Collaborative and AI-aided Exam Question Generation using Wikidata in Education

Philipp Scharpf[1], Moritz Schubotz[2], Andreas Spitz[1], André Greiner-Petter[3] and Bela Gipp[4]

[1] *University of Konstanz, Germany*
[2] *FIZ Karlsruhe, Germany*
[3] *NII Tokyo, Japan*
[4] *University of Göttingen, Germany*

Abstract

Since the COVID-19 outbreak, the use of digital learning or education platforms has substantially increased. Teachers now digitally distribute homework and provide exercise questions. In both cases, teachers need to develop novel and individual questions continuously. This process can be very time-consuming and should be facilitated and accelerated both through exchange with other teachers and by using Artificial Intelligence (AI) capabilities. To address this need, we propose a multilingual Wikimedia framework that allows for collaborative worldwide teacher knowledge engineering and subsequent AI-aided question generation, test, and correction. As a proof of concept, we present »PhysWikiQuiz«, a physics question generation and test engine. Our system (hosted by Wikimedia at https://physwikiquiz.wmflabs.org) retrieves physics knowledge from the open community-curated database Wikidata. It can generate questions in different variations and verify answer values and units using a Computer Algebra System (CAS). We evaluate the performance on a public benchmark dataset at each stage of the system workflow. For an average formula with three variables, the system can generate and correct up to 300 questions for individual students, based on a single formula concept name as input by the teacher.

1. Introduction and Motivation

With the rise of digital learning or education platforms, the frequency of teachers posing tasks and questions digitally has increased substantially. However, due to temporal constraints, it would be infeasible for teachers to constantly create novel and individual questions tailored to each different student. With the aid of Artificial Intelligence (AI), they can submit AI-generated learning tests more frequently, which can lead to student performance improvement. Moreover, many teachers develop exam questions without exchanging ideas or material with their peers. In many cases, this may unnecessarily cost them a lot of time and effort. Instead, they should be able to focus on explaining the concepts to their students. To address these shortcomings, we propose using Wikidata

Wikidata'22: Wikidata workshop at ISWC 2022
philipp.scharpf@uni-konstanz.de (P. Scharpf); moritz.schubotz@fiz-karlsruhe.de (M. Schubotz); andreas.spitz@uni-konstanz.de (A. Spitz); greinerpetter@nii.ac.jp (A. Greiner-Petter); gipp@cs.uni-goettingen.de (B. Gipp)

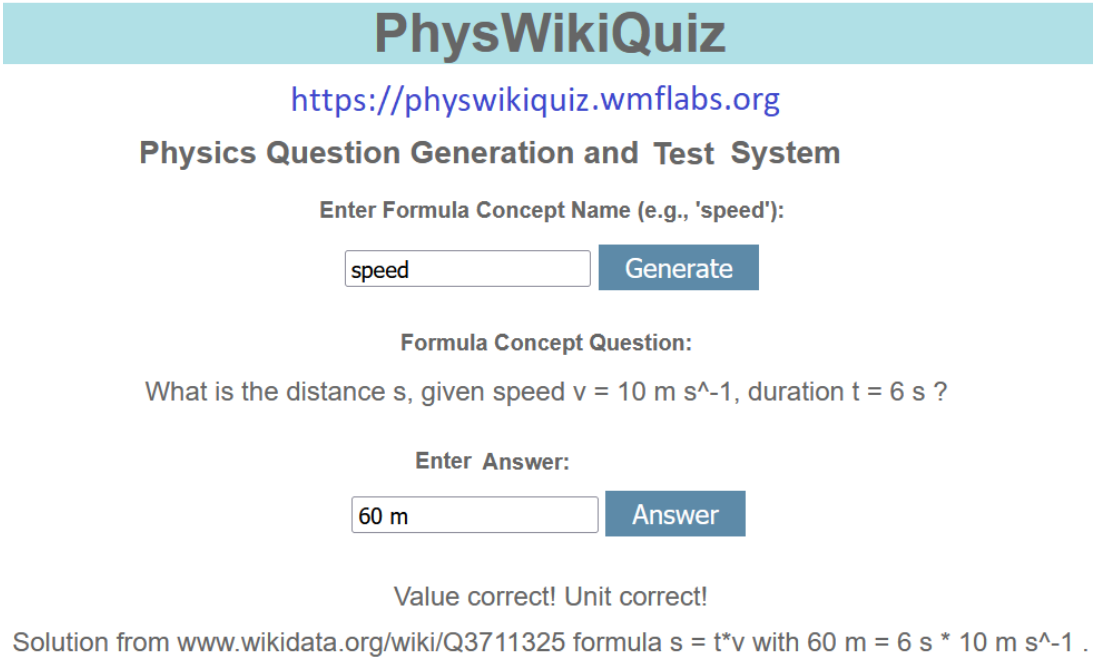

Figure 1: Example question generation (including variable names, symbols, and units) and answer correction (of both solution value and unit) for the formula concept name 'speed'. The PhysWikiQuiz system also generates an explanation text with information reference and calculation path.

