Physics Assessment Generation Through Pattern Matching and Large Language Models

Marchotridyo and Fariska Zakhralativa Ruskanda

September 6, 2024

Physics Assessment Generation Through Pattern Matching and Large Language Models

[1]Marchotridyo, [2]Fariska Zakhralativa Ruskanda
School of Electrical Engineering and Informatics
Bandung Institute of Technology
Bandung, Indonesia
[1]acoxstpd@gmail.com, [2]fariska.zr@staff.stei.itb.ac.id

Abstract*— Question generation has been an active area of research in Natural Language Processing (NLP) for some time, particularly for educational applications. This need has become even more pressing in the evolving educational landscape where online assessments are increasingly common. Our research focuses on generating physics assessments due to the unique challenge presented by the combination of generating both textual and numerical content. This paper presents an innovative approach to automated physics assessment generation by integrating pattern matching techniques with large language models (LLMs) which are Pegasus, T5, ChatGPT-3.5 Turbo, and Mistral 7B. The proposed method involves two main processes: generating variable values through pattern matching using regular expressions and paraphrasing the generated assessment questions using LLMs to ensure syntactic and semantic diversity. The generated paraphrases then get evaluated using automatic metrics (BLEU, METEOR, ROUGE, and ParaScore) and human assessments. The results indicate that LLMs with larger parameters used in this research, which are ChatGPT-3.5 Turbo and Mistral-7B, excel in generating high-quality paraphrases that are both syntactically correct and contextually meaningful. Both models achieved perfect human evaluation scores (3.000) compared to Pegasus (1.705) and T5 (1.529). Additionally, they received higher ParaScore scores, with ChatGPT-3.5 Turbo scoring 0.803 and Mistral-7B scoring 0.788, outperforming Pegasus (0.768) and T5 (0.760). Additionally, the results highlight the limitations of traditional n-gram based evaluation metrics and the potential of ParaScore as a more representative measure. This research contributes to the development of more reliable and varied question banks, aiding educators in creating personalized and cheat-resistant assessments.

Keywords—physics assessment generation, pattern matching, large language models, paraphrasing, regular expressions

I. INTRODUCTION

The field of education has undergone significant transformations with the birth of digital technologies and the internet. Traditional methods of teaching and assessment are increasingly being supplemented or replaced by online and automated systems. Among these innovations, question generation has emerged as a pivotal area of research within Natural Language Processing (NLP). Automated question generation holds the potential to revolutionize educational practices by enabling the creation of diverse, personalized, and scalable assessments [1][7]. This capability is particularly crucial in the context of physics education, where problem-solving and conceptual understanding are key.

Physics, with its unique blend of theoretical concepts and practical problem-solving, presents a distinct set of challenges for question generation. Unlike purely theoretical subjects, physics problems often require numerical computations and contextual scenarios that need to be both accurate and varied. This complexity requires sophisticated techniques that can handle both the linguistic and mathematical aspects of question generation.

In recent years, advances in artificial intelligence, particularly in the development of large language models (LLMs), have opened new avenues for automated question generation. LLMs, such as Pegasus, T5, ChatGPT-3.5 Turbo, and Mistral 7B, have demonstrated remarkable capabilities in understanding and generating human-like text [2][3][4][5][6]. These models, trained on vast amounts of data, can generate coherent and contextually appropriate text, making them ideal candidates for the task of question generation.

However, generating high-quality physics questions involves more than just creating grammatically correct sentences. It requires the integration of domain-specific knowledge, the ability to generate variable values for numerical problems, and the capability to paraphrase questions to introduce diversity while maintaining their core semantic meaning. This paper presents an innovative approach that combines pattern matching techniques with LLMs to address these challenges.

The proposed method involves two main processes: first, generating variable values (things that can be varied in questions, e.g. the speed of a car, the height of a ball being thrown from) through pattern matching using regular expressions; and second, paraphrasing the generated questions using LLMs to ensure syntactic and semantic diversity. Pattern matching techniques allow for the identification and manipulation of variable components within a question, enabling the generation of different numerical values while maintaining the logical structure of the problem. This step ensures that the generated questions are not mere replicas but variations that can challenge students' understanding and application of physics concepts.

