Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Fine-tuning and retrieval augmentation for crystallography and cryo-EM question answering using a community-derived corpus

Emad Alharbi

emalharbi@ut.edu.sa

University of Tabuk

Research Article

Keywords: crystallography, fine-tuning, question answering

Posted Date: November 19th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11741379/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Fine-tuning and retrieval augmentation for crystallography and cryo-EM question answering using a community-derived corpus

Emad Alharbi[1*]

[1*]Department of Computer Science, Faculty of Computers and Information Technology, University of Tabuk, Tabuk, 71491, Saudi Arabia.

Corresponding author(s). E-mail(s): emalharbi@ut.edu.sa;

Abstract

Structural biology mailing lists such as CCP4, CCP-EM, Phenix and COOT contain decades of expert answers to practical problems, but an answer is difficult to find even when it has been given before. A question-and-answer corpus was built from 103,600 messages across the four mailing lists, and threads were classified as questions. The answers were confirmed by the asker's reply or by the agreement of several participants, identified by three LLM models. This gave 4,320 curated question-and-answer pairs, which were used for fine-tuning and retrieval augmentation for six LLM models, two sizes each from the Mistral, Llama and Qwen families. The LLM models were evaluated under eight conditions: base and fine-tuned, each without retrieval and with retrieval from the software documentation, from the mailing-list corpus, and from both. The base models with mailing-list retrieval were preferred by the LLM judges over the base models alone, the fine-tuned models, and even the base models with retrieval from the software documentation. The data sets and the fine-tuned models are publicly available.

Keywords: crystallography, fine-tuning, question answering

1 Introduction

Over several decades, structural biology mailing lists such as CCP4, CCP-EM, Phenix and COOT have become a knowledge corpus that includes a large number of expert

answers to practical problems, from data processing and phasing to model building and refinement. This knowledge is valuable to new researchers; however, the large number of messages, and answers spread over several replies, make an answer difficult to find, even when it has been given before.

In the last few years, developers of crystallography programs have been implementing classical machine learning and artificial intelligence (AI) methods to improve their programs rather than relying on heuristic methods (for a review, see [1]), for example in model building [2, 3]. Large language models (LLMs) can now support researchers in their work, by answering general crystallography questions or by providing ready-to-use scripts to automate tasks such as running CCP4 programs [4]. These LLM models, such as GPT and Mistral, were trained on general domains and might generate uncertain answers which can be misleading [5–8]. The LLM models that were pretrained on specific research domains outperform their general models, for example, BioMistral, which uses the Mistral model as a foundation model and is further pretrained on the biomedical domain, and SciBERT, a BERT-based language model pretrained on scientific text [9, 10]. Therefore, general models need access to verified domain knowledge, either by further training or by retrieving it when a question is asked to answer these research-domain questions with more confidence.

The CCP4 Bulletin Board (CCP4BB) and the CCP-EM, Phenix and COOT mailing lists serve as a forum for experts to exchange ideas and discuss various aspects of crystallography and cryo-EM. Their discussion and questions over the years have created large datasets of questions and expert answers. In this paper, these datasets were used both to fine-tune LLMs and as retrieval sources, to answer crystallography and cryo-EM related questions and to provide advice for solving macromolecular problems.

2 Methods

2.1 Creating the training data sets

The data sets were collected from four mailing lists: CCP4, CCP-EM, Phenix and COOT Bulletin Boards. Table 1 shows the number of messages and email threads for these four mailing lists. The email threads were identified using the Message-ID where available; the Message-ID is a unique identifier assigned by the sending server and found in the replies to the thread, which can be used to reconstruct the threads. Otherwise, the message subject and timestamp were used: messages with the same subject and within two months of one another were grouped into a thread. The two-month window was chosen from the intervals between messages linked by Message-ID, of which 99.3% span less than 60 days. Messages with neither a Message-ID nor a subject, duplicate messages, where the same message was sent twice, and non-English messages were omitted from the data sets. The message threads with fewer than two replies were also omitted, as they are not useful for fine-tuning: a thread with no replies means the question was never answered, and a thread with one reply means the answer cannot be confirmed.

The message threads were classified into two groups: questions and non-questions, using three LLM models: Mistral-7B-Instruct-v0.3 [6], Meta-Llama-3-8B-Instruct [11] and Qwen2.5-7B-Instruct [12], providing each model with the message subject and

body. The classification was carried out without fine-tuning and the prompt was identical for the three models. A message was considered a question when at least two of the three models classified it as a question with at least 80% confidence (The same prompt used for the three models is reported in Appendix A). The confidence is the model's own estimate, which each model must meet individually, in addition to the two-of-three agreement.

The reply that answers each question was then identified. Three larger models were used to find out the answer from the thread replies, Meta-Llama-3-70B-Instruct, Qwen2.5-72B-Instruct and Mistral-Large-Instruct-2407. Each model was given the question, the replies and the email address of the person who asked the question so that the replies from other participants can be distinguished from the asker replies. The prompt was identical for the three models with no fine-tuning. Each model returns the group assigned to the thread, the reply index that answers the question and a short justification. Threads were classified into one of five groups as follows.

1. Confirmed: The asker explicitly confirmed the answer using confirmation keywords; for example, "it worked", "problem solved", "this fixed it".
2. Implicitly confirmed: The asker confirmed the answer, but indirectly, such as thanking the non-asker after their suggestions.
3. Majority confirmed: The majority of non-askers confirm the same answer.
4. Not confirmed: The asker clearly confirmed the answer did not work.
5. Uncertain: the question was not answered or had an ambiguous answer.

A thread was classified into one of these groups when two of the three models predicted the same classification group; otherwise, the thread was classified into the uncertain group. For the confirmed and implicitly confirmed groups, some of the models wrongly assigned a thread to one of these two positive groups although no reply from the asker confirmed the answer. Such cases were moved to the uncertain group. The three positive groups, confirmed, implicitly confirmed and majority confirmed, were used to fine-tune the models (The prompt used to identify the confirmed answer is reported in Appendix B).

The threads' replies are written as correspondence rather than as direct answers. They include email greetings, signatures, quotations from previous messages, and other features of email correspondence. Fine-tuning on such text teaches the model to reproduce the form of an email rather than the content of an answer. Therefore, the questions and answers were cleaned and rewritten as direct question-and-answer pairs using Mistral-7B-Instruct-v0.3 preserving the technical content. The two models, Meta-Llama-3-8B-Instruct and Qwen2.5-7B-Instruct, judged the rewritten answers to ensure the technical content was preserved, returning feedback to Mistral-7B-Instruct-v0.3 for another round of rewriting (The prompts used for the rewriting and for the judges are reported in Appendix C).

In addition to the mailing-list corpus, the CCP4.v.9.0.016 and Phenix.v.2.1-6408 documentation were prepared for retrieval. The two documentation sources were split into passages on heading boundaries to preserve the code and keyword blocks. Moreover, a third data set combines both CCP4 and Phenix documentation as well as question-and-answer pairs from the mailing lists. Retrieval was hybrid, fusing a BM25

ranking with a bge-large-en-v1.5 embedding ranking by reciprocal rank fusion, with the BM25 ranking given twice the weight of the embedding ranking, and the five highest-ranked passages were inserted into the prompt [13–15].

2.2 LLM model fine-tuning

Six models, from three model families, were fine-tuned on the data sets from Section 2.1: the Mistral model with 7B and 123B parameters, the Llama model with 8B and 70B, and the Qwen model with 7B and 72B. These three models are widely used on HuggingFace in their instruct versions, and were obtained in October and November 2025 (The model repositories are reported in Appendix E). Low-rank adaptation (LoRA) was used to fine-tune the models, a method that updates only a subset of model weights [16]. After several trials, the LoRA r (the dimension of the low-rank update matrices), alpha (the factor that scales the update's contribution), dropout, learning rate, and token maximum length were set to 64, 128, 0.05, 2.675e-05, and 2048, respectively, and these values were used for all six models. The LoRA adapters were applied to the attention and feed-forward projection layers. The fine-tuning was implemented in PyTorch with the Hugging Face Transformers, PEFT and Datasets libraries [17–19]. The small models, 7B and 8B, were fine-tuned on four A100 80GB GPUs; the 70B, 72B and 123B models were fine-tuned on eight A100 80GB GPUs using DeepSpeed ZeRO stage 3 with parameter and optimizer offloading [20, 21]. The fine-tuning ran for two epochs with early stopping to prevent overfitting when the loss function did not improve after three evaluations, which occurred every 10 steps. The data sets are split into 80% for training, 10% for evaluation, and the remainder for testing. The data split was stratified by mailing list so that every mailing list source appears in all the three parts of the data sets. Duplicate questions were merged into groups before the split, and no group spans two splits; paraphrased repeats of a question may nevertheless remain across splits. Each model was fine-tuned with three different random seeds to establish the variation between runs. The generated answers used in the evaluation were obtained from the models fine-tuned with the first seed; the three runs were used to assess validation-loss stability.

