

MAVE: Multi-Angle Voting for Emission in Closed-Book Knowledge Base Construction

Jonas Kunze and Andreas Both

 Web & Software Engineering (WSE) Research Group
Leipzig University of Applied Sciences (HTWK Leipzig)
Leipzig, Germany
{jonas.kunze, andreas.both}@htwk-leipzig.de

Abstract

Pretrained language models hold factual knowledge in their parameters, and the LM-KBC challenge measures how much of it can be turned into a knowledge base: given a subject and a relation, a system must return every object that completes the pair, and nothing at all when there is none. The 2026 edition forbids any lookup and caps all models used together at 32B parameters. Because the model may be chosen freely for the first time, a system has three things to decide: which model to run, how to put the question to it, and how to choose an answer from what comes back. We propose MAVE, which puts each item to one open-weight 24B model as several different questions and gives each question one vote. Each of the six relations applies its own rule for how much agreement it takes before an object is emitted. It reaches an official test macro-F1 of 0.6908, which places it second on the final leaderboard.

1 Introduction

Reading a pretrained language model as a knowledge base has been possible since Petroni et al. (2019) recovered relational facts from masked models, and the LM-KBC challenge has measured how far it carries since 2022 (Singhania et al., 2022). How far it carries has depended on what each edition allowed: with a lookup in Wikidata the 2024 winner reached an average F1 of 0.9224, and without it the 2025 winner reached 0.4439 (Bara, 2024; Albert-Roulhac and Zouaq, 2025). The 2026 edition keeps that closed book and replaces the fixed model of 2025 by a budget of 32B over open-weight models, which leaves a system three things to get right: which model to run, how it asks, and what it does with the answers.

A system that asks in many ways collects more candidate objects than it can emit, and it has to decide which of them to keep.

This paper therefore asks:

RQ With retrieval forbidden and the total LLM parameter count capped, how can the factual knowledge held in a pretrained model be extracted into a knowledge base?

Our answer is MAVE (Figure 1).¹ It puts each item to the model through several questions, lets each question cast exactly one vote whatever its number of calls, and decides per relation how much agreement a candidate needs.

2 Task and Data

An item is a subject entity together with a relation, for instance “Portugal” and `hasArea`, and the expected answer is the complete set of objects for that pair, taken from what the model memorized during pretraining (Kalo et al., 2026). That set may hold one object, many, or none at all.

The six relations. They differ in the shape of their answer, and Table 1 gives their sizes alongside the score our system reaches on each. Two ask for a number, the surface area of a geographic entity (`hasArea`) and the capacity of a venue (`hasCapacity`); one for at most one entity, the city in which a person died (`personHasCityOfDeath`); three for a set, the countries sharing a land border with a country (`countryLandBordersCountry`), the stock exchanges a company is listed on (`companyTradesAtStockExchange`), and the people who received an award (`awardWonBy`), the extreme case with 111.1 objects per item on average.

Scoring. The official score is the mean of the per-item F1 over all items of a split, so every item weighs the same, regardless of which relation it belongs to. Within an item, the predicted set is matched one-to-one against the gold set, and a prediction matches an object when it equals any of

¹The code, the resolved configuration of every run, and the verbatim prompts are available at <https://github.com/WSE-research/MAVE-LM-KBC-2026>.

Relation	Answer	Items			Empty gold	F1, MAVE		
		Train	Val	Test		Train	Val	Test
countryLandBordersCountry	set	67	68	67	22%	0.99 $\pm$ 0.001	0.98 $\pm$ 0.002	0.99
companyTradesAtStockExchange	set	100	100	100	36%	0.73 $\pm$ 0.012	0.78 $\pm$ 0.010	0.85
hasArea	number	100	100	100	0%	0.83 $\pm$ 0.012	0.85 $\pm$ 0.008	0.84
personHasCityOfDeath	one entity	100	100	100	41%	0.52 $\pm$ 0.025	0.61 $\pm$ 0.012	0.59
awardWonBy	set	10	10	10	0%	0.51 $\pm$ 0.006	0.24 $\pm$ 0.008	0.36
hasCapacity	number	100	97	98	0%	0.25 $\pm$ 0.006	0.26 $\pm$ 0.020	0.31
Macro-F1		477	475	475		0.637 $\pm$ 0.005	0.670 $\pm$ 0.007	0.6908
Organizers' baseline								0.3085

