard negatives (Li et al., 2024). Another fundamental development is the introduction of *asymmetric natural language instructions*, where a task-specific instruction is prepended to the query but not the target (Wang et al., 2024b), which allows one model to adapt to multiple retrieval tasks. Following earlier work on instruction and prompt-based learning (Brown et al., 2020; Jiang et al., 2022; Shin et al., 2020), embedding models have increasingly incorporated task instructions and synthetic data to improve instruction following (Dai et al., 2023; Wang et al., 2024a).

Instructed Retrieval Several datasets have been introduced to study instruction-following IR models (Oh et al., 2024; Weller et al., 2025a;b; Zhou et al., 2025; Song et al., 2025; Weller et al., 2025c; Zhuang et al., 2025), showing that models often struggle to follow the instructions. Instructions in these datasets cover various aspects, ranging from preferred source types to expert healthcare criteria, but are often complex and query-specific, with a separate instruction for each query. Therefore, they are not designed to measure systematic task-level effects, making it difficult to analyze changes in the representations. In a similar vein, Asai et al. (2023) introduce an annotation scheme for targets which allows the instruction to define which documents are relevant.

Study of representations The effects that the inputs have on representations have been studied by Yamagiwa et al. (2023), who find interpretable axes in embeddings, Lee et al. (2025a), who study geometry in LM input and output embeddings, Chang et al. (2022), who study how language is encoded in embeddings, and Lin et al. (2024), who study the relation of input and resulting representation in jailbreak settings. Gonzalez-Gutierrez & Hovy (2025) recently show that unrelated instructions can perform well on embedding evaluation tasks, and that while instructions cause shifts in representations, the causes, mechanisms, and the structure of these changes remain unknown.

Our contribution Benchmark evaluations indicate which models perform the best, but offer little insight into *why*. Related work demonstrates that instructions have complex, yet little studied effects on embedding representations. We contribute by analyzing the effects of roughly 50 diverse instructions with structural measures that quantify the shifts which the instructions introduce to query embeddings. Additionally, we identify a core weakness in the standard fine-tuning and evaluation setups: the lack of *query-side negatives*. Current frameworks select negatives from the *corpus* side and evaluate tasks in isolation, which we hypothesize leads to models that react to instructions inadequately. We validate this hypothesis by introducing *distractors*, query-side hard negatives, both in fine-tuning and evaluation, showing that training with distractors leads to substantially stronger and more consistent instruction-following behavior.

3 METHODS

Our experiments consist of the following steps: first, we analyze roughly 50 natural language instructions, examining how task-relatedness and structural measures relate to their performance. We then identify differences between bitext-mining and retrieval tasks, and introduce distractors (query-side negatives) to show that instructions generally do not affect the query embeddings adequately. Next, we fine-tune models using distractors to improve instruction-following while preserving performance on other tasks, and finally, we analyze the structural effects fine-tuning had on the models.

3.1 DATASETS, MODELS, AND INSTRUCTIONS

Datasets The datasets in this study are Tatoeba (Tatoeba community, 2021) and ARCChallenge (Clark et al., 2018; Xiao et al., 2024), as well as SQuAD.v1 (Rajpurkar et al., 2016), which is used for the fine-tuning experiment. Tatoeba and ARCChallenge are in the MMTEB benchmark (Muennighoff et al., 2023; Enevoldsen et al., 2025): The Tatoeba dataset comprises translation pairs for a bitext-mining task (from English to target language), and ARCChallenge consists of scientific questions and their multiple-choice answers for a retrieval task. To mitigate the possibility of data contamination affecting our results, we omit popular retrieval datasets such as Natural Questions (Kwiatkowski et al., 2019) and FEVER (Thorne et al., 2018), on which the models we test have been reportedly fine-tuned. We use recall@1 to evaluate Tatoeba and ndcg@10 for ARCChallenge, matching the metric used by MTEB.

We select eight languages from Tatoeba, maintaining diversity in language families and scripts: Arabic, Mandarin Chinese, German, Finnish, French, Spanish, Vietnamese, and Turkish. The selection

contains several high-resource languages (Arabic, Mandarin Chinese, German, French, and Spanish) for which strong performance is expected and in which failure to complete the task would be particularly striking, alongside languages with fewer resources (Finnish, Turkish, and Vietnamese).

Distractors We additionally generate synthetic data for our experiments. For each query in all of the datasets, we create a *distractor* (a semantically similar text meant to distract evaluation) by prompting GPT-5.4-mini to paraphrase the query while preserving its meaning. These paraphrase queries serve as hard query-side negatives, testing whether models are capable of using task-specific instructions to shift the query embedding away from plain semantic similarity matching toward the actual task-specific target. Generation details and prompts are provided in Appendix A.

Models The 8 models used in this study are Octen/Octen-Embedding-8B (Octen Team, 2025), codefuse-ai/F2LLM-v2-4B (Zhang et al., 2026), Qwen/Qwen3-Embedding-4B and 0.6B (Zhang et al., 2025), microsoft/harrier-oss-v1-0.6b (Microsoft, 2026), BAAI/bge-m3 (Chen et al., 2024), intfloat/multilingual-e5-large-instruct (Wang et al., 2024b), and google/embeddinggemma-300m (Schechter Vera et al., 2025), with Appendix B providing further details. We aimed to select prominent embedding models with varying performance on the multilingual MMTEB benchmark (Enevoldsen et al., 2025), and not limit our selection exclusively to the highest-scoring models. We also opted for variability in model size (in parameters) and embedding size (number of produced embedding dimensions), and lastly, omitted models with more than 8B parameters due to computational constraints. We use the *sentence-transformers* library (Reimers & Gurevych, 2019).

Instructions The full list of instructions we use in our experiments, the motivation for their selection, and other details are listed in Appendix C with selected examples in Table 1. We use 47 instructions (51 for Tatoeba: 4 additional instructions that mention the target language) in our first experiments and then narrow the selection to task-related instructions in the rest of the experiments. We intentionally include instructions for unrelated tasks, such as clustering, to test how the task-relevance of instruction influences the evaluation and to distinguish the varying ways different types of instructions can affect the resulting embeddings.

Table 1: Examples of instructions in our experiments, with {lang} replaced with target language in Tatoeba evaluation. Instructions are intentionally varied.

Retrieve parallel sentences.	*Retrieve the corresponding translation in {lang}.*
Retrieve text based on user query.	*Retrieve semantically similar text.*
Group documents related to the same topic.	*asdfjkl qpwoeiru zxcvbnm*

3.2 Structural Statistics

We formulate the setting of retrieval and bitext mining and our metrics in the following way. Let D_q be a collection of queries and D_c the target documents (usually, *corpus*), and $qrel : D_q \to D_c$ be a mapping between these sets marking relevance, more concretely, mapping a query $Q \in D_q$ to its correct target, $A \in D_c$. In our case, $qrel$ is always injective, but not necessarily surjective, and the relevance is always binary, meaning a target document either is or is not relevant to a given query. Distractors, D, are semantically related text for each Q. We denote adding the instruction P to query Q with PQ. Equations are given in Appendix D.

Similarity Improvement We calculate similarity increase for an instruction P over all $Q \in D_q$ for normalized embedding vectors as the difference of cosine similarities of Q and A compared to PQ and A (Eq. 1). If the value is greater than 0, adding the instruction P increased the overall similarity to A; otherwise, it decreased the overall similarity. Similarity improvement is likely the most natural and intuitive way of viewing the effect of the instruction: a capable embedding model should react to a relevant instruction and move the query embedding closer to the correct target. However, naively moving towards the correct target may also mean moving towards incorrect targets.

Displacement We calculate the displacement caused by adding instruction P to the query as the cosine distance between embeddings of Q and PQ (Eq. 2). Values close to 0 indicate high similarity between Q and PQ, meaning low displacement. We measure displacement to evaluate the magnitude of the effect adding instructions has: if Q and PQ are close, adding the instruction has only a limited effect. On the other hand, too much movement can be detrimental. We also

calculate the displacement wrt. the targets A, and distractors D, separately, using the same formula, but substituting $emb(Q)$ with the embedding of the relevant target $emb(A)$, and $emb(PQ)$ with $emb(D)$, respectively.

Angulation Finally, we calculate angulation toward the correct answer to account for the drawback of displacement described above. We measure the cosine similarity between the vector from $emb(Q)$ to $emb(PQ)$, $\vec{v}_{Q,PQ}$, and the vector from $emb(Q)$ to $emb(A)$, $\vec{v}_{Q,A}$ (Eq. 3). Value 1 indicates that all movement was in the direction towards the target A, while 0 indicates orthogonal movement.

These three measures capture intuitive expectations on the effects the instructions have on the embeddings, and are intended to diagnose key characteristics related to our experiments.

