information

MDPI

Article

LLM-as-a-Grader: Practical Insights from Large Language Models for Short-Answer and Report Evaluation

Grace Byun [†], **Swati Rajwal** [†] **and Jinho D. Choi** [*]

Department of Computer Science and Informatics, Emory University, Atlanta, GA 30322, USA; grace.byun@emory.edu (G.B.); swati.rajwal@emory.edu (S.R.)

* Correspondence: jinho.choi@emory.edu

† These authors contributed equally to this work.

Abstract

Large Language Models (LLMs) are increasingly explored for educational tasks such as grading, yet their alignment with human evaluation in real classrooms remains underexamined. In this study, we investigate the feasibility of using OpenAI GPT-4o to evaluate short-answer quizzes and project reports in an undergraduate Computational Linguistics course. We collect responses from approximately 50 students across five quizzes and receive project reports from 14 teams. LLM-generated scores are compared against human evaluations conducted independently by the course teaching assistants (TAs). Our results show that GPT-4o achieves strong correlation with human graders (up to 0.98) and exact score agreement in 55% of quiz cases. For project reports, it also shows strong overall alignment with human grading, while exhibiting some variability in scoring technical, open-ended responses. We release all code and sample data to support further research on LLMs in educational assessment. This work highlights both the potential and limitations of LLM-based grading systems and contributes to advancing automated grading in real-world academic settings.

Keywords: Large Language Model (LLM); automated grading; GPT; educational assessment; open-source toolkit

1. Introduction

Recent advances in Large Language Models (LLMs) have opened new possibilities for their application in educational contexts, including automated tutoring, feedback generation, and grading [1–3]. Prior studies have shown that LLMs can reduce educators' workloads by generating personalized materials and assessments, while emphasizing the importance of human–AI collaboration guided by instructors [4]. Automated grading systems, in particular, offer the potential for increased efficiency and scalability. However, their practical reliability and pedagogical value in real classrooms remain underexplored. Most prior works rely on controlled or synthetic settings, leaving open the question of how LLM-based graders perform on authentic student responses collected over a full academic semester.

In large courses, educators often handle a heavy grading load, especially for open-ended short-answer questions that require more than a simple right-or-wrong judgment. Unlike multiple-choice questions, short-answer responses require evaluators to assess conceptual understanding, tolerate diverse phrasings of the same idea, and provide meaningful feedback. Project reports present an even greater challenge, requiring comprehensive

Check for updates

Academic Editors: Joao Carlos Lopes Batista and Dimitris Apostolou

Received: 22 January 2026
Revised: 8 May 2026
Accepted: 8 May 2026
Published: 20 May 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

assessment across multiple dimensions such as technical soundness, analytical depth, and written clarity.

Domains such as computational linguistics are particularly well-suited for evaluating LLM-based grading, as they combine technical and linguistic reasoning. Students are expected to reason about language formally while also articulating their understanding in natural language. This combination of factual and analytical reasoning makes automated evaluation particularly challenging, as surface-level matching is insufficient and semantic understanding is required.

In this preliminary study, we investigate the use of GPT-4o [5] to automatically evaluate short-answer quizzes and final project reports in an undergraduate Computational Linguistics course. We also release an open-source auto-grading toolkit to support reproducibility and further research. Specifically, we address the following research questions:

- RQ1: How well do LLM-generated grades align with human evaluations?
- RQ2: What are the most common reasons for disagreement between LLM and human graders?
- RQ3: Can an open-source grading toolkit be developed to support LLM-based assessment in real-world educational settings?

To answer these questions, we collect responses from approximately 50 students across five quizzes and team project reports from a real undergraduate course. LLM outputs are compared with grades assigned independently by two human teaching assistants (TAs). We also introduce and publicly release LLM-as-a-TA, an open-source grading toolkit, along with all code and evaluation protocols, to encourage the adoption of LLM-based evaluation tools. Our findings offer insights into the strengths and limitations of using LLMs for academic evaluation and highlight considerations for their deployment in real-world classrooms. The remainder of this paper presents our dataset (Section 3), methodology (Section 4), and experimental results (Section 5), followed by a discussion of findings, limitations, and directions for future work (Section 7).