2.3 LLM model fine-tuning evaluation

Each model was evaluated under eight conditions, base and fine-tuned model alone, the base and fine-tuned models each with retrieval from CCP4 and Phenix documentation, the base and fine-tuned models with retrieval from the mailing-list training data set and each with retrieval from the documentation and training mailing list combined (The prompts used for the answer generation with and without retrieval sources are reported in Appendix D). The answers were generated with greedy decoding, with the input truncated to 3072 tokens and a maximum of 512 new tokens. The mailing-list retrieval index was built from the training split only; therefore, the test records themselves could not appear in the retrieved passages, although similar questions asked in separate threads may occur in both splits.

The models were evaluated using Precision, Recall and F1-score using BERTScore with roberta-large model and without baseline rescaling for the testing data sets from

Section 2.2. BERTScore compared the contextual embeddings of the reference and generated answers rather than matching words directly [22]. The tokens of the generated answer are matched with the most similar tokens of the reference answer using the cosine similarity and vice versa. Precision is the mean of these similarities over the tokens of the generated answer, recall is the mean over the tokens of the reference and F1 is their harmonic mean. The generated answer is used as the candidate when calculating the BERTScore. The text in the generated answer that does not match the reference can reduce precision; therefore, both precision and recall are reported. BERTScore measures similarity to a reference answer and does not verify knowledge correctness; therefore, two more evaluations were also conducted to address this problem.

The first is LLM-as-judge, the answers for each test question for all the eight conditions were provided to a judge model to choose the best match to the reference answer [23]. The judge ranked the eight answers based on their similarity to the reference answers. Each model's answers were judged by the three larger models, Meta-Llama-3-70B-Instruct, Qwen2.5-72B-Instruct and Mistral-Large-Instruct-2407. The order of the answers was shuffled for each question to eliminate any preference for the first answer [24].

The second evaluation is a comparison across the models, since the ranking of the eight conditions was carried out within a single model. For each test question, the answers generated under the condition most often ranked first by the majority of the judges in the first evaluation, by the six models, Mistral-7B-Instruct-v0.3, Meta-Llama-3-8B-Instruct, Qwen2.5-7B-Instruct, Meta-Llama-3-70B-Instruct, Qwen2.5-72B-Instruct and Mistral-Large-Instruct-2407, were judged by the three larger models; here no judge is independent, since each judge shares a family with two of the six models (The prompt used for both evaluations is reported in Appendix F).

3 Results

3.1 Data sets preparation

Messages were classified as questions or non-questions using the three small models described in Section 2.1. Figure 1 shows the three models and their classification for the messages. Mistral 7B, Qwen 7B, and Llama 8B models classified 10623, 9822 and 9561 threads as questions, respectively. However, at least two of the three models classified 10407 of the 11921 threads as questions with a confidence of at least 80%.

Figure 2 shows the number of answers classified into five groups: confirmed, implicitly confirmed, majority confirmed, not confirmed, and uncertain, using the three large versions of the Llama, Mistral, and Qwen. The Llama model confirmed most of the answers, and Qwen classified them as confirmed or uncertain. However, Mistral assigned most threads to the confirmed and implicitly confirmed groups, and only a few to the majority confirmed and not confirmed groups. After combining the models' classification, at least two of the models classified 2777 threads in confirmed group, 1116 into implicitly confirmed and 461 into majority confirmed.

Figures 3 and 4 show an example of the question and its answer, as extracted from the CCP4 mailing list, and after cleaning and rewriting. The original question opens

with a greeting and closes with the asker's initials, and the original answer is addressed to the asker by name and signed off. In both cases, the rewritten question and answer omitted these unrelated parts to provide a cleaner version. After rewriting, 33 answers and 1 question that were below the minimum length of 5 words were omitted, giving 4320 threads used for the fine-tuning and for retrieval (3455 for training, 432 for evaluation and 433 for testing). The excluded answers were mostly single commands given without explanation. Figure 5 summarises the number of messages, threads and question-and-answer pairs retained at each step of the data preparation.

3.2 Base model performance

Figure 6 shows the distribution of the BERTScore F1 over the 433 test questions for the base models, without fine-tuning or retrieval. The BERTScore F1 occupies a narrow band because the scores were calculated without baseline rescaling, an optional linear transformation based on scores for unrelated sentence pairs that spreads the values over a wider range; therefore, the differences between models are compared with the spread across the questions rather than read in absolute terms. The models have a central peak between 0.81 and 0.84, with most of the scores between 0.78 and 0.89. The BERTScore F1 for the Qwen 72B model ranges from 0.7082 to 0.9150, and for its smaller version from 0.7437 to 0.9080.

3.3 Models fine-tuning results

Table 2 shows the validation loss for each model, averaged over the three random seeds. Within each family, the larger model reached a lower loss than the smaller one. Both Mistral models reached their lowest loss at approximately one epoch, while the other four continued to improve until the end of the second epoch. The standard deviation across seeds was no greater than 0.0013 for every model, indicating that fine-tuning was stable between runs.

Table 3 shows the BERTScore precision, recall and F1 for each model for the eight conditions. The F1 score for the fine-tuning was higher than the base model for every retrieval source, with the highest F1 score for Mistral 123B with no retrieval source with 0.8570 compared to the base model with 0.8320 F1 score. Moreover, the precision was also higher for the fine-tuned model than its base model, with the highest improvement without retrieval achieved by Qwen 72B, from 0.7965 for its base model to 0.8561 when fine-tuned, and recall from 0.8485 to 0.8535; however, Mistral 7B showed the smallest improvement in precision, from 0.8239 to 0.8580, and recall slightly increased from 0.8508 to 0.8515. The fine-tuned models generated answers averaging 56 to 74 words, compared with the base models averaging 147 to 228 words, while the reference answers averaged 57 words.

3.4 LLM-as-judge results

Figure 7 shows the percentage of generated answers judged by three LLM models, and when two models ranked the same answer as first preference, for the base and fine-tuned models with and without the retrieval sources. For the Qwen 7B model, the best-ranked answer came from one of the three retrieval conditions for 38% of

the questions, compared with 7% for the base model alone, and for Mistral 123B for 37% of the questions, compared with 11% for the model without retrieval sources. For the fine-tuned Qwen 72B model, the best-ranked answer came from one of the three retrieval conditions for 23% of the questions, compared with 6% for the fine-tuned model alone. The models answered more questions closer to the reference answer when using the mailing-list data set rather than the software documentation; for example, Mistral 7B answered 17% of the questions closer to the reference answer when using the mailing list, compared with only 9% when using the software documentation.

Figure 8 shows the percentage of the fabricated generated answers, where the models invented program names or commands for the six models evaluated by three LLM judges, for the base and fine-tuned models with and without retrieval sources. The retrieval bars contain the answers generated with the three retrieval sources (1299 answers per bar), while the no-retrieval bars contain 433 answers each. The fabrication flags are the judges' assessments and were not verified against the software. The fine-tuned models had the highest percentage of fabricated answers, between 3% and 12%, compared with the base models, which had only between 1% and 4% fabricated answers. However, the use of the retrieval sources reduced the percentage of fabricated answers for both the base and fine-tuned models, to between 0% and 3% for the base models and between 3% and 9% for the fine-tuned models.

Figure 9 shows the percentage of the answers that the judge models found did not match the reference answers, where the models' answers contained real program names and commands that were not included in the reference answers, unlike the fabricated answers, where program names and commands were invented. Moreover, answers that agreed with the reference answers in general terms but did not share a specific program, keyword or conclusion were also considered no-match answers. For the majority of judges, the Llama 8B model had the highest percentage of no-match answers, at 6%, and Qwen, at both sizes, had the lowest, at 2.5%. Comparing the LLM judge models themselves in finding no-match answers shows that Llama 70B judged the highest number of answers as no match, compared with models such as Qwen 72B, which judged only 0.2% of the Qwen 7B answers as no match and none for the other models.

Figure 10 shows an example of a question, the reference answer and the answers generated by Mistral 123B, base and fine-tuned versions, with and without the mailing-list retrieval. The base model with retrieval answered the question the same as in the reference answer, while the base model without the retrieval accepted the premise of the question. The fine-tuned model answered the opposite of the reference answer, and with retrieval it repeated the same sentence until the generation limit was reached. Moreover, the fine-tuned model's answer had the highest BERTScore F1 among the four answers, but was not chosen by the LLM judges (four examples of the generated answers under the four conditions, where one of the conditions was ranked best by a majority of the judges, are shown in Appendix G).

Table 4 shows the percentage of the answers ranked first or among the top three across the models, judged by three LLM models, for the answers generated by the base model with mailing-list retrieval, the condition most often ranked first by the majority of the judges. The Mistral 123B answers were ranked first by all the judges

for 8.5% of the questions and among the top three for 34.4%, while its smaller version was less favoured by the judges: only 3.2% of its answers were ranked first by all the judges and 12.2% among the top three.

