EdTec-ItemGen: Enhancing Retrieval-Augmented Item Generation Through Key Point Extraction

Alonso Palomino[1,2] **David Buschhüter**[1] **Roland Roller**[1]
Niels Pinkwart[1] **Benjamin Paaßen**[1,2]

[1] German Research Center for Artificial Intelligence (DFKI), Germany, `<first>.<last>@dfki.de`
[2] Bielefeld University, Germany, `<first>.<last>@techfak.uni-bielefeld.de`

Abstract

A major bottleneck in exam construction involves designing test items (i.e., questions) that accurately reflect key content from domain-aligned curricular materials. For instance, during formative assessments in vocational education and training (VET), exam designers must generate updated test items that assess student learning progress while covering the full breadth of topics in the curriculum. Large language models (LLMs) can partially support this process, but effective use requires careful prompting and task-specific understanding. We propose a new key point extraction method for retrieval-augmented item generation that enhances the process of generating test items with LLMs. We exhaustively evaluated our method using a TREC-RAG approach, finding that prompting LLMs with key content rather than directly using full curricular text passages significantly improves item quality regarding key information coverage by 8%. To demonstrate these findings, we release EdTec-ItemGen, a retrieval-augmented item generation demo tool to support item generation in education.

1 Introduction

A key challenge in educational measurement is to construct high-quality exam questions or “test items” that effectively differentiate varying levels of student competency. Assessment organizations rely on subject matter experts to extract essential content from domain-specific curriculum materials for item construction (Lane et al., 2016). Thus, generative natural language processing-based techniques for automated item generation (AIG) have gained interest in educational measurement to reduce the high costs and labor of manual test item creation (Circi et al., 2023; Kyllonen et al., 2024).

The widespread adoption of large language models (LLMs) has significantly encouraged employing generative NLP for AIG (Laverghetta Jr. and Licato, 2023; Gorgun and Bulut, 2024; Chan et al., 2025). Despite their impressive performance across various subtasks, LLMs often struggle with hallucinations, bias, and limited domain-specific knowledge, diminishing their effectiveness in specialized tasks (Zhang et al., 2023; Huber and Niklaus, 2025; Gonen et al., 2025). Retrieval augmented generation (RAG) (Lewis et al., 2020; Fan et al., 2024) offers a solution by integrating domain-relevant knowledge, enhancing customization, reducing hallucinations, and improving access to authoritative and up-to-date information.

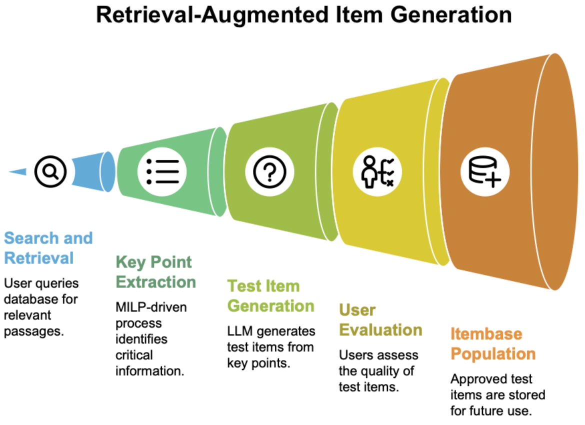

Retrieval-Augmented Item Generation
Search and Retrieval
User queries database for relevant passages.
Key Point Extraction
MILP-driven process identifies critical information.
Test Item Generation
LLM generates test items from key points.
User Evaluation
Users assess the quality of test items.
Itembase Population
Approved test items are stored for future use.

Figure 1: EdTec-ItemGen automates VET item generation by retrieving passages, extracting key points, and prompting an LLM to create test items.

To support bfz[1], one of the largest German Vocational Education and Training (VET) providers in manual item generation tasks, we deployed EdTec-ItemGen[2], a RAG platform for AIG, that leverages a new mixed-integer linear programming driven method (MILP-driven KPE) for key point extraction to assist educators in designing items for formative assessments.

Figure 1 summarizes the operational flow of

[1] `https://www.bfz.de/`

[2] System demonstration and code available at: `https://edtec-itemgen.xyz`

EdTec-ItemGen for supporting item generation tasks. The process begins with users retrieving relevant VET passages, pre-extracted from educational materials, using semantic search. An extractive summarization step then identifies the most salient factoids or "key points" through the proposed MILP-driven KPE approach and similarity scoring (see Section 3). These key points are subsequently used to instruct an LLM to generate new test items. Finally, test item designers evaluate and filter high-quality items via EdTec-ItemGen's user interface, which are later added to an internal item base for exam assembly and construction. Our work contributes as follows:

- Building on prior research on item retrieval for exam assembly and calibration (Palomino et al., 2024, 2025), we proposed a novel MILP-driven key point extraction method to enhance key information coverage on augmented generated items.
- An exhaustive evaluation and performance analysis, following the TREC-RAG 2024 evaluation approach (Pradeep et al., 2024a,b), demonstrating how the proposed key point extraction method enhances the retrieval-augmented item generation process (see Section 4).
- We deployed and released a fork multilingual system demonstration version of EdTec-ItemGen, our industry partner's RAG platform for AIG.

Section 3 presents our MILP-based key point extraction; Section 4 presents its evaluation under the TREC-RAG framework. Section 5 provides an application and demo system overview, and Section 6 concludes with future directions.

2 Related Work

To mitigate the complexity of manual test item construction, educators adopted automated approaches to simplify its development (Lane et al., 2016; Rudolph et al., 2019; Circi et al., 2023). Prior research in Automated Item Generation (AIG) transitioned from classic NLP methods, including shallow parsing, term and topic extraction, and the use of semantic resources like WordNet (Brown et al., 2005; Mitkov et al., 2006; Rus et al., 2011; Heilman and Smith, 2010; Chali and Hasan, 2015), to neural architectures including graph-neural networks and transformer-based models for AIG (Chan and Fan, 2019; Tuan et al., 2020; Qu et al., 2021; Yoshimi et al., 2023; Jahangir et al., 2024; Jamshidi and Chali, 2025).

The rise of LLMs has led educators to use prompt-based generative NLP for AIG (Dugan et al., 2022; Kyllonen et al., 2024); for example, Wu et al. (2024) propose a two-step multimodal framework that merges LLM-generated sub-questions about related entities into coherent items. Lin et al. (2024) proposed TASE-CoT, a few-shot method for type-aware semantic extraction of relevant item types and phrases to aid LLMs in generating refined items and answers requiring reasoning across multiple documents. Guo et al. (2024) generate knowledge-base questions by extracting a skeleton of interrogatives and auxiliaries from graph triples to steer GPT-3.5. Ashok Kumar and Lan (2024) fine-tune LLaMA-2 with negative Socratic example augmentation and direct preference optimization to boost programming-item validity.

