An Exploration of Open Source Small Language Models for Automated Assessment

Andrea Sterbini
Sapienza University of Rome, Italy
sterbini@di.uniroma1.it

Marco Temperini
Sapienza University of Rome, Italy
marte@diag.uniroma1.it

2024 28th International Conference Information Visualisation (IV) | 979-8-3503-8016-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/IV64223.2024.00064

***Abstract*—We explore the classification and assessment capabilities of a selection of Open Source Small Language Models, on the specific task of evaluating learners' Descriptions of Algorithms. The algorithms are described in the framework of programming assignments, to which the learners in a class of Basics in Computer Programming have to answer. The task requires to 1) provide a program, in Python, to solve the assigned problem, 2) submit a description of the related algorithm, and 3) participate in a formative peer assessment session, over the submitted algorithms. Can a Language Model, be it small or large, produce an assessment for the algorithm descriptions? Rather than using any of the most famous, huge, and proprietary models, here we explore Small, Open Source based, Language Models, i.e. models that can be run on relatively small computers, and whose functions and training sources are provided openly. We produced a ground-truth evaluation of a large set of algorithm descriptions, taken from one year of use of the Q2A-II system. In this we used an 8-value scale, grading the usefulness of the description in a Peer Assessment session. Then we tested the agreement of the models assessments with such ground-truth. We also analysed whether a pre-emptive, automated, binary classification of the descriptions (as useless/useful for a Peer Assessment activity) would help the models to grade the usefulness of the description in a better way.**

***Index Terms*—Keywords: Technology Enhanced Learning, Peer Assessment, Automated Assessment, Algorithm Description Quality Natural language processing (NLP), Transformer-based Large and Small Language Models, Open Source Small Language Models**

I. Introduction

For several years now, we have been employing Automated Assessment of Computer Programming Tasks (AACPT), at University level, through the system Q2A-II [1], [2], collecting a wealth of data about the students' homeworks. AACPT is considered very useful to support students to acquire and refine programming skills [3]–[5]. Q2A-II analyses a programming homework (on basics in Python programming language) from several viewpoints: unit testing is used to evaluate the program output correctness, and automated feedback is provided about code complexity. After such checks, the student can improve and resubmit the homework.

However, while it is implicit that a student should think about the algorithmic solution to the problem before coding, this aspect of the task is seldom considered in assessment systems. Algorithm definition and description skills are actually of great relevance in computer science and engineering courses. Moreover, they might be an important learning goal in a variety of other study curricula, such as in social sciences, music, architecture, or medicine [6]–[8].

The programming homework requires describing the algorithm used in the solution, and on this description a formative Peer Assessment (PA) activity is conducted, so that other students can provide feedback and suggestions for improvement. The learner can profit from the suggestions and submit a better version of the homework. The suggestions are eventually graded by the peers w.r.t. their usefulness. These PA activities are not mandatory, however the great majority of the learners participated (though at different levels of thoroughness).

One important aspect in the PA activity is in that, for the peers to have a real chance of producing a useful feedback, the algorithm description ought to be of good quality. By "good quality" we mean *a description that actually describes an algorithm*: be it correct or not, to allow for a punctual and useful feedback by a peer.

Because of the course numerosity (circa 500 submission per homework), a manual evaluation of the algorithm descriptions would be practically unfeasible, so it comes quite natural the idea of automatically assess the descriptions.

In the last very few years, Large Language Models (LLMs) have been used for several tasks related to learners' activity support and assessment [9]–[12], about programming tasks too [13], [14].

In general, the GPT (Generative Pre-Trained) model technology is applied in these studies (with ChatGPT in its versions from 3 to 4). On the other hand, beside proprietary LLMs, many new Open Source models of small size, but good performance have been released [14]–[20].

In this paper we focus on open-source Small Language Models (SLMs, models that would fit in 16GB RAM), and explore their use on the assessment of the algorithm description quality.

