



HINTEVAL: An Open-Source Python Toolkit for Hint Generation and Hint Evaluation

Jamshid Mozafari^{*}, Bhawna Piryani, Abdelrahman Abdallah, Adam Jatowt

University of Innsbruck, Innsbruck, Austria
{jamshid.mozafari, adam.jatowt}@uibk.ac.at

Abstract

Large Language Models (LLMs) increasingly provide direct answers to user questions, raising concerns about reduced engagement in critical thinking and problem-solving. Hint generation offers an alternative by guiding users toward answers without revealing them, while hint evaluation assesses the quality of such guidance. Research in this area is hindered by fragmented datasets, inconsistent annotation formats, and evaluation tools that are often dataset-specific or unavailable. To address these challenges, we introduce HINTEVAL¹, an open-source Python library² for unified hint generation and evaluation. HINTEVAL standardizes access to diverse hint datasets, supports answer-aware and answer-agnostic generation methods, and implements multiple evaluation metrics within a shared data model. The toolkit enables reproducible experimentation, cross-dataset analysis, and multi-dimensional evaluation with minimal engineering effort. We further demonstrate its utility through human studies in which participants assess generated hints and use them to answer questions, showing that hints can effectively support users in reaching correct answers. HINTEVAL is accompanied by comprehensive documentation³, an executable Google Colab notebook for rapid experimentation, and a demonstration video⁴. By promoting consistent evaluation practices and lowering barriers to entry, it facilitates systematic research on hint-based question answering (QA) in NLP and IR.

1 Introduction

Large Language Models (LLMs) have transformed Information Retrieval (IR) and Natural Language Processing (NLP) by enabling users to obtain

answers to diverse questions quickly and conveniently (Gemini Team et al., 2023; OpenAI et al., 2023; Dubey et al., 2024; Qin et al., 2024). While this paradigm improves access to information, it has also raised concerns about reduced engagement in critical thinking, reasoning, and memory retention due to increasing reliance on AI systems (Darvishi et al., 2024; Heersmink, 2024; Lee et al., 2025). This phenomenon, often referred to as *cognitive offloading*, highlights the need for AI systems that support rather than replace human reasoning (Singh et al., 2025; Gerlich, 2025).

To address this challenge, recent AI systems have begun exploring cognition-based interaction paradigms that guide users toward answers instead of providing them directly. Examples include ChatGPT *Study and Learn*⁵ mode and Google *LearnLM*⁶. These approaches reflect growing interest in *hint-based* question answering (QA), where hints provide clues that help users solve without explicitly revealing the answer (Hume et al., 1996). Such interactions encourage active reasoning while retaining the benefits of AI assistance.

Research in this area focuses on two complementary tasks: *Hint Generation* and *Hint Evaluation*. Hint Generation aims to produce guidance that leads users toward the correct answer without disclosing it (Jangra et al., 2025), while Hint Evaluation assesses hint quality along several dimensions (Jatowt et al., 2023; Bandura, 2013). Figure 1 shows example hints and their evaluations.

Despite growing interest in hint-based methods (Mozafari et al., 2026b), research progress remains hindered by fragmented datasets, heterogeneous annotation formats, and evaluation tools that are often dataset-specific or unavailable. Existing studies frequently introduce their own data formats, generation pipelines, and implementations,

^{*} Corresponding Author.

¹<https://github.com/DataScienceUIBK/HintEval>

²<https://pypi.org/project/hinteval/>

³<https://hinteval.readthedocs.io/>

⁴https://youtu.be/QTPvW4V_6KY

⁵<https://openai.com/index/chatgpt-study-mode/>

⁶<https://cloud.google.com/solutions/learnlm>

Question:	What musician's album 'Bad' produced five consecutive No. 1 singles on the Billboard Hot 100 chart?	Answer:	Michael Jackson	MJ		
Hint		Relevance	Readability	Convergence	Familiarity	Answer Leakage
He is called the "King of Pop".		0.89	0	1	0.98	0.37
His famous move is "moon walk".		0.90	0	1	0.93	0.39
"Beat It" is a song from his album "Thriller".		0.89	0	1	0.87	0.42
This musician has 39 Guinness World records.		0.91	0	0.64	1	0.53
His first wife is Lisa Marie Presley and second wife is Debbie Rowe.		0.91	0	1	0.30	0.37

Figure 1: Example hints for a sample question with scoring metrics. The metrics Relevance, Convergence, Familiarity, and Answer Leakage are rated on a scale from 0 to 1, where 0 represents the lowest and 1 the highest value. Higher scores in Relevance, Convergence, and Familiarity indicate better results, while a lower score is preferable for Answer Leakage. The Readability metric is scored as 0 (Beginner), 1 (Intermediate), or 2 (Advanced), with lower values indicating better readability.

making it difficult to compare methods, reproduce results, and conduct cross-dataset analyses. As a result, *comparing generation methods, studying evaluation metrics, and reproducing experimental findings require substantial engineering effort.*

To address these challenges, we introduce HINTEVAL, the first unified and extensible toolkit for hint generation and evaluation in NLP and IR. HINTEVAL standardizes access to diverse hint datasets, provides consistent interfaces for answer-aware and answer-agnostic hint generation, and implements widely used evaluation metrics within a shared data model. The toolkit enables reproducible experimentation, cross-dataset analysis, and large-scale evaluation with *minimal engineering overhead*. We further demonstrate its usefulness through human-centered evaluations. It is released as open-source software with documentation, preprocessed datasets, a Google Colab notebook, and reproducible pipelines. The main contributions of this paper are:

1. We introduce HINTEVAL, the first unified toolkit that standardizes datasets, generation interfaces, and evaluation metrics for hint-based research.
2. We provide a unified implementation of diverse hint evaluation metrics for systematic hint assessment.
3. We demonstrate the toolkit through automatic and human-centered studies of hint quality and effectiveness.
4. We release HINTEVAL as open-source software with documentation, preprocessed datasets, a Google Colab notebook, and reproducible pipelines.