as a multilingual framework that allows for collaborative worldwide teacher knowledge engineering and subsequent AI-aided question generation, test, and correction. Using Wikidata in education leads to the research problem need to compare and identify the best-performing methods to generate questions from Wikidata knowledge.

As a proof of concept for the physics domain, we develop and evaluate a »PhysWikiQuiz« question generator and solution test engine (example in Figure 1), hosted by Wikimedia at https://physwikiquiz.wmflabs.org with a demovideo available at https://purl.org/physwikiquiz. The system addresses the teacher's demand by automatically generating an unlimited number of different questions and values for each student separately. It employs the open access semantic knowledge-base Wikidata[1] to retrieve Wikimedia community-curated physics formulae with identifier (variables with no fixed value[2]) properties and units using their concept name as input. A given formula is then rearranged, i.e., solved for each occurring identifier by a Computer Algebra System to create more question sets. For each rearrangement, random identifier values are generated. Finally, the system compares the student's answer input to a CAS computed solution for both value and unit separately. PhysWikiQuiz also provides an API for integration in external education systems or platforms. To evaluate the system, we pose the following research questions for the assessment of test question generation from Wikidata knowledge (RQs):

[1] https://www.wikidata.org
[2] https://www.w3.org/TR/MathML3/chapter4.html#contm.ci

1. What are the state-of-the-art systems? How to address their shortcomings?
2. Which information retrieval methods and databases can we employ?
3. What performance can we achieve?
4. What are the contributions of the system's modules to this performance?
5. What challenges occur during implementation and operation?
6. How can we address these challenges?

Structure. The remainder of this paper is structured as follows. We discuss RQ 1 in Section 2, RQs 2-4 in Section 3 and 4, and RQs 5-6 in Section 5.

2. Background and Related Work

In this section, we review the prerequisite background knowledge the project builds upon, including the research gap and the employed methods.

Question Generation (QG) is a natural language processing task to generate question-answer (QA) pairs from data (text, knowledge triples, tables, images, and more)[3]. The generated QA pairs can then be employed in dialogue systems, such as Question Answering, chatbots, or quizzes. State of the art is to typically use neural networks to generate structured pairs out of unstructured content extracted from crawled web pages [1]. There is a number of datasets and models openly available with a competitive comparison at https://paperswithcode.com/task/question-generation. In the last decade, QA has been increasingly employed and researched for educational applications [2, 3]. Despite the large variety of techniques, in 2014 only a few had been successfully deployed in real classroom settings [4].

Automated Test Generation (ATG) for intelligent tutoring systems has so far been tackled using linked open data ontologies to create natural language multiple-choice questions [5]. The evaluation is typically domain-dependent. For example, Jouault et al. conduct a human expert evaluation in the history domain, comparing automatically with manually generated questions to find about 80% coverage [6]. Some approaches use Wikipedia-based datasets consisting of URLs of Wikipedia articles to generate solution distractors via text similarity [7]. Since Wikipedia is only semi-structured, it may be more efficient to instead employ highly structured databases. This was attempted by 'Clover Quiz', a trivia game powered by DBpedia for multiple-choice questions in English and Spanish. However, the creators observed the system to have high latency, which is intolerable for a live game. The limitations are addressed by creating questions offline through a data extraction pipeline [8]. For the mathematics domain, Wolfram Research released the Wolfram Problem Generator[4] for AI-generated practice problems and answers. The system covers arithmetics, number theory, algebra, calculus, linear algebra, and statistics, yet is restricted to core mathematics while physics is currently not supported. Current systems for the physics domain, e.g., 'Mr Watts Physics' [5] and 'physQuiz' [6], are

[3] https://www.microsoft.com/en-us/research/project/question-generation-qg
[4] https://www.wolframalpha.com/problem-generator
[5] http://wattsphysics.com/questionGen.html
[6] https://physics.mrkhairi.com

curated only by single maintainers, which leads to a very limited availability of concepts and questions (see Table 1).