4 Discussion

This paper presents the comparative evaluation of LLM fine-tuning and retrieval-augmented generation for answering crystallography and cryo-EM questions. The data sets were collected from the experts' emails and filtered to exclude non-question messages, and the message replies were classified to identify the confirmed answer, either by the asker's confirmation or the majority vote. Since not every answer in a public forum is correct, only these confirmed answers were used for fine-tuning and retrieval. Moreover, the dataset was cleaned to eliminate email-style writing and converted into direct questions and answers.

Six models were fine-tuned, from three model families, which are widely used in their instruct versions. LoRA was used as a light method to fine-tune the models without updating all the model weights. The LoRA arguments were set after several trials to obtain the best set. The number of epochs was set at a small number to reduce the risk of overfitting, with early stopping after three rounds without improvement, and to obtain the best model with the lowest loss score.

Comparing the models' performance using BERTScore shows that the F1 score improved for the fine-tuned models over their base models. The precision score improved more than the recall score for all the models. The improvement in the precision may partly reflect the fine-tuned models' answers being shorter and containing less text that is not in the reference answers.

Judging the generated answers using LLM models as judges shows that the base models answered the questions better when combined with retrieval sources. The improvement when using retrieval sources with base models suggests that providing the relevant context helps the models produce answers closer to the reference answers. These answers with retrieval are longer than the fine-tuned answers, which means length alone does not indicate a less correct answer, and the fine-tuned models' answers were flagged more often for naming a program or command that does not exist.

Comparing the different retrieval sources shows that using the mailing-list source assisted the models to answer the questions closer to the reference answers than using the software documentation, which suggests that the documentation might lack the answers.

The corpus was prepared automatically: the classification of the questions, the selection of the confirmed answers and the rewriting of the answers were carried out using LLMs. Moreover, the evaluation of the generated answers also used LLMs to measure agreement with the reference answers. Finally, the usefulness of the answers to researchers was not evaluated; a user study is left for future work.

The confirmed question-answer pairs prepared in this work might be useful for researchers' training and for further LLM fine-tuning. For future work, additional datasets collected from crystallography research publications, CCP4 Study Weekend talks, messages from newer platforms such as the Reciprocal Space Station forum, and

crystallography textbooks will be considered. It is envisaged that this could lead to better LLM models.

5 Conclusion

A curated question-and-answer corpus was built from four structural biology mailing lists and used for fine-tuning and retrieval for the LLMs. The fine-tuning improved the answers' similarity to the reference answers when measured with BERTScore; however, the LLM judges preferred the base model answers with mailing-list retrieval. Moreover, the mailing-list corpus was a more helpful retrieval source than the software documentation. The corpus, the code, and the fine-tuned models are publicly available for further research.

Acknowledgements. The author is grateful to the HPC team at the University of Tabuk for providing the GPUs that were used for the experiments in this project.

Declarations

Funding. Not applicable.

Competing interests. The author declares no competing interests.

Ethics approval and consent to participate. Not applicable.

Consent for publication. Not applicable.

Availability of data and materials. The data sets and code used in the experiments are available at https://doi.org/10.5281/zenodo.23079611. Moreover, the fine-tuned models are available on Hugging Face at https://hf.co/collections/ealharbi/crystallography-and-cryo-em-llms. A ready-to-use interface for the small fine-tuned models and for the base models with and without the retrieval source is available at https://huggingface.co/spaces/ealharbi/crystallography-cryoem-assistant.

Author contribution. E.A. is the sole author; he designed the study, performed the experiments and wrote the manuscript.

AI Use Declaration. An AI assistant (Claude Fable 5, Anthropic, https://claude.ai) was used to assist with writing the code and prompts and for AI-assisted copy editing of the manuscript text; all code and prompts were reviewed and tested by the author, who takes full responsibility for the content of this work. The use of LLMs within the research is described in the Methods section.

Table 1 Messages and threads from the four mailing lists. The email threads were identified by Message-ID where available and by subject and timestamp otherwise. The threads with at least two replies were used in the experiments.

			Threads	
Mailing list	Period	Messages	Fewer than two replies	Two or more replies
CCP4BB	2007–2025	63 278	3 080	6 700
CCP-EM	2013–2026	17 820	1 395	2 020
PhenixBB	2005–2026	15 262	1 621	2 120
COOT	2007–2026	7 240	658	1 081
Total		103 600	6 754	11 921

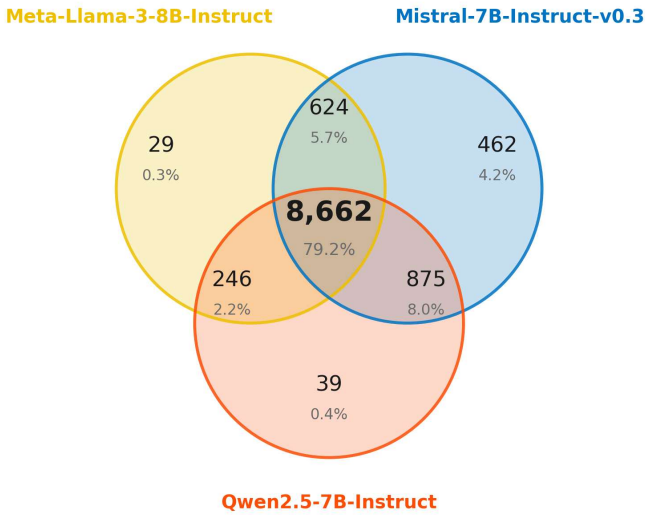


Fig. 1 Threads classified as questions by Mistral-7B-Instruct-v0.3, Meta-Llama-3-8B-Instruct and Qwen2.5-7B-Instruct. Each region shows the number of threads classified as questions by that combination of models.

Table 2 Validation loss for each model at the first evaluation and the best checkpoint, averaged over the three random seeds, with the standard deviation and the mean epoch at which the lowest loss was obtained.

Model	First loss	Best loss	SD	Reduction	Epoch
Qwen 7B	2.1909	1.8707	0.0001	14.6%	1.91
Qwen 72B	1.8253	1.6739	0.0002	8.3%	1.90
Mistral 7B	1.8430	1.6527	0.0013	10.3%	1.02
Mistral 123B	1.4584	1.4009	0.0013	3.9%	1.05
Llama 8B	2.1656	1.9472	0.0006	10.1%	1.90
Llama 70B	1.9578	1.7770	0.0006	9.2%	2.00

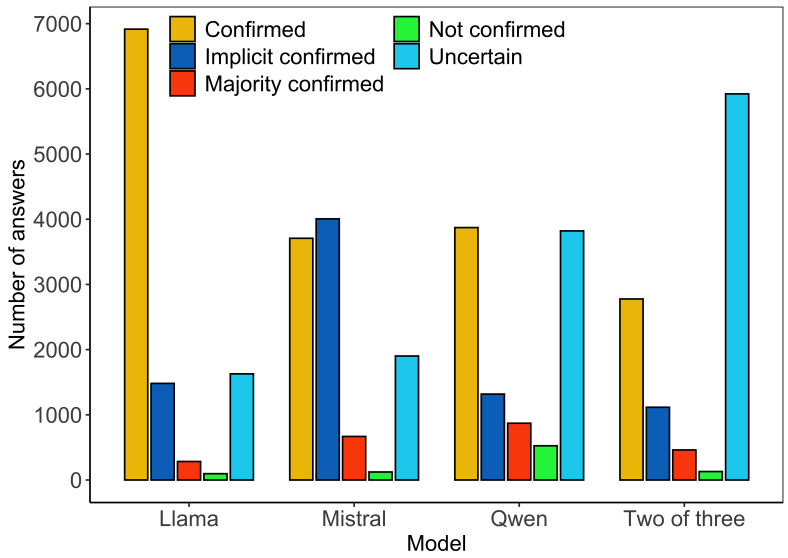


Fig. 2 Number of answers classified into five groups using the three large models, and the answers on which at least two of the models agreed.

Original	**Rewritten**
Hi everybody. I just was curious .... How often it is to find two or more crystalline forms in a single crystallization drop? Is it very common or is it way to rare? Thanks for your answers MP	Is it common or rare to find two or more crystal forms in a single crystallization drop?

Fig. 3 Original question as extracted from the CCP4 mailing list and the question after being rewritten.

Original	**Rewritten**
Hello MP, It happens, it is not that uncommon to find more than one space group in a drop, nor different morphologies (same or different space groups). Cheers, tom	It is not uncommon to find more than one space group or different morphologies in a single crystallization drop.

Fig. 4 Original answer as extracted from the CCP4 mailing list and the answer after being rewritten.

