# The Instruction Tax on Factual Recall: Base Models Can Beat Their Instruction-Tuned Counterparts

**Christophe Khalil***
American University of Beirut
cak29@mail.aub.edu

**Aya Mourad***
Sorbonne Université, Paris, France
aya.mourad@sorbonne-universite.fr

## Abstract

We present our solution to the AKBC LM-KBC 2026 Shared Task, where we achieved a score of 0.6836 on the official leaderboard, more than double the baseline of 0.2964. This task requires generating all possible objects for subject-relation pairs using a language model with up to 32 billion parameters, with no retrieval or fine-tuning involved. We show that the prompting methodology is as important as the model architecture: while the *instruction-tuned* model version achieves 0.5681 with its chat interface, the *base* model achieves 0.6836 when prompted using text completion. Thus, instruction tuning decreases factual recall; we refer to this effect as the *instruction tax*, which is caused by the model's weights rather than the prompting method.Our final solution is fairly straightforward: we prompt the base model with some examples from the training data, generate 64–256 outputs, and retain only the repeated ones. Two specialized methods are then applied for each relation: one improves the recall of large answer sets, and the other verifies a person's vitality before predicting their place of death.

## 1 Introduction

Knowledge graphs (KGs) serve as an essential framework for representing factual information in a structured format, enhancing semantic search and reasoning. Manual curation of such frameworks is expensive, and the resulting graphs always contain substantial information gaps (Schneider et al., 2022; Hofer et al., 2024). Although large language models (LLMs) have learned tremendous amounts of factual knowledge implicitly during their pre-training (Petroni et al., 2019), extracting this knowledge is challenging due to prompt sensitivity and hallucinations (Jiang et al., 2020; Zhang et al., 2025).

To evaluate progress on these extraction challenges, the AKBC 2026 Shared Task[1] assesses open-weight LLMs with fewer than 32 billion parameters under a strict *closed-book* setting, where retrieval, external knowledge sources, and fine-tuning are prohibited. The task requires models to predict all valid objects for a given subject–relation pair. For instance, the query *(Albert Einstein, city-of-death)* should produce the answer *Princeton*, whereas the query *(Barack Obama, city-of-death)* should return an empty set because no valid answer exists. This ability to abstain from producing unsupported facts is an essential part of the benchmark, as nearly 20% of the evaluation instances cannot be answered.

Our system is inspired by an empirical finding that we name the *instruction tax*. While *instruction-tuned* models (Ouyang et al., 2022) are generally favored for question answering tasks, our findings show that *instruction-tuned* models can sometimes significantly under-perform *base* models in retrieving facts. For instance, on the area relation, the *base* model queried by simple text completion outperforms the *instruction-tuned* model version by 0.29 through its chat interface, and by 0.30 under the same completion prompt. This gap persists despite prompting the *instruction-tuned* models using completion-style prompts.

Building on this finding, we developed a system based on five methodological principles: (1) choosing the *base* model through a fast probing process that takes about one GPU-hour; (2) prompting the *base* model using few-shot continuations rather than asking questions explicitly; (3) sampling 64–256 times for each row, and keeping only what recurs and returning empty if there is no recurrence; (4) adding a verification step for low-frequency examples in relations with hundreds of possible answers; and (5) breaking down more complicated

*Equal contribution.

[1] https://lm-kbc.github.io/challenge2026/



*Proceedings of the 11th Workshop on Automated Knowledge Base Construction (AKBC 2026)*, pages 518–526
October 28, 2026 ©2026 Association for Computational Linguistics

tasks, such as checking whether the person is dead before predicting the *city-of-death*. Code, prompts, fitted thresholds, and the commands that reproduce each table are available at https://github.com/aiamourad/lm-kbc-2026/.

## 2 Related Work

Language models have been tested for factual recall in multiple studies. The LAMA benchmark was introduced by Petroni et al. (2019) and consists of factual questions answered using the fill-in-the-blank format. Roberts et al. (2020) showed that a fine-tuned model can answer factual questions from its parameters alone, and Brown et al. (2020) showed the same without weight updates, using a few in-context examples as prompts. Still, factual recall is notably less effective for long-tail entities (Sun et al., 2024), making long-tail entities the hard case for our task. The LM-KBC benchmark (Singhania et al., 2022) adds an evaluation setting for factual recall that requires predicting the entire set of answers and compares the predictions against accepted aliases.