Once the variable values are generated, the next step is to paraphrase the questions using LLMs. Paraphrasing is crucial for creating a diverse set of questions that prevent rote memorization and cheating. By rephrasing the same question in multiple ways, educators can assess students' comprehension more effectively [1]. LLMs, with their advanced text generation capabilities, are well-suited for this task, as they can produce grammatically and semantically varied versions of the same question [2]. The paraphrases are then evaluated by both automatic metrics (BLEU, METEOR, ROUGE, and ParaScore) and human assessments.

This research contributes to the development of more reliable and varied question banks, aiding educators in creating personalized and cheat-resistant assessments [1][7]. By leveraging the strengths of both pattern matching and LLMs, our approach offers a scalable and efficient solution for automated physics question generation. While previous research primarily focused on generating either textual or numerical content for questions, our work introduces a method that generates and varies both types, enhancing the diversity and adaptability of assessment items. This paper is structured as follows: Section II reviews the related work in question generation. Section III details the methodology, including data representation and the integration of pattern matching with LLMs. Section IV presents the results and discusses the findings, and Section V concludes the paper with a summary and suggestions for future research.

II. Related Works

The field of automated question generation has seen significant advancements over the past few years, driven by developments in natural language processing (NLP) and artificial intelligence (AI). Various methodologies have been explored, ranging from rule-based systems to using neural networks [7][8][9]. This section reviews some contributions to the field, highlighting different approaches and their applications in educational contexts. By examining these related works, we can better understand the landscape of current research and how our approach compares and contributes to the existing body of knowledge.

A. Question Generation

Question Generation (QG) is a process in natural language processing (NLP) aimed at automatically creating question-answer pairs from various data sources such as text, knowledge bases, or tables. This technique is crucial in numerous applications including educational tools, dialogue systems, and intelligent tutoring systems. Utilizing neural networks, QG transforms unstructured content into structured question-answer pairs, enhancing the interactivity and effectiveness of learning platforms. The generated questions can be used in quizzes, educational games, and assessments, providing personalized learning experiences and aiding in knowledge retention [1].

B. Paraphrase

Paraphrasing involves rephrasing text to convey the same meaning using different words or structures. It is widely used in various applications such as simplifying content for easier understanding, avoiding plagiarism, and enhancing language models' training data through data augmentation. Over time, paraphrase generation has evolved from rule-based [17][18] and thesaurus-based approaches [19] to advanced neural network models [20]. These modern techniques, particularly those employing sequence-to-sequence models and transformers, enable the creation of more fluent, diverse, and contextually accurate paraphrases. This shift has significantly improved the quality and applicability of paraphrase generation in natural language processing tasks [2].

C. Paraphrase Evaluation Metrics

Commonly used metrics for evaluating paraphrase generation include both automatic and human evaluation methods. Automatic evaluation metrics frequently used are BLEU, METEOR, ROUGE, and TER [2]. BLEU, originally developed for machine translation, measures n-gram overlaps between generated paraphrases and reference texts [14]. METEOR addresses BLEU's limitations by considering synonymy and stemming, correlating better with human judgment [13]. ROUGE, especially its versions ROUGE-N and ROUGE-L, focuses on recall and the longest common subsequence, respectively [15]. TER calculates the number of edits needed to transform a generated paraphrase into a reference sentence, with lower scores indicating better quality [16]. Despite their prevalence, these metrics primarily measure surface-level similarity, prompting the use of human evaluation to assess semantic fidelity, fluency, and overall quality of paraphrases for a more comprehensive evaluation.

To address the limitations of existing evaluation metrics, we introduce the usage of ParaScore, a new metric specifically designed for paraphrase generation [11]. ParaScore integrates the strengths of both reference-based and reference-free metrics while explicitly modeling lexical divergence, which is a critical aspect of effective paraphrasing [11]. Unlike traditional metrics, ParaScore evaluates the quality of paraphrases by considering not only their semantic similarity to the input but also their lexical and syntactic variations [11]. This comprehensive approach ensures a more accurate alignment with human judgment and significantly improves the evaluation of paraphrase generation tasks.