2. Related Work

Prior studies have shown that LLMs can approximate human grading performance on a variety of academic tasks. LLM-generated scores correlate strongly with instructors' grades and fall within the range of normal inter-grader variability [6,7]. For example, ChatGPT-3.5 was able to match university instructors' exam scores within a 5–10% margin in around 70% of cases [7]. With proper prompting and rubric design, LLMs have also shown promise not only as scorers but as feedback providers—several systems use AI to generate step-by-step explanations alongside grades [8–11], or to rephrase incorrect student responses into corrected ones [12]. Our study builds on this direction, evaluating GPT-4o not only as a grader but as a source of formative feedback for students.

Recent work has also begun to examine LLM grading behavior across disciplines and model types. Grévisse [13] evaluated GPT-4 and Gemini on student responses from 12 undergraduate medical courses, finding that GPT-4 tended to assign lower grades than human evaluators—a conservative bias we also observe in our results. Poličar et al. [14] conducted a blind evaluation of LLMs in a real bioinformatics course, showing that well-prompted models can match human TA performance in scoring accuracy and feedback quality, with open-source models performing comparably to commercial ones. While most prior work has relied on GPT-3.5 or GPT-4 [2,11], we use GPT-4o throughout this study for its faster response times and stronger performance across academic and reasoning benchmarks.

3. Dataset

3.1. Quiz Dataset

We collect short-answer quiz responses from around 50 undergraduates enrolled in a Computational Linguistics course. A total of five quizzes are collected over the course of the 4-month semester. Each quiz includes 10–16 open-ended questions designed to assess students' understanding of key concepts. Unlike multiple-choice questions (MCQ) format, these open-ended questions require students to articulate their understanding in their own words. For each question, a gold-standard answer is provided by the course instructor to serve as the reference for evaluation. The questions are written at the undergraduate level, covering topics such as *n*-gram language models, vector space models, and basic parsing algorithms. The topics covered in each quiz are listed below. Full quiz questions are in Appendix A.1.

Each question has a maximum score of 0.2 points, with scores assigned in 0.1-point increments. Since students can phrase their responses differently, grading focuses on conceptual alignment with the gold answer rather than exact wording. If a student's answer is relevant and included at least one or two key ideas from the gold answer, they receive full points, even if some details are missing. Clearly irrelevant or empty responses receive a score of zero. This grading policy requires evaluators to assess conceptual understanding rather than surface-level correctness. All graders (human or LLM) must interpret diverse expressions of the same concept and go beyond lexical matching to verify semantic alignment. This makes the task significantly more complex than evaluating fixed-format or MCQ, where correctness is binary and easier to automate.

- Quiz 1: Text Processing
  Covers basic string manipulation, tokenization, normalization (lowercasing, stemming, lemmatization), and regular expressions.
- Quiz 2: Language Models
  Focuses on *n*-gram language models, smoothing techniques (e.g., Laplace), and evaluation metrics such as perplexity.
- Quiz 3: Vector Space Models
  Includes term weighting (TF-IDF), cosine similarity, and document classification using vector representations.
- Quiz 4: Distributional Semantics
  Tests understanding of co-occurrence matrices, word-context windows, dimensionality reduction, and distributional hypothesis.
- Quiz 5: Contextual Encoding
  Addresses contextual word representations using pre-trained models such as BERT, including their architectural differences and contextualization mechanisms.

3.2. Project Report Dataset

In addition to quizzes, we also evaluate project reports. These are team-based assignments in which students design and propose an NLP project aimed at addressing a real-world problem. Each student team submits one report, resulting in a total of 14 submissions. The final deliverable is a standardized 5–8 page document that follows a consistent structure comprising sections such as abstract, motivation, problem statement, related work, technical approach, results, and conclusion. This uniform format, together with a detailed grading rubric provided by the course instructor, enables consistent extraction and evaluation of key components across all reports. The following list provides examples of a few report topics submitted by the teams.

- CAAP (Capture Assistant in Academic Papers): An LLM-powered tool that extracts keyphrases and definitions from academic papers, combining textual and visual information to enhance comprehension of technical content.
- GitFolio: A web-based platform that converts static resumes into dynamic online portfolios using NLP, OCR, and chatbot assistance, targeting early-career professionals with limited web development experience.
- FutureFetch: A personalized job/internship recommendation system that extracts information from resumes using GPT-4o and filters opportunities via a custom Python 3.9 backend, evaluated through both quantitative metrics and live user feedback.
- CareerAi: A resume enhancement and job-matching system using NLP, regular expressions, and Gemini API, designed for resume formatting and recommendation of the relevant job listings through web scraping and structured parsing.
- MarketGuardian: A real-time scam detection system for e-commerce listings that analyzes textual and visual cues using LLMs, reverse image search, and keyphrase extraction to flag potentially fraudulent eBay listings.
- ATAP (Application Tracking Automation Program): An NLP-based tool that automates email parsing and application tracking for students, aiming to reduce inequality in opportunity access through centralized, real-time updates and support for diverse application types.