Instruction tuning enhances the ability to follow instructions but can impair other abilities (Ouyang et al., 2022), with Zhou et al. (2023) arguing that it alters behavior more than knowledge. This paper quantifies these impacts in terms of factual recall. In addition, our method employs repeated sampling, akin to the self-consistency approach (Wang et al., 2023; Manakul et al., 2023), and examines the trade-off between efficiency and recall in mixture-of-experts models (Fedus et al., 2022).

## 3 AKBC Shared Task & Metrics

### 3.1 Dataset

This task contains six relationships and has 477, 475, and 475 examples in the train/val/test splits, respectively. There are two numerical relationships (`hasArea`, `hasCapacity`) for which the answer is a single number, and it is correct if it differs from the ground-truth answer by no more than ±5%. `zzawardWonBy` may have up to several hundred objects that the model must list to be considered correct. The remaining three relationships (countries (or comparable territories) that share a land border with the subject (`countryLandBordersCountry`), the stock exchange(s) on which the company's shares are publicly traded (`companyTradesAtStockExchange`), the city where the person died (`personHasCityOfDeath`)) have small answer sets that are empty for 18–42% of training rows.

### 3.2 Official Scorer

The official scorer normalizes strings, maps a prediction to an entity if it equals one of that entity's aliases, and maps predictions to entities to maximize the number of matches; row scoring is done independently, and the scores are then averaged. Let $S$ denote the predicted set, $G$ the reference set, and $T$ the number of matches. The row score is:

$$F_1(S,G) = \begin{cases} 1 & S = G = \emptyset \\ 0 & \text{one of S,G empty} \\ 2T/(|S|+|G|) & \text{otherwise.} \end{cases} \tag{1}$$

The first two cases derive from the scoring conventions: abstention on an empty-answer row scores 1.0, whereas guessing on an empty-answer row scores 0. The baseline provided by the organizers never abstains and hence scores 0 on every empty-answer row. The leaderboard uses this scorer, and our validation scores come from running it locally.

## 4 Model Selection

***The knowledge probe*** Which model should we select, and how should we interact with it? To this end, we provide the answers by means of an easy and cheap procedure called the *knowledge probe*. It consists of about 100 examples with known answers. For each of them, we query each of the models under consideration using several phrasings to ask for the corresponding fact and sample their answers 8–12 times. Next, we compute how often each particular answer was generated.

This results in two scores for each candidate model. The first one is called *voted* and stands for the proportion of examples whose most frequently returned answer is the correct one. The second one is the *ceiling*, i.e., the proportion of examples for which the correct answer appears at least once in the generated responses. This score reflects the upper limit of performance under such a probing procedure, since no selection strategy can return an answer that does not occur in the samples.

We probe the two numeric relations, as each sample yields a single number that can be parsed and checked automatically against the 5% tolerance threshold.

**Which model version to ship** Table 1 highlights the results that were derived from around one GPU-

| family | *voted* accuracy base compl. | *voted* accuracy base chat | *voted* accuracy it-tuned compl. | tax Δ | *ceiling* base →chat |
|---|---|---|---|---|---|
| *Families publishing both model versions* | | | | | |
| gemma-3-12b | .720 | .370 | .320 | +.350 | .84→.47 |
| gemma-3-27b | **.880** | .590 | .580 | +.290 | .95→.69 |
| gemma-2-27b | .720 | .640 | .640 | +.080 | .82→.77 |
| Mistral-Nemo-2407 | .710 | .630 | .660 | +.080 | .90→.87 |
| *Instruction-tuned only* | | | | | |
| Mistral-Small-3.2-24B | — | .720 | .690 | — | — |
| Magistral-Small-2506 | — | .610 | .570 | — | — |
| Qwen3-30B-A3B (MoE) | — | .430 | .430 | — | — |
| Qwen3-30B-A3B-Think. | — | .430 | .410 | — | — |