D. Previous Research

The paper by Scharpf et al. (2022) presents an innovative approach to generating exam questions by utilizing public knowledge stored in Wikidata. The system, named PhysWikiQuiz, is designed to create physics-related questions based on formulas stored in Wikidata. This process involves leveraging a Computer Algebra System (CAS) to manipulate these formulas, thereby generating both the questions and the corresponding answers. Despite facing challenges in translating the raw formulas from Wikidata into coherent questions, the system shows significant potential for producing extensive educational content [8]. Unlike PhysWikiQuiz, our implementation does not incorporate a Computer Algebra System, differentiating our approach and methodology.

The paper by Tuloli et al. (2021) presents the development of an anti-cheating software tool designed for introductory linear algebra courses. Their research highlights that 61% of students admitted to having cheated on exams. The software includes a module that generates matrix multiplication problems by randomizing the numbers within given matrices, and it also produces the solutions for instructors to use during grading [7]. A key difference between their approach and ours is that Tuloli et al.'s work focuses solely on generating variable values (numbers) for mathematical problems, without addressing the generation of the textual content of the questions.

The paper by Thotad et al. (2022) presents a method for generating questions by leveraging natural language processing (NLP) techniques such as tokenization, part-of-speech (POS) tagging, and lemmatization to process a text corpus and generate questions from it. The system creates problems from sources like Wikipedia articles, extracting facts and converting them into questions. Additionally, it can generate plausible incorrect answers, facilitating the creation of multiple-choice questions with only one correct answer [9]. A key difference between their approach and ours is that Thotad et al. focus solely on the textual generation of problems, whereas our approach also involves the generation of numerical values, which is particularly relevant for physics questions.

III. METHODOLOGY

This section outlines the methodology employed in our research to generate automated physics questions. Our approach combines pattern matching techniques with large language models (LLMs) to create diverse and semantically accurate questions. The methodology is structured into four main components: data representation, usage of pattern matching, paraphrasing using LLMs, and evaluating the paraphrase results. Each component is integral to ensuring the generation of high-quality questions that are both syntactically correct and contextually relevant. The following subsections provide a detailed description of each component and the techniques used to implement them.

A. Designing Data Structure for Question Generation

To facilitate the generation of variables within questions, it is essential to store the questions in a way that makes their variables easily identifiable by the system. We opted to use a hash table to represent our question data, breaking it down into the following components:

1. Text: the text of question with variables turned to templates
2. Rules: rules to follow when a problem is generated later
3. Answer: a mathematical formula that can be evaluated by programming languages (in our paper, we use Python) that is the answer to the problem.
4. Solution: a text written in LaTeX format to show detailed steps on how to solve problem

A concrete example of this implementation to store a question can be seen at Table I (explanation is cut off, only shown to give a brief example).

TABLE I A CONCRETE EXAMPLE OF A QUESTION

<table>
<tr><td>Text</td><td>A tennis ball is falling from rest from a height of {{height}} m. If the gravity in that place is equal to 10 m/s², determine the speed of the tennis ball when it just reaches the ground, in m/s!</td></tr>
<tr><td>Rules</td><td><table><tr><td>Variable</td><td>Type</td><td>Rules</td></tr><tr><td>height</td><td>int</td><td>min: 5
max: 10</td></tr></table></td></tr>
<tr><td>Answer</td><td>(2*10*{{height}})**(0.5)</td></tr>
<tr><td>Explanation</td><td>This follows the equation of
$$ s = s_0 + v_0\cdot t + \frac{1}{2} …
So, the answer is {{answer}} m/s.</td></tr>
</table>

In Table I, an example of a question represented in its components is provided. The text component indicates the question text, which contains the variable height. Variables in a question are always denoted using {{ }}, such as {{height}} in the example.

The rules component specifies the rules that must be followed when filling the value of each variable. In this example, the height variable is given a constraint as an integer with a minimum of 5 and a maximum of 10.

The answer component specifies the equation used to solve the question. This part is structured so that it can be directly evaluated by a programming language.

The explanation component describes the steps taken to derive the answer component. This part is written in LaTeX format so that mathematical equations can be correctly displayed in the interface.