Table 1: The six relations, sorted by our F1 on the test split. “Answer” is the shape the task asks for, and every one of them may also be empty: “Empty gold” is the share of items where it is. Test scores are the organizers’ scorer on our submission; train and validation of our own evaluation against the released gold. Each $\pm$ is the standard deviation over five repeated draws of the same system.

that object’s aliases after normalization; on the two numeric relations, a prediction counts as correct within a tolerance of five percent. An item whose gold answer is the empty set scores 1.0 exactly when the prediction is empty as well, and 0.0 for any prediction.

Three degrees of freedom. The 2026 edition is the first in which the model may be chosen freely (Section 3), so all three questions named above are open at once. The first is settled in Section 5 by measurement, and the rest of the paper works on the other two.

The rules. Two constraints shape every design decision in this paper. The task is closed book, so retrieval and any lookup in an external corpus or knowledge base are forbidden at inference time, and fine-tuning of any kind is excluded. The parameter budget is 32B summed over every neural component used at inference, counted on the published total of each model, which quantization does not reduce and a mixture-of-experts model does not escape. What the rules leave open is multi-step inference and post-processing without a neural component. The system of Section 4 works entirely within that space.

3 Related Work

What a model has memorized. Closed-book question answering established that a model can answer from its parameters alone and that how much it answers grows with its size (Roberts et al., 2020). What it has kept is uneven, and the unevenness is not random: a fact is recalled roughly in proportion to how often documents about it appeared in pre-training, so entities in the tail of that distribution are missing however large the model (Kandpal et al.,

2023), and the popularity of the subject predicts when the parameters suffice and when they do not (Mallen et al., 2023). The six relations of this task differ along exactly that axis.

Four editions, four sets of rules. The challenge turned the question into a shared task with a fixed dataset, metric, and leaderboard (Singhania et al., 2022); 2026 is its fifth edition. The first two placed no restriction on the model (Singhania et al., 2022; Kalo et al., 2023), the 2024 edition capped it at ten billion parameters and left retrieval open (Kalo et al., 2024), and the 2025 edition removed retrieval and fixed one model for every participant (Kalo et al., 2025), which is where the two winning scores (cf. Section 1) come from.

Recurring techniques. Four moves are used by more than one system. The same question is put to the model in more than one wording (Alivanistos et al., 2022; He and Razniewski, 2025); chain-of-thought reasoning is added as a level of a prompt taxonomy (Bara, 2024), as reasoning paths distilled from a larger model (Albert-Roulhac and Zouaq, 2025), and as trainable prompts inside the chain (Creo et al., 2025); the model is asked to check its own candidates before they are emitted (Alivanistos et al., 2022; Clay et al., 2025); and the empty answer is treated as a case of its own rather than as a failure to answer (Alivanistos et al., 2022; Das et al., 2024). Two moves appear once each and are closest to ours: sampling one prompt twenty times and aggregating per relation (Albert-Roulhac and Zouaq, 2025), and having the model write a description first and read the answer out of it (He and Razniewski, 2025).