Table 1: Knowledge probe on `hasArea`, 100 validation rows, leading zeros omitted. Each row is one weights family, queried three ways: the base model by completion, and the it-tuned model by chat and by completion. *voted* is the accuracy of the best question form. Δ is the probe tax, base minus chat, on this relation alone; the full-system tax is Table 4. *ceiling* is the share of rows whose correct answer appears in at least one sample, base against chat.

hour of computing effort per model version, and this guided the selection of the model. In the probe we ask for the same fact in several different wordings, such as "The area of $X$ is" or "$X$ covers an area of". For each model version, we then report only the score of its best wording. We pick that wording using the same 100 rows we score on, so these numbers are a little too high. We treat every model version in the same way, but this helps a model more when its wordings give very different results. The table is therefore best used to rank the model versions, not to predict what one fixed prompt would score. Dense models outperformed the mixture-of-experts (MoE) model across all the experiments. While the evaluation counts an MoE model by its total number of parameters, not all of these parameters get activated per token, and the performance appears to depend on the amount of such computation. The reasoning-tuned model does not provide any noticeable improvement.

***The instruction tax*** We observe a discrepancy between the scores of the base 0.880 and instruction-tuned 0.59 version of the gemma-3-27b model. To analyze whether the following is resulting from the nature of the prompt, we prompt the instruction-tuned model using the same completion prompt (compl. under it-tuned in Table 1) used for the base model version. Even with an identical query, it fell short at 0.580, and the sample ceiling falls from 0.950 to 0.690. This gives us a clearer image that the tuned model forgot facts it learned.

To check how consistent this pattern is, we test three additional pairs (gemma-3-12b, gemma-2-27b, and Mistral-Nemo-2407). In all eight tests, the base model version is at least as good as the instruction-tuned model version on both numerical relations. This appears to be a feature not inherent to instruction tuning per se, but dependent on the particular tuning run. Inside the Gemma family, the difference is 0.290 for gemma-3-27b and 0.350 for gemma-3-12b, but only 0.080 for gemma-2-27b, even though it is the same size as gemma-3-27b. In other words, with one generation less, the same tuning carries only a quarter of the cost. Mistral-Nemo-2407 also shows this small difference of 0.080.

The key difference between the two cases lies in the ceiling. With a high tax, performance on the matched completion prompt becomes very poor: dropping from 0.950 to 0.690 in gemma-3-27b, and from 0.840 to 0.470 in gemma-3-12b. With a low tax, the ceiling remains almost intact, dropping merely from 0.820 to 0.770 in gemma-2-27b, and from 0.900 to 0.870 in Mistral-Nemo-2407. It seems, therefore, that a high tax results in the loss of some answers from the sampling distribution, whereas a low tax results in poor answer selection from the remaining sample. Since both can be quantified using the same one-hour probe, the distinction can be made prior to committing to a model version.

As shown in Section 6, the probe order matches the order of the official submission runs, suggesting that the probing evaluation serves as an effective proxy for full submissions. The final system employs the base model version, gemma-3-27b-pt (Gemma Team, 2025), for all six relations.

## 5 Methodology

For inference, we deploy only one model (*gemma-3-27b-pt*) for the whole set of tasks. This model has 27.43 billion parameters, obtained precisely by counting the released weights rather than relying on the model's name, which may be incorrect. Within the task's 32-billion-parameter budget, it is impossible to deploy any other useful model in parallel with it. Inference is provided with the help of vLLM (Kwon et al., 2023) using two H100 GPUs and thus is performed for the whole test split in just a few minutes.

| Relation | Completion format | $k$ | $n$ |
|---|---|---|---|
| hasArea | The total area of {s} is {v} square kilometres. | 5 | 64 |
| hasCapacity | The seating capacity of {s} is {v}. | 5 | 64 |
| countryLandBordersCountry | Countries sharing a land border with {s}: {v} | 12 | 128 |
| companyTradesAtStockExchange | {s} stock exchange listings: {v} | 12 | 256 |
| personHasCityOfDeath | City where {s} died (or none if alive as of 1 July 2026): {v} | 12 | 128 |
| awardWonBy | Recipients of {s}: {v} | 4 | 64 |

Table 2: Completion formats and sampling budgets ($k$: exemplars, $n$: samples per row); {s} is the subject and {v} the gold value or none.