B. Applying Pattern Matching for Question Generation

Pattern matching is used to convert all the variables in the stored data across all relevant components (text, answer, and explanation). We use regular expressions to parse the strings and fill in the templates. For example, Table II shows the transformation of Table I after the height variable is generated as 6.

TABLE II A QUESTION WITH ITS VARIABLE FILLED

Text	A tennis ball is falling from rest from a height of 6 m. If the gravity in that place is equal to 10 m/s^2, determine the speed of the tennis ball when it just reaches the ground, in m/s!
Answer	(2*10*6)**(0.5)
Explanation	This follows the equation of $$ s = s_0 + v_0\cdot t + \frac{1}{2} … So, the answer is {{answer}} m/s.

The regular expression works by detecting all the variables stored in the data, which is surrounded by double curly brackets ("{{" and "}}"). The algorithm used is as follows:

1. The text component is matched with the regular expression pattern r"{{(.+?)}}". This expression searches for parts of the text that begin with "{{" and end with "}}". For example, the text "Bob drives a car at a speed of {{speed}} m/s for {{time}} seconds." matched with this pattern will identify the variables "speed" and "time".
2. The variables identified in step 1 are transformed according to the applicable rules for those variables.
3. The answer component is calculated by evaluating the equation after replacing the variable values with the ones determined in step 2.
4. The variables in the explanation component are replaced with the values determined in step 2.

Specifically, for the answer variable, the value is replaced with the one calculated in step 3.

C. *Utilizing LLMs for Paraphrasing*

After questions are generated via pattern matching, it is used as an input for the next step: paraphrasing. To paraphrase, we use 2 different kinds of LLMs usage: one finetuned with the Quora dataset with less parameters and one uses an instructional model with more parameters. The finetuned models are Pegasus and T5 (both using the base model) and the instructional models used are Mistral-7B and ChatGPT-3.5 Turbo.

To prompt the models, we use a modification of a technique called template pattern [10]. As seen in Table III, we explicitly state what kind of response we expect to be returned from the LLM used. This is done with the intent and motivation so that the LLM is consistent with what it returns so the system can process it without any problems.

TABLE III PROMPT USED FOR PARAPHRASING

Role	**Instruction**
System	You are a helpful assistant. Follow the instructions given by the user. Return only a JSON object as asked.

User	You need to paraphrase a physics question.
	Return it in a .json format (do not format in ``json``` format. Return just the JSON object), according to this template:
	{
	"original_question": <original_question>,
	"paraphrased_question": <paraphrased_question>
	}
	Make sure to return only the JSON object and make sure that the JSON object is a valid JSON object.
	The question is: <insert_question_here>

D. *Assessing Paraphrase Quality*

To assess the quality of paraphrases, we employ various evaluation methods, ranging from automatic metrics to manual human evaluation. The automatic methods include commonly used n-gram-based metrics, which are BLEU, METEOR, and ROUGE [2]. Additionally, we utilize ParaScore, an advanced automatic evaluation method that leverages language models to better understand context and variations in the paraphrases [11].

For the manual human evaluation, we use a simple 1-3 scale:

1. A score of 1 indicates that the paraphrased question is unsolvable due to the removal of critical details (e.g., key variables are omitted in the paraphrased version).
2. A score of 2 signifies that the paraphrased question is solvable but difficult to understand (e.g., it contains grammatical errors).
3. A score of 3 means that the paraphrased question is both solvable and easy to understand.

We chose a smaller scale, unlike previous research that uses a 1-5 scale [12], to reduce the subjectivity of the grader. We believe that it is challenging to differentiate between paraphrases that are considered extremely good (e.g., a score of 5) and those that are just good (e.g., a score of 4).

Examples of paraphrase results with scores of 1, 2, and 3 can be found in Table IV. The second example receives a score of 2 because of confusing sentences, such as the final sentence, "measured as its distance from" (e.g., from where?). The third example is given a score of 1 because the paraphrased question omits a critical detail: the specific speed of 19 m/s mentioned in the reference question is replaced with the vague phrase "a high rate of speed."