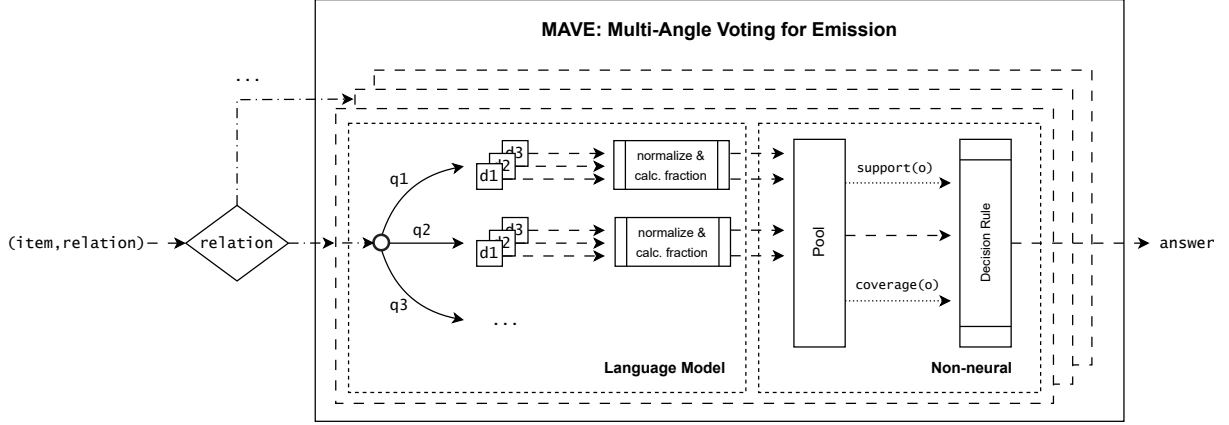


Figure 1: The pipeline for one item. The relation selects both the questions and the decision rule. Each question q_i is put to the model k times; its answers d_j are normalized and collapsed into one fraction per candidate object. The support and the coverage of each candidate o are then computed over the pool, and both go into the decision rule, which picks the final answer.

4 Method

MAVE asks the same fact in several different ways and lets the answers decide among themselves. Two rules govern the pool of candidates that comes back. A formulation counts once, however often it was asked, so a question put twenty times cannot outvote one put once. A formulation that recalls nothing abstains, so ignorance is never evidence against a candidate. What is then left to decide is a single quantity: how much agreement an object needs before it is emitted. Section 4.2 settles it.

The five steps below describe how the system turns a subject and a relation into an answer (Figure 1).

Step 1: Ask. Each relation has its own set of questions, between two and five of them, and they are not rewordings of one another but different text genres, listed in Table 2. Every question is put to the model k times. At $k = 1$ the call is greedy; for $k > 1$ the calls are sampled at a temperature above zero, so the k answers may differ.

Step 2: Normalize. The model maps the surface forms of one object onto a canonical form, so that “NYSE” and “New York Stock Exchange” become one candidate and two spellings the official scorer would accept as equal and never split a vote. The spelling finally printed is the one that occurred most often.

Step 3: From answers to fractions. The k answers to one question are first collapsed into a single fraction per candidate, namely how often that

question produced it, defined as

$$\text{share}_q(o) = \frac{|\{d \in D_q : o \in d\}|}{|D_q|}, \quad (1)$$

where D_q are the k answers to question q . A question asked once contributes 0 or 1; a question asked twenty times contributes anything in between. It is these fractions, one per question and candidate, that go into the pool.

Step 4: Two numbers over the pool. Let Q be the questions of the relation. For every candidate o , the pool yields exactly two quantities. The support averages what the questions gave it,

$$\text{support}(o) = \frac{1}{|Q|} \sum_{q \in Q} \text{share}_q(o), \quad (2)$$

while the coverage ignores how often a question produced the candidate and counts only how many questions produced it at all,

$$\text{coverage}(o) = |\{q \in Q : \text{share}_q(o) > 0\}|. \quad (3)$$

The denominator of the support is the number of questions asked, not the number that answered: a question that recalled nothing lowers the support of every candidate, but it is never a vote against one.

Step 5: Decision rule. Every relation has its own decision rule, which reads the pool together with these two numbers. The shape of the answer the task asks for, a number, at most one entity, or a set, narrows down which rules are possible, and within that the rule is chosen per relation.

4.1 Questions and Decision Rule of Each Relation

The six paragraphs below sketch each relation; the rules in full, with every prompt and every parameter, are in the repository.¹

Countries that share a land border. Three questions, emitted at a coverage of two. The third asks which landmass the subject sits on and who else is on it, and it has a second use: a country the model places alone on an island can have no land neighbors whatever the other two just said, so that answer can remove a neighbor, never add one.

Stock exchanges a company trades at. Five questions in five different text genres, emitted at a coverage of two. Several of them also let the model state that the company is not listed at all; if more answers deny a listing than affirm one, nothing is emitted, while an answer that recalls nothing counts for neither side.