**Never ask a question.** For each relation, there is one completion template (see Table 2). A prompt is composed of a few examples selected from the training dataset and ends with an unfinished example that corresponds to the subject of the query:

```
The total area of Lake Victoria is 68800
square kilometres.
The total area of Bodensee is 536 square
kilometres.
The total area of Vygozero is
```

Such a formulation will naturally encourage the continuation of the observed pattern rather than an interpretation of the input as an instruction. There is no chat template and no task-specific instruction, and the model receives only the text in the exact form it should produce.

There are two crucial design decisions here. First of all, the numerical examples are chosen in such a way that they cover the whole distribution of numbers in the training dataset, including one sample from each quintile. This will prevent a bias towards a certain numerical range in the prompt. Second, examples with an empty response in the case of set-valued relations make up about one third of the exemplars and are marked as "none".

**Ask many times, keep what repeats.** Each row is sampled $n$ times with different random seeds, and every output is truncated at the first line break. The temperature is 0.7 throughout. The exact value matters little: between 0.3 and 1.3 the differences stay within noise once the decision thresholds are refitted. Only very low temperatures do real damage, making the 64 samples nearly identical, collapsing the process to a single completion and removing the small benefit of aggregation.

We then parse the outputs into the required format for each relation type: a single numeric answer for numeric relations, and a comma-separated list for set-valued relations, where the output "none" denotes the empty set. Prior to aggregation, the responses are normalized in the same way as in the evaluation script: string-valued answers are merged according to the scorer's normalization rules, and numeric answers are grouped using the scorer's 5% tolerance.

For numerical relations, denote the parsed samples as $v_1, \ldots, v_m$, where $m \leq n$ because not every output parses into a number. For each candidate $c$, the cluster $C(c)$ is formed by all those samples lying within 5% of $c$, and the prediction is the median of the biggest cluster:

$$\hat{v} = \operatorname{median} C(c^*), \quad c^* = \arg\max_c |C(c)|, \quad (2)$$

where $C(c) = \{\, v_i : |v_i - c| \leq 0.05\, c \,\}$. This can be considered as majority voting modified to match the evaluation function: since the scorer allows any prediction that is within 5 percent of the reference value, votes are counted on tolerance clusters and not on numeric equality.

For relations with sets, we define $A_1, \ldots, A_m$ as the sets of answers after parsing. Support $s(a)$ for a candidate answer $a$ is defined as its frequency of recurrence, that is, the fraction of the $A_j$ that contain $a$:

$$s(a) = \tfrac{1}{m}\big|\{j : a \in A_j\}\big|, \quad S = \{a : s(a) \geq \tau_r\}. \quad (3)$$

The threshold $\tau_r$ is the sole parameter fitted for each relation independently. Instead of picking it manually, we pick the one that achieves the best official score on the training examples, and re-fit it each time either the number of samples $n$ or the number of exemplars $k$ is modified. These thresholds are more conservative for those relations where misprediction carries a high cost. In particular, for personHasCityOfDeath, the system will predict the city only if at least 40% of the 128 samples vote for it. For those relations having a bigger reference set, the fitted threshold is typically smaller. Abstention is a natural consequence of the same approach, since if none of the candidates achieves the needed recurrence rate, the predicted set is empty.

**Verifying rare candidates for `awardWonBy`.** The relation `awardWonBy` is especially challenging because its answer set can have hundreds of names. In such a situation, it is difficult to differentiate the genuine candidate, which only occurs three times among 64 samples, from the hallucinated candidate by recurrence alone. For this reason, we employ the model as a second-stage verifier using completion. We first present the model with ten verification examples from the training set that are evenly divided into positive and negative examples (e.g., `Did Barack Obama get the Nobel Peace Prize? yes` and `Did Marie Curie get the Turing Award? no`), and then pose the same question to the model regarding the candidate. We sample the verification question eight times, and let $q(a)$ be the ratio of samples answered with yes. We then add back the candidates whose support is at least $2/64$ and whose $q(a) \geq 0.75$. The two threshold values are chosen based on training examples. In the verification process, true positives are separated from false positives by means of the Area Under the Curve (AUC), which has a value of 0.86 on training candidates (0.75 on validation candidates). Lastly, we perform a non-neural clean-up procedure to combine duplicate names such as "Robert Caro" and "Robert A. Caro". Without performing this step, the scorer would consider both entries to be different predictions.