We address the reported shortcomings by presenting a system for the physics domain that allows for unlimited live question generation from community-curated (Wikidata) knowledge. Since Wikidata is constantly growing, our approach scales better than the aforementioned static resources curated by single teachers. Only 2% of unique concepts were available on 'physQuiz Equations' (8 out of 475) and 8% on 'Mr Watts Physics' (36 out of 475), yet 99% on our 'PhysWikiQuiz' (469 out of 475). In the case of mathematical knowledge, Wikidata currently contains around 5,000 statements that link an item concept name to a formula [9]. As stated above, almost 500 of them are from the physics domain. PhysWikiQuiz exploits this information to create, pose, and correct physics questions using mathematical entity linking [10] (in contrast to the competitors), which we review in the following.

Mathematical Entity Linking (MathEL) is the task of linking mathematical formulae or identifiers to unique web resources (URLs), e.g., Wikipedia articles. This requires formula concepts to be identified (first defined and later recognized). For this goal, a 'Formula Concept' was defined [11, 12] as a 'labeled collection of mathematical formulae that are equivalent but have different representations through notation, e.g., the use of different identifier symbols or commutations' [10]. Formulae appearing in different representations make it difficult for humans and machines to recognize them as instances of the same semantic concept. For example, the formula concept 'mass-energy equivalence' can either be written as $=^2$ or $=/^2$ or using a variety of other symbols. To facilitate and accelerate the creation of a large dataset [13] for the training of Formula Concept Retrieval (FCR) methods, a formula and identifier annotation recommender system for Wikipedia articles was developed [14]. The FCR approaches are intended to improve the performance of Mathematical Information Retrieval (MathIR) methods, such as Mathematical Question Answering (MathQA) [15, 16], Plagiarism Detection (PD), STEM literature recommendation or classification [17, 18].

3. Methods and Implementation

In this section, we describe the development of our PhysWikiQuiz physics question generation and test engine, along with the system workflow and module details. PhysWikiQuiz employs the method of Mathematical Entity Linking (see Section 2).

The prerequisites for the PhysWikiQuiz system are that it 1) is intended to generate questions as part of an education platform, 2) employs Wikidata as knowledge-base,

System	Mr Watts Physics	physQuiz Equations	PhysWikiQuiz
Concepts	36	8	469 (Wikidata)
Questions per concept	20	20	unlimited

Table 1
Comparison of PhysWikiQuiz scope to competitors.

3) works on formula concepts, 4) requires formula and identifier unit retrieval, and 5) utilizes a Computer Algebra System to correct the student's answer.

3.1. System Workflow

Figure 1 shows the PhysWikiQuiz User Interface (UI) for an example formula concept name input 'speed' with a defining formula of = /. The formula can be rearranged as = or = / (two question sets). For the identifier symbols , , and , their names 'velocity', 'distance', and 'duration' and units '`m s^-1`', '`m`', and '`s`' are retrieved from the corresponding Wikidata item[7]. In the example, the answer is considered as correct in both value '60' and unit '`m`'. If the user clicks again on the 'Generate' button, a new question with different formula rearrangement and identifier values is generated. For a system feedback of 'Value incorrect!' and/or 'Unit incorrect!', the student has the possibility to try other inputs by changing the input field content and clicking again on the 'Answer' button.

The PhysWikiQuiz workflow is divided into six modules (abbreviated by Mx in the following). In M1, formula and identifier data is retrieved from Wikidata. In M2, the formula is rearranged using the python CAS Sympy[8]. In M3, random values are generated for the formula identifiers. In M4, the question text is generated from the available information. In M5, the student's answer is compared to the system's solution. Finally, M6 generates an explanation text for the student. In case some step or module cannot be successfully executed, the user is notified, e.g., 'No Wikidata item with formula found.'

3.2. Modules

After the user inputs the formula concept name or Wikidata QID (see Figure 1), M1 retrieves the 'defining formula' and identifier properties. PhysWikiQuiz supports all current identifier information formats and strives to stay up to date. The identifier units need to be retrieved from the linked items (in some formats, also the names). Currently, units are stored using the 'ISQ dimension' property (P4020 in Wikidata). To make the format more readable for students, the unit strings (e.g., `'L T^-1'`) are translated into SI unit symbols[9] (e.g., `'m s^-1'`).

