HireIQ: AI-powered Aptitude Test Generator For Corporate Recruitment

This Research Project report is submitted to the Department of Computer Science as partial fulfillment of Master of Science in Computer/Data Science degree

by

Abdul Qadeer Khan

Supervised by
Maria Rahim Khowaja
Lecturer
Department of Computer Science
School of Mathematics and Computer Science (SMCS)
Institute of Business Administration (IBA), Karachi

Fall Semester 2024
Institute of Business Administration (IBA), Karachi, Pakistan

HireIQ: AI-powered Aptitude Test Generator For Corporate Recruitment

This Research Project report is submitted to the Department of Computer Science as partial fulfillment of Master of Science in Computer/Data Science degree

by

Abdul Qadeer Khan

Maria Rahim _____
Supervisor
Lecturer
Computer Science –SMCS
Institute of Business Administration (IBA), Karachi

Fall 2024
Institute of Business Administration (IBA), Karachi, Pakistan

Dedication

This project is dedicated to my family, whose unwavering support, love, and sacrifices have been the cornerstone of my journey throughout my master's degree. Their belief in my abilities has been my constant source of motivation, inspiring me to persevere through challenges and achieve my goals.

To my spouse, her endless patience, understanding, and encouragement during this academic journey have been my greatest strength. Her unwavering support has been a beacon of hope during the most challenging moments, and for that, I am eternally grateful.

To my mother and siblings, thankful for their steadfast belief in my ambitions and for always cheering me on. Their encouragement has fueled my determination and provided the confidence I needed to excel.

Lastly, I dedicate this work to the pursuit of knowledge and the transformative power of education. It is through relentless learning and collaboration that we can strive to make meaningful contributions to society, unlocking potential and shaping a brighter future for generations to come...

Acknowledgement

I would like to express my deepest appreciation to those who have directly or indirectly contributed to the successful completion of this project. First, I am sincerely thankful to my supervisor, **Maria Rahim**, for her invaluable guidance, constructive criticism, and encouragement throughout this journey. Her expertise and insights have been instrumental in overcoming challenges and refining this work.

I would also like to extend my gratitude to **Dr. Tariq Mahmood** for providing crucial assistance and clarity regarding the administrative processes of enrollment and submission. His support has been appreciated in navigating this academic journey.

I am equally grateful to the academic and administrative staff at **IBA**, whose unwavering support and provision of necessary resources created an environment conducive to learning and research. The opportunities and facilities provided have been a cornerstone of this project's success.

Finally, I would like to acknowledge the collaborative efforts of my peers and colleagues, whose constructive discussions and shared knowledge added depth to my learning experience. This acknowledgment would be incomplete without recognizing the role of the wider academic community, whose foundational research and resources laid the groundwork for this project. Thank you all for your invaluable contributions.

Table of Contents

Abstract

In this project, aims to develop a web application designed to automatically generate questions from a given passage, to create aptitude test for the corporate hiring. The tool will be used to create customized aptitude, technical, or behavioral tests for potential hires. This helps in quickly screening large numbers of candidates. Furthermore, companies can use HireIQ to assess the skills of current employees or applicants, ensuring that the workforce is competent in the necessary areas. Hire IQ engine will leverage LLMs, to create diverse questions. HireIQ will streamline the hiring process and will not only save time but also enhance the accuracy and fairness of evaluations, making it an asset in corporate recruitment.

Keywords: Large Language Models (LLMs), Sentence Transformer, deepeval , LangGraph, LangChain,

1. Introduction

Recruiting and hiring the right talent is a critical process for any organization. With the growing demand for skilled professionals, organizations face the challenge of efficiently evaluating candidates while maintaining high standards. The need for an automated screening system has never been more apparent. Traditional methods, often reliant on manual effort, are time-consuming, prone to inconsistencies, and struggle to keep pace with the fast-moving job market. An automated solution addresses these challenges by providing a faster, more reliable, and scalable way to screen candidates.

This tool automates the extraction of skill sets from job descriptions and generates customized quiz questions (Danielle R. Thomas 2024). Recruiters can easily edit these questions to create tailored screening tests. By utilizing the capabilities of LLMs, this tool simplifies and accelerates the evaluation process, ensuring a more structured and objective approach to assessing candidates.

1.1 Importance of a Screening Tool

1.1.1 Time Efficiency

An automated screening process can save substantial employee time by screening a larger number of applicants and give a better chance of hiring a better resource.