TABLE IV EXAMPLES OF HUMAN EVALUATION SCORING

Reference question	**Paraphrased question**	**Score**	**Normalized score**
A tennis ball is falling from rest from a height of 7 m. If the gravity in that place is equal to 10 m/s^2, determine the speed of the tennis ball when it just reaches the ground, in m/s!	Calculate the velocity of a tennis ball at the moment it hits the ground after falling freely from a height of 7 m, given the acceleration due to gravity at that location is 10 m/s^2.	3	1
An object is free falling from a height of 35 meters. What is the amount of time it needs to travel when it reaches the ground, in seconds?	An object is falling from a height of 35 meters while free falling from a height of. How much time does it take for the object to reach the ground, measured as its distance from.	2	0.667
Bob is driving his car at a constant speed of 19 m/s. He needs to arrive at ITB, which is 5 km away from where he is at currently. How long will the drive take, in seconds?	Bob is driving his car at a high rate of speed. He needs to arrive at ITB, which is 5 km away from where he is currently. How long will the drive take?	1	0.333

IV. RESULTS AND DISCUSSION

We collected a sample of 50 kinematics questions ranging from high-school to undergraduate level and paraphrased each of them using the models discussed earlier: ChatGPT-3.5 Turbo, Mistral 7B, Pegasus, and T5. Each paraphrase was evaluated using various metrics, which are BLEU, METEOR, ROUGE-1, ROUGE-2, ROUGE-L, ParaScore, and human evaluation. The results were averaged for each model and are presented in Table V.

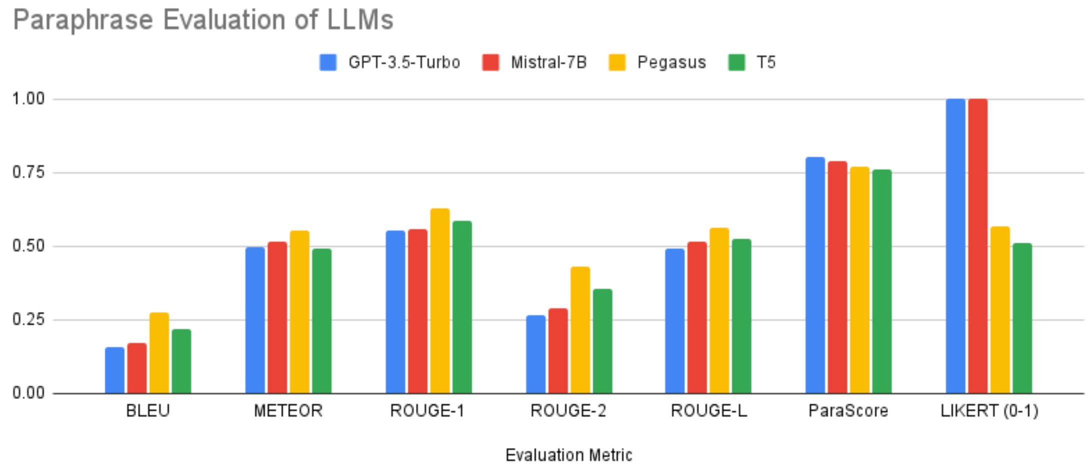

Figure 1. Average score of each metric per model

TABLE V AVERAGE SCORE OF EACH METRIC PER MODEL

Metric	Model	Average Score
BLEU	ChatGPT-3.5 Turbo	0.156
	Mistral 7B	0.171
	Pegasus	0.274
	T5	0.221
METEOR	ChatGPT-3.5 Turbo	0.497
	Mistral 7B	0.514
	Pegasus	0.554
	T5	0.490
ROUGE-1	ChatGPT-3.5 Turbo	0.554
	Mistral 7B	0.559
	Pegasus	0.630
	T5	0.584
ROUGE-2	ChatGPT-3.5 Turbo	0.267
	Mistral 7B	0.290
	Pegasus	0.432
	T5	0.354
ROUGE-L	ChatGPT-3.5 Turbo	0.494
	Mistral 7B	0.515
	Pegasus	0.562
	T5	0.527
ParaScore	ChatGPT-3.5 Turbo	0.803
	Mistral 7B	0.788
	Pegasus	0.768
	T5	0.760
Human evaluation	ChatGPT-3.5 Turbo	3.000
	Mistral 7B	3.000
	Pegasus	1.705
	T5	1.529

An easier view of the data is provided in Figure I. From this figure, we can observe that in terms of automatic evaluation metrics that use n-gram methods (BLEU, METEOR, ROUGE), the finetuned LLMs with fewer parameters (Pegasus and T5) outperform the larger, instruction-based LLMs (ChatGPT-3.5 Turbo and Mistral 7B). However, the conclusions are reversed when considering the results from ParaScore and human evaluations.