Having retrieved the required formula and identifier information, M2 is called to generate possible rearrangements using the CAS of SymPy[8], a python library for symbolic mathematics [19]. Since the 'defining formula' property of the Wikidata item stores the formula in LaTeX format, which is different from the calculable Sympy CAS representation, a translation is necessary. There are several possibilities available for this task. The python package LaTeX2Sympy[10] is designed to parse LaTeX math expressions and convert it into the equivalent SymPy form. The Java converter LaCASt [20, 21], provided by the VMEXT [22] API[11] translates a semantic LaTeX string to a specified CAS. In our system

[7] https://www.wikidata.org/wiki/Q3711325
[8] https://www.sympy.org
[9] https://en.wikipedia.org/wiki/International_System_of_Quantities
[10] https://github.com/OrangeX4/latex2sympy
[11] https://vmext-demo.formulasearchengine.com/swagger-ui.html

evaluation (Section 4), we compare the performance of both translators. For them to work correctly, PhysWikiQuiz performs a number of LaTeX cleanings beforehand, such as replacements and removals that improve the translation performance.

With the Sympy calculable formula representation available, M3 is ready to replace the right-hand side identifiers with randomly generated integer values. A lower and upper value can be chosen freely. We use the default range from 1 to 10 in our evaluation. Finally, having successfully replaced the right-hand side identifiers by their respective generated random values, the left-hand side identifier value is calculated. The value is later compared to the student input by M5 (answer correction) to check the validity of the question-answer value. At this stage, all information needed to generate a question is available: (1) the formula, (2) the identifier symbols, (3) the identifier (random) values, and (4) the identifier units. M4 generates the question text by inserting the respective information into gaps of a predefined template with placeholders for formula identifier names, symbols, and units. For a question text example, refer to the screenshot in Figure 1.

After the question text is displayed by the UI, the student can enter an answer consisting of value and unit for the left-hand side identifier solution. The information is then parsed by M5. It is subsequently compared to the value output of M1 (solution unit) and M3 (solution value). The student gets feedback on the correctness of value and unit separately. The system accepts fractions or decimal numbers as input (e.g., $5/2 = 2.5$), which is then compared to the solution with a tolerance that can be specified (default value is ś1%). Finally, after the question is generated and the correctness of the solution is assessed by the system, M6 generates an explanation such that the student can understand how a given solution is obtained. The system returns and displays an explanation text storing left- and right-hand side identifier names, symbols, values, and units (see M4). For an explanation text example, refer to the screenshot in Figure 1.

4. Evaluation

In this section, we present and discuss the results of a detailed PhysWikiQuiz system evaluation at each individual stage of its workflow. We carry out module tests for the individual modules and an integration test to assess the overall performance on a formula concept benchmark dataset (see Section 4.1). All detailed tables can be found in the `evaluation` folder of the repository[12].

4.1. Benchmark Dataset

The open-access platform 'MathMLben'[13] stores and displays a benchmark of semantically annotated mathematical formulae [13]. They were extracted from Wikipedia, the arXiv and the Digital Library of Mathematical Functions (DLMF)[14] and augmented by Wikidata

[12] https://github.com/ag-gipp/PhysWikiQuiz/blob/main/evaluation
[13] https://mathmlben.wmflabs.org/
[14] https://dlmf.nist.gov

Translator	quest. OR corr.	quest. AND corr.	only quest.	none
LaTeX2Sympy	48%	20%	29%	52%
LaCASt	44%	26%	18%	56%

Table 2
Comparison of *LaTeX2Sympy* and *LaCASt* translator in overall system performance for question (quest.) generation and correction (corr.) ability.

markup [11]. The benchmark can be used to evaluate a variety of MathIR tasks, such as the automatic conversion between different CAS [13] or MathQA [15]. The system visualizes the formula expression tree using VMEXT [22] to reveal how a given formula is processed. In our PhysWikiQuiz evaluation, we employ a selection of formulae from the MathMLben benchmark. The formula concepts were extracted from Wikipedia articles using the formula and identifier annotation recommendation system [14, 10] »AnnoMathTeX«[15].

4.2. Overall System Performance

Table 3 shows example evaluation results (selection of instances and features) on the MathMLben formula concept benchmark. For each example concept in the benchmark selection, e.g., 'acceleration' (GoldID 310 or Wikidata Q11376), the individual modules are tested individually.