2 Related Work

Hint generation originated in intelligent tutoring systems, where hints were designed to support reasoning and learning by guiding students toward solutions rather than providing direct answers (McBroom et al., 2021; Kochmar et al., 2022; Price et al., 2019). More recently, hint-based interactions have been adapted to QA, with studies exploring answer-aware and answer-agnostic generation strategies, dataset construction, and automatic evaluation methods (Mozafari et al., 2024a,b, 2025). A comprehensive overview of recent advances in hint generation is provided by Jangra et al. (2025).

Most existing work builds on factual QA benchmarks such as TriviaQA (Joshi et al., 2017), Natural Questions (Kwiatkowski et al., 2019), and WebQuestions (Berant et al., 2013). Representative resources, including TriviaHG (Mozafari et al., 2024b), HintQA (Mozafari et al., 2024a), WikiHint (Mozafari et al., 2025), KG-Hint (Jatowt et al., 2023), differ in their data sources, annotation schemes, and generation settings. Early approaches relied on human-authored clues or knowledge graphs, whereas recent work employs LLMs for scalable hint generation (Dubey et al., 2024; Jangra and Muresan, 2025). Hint evaluation further extends traditional QA evaluation by considering dimensions such as Relevance, Readability, Convergence, Familiarity, and Answer Leakage (Mozafari et al., 2024b, 2025, 2026a). However, existing implementations are typically dataset-specific, making systematic comparison across datasets and generation methods difficult.

General NLP frameworks support model development and experimentation, but they are not designed for hint-oriented workflows (Lin et al., 2021; Ushio et al., 2023). Existing hint-related work typi-

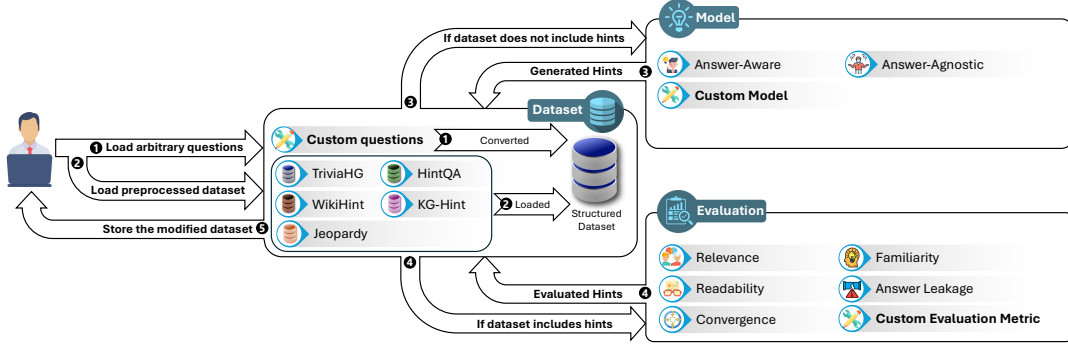


Figure 2: Workflow of HINTEVAL: ① Questions are loaded and converted into a structured dataset using the Dataset module. ② Users can load preprocessed datasets as a structured dataset. ③ Hints can be generated for each question using the Model module and stored in the dataset object. ④ The Evaluation module assesses all generated hints and questions using various evaluation metrics, storing the results in the dataset object. ⑤ The updated dataset can be saved and reloaded as needed.

```
@hinteval.cores.evaluation_metrics.familiarity.Wikipedia
def evaluate(self,
             sentences: List[Question | Hint | Answer],
             **kwargs: Any) -> List[float]

Evaluates the familiarity of the given Question, Hint, or Answer using the number of views of corresponding
Wikipedia pages [35] ①

Notes ②
This function stores the scores as Metric objects within the metrics attribute of the Question, Hint, or Answer,
with names based on the method, such as "familiarity-wikipedia-sm".
This function also stores number of views for each entity as Entity objects within the entities attribute.

Examples ③
>>> from hinteval.cores import Question, Hint
>>> from hinteval.evaluation.familiarity import Wikipedia
>>>
>>> wikipedia = Wikipedia(spacy_pipeline="en_core_web_sm")
>>> sentence_1 = Question("What is the capital of Austria?")
>>> sentence_2 = Hint("This city, once home to Mozart and Beethoven, is the capital of Austria.")
>>> sentences = [sentence_1, sentence_2]
>>> results = wikipedia.evaluate(sentences)
>>> print(results)
# [1.0, 1.0]
>>> metrics = [{f'{metric_key}: {metric_value.value}' for sent in sentences for metric_key,
metric_value in
... sent.metrics.items()}]
>>> print(metrics)
# [{"familiarity-wikipedia-sm: 1.0", "familiarity-wikipedia-sm: 1.0"}]
>>> entities = [{f'{entity.entity}:{entity.metadata["wiki_views_per_month"]}' for sent in sentences
for entity in
... sent.entities}]
>>> print(entities)
# [{"austria: 248164", "mozart: 233219", "beethoven: 224128", "austria: 248164"}]

References ④
[35] Jamshid Mozafari, Anubhav Jangra, and Adam Jatowt. 2024. TriviaHG: A Dataset for Automatic Hint
Generation from Factoid Questions. In Proceedings of the 47th International ACM SIGIR Conference on
Research and Development in Information Retrieval (SIGIR '24). Association for Computing Machinery, New
York, NY, USA, 2060–2070. https://doi.org/10.1145/3626772.3657855

See Also ⑤
WordFrequency
Class for evaluating familiarity of Question, Hint, or Answer based on word frequency analysis on Common
Crawl.

Params: sentences - List of sentences to evaluate. ⑥
kwargs - Additional keyword arguments.
Returns: List of familiarity scores for each sentence. ⑦
```

Figure 3: Docstring for the evaluate function of the Wikipedia method within the Familiarity metric. It includes ① a function description, ② metric- and method-specific notes, and ③ a usage example. The documentation also provides ④ references to relevant publications, ⑤ links to related functions, ⑥ parameter descriptions, and ⑦ return value information.

cally releases datasets, generation methods, or evaluation procedures independently. To the best of our knowledge, no prior toolkit unifies hint datasets, hint generation, and multi-dimensional hint evaluation within a single extensible framework. This gap motivates the development of HINTEVAL.