1.1.1 Improve Accuracy

LLMs can understand complex job descriptions to extract relevant Technical and non-technical skill sets. For better matching.

1.1.2 Objective Evaluation

Bias can inadvertently creep into manual evaluation processes. By using AI to standardize the assessment, this tool ensures a fair and consistent evaluation of all candidates, focusing solely on their skills and knowledge.

1.1.3 Enhanced Candidate Experience

By providing a well-structured and job-specific screening process, candidates are more likely to feel engaged and valued. This can contribute to a positive perception of the organization.

1.1.4 Alternate use cases

The core of the screening test can be utilized in other flows too like evaluation employee knowledge base.

2. Application flow

Core Application is based on two prompt chains, one for generating quizzes from a given context and the second is to extract skill set from a given job description and generate context against those skill (Figure 2.1) which would be passed into the first chain (Ayan Kumar Bhowmick 2023).

Prompt chains were broken down to avoid problems like lost in context/instructions, breaking down prompts also increases testability where each prompt can be tested individually, this also allowed us to execute subsequent steps parallelly speeding up the execution.

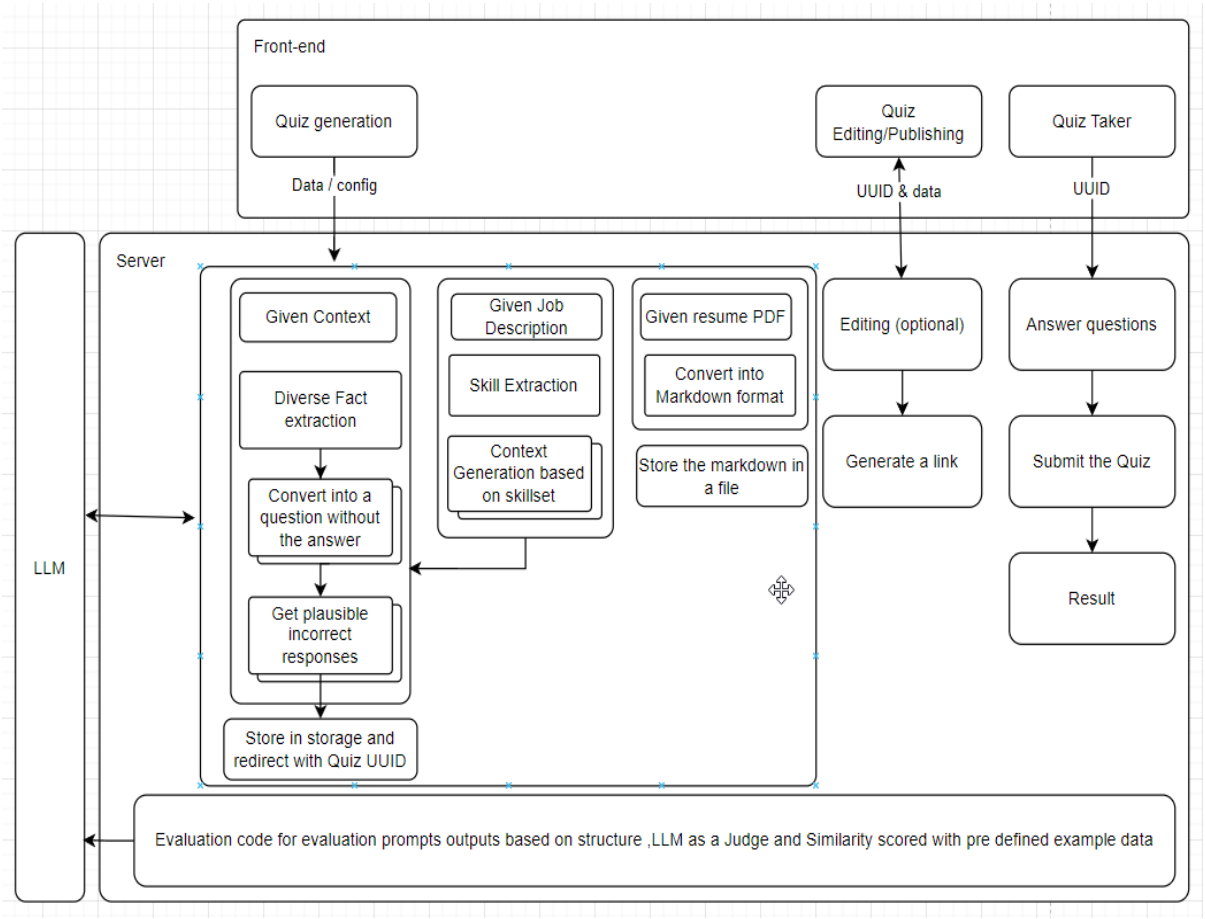

Figure 2.1

2.1 Generating quizzes from a given context

This chain is used to generate Quizzes from a given context (Figure 2.2)