4 Results

4.1 Instructions vs. Evaluation Performance

Table 2: Evaluation of relevant against other instructions. Star (*) indicates significance $p < 0.05$.

	ARCChallenge(ndcg@10)			Tatoeba(recall@1)		
Model	top-k	R^2	MV	top-k	R^2	MV
BAAI/bge-m3	0.286	0.059	0.457*	0.350	**0.065**	0.554*
codefuse-ai/F2LLM-v2-4B	**0.571**	**0.272**	**0.814***	0.538	0.014	0.750*
google/embeddinggemma-300m	0.143	0.000	-0.021	0.325	0.011	0.252*
intfloat/multilingual-e5-large-instruct	0.429	0.082	0.571*	0.575	0.052	0.787*
microsoft/harrier-oss-v1-0.6b	0.429	0.103	0.543*	0.588	0.056	0.81*
Octen/Octen-Embedding-8B	0.429	0.053	0.536*	**0.650**	0.024	0.779*
Qwen/Qwen3-Embedding-0.6B	0.429	0.092	0.600*	**0.650**	0.054	**0.821***
Qwen/Qwen3-Embedding-4B	0.143	0.069	0.436*	0.638	0.062	0.793*

We compare the evaluation scores of *relevant* instructions (defined as instructions related to retrieval or bitext-mining, respectively, separately in each scenario, see Appendix C) to those of other instructions with three methods: R^2 obtained for fitting a linear model on relevance-labels and scores, the effect-sizes of Mann-Whitney U-test (MW) (one-tailed, with the assumption that relevant instructions should yield higher scores), and by analyzing the fraction of relevant instructions in the top-k scoring instructions (with k defined as number of relevant instructions). For Tatoeba, which comprises 8 languages with differences in scores across one model, we use a linear mixed model, control for language, and report R^2 as the variance explained by the relevance label, making the value comparable to that of ARCChallenge. In the two other statistics, we normalize the evaluation scores before calculation to account for per-language variation.

The results are presented in Table 2. The fraction of relevant instructions in top-k scoring instructions indicates that non-relevant instructions are found in top results in both datasets and all models, and often make up the majority. For example, even one of the models receiving the highest scores on ARCChallenge, Octen/Octen-Embedding-8B, has a keysmash instruction (see Table 1) in the top results. Outside the clear outlier, codefuse-ai/F2LLM-v2-4B, the R^2-values are also consistently close to zero. On the other hand, the effect sizes of MW are large, which indicates that the distributions of relevant and other instructions are significantly different. Concretely, relevant instructions are always seen in the top of the results, but unrelated instructions regularly reach comparable performance. In conclusion, the relevance to task offers a distributional signal wrt. resulting performance in the standard embedding evaluation setup, but the dominant source of variation in scores is something other than intuitive and straightforward relatedness to given tasks, which aligns with related work.

4.2 Similarity Increase, Displacement and Angulation

If the presumed quality and task-relatedness of the instruction does not fully explain the evaluation results, the most natural next step is to investigate how the instructions affect the embeddings. We analyze displacement, similarity increase, and angulation as defined in Section 3.2.

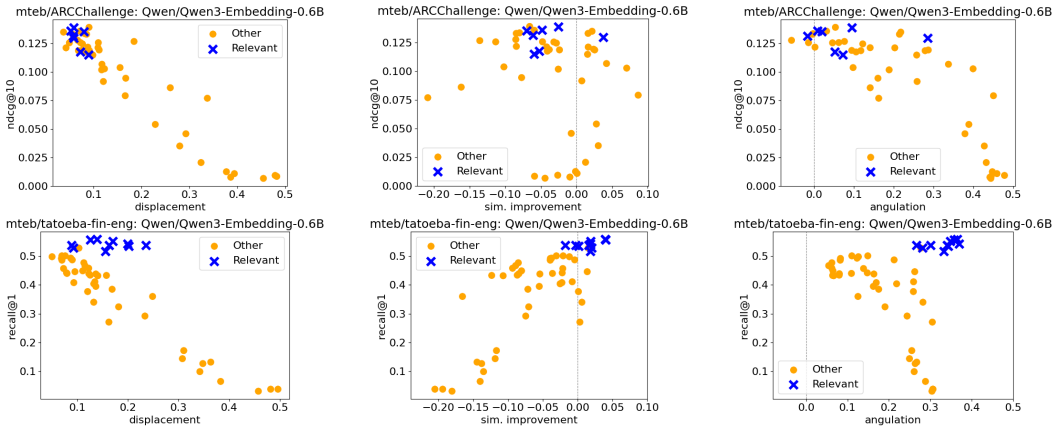


Figure 1: ARCChallenge (up) and Tatoeba:fin-eng (down), displacement, sim. improvement, and angulation. Each marker is a separate instruction. On Tatoeba, relevant instructions are highly distinguishable, unlike ARCChallenge, where especially displacement is low for relevant instructions.

Figure 1 shows the results for Qwen/Qwen-Embedding-0.6b on ARCChallenge and Tatoeba. Clearly, overall low displacement correlates with performance, and the relevant instructions are not distinguishable from the others at the top end of the scores on ARCChallenge. Similarity improvement does not correlate with performance for relevant instructions, with the majority of retrieval instructions even found on the negative side. Angulation shows that the majority of the instructions receiving high scores also do not move along the query-target line. The same cannot be said for Tatoeba: relevant instructions display almost full separation from others. A relationship between less displacement and higher score is present, but its strength is pronouncedly different between relevant and other instructions. Relevant instructions display more angulation while retaining high performance. Similarity improvement for relevant instructions sits at approximately 0, and is negative for other instructions. Across other models, we see varying levels of dependence and separation of relevant and other instructions. To quantify these relationships for all models, we use cross-validated (N=200, with 5 stratified folds) ROC-AUC of logistic regression, with evaluation scores and each structural metric as predictors, to measure separation between relevant and other instructions, and additionally Spearman's r to measure the monotonicity of the observed relationships.

The results in Table 3 match those of Figure 1: values are lower for ARCChallenge compared to Tatoeba, which means the relevant instructions have more distinct behavior under the structural metrics on Tatoeba than ARCChallenge. The Spearman correlations, which demonstrate the monotonicity of the score vs. structural metric relationship, are presented in Table 4, and they display the dominance of similarity improvement on Tatoeba and the negative relationship between displacement and evaluation score for both datasets. Additionally, we visualize one clear outlier, codefuse-ai/F2LLM-v2-4B, the only model showing values above 0.9 on ARCChallenge, in Appendix E.

4.3 Why does this happen?

The above results confirm that low displacement and small similarity gains show high results in standard evaluation. In Tatoeba, we saw a more desirable behavior: even if relevant instructions are not perfectly distinguishable by score, they have meaningfully different values in displacement, similarity improvement, and angulation compared to other instructions. Our hypothesis is that current embedding fine-tuning and evaluation schemes do not incentivize models to react to instructions: in contrastive learning, in-batch and *target-side* hard negatives may not sufficiently push instructed query embedding toward the answer region, especially when the evaluation corpus contains only plausible targets. In translation, mixing tasks during fine-tuning exposes the model to slightly more varied negatives, which may improve language sensitivity and explain the better results on Tatoeba.

To test our assumption, we introduce distractors to the retrieval pool to probe whether the instructions can survive the added text that is related semantically but not according to the instruction. For example, in ARCChallenge, the instruction to retrieve the answer should result in retrieving the

Table 3: Cross-validated ROC-AUC of logistic regression using the evaluation score and each structural metric (displacement (displ.), similarity improvement (sim.) and angulation (angl.)) as the predictors. High values indicate greater separability between relevant and other instructions, with 0.5 corresponding to chance level. Relevant instructions are more separable on Tatoeba.

	ARCChallenge (ndcg@10)			Tatoeba (recall@1)		
Model	displ.	sim.	angl.	displ.	sim.	angl.
BAAI/bge-m3	0.701	0.696	0.700	**0.838**	**0.827**	**0.870**
codefuse-ai/F2LLM-v2-4B	**0.916**	**0.910**	**0.918**	**0.890**	**0.976**	**0.950**
google/embeddinggemma-300m	0.520	0.184	0.448	0.631	0.655	0.681
intfloat/multilingual-e5-large-instruct	0.784	**0.800**	0.769	**0.877**	**0.999**	**1.000**
microsoft/harrier-oss-v1-0.6b	0.780	**0.836**	**0.877**	**0.896**	**0.877**	**0.948**
Octen/Octen-Embedding-8B	0.742	0.707	0.723	**0.942**	**0.883**	**0.999**
Qwen/Qwen3-Embedding-0.6B	**0.843**	0.746	0.776	**0.958**	**0.880**	**0.998**
Qwen/Qwen3-Embedding-4B	0.713	0.674	0.692	**0.881**	**0.965**	**0.986**

Table 4: Spearman correlations between evaluation score and displacement (displ.), similarity improvement (sim.) and angulation (angl.). Values summarize the monotonicity seen in the results: the relationship between low displacement and high score is consistent in both datasets, while similarity improvement and angulation show more variation. Star (*) indicates significance $p < 0.05$.