4. Methodology

Prompting and Scoring Strategy

To evaluate student quiz responses, we use a Python-based grading script that interacts with the OpenAI GPT-4o API. If a student's response is empty, a score of 0 is assigned automatically. For all other cases, GPT-4o is prompted with the quiz question, the reference answer (i.e., gold answer), and the student response. The model is instructed to assign a score from a fixed set of valid values (e.g., [0.0, 0.1, . . . , 0.2]) based on how well the student's answer matches the key ideas in the reference. If the model assigns a score less than full credit, it also provides a short explanation for the deduction. All GPT-4o API calls are made using a temperature of 0 to ensure deterministic outputs. Each student response is evaluated in a single API call.

To evaluate project reports, our autograder extracts text from PDF files using the PyMuPDF [15] library and submits it to GPT-4o's API for grading. Each report is assessed using a fixed rubric with eight sections: Abstract, Introduction, Related Work, Approach, Results, Conclusion, References, and Format, totaling 9 points. Each section has a specified maximum point value, and partial credit is allowed. GPT-4o is prompted to score each section individually, and generate an overall summary with a brief score justification.

The overall grading process is shown in Figure 1. The prompts used for grading quiz and report is presented in Figures 2 and 3, respectively. Two PhD students in Computer Science serve as human graders. Student responses are divided between the two TAs, with each response graded by exactly one human grader. This protocol reflects typical TA workload distribution in real courses, though it precludes direct TA-TA agreement comparison.