We found that automatic evaluations using n-gram methods do not correlate well with, and often contradict, human evaluation results. This discrepancy arises because n-gram based evaluations do not account for synonyms and lack a true understanding of the semantic meaning between the reference and the paraphrased questions. Rather than rewarding lexical variation, metrics like BLEU penalize paraphrases that significantly differ in wording from the reference question [2][14]. As shown in Table VI, when a paraphrased question uses many different words from the reference (e.g., "slowing down" instead of "decelerating", "constant" instead of "consistent"), the BLEU score is very low.

TABLE VI AN EXAMPLE OF BLEU SCORE NOT CORRESPONDING TO HUMAN SCORE

Reference question	**Paraphrased question**	**BLEU score**	**Human score**
A particle is decelerating with a constant deceleration. Its' speed has reduced from 25 m/s into 10 m/s after moving for 90 m. What distance does the particle need to travel again for it to stop (in meters)?	If a particle is slowing down with a consistent rate and it went from 25 m/s to 10 m/s while traveling 90 meters, how far does it still need to travel to come to a complete stop (in meters)?	3.8E-78	3

Among the n-gram based evaluations, METEOR correlates most closely with human evaluations. This is because METEOR can recognize synonyms through WordNet and perform stemming [13]. Additionally, METEOR employs a chunking mechanism to grade variations more effectively [13].

As an automated evaluation method, ParaScore outperforms all n-gram based metrics by aligning more closely with human evaluations. ParaScore's ability to convert sentences into embeddings allows it to understand the connections between the reference and paraphrased questions more deeply [11]. However, the differences in scores are not as pronounced as those from human evaluations, suggesting that ParaScore alone is not sufficient to fully capture the performance differences between models.

Therefore, human evaluations remain essential for accurately assessing paraphrase quality.

We observed that the finetuned LLMs with fewer parameters, which were trained on the Quora dataset, struggle to identify and retain critical parts of the questions, often omitting them in the paraphrased versions. As shown in Table VII, both the Pegasus and T5 models removed essential numerical details (e.g., the height and speed of an object) that were present in the reference questions.

TABLE VII EXAMPLES OF INEFFECTIVE PARAPHRASES

Model	**Reference question**	**Paraphrased question**
Pegasus	An object is free falling from a height of 35 meters. What is the amount of time it needs to travel when it reaches the ground, in seconds?	An object is falling from a height. What is the amount of time it takes for it to reach the ground?
T5	An object, who is originally at the origin, is moving with a constant velocity of v = (4i - 6j) m/s. After moving for 5 seconds, how far would have the object travelled, in seconds?	An object, which is at the origin, is moving with a constant velocity of v. If an object moved for 5 seconds, and then moved for another 5 seconds, how far would.

V. CONCLUSION

This research presents an approach to automated physics question generation by integrating pattern matching techniques with large language models (LLMs). By using regular expressions, we efficiently identify and generate variable values within question templates, ensuring the logical structure and accuracy of the problems. The subsequent paraphrasing of questions using LLMs enhances the diversity and semantic richness of the questions, making them more challenging and engaging for students.

Our evaluation, incorporating both automatic metrics such as BLEU, METEOR, ROUGE, and ParaScore, and manual human assessments, demonstrates the effectiveness of our approach. The results indicate that LLMs with larger parameters used in this research, which are ChatGPT-3.5 Turbo and Mistral-7B, excel in generating high-quality paraphrases that are both syntactically correct and contextually meaningful. Both models achieved perfect human evaluation scores (3.000) compared to Pegasus (1.705) and T5 (1.529). Additionally, they received higher ParaScore scores, with ChatGPT-3.5 Turbo scoring 0.803 and Mistral-7B scoring 0.788, outperforming Pegasus (0.768) and T5 (0.760). Our evaluation also shows how traditional n-gram metrics (BLEU, METEOR, and ROUGE) does not correlate well with human evaluation.