	ARCChallenge(ndcg@10)			Tatoeba(Recall@1)		
Model	displ.	sim.	angl.	displ.	sim.	angl.
BAAI/bge-m3	**-0.823***	**0.555***	**-0.763***	**-0.837***	**0.839***	**-0.510***
codefuse-ai/F2LLM-v2-4B	0.207	**0.884***	**0.522***	**-0.826***	**0.848***	0.168*
google/embeddinggemma-300m	**-0.772***	0.028	**-0.542***	**-0.658***	**0.516***	**-0.503***
intfloat/multilingual-e5-large-instruct	**-0.639***	-0.132	-0.369*	**-0.699***	**0.864***	0.118*
microsoft/harrier-oss-v1-0.6b	**-0.812***	0.049	**-0.514***	-0.323*	**0.54***	0.164*
Octen/Octen-Embedding-8B	**-0.587***	0.496*	-0.091	-0.321*	**0.695***	0.251*
Qwen/Qwen3-Embedding-0.6B	**-0.845***	-0.265	**-0.772***	**-0.609***	**0.792***	0.215*
Qwen/Qwen3-Embedding-4B	**-0.711***	-0.181	**-0.672***	**-0.593***	**0.800***	0.478*

answer, and not a related question. We evaluate with Recall@1 to pinpoint the effect of one distractor. Results of this evaluation are in Table 5, where we show the maximal performance over all instructions (and all languages in Tatoeba). They show that the addition of distractors completely destroys the performance for all instructions on ARCChallenge. Related retrieval instructions are also generally worse. On Tatoeba, two models almost perfectly retain the performance, and the best-scoring instructions are relevant. Figure 2 illustrates the effect of distractors with these two models and relevant instructions with respect to placement between distractor, query, instructed query, and target, summarizing aspects of displacement, similarity improvement and angulation into one figure: clearly, relatively more movement away from query and more towards the target help retain performance when distractors are added. From these results, we see that models that sufficiently react to instructions exist, but the majority fail to do so. In cases where distractors do not destroy the performance, the fraction of movement should be more on the target side. Next, we experiment with fine-tuning the model to explicitly have this characteristic and successfully respond to instructions.

5 Distractor Fine-tuning

To test whether distractor robustness can be improved by including such examples during model training, we fine-tune the Qwen/Qwen3-Embedding-0.6B model, for which all retrieval prompts were affected by added distractors. We fine-tune separate models for QA and bitext mining, with the latter fine-tuned multilingually on a mixture of all eight languages. Fine-tuning is performed using the SWIFT framework (Zhao et al., 2024). Details of the dataset splits and sizes, data preprocessing, and fine-tuning parameters are available in Appendix F.

We use SQuAD.v1 for QA fine-tuning, since ARCChallenge contains only a test set, and obtain Tatoeba training examples from the OPUS dataset (Tiedemann, 2012). We use the same training

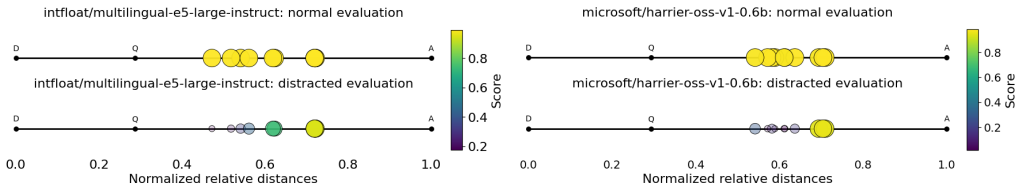


Figure 2: Taoteba:deu for multilingual-e5-large-instruct (left) and harrier-oss-v1-0.6b (right): Evaluation without (up) and with distractors (down). Each circle corresponds to a relevant instruction-query, and its position indicates its relative normalized displacement between Q=query, D=distractor, and A=target. The more movement towards the answer, the better the score is retained.

Table 5: Distractor results for ARCChallenge and Tatoeba, both with maximal recall@1 over all instructions (and all languages in Tatoeba): normal column refers to normal evaluation setup with no distractors, distr. column to setup where distractors are introduced to the target corpus, and rel. showing the best performance of relevant instructions, if lower than the distr. column.

	ARCChallenge			Tatoeba (max)		
Model	normal	distr.	rel.	normal	distr.	rel.
BAAI/bge-m3	0.032	0.001	0.000	0.987	0.471	–
codefuse-ai/F2LLM-v2-4B	0.214	0.062	–	0.998	0.626	0.55
google/embeddinggemma-300m	0.019	0.005	0.002	0.725	0.25	–
intfloat/multilingual-e5-large-instruct	0.074	0.007	0.0	**0.994**	**0.954**	–
microsoft/harrier-oss-v1-0.6b	0.106	0.007	–	**0.988**	**0.956**	–
Octen/Octen-Embedding-8B	0.149	0.005	0.001	0.992	0.671	–
Qwen/Qwen3-Embedding-0.6B	0.059	0.007	0.0	0.977	0.500	–
Qwen/Qwen3-Embedding-4B	0.112	0.038	–	0.994	0.529	–

setup for both tasks. Each example consists of a query (Q), its retrieval target (A), and a distractor (D), which is a generated paraphrase of Q. For QA, Q is a question and A its answer paragraph, while for bitext mining, Q is an English sentence and A its target-language translation. To avoid catastrophic forgetting, we fine-tune jointly on retrieval (QA, bitext mining) and semantic textual similarity (STS), constructing $P_{task}Q \rightarrow A$ for the retrieval tasks and $P_{sts}Q \rightarrow D$ for STS, where P_{task} is P_{qa} or P_{bitext}. Thus, D serves as a positive target for STS but a query-side negative for the retrieval tasks. Each positive target is accompanied by eight random target-side negatives. For 50% of the training examples, one negative is replaced by the wrong-task target: query-side distractor D for the retrieval tasks and A for STS. QA and bitext models are fine-tuned separately, but for bitext mining, examples from all eight languages are pooled with the language specified in instructions.

5.1 Evaluation and Results

We evaluate the fine-tuned models under three settings: *Normal*, *Random distractors*, and *Paraphrase distractors*. *Normal* measures standard retrieval performance without added distractors. *Random* adds random query-side examples as distractors (other questions for QA, and other English sentences for bitext mining), allowing us to analyze both the effect of increased candidate set and whether arbitrary query-side distractors degrade retrieval performance. *Paraphrase* adds query-side distractors that are semantically highly similar, but do not fulfill the task instructions, testing the model's ability to follow instructions instead of plain semantic similarity. In both settings, one distractor is added per query, increasing the candidate pool size by the total number of queries.

Figures 3 and 4 show SQuAD and Tatoeba development performance over the fine-tuning steps. Fine-tuning substantially improves performance in the distractor setting, while performance in the normal setting changes only slightly. Most of the improvement occurs within the first 5,000 steps (approx. 30% of one epoch, 140K/150K training examples in total), after which development performance plateaus. The final test-set results are reported in Table 6 for the baseline Qwen/Qwen3-Embedding-0.6B model and its fine-tuned variants. A similar trend is observed for both SQuAD and Tatoeba. For the baseline model, *paraphrase distractors* almost completely destroy Recall@1

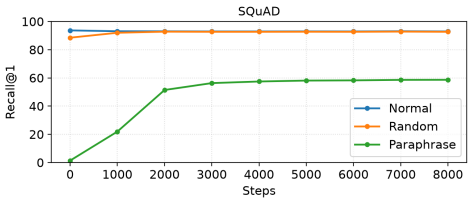


Figure 3: SQuAD development performance across fine-tuning steps.

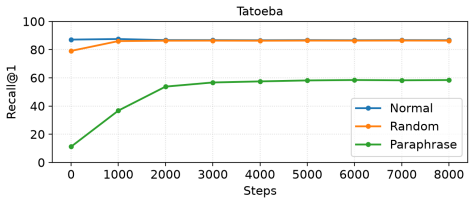


Figure 4: Tatoeba development performance across fine-tuning steps.

Table 6: Recall@1 and Recall@5 test results on SQuAD and Tatoeba for the baseline Qwen/Qwen3-Embedding-0.6B compared to fine-tuned models. Tatoeba results are reported as a mean across 8 languages (per language results in Appendix G).