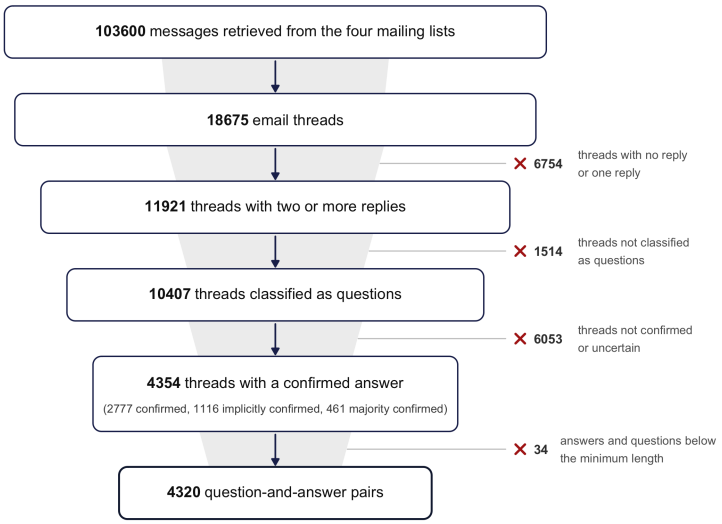


Fig. 5 Number of messages retrieved from the four mailing lists, the number of threads after the messages were combined into threads, the number of threads with two or more replies, the number of threads classified as questions, the number of threads with a confirmed answer, and the number of question-and-answer pairs used in the experiments.

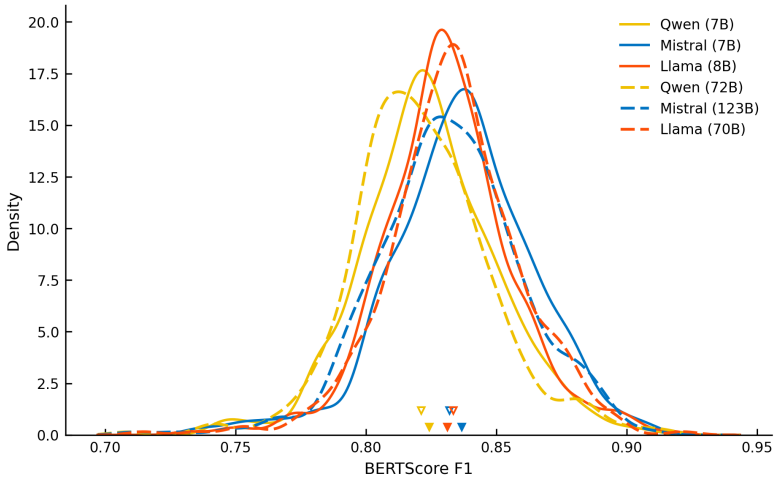


Fig. 6 Distribution of the BERTScore F1 over the 433 test questions for the base models, without fine-tuning or retrieval.

Table 3 BERTScore precision, recall and F1 for each model under the eight evaluation conditions for the 433 test questions for the base and fine-tuned models.

		Base model			Fine-tuned model		
Model	Retrieval	P	R	F1	P	R	F1
Qwen 7B	None	0.8041	0.8465	0.8242	0.8540	0.8511	0.8517
	Documentation	0.8067	0.8468	0.8258	0.8535	0.8510	0.8515
	Mailing lists	0.8181	0.8512	0.8339	0.8553	0.8524	**0.8532**
	Both	0.8146	0.8504	0.8316	0.8546	0.8525	0.8530
Qwen 72B	None	0.7965	0.8485	0.8212	0.8561	0.8535	0.8542
	Documentation	0.7963	0.8485	0.8211	0.8565	0.8533	0.8543
	Mailing lists	0.8002	0.8508	0.8242	0.8580	0.8546	0.8557
	Both	0.8007	0.8514	0.8247	0.8589	0.8544	**0.8561**
Mistral 7B	None	0.8239	0.8508	0.8367	0.8580	0.8515	**0.8536**
	Documentation	0.8254	0.8513	0.8377	0.8548	0.8511	0.8519
	Mailing lists	0.8328	0.8553	0.8435	0.8492	0.8514	0.8496
	Both	0.8228	0.8522	0.8369	0.8435	0.8486	0.8454
Mistral 123B	None	0.8168	0.8488	0.8320	0.8630	0.8527	**0.8570**
	Documentation	0.8199	0.8497	0.8341	0.8609	0.8532	0.8562
	Mailing lists	0.8208	0.8519	0.8355	0.8560	0.8545	0.8545
	Both	0.8192	0.8493	0.8335	0.8472	0.8512	0.8486
Llama 8B	None	0.8160	0.8478	0.8312	0.8540	0.8503	0.8511
	Documentation	0.8170	0.8480	0.8318	0.8541	0.8507	0.8516
	Mailing lists	0.8228	0.8508	0.8362	0.8496	0.8512	0.8497
	Both	0.8254	0.8498	0.8369	0.8526	0.8523	**0.8517**
Llama 70B	None	0.8189	0.8496	0.8335	0.8556	0.8526	**0.8533**
	Documentation	0.8210	0.8495	0.8346	0.8511	0.8532	0.8514
	Mailing lists	0.8360	0.8534	0.8442	0.8476	0.8542	0.8502
	Both	0.8334	0.8519	0.8421	0.8483	0.8533	0.8500

Table 4 Comparison of the six models under the base with mailing-list retrieval condition. The percentage of the 433 test questions on which the model's answer was ranked first or among the top three by each judge, and on which the three judges agreed on the same best answer or all ranked it among the top three.

	Qwen 72B judge		Llama 70B judge		Mistral-Large judge		All judges agree	
Model	First	Top-3	First	Top-3	First	Top-3	First	Top-3
Mistral 123B	24.5%	64.9%	13.6%	43.9%	23.8%	66.5%	8.5%	34.4%
Qwen 72B	23.3%	67.4%	14.8%	45.0%	22.6%	64.7%	6.5%	31.6%
Llama 70B	19.2%	52.2%	17.6%	45.5%	21.7%	54.5%	6.9%	26.6%
Qwen 7B	11.3%	49.2%	9.7%	38.3%	11.8%	44.6%	2.5%	20.6%
Llama 8B	12.2%	34.2%	11.8%	32.6%	9.5%	30.5%	2.8%	12.9%
Mistral 7B	9.5%	32.1%	11.5%	27.5%	8.1%	31.6%	3.2%	12.2%

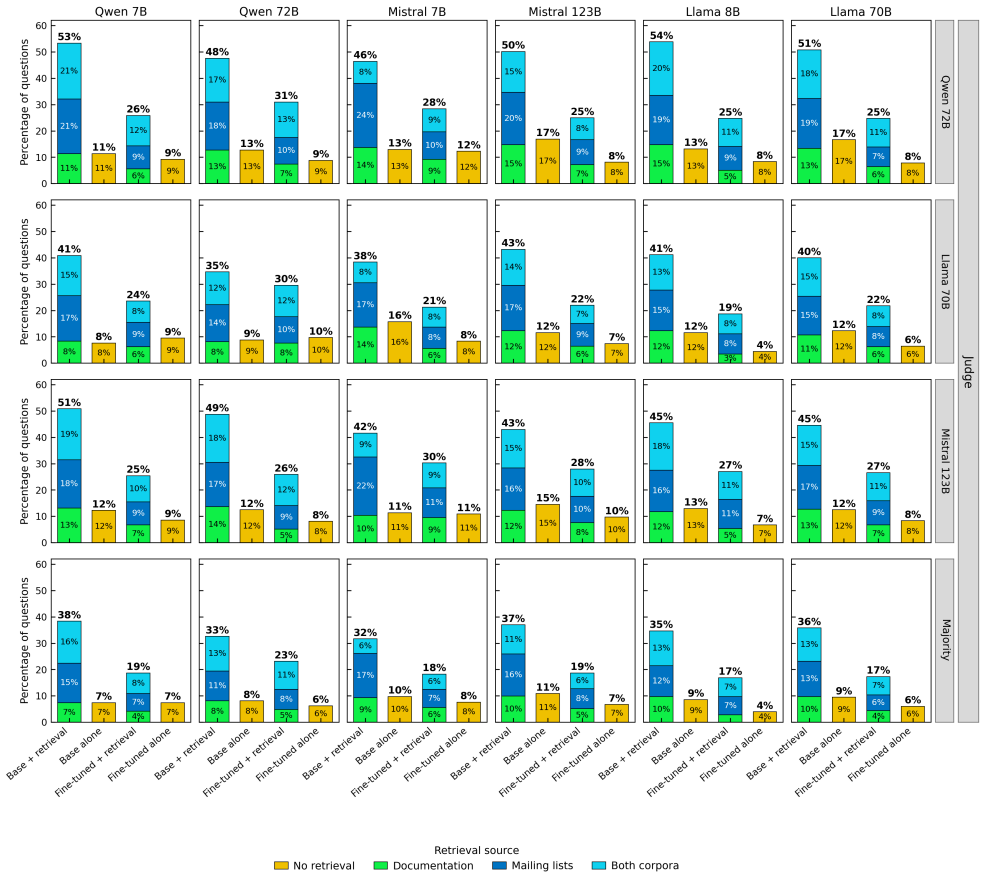


Fig. 7 Percentage of the 433 test questions whose generated answer was ranked best, for the base and fine-tuned models with and without retrieval sources evaluated by the three large models, and when at least two judges agreed on the same answer. In the retrieval bars, the three retrieval sources are stacked; each segment is the percentage of the 433 test questions answered best under that source, and the label above the bar is their sum.