3 HINTEVAL

HINTEVAL is a Python toolkit designed to simplify hint generation and evaluation, distributed via PyPI⁷, and designed to integrate naturally with common machine learning workflows based on libraries such as PyTorch (Paszke et al., 2019). The toolkit can be installed using:

```
$ pip install hinteval
```

This paper introduces version 0.1.0, the latest public release. HINTEVAL is open-source⁸ and actively maintained. The toolkit provides documentation, preprocessed datasets, reproducible examples, and a Google Colab notebook demonstrating dataset loading, hint generation, and evaluation workflows. These resources enable users to reproduce the experiments presented in this paper with minimal setup effort.

HINTEVAL consists of three main modules: *Dataset*, *Model*, and *Evaluation*. The *Dataset* module standardizes access to heterogeneous hint datasets, the *Model* module supports hint generation, and the *Evaluation* module provides a unified interface for computing hint-related metrics. Figure 2 illustrates the overall workflow.

All public classes and methods in HINTEVAL are documented using structured docstrings that are automatically rendered in the online documentation. Figure 3 illustrates an example documentation page for the Wikipedia-based familiarity evaluator, including descriptions, usage examples, references, cross-links, and parameter and return specifications.

⁷<https://pypi.org/project/hinteval/>

⁸released under the Apache-2.0 License.

Dataset	Questions	Hints	Source	Aw.	Ag.
TriviaHG	16,645	160,228	LLM	✓	✗
WikiHint	1,000	5,000	Human/LLM	✓	✓
HintQA	16,955	305,624	LLM	✗	✓
KG-Hint	30	307	KG	✗	✓
Jeopardy	68,886	347,484	Human/Machine	✗	✓

Table 1: Summary of the preprocessed hint datasets supported by HINTEVAL. Columns Aw. and Ag. indicate whether the dataset contains answer-aware and answer-agnostic hints, respectively.

3.1 Dataset

A central goal of HINTEVAL is to enable seamless experimentation across multiple hint datasets with minimal engineering effort. To this end, we introduce a unified dataset abstraction that harmonizes diverse annotation styles and storage formats, as shown in Figure 4.

Each dataset comprises one or more subsets (e.g., train, validation, test), where each subset contains a collection of instances indexed by unique question IDs. An instance consists of a Question, one or more Answer objects, and a set of Hint objects. This structure supports questions with multiple valid answers and multiple associated hints.

The Question, Answer, and Hint classes share common attributes for entities and evaluation metrics. Named entities are automatically extracted using spaCy (Honnibal et al., 2020), while metric values are stored in a unified format to support consistent analysis across datasets and evaluation methods. Dataset-specific features are preserved through a flexible metadata field.

HINTEVAL provides preprocessed versions of several existing hint datasets, which can be downloaded and loaded with a few lines of code. Users may also define custom datasets using the same abstraction. Datasets can be saved and loaded using JSON format, enabling easy reuse and sharing across experiments. Table 1 summarizes the preprocessed datasets currently supported by HINTEVAL. The supported datasets cover human-authored, knowledge-graph-based, and LLM-generated hints, as well as both answer-aware and answer-agnostic generation settings.

To demonstrate the dataset construction capabilities of HINTEVAL, we provide a large-scale Jeopardy-based resource containing 68,886 questions and 347,484 hints. The resource is derived from original Jeopardy clues, which are treated as hints, while questions are generated using LLMs and subsequently evaluated using HINTEVAL. This

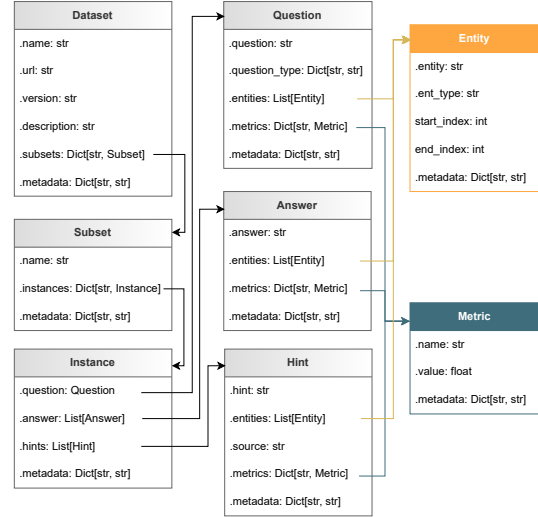


Figure 4: Schema of the Dataset class, illustrating the objects used to represent hint datasets in HINTEVAL. Arrows indicate relationships between the objects.

example demonstrates how HINTEVAL simplifies the construction and evaluation of new hint datasets while supporting large-scale experimentation. Additional details of the dataset creation pipeline are provided in Appendix A.

3.2 Model

HINTEVAL includes two built-in hint generation models: *Answer-Aware* and *Answer-Agnostic*.

The *Answer-Aware* model generates hints given both a question and its ground-truth answer. While this setting typically produces higher-quality hints (Mozafari et al., 2025), it assumes that gold answers are available. This approach was used to construct datasets such as TriviaHG (Mozafari et al., 2024b). In HINTEVAL, this model is implemented via the AnswerAware class, which populates instances with generated hints. In practice, *it can support teachers in preparing lesson materials*.