Using a workflow evaluation automation script, we create two separate evaluation tables for the two LaTeX to SymPy translators that we employ (LaTeX2Sympy and LaCASt, see the description of M2 in Section 3.2). The overall system performance using the LaTeX2Sympy converter is the following. For 20% of the concepts, all modules are working properly, and PhysWikiQuiz can provide both a question text, an answer verification with correct internal calculation, and an explanation text. For 29%, only the question can be displayed, but the system's calculation is wrong, such that the answer correction and explanation text generation do not work correctly. For 52%, PhysWikiQuiz cannot provide a question. In summary, the system is able to yield 48% 'question or correction'[16], 20% 'question and correction', 29% question, and 52% none. The overall system performance using the LaCASt converter is the following. For 26% of the concepts, all modules are working properly. For 18%, only the question can be displayed. For 56%, PhysWikiQuiz can not provide a question. In summary, the system is able to yield 44% 'question or correction', 26% 'question and correction', 18% question, 56% none.

Table 2 summarizes the performance comparison of the two translators. We include a detailed discussion of the issues in external dependencies that cause this relatively low performance in Section 5. Overall, LaCASt performs better in generating both question and correction but cannot provide either question or correction on slightly more instances. We deploy LaCASt in production.

[15] https://annomathtex.wmflabs.org
[16] Although the case of 'no question but correction' is not very intuitive, it did occur.

GoldID	QID	Name	Identifier semantics	Formula translation	Explanation text
310	Q11376	acceleration	yes	no	yes
311	Q186300	angular acceleration	yes	no	yes
312	Q834020	angular frequency	yes	yes	yes
Total		**Performance**	**97% yes**	**60% yes**	**27% yes**

Table 3
Three example evaluation results out of a formula concept selection from the benchmark MathMLben (https://mathmlben.wmflabs.org/). Each individual module of the PhysWikiQuiz workflow is evaluated. Here, we only show a summary of the main steps (last three columns condensed from eight, see the repository).

4.3. Module Evaluation

In the following, we present a detailed evaluation of the individual modules or stages in the workflow.

Retrieval Formula Identifier Semantics and Units The first stage of module tests is the assessment of the correct retrieval of the identifier semantics. Since names and symbols are fetched from Wikidata items that are linked to the main concept item, the retrieval process is prone to errors. However, we find that for 97% of the concepts, identifier properties are available in some of the supported formats.

Retrieval of Formula and Identifier Units The next workflow stage we evaluate is the formula and identifier unit retrieval. For 53% of the test examples, a formula unit is available on the corresponding main concept Wikidata item. For the remaining 47%, identifier units are available on the respective linked Wikidata items.

LaTeX to SymPy Translation The subsequent module tests are concerned with the LaTeX to SymPy translations, which is mandatory for having Sympy rearrange the formula and yield a right-hand side value given random identifier value substitutions (modules 2 and 3). We evaluate the two converters LaTeX2Sympy and LaCASt in comparison, which were introduced in Section 3.2. LaTeX2Sympy is able to yield a correct and calculable SymPy formula in 50% of the cases. Moreover, it can provide usable Sympy identifiers for the substitutions in 47% of the cases. For LaCASt, the SymPy formula is correct in 60% and the SymPy identifiers in 47%. This means that LaCASt has a better translation performance (10% more), while the identifier conversion remains the same.

Formula Rearrangement Generation Formula rearrangements enhance the availability of additional question variations. In the case of our example 'speed', when using Sympy rearrangements, the other variables 'distance' and 'durations' can also be queried, providing additional concept questions. For lengthy formulae, PhysWikiQuiz can generate a very large amount of question variations. But even for a small formula with 2 identifiers, there are already many possibilities by substituting different numbers as identifier values.

On average, the formulae in the test set contain 3 identifiers. Substituting combinations of numbers from 1 to 10, this leads to several hundred potential questions per formula concept. We find that in 27% of the cases, Sympy can rearrange the 'defining formula.' The result is the same for both LaTex2Sympy and LaCASt translation. In comparison to a workflow without M2, more than 300 additional questions can be generated.

Right-Hand Side Substitutions and Explanation Text Generation The last two module test evaluations assess the success of right-hand side substitutions and explanation text generation. For LaTeX2Sympy, 45% of substitutions are made correctly, whereas LaCASt achieves 53%. Both translators generate correct identifier symbol-value-unit substitutions for the explanation text in 39% of the test cases.

5. Discussion

In this section, we discuss our results, contribution, and retrieval challenges of the individual workflow stages and modules. The full list of challenges can be found in the repository[12].

Results and Contribution Wikidata currently contains around 5,000 concept items with mathematical formula. Out of these, about 500 are from the physics domain. Using a Computer Algebra System, PhyWikiQuiz can generate concept questions with value and unit and corrections in around 50% of the cases. For a detailed analysis of the errors in the remaining 50% and a discussion of the challenges to tackle, see the next subsection.