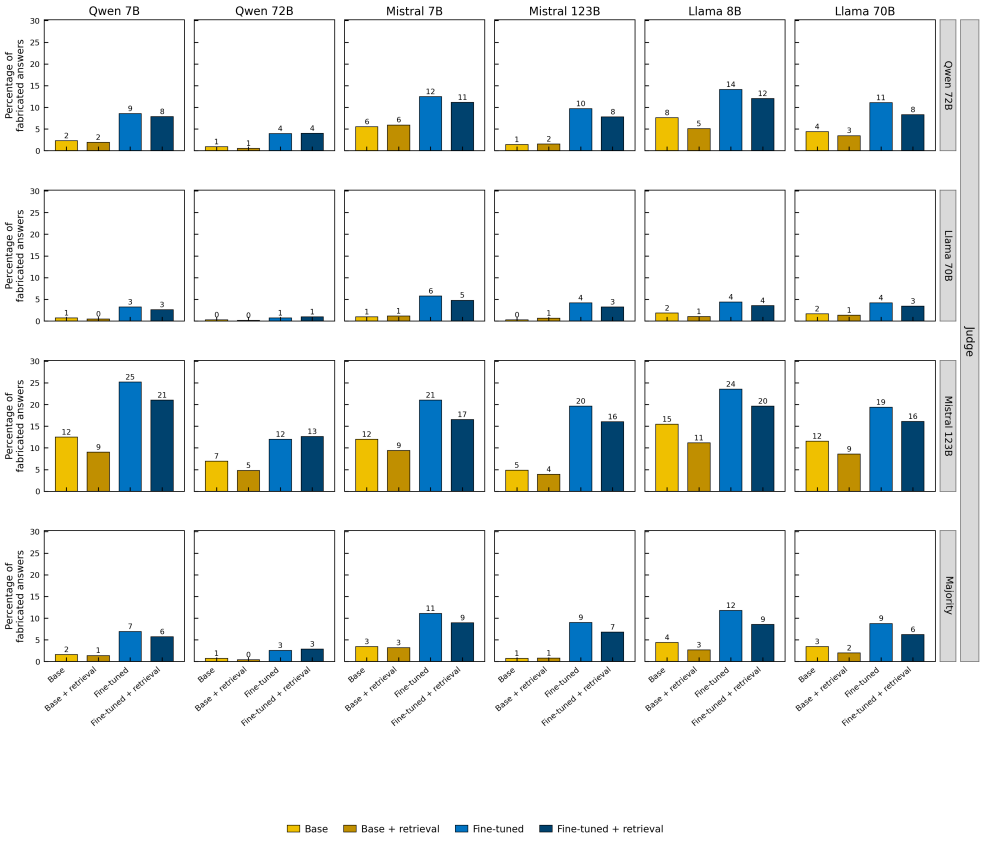


Fig. 8 Percentage of fabricated answers for the base and fine-tuned models with and without retrieval sources evaluated by the three large models, and when at least two judges agree that the same answer is fabricated.

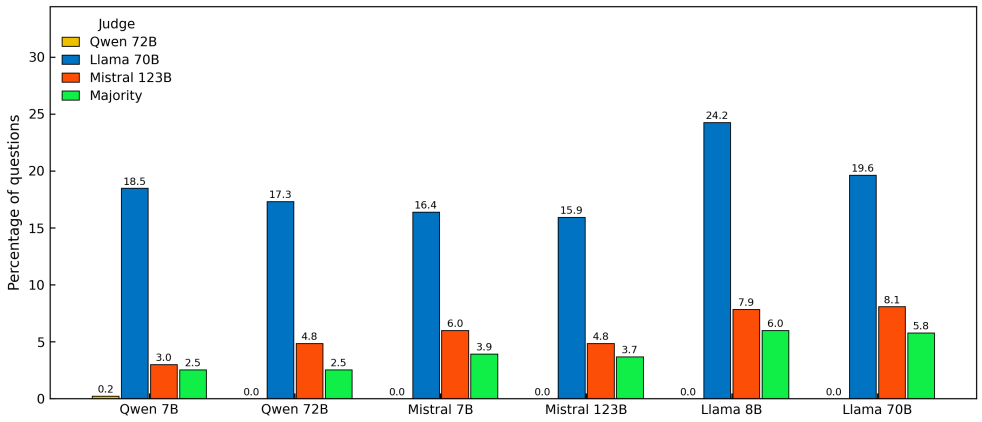


Fig. 9 Percentage of questions on which a judge found that none of the eight candidate answers matched the reference, that is, none shared a specific program, keyword or conclusion with it, evaluated by the three large models, and on which at least two judges agreed that none matched.

Question

1. How can a low clashscore indicate a high number of bad contacts in Phenix.refine? What strategies can be used to improve the clashscore? 2. Is there an acceptable lower limit for B-factor values in X-ray crystallography? Are there any suggestions for adjusting the adp restraints in Phenix.refine?

Reference answer

The lower the clashscore in Phenix.refine, the better. A lower clashscore indicates fewer bad contacts. If the clashscore is lower than others (as shown by Polygon), it means you are doing better on this one. Zero clashscore means you don't have any single steric clash, which is good. For B-factor values in X-ray crystallography, there isn't a specific acceptable lower limit. However, you can refer to the recent thread on ccp4bb titled "acceptable difference between Average B-factor and Wilson B" for more information. The acceptable limit depends on how you meaningfully and clearly define "low".

Base model (F1 = 0.8137)

1. Low clashscore and bad contacts. **A low clashscore in Phenix.refine may still indicate a high number of bad contacts if the overall number of contacts is high.** Clashscore is normalized by the number of contacts, so even a low score can represent many clashes in a large structure. *Improvement strategies:* manual inspection — check and adjust problematic areas using a molecular graphics program like Coot; refinement settings — adjust settings in Phenix.refine, such as increasing the weight of the clashscore in the target function; resolution — ensure the resolution is not overestimated; water molecules — carefully inspect and place water molecules, as they often contribute to clashes.

2. B-factor values and ADP restraints. There is no strict lower limit for B-factors, but values below 10 Å^2 are uncommon and may indicate over-refinement or incorrect restraints. To adjust the ADP restraints, `adp_restraints_weight` controls the restraint weight in Phenix.refine, and decreasing it allows more freedom in B-factor refinement; `adp_restraints_target` sets a desired average B-factor value. Group ADP refinement is worth considering for lower resolution data, and TLS (translation/libration/screw) refinement for high-resolution data, to model anisotropic motion.

Base model with mailing-list retrieval (F1 = 0.8336)

1. **Low clashscore and bad contacts. A low clashscore in Phenix.refine typically indicates fewer steric clashes, not more bad contacts.** However, if the clashscore is low but there are still many bad contacts, it might suggest that the clashscore is not capturing all the problematic interactions. This could happen if the clashscore is not being calculated accurately or if there are issues with the model that are not reflected in the clashscore. *Strategies to improve the clashscore:* upgrade Phenix — updates can improve geometry and clashscore calculations; manual rebuilding — inspect and fix clashes using a program like Coot; explicit hydrogen atoms — use them for better visualization and assessment of clashes; adjust weights — balance the geometry and X-ray terms in the refinement; hydrogens in riding mode — set `hydrogens.mode=riding` in Phenix.

2. Acceptable lower limit for B-factors. There is no strict lower limit for B-factor values in X-ray crystallography, but very low B-factors (below 10 Å^2, say) can indicate over-restrained refinement or errors in the model. B-factors should reflect the displacement of atoms due to thermal motion and static disorder, and very low values might suggest that this displacement is not being adequately modelled. *Adjusting ADP restraints in Phenix.refine:* `adp.restraints.weight` sets the weight of the atomic displacement parameter restraints, where increasing it tightens the restraints and decreasing it loosens them; `adp.restraints.target` sets a target B-factor to guide the refinement towards a desired average value.

Fine-tuned model (F1 = 0.8706)

The clashscore in Phenix.refine is a measure of the number of bad contacts in a model. A low clashscore indicates a high number of bad contacts. To improve the clashscore, you can use the following strategies: (1) check the model for any obvious errors, such as incorrectly placed atoms or incorrect bond lengths; (2) use the "clashscore" keyword in Phenix.refine to calculate the clashscore for your model; (3) adjust the adp restraints in Phenix.refine to improve the B-factor values. The acceptable lower limit for B-factor values in X-ray crystallography is not specified, but you can adjust the adp restraints to improve the B-factor values.