Provided a suitable prompting, and the definitions of the grades to assess the algorithms, we compared the selected SLMs, w.r.t. the task grading each description. Our research questions (RQs) can be listed as follows:

RQ1 *To what extent the analysed SLMs are able to perform a reliable assessment of the algorithm description quality?*

We noticed, from the ground truth we defined by manually assessing one year of homeworks, that the answers with the lowest grades would be useless for PA activity. Actually it would rather make sense to grade only the remaining

2375-0138/24/$31.00 ©2024 IEEE
DOI 10.1109/IV64223.2024.00064

algorithms. So we considered whether the automated grading performance on the sufficient algorithms would improve. Consequently, we added the following RQs to our study:

RQ2 *How much the SLMs are good at making a binary classification of the algorithm descriptions, to detect those that can't be considered useful for the following PA activity?*

The binary classification would part the descriptions in "Non Algorithm" and "Algorithm" ones; then

RQ3 *Will the SLMs produce a better assessment, if they were asked to grade only the "Algorithm" subset of descriptions?*

In the next section we present some notes about related research; then we describe the method of our experimental activity, in Sec. III. Sec. IV will discuss our findings, and the last section will wrap-up with final considerations about current and future work.

II. Related Work

The use of SLMs/LLMs to support educational tasks is a somewhat recent research activity, since only recently effective models have been made available, with new and improved models being released or updated at a fast pace.

In [21] the use of an LLM (GPT3.5) for automated essay scoring is studied. The results show that students and teachers could benefit from such a tool, provided that they are suitably instructed to use it effectively and ethically, in particular while learning a second language. The authors also add that such an "ever-present assistant" can not be yet considered as a replacement for human intervention.

In [9] the feedback produced by ChatGPT is studied from the viewpoints of readability, correctness (with respect to teacher's assessment), and effectiveness of the feedback aspects. The conclusions are in that 1) the feedback has good readability, granting that the "feedback message" could be easily grasped by the learner, and used to improve; 2) the assessment correctness is lower than expected, although further attempts to train the LLM could allow getting better results; 3) the feedback is often process-focused, rather than just evaluating an outcome, so is being more useful for the leaner.

In [13], an experiment on several programming tasks is described. In 3 out of four tasks, a group of participants was allowed to get feedback from an LLM (a GPT3.5-based chatbot). 46% of the students praised the system's feedback, and 19% found it useless. The authors discuss strategies to improve the educational use of LLMs, such as fine-tuning on the course topics, and development of better prompting techniques. Moreover, the authors point out that the learners appeared to rapidly become "too accustomed" to the automated feedback, and found difficulties and needed time to get back to the previous state (without automated feedback). This might mean that the students were not using the feedback as a chance for further curious and creative study, which may be negative, and so the part of the experiment not providing automated feedback resulted particularly revealing and useful.

In [10] the authors state that several alternatives to OpenAI GPT model are available, and some of them are "multilingual, having been trained on corpora from multiple languages, albeit with a strong bias towards English". Be they open source or not, the authors consider activities, for the LLMs, applicable in the educational field, such as content creation, adaptation, assessment, feedback. It is noted that "[this] is a fast evolving research field, one being driven by industry developments", and expressed preoccupations for the kernel of risk this would involve. Risks are in the lack of needed safeguards on the use of technology for education, lack of public knowledge about the source of training data (with consequent risk of great/hidden bias in these LLMs' answers), and too easily given trust to such answers. Ethical issues, related to data protection, environmental impact of big and so-growing models, and internet addiction, are also mentioned.

We very much agree with the above-mentioned caveat (and sorry for polarizing the above citation mainly on them). The idea of studying the behaviour of open source and open trained models is, in fact, among our motivations for the work described in this paper.

III. Experimental Activity on the Selected SLMs

We summarize here method and results of this study.

From Q2A-II we extracted one year of submissions (circa 1600, year 2019-20) and graded the algorithm descriptions with respect to their clearness and usefulness to get suggestions from peers. The grades were given by one domain expert, which also defined the prompt describing the grading criteria to the SLMs. In a near future the dataset will be graded by other domain experts to improve its reliability. Then we asked, to a selection of SMLs (described in III-A), to grade the algorithm descriptions' quality, on an 8-grades scale ("BAD" to "GOOD", Fig.2, prompt in III-B). In this way, we initially compared the models' grading performance.

Moreover, we asked for a binary classification of the algorithm descriptions, to determine those insufficient to foster the PA activity (Fig.1, prompt in III-B). A second binary classification was also obtained by considering as insufficient those descriptions that were graded "BAD" or "LIMBAD", in the 8-values scale, by the models. Both classifications were in agreement with the ground truth, although with different degrees of success (see IV-D and IV-E).