Our contribution is a proof of concept for the physics domain to use Wikidata in education. We develop a »PhysWikiQuiz« question generator and solution test engine and evaluate it on an open formula concept benchmark dataset. Our work addresses the research gap in comparing methods to generate physics questions from Wikidata knowledge. We find that using Wikidata and a Computer Algebra System, it is possible to generate an unlimited amount of physics questions for a given formula concept name. Although they all follow a very similar template with very little variation, they contain different variable values, which makes them suitable to provide individual questions for various students.

5.1. Challenges and Limitations

Formula Semantics and Translation We manually examine the concepts for which the Wikidata items do not provide units. For some of them, we identify semantic challenges. In our estimation, the concepts either (1) should not have a unit ('ISQ dimension' property) or (2) it is debatable whether they should have one. Example QIDs for the respective cases can be found in the repository[12]. In the first case, the respective formulae do not describe physical quantities but formalisms, transformations, systems, or objects. In particular, the formula right-hand side identifier that is calculated does not correspond to the concept item name. In the second case, the corresponding formula

provides the calculation of a physical quantity that is not reflected in the concept name. Finally, there is a third case in which the concept item should have a unit property since the formula describes a physical quantity on the right-hand side that is defined by the concept name. Examining the examples for which the converters cannot provide a properly working translation, we find some challenges that require the development of more advanced LaTeX cleaning methods. Derivative fractions can contain identifier differentiation with or without separating spaces. For example, 'acceleration' can be calculated either as `\frac{d v}{d t}` or `\frac{dv}{dt}`. The first formula is correctly translated to the calculable SymPy form `Derivative(v, t)`, whereas the second does not work. Unfortunately, the spaces cannot be introduced automatically in the arguments without losing generality (e.g., `dv` could also mean a multiplication of some identifiers d and v as `d * v`). Implicit multiplication is a general problem. However, it is very likely for a `\frac{}{}` expression with leading `d` symbols in its arguments to contain a derivative, and the risk of losing generality should maybe be taken. In the case of partial derivatives, such as `\frac{\partial v}{\partial t}` the problem does not arise since `\partial` needs a following space to be a proper LaTeX expression. Some formulae are not appropriate for PhysWikiQuiz question generation and test. The expression `\sum_{i=1}^n m_i(r_i - R) = 0` in 'center of mass ' (Q2945123) does not have a single left-hand-side identifier to calculate. The right-hand side is always zero. The equation (correctly) also does not have a formula unit. Finally, expressions like `p_{tot,1} = p_{tot,2}` in 'conservation of momentum' (Q2305665) are no functional linkage of identifier variables and thus do not serve as basis for calculation questions.

Identifier Substitutions and Explanation Text Generation For about half of the test examples, the substitution is unsuccessful due to some peculiarities in the defining formula. The full list can be found in the repository[12]. We encounter the problems that (1) substitutions cannot be made if identifier properties are not available, (2) for some equations, the left-hand side is not a single identifier, but a complex expression or the right-hand side is zero, (3) two equation signs occur in some instances, and (4) identifier properties and formula are not matching in their Wikidata items for some items. The last stage in our workflow evaluation is the assessment of the explanation text correctness. All in all, for 27% of the concepts, explanation texts can be generated, out of which 39% contain correct identifier symbol-value-unit substitutions. We conclude that the calculation path display is error-prone and outline some challenges in the following. For the explanation texts that are incorrect, we identify some of the potential reasons. We find that (1) in some cases, operators like multiplications are missing, (2) some equations contain dimensionless identifiers, for which the unit is written as the number 1, and (3) in case integrals appear in the formulae, sometimes a mixture of non-evaluated expressions and quantities is displayed.

5.2. Takeaways

Answering the research questions. Having implemented and evaluated the system, we can answer our research question as follows:

1. PhysWikiQuiz outperforms its competitors by providing a constantly growing number of more than 10 times additional community-curated questions.
2. We employ and adapt the method of Mathematical Entity Linking of formula concepts for question generation using Wikidata.
3. About 50% of the benchmark formula Wikidata items can be successfully transformed into questions with correction and explanation. For the remaining cases, we provide an extensive error analysis.
4. The performance directly depends on formula and identifier name, symbol and unit retrieval, as well as translation to and solving by a CAS.
5. The bottleneck is caused by the dependencies, such as the CAS Sympy and translator LaCASt. A clearer community agreement on data quality guidelines in Wikidata would also improve the results.
6. We can improve the quality of the formula cleanliness with user feedback by addressing the issues in the dependencies.