The *Answer-Agnostic* model generates hints using only the question, making it applicable when answers are unavailable. Although more flexible, this approach may yield lower-quality hints due to uncertainty about the correct answer (Mozafari et al., 2025). Datasets such as HintQA (Mozafari et al., 2024a) adopt this setting, while WikiHint (Mozafari et al., 2025) combines both approaches. The model is implemented as the AnswerAgnostic class. In practice, *it can help users avoid cognitive offloading by providing hints instead of direct answers*.

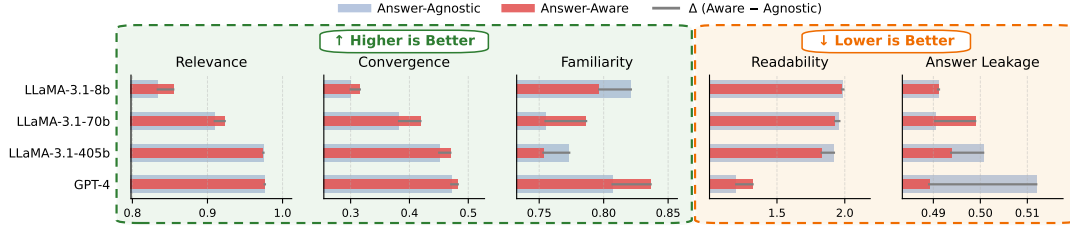


Figure 5: Comparison of hint generation strategies across four LLMs using evaluation metrics.

Both models inherit from a `Model` base class and expose a unified `generate` interface, enabling integration with datasets and evaluation modules. Users can extend the toolkit by implementing custom models that conform to the `Model` interface.

3.3 Evaluation

Hint evaluation assesses the quality of generated hints using multiple complementary criteria. HINTEVAL implements five evaluation dimensions—*Relevance*, *Readability*, *Convergence*, *Familiarity*, and *Answer Leakage*—through fifteen evaluation methods and thirty-five sub-methods. *Relevance* measures the semantic relatedness between a hint and its question, *Readability* captures the ease with which a hint can be understood, *Convergence* evaluates how effectively a hint narrows the space of possible answers, *Familiarity* estimates how common or well-known the hinted information is, and *Answer Leakage* assesses the extent to which a hint reveals the answer.

Early work introduced evaluation metrics for Convergence and Familiarity (Mozafari et al., 2024b), while subsequent studies proposed methods for Relevance, Readability, and Answer Leakage (Mozafari et al., 2025). However, existing implementations are dataset-specific, computationally expensive, or tightly coupled to particular schemas.

HINTEVAL implements all evaluation metrics in a dataset-agnostic manner, enabling them to operate on any dataset represented using the `Dataset` abstraction. In addition, HINTEVAL introduces multiple evaluation methods per metric, ranging from lightweight statistical approaches to neural and LLM-based methods, allowing users to balance accuracy and computational cost. Table 5 in the Appendix compares the supported evaluation methods across metrics. The assessments are qualitative guidelines based on the computational characteristics reported in the original publications and our experience implementing and using the methods within HINTEVAL. They are intended to

assist users in selecting evaluation methods rather than to serve as formal benchmark measurements.

Evaluation metrics in HINTEVAL inherit from a common `Evaluation` base class. Users can define custom metrics by implementing the `evaluate` function, enabling seamless extension and integration with the existing evaluation pipeline.

Listing 1 presents a code example using HINTEVAL that demonstrates the three main modules of the pipeline, from dataset construction to result storage. HINTEVAL supports running LLMs either locally or via remote APIs, allowing users to select an execution mode based on their available computational resources. Models hosted on HuggingFace⁹ can be used seamlessly. For further details, we refer readers to the online documentation, which provides comprehensive explanations and illustrative examples.

4 Experiments and Results

We demonstrate three representative capabilities of HINTEVAL. Specifically, we show how the toolkit supports systematic comparison of hint generation strategies, exploratory analyses of evaluation metrics, and validation of generated hints through human studies. These examples illustrate typical research workflows enabled by HINTEVAL.

4.1 Comparing Hint Generation Strategies

HINTEVAL provides a unified evaluation pipeline that enables comparison of hint generation strategies across models and datasets. As an example, we compare Answer-Aware and Answer-Agnostic generation using four LLMs on the WikiHint dataset.

Figure 5 shows that Answer-Aware generation generally improves Relevance, Convergence, and Readability while reducing Answer Leakage. The differences become smaller for stronger LLMs, whereas Familiarity remains largely unchanged.

⁹<https://huggingface.co/>

```

from hinteval import Dataset
from hinteval.cores.dataset_core import Instance,
Subset
from hinteval.model import AnswerAware,
AnswerAgnostic
from hinteval.evaluation.relevance import Rouge
from hinteval.evaluation.readability import
MachineLearningBased
from hinteval.evaluation.convergence import
NeuralNetworkBased
from hinteval.evaluation.familiarity import
Wikipedia
from hinteval.evaluation.answer_leakage import
Lexical

##### Dataset #####
instance_1 = Instance.from_strings('Q1', ['A11'],
[])
instance_2 = Instance.from_strings('Q2', ['A21'],
[])

subset = Subset('test')
subset.add_instance(instance_1)
subset.add_instance(instance_2)

dataset = Dataset('own_dataset')
dataset.add_subset(subset)

##### Model #####
instances = dataset['test'].get_instances()

answer_aware = AnswerAware('Qwen/Qwen3-8B',
num_of_hints=3)
answer_aware.generate(instances)

##### Evaluation #####
relevance = Rouge('rougeL')
    .evaluate(instances)
readability = MachineLearningBased('xgboost')
    .evaluate(instance_1.hints)
convergence = NeuralNetworkBased('bert-base')
    .evaluate(instances)
familiarity = Wikipedia()
    .evaluate(instance_2.hints)
answer_leakage = Lexical('exclude_stop_words')
    .evaluate(instances)

##### Store #####
dataset.store_json('./path/to/dataset.json')

```

Listing 1: An example showing the HINTEVAL workflow for creating a custom dataset (own_dataset) with two instances and a test subset, generating three hints using AnswerAware with Qwen3-8B, evaluating them with five metrics, and saving the results to dataset.json.