Prior work includes Pochiraju et al. (2023), which maps sentences via ConceptNet/WordNet rules, and Guinet et al. (2024), which fine-tunes LLMs for exam generation and filters items by syntax, incorrectness, self-containment, and embedding similarity. Poon et al. (2024) show that few-shot LLM prompts yield more higher-order Chinese reading items than traditional methods, while Mucciaccia et al. (2025) combine role-based prompts, glossaries, one-shot examples, and chain-of-thought reasoning to generate and evaluate university-level items. Although LLM prompting is widespread in AIG, hallucinations, knowledge-reliability issues, and opaque reasoning persist without careful prompt design (Fan et al., 2024). To address these limitations, retrieval-augmented generation (RAG) emerged as a popular strategy to enhance generative NLP performance (Lewis et al., 2020; Fan et al., 2024).

This work introduces a novel key point extraction method to enhance the retrieval-augmented item generation process in German Vocational Education and Training (VET). We released a public fork of our industry partner's internal tool and APIs to demonstrate this. The closest studies are Pochiraju et al. (2023), which maps sentences via ConceptNet/WordNet rules, and Guinet et al. (2024), which fine-tunes LLMs for exam generation and filters items by length, incorrectness, self-containment, and embedding similarity. However, our simpler approach applies extractive summarization to VET passages, isolating key nuggets and preserving only essential content. Ultimately, we evaluated our approach with Pradeep et al. (2024a,b) framework.

3 Enhancing Augmented Item Generation Via Key Point Extraction

Key Point Extraction (KPE) is an extractive summarization approach that selects high-level factoid statements capturing the main aspects of a passage (Bar-Haim et al., 2020, 2021). Because item designers already distill such facts from domain-specific VET contents, we add KPE to our item-generation RAG pipeline to replicate this manual process.

MILP-driven KPE After splitting the input passages into sentences, we employed Jina-ColBERT[3] (Jha et al., 2024), an efficient multi-vector neural re-ranker model, to compute sentence embeddings and derive a similarity score for each pair of candidate sentences, ultimately producing a similarity matrix. Then, we apply a Mixed-Integer Linear Programming driven formulation (MILP-driven KPE) that aims to balance maximizing the relevance, while minimizing redundancy of a set of candidate sentences. Essentially, our goal is to subselect K candidate sentences with maximum relevance to the original input passage, such that their pairwise similarity is minimized. Assume we have m candidate sentences, each with a relevance score $r_1, \dots, r_m$ relative to the original input passage and a pairwise similarity matrix $S \in \mathbb{R}^{m\times m}$. Then, we aim to solve:

$$\begin{aligned} \min_{\vec{x}\in\{0,1\}^m} \quad & \lambda \cdot \vec{x}^T \cdot S \cdot \vec{x} - \vec{r}^T \cdot \vec{x} \\ \text{such that} \quad & \vec{1}^T \cdot \vec{x} = K, \end{aligned} \tag{1}$$

The objective in Eq. (1) formulates the task as a quadratic knapsack problem (Pisinger, 2007). It selects exactly K sentences, rewarding their individual relevance while penalizing pairwise similarity (redundancy), with λ controlling the relevance-redundancy trade-off. We linearize it as a MILP problem with Glover and Woolsey (1974) method:

$$\begin{aligned} \min_{\vec{x}\in\{0,1\}^m, \vec{z}\in\mathbb{R}^m} \quad & \lambda \cdot \vec{1}^T \cdot \vec{z} - \vec{r}^T \cdot \vec{x} \\ \text{such that} \quad & \vec{1}^T \cdot \vec{x} = K, \\ & l_i \cdot x_i \le z_i \le u_i \cdot x_i \quad \forall i \\ & \vec{s}_i^T \cdot \vec{x} - u_i \cdot (1 - x_i) \le z_i \quad \forall i \\ & z_i \le \vec{s}_i^T \cdot \vec{x} - l_i \cdot (1 - x_i) \quad \forall i \end{aligned} \tag{2}$$

[3] https://huggingface.co/jinaai/jina-colbert-v2

where $\vec{s}_i$ is the i-th column of S, z_i is a slack variable expressing $\sum_{j\neq i} s_{i,j} \cdot x_j$, $l_i = -\sum_{j=1}^{m} |s_{i,j}|$ is a lower-bound for z_i and $u_i = \sum_{j=1}^{m} |s_{i,j}|$ is an upper-bound for z_i. Eq. (2) replaces the quadratic term with linear constraints that add one slack variable per sentence, set to the sum of its similarities to the selected sentences, making the objective linear. By employing similarity scoring and linearized constrains, our method extracts key statements efficiently.

4 Experiments and Results

Although LLMs have been shown to be more effective as weak labelers when combined with custom models like DistilBART rather than for direct extractive summarization (Mishra et al., 2023), training these models requires high-quality ground truth data, careful prompt design, and challenges for optimizing task-specific objectives. Furthermore, LLM's reliance on superficial features like sentence position rather than distinguishing content importance may make LLMs encounter challenges in extractive summarization tasks (Zhang et al., 2023). Therefore, we hypothesize that applying MILP-driven KPE to instruct an LLM in item generation can significantly improve both information coverage and the quality of test items, while reducing the reliance on resource-intensive training iterations.

4.1 Dataset

As documents for retrieval, we utilized a sample corpus of 1,110 VET passages drawn from bfz's proprietary teaching materials, covering nine high-demand occupational VET topics relevant to the German job market. These topics range from "German Language Competence" and "Use of Technology" to "Storage and Logistics" and "Content Creation" (see Appendix A1 for more details).

4.2 Passage Retrieval

We employed a dense retrieval approach to model the retrieval step, building on prior research in item retrieval for exam assembly (Palomino et al., 2024, 2025). We used Reimers and Gurevych (2019) [4] embeddings with the `faiss` library (Douze et al., 2024) to efficiently perform approximate nearest neighbor search, with ten topic skill queries over the VET corpus. To ensure a realistic search and retrieval scenario, these queries were created using

[4] https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2

synonymic terms rather than the actual topic labels available in the VET corpus.

4.3 Augmented Item Generation

Given a ranked list of the top 25 relevant VET passages retrieved via dense retrieval, we investigated two setups:

1. **With MILP-driven KPE:** After extracting the top 15 key points using MILP-driven KPE, we prompted GPT-4o to generate a test item based on these points.
2. **Without MILP-driven KPE:** We instructed GPT-4o to directly derive the top 15 key statements from a given passage and then, with that information, generate a test item.

Overall, both setups were used prompts for instructing GPT-4o[5] to generate new test items (Refer to appendix A.2 for more details). While for setup (1), the underlying assumption is that the factoid extraction step will reveal the most important information leading to better prompts that produce better augmented generated items. For setup (2), the assumption is that a single prompt instructing the LLM to first identify the top key factoids from a given passage will be sufficient to generate higher-quality items that effectively cover the passage's most essential statements.