**Is the person dead or alive?** Approximately 40% of the death-relation rows refer to people who are still alive, and the correct answer in such cases is the empty set. We took a two-step approach in which we identify whether a given subject is dead or alive. If the person is alive, we ask the model to assign the empty set. On the other hand, if the model is confident that the person is indeed dead, we ask it to recall the location of their death.

Information about whether the person is alive or dead is far more prevalent than information about where they died. The location of the death is tail knowledge, while the person's living status is more widespread. We exploit this difference as an initial filter, which we call the *existence check*: it predicts whether an object exists for the row at all, rather than which object it is. For every query, we take 16 samples of a separate completion, built from ten exemplars evenly split between living and dead subjects of the training split, of the form `As of 1 July 2026, {s} is alive/dead`, followed by the query subject. If all 16 samples predict that the person is alive, then the answer must be the empty set.

## 6 Results

Table 3 gives the final scores and Table 4 the comparisons across models and interfaces, all from official submissions. Relations contribute 100 rows each except `awardWonBy` (10), `countryLandBordersCountry` (67) so "All relations" averages over 475 rows, not six scores; the last row scores the abstention decision, with the empty set as the positive class.

| | val | test (official) | | |
|---|---|---|---|---|
| Relation | F1 | P | R | F1 |
| `awardWonBy` | 0.266 | 0.389 | 0.343 | 0.338 |
| `companyTradesAtStockExchange` | 0.765 | 0.893 | 0.932 | 0.872 |
| `countryLandBordersCountry` | 0.989 | 0.962 | 0.976 | 0.958 |
| `hasArea` | 0.900 | 0.820 | 0.820 | 0.820 |
| `hasCapacity` | 0.299 | 0.316 | 0.316 | 0.316 |
| `personHasCityOfDeath` | 0.550 | 0.900 | 0.630 | 0.570 |
| **All relations** | **0.669** | **0.759** | **0.712** | **0.684** |
| Abstention decision | — | .722 | 0.892 | 0.798 |

Table 3: Final system. Test columns are official; validation is the same scorer run locally. Formats and thresholds are fitted on training rows only.

Firstly, there is an *instruction tax*, which is significant, especially when it comes to relations that rely heavily on exact recall (Table 4). Within one weights family, using the *base* model for completion instead of the instruction-tuned model via the chat interface increases the score by $+0.116$. The greatest improvement can be seen in the numeric relations, namely area ($+0.203$) and capacity ($+0.156$). In fact, the *base* model outperforms the instruction-tuned model on every single relation.

The comparison only makes sense if the instruction-based model is a strong one. To ensure this, we created a detailed prompt with a few-shot examples and challenged the model with yes-or-no questions for each query. More details are provided in Appendix B. Even after giving the instruction-tuned system all these additional advantages, the simpler completion-based system still performed better on every relation. Our findings imply a strong difference between the *base* and *instruction-tuned* models when it comes to factual knowledge recall: the facts recoverable by the *base* model become extremely rare in the output of the *instruction-tuned* model.

Secondly, the ranking method of Table 1 has predicted each ordering in the official submission runs

| Relation | official test F1 | | |
|---|---|---|---|
| | it, chat | pt, compl. | Δ |
| hasArea | 0.617 | 0.820 | +0.203 |
| hasCapacity | 0.160 | 0.316 | +0.156 |
| companyTradesAtStockExchange | 0.784 | 0.872 | +0.088 |
| personHasCityOfDeath | 0.480 | 0.570 | +0.090 |
| awardWonBy | 0.271 | 0.338 | +0.067 |
| countryLandBordersCountry | 0.946 | 0.958 | +0.012 |
| **All relations** | 0.568 | **0.684** | +0.116 |
| Mistral-Small-3.2-24B, chat | | 0.613 | |
| Mistral-Small-3.2-24B, compl. | | 0.615 | |
| Magistral-Small-2506, compl. | | 0.578 | |
| Qwen3-30B-A3B (MoE), chat | | 0.534 | |
| Qwen3-30B-A3B (MoE), compl. | | 0.551 | |
| Qwen3-30B-A3B-Think., compl. | | 0.510 | |
| task baseline | | 0.2964 | |