Addressing the challenges. To tackle the current limitations, we propose the following solutions:

- Formula semantics: Limit use to concepts that can be indisputably associated with formulae and units to avoid unreliability due to community objection.
- Formula translation: Increasingly improve the converter performance by receiving and implementing community feedback to enhance concept coverage.
- Identifier substitutions: Motivate the Wikidata community to seed the missing identifier properties. This will increase coverage by enabling lacking identifier value substitutions.
- Explanation text generation: The problems are expected to be settled with increased data quality of the formula items in Wikidata.

Despite the challenges, we have already built an in-production system (with 13 times more coverage than its best-performing competitor) that can and will be used by teachers in practice.

6. Conclusion and Future Work

In this paper, we present »PhysWikiQuiz«, a physics question generation and test engine. Our system can provide a variety of different questions for a given physics concept, retrieving formula information from Wikidata, correcting the student's answer, and explaining the solution with a calculation path. We separately evaluate each of the six modules of our system to identify and discuss systematic challenges in the individual stages of the workflow. We find that about half of the questions cannot be generated or corrected due to issues that can be addressed by improving the quality of the external dependencies (Wikidata, LaTeX2Sympy, LaCASt, and Sympy) of our system. Our

application demonstrates the potential of mathematical entity linking for education question generation and correction.

PhysWikiQuiz is listed on the 'Wikidata tool pages' for querying data[17]. We welcome the reader to test our system and provide feedback for improvements. If the population of mathematical Wikidata items continues (e.g., by using tools such as »AnnoMathTeX«[15]), our system will be able to increasingly support additional questions. We will continue to assess the overall effectiveness of the knowledge transfer from Wikipedia articles to Wikidata items to PhysWikiQuiz questions. Moreover, we are developing an automation for the Wikidata physics concept item bulk to detect if the question generation or correction is correct, or if the respective items need human edits to make PhysWikiQuiz work. Detecting these cases will extend the system's operating range and ensures that it works despite the limitations. We also plan to test the system with a larger group of end users.

As a long-term goal, we envision integrating our system into larger education platforms, allowing teachers to simply enter a physics concept about which they want to quiz the students. Students would then receive individually generated questions (via app push notification) on their mobile phones. Having collected all the answers, teachers could then obtain a detailed analysis of the student's strengths and weaknesses and use them to address common mistakes in their lectures. We will evaluate the integrated system with teachers. Finally, we plan to extend our framework with additional question domains, possibly integrating state-of-the-art external dependencies, Wikifunctions[18], and language models as they are developed to increase the coverage further. With PhyWikiQuiz and its extensions to other educational domains, we hope to make an important contribution to the 'Wikidata for Education' project[19].

Acknowledgments

This work was supported by the German Research Foundation (DFG grant GI-1259-1).

References

[1] N. Duan, D. Tang, P. Chen, M. Zhou, Question generation for question answering, in: M. Palmer, R. Hwa, S. Riedel (Eds.), Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017, Association for Computational Linguistics, 2017, pp. 866–874. URL: https://doi.org/10.18653/v1/d17-1090. doi:10.18653/v1/d17-1090.

[2] G. Chen, J. Yang, C. Hauff, G. Houben, Learningq: A large-scale dataset for educational question generation, in: Proceedings of the Twelfth International Conference on Web and Social Media, ICWSM 2018, Stanford, California, USA,

[17] https://www.wikidata.org/wiki/Wikidata:Tools/Query_data
[18] https://wikifunctions.beta.wmflabs.org
[19] https://www.wikidata.org/wiki/Wikidata:Wikidata_for_Education

June 25-28, 2018, AAAI Press, 2018, pp. 481–490. URL: https://aaai.org/ocs/index.php/ICWSM/ICWSM18/paper/view/17857.

[3] M. Srivastava, N. Goodman, Question generation for adaptive education, in: C. Zong, F. Xia, W. Li, R. Navigli (Eds.), Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021, Association for Computational Linguistics, 2021, pp. 692–701. URL: https://doi.org/10.18653/v1/2021.acl-short.88. doi:10.18653/v1/2021.acl-short.88.

[4] N. Le, T. Kojiri, N. Pinkwart, Automatic question generation for educational applications - the state of art, in: T. V. Do, H. A. L. Thi, N. T. Nguyen (Eds.), Advanced Computational Methods for Knowledge Engineering - Proceedings of the 2nd International Conference on Computer Science, Applied Mathematics and Applications, ICCSAMA 2014, 8-9 May, 2014, Budapest, Hungary, volume 282 of *Advances in Intelligent Systems and Computing*, Springer, 2014, pp. 325–338. URL: https://doi.org/10.1007/978-3-319-06569-4_24. doi:10.1007/978-3-319-06569-4_24.