This allowed us, by limiting the automated grading to the sufficient descriptions, to improve the automated grading performance. This happens with both the first (IV-C) and the second (IV-E) above-mentioned binary classifications.

A. Analysed Small Language Models

We have selected six of the most recent open-source SLMs designed for code generation, code analysis and code description tasks (see Fig. I) [17]–[20], [22]. These models have been all released recently by big companies with Open Source licences. They normally are available in a variety of dimensions, from small (7 billion parameters), that can run fast within 16GB of Ram, to big (70 billion parameters), that

need 64GB Ram and run an order of magnitude slower. In the following, we have chosen the smaller ones to test systems that could be easily managed without the need for huge servers, or complex cluster architectures of servers.

B. Prompt definitions

We asked the models two questions:

first question: IsAlgorithmP?: Given the **task** and a student's **algorithm description**, is the algorithm clear and related to the task description? (see Fig. 1)

second question: Generate Grade: Grade the previous algorithm description by the following grade scale (see Fig. 2)

C. Answer Detection

The models' answers can be verbose, and sometimes without the explicitly requested YES/NO (or grade label) answer.

To recognize the verbose answers, we searched for an explicit YES or NO (or grade) word. As we would not add a semantic analysis of the answer at this stage of the work, we got some missing answers, which were particularly large for a couple of SLMs, as shown from the figures. The remaining 4 models, instead, produced concise answers that worked well with our simple answer detection method.

IV. Findings

A. Generated grades error

We first compute the grading error, by converting the generated and teacher grades/labels into a 0..7 scale, then computing the absolute difference between the two values.

The columns of Tab. II show:

YES, NO: how many descriptions have been marked as IsAlgorithmP=YES or NO by the first question posed to the SLM

MSE: the Mean Squared Error

RMSE: the Root Mean Squared Error

MAE: the Mean Absolute Error

STD: the Standard Deviation

Accuracy: the percentage of exactly graded descriptions

Accuracy +-1: the percentage of generated grades within 1 grade from the teacher one

The StDev is low enough that we can consider **RQ1** answered positively when using the **mistral** model.

B. On the asymmetric distribution of IsAlgorithmP=YES/NO

We check that the asymmetric distribution of non-algorithms to algorithms (almost 20% NO and 80% YES) is not distorting our results, by randomly selecting the same number of IsAlgorithmP=YES and NO, i.e. by reducing the bigger group to the size of the smaller one with random sampling. The resulting stdev increases very little, thus we conclude that the asymmetric distribution is not responsible for the earlier relatively low errors (see Tab. III).

C. Errors on the subset IsAlgorithmP=YES

When we select the subset of algorithms detected as IsAlgorithmP=YES, the stdev lowers to 0.88, improving the precision of the grades obtained in the second question posed to the SLM (see Tab. IV).

D. Quality of the first answer

We check how well the first SLM answer (IsAlgorithmP=YES/NO) classifies the algorithm descriptions. In this case we consider the teacher grades [BAD, LIMBAD] as the true values for NO and the remaining ones as YES. We select this particular asymmetric selection both because we want to simply distinguish non-algorithms from documents that contains at least some algorithmic description. Moreover, because it corresponds to earlier classification results we obtained with automatic clustering of documents based on text similarity [23]. The results in Tab. V show very good results for the **llama3** model. This answers positively to **RQ2**.

E. Binary classification based on generated grades

If, in reverse, we use the generated grades (the SLM answers to the second question) as a binary classifier where grades [BAD, LIMBAD] means FAIL and the rest PASS, we see that the SLM grades works very well, when compared to the same definition of FAIL/PASS for the teacher grades (see Tab. VI).

When we consider only the subset of good descriptions as classified by the previous binary classifier, we get a better result (see Tab. VII), which is expected. This answers positively to **RQ3**.

In many of the above cases, the two best performing SLMs are **llama3** and **mistral**. This could be related to the fact that the algorithm and task descriptions are in Italian, that **mistral** is a multilingual model trained also with Italian texts [17], and that **llama3** is mainly trained in English but then is fine-tuned on texts in other languages [22]. Next runner-up is **mixtral**, which is larger (8x7B parameters), 3-4 times slower, and has been produced by MistralAi, which have released also **mistral**.