The surface area of a geographic feature. Three questions, two of which first have the model write a passage and then read the number out of it. Here support decides: the system slides a window of five percent, the scorer’s own tolerance, over the numbers in the pool and emits the center of the window whose numbers carry the most support.

The city in which a person died. Four questions of one shape, each having the model write a passage and then read the city out of it. The city with the most support is emitted only when its coverage is more than half of the four, and nothing otherwise, which matters because for 41% of the items the gold answer is empty and an empty prediction scores a full point.

The people who received an award. Four questions that cut the same question differently, emitted at a coverage of two. Cutting is the only way to ask at all here, because the gold holds 111 names per item on average, far more than any single answer returns.

The capacity of a venue. Two questions, each asked twenty times at a temperature of 1.0. Numbers are weighed as for the area, and this is the one relation where the system also decides how many numbers to emit: a second one caps the item at 0.667, and it is emitted when its support is at least half that of the first.

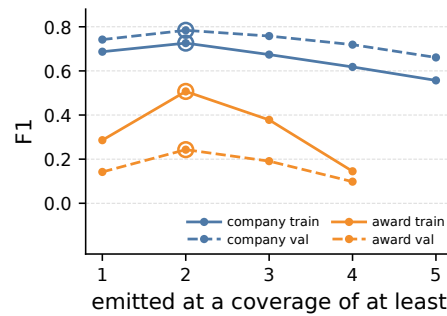


Figure 2: F1 of the two set-valued relations that can reach more than two agreeing questions, at every coverage level, with everything else held fixed. The rings mark the level the system uses.

4.2 How Much Agreement a Candidate Needs

The three relations that answer with a set leave one quantity open: how many of their questions must have named a candidate before it is emitted. Emitting at a lower coverage takes in more candidates, which raises recall and lowers precision inside the same item; emitting at a higher one does the reverse. Both directions cost score, so the level is settled by measurement on the labelled splits, with everything but the level held fixed. Two is best for `companyTradesAtStockExchange` and `awardWonBy` on train and on validation alike, and the fall-off on either side is steep (Figure 2). `countryLandBordersCountry` has only two questions feeding its pool and therefore no level above two; there one scores 0.017 higher than two on each split, on a relation that already reaches 0.99, so two is applied throughout.

5 Experimental Setup

Which model. We settled the choice of model by measurement. Running the same pipeline on three open-weight models inside the budget, with the questions, draw counts, seeds, temperatures, decision rules, and post-processing all held fixed, gives a validation macro-F1 of 0.6699 for `mistral-small-3.2` at 24B, 0.5645 for `qwen3.6` at 27B, and 0.5293 for `gemma-4` at 31B. The distance is large and does not follow the parameter count, since the 31B model is the weakest of the three.

That comparison is unfair in one direction: every question, draw count, and decision rule was chosen while `mistral-small-3.2-24b-instruct` was in place, so it says how much of our score the backbone carries and not which model is strongest when each is tuned for itself. Both of the others did hold

that position earlier in this project, and gemma-4 was re-tuned rather than only swapped in; its emission thresholds refit relation by relation and its numeric questions rebuilt four ways; it still lost on every relation that asks for an exact figure about an entity that is not famous. Every prompt therefore goes to mistral-small-3.2-24b-instruct, served through a commercial provider with its reasoning mode disabled.

The budget also allows a second model of up to 8B besides the 24B one, and for the three, we tried it, and it did not pay off. Switching llama-3.1-8b off was better or equal on both relations it had served and both splits; ministral-8b lost on both labelled splits, and qwen3-8b, the model the 2025 edition prescribed, reached 0.567 as a partner where the pair it was compared against reached 0.582. A smaller model names few objects that the larger one misses, so it corroborates what is already there and adds false positives of its own.

Temperature. The first draw of every question is greedy and the further draws are sampled at temperature 0.7, where the question varies a judgment, and at 1.0, where it widens a pool of numbers. Seeds are fixed per question, and every response is cached, so re-aggregating a decision over an unchanged pool of draws is exactly reproducible. Across the six relations, the shipped configuration issues 32.0 calls to the model per item.