In conclusion, this research contributes to the field of educational technology by offering a scalable and efficient solution for automated question generation. By combining pattern matching with advanced AI models, we provide a methodology that can be adapted to various subjects beyond physics, paving the way for more personalized and cheat-resistant assessments. Future work could explore the integration of additional AI techniques and the expansion of this approach to other areas of education, further enhancing the impact and applicability of automated question generation.

REFERENCES

[1] P. Scharpf, M. Schubotz, A. Spitz, A. Greiner-Petter, and B. Gipp, “Collaborative and AI-aided Exam Question Generation using Wikidata in Education,” 2022. [Online]. Available: https://purl.org/

[2] J. Zhou and S. Bhat, "Paraphrase Generation: A Survey of the State of the Art," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic, 2021, pp. 5075-5086.

[3] P. P. Ray, “ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope,” Jan. 01, 2023, *KeAi Communications Co*. doi: 10.1016/j.iotcps.2023.04.003.

[4] J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu, “PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization,” Dec. 2019, [Online]. Available: http://arxiv.org/abs/1912.08777

[5] A. Q. Jiang et al., “Mistral 7B,” Oct. 2023, [Online]. Available: http://arxiv.org/abs/2310.06825

[6] C. Raffel et al., “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer,” Oct. 2019, [Online]. Available: http://arxiv.org/abs/1910.10683

[7] M. S. Tuloli, M. Latief, and M. Rohandi, “Anti-cheating software tool: Prototype of problem generator software for linear algebra introductory test,” *IOP Conf Ser Mater Sci Eng*, vol. 1098, no. 3, p. 032025, Mar. 2021, doi: 10.1088/1757-899x/1098/3/032025.

[8] P. Scharpf, M. Schubotz, A. Spitz, A. Greiner-Petter, and B. Gipp, “Collaborative and AI-aided Exam Question Generation using Wikidata in Education,” 2022. [Online]. Available: https://purl.org/

[9] P. Thotad, S. Kallur, and S. Amminabhavi, “Automatic Question Generator Using Natural Language Processing,” *Journal of Pharmaceutical Negative Results*, vol. 13, 2022, doi: 10.47750/pnr.2022.13.S10.330.

[10] J. White et al., “A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT,” Feb. 2023, [Online]. Available: http://arxiv.org/abs/2302.11382

[11] L. Shen, L. Liu, H. Jiang, and S. Shi, “On the Evaluation Metrics for Paraphrase Generation,” Feb. 2022, [Online]. Available: http://arxiv.org/abs/2202.08479

[12] A. Gupta, A. Agarwal, P. Singh, and P. Rai, “A Deep Generative Framework for Paraphrase Generation,” Sep. 2017, [Online]. Available: http://arxiv.org/abs/1709.05074

[13] M. Denkowski and A. Lavie, “Meteor Universal: Language Specific Translation Evaluation for Any Target Language,” *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pp. 376–380, 2014.

[14] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a Method for Automatic Evaluation of Machine Translation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, Pennsylvania*, USA, 2002, pp. 311-318.

[15] C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," in *Text Summarization Branches Out*, Barcelona, Spain, 2004, pp. 74-81.

[16] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul, "A Study of Translation Edit Rate with Targeted Human Annotation," in *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, Cambridge, Massachusetts, USA, 2006, pp. 223-231.

[17] K. McKeown, "Paraphrasing questions using given and new information," *American Journal of Computational Linguistics*, vol. 9, no. 1, pp. 1–10, 1983.

[18] D. Lin and P. Pantel, "Discovery of inference rules for question-answering," *Natural Language Engineering*, vol. 7, no. 4, pp. 343–360, 2001.

[19] I. A. Bolshakov and A. Gelbukh, "Synonymous paraphrasing using wordnet and internet," *International Conference on Application of Natural Language to Information Systems*, 2004, pp. 312–323.

[20] A. Gupta, A. Agarwal, P. Singh, and P. Rai, "A deep generative framework for paraphrase generation," *arXiv preprint arXiv:1709.05074*, 2017.