[5] V. E. V, P. S. Kumar, Automated generation of assessment tests from domain ontologies, Semantic Web 8 (2017) 1023–1047. URL: https://doi.org/10.3233/SW-170252. doi:10.3233/SW-170252.

[6] C. Jouault, K. Seta, Y. Hayashi, Content-dependent question generation using lod for history learning in open learning space, Transactions of the Japanese Society for Artificial Intelligence (2016) LOD–F.

[7] R. Shah, D. Shah, L. Kurup, Automatic question generation for intelligent tutoring systems, in: 2017 2nd International Conference on Communication Systems, Computing and IT Applications (CSCITA), IEEE, 2017, pp. 127–132.

[8] G. Vega-Gorgojo, Clover quiz: A trivia game powered by dbpedia, Semantic Web 10 (2019) 779–793. URL: https://doi.org/10.3233/SW-180326. doi:10.3233/SW-180326.

[9] P. Scharpf, M. Schubotz, B. Gipp, Mathematics in wikidata, in: Wikidata@ISWC, volume 2982 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2021.

[10] P. Scharpf, M. Schubotz, B. Gipp, Fast linking of mathematical wikidata entities in wikipedia articles using annotation recommendation, in: WWW (Companion Volume), ACM / IW3C2, 2021, pp. 602–609.

[11] P. Scharpf, M. Schubotz, B. Gipp, Representing mathematical formulae in content mathml using wikidata, in: BIRNDL@SIGIR, volume 2132 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2018, pp. 46–59.

[12] P. Scharpf, M. Schubotz, H. S. Cohl, B. Gipp, Towards formula concept discovery and recognition, in: BIRNDL@SIGIR, volume 2414 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2019, pp. 108–115.

[13] M. Schubotz, A. Greiner-Petter, P. Scharpf, N. Meuschke, H. S. Cohl, B. Gipp, Improving the representation and conversion of mathematical formulae by considering their textual context, in: JCDL, ACM, 2018, pp. 233–242.

[14] P. Scharpf, I. Mackerracher, M. Schubotz, J. Beel, C. Breitinger, B. Gipp, *AnnoMath*

TeX - a formula identifier annotation recommender system for STEM documents, in: RecSys, ACM, 2019, pp. 532–533.

[15] M. Schubotz, P. Scharpf, K. Dudhat, Y. Nagar, F. Hamborg, B. Gipp, Introducing mathqa - A math-aware question answering system, Information Discovery and Delivery 42, No. 4 (2019) 214–224. doi:10.1108/IDD-06-2018-0022.

[16] P. Scharpf, M. Schubotz, A. Greiner-Petter, M. Ostendorff, O. Teschke, B. Gipp, Arqmath lab: An incubator for semantic formula search in zbmath open?, in: CLEF (Working Notes), volume 2696 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2020.

[17] P. Scharpf, M. Schubotz, A. Youssef, F. Hamborg, N. Meuschke, B. Gipp, Classification and clustering of arxiv documents, sections, and abstracts, comparing encodings of natural and mathematical language, in: JCDL, ACM, 2020, pp. 137–146.

[18] M. Schubotz, P. Scharpf, O. Teschke, A. Kühnemund, C. Breitinger, B. Gipp, Automsc: Automatic assignment of mathematics subject classification labels, in: CICM, volume 12236 of *Lecture Notes in Computer Science*, Springer, 2020, pp. 237–250.

[19] D. Joyner, O. Certík, A. Meurer, B. E. Granger, Open source computer algebra systems: Sympy, ACM Commun. Comput. Algebra 45 (2011) 225–234.

[20] A. Greiner-Petter, M. Schubotz, H. S. Cohl, B. Gipp, Semantic preserving bijective mappings for expressions involving special functions between computer algebra systems and document preparation systems, Aslib J. Inf. Manag. 71 (2019) 415–439.

[21] A. Greiner-Petter, M. Schubotz, C. Breitinger, P. Scharpf, A. Aizawa, B. Gipp, Do the math: Making mathematics in wikipedia computable, IEEE Transactions on Pattern Analysis and Machine Intelligence (2022).

[22] M. Schubotz, N. Meuschke, T. Hepp, H. S. Cohl, B. Gipp, VMEXT: A visualization tool for mathematical expression trees, in: CICM, volume 10383 of *Lecture Notes in Computer Science*, Springer, 2017, pp. 340–355.