This example¹⁰ demonstrates how HINTEVAL enables comparison of hint generation strategies using multiple complementary evaluation metrics. Researchers can readily apply the same pipeline to compare their own methods with existing approaches with minimal additional code.

4.2 Analyzing Evaluation Metrics

Beyond evaluating individual hints, HINTEVAL supports large-scale analyses of relationships among evaluation metrics.

Figure 6 presents correlations between the five

¹⁰<https://github.com/DataScienceUIBK/HintEval/blob/main/experiments/aware-vs-agnostic/aware-vs-agnostic.py>

Metric	Relevance	Readability	Familiarity	Convergence	Answer Leakage
Mean Rating	4.01	4.47	3.81	3.57	1.76

Table 2: Human evaluation results on 2,214 hints. Ratings are on a five-point Likert scale (1: very low, 5: very high). Higher scores are preferable for Relevance, Readability, Familiarity, and Convergence, whereas lower scores are preferable for Answer Leakage.

Metric	Relevance	Readability	Familiarity	Convergence	Answer Leakage
Spearman’s ρ	0.45	0.41	0.53	0.61	0.53

Table 3: Correlation between automatic evaluation metrics and human ratings. Spearman’s ρ is computed using the subset of hints evaluated by human annotators.

evaluation dimensions computed on TriviaHG and WikiHint. Most correlations are weak and dataset-dependent, indicating that the metrics capture complementary aspects of hint quality. The only consistent trend is a positive correlation between Convergence and Answer Leakage, highlighting the inherent trade-off between producing informative hints and avoiding answer disclosure.

This example¹¹ illustrates how HINTEVAL enables researchers to easily analyze relationships among evaluation metrics on their own datasets using a unified analysis pipeline with minimal additional code.

4.3 Human Evaluation

Automatic metrics provide scalable evaluation, but human assessment remains essential for validating the quality and usefulness of hints generated by HINTEVAL. We conducted two complementary studies focused on hint quality and effectiveness: (1) a hint quality assessment and (2) a hint-assisted question answering study. We additionally conducted a separate toolkit usability study to assess the ease of use of HINTEVAL. Details of the human evaluation protocols are provided in Appendix B.

Hint Quality Assessment. We randomly selected 500 questions from the TriviaQA (Joshi et al., 2017) and NQ (Kwiatkowski et al., 2019) test sets covering diverse categories. For each question, we generated five hints using LLaMA-3.3-70B, resulting in 2,500 hints. To reduce answer leakage, we removed hints with an Answer Leakage score higher than 0.8, leaving 2,214 hints for evaluation.

Twenty human annotators participated in the

¹¹<https://github.com/DataScienceUIBK/HintEval/blob/main/experiments/metric-agreement/metric-agreement.py>

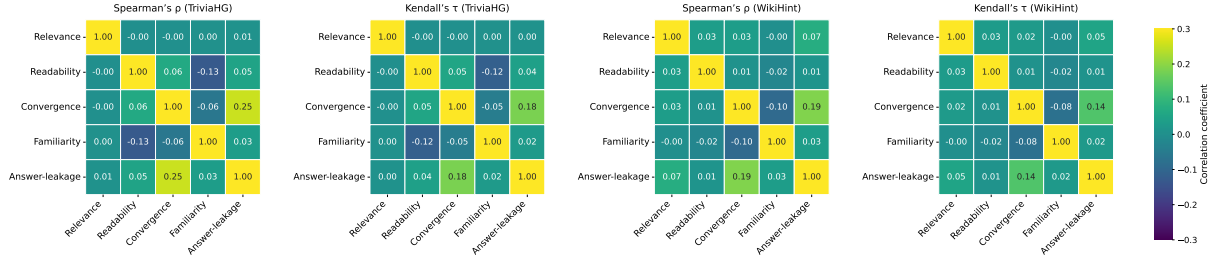


Figure 6: Correlation among evaluation metrics computed using Spearman’s ρ and Kendall’s τ for the TriviaHG and WikiHint datasets.

Statement	Mean
The toolkit was easy to install.	4.8
The API was easy to understand.	4.2
The documentation was helpful.	4.7
The provided examples were useful.	4.5
The toolkit was easy to use.	4.2
I would use HINTEVAL in future research.	3.8

Table 4: Usability evaluation of HINTEVAL.

study. The hints were distributed among annotators and assessed using a five-point Likert scale (1: very low, 5: very high) across the five dimensions implemented in HINTEVAL. Higher values indicate better quality for Relevance, Readability, Familiarity, and Convergence, whereas lower values indicate less Answer Leakage and are therefore preferable.

Table 2 reports the average ratings. Overall, annotators judged the generated hints to be relevant, understandable, and effective in guiding users toward the correct answer while avoiding direct answer disclosure. In particular, Relevance and Readability achieved average scores above 4, whereas Answer Leakage received a low score.

To assess whether the automatic evaluation metrics align with human judgment, we computed Spearman’s ρ between automatic metric scores and human ratings. As shown in Table 3, all metrics exhibit moderate positive correlations (0.41–0.61), providing evidence that the evaluation methods implemented in HINTEVAL capture aspects of hint quality that are consistent with human assessments.

Hint-Assisted Question Answering. To evaluate the usefulness of the generated hints, we recruited a separate group of thirty participants who had not taken part in the previous study.

Participants first attempted to answer the selected questions without hints. Questions answered correctly were excluded from the second phase, as no assistance was required. The remaining unan-

swered questions were then presented together with their corresponding hints.