4.4 Evaluation Approach

To assess the performance of our experimental setups during the augmented item generation phase, we employed Pradeep et al. (2024a,b) TREC-RAG style automated ad-hoc evaluation approach. Based on Voorhees and Buckland (2003), the TREC-RAG evaluation approach involves using the AutoNuggetizer framework by harnessing an LLM to derive a concise set of factoid units that can be binary evaluated based on whether they contain either "Vital", "Ok" or "Not Vital" *information nuggets* required to address a given information need. For each passage in our VET corpus, we instructed GPT-4o to extract 30 information nuggets and subsequently select and label the top 20 most relevant ones as ground truth for final evaluation. While nuggets containing essential information for generating comprehensive items are labeled as "vital", nuggets with valuable yet non-essential information are labeled as "okay". Subsequently, we evaluated the augmented generated responses against the created nuggets by instructing GPT-4o to numerically determine whether each nugget was fully, partially, or not supported by each generated item response. We then computed the four metrics proposed by Pradeep et al. (2024a), providing an exhaustive evaluation of information coverage across the augmented test item responses produced by GPT-4o:

1. **Vital Strict (Vstrict):** Applies strict matching criteria, counting only for full support matches (i.e., 1.0 and 0.0 respectively).
2. **Vital (V):** Calculates the average score for nuggets labeled as "vital" using a scoring system with three levels (1.0 for full support, 0.5 for partial support, and 0 for no support).
3. **Weighted (W):** It assigns weights of 1.0 to vital nuggets and 0.5 to okay nuggets, then calculates the average by dividing the total vital nugget score by the sum of vital nuggets plus half the number of okay nuggets.
4. **All (A):** The average across all nuggets, both vital and okay, using the same three-level scoring system to assess the broadest measure of generated responses completeness.

Additionally, as a proxy to assess the quality of the augmented generated test items, we instructed GPT-4o to rate grammatical quality, readability, and succinctness on a scale from 1.0 to 5.0. We also measured the length of each item. While prompting GPT-4o we employed `instructor` library enforcing consistent prompt formatting and deterministic API calls fixing the temperature parameter to 0. Similarly to Voorhees and Buckland (2003) and Pradeep et al. (2024a), by combining the above metrics, we systematically quantified key factual coverage and clarity across augmented generated item responses.

4.5 Results

Based on Pradeep et al. (2024a,b) TREC-RAG framework, we evaluated the impact of integrating our new MILP-driven KPE method by assessing the information coverage and quality of generated test items. Table 1 summarizes the results metrics for the different levels of information coverage and item quality. From an information coverage perspective, regarding how well EdTec-ItemGen's produced items covering essential information necessary for generating good multiple-choice test items, when employing MILP-driven KPE, we observed an increased Vstrict score from 0.29 to 0.36 (e.g.,

[5] https://openai.com/index/hello-gpt-4o/

+0.07 absolute, +24.14% relative). Similarly, as for how well, on average, full and partial essential information nuggets were matched in the augmented generated response, we observed an absolute increase of 7% from 0.35 to 0.42 in the Vital (V) score. While evaluating by weighting the importance of ground truth nuggets based on their relevance (vital Vs. ok), we observed an improvement of 6% when using MILP-driven KPE extraction to instruct GPT-4o in creating multiple-choice test items. When considering all metrics, that is, when averaging Vstrict, V, and W scores, when employing MILP-driven KPE, we observed an average improvement of 8% over GPT-4o augmented generation responses. From an item quality perspective, we observed improvements in grammatical and readability scores when employing MILP-driven KPE, with relative increases of 2.7% and 2.73%, respectively, at the expense of succinctness, which decreased by 1.25%. Also, we noted an increase in the length of the generated item responses by 13.62%. Ultimately, we conducted a Wilcoxon significance test comparing setups observing significant differences ($p < 0.05$) on all metrics except succinctness ($p = 0.30$).

5 System Overview

EdTec-ItemGen aims to support educators in test item generation tasks at bfz, Germany's largest provider of vocational education and training (VET) services. Designing formative assessments is time-intensive, particularly when test items must cover specific curricular content. This process is slow and prone to errors. EdTec-ItemGen replaces the drafting process with retrieval-augmented item generation. A dense retriever searches from internal educational materials, while MILP-driven KPE extracts essential concepts to instruct an LLM in writing a candidate item. Educators then review each suggested item, rating clarity and difficulty while discarding spurious items. EdTec-ItemGen also serves as a crowdsourcing infrastructure, collecting user interactions to drive future research on generative retrieval models for question banks (see Appendix A2 for industry application details).

Frontend The platform interface, illustrated in Figure 3, implements the complete item-generation workflow in HTML and JavaScript. Fig. 3(a) enables users to query the pre-processed VET corpus and select an LLM version (GPT-4o-mini or GPT-3.5 in the demo fork), thus reducing dependence on a single model. An upload widget is also available, allowing custom file ingestion in CSV format. Fig. 3(b) shows retrieved passages with MILP-KPE key points highlighted, so users can assess content coverage. Fig. 3(c) displays the generated items: green marks accepted items, while red flags inconsistent ones. Fig. 3(d) presents the validation view, where users approve or reject the generated items before pushing them to the main item base for exam assembly.

Backend The backend architecture, shown in Figure 2, is implemented in Python. An asynchronous worker pool manages concurrent Flask API requests. Semantic search is handled using multilingual embeddings with the `FAISS` library. MILP-driven KPE employs `SciPy`'s native optimizer for efficient key point extraction to support retrieval-augmented item generation. Extracted key points are passed to an LLM to guide the retrieval-augmented item generation phase. Each request returns a JSON response with the ranked documents, extracted key points, and the LLM-generated item. User interactions from the interface are logged in the backend using SQLite and made accessible through an internal endpoint for later analysis and review. To extend applicability to other languages, we expose language-agnostic APIs that support the generation of multilingual items from user-custom data (see Appendix A3 for more details).