V. Discussion and Conclusions

We presented an exploration of the assessment capabilities of several Open Source SLMs, applied to the students' algorithm descriptions collected in our Q2A-II system. We were interested in *the quality of algorithm descriptions "as descriptions"* rather than in their correctness, because they must be useful for a formative Peer Assessment activity, and an insufficient description would be useless. The results of our experiments show that the grading performance of the models is varied and (as an answer to RQ1) the best performances on exact grading are low, while a bit better when +/-1 grade of precision is accepted. Such performance improves once we limit the automated grading to the subset of descriptions "sufficient" to foster the PA activity (giving a positive answer to RQ3). In this respect, RQ2 has a positive answer too, as we saw that the binary classification performed by the models has high F1 values, when compared with the ground truth. This is

Model	PARAMS	Models from	Released	Learned on
wizardlm2	7B	Microsoft	apr 2024	Multilingual
codellama	7B	Meta/Facebook (based on llama2)	aug 2023	English
codegemma	9B	Google	may 2024	English
mixtral	8x7B	MistralAI (mixture of 8 experts)	dec 2023	English, French, Italian, German and Spanish
llama3	8B	Meta/Facebook	apr 2024	English + multilanguage refinement
mistral	7B	MistralAI	sep 2023	English ?

TABLE I
OPEN SOURCE SMALL LANGUAGE MODELS EXAMINED.

```
You are a teacher of Python. Your students are learning Python programming at the university
level of first year in a bachelor on Computer Science. Your students have the task to define an
algorithm to solve the following exercise:
START EXERCISE TEXT
{master}
END EXERCISE TEXT
The algorithm description is to be assessed with respect to its quality as a description, rather
than its correctness as a solution to the programming problem. A good description of the
algorithm is considered the best way to allow the student to discuss with her/his peers about
the correctness of the algorithm, and possible corrections to make it better. Please tell me if
the following algorithm description clearly describes an algorithm related to the problem
defined in the exercise text, by stating either 'YES' or 'NO'; Here is the algorithm description:
START ALGORITHM DESCRIPTION
{algorithm}
END ALGORITHM DESCRIPTION
```

Fig. 1. First question posed to the SLMs.

```
Please grade the previous algorithm description, by one of the grades
'BAD','LIMBAD','LIMITED','LIMPASS', 'PASSLIM','PASS','PASSGOOD','GOOD'.
Consider that the above grades are defined as follows:
- 'BAD': the description doesn't provide a perceivable algorithm definition, it is not
to the point with respect to the exercise text, or it is a generic help request.
- 'LIMBAD': this is an attempt at describing some parts of the algorithm, but there is
too little to consider it as an algorithm description.
- 'LIMITED': there is an attempt to describe the algorithm, but such description is
very short and incomplete, so that a coherent assessment is not  feasible. This is too
poor to really guide alone the implementation of a program. This is too poor to allow
to help making it better with some pointing out of errors and corrections. It may be
good to start clarifying the author's mind.
- 'LIMPASS': The algorithm description is better than LIMITED, yet it is closer to
LIMITED than to PASS.
- 'PASSLIM': The algorithm description is almost PASS. Some details are missing to
consider it a comprehensive guide to the implementation of the program, however the
idea of the algorithm is described.
- 'PASS': The algorithm description is quite clear, it lacks some details and is still
not a complete guide for the implementation of a program. However, it is likely that
this description can allow for discussion with peers about how to make the algorithm
more complete.
- 'PASSGOOD': The algorithm is well described and with several details. It is rather
discursive, and sometimes with steps not clearly separated and ordered. This
description provides a fair source for discussion with the peers, toward making the
algorithm better, or correct if it is not correct.
- 'GOOD': Complete description and rich amount of details; there is a clear
sequencing/enumeration of the steps. This description provides a good source for
discussion with the peers, toward making the algorithm better, or correct if it is
not correct.
Answer only the grade, without any explanation.
```

Fig. 2. Second question posed to the SLMs.