Fine-tuned model with mailing-list retrieval (F1 = 0.7855)

The clashscore in Phenix.refine is calculated using the MolProbity server, which is not the same as the clashscore in the Phenix GUI. The clashscore in the Phenix GUI is calculated using the Phenix software, which is not the same as the clashscore in the MolProbity server. The clashscore in the Phenix GUI is calculated using the Phenix software, which is not the same as the clashscore in the MolProbity server.

[the same sentence repeats until the generation limit is reached]

Fig. 10 Question from the test set, the reference answer that resolved it, and the answers generated by the base and fine-tuned Mistral 123B with and without mailing-list retrieval, together with their BERTScore F1.

Appendix A Questions classification prompt templates

```
[System]
You are a careful classifier. Output STRICT JSON ONLY. Decide if
the provided message is a question. Consider explicit question
marks, interrogatives (who/what/when/where/why/how), and implicit
requests (e.g., 'how to', 'could you', 'is it possible', 'any
idea', 'I wonder if'). Return JSON exactly as: {is_question:
true|false, confidence: float, reason: short}

[User]
Message:{TEXT}
Instructions:
-If any part asks for info/help/confirmation, classify as a
question.
-If it's a statement/announcement/answer only, classify as not a
question.
Return ONLY JSON, no prose.
```

Appendix B Answers classification prompt templates

```
[System]
You are a scientific email-thread adjudicator. Output MINIFIED
JSON ONLY with the required keys. No extra text.

[User]
We have a crystallography Q&A email thread.

Your task is to decide whether the original question was
ultimately solved, and if so, which REPLIES contain the confirmed
solution.

Statuses (choose the best):
- "confirmed": asker explicitly confirms success (e.g., \it
worked", \problem solved", \this fixed it").
- "implicit_confirmed": asker clearly implies success (e.g.,
specific thanks after trying a suggestion, \this did the trick").
- "majority_confirmed": several non-asker replies recommend the
SAME actionable approach; no later refutation from the asker.
- "not_confirmed": asker clearly says it did not work or the issue
```

```
is still unresolved.
- "uncertain": thread is genuinely ambiguous or incomplete; you
cannot tell if it was solved or not.

WHO WROTE A REPLY MATTERS:
- A reply from one of the ASKER EMAILS is the asker speaking
again. Such a message CONFIRMS that the thread was resolved, but
is usually not itself the answer.
- Include an asker's own reply in "confirmed_reply_indices" ONLY
when it states what actually worked (names a program, keyword,
setting, file or procedure). "I have finally solved the structure,
thanks to all" states nothing and must NOT be indexed.
- Whenever a non-asker reply contains the solution, index that
reply rather than the asker's confirmation of it.

A REPLY THAT ONLY REPORTS SUCCESS IS NOT AN ANSWER:
- "Problem solved!", "That worked, thanks!", "Fixed it!" carry no
information a future reader could act on. Do not index them.
- If no reply in the thread says what to do, the correct status is
"uncertain" with an empty "confirmed_reply_indices", even when the
thread clearly ended happily.

ATTRIBUTE ACCURATELY:
- The justification must not credit one replier with something
another replier said. If two replies differ, describe them
separately or cite only the one you indexed.

IMPORTANT GUIDELINES:
- Prefer one of "confirmed", "implicit_confirmed",
"majority_confirmed", or "not_confirmed" whenever possible.
- Use "uncertain" ONLY when there is truly no clear evidence
either way.
- If at least one reply clearly contains the solution or the
asker's explicit confirmation, you MUST choose a positive status
("confirmed", "implicit_confirmed", or "majority_confirmed") and
include at least one index in "confirmed_reply_indices".
- If multiple replies all clearly contribute to the confirmed
solution, you may include multiple indices (typically 1{3).
- If the asker explicitly says the suggestions did not work or the
problem persists, choose "not_confirmed" (even if people proposed
guesses).
- If replies are only guesses and the asker never comes back, and
you cannot infer success, then use "uncertain".
```

```
REPLIES are zero-based indexed as shown below.
For each index you include in "confirmed_reply_indices", there
must be clear textual evidence that this reply is part of the
accepted / confirmed solution.

Output MINIFIED JSON ONLY. Use EXACT keys and types.

Return JSON with EXACT keys:
"final_status": "confirmed" | "implicit_confirmed" |
"majority_confirmed" | "not_confirmed" | "uncertain",
"confirmed_reply_indices": [ints], // zero-based indices into
REPLIES
"justification": "<=200 chars explanation",
"evidence_snippets": ["short quotes supporting the decision"]

QUESTION (subject + body):
question

ASKER EMAILS: asker_emails

REPLIES (index, from, Reply-To, excerpted text):
replies

Return MINIFIED JSON on ONE line. No extra text.
```

Appendix C Answers cleaning and re-writing prompt templates

```
[Question System]
You extract the true main question from a noisy email thread.

Your job:
- Read the raw email body exactly as given.
- Keep the main question from the current sender.
- Keep enough technical context to understand the problem.
- Keep important experimental conditions, software names,
parameters, measurements, concentrations, pH, temperature, and
failed attempts when they matter.
- Remove quoted earlier emails, duplicate copies, MIME garbage,
signatures, unsubscribe text, mailing list footer, and repeated
text.
```

```
- Remove opening and closing phrases such as: Dear all, Hi,
Hello, Thanks, Thank you, Best regards, Sincerely, Cheers, and the
sender's name if they are not technically relevant.
- Preserve the original meaning and key technical details.
- Stay close to the original wording when practical, but small
cleanup and light compression are allowed.
- Do not invent information.
- Do not add labels like ''The cleaned question is:''.
- Do not surround the output with quotation marks.
- Return only the cleaned question text.
- NO EXTRA TXT ONLY THE CLEANED QUESTION.

[Answer System]
You extract the true answer from a noisy reply email.

Your job:
- Read the raw reply body exactly as given.
- Keep only the answer written by the replier.
- Keep technically relevant content that is part of the answer.
- Keep supporting explanation, conditions, caveats, workflow
steps, software names, parameters, commands, and examples when
they help explain the answer.
- Remove quoted earlier emails, copied question text, mailing
list footer, signatures if not part of the answer, MIME garbage,
unsubscribe text, and repeated text.
- Preserve the original meaning and key technical details.
- Stay close to the original wording when practical, but small
cleanup and light compression are allowed.
- Do not invent information.
- Do not add labels like ''The cleaned answer is:''.
- Do not surround the output with quotation marks.
- Return only the cleaned answer text.
- NO EXTRA TXT ONLY THE CLEANED ANSWER.

[Question User]
Extract the cleaned question from this email.

SUBJECT:
{subject}

RAW_EMAIL_BODY:
{body}

[Answer User]
```

```
Extract the cleaned answer from this reply email.

QUESTION:
{question}

RAW_REPLY_BODY:
{answer}

[Judge System]
You are a quality judge for extraction output.

Judge whether the candidate follows the extraction rules well
enough for this task.

Important:
- Prefer preserving the original meaning and key technical
details.
- Staying close to the original wording is good, but minor
cleanup, compression, and light rephrasing are acceptable.
- Reject outputs that add invented information.
- Reject outputs that add wrapper text such as ''The cleaned
answer is:'' or ''The cleaned question is:''.
- Reject outputs wrapped in quotation marks if those quotes were
added by the model.
- Reject outputs that remove too much important technical context.

Return JSON only with exactly these keys:
{
''approved'': true or false,
''score'': 0.0 or 0.5 or 1.0,
''feedback'': ''short specific revision feedback''
}

Scoring:
- 1.0 = good extraction; preserves meaning and key technical
details
- 0.5 = partly acceptable but needs revision
- 0.0 = poor extraction, invented content, too much loss of
meaning, or obvious wrapper text

[Question Judge User]
Task type: cleaned_question
```

```
Rules summary:
- Keep the main question from the current sender.
- Keep enough technical context to understand the problem.
- Remove quoted earlier emails, MIME garbage, repeated text,
footer, and non-technical greetings/signatures.
- Preserve the original meaning and key technical details.
- Minor cleanup and light rephrasing are acceptable.
- Do not add wrappers or quotation marks.
- Output only the cleaned question.

SUBJECT:
{subject}

RAW_EMAIL_BODY:
{body}

CANDIDATE_OUTPUT:
{candidate}

[Answer Judge User]
Task type: cleaned_answer

Rules summary:
- Keep only the answer written by the replier.
- Keep technically relevant explanation, workflow steps, caveats,
commands, parameters, and examples if they are part of the answer.
- Remove quoted earlier emails, copied question text, footer, MIME
garbage, repeated text, and non-technical greetings/signatures.
- Preserve the original meaning and key technical details.
- Minor cleanup and light rephrasing are acceptable.
- Do not add wrappers or quotation marks.
- Output only the cleaned answer.

QUESTION:
{question}

RAW_REPLY_BODY:
{answer}

CANDIDATE_OUTPUT:
{candidate}

[Question Regeneration User]
```

```
Your previous output needs revision.

Please improve it while keeping the same meaning and key technical
details.

Rules:
- Keep the main question from the current sender.
- Keep enough technical context to understand the problem.
- Minor cleanup and light rephrasing are acceptable.
- Do not invent information.
- Do not add labels like ''The cleaned question is:''.
- Do not add quotation marks.
- Remove quoted earlier emails, signatures, MIME garbage, footers,
and repeated text.
- Return only the cleaned question text.

SUBJECT:
{subject}

RAW_EMAIL_BODY:
{body}

PREVIOUS_OUTPUT:
{previous_output}

JUDGE_FEEDBACK:
{feedback}

[Answer Regeneration User]
Your previous output needs revision.

Please improve it while keeping the same meaning and key technical
details.

Rules:
- Keep only the replier's answer.
- Keep technically relevant explanation and steps when useful.
- Minor cleanup and light rephrasing are acceptable.
- Do not invent information.
- Do not add labels like ''The cleaned answer is:''.
- Do not add quotation marks.
- Remove quoted earlier emails, copied question text, signatures,
MIME garbage, footers, and repeated text.
```

```
- Return only the cleaned answer text.

QUESTION:
{question}

RAW_REPLY_BODY:
{answer}

PREVIOUS_OUTPUT:
{previous_output}

JUDGE_FEEDBACK:
{feedback}
```