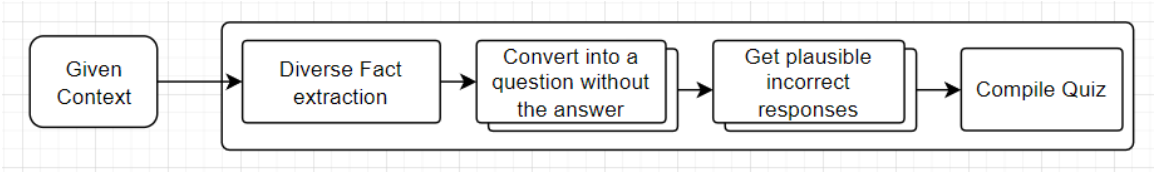

Figure 2.2

2.2 Skillset Extraction and context generation

This chain is used to extract Skillset from a job description then generates content for those skillset (Figure 2.3)

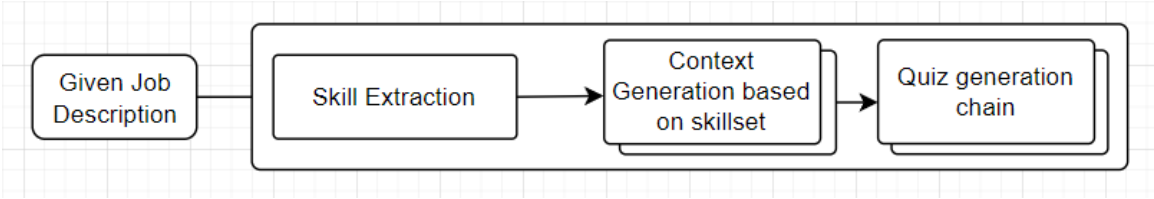

Figure 2.3

2.2 Resume Processing

Upload resumes are converted into markdown for further processing, few prompts were written to generate comparison between Job description and resume, with further work this can be integrated for personalizing the and generating unique base and bonus questions.

3. Testing & Evaluation

A good Automated testing process can save a lot of time from manual efforts, thus we implemented the below Evaluation techniques.

3.1 Structural output evaluation

The application is highly reliant on proper Json being generated by LLMs, for structural evaluation/correct, I used pydantic and JsonCorrectnessMetric from deepEval to test the generated Json.

Results: All the prompts were evaluated for structural correctness and (o1-mini , gpt-4, gpt-4o-mini, gpt-3.5-turbo) were evaluated, all of them performed good, there was the difference of pure Json and mark down Json being produced as string, to handle that a util function was written which would convert both cases into json.

3.2 LLM as a judge using chain of thought

Used Chain of thought for evaluating hallucination when doing fact extraction. (Yang Liu 2023) Have observer flakiness in these tests

- Example:
  - Compare the extracted facts with the expected output. Verify alignment in terms of content, order, and completeness
  - There should be no additional, misleading, or omitted facts
  - Compare the extracted facts with the expected output. Verify alignment in terms of content, order, and completeness
- **Results:** In our test for **fact extraction**, we tested (gpt-4o-mini, 3.5-turbo), gpt-4o-mini got a better score so it was selected.

3.3 Comparing different LLMs based on a Dataset

For a list of LLMs and a given dataset with manually generated ground truth for input and expected output cost and similarly score generated to find the Best LLM based on cost and similarly score (Figure 3.1).