Model	IsAlgorithm	YES	NO	MSE	RMSE	MAE	STD	Accuracy	Accuracy +-1
codegemma	ALL	781	805	4.27	2.07	1.62	1.28	19%	52%
codellama	ALL	227	569	6.68	2.58	2.02	1.61	18%	44%
llama3	ALL	1319	270	3.49	1.87	1.43	1.20	22%	62%
mistral	**ALL**	**1116**	**396**	**2.65**	**1.63**	**1.28**	**1.01**	**23%**	**64%**
mixtral	ALL	920	660	4.08	2.02	1.57	1.27	18%	58%
wizardlm2	ALL	395	661	7.64	2.76	2.11	1.78	18%	47%

TABLE II
ERRORS OF THE GENERATED GRADES W.R.T. THE TEACHER GRADE FOR ALL ALGORITHMS. THE BEST STDEV (1.01) IS OBTAINED WITH THE **MISTRAL** MODEL.

Model	IsAlgorithm	YES	NO	MSE	RMSE	MAE	STD	Accuracy	Accuracy +-1
codegemma	reduced	781	781	4.19	2.05	1.61	1.27	19%	53%
codellama	reduced	227	227	7.80	2.79	2.20	1.73	19%	39%
llama3	reduced	270	270	3.11	1.76	1.30	1.20	29%	65%
mistral	**reduced**	**396**	**396**	**2.53**	**1.59**	**1.21**	**1.03**	**27%**	**66%**
mixtral	reduced	660	660	4.41	2.10	1.63	1.32	17%	58%
wizardlm2	reduced	395	395	5.78	2.41	1.81	1.59	20%	53%

TABLE III
ERRORS FOR REDUCED SUBSET OF GENERATED GRADES W.R.T. THE TEACHER GRADE, WITH THE SAME AMOUNT OF YES AND NO ANSWERS TO FIRST QUESTION. STDEV IS SLIGHTLY HIGHER (1.03) FOR THE **MISTRAL** MODEL.

Model	IsAlgorithm	YES	NO	MSE	RMSE	MAE	STD	Accuracy	Accuracy +-1
codegemma	YES	781	0	3.06	1.75	1.35	1.11	26%	58%
codellama	YES	227	0	5.65	2.38	1.81	1.54	24%	48%
llama3	YES	1319	0	3.59	1.89	1.49	1.18	19%	61%
mistral	YES	1116	0	2.69	1.64	1.35	0.93	16%	63%
mixtral	**YES**	**920**	**0**	**2.20**	**1.48**	**1.19**	**0.88**	**21%**	**68%**
wizardlm2	YES	395	0	2.92	1.71	1.35	1.05	19%	62%

TABLE IV
ERRORS FOR THE ISALGORITHMP=YES SUBSET OF GENERATED GRADES W.R.T. THE TEACHER GRADES. STDEV IMPROVES TO 0.88 FOR THE **MIXTRAL** MODEL.

	Precision	Recall	F1	Accuracy	Specificity
codegemma	98.8%	52.6%	68.7%	55.9%	93.0%
codellama	97.8%	31.9%	48.1%	40.2%	95.3%
llama3	**98.5%**	**88.8%**	**93.4%**	**88.4%**	**84.1%**
mistral	99.2%	79.4%	88.2%	80.5%	92.8%
mixtral	98.9%	62.7%	76.7%	65.1%	92.2%
wizardlm2	99.2%	39.3%	56.3%	43.3%	96.1%

TABLE V
QUALITY OF THE ISALGORITHMP=YES/NO ANSWER W.R.T. THE TEACHER GRADES (WHERE WE TAKE NO=[BAD, LIMBAD] GRADES). WE GET THE BEST RESULTS FOR MODEL **LLAMA3**.

Model	IsAlgorithm	YES	NO	TP	TN	FP	FN	Prec	Recall	F1	Acc	Spec
codegemma	ALL	781	805	1248	43	88	216	93%	85%	89%	81%	33%
codellama	ALL	227	569	1345	13	114	117	92%	92%	92%	85%	10%
llama3	**ALL**	**1319**	**270**	**1400**	**95**	**36**	**73**	**97%**	**95%**	**96%**	**93%**	**73%**
mistral	**ALL**	**1116**	**396**	**1464**	**27**	**97**	**8**	**94%**	**99%**	**97%**	**93%**	**22%**
mixtral	ALL	920	660	1088	90	41	385	96%	74%	84%	73%	69%
wizardlm2	ALL	395	661	959	83	45	478	96%	67%	79%	67%	65%

TABLE VI
QUALITY OF A BINARY CLASSIFICATION BASED ON THE GENERATED GRADES W.R.T. THE TEACHER GRADES (WITH FAIL=[BAD, LIMBAD] AND PASS EQUAL TO THE REMAINING GRADES).