6 Conclusions

Collaborating with bfz, one of Germany's largest VET services providers, we explored employing retrieval-augmented generation to assist educators with manual item construction tasks. Although prompting LLMs with custom VET curricular materials is useful for rapid test item generation, we explored whether incorporating a new MILP-driven key point extraction (KPE) method can enhance prompting during the augmented item generation phase. We evaluated our new method under the TREC-RAG framework. Our evaluation indicates that MILP-driven KPE significantly enhances essential content coverage and item quality in retrieval-augmented generation from internal VET passages, improving LLM-based item generation performance on domain-specific curricular material. Specifically, when employing MILP-driven KPE, results improved across all metrics, namely Vstrict increased by 7%, (V) by 7%, (W) by 6%, and (A) by 8%, with relative gains ranging from

(a)

Metric	With	Without	Δ Abs.	Δ Rel. (%)
		Nugget Coverage		
Vital Strict (Vstrict)	0.36	0.29	**+0.07 ↑**	**+24.14**
Vital (V)	0.42	0.35	**+0.07 ↑**	**+20.00**
Weighted (W)	0.38	0.32	**+0.06 ↑**	**+18.75**
All (A)	0.40	0.32	**+0.08 ↑**	**+25.00**
		Test Item Quality		
Grammar Score	4.94	4.81	**+0.13 ↑**	**+2.70**
Succinctness Score	3.95	4.00	**-0.05 ↓**	**-1.25**
Readability Score	4.90	4.77	**+0.13 ↑**	**+2.73**
Response Length (L)	32.28	28.41	**+3.87 ↑**	**+13.62**

(b)

Metric	W Stat.	p-value	Sig.
Vital Strict (Vstrict)	1900.5	0.021	Yes
Vital (V)	3284.5	0.032	Yes
Weighted (W)	3923.0	0.026	Yes
All (A)	4644.5	0.0003	Yes
Grammar Score	239.0	0.0029	Yes
Succinctness Score	1498.5	0.306	No
Readability Score	559.5	0.0026	Yes
Response Length (L)	9285.5	0.0009	Yes

Table 1: System performance with and without MILP-driven KPE. (a) Metrics for both settings and their Δ absolute and relative differences. (b) Wilcoxon signed-rank test comparing both setups ($p < 0.05$ indicates significance).

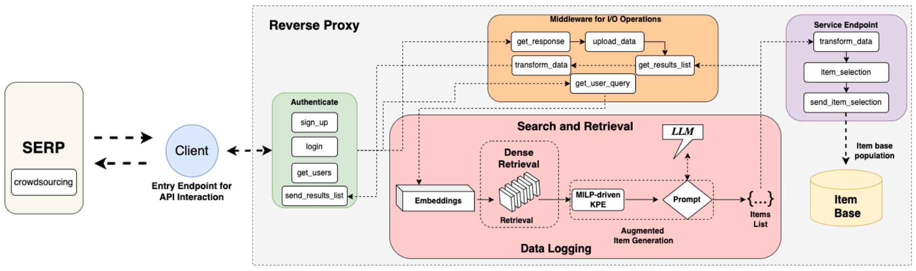
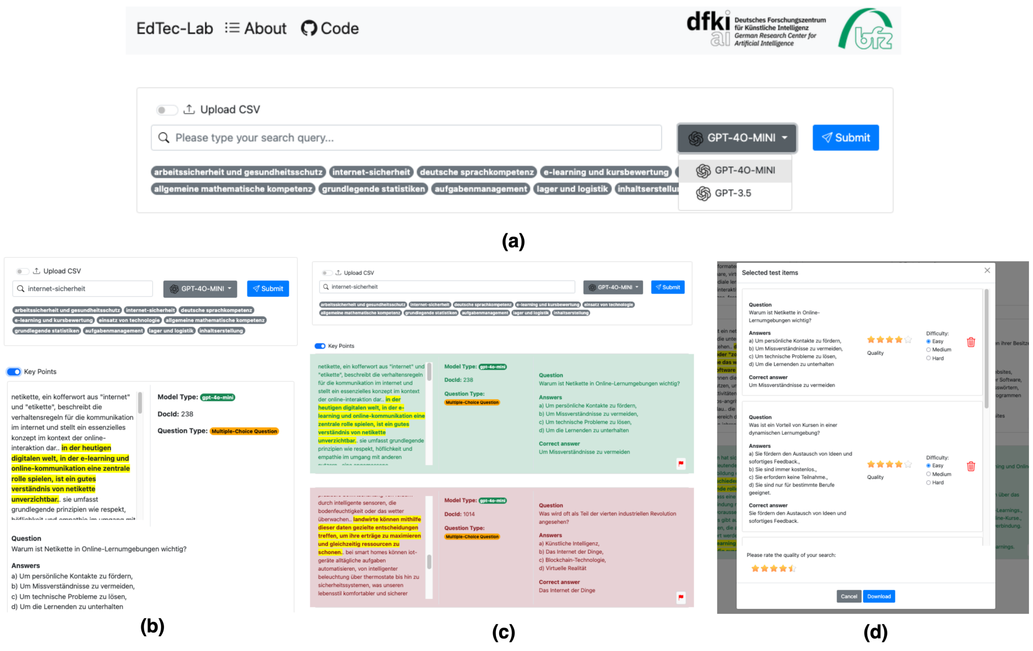


Figure 2: Demo architecture of EdTec-ItemGen, illustrating client interactions, API endpoints, retrieval and augmented item generation method, and integration with data logging, and external data repositories.



Question
Warum ist Netikette in Online-Lernumgebungen wichtig?

Answers
a) Um persönliche Kontakte zu fördern,
b) Um Missverständnisse zu vermeiden,
c) Um technische Probleme zu lösen,
d) Um die Lernenden zu unterhalten

Correct answer
Um Missverständnisse zu vermeiden

Question
Was wird oft als Teil der vierten industriellen Revolution angesehen?

Answers
a) Künstliche Intelligenz,
b) Das Internet der Dinge,
c) Blockchain-Technologie,
d) Virtuelle Realität

Correct answer
Das Internet der Dinge

Figure 3: EdTec-ItemGen: Demo Version Frontend and User Interface Overview.

18.75% to 25%. In terms of item quality, KPE significantly improves grammar and readability by 2.70% and 2.73%, respectively, while increasing item length by 13.62%. Overall, our approach enhances the coverage of essential curricular content and improves the clarity of language in test items.

Future work will analyze session logs to research how users generate and evaluate items using our method. We plan to further investigate how to employ and integrate linear constraint-based test-assembly models (Linden et al., 2005) with LLMs to control and generate different test item types (e.g., matching, cloze) to expand on item's psychometric coverage beyond recall.

Limitations and Ethics Statement

For this work, we maintained strict confidentiality to protect our partner's product and intellectual property, in full compliance with required privacy standards. Although EdTec-ItemGen effectively supports VET educators in retrieval-augmented item generation tasks using domain-specific curricular contents, some limitations remain. For instance, we employed an ad-hoc TREC RAG-style evaluation to transfer a usable platform to our partners rapidly. This was useful for designing, assessing, and deploying our platform under cold start conditions to our partner's use case. Nevertheless, our TREC-RAG style approach (Pradeep et al., 2024a,b) relies on synthetic relevance judgments, which have recently gained traction in the information retrieval community (Faggioli et al., 2023; He et al., 2024). Still, real human VET expert evaluations naturally provide more accurate measures of augmented-generated item quality. Nevertheless, our platform allows users to annotate item quality or difficulty during the generation process to address this limitation. In the future, we plan to integrate these real human annotations into our evaluation approach, thereby enhancing the reliability of our partner's item generation workflow.