Table 4: The *instruction tax* at full-system scale, all scores official. Upper block: the gemma-3-27b family, gemma-3-27b-it by chat against gemma-3-27b-pt by completion, with Δ the second minus the first for each relation. Lower block: model versions with no base sibling, so there is nothing to subtract; each row is a single score, marked *chat* when the model was asked through its chat interface and *compl.* when it was given the completion prompt instead.

accurately, as the lower block of Table 4 shows. The dense models were ranked ahead of the MoE model (0.568 and 0.613 against 0.534) under the same instruction-tuned system, and the *base* checkpoint was ranked ahead of all of them at 0.681. The point is that the decisions which are key to achieving the highest score, model and interface, can be determined using only one GPU hour of computing time. Thirdly, sampling should be increased until little or nothing is gained, and this appears to be contingent on the relation. The official score increased from 0.6218 with $n$=8 samples to 0.6292 with $n$=24 samples, and again to 0.6604 with $n$=64 samples, the latter one being computed using the company format in Table 2. More samples were useful for some relations; for set relations, the gains continued until $n$=128, while for the company relation, the optimal number was found to be $n$=256. The verification rescue contributed an additional +0.027 on its relation, and the existence check contributed +0.030 on the death relation, which gave 0.6809. Doubling the samples for the numeric relations ($n$=128) did not help, so $n$=64 already saturates them.

Last, much of the gain comes from knowing when to say nothing (last two rows of Table 3). Empty exemplar lines plus the training-fitted threshold give an abstention F1 of 0.773; the existence check raises it to 0.798 and death precision to 0.900. The check catches an error recurrence cannot: the model naming a city, consistently, for someone still alive.

## 7 Conclusion

The design of our system rests on one insight: when it comes to factual recall, instruction tuning is not worth the expense. On the same parameters, gemma-3-27b-it scores 0.568 through its chat interface, while gemma-3-27b-pt scores 0.684 by completion. The rest follows: a one-GPU-hour probe selects the model version and the interface, each row is sampled between 64 and 256 times from the completion prompt, and only those responses above a fitted threshold are retained, making abstention the norm rather than the exception. The final score is 0.6836, against 0.2964 for the task's baseline. Two takeaways should apply elsewhere: the model version and the interface need to be tested, and where the best answer is often empty, knowing when not to answer helps almost as much as knowing more.

## Limitations

The instruction tax is calculated on four (base, instruct) pairs. Officially, changing the weights alone costs 0.085 for gemma-3-27b and gemma-3-12b, whose sampling ceiling collapses, but 0.016 for Mistral-Nemo-2407 and 0.000 for gemma-2-27b, whose ceiling holds: the size of the tax depends on the tuning process, and four families are not enough to determine its applicability. The system is also quite costly, taking 64–256 generations to process each row, while the instruction-tuned baseline uses only one, and it is not zero-shot: the prompts are constructed from the training rows with labels, and the thresholds have to be recalibrated every time either $n$ or $k$ changes. The verification rescue and the existence check were each crafted by hand for a single relation.

## Acknowledgments

This work was performed using HPC resources from GENCI-IDRIS (Grant 2025-AD011016170).

## References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda

Askell, et al. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33.

Krzysztof Dembczyński, Willem Waegeman, Weiwei Cheng, and Eyke Hüllermeier. 2011. An exact algorithm for f-measure maximization. In *Advances in Neural Information Processing Systems (NIPS)*.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23.

Gemma Team. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.

Marvin Hofer, Daniel Obraczka, Alieh Saeedi, Hanna Köpcke, and Erhard Rahm. 2024. Construction of knowledge graphs: Current state and challenges. *Information*, 15(8):509.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles (SOSP)*.

Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. 2023. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Phillip Schneider, Tim Schopf, Juraj Vladika, Mikhail Galkin, Elena Simperl, and Florian Matthes. 2022. A decade of knowledge graphs in natural language processing: A survey. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 601–614.