Model	MSE GG	RMSE GG	MAE GG	STD GG	Accuracy GG	Accuracy +-1 GG
mistral	2.69	1.64	1.30	1.00	22%	64%
llama3	3.63	1.91	1.49	1.19	20%	61%
mixtral	**2.54**	**1.59**	**1.26**	**0.97**	**22%**	**66%**
codegemma	3.36	1.83	1.45	1.12	22%	56%
codellama	5.83	2.41	1.88	1.51	19%	47%
wizardlm2	3.47	1.86	1.44	1.18	23%	60%

TABLE VII
ERRORS FOR THE SUBSET OF DESCRIPTIONS SELECTED BY A BINARY CLASSIFICATION BASED ON THE GENERATED GRADES (SECOND QUESTION).

true for both the classification criteria we used: one based on the models' answers to the first question, and the other based on the low grades assigned by the models to the descriptions.

Educators that want to get help in grading from SLMs can process submissions as we did, by giving the SLM a prompt with a clear definition of the grade scale, together with the essay to be assessed. We did not ask for grades explanations, but to do that is trivial.

There appears to be plenty of future work to be done on this line of research, such as: improving the SLM answer recognition, refining the prompts to get more explicit answers, widen the analysis to medium and large-sized language models (the available open-source ones range from 70 billions to 300 billions parameters). Moreover, this kind of technology could be applied to the assessment/classification of the peers' suggestions for algorithm improvement, or to the comparison of the described algorithm to the actual code submitted.

REFERENCES

[1] S. Papandrea, A. Sterbini, M. Temperini, and E. Popescu, "Q2A-I: a support platform for computer programming education, based on automated assessment and peer learning," in *Advances in Web-Based Learning–ICWL 2018: 17th International Conference, Chiang Mai, Thailand, August 22-24, 2018, Proceedings 17*, ser. LNCS, vol. 11007. Springer, 2018, pp. 3–13.

[2] A. Sterbini and M. Temperini, "Q2A-II, a system to support peer assessment on homework: A study on four years of use," in *Emerging Technologies for Education, Proc. 8th International Symposium on Emerging Technologies for Education, SETE 2023*, ser. LNCS, vol. in printing. Springer, 2024, pp. 1–15.

[3] V. Pieterse, "Automated assessment of programming assignments," in *Proc. CSERC'13*, 2013, pp. 45–56.

[4] M. Pozenel, L. Furst, and V. Mahnic, "Introduction of the automated assessment of homework assignments in a university-level programming course," in *Proc. MIPRO'15*, 2015, pp. 761–766.

[5] J. C. Paiva, J. P. Leal, and Á. Figueira, "Automated assessment in computer science education: A state-of-the-art review," *ACM Transactions on Computing Education (TOCE)*, vol. 22, no. 3, pp. 1–40, 2022.

[6] T.-C. Hsu, S.-C. Chang, and Y.-T. Hung, "How to learn and how to teach computational thinking: Suggestions based on a review of the literature," *Computers & Education*, vol. 126, pp. 296–310, 2018.

[7] K. Ostrowska-Wawryniuk, M. Strzała, and J. Słyk, "Form follows parameter: Algorithmic-thinking-oriented course for early-stage architectural education," *Nexus Network Journal*, vol. 24, no. 2, pp. 503–522, 2022.

[8] C. Lu, R. Macdonald, B. Odell, V. Kokhan, C. Demmans Epp, and M. Cutumisu, "A scoping review of computational thinking assessments in higher education," *Journal of Computing in Higher Education*, vol. 34, no. 2, pp. 416–461, 2022.

[9] W. Dai, J. Lin, H. Jin, T. Li, Y.-S. Tsai, D. Gašević, and G. Chen, "Can large language models provide feedback to students? a case study on chatgpt," in *2023 IEEE International Conference on Advanced Learning Technologies (ICALT)*, 2023, pp. 323–325.