Acknowledgements

This work was funded by the Federal Ministry of Research, Technology and Space (BMFTR) as part of the AZUKIT project (funding code 21IVP056C) and by the KI-Akademie OWL project (grant no. 01IS24057A). The AZUKIT project is run under the innovation competition InnoVET PLUS, implemented by the Federal Institute for Vocational Education and Training (BIBB), while KI-Akademie OWL is supported by the Project Management Agency of the German Aerospace Centre (DLR). The responsibility for the content lies with the authors.

References

Nischal Ashok Kumar and Andrew Lan. 2024. Improving socratic question generation using data augmentation and preference optimization. In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, pages 108–118, Mexico City, Mexico. Association for Computational Linguistics.

Roy Bar-Haim, Lilach Eden, Roni Friedman, Yoav Kantor, Dan Lahav, and Noam Slonim. 2020. From arguments to key points: Towards automatic argument summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4029–4039, Online. Association for Computational Linguistics.

Roy Bar-Haim, Lilach Eden, Yoav Kantor, Roni Friedman, and Noam Slonim. 2021. Every bite is an experience: Key Point Analysis of business reviews. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3376–3386, Online. Association for Computational Linguistics.

Jonathan Brown, Gwen Frishkoff, and Maxine Eskenazi. 2005. Automatic question generation for vocabulary assessment. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 819–826, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Yllias Chali and Sadid A. Hasan. 2015. Towards topic-to-question generation. *Computational Linguistics*, 41(1):1–20.

Kuang Wen Chan, Farhan Ali, Joonhyeong Park, Kah Shen Brandon Sham, Erdalyn Yeh Thong Tan, Francis Woon Chien Chong, Kun Qian, and Guan Kheng Sze. 2025. Automatic item generation in various stem subjects using large language model prompting. *Computers and Education: Artificial Intelligence*, 8:100344.

Ying-Hong Chan and Yao-Chung Fan. 2019. A recurrent BERT-based model for question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 154–162, Hong Kong, China. Association for Computational Linguistics.

Ruhan Circi, Juanita Hicks, and Emmanuel Sikali. 2023. Automatic item generation: foundations and machine learning-based approaches for assessments. In *Frontiers in Education*, volume 8, page 858273. Frontiers Media SA.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library.

Liam Dugan, Eleni Miltsakaki, Shriyash Upadhyay, Etan Ginsberg, Hannah Gonzalez, DaHyeon Choi, Chuning Yuan, and Chris Callison-Burch. 2022. A feasibility study of answer-agnostic question generation for education. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1919–1926, Dublin, Ireland. Association for Computational Linguistics.

Guglielmo Faggioli, Laura Dietz, Charles LA Clarke, Gianluca Demartini, Matthias Hagen, Claudia Hauff, Noriko Kando, Evangelos Kanoulas, Martin Potthast, Benno Stein, et al. 2023. Perspectives on large language models for relevance judgment. In *Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 39–50.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 6491–6501, New York, NY, USA. Association for Computing Machinery.

Fred Glover and Eugene Woolsey. 1974. Technical note—converting the 0-1 polynomial programming problem to a 0-1 linear program. *Operations Research*, 22(1):180–182.

Hila Gonen, Terra Blevins, Alisa Liu, Luke Zettlemoyer, and Noah A. Smith. 2025. Does liking yellow imply driving a school bus? semantic leakage in language models. In *Proceedings of the 2025 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Albuquerque, New Mexico, USA. Association for Computational Linguistics.

Guher Gorgun and Okan Bulut. 2024. Instruction-tuned large-language models for quality control in automatic item generation: A feasibility study. *Educational Measurement: Issues and Practice*.

Gauthier Guinet, Behrooz Omidvar-Tehrani, Anoop Deoras, and Laurent Callot. 2024. Automated evaluation of retrieval-augmented language models with task-specific exam generation. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Shasha Guo, Lizi Liao, Jing Zhang, Yanling Wang, Cuiping Li, and Hong Chen. 2024. SGSH: Stimulate large language models with skeleton heuristics for knowledge base question generation. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4613–4625, Mexico City, Mexico. Association for Computational Linguistics.

Gurobi Optimization, LLC. 2024. Gurobi Optimizer Reference Manual.

Xingwei He, Zhenghao Lin, Yeyun Gong, A-Long Jin, Hang Zhang, Chen Lin, Jian Jiao, Siu Ming Yiu, Nan Duan, and Weizhu Chen. 2024. AnnoLLM: Making large language models to be better crowdsourced annotators. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, pages 165–190, Mexico City, Mexico. Association for Computational Linguistics.

Michael Heilman and Noah A. Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617, Los Angeles, California. Association for Computational Linguistics.

Thomas Huber and Christina Niklaus. 2025. LLMs meet bloom‘s taxonomy: A cognitive view on large language model evaluations. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 5211–5246, Abu Dhabi, UAE. Association for Computational Linguistics.

Khushnur Jahangir, Philippe Muller, and Chloé Braud. 2024. Complex question generation using discourse-based data augmentation. In *Proceedings of the 5th Workshop on Computational Approaches to Discourse (CODI 2024)*, pages 105–119, St. Julians, Malta. Association for Computational Linguistics.

Samin Jamshidi and Yllias Chali. 2025. GNET-QG: Graph network for multi-hop question generation. In *Proceedings of the Workshop on Generative AI and Knowledge Graphs (GenAIK)*, pages 20–26, Abu Dhabi, UAE. International Committee on Computational Linguistics.

Rohan Jha, Bo Wang, Michael Günther, Georgios Mastrapas, Saba Sturua, Isabelle Mohr, Andreas Koukounas, Mohammad Kalim Wang, Nan Wang, and Han Xiao. 2024. Jina-ColBERT-v2: A general-purpose multilingual late interaction retriever. In *Proceedings of the Fourth Workshop on Multilingual Representation Learning (MRL 2024)*, pages 159–166, Miami, Florida, USA. Association for Computational Linguistics.

Patrick Kyllonen, Amit Sevak, Teresa Ober, Ikkyu Choi, Jesse Sparks, and Daniel Fishtein. 2024. Charting the future of assessments. *ETS Research Report Series*, 2024(1):1–62.

Suzanne Lane, Mark R Raymond, and Thomas M Haladyna. 2016. *Handbook of test development*. Routledge.

Antonio Laverghetta Jr. and John Licato. 2023. Generating better items for cognitive assessments using large language models. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 414–428, Toronto, Canada. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Manling Li, Shiyu Zhao, Qineng Wang, Kangrui Wang, Yu Zhou, Sanjana Srivastava, Cem Gokmen, Tony Lee, Erran Li Li, Ruohan Zhang, et al. 2024. Embodied agent interface: Benchmarking llms for embodied decision making. *Advances in Neural Information Processing Systems*, 37:100428–100534.

Zefeng Lin, Weidong Chen, Yan Song, and Yongdong Zhang. 2024. Prompting few-shot multi-hop question generation via comprehending type-aware semantics. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3730–3740, Mexico City, Mexico. Association for Computational Linguistics.