		SQuAD			Tatoeba (mean)		
Metric	Model	normal	random	paraphr.	normal	random	paraphr.
Recall@1	baseline	0.724	0.685	0.014	0.844	0.769	0.102
	fine-tuned	0.706	0.702	0.470	0.837	0.833	0.569
Recall@5	baseline	0.918	0.892	0.778	0.926	0.878	0.832
	fine-tuned	0.912	0.910	0.890	0.923	0.921	0.908

performance, despite being instructed to retrieve the target, the model typically retrieves the query-side distractor instead. By contrast, *random distractors* cause only a moderate drop in performance, indicating that simply increasing the candidate set or adding unrelated queries does not substantially mislead the model. A similar pattern is visible for Recall@5, where adding a single semantically similar paraphrase distractor per query does not always push the correct target out of the top five. This highlights the need for semantically similar query-side distractors in evaluation.

The fine-tuned models show different behavior when distractors are added. Their performance is nearly unaffected by random distractors, and although paraphrase distractors still reduce performance, the degradation is substantially smaller than for the baseline. This suggests that the model can learn to follow task-specific instructions from a relatively small amount of fine-tuning data. Importantly, this improvement does not substantially degrade standard retrieval. The *Normal* results in Table 6 remain close to the baseline. In Appendix H, we show that the improved robustness to paraphrase distractors is not due to task-specific fine-tuning alone, but specifically to the addition of query-side negatives. When the models are fine-tuned on the same data (SQuAD, Tatoeba) using the same setup but excluding query-side negatives, performance under paraphrase distractors remains flat, and the gap between the *Normal* and *Random* evaluation settings remains.

5.2 After Fine-tuning: What happened?

Fine-tuning greatly increased the performance with distractors. Next, we shortly analyze the effects the fine-tuning had on the structural metrics introduced in Section 3.2 and out-of-domain tasks. In the SQuAD-finetuned model, similarity improvement increases: on average, after finetuning, all

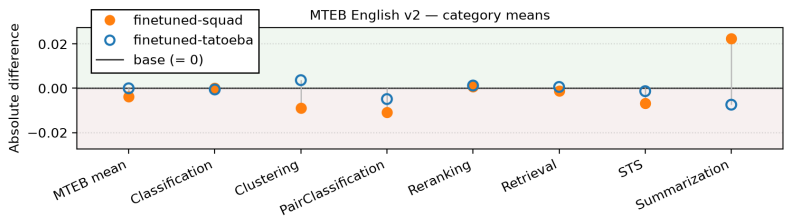


Figure 5: Performance on the MMTEB benchmark categories for fine-tuned models. The absolute differences are small, meaning our fine-tuning setup does not affect the overall performance.

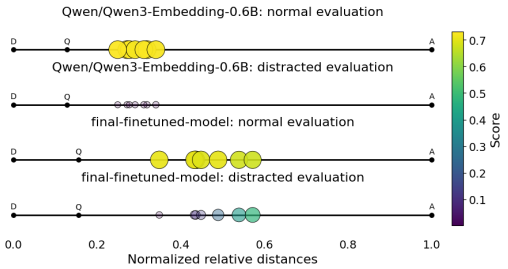


Figure 6: SQuAD: before and after for relevant instructions, with and without distractors.

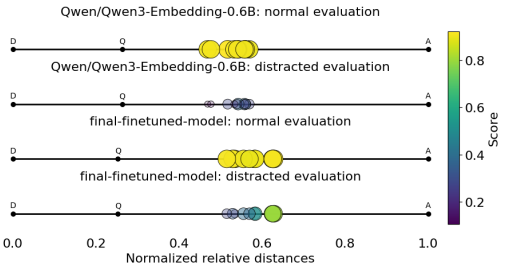


Figure 7: Tatoeba:fra: before and after for relevant instructions, with and without distractors.

relevant instructions increase similarity to the answer more than before finetuning, not only the finetuning instruction. The effect is opposite for Tatoeba: the similarity improvement decreases slightly in all languages, but less for the relevant instructions. Displacement grows in both finetuned models. Both models also increase in angulation, and especially on Tatoeba, most increase is seen in the prompt where the language is explicitly mentioned, which is a desirable outcome of the fine-tuning. Figures 6 and 7 illustrate the changes with relative distance plots. Fine-tuning successfully yielded characteristics identified as beneficial in Section 4: the instructions' effect is larger, and the performance is retained better when distractors are added to the retrieval pool.

Out-of-domain performance We also ran the full MTEB English v2 evaluation (Enevoldsen et al., 2025) on the finetuned models, the results are shown in Figure 5, averaged per task category, and in Appendix I in full. They show that while the fine-tuning for distractors has effects on the results on other benchmarks, the differences are small and distractor fine-tuning does not destroy the performance on other tasks. We also evaluated the SQuAD-fine-tuned model on ARCChallenge, see Appendix E Table 11. While the baseline performance is already low, fine-tuning manages to make the paraphrase distracted evaluation slightly improve for Recall@1 and Recall@5.

6 Discussion and Conclusion

In this paper, we hypothesized that current training and evaluation setups may overestimate the robustness of instructed text embedding models, and may not fully capture challenges encountered in real-world retrieval settings, particularly if candidates contain semantically similar query-side distractors. We investigated this claim from multiple angles: how instruction relevance is tied to performance, how simple structural measures show the effect of instructions, how introducing distractors negatively affects the models, and how fine-tuning with carefully selected negatives can mitigate this effect substantially. Our findings align with prior work showing that instructions have complex and poorly understood effects on embedding models, while extending this line of research toward geometric interpretations of representational change. Beyond analysis, we introduce both a probing framework for assessing inadequate instruction-following and a fine-tuning strategy to address it. Taken together, our results suggest that the standard evaluation paradigm may overestimate genuine instruction sensitivity. We therefore argue for a shift toward evaluation settings that pool across tasks, exposing models to instruction-sensitive, multi-task retrieval challenges.

The main limitation of our study is the number of tasks investigated. We focus on retrieval and bitext-mining, and leave out other common embedding tasks, such as clustering. Similarly, while we study 8 prominent text embedding models, we cannot guarantee that the results generalize beyond this set. However, the value of the introduced methodology does not depend on universality, and the ability to identify and correct inadequate instruction-following behavior and the fine-tuning framework remains applicable to any model exhibiting similar characteristics. To reduce sensitivity to instruction selection, we deliberately use a large and varying instruction set with varying quality. The synthetic distractor data is generated using a single model and we do not ablate over different model choices. However, because paraphrasing of short queries imposes relatively constrained generation, we expect the choice of generation model to have only a limited effect on the results. In the future, we will extend the analysis to cover more task categories and query–target relations.

AI USE STATEMENT

In this work, we used generative AI tools for generating synthetic data, language editing, as one of the tools for literature survey, and code auditing and generation. We have not used generative AI tools to help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean and reformat datasets, support qualitative and thematic data analysis, or interpret results. Formulating mathematical claims, providing critical ingredients for proving mathematical claims, and assisting in the writing of proofs are not applicable to this work. We have reviewed all AI-assisted work: fixes suggested in code auditing and generated code were reviewed for logical consistency, and language editing is used to polish human-written text, not generate new text. Synthetic data generated using generative AI includes synthetic query paraphrases, and the use of synthetic data is clearly articulated in the paper with generation prompts provided in the Appendix. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We will release the code to replicate our results upon publication of this work. We will also release the synthetic datasets generated as part of this work. Otherwise, our experiment use open models and freely available datasets. Appendices detail the instructions, computational cost, hardware, synthetic dataset generation, and fine-tuning experiment details, such as the query-side hard negative selection. Computational details and resource usage are provided in Appendix J.

REFERENCES

Akari Asai, Timo Schick, Patrick Lewis, Xilun Chen, Gautier Izacard, Sebastian Riedel, Hannaneh Hajishirzi, and Wen-tau Yih. Task-aware retrieval with instructions. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 3650–3675, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.225. URL https://aclanthology.org/2023.findings-acl.225/.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Tyler A. Chang, Zhuowen Tu, and Benjamin K. Bergen. The geometry of multilingual language model representations. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 119–136, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.9. URL https://aclanthology.org/2022.emnlp-main.9/.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation, 2024.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018. URL https://arxiv.org/abs/1803.05457.

Zhuyun Dai, Vincent Y Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith Hall, and Ming-Wei Chang. Promptagator: Few-shot dense retrieval from 8 examples. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=gmL46YMpu2J.

Gabriel de Souza P. Moreira, Radek Osmulski, Mengyao Xu, Ronay Ak, Benedikt Schifferer, and Even Oldridge. NV-retriever: Improving text embedding models with effective hard-negative mining, 2024. URL https://arxiv.org/abs/2407.15831.