How configurations were selected. Every design decision was made on the training and validation splits; the test split ships without gold answers, so every test number in this paper comes from the organizers’ scorer on a submission of ours.

6 Results

MAVE reaches an *official test macro-F1* of 0.6908, scored by the organizers on the submission platform. That is second on the final leaderboard, behind a 31B model at 0.7060. The organizers’ baseline, a single direct generation per item, reaches 0.3085 on the same split. Table 1 gives the score per relation on the three splits. The six are taken one at a time below against Table 2, which counts for each question the share of the gold it names (recall) and the share of what it names that is gold (precision).

6.1 The Six Relations

The relation `countryLandBordersCountry` is the one that the model simply knows. A single question

already names 195 of the 198 gold neighbors on train at a precision of 0.99, and the second question adds none of the missing three there and two on validation, so the pool is at its ceiling with one question and the rule only has to avoid spoiling it.

`companyTradesAtStockExchange` shows the trade-off the whole system is built around. The opening of an encyclopedia article reaches the most exchanges of the five questions and is the least precise of them, while the plain question reaches the fewest at the highest precision. The pool that collects them is therefore less precise than any single question in it, 0.42 against 0.52 to 0.73 on train, and buying that precision back is what the quorum of Section 4.2 spends its two agreeing questions on. A third of the companies are not listed anywhere, and there the full point goes to an empty prediction, so part of what the quorum buys is the ability to stay silent.

The objects for the relation `hasArea` are asked three ways that turn out to be one. All three land within four points of one another in recall and in precision alike, and the pool ends up three points above the best of them on train and five on validation, so a fourth way of asking would be wasted.

`personHasCityOfDeath` gains the most from asking differently and loses the most at the decision. Having the model narrate a career or the final years and then reading the city out of that passage reaches 0.59 and 0.53 of the gold cities on train, where the two questions that ask more directly reach 0.40 each, and the pool reaches 0.64 against a score of 0.52. At a pool precision of 0.07, the right city is usually in the pool and usually outvoted, so what is missing here is the choice among candidates the model did produce.

In our setting, the best results for the relation `awardWonBy` were achieved while asking in slices. With 111 names per item on average, no single answer holds the set, and the question that asks one year at a time names 609 of the 764 gold names on train, 90 of which no other question names at all. With ten items per split, though, nothing measured on this relation resolves below a tenth of a point.

`hasCapacity` is the second relation whose loss sits in the decision. Its two questions each find twelve to sixteen values the other misses, which lifts the pool to 0.68 on train, where the best single question reaches 0.54, and the relation still scores 0.25. The scorer’s tolerance of five percent is wide, so some of that recall could be luck: a pool of the same size drawn at random from the training

Question	Recall		Precision	
	Tr	Va	Tr	Va
countryLandBordersCountry				
which countries border it	.98	.96	.99	.93
how many, and which ones	.98	.93	.97	.92
<i>any of them</i>	.98	.97	.98	.91
companyTradesAtStockExchange				
the plain question	.51	.62	.73	.85
a company record as JSON	.53	.53	.69	.78
an encyclopedia opening	.72	.76	.52	.62
the home-market language	.56	.66	.57	.62
the ticker symbol	.56	.67	.65	.76
<i>any of them</i>	.81	.84	.42	.46
hasArea				
a gazetteer entry	.85	.87	.60	.60
comparable features	.85	.86	.63	.60
the two nearest in area	.83	.84	.59	.58
<i>any of them</i>	.88	.92	.52	.51
personHasCityOfDeath				
a short biography	.40	.44	.19	.22
the year, then the city	.40	.41	.34	.38
the career city by city	.59	.67	.11	.14
the final years	.53	.62	.11	.14
<i>any of them</i>	.64	.74	.07	.09
awardWonBy				
one period at a time	.58	.16	.57	.25
the plain question, 8 draws	.47	.14	.27	.16
one field of work at a time	.33	.16	.30	.16
one year at a time	.80	.28	.28	.14
<i>any of them</i>	.85	.36	.19	.10
hasCapacity				
an encyclopedia opening	.54	.65	.12	.12
a venue record as JSON	.52	.66	.10	.14
<i>any of them</i>	.68	.78	.11	.12

Table 2: What each question puts into the pool, before any decision rule sees it, on train (Tr) and validation (Va): the share of the gold it names (recall), and the share of what it names that is gold (precision). The “any of them” row is the whole pool, and its recall is the ceiling on the relation.

distribution of capacities lands inside the window for 42.0% of the items. The model reaches 68.0% and 78.4%, six standard deviations above, so the values are recalled and then lost at the decision.