Wim J Linden et al. 2005. *Linear models for optimal test design*. Springer.

Nishant Mishra, Gaurav Sahu, Iacer Calixto, Ameen Abu-Hanna, and Issam Laradji. 2023. LLM aided semi-supervision for efficient extractive dialog summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10002–10009, Singapore. Association for Computational Linguistics.

Ruslan Mitkov, Ha Le An, and Nikiforos Karamanis. 2006. A computer-aided environment for generating multiple-choice test items. *Natural language engineering*, 12(2):177–194.

Sérgio Silva Mucciaccia, Thiago Meireles Paixão, Filipe Wall Mutz, Claudine Santos Badue, Alberto Ferreira de Souza, and Thiago Oliveira-Santos. 2025. Automatic multiple-choice question generation and evaluation systems based on LLM: A study case with university resolutions. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 2246–2260, Abu Dhabi, UAE. Association for Computational Linguistics.

Alonso Palomino, Andreas Fischer, David Buschhüter, Roland Roller, Niels Pinkwart, and Benjamin Paassen. 2025. Mitigating bias in item retrieval for enhancing exam assembly in vocational education services. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: Industry Track)*, pages 183–193, Albuquerque, New Mexico. Association for Computational Linguistics.

Alonso Palomino, Andreas Fischer, Jakub Kuzilek, Jarek Nitsch, Niels Pinkwart, and Benjamin Paassen. 2024. EdTec-QBuilder: A semantic retrieval tool for assembling vocational training exams in German language. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: System Demonstrations)*, pages 26–35, Mexico City, Mexico. Association for Computational Linguistics.

Laurent Perron and Frédéric Didier. Cp-sat.

David Pisinger. 2007. The quadratic knapsack problem—a survey. *Discrete Applied Mathematics*, 155(5):623–648.

Dhanamjaya Pochiraju, Abhinav Chakilam, Premchand Betham, Pranav Chimulla, and S Govinda Rao. 2023. Extractive summarization and multiple choice question generation using xlnet. In *2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 1001–1005.

Yin Poon, John Sie Yuen Lee, Yu Yan Lam, Wing Lam Suen, Elsie Li Chen Ong, and Samuel Kai Wah Chu. 2024. Few-shot question generation for reading comprehension. In *Proceedings of the 10th SIGHAN Workshop on Chinese Language Processing (SIGHAN-10)*, pages 21–27, Bangkok, Thailand. Association for Computational Linguistics.

Ronak Pradeep, Nandan Thakur, Sahel Sharifymoghaddam, Eric Zhang, Ryan Nguyen, Daniel Campos, Nick Craswell, and Jimmy Lin. 2024b. Ragnarök: A reusable rag framework and baselines for trec 2024 retrieval-augmented generation track. *arXiv preprint arXiv:2406.16828*.

Ronak Pradeep, Nandan Thakur, Shivani Upadhyay, Daniel Campos, Nick Craswell, and Jimmy Lin. 2024a. Initial nugget evaluation results for the trec 2024 rag track with the autonuggetizer framework. *arXiv preprint arXiv:2411.09607*.

Fanyi Qu, Xin Jia, and Yunfang Wu. 2021. Asking questions like educational experts: Automatically generating question-answer pairs on real-world examination data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2583–2593, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Aninditha Ramesh, Arav Agarwal, Jacob Arthur Doughty, Ketan Ramaneti, Jaromir Savelka, and Majd Sakr. 2024. A benchmark for testing the capabilities of llms in assessing the quality of multiple-choice questions in introductory programming education. In *Proceedings of the 2024 on ACM Virtual Global Computing Education Conference V. 1*, SIGCSE Virtual 2024, page 193–199, New York, NY, USA. Association for Computing Machinery.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Michael J Rudolph, Kimberly K Daugherty, Mary Elizabeth Ray, Veronica P Shuford, Lisa Lebovitz, and Margarita V DiVall. 2019. Best practices related to examination item construction and post-hoc review. *American journal of pharmaceutical education*, 83(7):7204.

Vasile Rus, Paul Piwek, Svetlana Stoyanchev, Brendan Wyse, Mihai Lintean, and Cristian Moldovan. 2011. Question generation shared task and evaluation challenge: Status report. In *Proceedings of the 13th European Workshop on Natural Language Generation*, ENLG '11, page 318–320, USA. Association for Computational Linguistics.

Luu Anh Tuan, Darsh Shah, and Regina Barzilay. 2020. Capturing greater context for question generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):9065–9072.

Ellen M Voorhees and L Buckland. 2003. Overview of the trec 2003 question answering track. In *TREC*, volume 2003, pages 54–68.

Ian Wu, Sravan Jayanthi, Vijay Viswanathan, Simon Rosenberg, Sina Khoshfetrat Pakazad, Tongshuang Wu, and Graham Neubig. 2024. Synthetic multimodal question generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 12960–12993, Miami, Florida, USA. Association for Computational Linguistics.

Jie Jw Wu and Fatemeh H. Fard. 2025. Humaneval-comm: Benchmarking the communication competence of code generation for llms and llm agent. *ACM Trans. Softw. Eng. Methodol.* Just Accepted.

Nana Yoshimi, Tomoyuki Kajiwara, Satoru Uchida, Yuki Arase, and Takashi Ninomiya. 2023. Distractor generation for fill-in-the-blank exercises by question type. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 276–281, Toronto, Canada. Association for Computational Linguistics.

Haopeng Zhang, Xiao Liu, and Jiawei Zhang. 2023. Extractive summarization via ChatGPT for faithful summary generation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3270–3278, Singapore. Association for Computational Linguistics.

A Appendix

A.1 Dataset Details

Table 2 summarizes the frequency across these topics, including the top three terms, unique terms, sentence counts, top terms, and document counts.

Topic	Avg. Terms	Uniq. Terms	Sent. Count	Top 3 Terms	Docs
General Mathematical Competence	324.4	223	38.3	cross multiplication understanding mathematical	70
Occupational Safety & Health Protection	323.2	240	36.8	employees safety workplace	64
Task Management	323.8	239	37.9	tasks task management important	112
German Language Competence	306.5	227	35.1	education professional significance	234
E-learning & Course Evaluation	320.1	244	37.4	learners education learning	195
Use of Technology	333.8	252	36.3	technology technologies education	92
Basic Statistics	328.3	226	36.7	data values median	83
Content Creation	332.5	244	37.1	content information education	61
Storage & Logistics	310.4	238	36.1	company logistics storage	199
Total	318.7	237	36.6	learners education professional	1,110

Table 2: Key statistics across topics, including average terms, unique terms, sentence counts, key terms, and document counts.

The VET corpus is a domain-specific collection curated to support manual test item design for formative assessments. Each passage delivers authentic trade skill content specifically curated to the context of vocational education and training (VET) in Germany.