Kenneth Enevoldsen, Isaac Chung, Imene Kerboua, Márton Kardos, Ashwin Mathur, David Stap, Jay Gala, Wissam Siblini, Dominik Krzemiński, Genta Indra Winata, Saba Sturua, Saiteja Utpala, Mathieu Ciancone, Marion Schaeffer, Diganta Misra, Shreeya Dhakal, Jonathan Rystrøm, Roman Solomatin, Ömer Veysel Çağatan, Akash Kundu, Martin Bernstorff, Shitao Xiao, Akshita Sukhlecha, Bhavish Pahwa, Rafał Poświata, Kranthi Kiran GV, Shawon Ashraf, Daniel Auras, Björn Plüster, Jan Philipp Harries, Loïc Magne, Isabelle Mohr, Dawei Zhu, Hippolyte Gisserot-Boukhlef, Tom Aarsen, Jan Kostkan, Konrad Wojtasik, Taemin Lee, Marek Suppa, Crystina Zhang, Roberta Rocca, Mohammed Hamdy, Andrianos Michail, John Yang, Manuel Faysse, Aleksei Vatolin, Nandan Thakur, Manan Dey, Dipam Vasani, Pranjal A Chitale, Simone Tedeschi, Nguyen Tai, Artem Snegirev, Mariya Hendriksen, Michael Günther, Mengzhou Xia, Weijia Shi, Xing Han Lù, Jordan Clive, Gayatri K, Maksimova Anna, Silvan Wehrli, Maria Tikhonova, Henil Shalin Panchal, Aleksandr Abramov, Malte Ostendorff, Zheng Liu, Simon Clematide, Lester James Validad Miranda, Alena Fenogenova, Guangyu Song, Ruqiya Bin Safi, Wen-Ding Li, Alessia Borghini, Federico Cassano, Lasse Hansen, Sara Hooker, Chenghao Xiao, Vaibhav Adlakha, Orion Weller, Siva Reddy, and Niklas Muennighoff. MMTEB: Massive multilingual text embedding benchmark. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=zl3pfz4VCV.

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

Cesar Gonzalez-Gutierrez and Dirk Hovy. Do prompts reshape representations? an empirical study of prompting effects on embeddings, 2025. URL https://arxiv.org/abs/2510.19694.

Ting Jiang, Jian Jiao, Shaohan Huang, Zihan Zhang, Deqing Wang, Fuzhen Zhuang, Furu Wei, Haizhen Huang, Denvy Deng, and Qi Zhang. PromptBERT: Improving BERT sentence embeddings with prompts. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 8826–8837, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.603. URL https://aclanthology.org/2022.emnlp-main.603/.

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

Andrew Lee, Melanie Weber, Fernanda Viégas, and Martin Wattenberg. Shared global and local geometry of language model embeddings. In *Second Conference on Language Modeling*, 2025a. URL https://openreview.net/forum?id=aJDykpJAYF.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. NV-embed: Improved techniques for training LLMs as generalist embedding models. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=lgsyLSsDRe.

Xiaopeng Li, Xiangyang Li, Hao Zhang, Zhaocheng Du, Pengyue Jia, Yichao Wang, Xiangyu Zhao, Huifeng Guo, and Ruiming Tang. Syneg: Llm-driven synthetic hard-negatives for dense retrieval, 2024. URL https://arxiv.org/abs/2412.17250.

Yuping Lin, Pengfei He, Han Xu, Yue Xing, Makoto Yamada, Hui Liu, and Jiliang Tang. Towards understanding jailbreak attacks in LLMs: A representation space analysis. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 7067–7085, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.401. URL https://aclanthology.org/2024.emnlp-main.401/.

Microsoft. Harrier-oss-v1: Open-source multilingual text embeddings, 2026. URL https://labs.ai.azure.com/innovations/harrier-oss-v1/.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. MTEB: Massive text embedding benchmark. In Andreas Vlachos and Isabelle Augenstein (eds.), *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 2014–2037, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.eacl-main.148. URL https://aclanthology.org/2023.eacl-main.148/.

Octen Team. Octen series: Optimizing embedding models to #1 on rteb leaderboard, 2025. URL https://octen-team.github.io/octen_blog/posts/octen-rteb-first-place/.

Hanseok Oh, Hyunji Lee, Seonghyeon Ye, Haebin Shin, Hansol Jang, Changwook Jun, and Minjoon Seo. Instructir: A benchmark for instruction following of information retrieval models, 2024. URL https://arxiv.org/abs/2402.14334.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In Jian Su, Kevin Duh, and Xavier Carreras (eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL https://aclanthology.org/D16-1264.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. URL http://arxiv.org/abs/1908.10084.

Henrique Schechter Vera, Sahil Dua, and EmbeddingGemma Team. Embeddinggemma: Powerful and lightweight text representations, 2025. URL https://arxiv.org/abs/2509.20354.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4222–4235, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.346. URL https://aclanthology.org/2020.emnlp-main.346/.

Tingyu Song, Guo Gan, Mingsheng Shang, and Yilun Zhao. IFIR: A comprehensive benchmark for evaluating instruction-following in expert-domain information retrieval. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 10186–10204, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.511. URL https://aclanthology.org/2025.naacl-long.511/.

Tatoeba community. Tatoeba: Collection of sentences and translations, 2021.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a large-scale dataset for fact extraction and VERification. In Marilyn Walker, Heng Ji, and Amanda Stent (eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association*

for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pp. 809–819, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1074. URL https://aclanthology.org/N18-1074.

Jörg Tiedemann. Parallel data, tools and interfaces in OPUS. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis (eds.), *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pp. 2214–2218, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA). URL https://aclanthology.org/L12-1246/.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Improving text embeddings with large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11897–11916, Bangkok, Thailand, August 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.642. URL https://aclanthology.org/2024.acl-long.642/.

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

Orion Weller, Benjamin Chang, Sean MacAvaney, Kyle Lo, Arman Cohan, Benjamin Van Durme, Dawn Lawrie, and Luca Soldaini. FollowIR: Evaluating and teaching information retrieval models to follow instructions. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 11926–11942, Albuquerque, New Mexico, April 2025a. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.597. URL https://aclanthology.org/2025.naacl-long.597/.

Orion Weller, Benjamin Chang, Eugene Yang, Mahsa Yarmohammadi, Samuel Barham, Sean MacAvaney, Arman Cohan, Luca Soldaini, Benjamin Van Durme, and Dawn Lawrie. Followir: A multilingual benchmark for instruction following in retrieval. In *Advances in Information Retrieval: 47th European Conference on Information Retrieval, ECIR 2025, Lucca, Italy, April 6–10, 2025, Proceedings, Part II*, pp. 295–310, Berlin, Heidelberg, 2025b. Springer-Verlag. ISBN 978-3-031-88710-9. doi: 10.1007/978-3-031-88711-6_19. URL https://doi.org/10.1007/978-3-031-88711-6_19.

Orion Weller, Benjamin Van Durme, Dawn Lawrie, Ashwin Paranjape, Yuhao Zhang, and Jack Hessel. Promptriever: Instruction-trained retrievers can be prompted like language models. In *The Thirteenth International Conference on Learning Representations*, 2025c. URL https://openreview.net/forum?id=odvSjn416y.

Chenghao Xiao, G Thomas Hudson, and Noura Al Moubayed. Rar-b: Reasoning as retrieval benchmark. *arXiv preprint arXiv:2404.06347*, 2024.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=zeFrfgyZln.

Hiroaki Yamagiwa, Momose Oyama, and Hidetoshi Shimodaira. Discovering universal geometry in embeddings with ICA. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4647–4675, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.283. URL https://aclanthology.org/2023.emnlp-main.283/.

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. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025.

Ziyin Zhang, Zihan Liao, Hang Yu, Peng Di, and Rui Wang. F2llm-v2: Inclusive, performant, and efficient embeddings for a multilingual world, 2026. URL https://arxiv.org/abs/2603.19223.

Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. Swift:a scalable lightweight infrastructure for fine-tuning, 2024. URL https://arxiv.org/abs/2408.05517.

Jianqun Zhou, Yuanlei Zheng, Wei Chen, Qianqian Zheng, Shang Zeyuan, Wei Zhang, Rui Meng, and Xiaoyu Shen. Beyond content relevance: Evaluating instruction following in retrieval models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=OlRjxSuSwl.

Yuchen Zhuang, Aaron Trinh, Rushi Qiang, Haotian Sun, Chao Zhang, Hanjun Dai, and Bo Dai. Towards better instruction following retrieval models, 2025. URL https://arxiv.org/abs/2505.21439.