6.2 Against the 2025 Winner

ReWiSe won the 2025 edition with one question asked twenty times, chain-of-thought examples, and one aggregation rule per relation (Albert-Roulhac and Zouaq, 2025). Re-running it on the 2025 validation split scores 0.424 against the 0.415 its authors report, with no relation off by more than 0.023. Table 3 then carries it to this year’s data and to our backbone: on qwen3-8b the two systems are level, so the distance on the stronger model is as much a property of the backbone as of the method.

2026 validation	qwen3-8b	mistral-24b
ReWiSe	0.435	0.624
MAVE	0.443	0.670

Table 3: Validation F1 of the 2025 winner and of MAVE on the 2026 data, run on the model that edition prescribed for everyone and on ours.

6.3 How Far the Same System Drifts

The same configuration does not return the same answer twice. Four of the six relations sample at a temperature above zero, so their pools are redrawn on every run, and even the greedy calls do not repeat, because a model served through a hosted endpoint answers the same prompt differently from one run to the next (Atil et al., 2025). The spread beside each score in Table 1 is what redrawing alone does, and it understates the whole: drawn a second time from scratch, one configuration moved by 0.011 on train and 0.031 on validation, and single relations further. A difference of one or two items is therefore below what two runs resolve.

7 Conclusion

On RQ, what carries under a closed book and a capped parameter count is asking one open-weight 24B model in many ways: from the 0.3085 of a single direct generation to an official macro-F1 of 0.6908. Asking again pays where the questions reach the fact differently, above all where the model writes a passage first and the answer is read out of it, and it pays nothing where they repeat one another. It is reached on 24B of the 32B the rules allow, a quarter under the cap, and none of the three smaller partners we tried beside that model paid for its parameters: a competitive system here is a matter of how the model is asked rather than of how many parameters are spent. What limits the system now is no longer only what the model recalls: on the two weakest relations the right answer is usually in the pool and gets outvoted. Of the 21 system papers of the four earlier editions, one gives an interval around its headline score (Bara, 2024); on splits this size a spread belongs beside every number.

Limitations. The results rest on one model of one family, and the decision layer had to be re-derived when we changed models, which suggests the quorum is a property of the model as much as of the relation. Our score is higher on test than on either labelled split, so the numbers we can verify are the conservative ones.

Acknowledgments

We thank Optimax Energy GmbH for their collaboration as a practice partner. This work was funded by the German Federal Ministry of Research, Technology and Space (BMFTR), grant number 03FHP239.