[10] A. Caines, L. Benedetto, S. Taslimipoor, C. Davis, Y. Gao, O. Andersen, Z. Yuan, M. Elliott, R. Moore, C. Bryant, M. Rei, H. Yannakoudakis, A. Mullooly, D. Nicholls, and P. Buttery, "On the application of large language models for language teaching and assessment technology," 2023.

[11] E. Kasneci, K. Sessler, S. Küchemann, M. Bannert, D. Dementieva, F. Fischer, U. Gasser, G. Groh, S. Günnemann, E. Hüllermeier, S. Krusche, G. Kutyniok, T. Michaeli, C. Nerdel, J. Pfeffer, O. Poquet, M. Sailer, A. Schmidt, T. Seidel, M. Stadler, J. Weller, J. Kuhn, and G. Kasneci, "Chatgpt for good? on opportunities and challenges of large language models for education," *Learning and Individual Differences*, vol. 103, p. 102274, 2023.

[12] A. Bewersdorff, K. Seßler, A. Baur, E. Kasneci, and C. Nerdel, "Assessing student errors in experimentation using artificial intelligence and large language models: A comparative study with human raters," *Computers and Education: Artificial Intelligence*, vol. 5, no. 100177, pp. 1–40, 2023.

[13] M. Pankiewicz and R. S. Baker, "Large language models (gpt) for automating feedback on programming assignments," 2023.

[14] D. Guo, Q. Zhu, D. Yang, Z. Xie, K. Dong, W. Zhang, G. Chen, X. Bi, Y. Wu, Y. K. Li, F. Luo, Y. Xiong, and W. Liang, "Deepseek-coder: When the large language model meets programming – the rise of code intelligence," 2024.

[15] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, and S. e. a. Bhosale, "Llama 2: Open foundation and fine-tuned chat models," 2023.

[16] E. Nijkamp, H. Hayashi, C. Xiong, S. Savarese, and Y. Zhou, "Codegen2: Lessons for training llms on programming and natural languages," in *Proc. 11th International Conference on Learning Representations, ICLR2023*, 2023. [Online]. Available: https://openreview.net/group?id=ICLR.cc/2023

[17] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed, "Mistral 7b," 2023.

[18] C. Xu, Q. Sun, K. Zheng, X. Geng, P. Zhao, J. Feng, and D. J. Chongyang Tao, "Wizardlm: Empowering large language models to follow complex instructions," 2023.

[19] B. Rozière, J. Gehring, F. Gloeckle, S. Sootla, I. Gat, X. E. Tan, Y. Adi, J. Liu, R. Sauvestre, T. Remez, J. Rapin, A. Kozhevnikov, I. Evtimov, J. Bitton, M. Bhatt, C. C. Ferrer, A. Grattafiori, W. Xiong, A. Défossez, J. Copet, F. Azhar, H. Touvron, L. Martin, N. Usunier, T. Scialom, and G. Synnaeve, "Code llama: Open foundation models for code," 2024.

[20] CodeGemma Team, A. J. Hartman, A. Hu, C. A. Choquette-Choo, H. Zhao, J. Fine, J. Hui, J. Shen, J. Kelley, J. Howland, K. Bansal, L. Vilnis, M. Wirth, N. Nguyen, P. Michel, P. Choy, P. Joshi, R. Kumar, S. Hashmi, S. Agrawal, S. Zuo, T. Warkentin, and Z. e. a. Gong, "Codegemma: Open code models based on gemma," 2024. [Online]. Available: https://goo.gle/codegemma

[21] A. Mizumoto and M. Eguchi, "Exploring the potential of using an ai language model for automated essay scoring," *Research Methods in Applied Linguistics*, vol. 2, no. 2, p. 100050, 2023.

[22] AIMeta, "Llama 3 model card," 2024. [Online]. Available: https://llama.meta.com/docs/model-cards-and-prompt-formats/meta-llama-3/

[23] A. Sterbini and M. Temperini, "Automated analysis of algorithm descriptions quality, through large language models," in *Intelligent Tutoring Systems, International Conference, ITS2024, Proc.*, ser. LNCS. Springer, 2024, p. in printing.