A Generating Query-Side Distractors

For each query used in our experiments, we generated exactly one paraphrase to serve as query-side distractors. This totals to 185,254 generated paraphrases. We used OpenAI's GPT-5.4-mini (version 2026-03-17) through their Batch API. The total cost of the paraphrase generation is $10.22, the majority accounting for the fine-tuning data.

Generation prompts are provided in Table 7, with {input} replaced with the actual text.

B Embedding Models

Table 8 lists the number of parameters and dimensionality of the output embedding for each model. The licences for the models are

MIT: BAAI/bge-m3 , intfloat/multiligual-e5-large-instruct, microsoft/harrier-oss-v1-0.6b

Apache-2.0: Qwen/Qwen3-Embedding-4B, Qwen/Qwen3-Embedding-0.6B, Octen/Octen-Embedding-8B, codefuse-ai/F2LLM-v2-4B

Gemma Licence: google/embeddinggemma-300m.

Table 7: Prompts used to generate synthetic distractors.

Dataset	Prompt
SQuAD	Paraphrase the following question while preserving its meaning exactly. Use different wording and avoid producing a near-identical phrasing. Output only the paraphrased question and nothing else. Question: {input}
Tatoeba	Paraphrase the following text while preserving its meaning exactly. Use different wording and avoid producing a near-identical phrasing. Output only the paraphrased text and nothing else. Text: {input}
ARCChallenge	Paraphrase the following text while preserving its meaning exactly. Use different wording and avoid producing a near-identical phrasing. Output only the paraphrased text and nothing else. Text: {input}

Table 8: Models used in this study.

Model	Parameters	Dimensions	Citation
Octen/Octen-Embedding-8B	7.6B	4096	Octen Team (2025)
codefuse-ai/F2LLM-v2-4B	4B	2560	Zhang et al. (2026)
Qwen/Qwen3-Embedding-4B	4B	2560	Zhang et al. (2025)
Qwen/Qwen3-Embedding-0.6B	596M	1024	Zhang et al. (2025)
microsoft/harrier-oss-v1-0.6b	596M	1024	Microsoft (2026)
BAAI/bge-m3	568M	1024	Chen et al. (2024)
intfloat/multilingual-e5-large-instruct	560M	1024	Wang et al. (2024b)
google/embeddinggemma-300m	308M	768	Schechter Vera et al. (2025)

C INSTRUCTIONS

We list all 47 (51 for bitext-mining) instructions in Table 9. We use the instruction template *Instruct: <Instruction> \nQuery: <Query>*. Additionally, we also calculate the query embeddings without instructions (or the template), making the total number of dataset embedding iterations 48/52. The instruction selection was started by searching for task-specific instructions, which are given for certain datasets, such as ARCChallenge. Next, we included the MTEB "abstask" instructions, instructions defined per task category in the MTEB library. We then augmented this selection with 1. task-specific instructions for unrelated tasks, 2. Vague instructions, written by authors, and 3. two to five LLM-generated instructions per category (Claude-Sonnet-4.6, with prompt "Generate 10 instructions of varying quality for MTEB-style {task category name} embedding evaluation"), out of which we selected 2-5 with the least overlap with the instructions selected in the previous steps to diversify the selection. For bitext-mining, 4 instructions are included with and without the language specified in the instruction; for retrieval, only the instructions without language specification are included (concretely: "in {lang}" omitted). This makes bitext-mining have 4 additional instructions.

D FORMULAS FOR STRUCTURAL METRICS

The formulas we use to calculate similarity improvement, displacement, and angulation are presented in Equations 1–3. The embedding of a text is represented as $emb(\cdot)$, cosine similarity as $\cos(\cdot,\cdot)$, A represents the relevant target for each query Q in the collection of queries D_q, and PQ represents the combination of instruction P and query Q. The vector from $emb(Q)$ to $emb(PQ)$ is $\vec{v}_{Q,PQ}$, and the vector from $emb(Q)$ to $emb(A)$ is $\vec{v}_{Q,A}$.

$$\text{Similarity Impr.}(P) = \frac{1}{|D_q|}\sum_{Q\in D_q} \cos(emb(PQ), emb(A)) - \cos(emb(Q), emb(A)), \quad (1)$$

$$\text{Displacement}(P) = \frac{1}{|D_q|}\sum_{Q\in D_q} 1 - \cos(emb(PQ), emb(Q)). \quad (2)$$

$$\text{Angulation}(P) = \frac{1}{|D_q|}\sum_{Q\in D_q} \cos(\vec{v}_{Q,PQ}, \vec{v}_{Q,A}). \quad (3)$$

E SUPPORTING RESULTS

Figure 8 presents the results for the outlier model, codefuse-ai/F2LLM–v2-4B, on ndgc@10. Especially for similarity improvement and angulation, high values indicate high performance on relevant instructions. The relevant instruction with the highest displacement, similarity improvement, and angulation is by far the best performing. Despite this, when evaluating with the distractors and recall@1, Figure 9 shows that even the best instruction cannot deal with distractor evaluation fully.

In Table 10 we present the instructions that most improved their performance in distractor finetuning. Naturally, the finetuning instructions are consistently the instructions with the most improvement,

Instruction	Source	Motivation	Relevant
”Identify categories in user passages.”	MTEB, clustering	MTEB abstask	
”Classify user passages.”	MTEB, classification	MTEB abstask	
”Retrieve text that are semantically similar to the given text.”	MTEB, pair classification	MTEB abstask	
”Retrieve text based on user query.”	MTEB, retrieval	MTEB abstask	Retrieval
”Retrieve semantically similar text.”	MTEB, STS	MTEB abstask	
”Given a news summary, retrieve other semantically similar summaries.”	MTEB, summarization	MTEB abstask	
”Retrieve parallel sentences.”	MTEB, bitext-mining	MTEB abstask	Bitext-mining
”Retrieve the answer to the question.”,	MTEB, ARCChallenge	Task-specific	Retrieval
”Given a question, retrieve the passage that best answers it.”,	Generated+Authors	Retrieval, diversity	Retrieval
”Retrieve”,	Generated+Authors	Retrieval, diversity	Retrieval
”Find the most relevant passage that directly answers the question.”,	Generated+Authors	Retrieval, diversity	Retrieval
”Given a question, find a related document.”	Generated+Authors	Retrieval, diversity	Retrieval
f”Retrieve the corresponding translation in {lang}.”,	Generated+Authors	Bitext, diveristy	Bitext-mining x2
f”Given an English sentence, find its translation in {lang}.”,	Generated+Authors	Bitext, diveristy	Bitext-mining x2
f”Retrieve parallel sentences in {lang}”,	Generated+Authors	Bitext, diveristy	Bitext-mining x2
f”Translate to {lang}.”,	Generated+Authors	Bitext, diveristy	Bitext-mining x2
”Find a sentence that has similar meaning.”,	Generated+Authors	Bitext, diveristy	Bitext-mining
”Summarize the given paragraph into a short paragraph.”,	Generated+Authors	Summ., diversity	
”Match the text to the short abstract.”,	Generated+Authors	Summ., diversity	
”Retrieve a related summary.”	Generated+Authors	Summ., diversity	
”Classify texts as entailment or contradiction.”	Generated+Authors	Pair clf, diverisity	
”Determine whether the following two passages are contradictory, entailed, or neutral.”	Generated+Authors	Pair clf, diverisity	
”Classify based on semantic similarity.”	Generated+Authors	Pair clf, diverisity	
”Classify the following text into a meaningful cluster based on its content.”,	Generated+Authors	Classif., diversity	
”Group documents related to the same topic.”,	Generated+Authors	Clustering, diversity	
”Cluster.”	Generated+Authors	Clustering, diversity	
”Retrieve a similar passage.”,	Generated+Authors	STS, diversity	
”Represent this sentence for a natural language understanding task.”,	Generated+Authors	STS, diversity	
”Group passages based on semantic similarity.”	Generated+Authors	STS, diversity	
”Retrieve duplicate questions from the forum.”,	MTEB, Sprint-duplicate-questions	Task-specific	
”Retrieve the most relevant problem description for the given code implementation.”	MTEB, coreb-retrieval	Task-specific	
”Rerank products by relevance to the e-commerce query.”,	MTEB, E-commerce reranking	Task-specific	
”Classify the sentiment of the given review as positive or negative.”,	MTEB, Amazon-review-multi	Task-specific	
”Identify the topic or theme of Reddit posts based on the titles.”	MTEB, reddit-clustering	Task-specific	
”Identify the main and secondary category of Arxiv papers based on the titles.”	MTEB, arxiv-clustering	Task-specific	
”Generate a creative story based on the prompt.”,	Generated+Authors	No corresponding task	
”Extract all named entities from the text and categorise them as PERSON, LOCATION, or ORGANIZATION.”,	Generated+Authors	No corresponding task	
”Search for related information.”,	Generated+Authors	Vague	
”Find documents related to the topic.”,	Generated+Authors	Vague	
”Find something useful.”,	Generated+Authors	Vague	
”Retrieve the passage that contains overlapping vocabulary with the source sentence.”,	Generated+Authors	Vague	
”asdfjkl qpwoeiru zxcvbnm”,	Authors	Vague	
”hgJKSbf oiawnef LKJHDS kdjfbs”,	Authors	Vague	
”!!!! ??? ### @@@”,	Authors	Vague	
”Given a question, retrieve Wikipedia passages that answer the question.”	Authors	Finetuning	Retrieval
”Given a question, retrieve questions that are semantically equivalent to the given question.”	Authors	Finetuning	

Table 9: Instructions used in our experiments, their sources, reasoning for their selection, and which task they are considered to be relevant.