Without hints, participants correctly answered 89 of the 500 questions (17.8%). After receiving hints, they correctly answered 312 of the remaining 401 unanswered questions, corresponding to a success rate of 77.8%. Overall question-answering accuracy increased from 17.8% to 80.2%.

These findings demonstrate that hints generated and evaluated using HINTEVAL effectively support users in answering questions while encouraging reasoning rather than directly revealing the answer.

Toolkit Usability Study. We conducted a usability study with ten students and researchers familiar with NLP or IR. Participants completed representative tasks using HINTEVAL, including dataset loading, hint generation, evaluation, and result export, before rating the toolkit on six five-point Likert-scale questions as shown in Table 4.

Overall, participants found HINTEVAL easy to learn and use, with positive ratings for the API, documentation, and example code, suggesting that the toolkit can be readily adopted for hint-generation research.

5 Conclusion

We presented HINTEVAL, an open-source toolkit for hint generation and evaluation that unifies datasets, generation interfaces, and evaluation metrics within a common framework. By standardizing data representations and evaluation workflows, HINTEVAL enables reproducible and cross-dataset experimentation with minimal engineering effort. Through automatic and human-centered studies, we demonstrated its utility for evaluating hint quality and supporting question answering. Future work will extend the framework with additional generation methods and evaluation metrics, and support for emerging LLMs.

Ethical Considerations

HINTEVAL is designed to support reasoning-focused interactions by providing hints rather than direct answers. As a research toolkit, it enables the generation and evaluation of hints for question-answering tasks, educational applications, and studies of human–AI interaction.

Several ethical considerations remain relevant. First, generated hints may inherit factual inaccuracies, hallucinations, or social biases from the underlying language models and datasets. Second, hints may inadvertently reveal answers, reducing their pedagogical value or enabling misuse in assessment settings. Third, the quality and fairness of generated hints may vary across domains, topics, and user populations.

To mitigate these risks, HINTEVAL incorporates multiple evaluation dimensions, including relevance, convergence, familiarity, readability, and answer leakage, allowing researchers and practitioners to assess hint quality and identify potentially problematic outputs. The toolkit is intended for research and educational purposes, and human oversight remains important when deploying hint-based systems in real-world settings.

Limitations

While HINTEVAL provides a unified and extensible framework for hint generation and evaluation, several limitations remain in the current version and experimental evaluation.

- **Domain coverage.** The current experiments primarily focus on factual and open-domain QA datasets. Further evaluation is needed to assess the generalizability of HINTEVAL to other domains and task settings.
- **Human evaluation.** The human studies involve relatively small groups of students and researchers, and the hint quality assessment is based on hints generated by a single LLM. These factors may limit the generalizability of the findings across broader user populations and generation models.

Acknowledgments

The computational results presented here have been achieved (in part) using the LEO HPC infrastructure of the University of Innsbruck and the Austrian Scientific Computing (ASC) infrastructure.

References

- Albert Bandura. 2013. The role of self-efficacy in goal-based motivation. *New developments in goal setting and task performance*, pages 147–157.
- Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. [Semantic parsing on Freebase from question-answer pairs](#). In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.
- Ali Darvishi, Hassan Khosravi, Shazia Sadiq, Dragan Gaevi, and George Siemens. 2024. [Impact of ai assistance on student agency](#). *Computers & Education*, 210:104967.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, and Others. 2024. [The Llama 3 Herd of Models](#). *arXiv e-prints*, arXiv:2407.21783.
- Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson, and Others. 2023. [Gemini: A Family of Highly Capable Multimodal Models](#). *arXiv e-prints*, arXiv:2312.11805.
- Michael Gerlich. 2025. [AI Tools in Society: Impacts on Cognitive Offloading and the Future of Critical Thinking](#). *Societies*, 15(1).
- Richard Heersmink. 2024. [Use of large language models might affect our cognitive skills](#). *Nature Human Behaviour*, 8(5):805–806.
- Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. [spaCy: Industrial-strength Natural Language Processing in Python](#).
- Gregory Hume, Joel Michael, Allen Rovick, and Martha Evens. 1996. [Hinting as a tactic in one-on-one tutoring](#). *The Journal of the Learning Sciences*, 5(1):23–47.
- Anubhav Jangra, Jamshid Mozafari, Adam Jatowt, and Smaranda Muresan. 2025. [Navigating the landscape of hint generation research: From the past to the future](#). *Transactions of the Association for Computational Linguistics*, 13:505–528.
- Anubhav Jangra and Smaranda Muresan. 2025. Designing and evaluating hint generation systems for science education. *arXiv preprint arXiv:2510.21087*.
- Adam Jatowt, Calvin Gehrer, and Michael Färber. 2023. [Automatic hint generation](#). In *Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval, ICTIR '23*, pages 117–123, New York, NY, USA. Association for Computing Machinery.

- Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. [TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.
- Ekaterina Kochmar, Dung Do Vu, Robert Belfer, Varun Gupta, Iulian Vlad Serban, and Joelle Pineau. 2022. [Automated data-driven generation of personalized pedagogical interventions in intelligent tutoring systems](#). *International Journal of Artificial Intelligence in Education*, 32(2):323–349.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, and Others. 2019. [Natural questions: A benchmark for question answering research](#). *Transactions of the Association for Computational Linguistics*, 7:452–466.
- Hao-Ping (Hank) Lee, Advait Sarkar, Lev Tankelevitch, Ian Drosos, Sean Rintel, Richard Banks, and Nicholas Wilson. 2025. [The Impact of Generative AI on Critical Thinking: Self-Reported Reductions in Cognitive Effort and Confidence Effects From a Survey of Knowledge Workers](#). In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI ’25, New York, NY, USA. Association for Computing Machinery.
- Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. [Pyserini: A python toolkit for reproducible information retrieval research with sparse and dense representations](#). In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’21, pages 2356–2362, New York, NY, USA. Association for Computing Machinery.
- Jessica McBroom, Irena Koprinska, and Kalina Yacef. 2021. [A survey of automated programming hint generation: The hints framework](#). *ACM Comput. Surv.*, 54(8).
- Jamshid Mozafari, Abdelrahman Abdallah, Bhawna Piryani, and Adam Jatowt. 2024a. [Exploring hint generation approaches for open-domain question answering](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 9327–9352, Miami, Florida, USA. Association for Computational Linguistics.
- Jamshid Mozafari, Florian Gerhold, and Adam Jatowt. 2025. [Wikihint: A human-annotated dataset for hint ranking and generation](#). In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’25, pages 3821–3831, New York, NY, USA. Association for Computing Machinery.
- Jamshid Mozafari, Anubhav Jangra, and Adam Jatowt. 2024b. [Triviahg: A dataset for automatic hint generation from factoid questions](#). SIGIR ’24, pages 2060–2070, New York, NY, USA. Association for Computing Machinery.
- Jamshid Mozafari, Bhawna Piryani, and Adam Jatowt. 2026a. [Context convergence improves answering inferential questions](#). In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’26, pages 4010–4015, New York, NY, USA. Association for Computing Machinery.
- Jamshid Mozafari, Hamed Zamani, Guido Zuccon, and Adam Jatowt. 2026b. [Inferential question answering](#). In *Proceedings of the ACM Web Conference 2026*, WWW ’26, pages 2384–2395, New York, NY, USA. Association for Computing Machinery.
- OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, and Others. 2023. [GPT-4 Technical Report](#). *arXiv e-prints*, arXiv:2303.08774.
- Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, and Others. 2019. *PyTorch: an imperative style, high-performance deep learning library*. Curran Associates Inc., Red Hook, NY, USA.
- Thomas W. Price, Yihuan Dong, Rui Zhi, Benjamin Paaßen, Nicholas Lytle, Veronica Cateté, and Tiffany Barnes. 2019. [A comparison of the quality of data-driven programming hint generation algorithms](#). *International Journal of Artificial Intelligence in Education*, 29(3):368–395.
- Libo Qin, Qiguang Chen, Xiachong Feng, Yang Wu, Yongheng Zhang, Yinghui Li, Min Li, Wanxiang Che, and Philip S. Yu. 2024. [Large Language Models Meet NLP: A Survey](#). *arXiv e-prints*, arXiv:2405.12819.
- Anjali Singh, Karan Taneja, Zhitong Guan, and Avijit Ghosh. 2025. Protecting human cognition in the age of AI. *arXiv preprint arXiv:2502.12447*.
- Asahi Ushio, Fernando Alva-Manchego, and Jose Camacho-Collados. 2023. [A practical toolkit for multilingual question and answer generation](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 86–94, Toronto, Canada. Association for Computational Linguistics.

Metric	Method	Preferred Device	Cost Effectiveness	Accuracy	Execution Speed
Relevance	ROUGE	CPU	High	Low	Very Fast
	Non-Contextual	CPU/GPU	High	Moderate	Fast
	Contextual	GPU	Moderate	High	Moderate
	LLM	GPU	Low	Very High	Slow
Readability	Traditional	CPU	High	Low	Very Fast
	Machine-Learning	CPU	High	Moderate	Moderate
	Neural-Network	GPU	Moderate	High	Moderate
	LLM	GPU	Low	Very High	Slow
Convergence	Specificity	GPU	Moderate	Low	Moderate
	Neural-Network	GPU	Moderate	Moderate	Moderate
	LLM	GPU	Low	High	Slow
Familiarity	Word-Frequency	CPU	Very High	Low	Very Fast
	Wikipedia	CPU	High	High	Slow
Answer Leakage	Lexical	CPU	Very High	Low	Very Fast
	Contextual	GPU	Moderate	High	Moderate

Table 5: Comparison of evaluation methods implemented in HINTEVAL. Accuracy, cost-effectiveness, and speed are qualitative assessments intended to guide method selection.

A Jeopardy Dataset Construction

To demonstrate the dataset construction capabilities of HINTEVAL, we created a large-scale Jeopardy-based resource, named Jeopardy, containing 68,886 questions and 347,484 hints.

The resource was derived from the publicly available Jeopardy clue collection¹². Since Jeopardy clues provide information that guides participants toward a correct response, we treat them as hints. For each clue, an LLM was prompted to generate a question. The resulting question–hint pairs were subsequently processed using HINTEVAL.

The construction pipeline consisted of four stages: **(1)** Collection of Jeopardy clues and associated metadata. **(2)** Conversion of clues into hint instances using the HINTEVAL data model. **(3)** Generation of questions using LLaMA 3.3 70B. **(4)** Evaluation of the resulting hint–question pairs using HINTEVAL, including entity extraction and computation of all supported evaluation metrics.

The final resource contains 68,886 questions, 347,484 hints, automatically extracted entities, and precomputed evaluation scores. The resulting dataset demonstrates how HINTEVAL can be used not only to evaluate existing hint datasets but also to construct, enrich, and analyze new hint-oriented resources at scale.

B Human Evaluation Protocol

Hint Quality Assessment. We randomly selected 500 questions from the TriviaQA and NQ test sets and generated five hints per question using LLaMA-3.3-70B. After filtering hints with an

¹²https://github.com/jwolle1/jeopardy_clue_dataset

Answer Leakage score higher than 0.8, 2,214 hints remained.

Twenty annotators, consisting of students and researchers, participated in the study. To balance annotation quality and effort, the generated hints were distributed among annotators such that each hint was independently evaluated by two annotators. Annotators assessed each hint using a five-point Likert scale (1: very low, 5: very high) for Relevance, Readability, Familiarity, Convergence, and Answer Leakage. Higher ratings are preferable for Relevance, Readability, Familiarity, and Convergence, whereas lower ratings indicate less Answer Leakage and are therefore preferable.