A.2 Industry Application

As traditional test item construction requires multiple manual review cycles, AIG has become a key capability for educational and assessment institutions. Generating items that are clear, readable, and aligned with curricular materials is essential for valid assessment in high-stakes contexts. LLMs offer scalable item generation while reducing manual effort (Kyllonen et al., 2024). However, their integration is limited by issues such as hallucinations, lack of domain expertise, and restricted access to private sources (Li et al., 2024; Ramesh et al., 2024; Wu and Fard, 2025).

Retrieval-augmented generation (RAG) addresses these challenges by enabling LLMs to incorporate external knowledge for domain-specific item generation. In collaboration with bfz, a major German VET provider, we developed EdTec-ItemGen-a RAG-based platform supporting VET educators in creating and updating internal item repositories. Building on prior work in item retrieval for exam assembly (Palomino et al., 2024, 2025), we integrated a MILP-driven approach for key point extraction (KPE) to identify essential content from VET passages. As illustrated in Figure 3, EdTec-ItemGen presents search results retrieved via dense semantic search. The MILP-based KPE module extracts key information used to prompt the LLM, which then generates a candidate item. Users review the generated items and decide which should be retained for inclusion in the item base.

A.3 System and API Overview Details

This section outlines the frontend and backend components of EdTec-ItemGen.

A.3.1 Frontend

Figure 3 displays the demo interface, which supports a human-in-the-loop workflow for LLM-based retrieval-augmented item generation.

A.3.2 Backend

The backend is built using the Flask[6] framework, exposing RESTful endpoints for search, CSV upload, and item generation. Python's concurrency library enables parallel execution across pipeline components. Deployment uses Nginx[7] as a reverse proxy and runs on an Amazon EC2[8] instance with

[6]https://flask.palletsprojects.com/en/stable/
[7]https://nginx.org/
[8]https://aws.amazon.com/en/ec2/

three vCPUs for cost-effective performance only on the system demonstration version. Gunicorn is used as the WSGI server for robust process and connection management.

The platform supports multilingual input via GPT-4. It detects the CSV language and prompts the LLM accordingly during item generation. Public APIs are available at `https://api.edtec-itemgen.xyz/search`.

For large-scale scenarios, such as extensive item banks, performance is maintained through parallelization and solvers like OR-Tools (Perron and Didier) or Gurobi (Gurobi Optimization, LLC, 2024).

Below are `curl`[9] examples for API usage:

1) Upload Data Endpoint Upload a CSV file. The response returns an `ID` used for subsequent requests. *Note:* User data is not stored permanently; logs are cleared post-upload.

```
curl -X POST \
 -F "file=@/path/to/your/file/example.csv" \
 https://api.edtec-itemgen.xyz/upload_csv
```

2) Item-RAG Endpoint Use the `upload_id` to generate items:

```
curl -X POST \
 -H "Content-Type: application/json" \
 -d '{
 "query": "safety",
 "upload_id": "<your file ID>",
 "k": 25,
 "llm_version": "GPT-4O-MINI",
 "kpe": "true"
 }' \
 https://api.edtec-itemgen.xyz/search
```

This retrieves relevant content from the uploaded CSV and generates multiple-choice items accordingly.

A.4 Prompt Details

Our prompting approach relied on the `instructor`[10] library, ensuring consistency and control over output formatting and prompt instructions. The `instructor` output was parameterized as JSON, thereby avoiding any output misformatting.