- **Evaluated models:** (o1-mini, gpt-4, gpt-4o-mini, gpt-3.5-turbo)
- **Similarity score calculation:** used **SentenceTransformer('all-MiniLM-L6-v2')** to calculate similarity between input (Job description) and expected output (skillset)
- **Results:** gpt-4o-mini was select for Job description Skill Extraction based on high similarity score

Test Case: Job Description Skill Extraction Test

Prompt: Extract a list of skill set based on the below description skills should include tools, frameworks , technologies , interpersonal/transferable, managerial ----- job Description: {jobDescription} ----- These skill should not include work type, certifications and background checks Format output in json [{{ "category": name, "skills": [] }}]

Model Name	Score	Cost	Actual Output	Expected Output	Differences
gpt-3.5-turbo	0.98215...	$0.0004...	"CSS", "Material Design", "Bootstrap", "Foundation", "JIRA", "Git", "Gitlab", "BitBucket"] }, { "category": "frameworks", "skills": [	"BitBucket"] }, { "category": "Frameworks", "skills": ["Angular", "Vue", "React", "Material Design", "Bootstrap", "Foundation"]	~~communication skills",~~ 0: "Problem-solving", ~~1: "Strong collaboration and teamwork skills",~~ 1: "Troubleshooting", 2: "Strong verbal communication", 3: "Strong written communication", 4: "Collaboration",
gpt-4o-mini	0.99385...	$0.0001...	}, { "category": "Interpersonal/Transferable Skills", "skills": ["Problem-solving", "Troubleshooting", "Verbal communication", "Written communication", "Collaboration", "Teamwork"] },	}, { "category": "Interpersonal/Transferable", "skills": ["Problem-solving", "Troubleshooting", "Strong verbal communication", "Strong written communication", "Collaboration", "Teamwork"]	}, 3: { category: ~~"Interpersonal/Transferable Skills"~~ "Interpersonal/Transferable", skills: [~~2: "Verbal communication",~~ 2: "Strong verbal communication", ~~3: "Written communication",~~
gpt-4	0.99172...	$0.02346	}, { "category": "Technologies", "skills": ["JavaScript", "HTML", "CSS"] }, { "category": "Interpersonal/Transferable", "skills": [	}, { "category": "Technologies", "skills": ["JavaScript", "HTML", "CSS"] }, { "category": "Interpersonal/Transferable", "skills": [	[3: { skills: [~~2: "Verbal and written communication",~~ 2: "Strong verbal communication", 3: "Strong written communication"],
o1-mini	0.98604...	$0.0114...	[{ "category": "Technologies", "skills": ["JavaScript", "HTML", "CSS"] }, { "category": "Frameworks", "skills": ["Angular",	[{ "category": "Tools", "skills": ["JIRA", "Git", "Gitlab", "BitBucket"] }, { "category": "Frameworks", "skills": [	[0: { category: ~~"Technologies"~~ "Tools", skills: [~~0: "JavaScript",~~ 0: "JIRA", ~~1: "HTML",~~ 1: "Git", ~~2: "CSS",~~ 2: "Gitlab", 3: "BitBucket"

Figure 3.1

4. Libraries and Models Used

4.1 LangGraph

- **Library:** LangGraph (0.2.53)
- **Purpose:** library for building stateful, multi-actor applications with LLMs, used to create agent and multi-agent workflows.
- **Usage**: The library is used to generate node chains to execute different LLM calls and to stream progress of node execution, Lang graph was used to future proof the application as parts of the application logic can be converted into an agentic flow with feedback loop. Lang graph was configured to stream update responses to the front end for progress/set update.

4.2 LangChain

- **Library:** langchain-community (0.3.9) & langchain (0.3.9)
- **Purpose:** LangChain Community contains third-party integrations that implement the base interfaces defined in LangChain Core, making them ready-to-use in any LangChain application.
- **Usage**: The library is used to invoke different LLM models and get metadata like Cost for processed input and output tokens, it was also used to define prompt templates.

4.3 Deepeval

- **Library:** deepeval (2.0.6)
- **Purpose**: DeepEval is a simple-to-use, open-source LLM evaluation framework, for evaluating and testing large-language model systems. It is similar to Pytest but specialized for unit testing LLM outputs. DeepEval incorporates the latest research to evaluate LLM outputs based on metrics such as G-Eval, hallucination, answer relevancy, RAGAS, etc., which uses LLMs and various other NLP models that run locally on your machine for evaluation.
- **Usage:** The library was used to evaluate prompts; it allowed us to write custom tests like calculating similarity scores between input and expected output. Because

we required flexibility and Possibility of integration with LangFuse for observability PromptFoo was avoided. Other than the custom matrix GEval, JsonCorrectnessMetric, SummarizationMetric were used.