Sneha Singhania, Tuan-Phong Nguyen, and Simon Razniewski. 2022. LM-KBC: Knowledge base construction from pre-trained language models. In *Proceedings of the Semantic Web Challenge on Knowledge Base Construction from Pre-trained Language Models (LM-KBC @ ISWC)*, CEUR Workshop Proceedings.

Kai Sun, Yifan Ethan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. 2024. Head-to-tail: How knowledgeable are large language models (LLMs)? A.K.A. will LLMs replace knowledge graphs? In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations (ICLR)*.

Nan Ye, Kian Ming A. Chai, Wee Sun Lee, and Hai Leong Chieu. 2012. Optimizing f-measures: A tale of two approaches. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. 2025. siren's song in the ai ocean: A survey on hallucination in large language models. *Computational Linguistics*, 51(4):1373–1418.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2023. LIMA: Less is more for alignment. In *Advances in Neural Information Processing Systems*, volume 36.

## A Base and Instruction-Tuned Checkpoints, End to End

In Table 5, the whole system developed in Section 5 is run on each family releasing both checkpoints, with one call per checkpoint, and is evaluated on the official test set. Everything else in the pipeline, the sampling budget, and the training process is exactly the same for the eight runs; the only difference is in the weights. The probe columns reproduce Table 1. Eight runs cost 7 GPU-hours on two H100 GPUs.

The cost of each of these jobs is given in Table 6. All jobs use two H100 GPUs, and thus the number of GPU hours is double the wall time. A job

| | official test | | | probe | |
|---|---|---|---|---|---|
| family | base | tuned | Δ | Δ | ceiling |
| gemma-3-27b | .6836 | .5932 | +.0904 | .290 | .95→.69 |
| gemma-3-12b | .6018 | .5166 | +.085 | .350 | .84→.47 |
| Mistral-Nemo | .5881 | .5724 | +.016 | .080 | .90→.87 |
| gemma-2-27b | .6190 | .6189 | +.000 | .080 | .82→.77 |

Table 5: The complete system on four base/instruct pairs, official test, leading zeros omitted. Δ: cost of the tuned weights; *ceiling* from Table 1.

consists of both validation and testing, including the threshold fitting discussed in Section 5; only gemma-3-27b-pt reuses the thresholds already fitted for it and does not need to fit them again, which makes it the cheapest one. The most expensive is gemma-2-27b due to the inefficient attention implementation.

| | probe | | full system | |
|---|---|---|---|---|
| checkpoint | wall | GPU-h | wall | GPU-h |
| gemma-3-27b-pt | 0:03 | 0.1 | 0:21 | 0.7 |
| gemma-3-27b-it | 0:09 | 0.3 | 0:29 | 1.0 |
| gemma-2-27b | 0:04 | 0.1 | 0:38 | 1.3 |
| gemma-2-27b-it | 0:05 | 0.2 | 0:23 | 0.8 |
| gemma-3-12b-pt | 0:08 | 0.3 | 0:29 | 1.0 |
| gemma-3-12b-it | 0:12 | 0.4 | 0:22 | 0.7 |
| Mistral-Nemo-Base-2407 | 0:07 | 0.2 | 0:25 | 0.8 |
| Mistral-Nemo-Instruct-2407 | 0:10 | 0.3 | 0:24 | 0.8 |
| Mistral-Small-3.2-24B | 0:06 | 0.2 | 0:33 | 1.1 |
| Magistral-Small-2506 | 0:05 | 0.2 | 0:27 | 0.9 |
| Qwen3-30B-A3B | 0:06 | 0.2 | 0:31 | 1.0 |
| Qwen3-30B-A3B-Think. | 1:01 | 2.0 | 0:28 | 0.9 |

Table 6: Wall time and GPU-hours per checkpoint; two H100s per job. A full run covers validation and test with fitting. Below the rule: no base sibling.

## B The Instruction-tuned Comparator in Full

This appendix outlines the instruction-tuned system deployed for the "it, chat" row in Table 4. Of all the instruction-tuned models we experimented with, this is our best system, which scored 0.5681 with gemma-3-27b-it and 0.6127 with Mistral-Small-3.2-24B. The second model achieved 0.6161 when the border rules discussed in Appendix B were applied. This system utilizes a very simple process that entails generating the potential answers, evaluating them separately, calibrating the scores, and choosing the highest-scoring combination.