Annotators were instructed to evaluate each hint independently and assign ratings according to their subjective judgment of each dimension.

Hint-Assisted Question Answering. A separate group of thirty participants, consisting of students and researchers, took part in the hint-assisted question answering study. Questions were distributed among participants.

The study followed a three-stage process: **(1)** Participants first attempted to answer each question without using hints. If the answer was correct, they proceeded to the next question. **(2)** If the answer was incorrect or unknown, participants were shown the corresponding hints and could review them sequentially while attempting to solve the question. **(3)** If participants were still unable to answer after reviewing all available hints, they were allowed to skip the question.

The primary evaluation measure was QA accuracy before and after exposure to hints.

C Evaluation Methods and Sub-methods

HINTEVAL organizes hint evaluation into five dimensions, fifteen methods, and thirty-five sub-methods. The methods provide alternative ways to estimate each evaluation dimension, while sub-methods differ in model architecture, representation, provider, or preprocessing choice. Table 6 summarizes the complete mapping.

Because these approaches rely on different signals, methods targeting the same dimension may produce different scores and should be interpreted as complementary rather than interchangeable.

Relevance. Relevance measures the semantic relationship between a hint and its corresponding question. HINTEVAL supports four method fam-

Metric	Implemented Methods
Relevance	ROUGE-1, ROUGE-2, ROUGE-L, GloVe-6B, GloVe-42B, BERT-base, RoBERTa-large, HuggingFace LLMs, TogetherAI LLMs, OpenAI GPT models
Readability	Gunning-Fog Index, Flesch Reading Ease, Coleman-Liau Index, SMOG Index, Automated Readability Index, XGBoost, Random Forest, BERT-base, RoBERTa-large, HuggingFace LLMs, TogetherAI LLMs, OpenAI GPT models
Convergence	Specificity (BERT-base), Specificity (RoBERTa-large), BERT-base, RoBERTa-large, LLaMA 3.1 8B, LLaMA 3.1 70B
Familiarity	Word Frequency (with stop words), Word Frequency (without stop words), Wikipedia-based Familiarity
Answer Leakage	Lexical Overlap (with stop words), Lexical Overlap (without stop words), Contextual Similarity (with stop words), Contextual Similarity (without stop words)

Table 6: Evaluation methods currently implemented in HINTEVAL, grouped by evaluation metric.

ilies. ROUGE-1, ROUGE-2, and ROUGE-L capture lexical overlap at the unigram, bigram, and sequence levels, respectively. The non-contextual method instead uses fixed GloVe representations, with GloVe-6B and GloVe-42B providing two alternative embedding resources. Contextual methods use BERT-base and RoBERTa-large to capture semantic relationships beyond direct word overlap. In the reported evaluation, RoBERTa-large achieved higher MAP and MRR than BERT-base, while GloVe-42B slightly outperformed GloVe-6B. LLM-based variants provide another formulation in which models accessed through HuggingFace, TogetherAI, or OpenAI are used to assess question–hint relevance.

Readability. Readability measures how easily a hint can be understood. Traditional sub-methods include Gunning-Fog, Flesch Reading Ease, Coleman-Liau, SMOG, and Automated Readability Index, all of which estimate difficulty from surface-level textual characteristics. Machine-learning alternatives use XGBoost and Random Forest, while neural approaches use BERT-base and RoBERTa-large to predict readability from contextualized text representations. In the reported experiments, XGBoost performed slightly better than Random Forest, and RoBERTa-large achieved higher accuracy and F1 than BERT-base. LLM-based methods provide an additional option for directly judging the readability of a hint.

Convergence. Convergence captures how effectively a hint narrows the space of plausible answers. The specificity-based method uses BERT-base and RoBERTa-large to estimate how specific or general the information in a hint is. Neural convergence methods use the same model families but are trained directly on convergence values, making them conceptually different from the specificity pre-

dictors despite sharing similar architectures. In the reported evaluation, their predicted convergence values achieved Pearson correlations of 0.561 for BERT-base and 0.611 for RoBERTa-large with the ground-truth values. The LLM-based approach follows a different strategy by generating candidate answers and evaluating how strongly a hint reduces the set of plausible candidates.

Familiarity. Familiarity estimates how common or well-known the information expressed in a hint is. The word-frequency method uses corpus-based lexical frequency and is implemented both with and without stop words. The two variants share the same underlying formulation but differ in whether frequent functional words contribute to the resulting score. The Wikipedia-based method instead focuses on entities and uses the popularity of their corresponding Wikipedia pages as an indicator of familiarity. These approaches therefore capture different aspects of familiarity: lexical commonness and entity-level popularity.

Answer Leakage. Answer Leakage measures the extent to which a hint reveals its corresponding answer. The lexical method captures explicit word-level overlap between hints and answers and provides variants with and without stop words. The contextual method instead uses semantic representations, allowing it to detect similarity even when the hint and answer use different surface forms. Consequently, lexical and contextual approaches provide complementary views of answer disclosure, with the former emphasizing direct overlap and the latter capturing semantically related expressions.

Relationship among sub-methods. Sub-methods within the same method typically differ in a narrow modeling or preprocessing choice, such as transformer architecture, embedding resource, model size, provider, or stop-word handling. In contrast, different methods within the same evaluation dimension may rely on different signals. For example, Relevance can be estimated through lexical overlap, static embeddings, contextual representations, or LLM judgments, while Familiarity can be estimated through word frequency or Wikipedia-based entity popularity.

The correlation analysis in Section 4.2 focuses on relationships among the five high-level evaluation dimensions using Spearman’s ρ and Kendall’s τ , rather than agreement among individual sub-methods.