Appendix D Answer generation prompt templates

```
[System, without retrieval]
You are a helpful crystallography assistant. Be concise and
precise.

[System, with retrieval]
You are a helpful crystallography assistant. Be concise and
precise.
Reference material retrieved for this question is given below. Use
it when it is relevant; ignore it when it is not, and answer from
your own knowledge instead.
If the reference material does not address the question that was
asked, disregard it entirely. Do not answer a different question
because the reference material happens to discuss one.

Reference material:
{context}

[User]
{question}
```

Appendix E Base model repositories used for the fine-tuning

Table E1 Base model repositories used for the fine-tuning and their download links.

Family	Parameters	Hugging Face repository
Mistral	7B	`mistralai/Mistral-7B-Instruct-v0.3`
Mistral	123B	`mistralai/Mistral-Large-Instruct-2407`
Llama	8B	`meta-llama/Meta-Llama-3-8B-Instruct`
Llama	70B	`meta-llama/Meta-Llama-3-70B-Instruct`
Qwen	7B	`Qwen/Qwen2.5-7B-Instruct`
Qwen	72B	`Qwen/Qwen2.5-72B-Instruct`

Appendix F LLM-as-judge prompt template

```
[Judge System]
You are comparing answers against a reference answer that is known
to have resolved the question. You are not judging the answers on
your own knowledge of crystallography: you are judging how well
each one agrees with the reference.

[Judge User]
A crystallographer asked the following question on a mailing list.

QUESTION:
{question}

This is the REFERENCE ANSWER. It is the reply that actually
resolved the
question, and for the purpose of this comparison it is correct:

REFERENCE ANSWER:
{gold}

{n} candidate answers follow.

{answers}

Your task is to decide which candidate best matches the reference
answer.
Judge only by agreement with the reference. Do not reward an
answer for
being thorough, well organised, cautious, or for containing advice
you
believe to be good but which the reference does not give.

Work through these in order:

1. Does the candidate name the same program, command, keyword,
file or
setting as the reference? This is the strongest signal. An answer
naming
the same specific thing as the reference is better than one that
does
not, however well written.
```

```
2. If not the same one, does it name a different tool that does
the same
job? Give partial credit: a different valid route to the
reference's
outcome is worth more than generic advice, but less than naming
what the
reference names.

3. Does it reach the same conclusion or diagnosis as the
reference, even in
different words?

An answer that lists many possibilities, one of which happens to
match the
reference, is weaker than one that identifies the reference's
answer
directly. An answer that contradicts the reference is wrong
regardless of
how plausible it sounds.

Reply with JSON only, no other text:

{{''best'': <number>,
''second'': <number>,
''third'': <number>,
''matched'': ''the specific program, keyword or conclusion from
the reference that the best answer shares, or 'none' if it merely
agrees in general terms'',
''reason'': ''one sentence on why it matches the reference more
closely than the others'',
''fabricated'': [<numbers of any candidates naming a program,
command, keyword or option that does not exist>]}}
```

Appendix G Examples of the generated answers

Question

How to move a 5bp long double helix DNA molecule or a zone that spans multiple chains, as the current rotation/translation method does not work for such structures?

Reference answer

Rotate/Translate using the Molecule... toolbar togglebutton menu extension.

Base model

To move a 5 bp long double helix DNA molecule or a zone spanning multiple chains in a molecular modeling software where standard rotation/translation methods are ineffective, you can use the following approaches: *define a selection* — use the selection tools to specifically select the 5 bp DNA segment or the zone spanning multiple chains, for example in PyMOL: `select my_segment, resi 1-5 and chain A`; *use custom transformations* — apply custom transformations to the selected segment using matrix operations or specific commands [. . .]

Base model with mailing-list retrieval

To move a 5 bp long double helix DNA molecule or a zone that spans multiple chains, especially when the standard rotation/translation methods are not effective, you can follow these steps: *use molecular dynamics (MD) simulations* — software: GROMACS, AMBER, or NAMD; set up a short MD simulation to allow the DNA molecule to relax and move within the context of the entire structure, which can help in repositioning the DNA without disrupting the overall structure; *manual adjustment in Coot* — software: Coot; load the structure into Coot [. . .]

Fine-tuned model (preferred, 3 of 3 judges)

To move a 5bp long double helix DNA molecule or a zone that spans multiple chains in Coot, **you can use the "Move Zone" function**. This function allows you to move a selected zone of your structure, which can be useful for moving a DNA molecule or a zone that spans multiple chains.

Fine-tuned model with mailing-list retrieval

To move a 5bp long double helix DNA molecule or a zone that spans multiple chains, you can use the "rotate around" feature in Phaser, available in both CCP4 and Phenix. In ccp4i, you can use the "Brute" rotation function target in the "Additional Search Parameters" folder, choose "Around an angle", provide the Euler angles found during the search for the first domain, and set a range of 20-30 degrees for allowable orientation differences [. . .]

Fig. G1 Four answers generated by the base and fine-tuned Qwen 72B under the four conditions.

Question

How to generate a solvent mask containing values between 0 and 1 for use in Relion 3D classification, as the current mask is giving an error stating that it should only contain values between 0 and 1, and adding the extra parameter `--solvent_mask2 1` did not resolve the issue? Is there any additional parameter to apply solvent mask values between 0 and 1?

Reference answer

Set all values below 0 to 0 and all values larger than 1 to 1 in the mask.

Base model

To generate a solvent mask with values between 0 and 1 for use in Relion 3D classification, you can use the `relion_mask_create` command with the `--soft_edge` parameter. Here's a concise way to do it: *create the initial mask:* `relion_mask_create --i input_map.mrc --o initial_mask.mrc --ini_threshold 0.02 --width_soft_edge 5 --extend_inimask 2`; *normalize the mask to values between 0 and 1:* `relion_image_handler --i initial_mask.mrc --o normalized_mask.mrc --multiply_constant 1.0 --add_constant 0.0 --normalize` [. . .]

Base model with mailing-list retrieval

To generate a solvent mask containing values between 0 and 1 for use in Relion 3D classification, you can use the `relion_mask_create` command with appropriate parameters. Here's a concise way to do it: *create the mask:* `relion_mask_create --i input_map.mrc --o output_mask.mrc --ini_threshold 0.02 --width_soft_edge 5 --lowpass 15 --extend_inimask 2`, where `--i input_map.mrc` specifies the input map, `--o output_mask.mrc` specifies the output mask file [. . .]

Fine-tuned model

To generate a solvent mask containing values between 0 and 1 for use in Relion 3D classification, you can use the following command: `relion_mask_create --i input.mrc --o output.mrc --ini_threshold 0.05 --width_soft_edge 5 --extend_inimask_size 10 --width_edge_mask 10`. This command will create a mask with a soft edge, which should help resolve the issue with the current mask giving an error stating that it should only contain values between 0 and 1.

Fine-tuned model with mailing-list retrieval (preferred, 2 of 3 judges)

To generate a solvent mask containing values between 0 and 1 for use in Relion 3D classification, you can use the following command: `relion_image_handler --i mask.mrc --o mask_0_1.mrc --threshold 0 1`. **This command will threshold the values in the mask to ensure they are between 0 and 1**, potentially resolving the error you encountered.