[9] https://curl.se
[10] https://python.useinstructor.com/

Goal	Prompt in German	Translated Prompt in English
1. Clarity Evaluation – Instruction's Context	Du bist ein Experte für die Bewertung der Qualität von Prüfungsfragen. Gib ausschließlich ein JSON-Objekt mit den Schlüsseln: "grammar_score" (Ganzzahl 1 bis 5), "succinctness_score" (Ganzzahl 1 bis 5), "readability_score" (Ganzzahl 1 bis 5), "explanation" (kurze Begrundung). Beispielformat: {"grammar_score": 4, "succinctness_score": 3, "readability_score": 5, "explanation": "Die Frage ist grammatisch gut, sehr klar..."}. Kein zusätzlicher Text oder andere Schlüssel.	You are an expert in evaluating the quality of exam questions. Provide only a JSON object with the following keys: "grammar_score" (an integer from 1 to 5), "succinctness_score" (an integer from 1 to 5), "readability_score" (an integer from 1 to 5), "explanation" (a brief explanation). Example format: {"grammar_score": 4, "succinctness_score": 3, "readability_score": 5, "explanation": "The question is grammatically sound and very clear..."}. No additional text or keys.
2. Clarity Evaluation – User Request	Bewerten Sie die folgende Multiple-Choice-Prüfungsfrage in deutscher Sprache: {question}. Bewerten Sie sie auf einer Skala von 1,0 bis 5,0 für die folgenden Kriterien: – Grammatik (grammar_score): Bewerten Sie die grammatikalische Richtigkeit. – Prägnanz (succinctness_score): Beurteilen Sie, wie prägnant und direkt die Frage ist. – Lesbarkeit (readability_score): Beurteilen Sie, wie leicht die Frage zu verstehen ist. Die Ergebnisse werden im folgenden JSON-Format zurückgegeben: {"grammar_score": <score>, "succinctness_score": <score>, "readability_score": <score>, "explanation": <Kurzerläuterung zu den angegebenen Punktzahlen>}.	Evaluate the following multiple-choice exam question in German language: {question}. Rate it on a scale from 1.0 to 5.0 for the following criteria: – Grammar (grammar_score): Assess grammatical accuracy. – Conciseness (succinctness_score): Evaluate how concise and direct the question is. – Readability (readability_score): Judge how easy it is to understand. Return the results in the following JSON format: {"grammar_score": <score>, "succinctness_score": <score>, "readability_score": <score>, "explanation": <brief explanation for the given scores>}.
3. Nugget Coverage Scoring – Instruction's Context	Sie sind ein Experte für die Bewertung der Qualität von Multiple-Choice-Prüfungsfragen basierend auf vorab bewerteten, gekennzeichneten Informationsnuggets – relevante Fakten, die die Frage und ihre Antworten abdecken müssen. Befolgen Sie diese Richtlinien zur Berechnung der Metriken: 1. Weisen Sie Werte zu: - support = 1, - partial_support = 0,5, - not_support = 0. 2. Metriken: - A (All) Score: Durchschnitt aller Nugget-Werte. - V (Vital) Score: Durchschnitt der Werte für Nuggets, die als "Vital" gekennzeichnet sind. - W (Gewichteter) Score: Gewichteter Durchschnitt, wobei "Vital"-Nuggets ein Gewicht von 1 und "OK"-Nuggets ein Gewicht von 0,5 haben. - Vstrict (Vital Strict) Score: Durchschnitt für "Vital"-Nuggets, wobei nur support mit 1 gewertet wird (partial_support = 0). Geben Sie Ihre Antwort als gültiges JSON-Objekt mit den folgenden Schlüsseln an: {"Vstrict_GPT": <Score>, "V_GPT": <Score>, "W_GPT": <Score>, "A_GPT": <Score>"}.	You are an expert in evaluating multiple-choice exam questions based on pre-assessed, labeled information nuggets – relevant factoids that the question and its answers must cover. Follow these guidelines to calculate the metrics: 1. Assign Values: - support = 1, - partial_support = 0.5, - not_support = 0. 2. Metrics: - A (All) Score: Average of all nugget values. - V (Vital) Score: Average of values for nuggets labeled as "Vital." - W (Weighted) Score: Weighted average where "Vital" nuggets have a weight of 1 and "OK" nuggets a weight of 0.5. - Vstrict (Vital Strict) Score: Average for "Vital" nuggets, counting only support as 1 (partial_support = 0). Provide your answer as a valid JSON object with the following keys: {"Vstrict_GPT": <score>, "V_GPT": <score>, "W_GPT": <score>, "A_GPT": <score>"}.
4. Nugget Coverage Scoring – User Request	Bewerten Sie die folgende Kombination aus Frage und Kandidatenantwort: {question} {candidate_answer}. Liste der Nuggets (Format: Nugget: <Text>, Wichtigkeit: <Vital/OK>): {nuggets_text}. Für jedes Nugget ist zu bewerten, wie gut es durch die kombinierte Fragestellung und Antwort abgedeckt wird. Weisen Sie jedem Nugget eine der folgenden Abdeckungsstufen zu: - support: Das Nugget wird vollständig oder klar abgedeckt. - partial_support: Das Nugget wird teilweise abgedeckt, aber es fehlt an vollständiger Abdeckung. - not_support: Das Nugget wird gar nicht abgedeckt. Antworten Sie mit einem gültigen JSON-Objekt im folgenden Format: {"nugget_coverage": [{"nugget": <Text>, "coverage": <support/partial_support/not_support>}, ...]}.	Evaluate the following combination of question and candidate answer: {question} {candidate_answer}. List of Nuggets (Format: Nugget: <Text>, Importance: <Vital/OK>): {nuggets_text}. For each nugget, assess how well it is covered by the combined question and answer. Assign one of the following coverage levels to each nugget: - support: The nugget is fully addressed or clearly covered. - partial_support: The nugget is partially addressed but lacks full coverage. - not_support: The nugget is not addressed at all. Respond with a valid JSON object in the following format: {"nugget_coverage": [{"nugget": <Text>, "coverage": <support/partial_support/not_support>}, ...]}.
5. Augmented Item Generation (Based on Key Points)	Ihre Aufgabe ist es, eine Multiple-Choice-Frage zu erstellen, die auf den Top 15 Schlüsselaussagen {kp} basiert. Die Ausgabe sollte ein JSON-Objekt im Format sein: {"question": <Question text>, "answers": ["Antwort 1", "Antwort 2", "Antwort 3", "Antwort 4"], "correct_answer": <richtige Antwort>}. Die Frage muss so formuliert sein, dass sie nur mit den angegebenen Antwortmöglichkeiten beantwortet werden kann und nur eine richtige Antwort existiert. Speichern Sie die richtige Antwort im JSON-Feld "correct_answer".	Create a multiple-choice question based on the top 15 key statements {kp}. The output should be a JSON object in the format: {"question": <Question text>, "answers": ["Answer 1", "Answer 2", "Answer 3", "Answer 4"], "correct_answer": <correct answer>}. The question must be formulated so that it can only be answered based on the provided answer options and has only one correct answer. Store the correct answer in the JSON field "correct_answer".
6. Augmented Item Generation (Based on Full Text)	Ihre Aufgabe ist es, eine Multiple-Choice-Frage zu erstellen, die auf den 15 wichtigsten Aussagen des folgenden Textes {input_text} basiert. Die Ausgabe sollte ein JSON-Objekt im Format sein: {"question": <Question text>, "answers": ["Antwort 1", "Antwort 2", "Antwort 3", "Antwort 4"], "correct_answer": <richtige Antwort>}. Die Frage muss so formuliert sein, dass sie nur anhand der verfügbaren Antwortmöglichkeiten beantwortet werden kann und nur eine richtige Antwort besitzt. Speichern Sie die richtige Antwort im JSON-Feld "correct_answer".	Create a multiple-choice question based on the top 15 key statements of the following text {input_text}. The output should be a JSON object in the format: {"question": <Question text>, "answers": ["Answer 1", "Answer 2", "Answer 3", "Answer 4"], "correct_answer": <correct answer>}. The question must be formulated so that it can only be answered based on the provided answer options and has only one correct answer. Store the correct answer in the JSON field "correct_answer".
7. AutoNuggetizer – System Instruction for Nugget Extraction	Sie sind ein Experte für Bildungsinhalte. Ihre Aufgabe ist es, wesentliche Aussagen oder Schlüsselinformationen ("nuggets") – Faktengrundlagen oder Aussagen, die aus dem bereitgestellten Inhalt abgeleitet wurden – zu extrahieren. Antworten Sie mit einem gültigen JSON-Objekt, das genau die folgenden Schlüssel enthält: {keys_str}. Jeder Schlüssel muss einer Liste mit genau {nuggets_per_category} Elementen zugeordnet sein. Jedes Element muss ein JSON-Objekt mit folgender Struktur sein: {"nugget": <Die extrahierte Schlüsselaussage oder Information>, "importance": <Relevanzniveau: 'Vital', 'OK' oder 'Not Vital'>, "source_docid": <Dokument-ID>}.	Extract essential statements or key information ("nuggets") – factoids or statements derived from the provided content – using a valid JSON object containing exactly the following keys: {keys_str}. Each key must be associated with a list of exactly {nuggets_per_category} elements. Each element should be a JSON object with the following structure: {"nugget": <The extracted key fact or statement>, "importance": <Relevance level: 'Vital', 'OK', or 'Not Vital'>, "source_docid": <Document ID>}.
8. AutoNuggetizer – User Request for Nugget Extraction	Gegebenen Inhalt: {passage}. Antworte ausschließlich mit einem JSON-Objekt, das genau die folgenden Schlüssel enthält: {keys_str}. Jeder Schlüssel muss einer Liste mit genau {nuggets_per_category} Elementen zugeordnet sein. Jeder Eintrag in der Liste muss ein Objekt mit den Schlüsseln "nugget", "importance" und "source_docid" sein.	Given this content: {passage}. Respond solely with a JSON object that contains exactly the following keys: {keys_str}. Each key must have a list with exactly {nuggets_per_category} elements. Each entry in the list must be an object with the keys "nugget", "importance", and "source_docid".

Table 3: Employed prompts with original German texts and their English corresponding translations.