**Propose.** The relation question is asked of the model for each row $n$ times. Instead of using temperature to generate different outputs, we provide each sample with a new *approach instruction*, which is a one-sentence hint on how to solve the problem, e.g., "Walk around the country's borders clockwise and name its neighbours" or "First figure out whether the person was alive in 2026." In practice, it results in more productive disagreements among the samples compared to changing the temperature. Constrained decoding is used to make sure that the outputs remain in the same JSON format. Each of the prompts contains the definition of the relation given in the task and six example rows from the training set.

**Group.** Identical to the main system: the scorer's own normalisation for strings, 5% tolerance clusters for numbers.

**Verification.** Each candidate solution is then verified individually. The model receives a statement of the form "X is one of the correct answers to . . . ", and replies with either "Yes" or "No". Rather than sampling this reply, we take the log-probabilities of the first token: the score $q$ is the probability mass on the tokens spelling "Yes", divided by the mass on "Yes" plus "No". The value of $q$ obtained in this way is not always equally robust for different relations. For instance, it achieves an AUC of 0.931 for `awardWonBy` while reaching 0.450 for the border relation, a result inferior even to the random chance baseline. Owing to such variations, we do not fix any threshold for $q$; rather, we consider it to be one signal among others forwarded to the next step.

**Calibrate.** For each relation $r$, a logistic regression fitted using the training rows maps the signals to a probability. Let $f_i$ be the recurrence probability for candidate $i$, $q_i$ its verification probability, $\rho_i$ its rank among the row's candidates in $(0, 1)$ space, and $\pi_i = 1/(1 + C)$ with $C$ being the number of candidates; this is a five-parameter model per relation (four weights and a bias), fitted with an $L_2$ penalty.

**Decode.** The system outputs the set having the highest expected row score based on these probabilities. Rank the candidates by $p_i$ and let $S_k$ contain the top $k$; then only the $C+1$ sets $S_0, \ldots, S_C$ have to be compared (Dembczyński et al., 2011; Ye et al., 2012):

$$S^* = \underset{k \in \{0,\ldots,C\}}{\arg\max} \; \mathbb{E}[F_1(S_k, G)] \,. \tag{4}$$

In the case when the real set can have at most one member, the candidates are mutually exclusive, and

the expectation admits a closed-form solution; otherwise, they are assumed to be independent, and the expectation is computed using dynamic programming on the $p_i$. The null set is the special case of $k{=}0$ and therefore does not require any threshold. In the case of `awardWonBy`, the model also infers the number of recipients and thus accounts for correct answers that are not in the candidate pool.

**The Border Rules.** The border relation is characterized by two features which a purely statistical pipeline cannot take advantage of: the borders are symmetrical, and each training/validation instance is a full list of its subject's neighbours. Representing by $\mathcal{G}(X)$ the reference list of neighbours of a country $X$ that is a subject of the provided data, and by $\mathrm{dom}(\mathcal{G})$ the set of such subjects,

$$Y \in \mathcal{G}(X) \;\Rightarrow\; \mathit{border}(Y, X), \tag{5}$$

$$X \in \mathrm{dom}(\mathcal{G}) \wedge Y \notin \mathcal{G}(X) \;\Rightarrow\; \neg\, \mathit{border}(Y, X). \tag{6}$$

Rule 5 adds missing neighbours (checked against the validation gold, the rule proposed 47 neighbours and 46 were correct; during test, on the Mistral-Small-3.2-24B chat system, it added five cases like The Gambia for Senegal); Rule 6 deletes predicted neighbours which contradict the complete reference (during test, it deleted four cases like Sweden–Denmark, linked by a bridge not counted in the reference).

Both of them use reference data only, and predictions by the system itself are not considered evidence at all. In the case of the Mistral-Small-3.2-24B instruction-tuned system, the rules increased borders from 0.9487 to 0.9729 official and from 0.6127 to 0.6161 total. In the completion system, in which borders achieve 0.989 on validation with recall 1.000, the rules failed to pass the validation gate and were not used (Section 6).