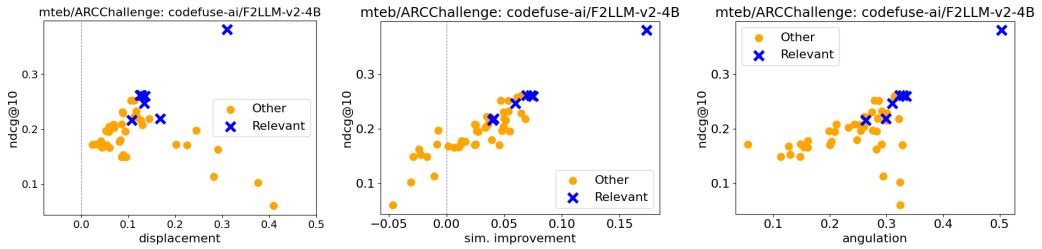


Figure 8: Displacement, similarity improvement, and angulation on outlier codefuse-ai/F2LLM-v2-4B against ndcg@10. Best performing and in all metrics clearly distinguishable instruction is "Retrieve the answer to the question."

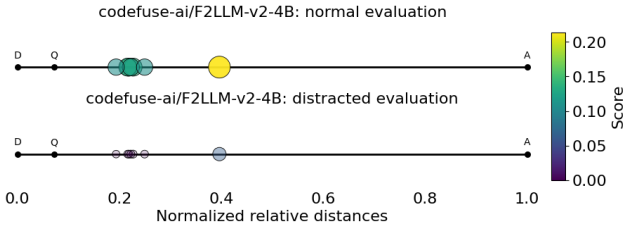


Figure 9: Relative distance plot for codefuse-ai/F2LLM-v2-4B on recall@1: While the best-performing instruction displays large displacement, high similarity increase, and angulation toward the correct target, distractors still inhibit performance greatly.

but the improvements are also substantial for other relevant instructions. Furthermore, all but one of the instructions listed are relevant to the task, indicating that fine-tuning was successful in incentivizing the model to react to instructions.

F Fine-Tuning Details

F.1 SQuAD Training Data

We use SQuAD v1.1 with the provided train / eval split, reserving the evaluation set for final results only. The original training set is divided into three subsets: 10,572 examples for experiments with random-question distractors, matching the number of paraphrase distractors, 2,009 examples for development and checkpoint selection, and the remaining 75,018 examples for fine-tuning.

Each examples produces two fine-tuning instances, one for QA and one for STS, resulting in 150,036 training examples. Because multiple SQuAD questions can share the same answer paragraph, negatives are sampled only from examples with different answer paragraphs to avoid introducing false negatives.

F.2 Tatoeba Training Data

Since MTEB contains only test data, we obtain training data from OPUS Tatoeba collection (v2026-07-08) (Tiedemann, 2012). We first deduplicate the data so that each source and target sentence appears only once. To avoid accidental train–test overlap, we remove pairs whose source or translation appears in any language's test set. We also remove pairs for which either sentence has a similarity above 0.8 to any test sentence in any language. Similarity is measured as the cosine similarity between TF-IDF vectors based on character 4-grams, after lowercasing and removing pucntuation. After filtering, we randomly sample up to 10,000 pairs per language, using 1,000 for development and the remaining for training. Vietnamese has only 7,913 pairs after filtering, resulting in 6,913 training and 1,000 development examples. All other languages contain 9,000 training and 1,000 development examples. All test sets contain 1,000 examples. For the random-distractor experiment,

Table 10: Top 3 instructions that improved their performance (Recall@1) the most during fine-tuning.

Dataset	Most improved instructions	Impr.
Tatoeba:cmn	Retrieve the corresponding translation in Mandarin Chinese.	0.49
	Given an English sentence, find its translation in Mandarin Chinese.	0.38
	Retrieve the corresponding translation.	0.30
Tatoeba:fin	Retrieve the corresponding translation in Finnish.	0.23
	Given an English sentence, find its translation in Finnish.	0.21
	Translate to Finnish.	0.16
Tatoeba:vie	Retrieve the corresponding translation in Vietnamese.	0.46
	Retrieve the corresponding translation.	0.35
	Given an English sentence, find its translation in Vietnamese.	0.30
Tatoeba:ara	Retrieve the corresponding translation in Arabic.	0.35
	Given an English sentence, find its translation in Arabic.	0.25
	Retrieve parellel sentences in Arabic.	0.21
Tatoeba:tur	Retrieve the corresponding translation in Turkish.	0.24
	Given an English sentence, find its translation in Turkish.	0.21
	Translate to Turkish.	0.19
Tateoba:fra	Retrieve the corresponding translation in French.	0.53
	Translate to French.	0.40
	Given an English sentence, find its translation in French.	0.40
Tatoeba:spa	Retrieve the corresponding translation in Spanish.	0.35
	Retrieve the corresponding translation.	0.32
	Given an English sentence, find its translation in Spanish.	0.22
Tatoeba:deu	Retrieve the corresponding translation in German.	0.41
	Retrieve the corresponding translation.	0.35
	Given an English sentence, find its translation in German.	0.23
SQuAD	Given a question, retrieve Wikipedia passages that answer the question.	0.46
	Given a question, find a related document.	0.39
	Find documents related to the topic.	0.33

Table 11: Fine-tuned performance for ARCChallenge: The baseline performance for ARCChallenge is low, but recall performance with paraphrase distractors increases while rank-based NDCG goes down. Note that the rank-based approach does not reveal distractor performance as strongly as recall@1.

	Recall@1		Recall@5		NDCG@10	
Model	Normal	Paraphr.	Normal	Paraphr.	Normal	Paraphr.
baseline	0.059	0.000	0.183	0.138	0.139	0.097
SQuAD-finetuned	0.051	0.003(↑)	0.173	0.142(↑)	0.130	0.094 (↓)

we additionally sample 1,000 English sentences from monolingual English Tatoeba, applying the same filtering against all training, development, and test sets.

F.3 Fine-Tuning Parameters

Fine-tuning is performed with the SWIFT framework (Zhao et al., 2024) using full-parameter fine-tuning, an InfoNCE loss, and cosine learning rate scheduler. We fine-tune for one epoch and save a checkpoint every 1,000 steps. The final checkpoint is selected based on performance in the paraphrase distractor setting. We use per-device training batch size is of 8, gradient accumulation of 1, a learning rate of 1e-6, a warmup ratio of 0.05, weight decay of 0.01, InfoNCE temperature of 0.05, and a maximum sequence length of 1,024 tokens.

We do not use fake-negative masking, a similarity-margin filtering technique commonly used to avoid false negatives, because this could mask the query-side distractors that we explicitly want the model to learn from. Query-side negatives are included in 50% of the training examples to balance learning from these hard negatives while preserving standard retrieval behavior. Because query-side negatives are assumed to have considerably higher similarity, they can contribute disproportionately to the InfoNCE objective, where softmax weighting emphasizes high-similarity negatives.

F.4 Fine-Tuning Instructions

The following instructions are used in fine-tuning:

- SQuAD QA: *Given a question, retrieve Wikipedia passages that answer the question.*
- SQuAD STS: *Given a question, retrieve questions that are semantically equivalent to the given question.*
- Tatoeba bitext: *Retrieve the corresponding translation in* $\{lang\}$.
- Tatoeba STS: *Retrieve semantically similar sentences.*

The same task-specific instractions are also used in the main evaluation results.

G Language-Specific Results for Tatoeba

Tatoeba fine-tuning performance across checkpoints is shown separately for each language in Figure 10, together with the average over languages.

H Fine-Tuning Without Query-Side Negatives

To ensure that the improved performance in the paraphrase distractor setting does not result merely from additional task-specific fine-tuning, we also fine-tuned models on the same SQuAD and Tatoeba data using the same training setup, but without query-side negatives. Figure 11 shows performance across fine-tuning checkpoints. Models trained without query-side negatives show no improvement in the distractor setting, indicating that the gains result from the added negatives rather than task-specific fine-tuning alone.