References

- Edouard Albert-Roulhac and Amal Zouaq. 2025. [ReWiSe: Relation-wise self-consistency for LLM probing](#). In *Proceedings of the Workshop on Knowledge Base Construction from Pre-trained Language Models (KBC-LM) and the Challenge (LM-KBC)*, volume 4041 of *CEUR Workshop Proceedings*.
- Dimitrios Alivanistos, Selene Baez Santamaria, Michael Cochez, Jan-Christoph Kalo, Emile van Krieken, and Thiviyan Thanapalasingam. 2022. [Prompting as probing: Using language models for knowledge base construction](#). In *Proceedings of the Semantic Web Challenge on Knowledge Base Construction from Pre-trained Language Models (LM-KBC)*, volume 3274 of *CEUR Workshop Proceedings*.
- Berk Atıl, Sarp Aykent, Alexa Chittams, Lisheng Fu, Rebecca J. Passonneau, Evan Radcliffe, Guru Rajan Rajagopal, Adam Sloan, Tomasz Tudrej, Ferhan Ture, Zhe Wu, Lixinyu Xu, and Breck Baldwin. 2025. Non-determinism of “deterministic” LLM system settings in hosted environments. In *Proceedings of the 5th Workshop on Evaluation and Comparison of NLP Systems*, pages 135–148, Mumbai, India. Association for Computational Linguistics.
- Davide Mario Ricardo Bara. 2024. [Prompt engineering for tail prediction in domain-specific knowledge graph completion tasks](#). In *Proceedings of the Workshop on Knowledge Base Construction from Pre-trained Language Models (KBC-LM) and the Challenge (LM-KBC)*, volume 3853 of *CEUR Workshop Proceedings*.
- Alex Clay, Ernesto Jiménez-Ruiz, and Pranava Madhyastha. 2025. [Judge, generator, executioner: Utilizing an LLM for KBC](#). In *Proceedings of the Workshop on Knowledge Base Construction from Pre-trained Language Models (KBC-LM) and the Challenge (LM-KBC)*, volume 4041 of *CEUR Workshop Proceedings*.
- Aldan Creo, Christophe Gueret, Alberto Bernardi, Adrianna Janik, and Luca Costabello. 2025. [Soft thinking: Enhancing knowledge base completion through trainable prompts in the chain of thought](#). In *Proceedings of the Workshop on Knowledge Base Construction from Pre-trained Language Models (KBC-LM) and the Challenge (LM-KBC)*, volume 4041 of *CEUR Workshop Proceedings*.
- Arunav Das, Nadeen Fathallah, and Nicole Obretincheva. 2024. [Navigating nulls, numbers and numerous entities: Robust knowledge base construction from large language models](#). In *Proceedings of the Workshop on Knowledge Base Construction from Pre-trained Language Models (KBC-LM) and the Challenge (LM-KBC)*, volume 3853 of *CEUR Workshop Proceedings*.
- Jingbo He and Simon Razniewski. 2025. [Combining self-retrieval-augmented generation with divide-and-conquer for language model-based knowledge base construction](#). In *Proceedings of the Workshop on Knowledge Base Construction from Pre-trained Language Models (KBC-LM) and the Challenge (LM-KBC)*, volume 4041 of *CEUR Workshop Proceedings*.
- Jan-Christoph Kalo, Tuan-Phong Nguyen, Simon Razniewski, and Bohui Zhang. 2024. [Preface: LM-KBC challenge 2024](#). In *Proceedings of the Workshop on Knowledge Base Construction from Pre-trained Language Models (KBC-LM) and the Challenge (LM-KBC)*, volume 3853 of *CEUR Workshop Proceedings*.
- Jan-Christoph Kalo, Tuan-Phong Nguyen, Simon Razniewski, and Bohui Zhang. 2026. [AKBC Shared Task 2026: Predicting complete knowledge base entries from language models](#). In *Proceedings of the 10th Workshop on Automated Knowledge Base Construction (AKBC) at EMNLP 2026*. Association for Computational Linguistics.
- Jan-Christoph Kalo, Simon Razniewski, Bohui Zhang, and Tuan-Phong Nguyen. 2025. [LM-KBC 2025: 4th challenge on knowledge base construction from pre-trained language models](#). In *Proceedings of the Workshop on Knowledge Base Construction from Pre-trained Language Models (KBC-LM) and the Challenge (LM-KBC)*, volume 4041 of *CEUR Workshop Proceedings*.
- Jan-Christoph Kalo, Sneha Singhania, Simon Razniewski, and Jeff Z. Pan. 2023. [LM-KBC 2023: 2nd challenge on knowledge base construction from pre-trained language models](#). In *Proceedings of the Workshop on Knowledge Base Construction from Pre-trained Language Models (KBC-LM) and the Challenge (LM-KBC)*, volume 3577 of *CEUR Workshop Proceedings*.
- Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large language models struggle to learn long-tail knowledge. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pages 15696–15707.
- Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. [When not to trust language models: Investigating effectiveness of parametric and non-parametric memories](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822, Toronto, Canada. Association for Computational Linguistics.

- Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. 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)*, pages 5418–5426, Online. Association for Computational Linguistics.
- 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)*, volume 3274 of *CEUR Workshop Proceedings*.