4.4 Sentence Transformers

- **Library:** sentence-transformers (3.3.1) (all-MiniLM-L6-v2)
- **Purpose**: This framework provides an easy method to compute dense vector representations for sentences, paragraphs, and images. The models are based on transformer networks like BERT / RoBERTa / XLM-RoBERTa etc. and achieve state-of-the-art performance in various tasks. Text is embedded in vector space such that similar text are closer and can efficiently be found using cosine similarity.
- **Usage:** The library was used to calculate similarity scores for the input and expected output, all-MiniLM-L6-v2 was used as it had a higher benchmark score for small size models.

4.4 Jinja2

- **Library:** Jinja2 (3.1.4)
- **Purpose**: Jinja is a fast, expressive, extensible templating engine. Special placeholders in the template allow writing code similar to Python syntax. Then the template is passed data to render the final document.
- **Usage:** The library was used to calculate similarity scores for the input and expected output. It was selected for ease of use, quick configuration, and low learning curve.

4.5 Pydantic

- **Library:** pydantic (2.8.2)
- **Purpose**: Data validation using Python type hints. Fast and extensible, Pydantic plays nicely with your linters/IDE/brain. Define how data should be in pure, canonical Python 3.8+; validate it with Pydantic.

- **Usage:** Pydantic's BaseModel and RootModel were used to define Json object structure which was used in validating output from LLM

4.5 Pymupdf4llm

- **Library:** pydantic (2.8.2)
- **Purpose**: This package converts the pages of a PDF to text in Markdown format using PyMuPDF.

  Standard text and tables are detected, brought in the right reading sequence and then together converted to GitHub-compatible Markdown text.

  Header lines are identified via the font size and appropriately prefixed with one or more '#' tags.

  Bold, italic, mono-spaced text and code blocks are detected and formatted accordingly. The similarity applies to ordered and unordered lists.

- **Usage:** Tested different PDF readers with LLMs and Markdown converters performed the best for questions and answering, out of MinerU , marker and pymupdf4llm was selected for speed and ease of integration.

4.6 Others

Server was built using **uvicorn** (0.31.1) and **fastapi** (0.115.5) frontend was built using **react** (18.3.1) and **vite** (5.4.8)

5. Prompt Enginering

5.1 Quiz generation from Fact Extraction

5.1.1 Fact Extraction Prompt

This prompt is used to extract facts from a given context

Prompt:

"""

you are a summarizer, and you need extract main facts from the text below
-----
Text: {text}

Format output as an array
[
"fact1",
"fact2",
"fact3"
]

"""

Variable Used:

- {text}: Any text can be entered here to extract facts out of it.

Purpose of the Prompt:

- Extract facts out of given peace of text
- Formats the output in a Json array format

5.1.2 Question generation

This prompt takes in a fact and splits it into a question and an answer

Prompt:

"""

Generate a question and an answer out of the below test, the answer should be concise as possible
-----
facts: {text}

Format output in object
"Question":,
"Answer":

"""

Variable Used:

- {text} takes in any fact

Example: "Jon is 5 feet tall" > {Question: “how tall is jon”, Answer: “5 feet”}

Purpose of the Prompt:

- Split a fact into Question and an answer
- Formats the output in a Json object format

5.1.3 Quiz generation prompt

This prompt takes in Question and Answer and generates 3 false options for them

Prompt:

"""

Use the following Question and Answer to generate three false options for a quiz as output these options should not be the correct answer to the question

---

Question: {Question}

Answer: {Answer}

Format output in object that can be parsed in json

[

1:[value1],

2:[value2],

3:[value3],

]

"""

Variable Used:

- {Question}
- {Answer}

Purpose of the Prompt:

- Generates 3 false answers from the given question and answer.
- Formats the output in Json array format.

5.2 Skill set extraction and context generation

The purpose of the chain is to extract skill sets from job description and generate content for those skills which will be used to generate the assessment test.

5.2.1 Skill Extraction Prompt

Takes in job description and splits generates a list of technical and interpersonal skills required.

Prompt:

"""

Extract a list of skill set based on the below description

skills should include tools, frameworks, technologies, interpersonal/transferable, managerial

---

job Description:

{jobDescription}

These skills should not include work type, certifications, and background checks
Format output in json
[
{{
"category": name,
"skills": []
}}
]
"""

Variable Used:

• {jobDescription}: takes in a job description

Purpose of the Prompt:

- Extracts skill from the job description
- Segments the skills into related categories
- Outputs structured Json which can be parsed by the system

5.2.2 Generate Context From skillset group

Takes in a list of skillsets and generates context for them, it has instructions related to structure, depth, constraints, and output format. This output will be used in the fact generation chain. (Nicy Scaria 2024)