I Full MTEB results for fine-tuned models

The full results on the MTEB English v2 benchmark are shown in Figure 12. The benchmark includes 41 tasks across 7 categories. We report the absolute score difference (fine-tuned score minus base score), with all MTEB scores running between [0,1]. Positive values indicate better performance after fine-tuning. The MTEB mean is the arithmetic mean across all tasks.

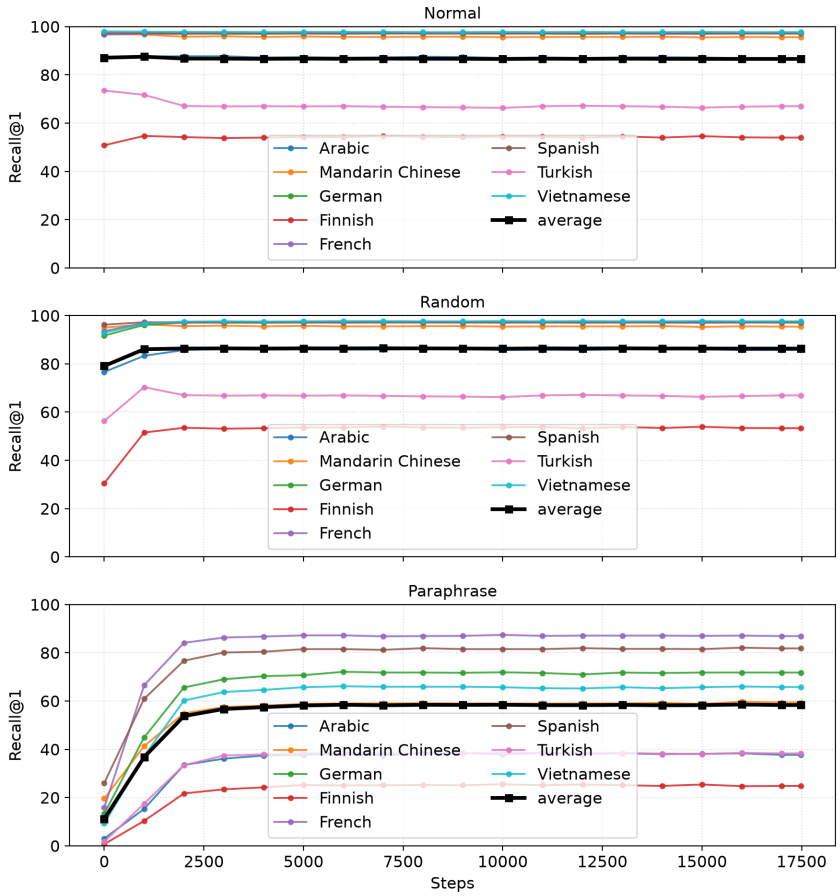


Figure 10: Language-specific results for Tatoeba fine-tuning checkpoints on the devevelopment data.

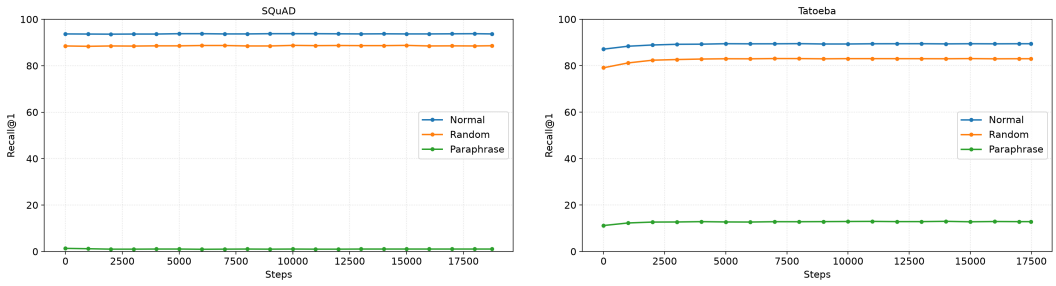


Figure 11: Development set performance during SQuAD and Tatoeba fine-tuning without query-side negatives. Continued fine-tuning without inserted distractors does not improve performance in the paraphrase distractor setting.

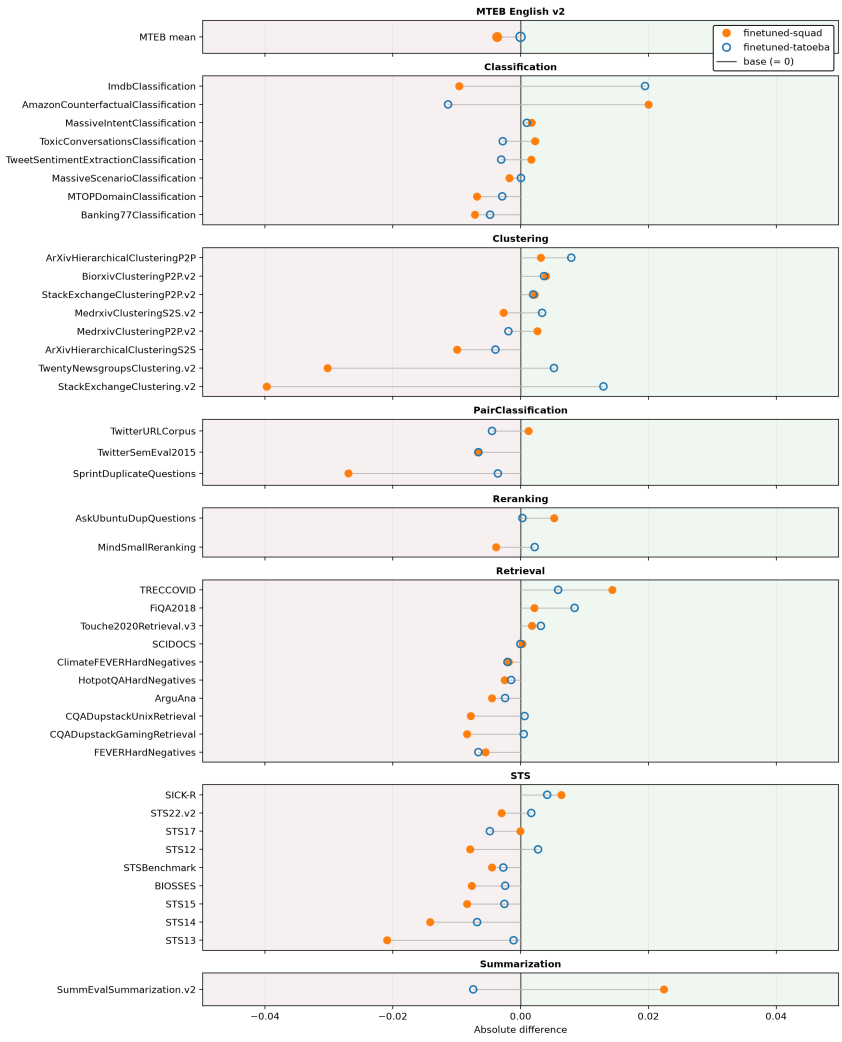


Figure 12: MTEB English v2 benchmark results for the fine-tuned models, compared with the base model. We report the absolute difference, where positive values indicate better performance after fine-tuning.

J COMPUTATIONAL DETAILS

We use AMD MI250x GPUs & Nvidia GH200 Grace Hopper GPUs, and AMD EPYC 7763 & AMD Turin 9965 CPUs on an *anonymous HPC environment* to run all experiments. All the following compute estimates are top-end approximations.

Fine-tuning We fine-tune 4 models (two datasets: SQuAD and Tatoeba, two settings: paraphrase distractors and control). Training each model took approximately 12 hours on one GPU.

Structure We calculate structural metrics (all instructions together), 8 models and 9 datasets (ARC-Challenge + 8 Tatoeba datasets) and additionally we run the fine-tuning comparisons, which include the 2 fine-tuned models, one on SQuAD, one on all Tatoeba languages, and the baseline model for SQuAD: all together 82 combinations. Each combination was run on 2 GPUs, and took approximately 20 to 60 minutes, depending on model size.

Evaluation We evaluated the same 82 combinations as above in both standard and distracted setting. We again used 2 GPUs and the evaluation took approximately 45 to 90 minutes for ARCChallenge and Tatoeba (both with $\tilde{1}000$ examples), and approximately 6-7 hours for the two SQuAD ($\tilde{1}0$k examples) settings. Running the additional MTEB English v2 benchmark on one GPU took 8 hours per model.

Altogether, we used at most 504 GPU hours.