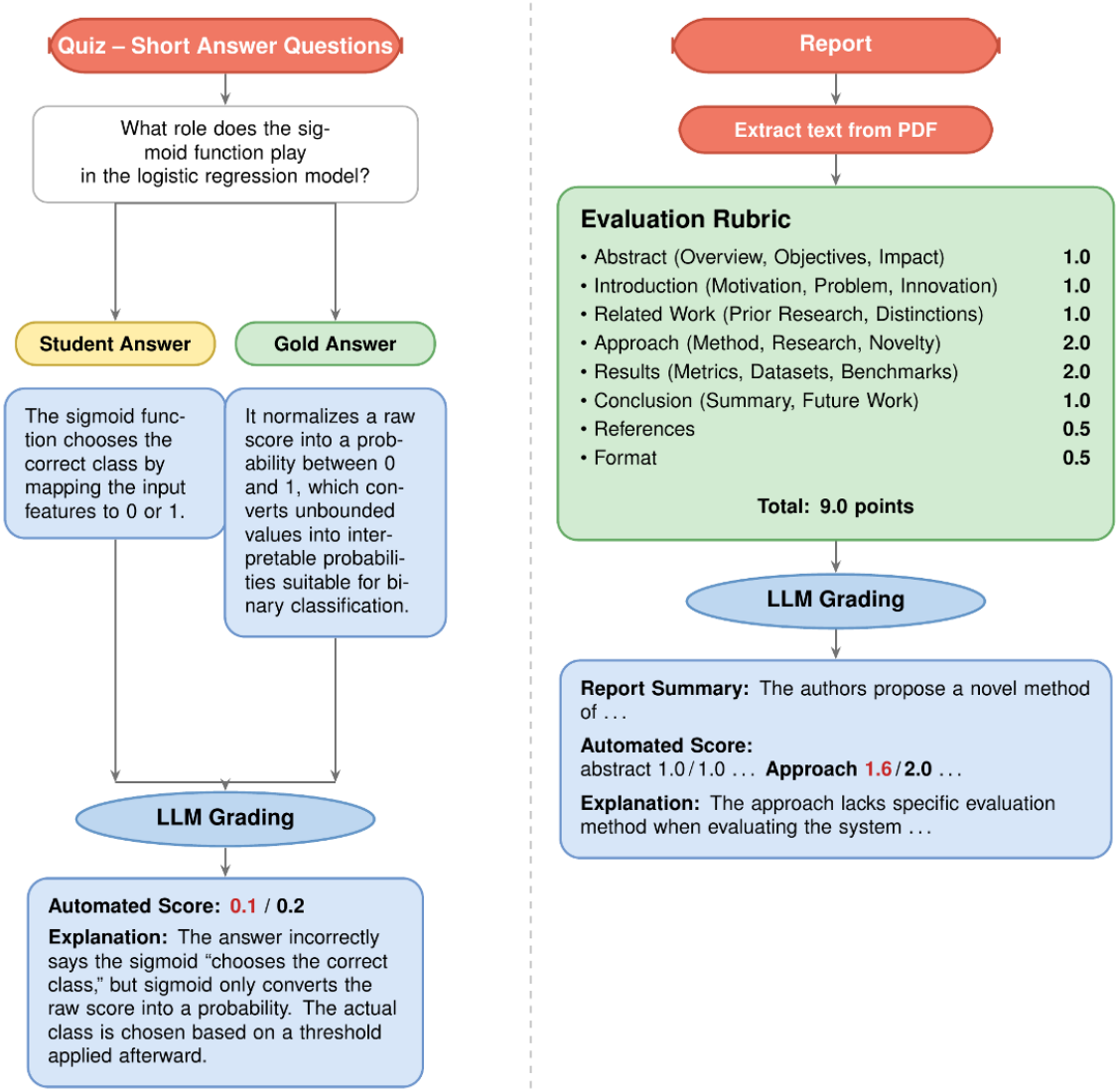


Figure 1. Our toolkit evaluates both short-answer quiz responses (**left**) and reports (**right**). For quizzes, a student answer is compared to a reference answer and scored based on correctness. For reports, text is extracted from PDF files and evaluated based on pre-defined rubric. Explanations are generated in both cases to justify the score.

Prompt

```
You are grading a student's answer to a quiz question. Follow these rules:

1. The maximum score for this question is {full_str}.
2. You can only choose a score from this set: {valid_str}.
3. Return the grade on a single line exactly in the format: ''Grade: X''
   where X is one of {valid_str}.
4. If you give a grade less than {full_str}, then on the next line provide a short single-sentence explanation starting with
   ''Explanation:''.
5. If you give a grade equal to {full_str}, do not provide any explanation or additional lines. End your response immediately
   after ''Grade: {full_str}''.
6. If the student's answer is relevant to the question and includes at least one or two key ideas from the gold answer, give
   the maximum score.
   Even if some details are missing, as long as the student shows general understanding, give the full points.

Example:
Question: All delimiters used in our implementation are punctuation marks. What types of tokens should not be split by such
   delimiters?
Gold Answer: URLs, email addresses, decimal numbers, and abbreviations
Student Answer: email and decimal numbers.
Grade: 0.2

7. If the student's answer is empty or obviously irrelevant, it cannot capture the essential idea, so you must give 0.

Question: {question}
Gold Answer: {gold_answer}
Student Answer: {student_answer}
Grade:
```

Figure 2. Prompt used to grade quiz responses.

Prompt

```
You are a teaching assistant evaluating project reports.
The report should be graded out of 9 points using the detailed breakdown below.
Provide a score for each criterion based on the content, and~give a short justification (1-2 sentences) for each~score.

Use the following rubric:

1. **Abstract (1 point)**
- Is the abstract concise and informative?
- Does it clearly summarize the project's objectives, intellectual merit, and~broader societal impact?
- Vague or general abstracts should receive significant~deductions.

2. **Introduction (1 point)**
- Are the project goals and motivation stated clearly?
- Is the problem defined in a way that demonstrates originality and relevance?
- Deduct points if the introduction lacks context, fails to specify what is being addressed, or~overstates~contributions.

3. **Related Work (1 point)**
- Does the report review relevant prior research?
- Are key papers cited and discussed critically?
- Deduct points if the section is too brief, lacks citations, or~fails to distinguish the proposed work from prior~work.

4. **Approach (2 points)**
- Is the proposed method clearly explained and technically sound?
- Are steps, models, or~algorithms described with adequate detail?
- Deduct points for vagueness, overgeneralizations, or~missing methodological details.
- If novelty is claimed, is it justified?

5. **Results (2 points)**
- Are preliminary results presented, or~is there a clear evaluation plan?
- Are metrics, datasets, or~benchmarks described properly?
- Deduct points if results are missing, vague, or~unsupported.

6. **Conclusion (1 point)**
- Does the conclusion effectively summarize findings and future work?
- Deduct points if it simply repeats earlier sections or lacks~insight.

7. **References (0.5 points)**
- Are references complete, recent, and~properly formatted?
- Deduct for missing key works or inconsistent~formatting.

8. **Format (0.5 points)**
- Is the report well-organized and easy to read?
- Deduct points for poor formatting, grammar issues, or~unclear~diagrams.

Instructions:
- For each category, assign a score from 0 to the max points listed.
- Be critical.
- Give a short justification (2-3 sentences).
- Then, sum the scores and give a final total (out of 9 points).
- If a section is missing or weak, explain it~directly.

- Format your response EXACTLY like this (including ' - ' for comments):

Abstract: 1 - [comment]
Introduction: 0.7 - [comment]
Related Work: 1 - [comment]
Approach: 1.6 - [comment]
Results: 1.8 - [comment]
Conclusion: 1 - [comment]
References: 0.5 - [comment]
Format: 0.5 - [comment]
Total: 8.1


Summary: For any category where you deducted points, explain **clearly and specifically**:
- What was missing or unclear
- Why that issue is important
- Give concrete examples or quote from the report if~possible

Finally, provide a brief summary (3-4 lines) of the report's main idea and content, exactly in this format:
ReportSummary: [your 3-4 line summary here]


Here is the report for evaluation:

{report_text}
```