Prompt:

"""

Given a list of skills, generate a detailed and unique context for each skill. The context should provide comprehensive information about the skill to be used for quiz generation. Follow these guidelines:

1. **Structure**: for the given skillset, create paragraph of concepts that an expert in that skill should know. The context should include:
    a. few expert level concepts of a skill or tool that is concisely defined.
    b. Key principles related to that skill or tool that an expert should know.
2. **Depth**: Ensure the context is rich in detail and avoids generic statements. Include any nuances or specifics that would make the context unique and informative.
3. **Output Format**: The output should be structured as follows:

    [Skill Name]: [Detailed Context] [Skill Name]: [Detailed Context] ...

    Generate context only for the provided skills.

4. **Constraints**:
    a. Avoid using contexts from external references.
    b. Avoid code snippets.
    c. Avoid asking simple questions like what a specific tool does.
    d. Ensure the descriptions are written in an educational and engaging tone suitable for fact generation.

Input: skillset: {skillset}

Expected Output:

- [Skill Name]: [Detailed Context]
- [Skill Name]: [Detailed Context]

"""

Variable Used:

• {skillset}: This would be a list of skill set belonging to a particulate category

Example: below is a skill set extracted from a job description belonging to a managerial category.

"skills": ["Scrum", "Agile development"]

Purpose of the Prompt:

- To generate context for a skill set belonging to the same category.
- Formats the into a Json object with key defining the skill name and having context as value.

6.Future Work

6.1 Disaster recovery

As we know the demand for LLMs is ever increasing and we have seen major outages or LLMs as a service such as Open AI it becomes important that these outages do not affect the business.

6.1.1 Back up LLMs

HireIQ compares different LLM with golden examples for cost optimization, same report can be used to configure backup LLMs so if the primary LLM is down, the application will automatically switch to the backup LLM. This will production application more stable

6.2 Security

LLMs are known to be prone to prompt injection Attacks Security instructions can be added to avoid prompt leaking and other attacks

6.3 Better Context generation Agentic Loop flow for

Context generation can be improved with a feedback loop which would generate the Context for a skill run another LLM request for change of thought feedback and generate a better context

6.4 LLM Observability

LLM application running in production needs to have Observability for Continues improvement. In HireIQ Observability can be configured by integrating LangSmith or LangFuse.

6.5 Prompt and dataset management

HireIQ flow is compatible with a prompt and dataset management tool like LangFuse using that for prompt Management will result in faster prompt iterations and testing without requiring developers

6.6 Fine tuning opportunities

Skill extraction (Technical/Interpersonal) for job description and resume can be done through a fine-tuned Smaller LLM this can give better similarity scores. Fine tuning might be possible by generating a synthetic dataset of resume and job description. Using a dataset like https://www.kaggle.com/datasets/arbazkhan971/allskillandnonskill/

6.7 cheating and proxy test

Cheating by using LLMs or internet search on the same device can be detected in the browser by checking for tab switched, camera can be turned on and some frames can be processed to see the person to identify proxy test. Furthermore, some questions can be audio based to test audio video sync.

6.8 better methods for measuring hallucinations

We are using LLM as a judge and chain of thought to test to check for hallucinations manual using another NLP based approach can be more stable. As LLM as a judge can be flakey sometimes.

7. Repository Link

https://github.com/qadeerk/quizGen

8. References

Ayan Kumar Bhowmick, Ashish Jagmohan, Aditya Vempaty, Prasenjit Dey, Leigh Hall, Jeremy Hartman, Ravi Kokku, Hema Maheshwari. 2023. "Automating question generation from educational text." *arxiv* 14. https://arxiv.org/abs/2309.15004.

Danielle R. Thomas, Conrad Borchers, Sanjit Kakarla, Jionghao Lin, Shambhavi Bhushan, Boyuan Guo, Erin Gatz, Kenneth R. Koedinger. 2024. "Does Multiple Choice Have a Future in the Age of Generative AI? A Posttest-only RCT." *arxiv* 17. https://arxiv.org/abs/2412.10267.

Nicy Scaria, Suma Dharani Chenna, Deepak Subramani. 2024. "Automated Educational Question Generation at Different Bloom's Skill Levels using Large Language Models: Strategies and Evaluation." *arxiv* 14. https://arxiv.org/abs/2408.04394.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, Chenguang Zhu. 2023. "G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment." *arxiv* 11. https://arxiv.org/abs/2303.16634.