Fig. G2 Four answers generated by the base and fine-tuned Mistral 123B under the four conditions.

Question

Can I download and install the latest Phenix release on a headless computing cluster using `curl` or `wget` commands, but get zero-size files when downloading the provided links?

Reference answer

```
wget --user=TheUser
--password=ThePassword
"LinkYouGotFromPhenixDownloadSite"
-O phenix-installer.tar.gz
```

Base model

The issue is likely due to the Phenix website requiring a login to access the download links. `curl` and `wget` commands won't work as they don't preserve the login session. Instead, you can use `wget` with the `--load-cookies` option to download the files. Here's an example: first, log in to the Phenix website and save the cookies to a file using your browser or a tool like `wget`: `wget --save-cookies=phenix_cookies.txt --keep-session-cookies https://www.phenix-online.org/download/;` then, use the saved cookies to download the Phenix release [. . .]

Base model with mailing-list retrieval (preferred, 3 of 3 judges)

Yes, you can download and install the latest Phenix release on a headless computing cluster using `curl` or `wget` commands. However, the zero-size files issue is likely due to the Phenix website requiring a login or authentication before allowing downloads. You need to authenticate or login to the Phenix website before downloading the installer. **You can do this by using the `--user` or `-u` option with `curl` or `wget` to provide your Phenix login credentials.** For example, with `curl`: `curl -u your_username:your_password -O` https://www.phenix-online.org/download/phenix-installer-⟨version⟩ [. . .]

Fine-tuned model (flagged by the judges as inventing content)

The issue with downloading the latest Phenix release on a headless computing cluster using `curl` or `wget` commands, resulting in zero-size files, is due to the links being protected by a cookie. To resolve this, you can use the `--cookie` option with `wget` to specify the cookie. For example: `wget --cookie=phenix_download=1` http://www.phenix-online.org/download/phenix-1.7.3-928-64bit-intel-linux-2.6-x86_64.tar.gz [. . .]

Fine-tuned model with mailing-list retrieval (flagged by the judges as inventing content)

The issue with downloading the latest Phenix release on a headless computing cluster using `curl` or `wget` commands is due to the fact that the links provided are not direct links to the files. Instead, they are links to a web page that requires a browser to download the files. To resolve this issue, you can use the `curl` command with the `-b` option to specify the cookie file, and the `-c` option to specify the cookie jar file. [. . .]

Fig. G3 Four answers generated by the base and fine-tuned Llama 70B under the four conditions.

Question

Can the same settings as in particle picking be applied in other tasks for improved display of micrographs and particles? If not, what are some useful parameters for the `relion_image_handler` program?

Reference answer

You can apply low-pass filtering to micrographs using the `--low_pass` parameter in the `relion_image_handler` program for improved display. This makes it easier to see the particles.

Base model (preferred, 3 of 3 judges)

While particle picking settings can be adapted for other tasks, they may not always be directly applicable. However, some parameters can be reused or modified to improve display of micrographs and particles in other tasks. For Relion's `image_handler` program, some useful parameters for improved display are: `--lowpass` — **apply a low-pass filter to reduce noise and improve visibility**; `--highpass` — apply a high-pass filter to enhance high-frequency features; `--normalize` — normalize the image intensity to improve contrast; `--gamma` — adjust the gamma correction [. . .]

Base model with mailing-list retrieval

The settings used in particle picking, such as the image size and binning, can be applied to other tasks in Relion to improve the display of micrographs and particles. However, not all settings are applicable to all tasks. For the `relion_image_handler` program, some useful parameters include: `-bin` or `--bin` — this parameter allows you to specify the binning factor for the micrographs, where a binning factor of 1 means no binning, while a higher value means binning the images [. . .]

Fine-tuned model

The `relion_image_handler` program can be used to display micrographs and particles with the same settings as in particle picking. To do this, you can use the `--display` option. For example, to display a micrograph with the same settings as in particle picking, you can use the command: `relion_image_handler --display --i micrograph.mrc`. To display a particle with the same settings as in particle picking, you can use the command: `relion_image_handler --display --i particle.star --j 1`. The `--j` option is used to specify the particle number.

Fine-tuned model with mailing-list retrieval

The same settings as in particle picking can be applied in other tasks for improved display of micrographs and particles. The `relion_image_handler` program can be used to adjust the display settings.

Fig. G4 Four answers generated by the base and fine-tuned Llama 8B under the four conditions.

References

[1] Vollmar, M., Evans, G.: Machine learning applications in macromolecular x-ray crystallography. Crystallography Reviews **27**(2), 54–101 (2021)

[2] Bond, P.S., Wilson, K.S., Cowtan, K.D.: Predicting protein model correctness in coot using machine learning. Biological Crystallography **76**(8), 713–723 (2020)

[3] Alharbi, E., Calinescu, R., Cowtan, K.: Buccaneer model building with neural network fragment selection. Biological Crystallography **79**(4), 326–338 (2023)

[4] Agirre, J., Atanasova, M., Bagdonas, H., Ballard, C.B., Baslé, A., Beilsten-Edmands, J., Borges, R.J., Brown, D.G., Burgos-Mármol, J.J., Berrisford, J.M., *et al.*: The ccp4 suite: integrative software for macromolecular crystallography. Biological Crystallography **79**(6), 449–461 (2023)

[5] OpenAI: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023) https://doi.org/10.48550/arXiv.2303.08774

[6] Jiang, A.Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D.S., Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L.R., Lachaux, M.-A., Stock, P., Scao, T.L., Lavril, T., Wang, T., Lacroix, T., Sayed, W.E.: Mistral 7B (2023). https://arxiv.org/abs/2310.06825

[7] Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y.J., Madotto, A., Fung, P.: Survey of hallucination in natural language generation. ACM computing surveys **55**(12), 1–38 (2023)

[8] Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B., *et al.*: A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. ACM Transactions on Information Systems **43**(2), 1–55 (2025)

[9] Labrak, Y., Bazoge, A., Morin, E., Gourraud, P.-A., Rouvier, M., Dufour, R.: BioMistral: A collection of open-source pretrained large language models for medical domains. Findings of the Association for Computational Linguistics: ACL 2024, 5848–5864 (2024) https://doi.org/10.18653/v1/2024.findings-acl.348

[10] Beltagy, I., Lo, K., Cohan, A.: Scibert: A pretrained language model for scientific text. Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP), 3615–3620 (2019)

[11] Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al.: The llama 3 herd of models. arXiv preprint arXiv:2407.21783 (2024)

[12] Bai, J., Bai, S., Chu, Y., Cui, Z., Dang, K., Deng, X., Fan, Y., Ge, W., Han, Y., Huang, F., et al.: Qwen technical report. arXiv preprint arXiv:2309.16609 (2023)

[13] Robertson, S., Zaragoza, H.: The probabilistic relevance framework: Bm25 and beyond. Foundations and trends® in information retrieval **4**(1-2), 1–174 (2009)

[14] Cormack, G.V., Clarke, C.L., Buettcher, S.: Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 758–759 (2009)

[15] Xiao, S., Liu, Z., Zhang, P., Muennighoff, N., Lian, D., Nie, J.-Y.: C-pack: Packed resources for general chinese embeddings. In: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 641–649 (2024)

[16] Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: LoRA: Low-rank adaptation of large language models. International Conference on Learning Representations (2022)

[17] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019)

[18] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al.: Transformers: State-of-the-art natural language processing. Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations, 38–45 (2020)

[19] Lhoest, Q., Del Moral, A.V., Jernite, Y., Thakur, A., Von Platen, P., Patil, S., Chaumond, J., Drame, M., Plu, J., Tunstall, L., et al.: Datasets: A community library for natural language processing. Proceedings of the 2021 conference on empirical methods in natural language processing: system demonstrations, 175–184 (2021)

[20] Rasley, J., Rajbhandari, S., Ruwase, O., He, Y.: Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining, 3505–3506 (2020)

[21] Rajbhandari, S., Rasley, J., Ruwase, O., He, Y.: Zero: Memory optimizations toward training trillion parameter models. SC20: international conference for high performance computing, networking, storage and analysis, 1–16 (2020). IEEE

[22] Zhang, T., Kishore, V., Wu, F., Weinberger, K.Q., Artzi, Y.: Bertscore: Evaluating text generation with bert. arXiv preprint arXiv:1904.09675 (2019)

[23] Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E., *et al.*: Judging llm-as-a-judge with mt-bench and chatbot arena. Advances in neural information processing systems **36**, 46595–46623 (2023)

[24] Wang, P., Li, L., Chen, L., Cai, Z., Zhu, D., Lin, B., Cao, Y., Kong, L., Liu, Q., Liu, T., Sui, Z.: Large language models are not fair evaluators. Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 9440–9450 (2024) https://doi.org/10.18653/v1/2024.acl-long.511