Figure 3. Prompt used to grade project reports. Note: ** denotes bold text in the prompt formatting instructions.

5. Results

5.1. Grading Comparison on Quizzes

We conduct a statistical comparison of scores assigned by LLM and a human grader across five quizzes. For each quiz, we examine the number of graded submissions, the mean scores assigned by human graders, the mean absolute difference between scores, Wilcoxon signed-rank test results, and Pearson correlations. We use the Wilcoxon signed-rank test instead of a paired t-test due to the discrete and bounded nature of the scoring scale. Table 1 compares GPT-generated grades with manual grades. The average difference between the two sets of grades is small, ranging from 0.03 to 0.12 points.

For four out of five assignments (Quiz 1–4), the differences are statistically significant ($p < 0.05$), indicating that the observed differences are unlikely to be due to random variation and instead reflect a systematic scoring gap between GPT and the human grader.

Quiz 5 showed no significant difference ($p = 0.394$), suggesting that GPT and manual grades are very similar for that assignment. Despite these mean differences, the correlation between GPT and manual grades remains strong across all quizzes. Correlation values range from 0.62 to 0.97, with all correlations statistically significant. Overall, the correlation is 0.98, indicating that GPT grading closely aligns with human grading in ranking student performance.

Table 1. GPT vs. manual grading statistics across five quizzes. Reported per assignment: grader means, mean absolute difference, Pearson correlation, and Wilcoxon signed-rank test results (non-parametric test used given discrete score distributions). Significance threshold: $p < 0.05$.

Quiz	n	GPT Mean	Manual Mean	Mean Abs Diff	Wilcoxon W	p-Value	Corr	Corr p-Value
1. Text Processing	53	1.85	1.93	0.12	121	1.37×10^{-3}	0.62	7.66×10^{-7}
2. Language Models	49	2.08	2.18	0.10	0	1.41×10^{-7}	0.94	6.20×10^{-24}
3. Vector Space Models	52	1.71	1.74	0.03	0	1.73×10^{-3}	0.97	1.66×10^{-31}
4. Distributional Semantics	52	2.94	3.00	0.06	7.5	6.59×10^{-5}	0.88	9.07×10^{-18}
5. Contextual Encoding	52	2.69	2.70	0.03	23.5	3.94×10^{-1}	0.92	1.47×10^{-22}
Overall	258	2.26	2.31	0.07	710	8.76×10^{-14}	0.98	2.34×10^{-186}

To further assess the correspondence between GPT-generated scores and human evaluations, we categorize each student score based on how closely GPT matched the manual grade. Of all the cases ($n = 258$), we find that GPT's score exactly matched the human-assigned score in 55% of cases, while 38.8% were under-graded and 6.2% over-graded. To contextualize the 55% exact agreement, we compute expected chance agreement from the marginal score distributions. The chance-level agreement is 7.4%, resulting in a Cohen's kappa of 0.515, indicating moderate agreement beyond chance and suggesting that the observed agreement is not driven by score distribution alone. Although the overall Pearson correlation of 0.98 indicates that GPT-4o closely tracks the rank ordering of human grades, it does not imply score-level agreement. The asymmetric distribution of disagreements reveals a systematic conservative bias that correlation alone would not capture.

5.2. Grading Comparison on Project Reports

We compare section-wise scores assigned independently by GPT and a human TA to each project report. Table 2 reports the average scores and results of the Wilcoxon signed-rank test across the rubric sections. Across most sections, LLM evaluations closely match those of the human. No statistically significant differences are observed in Introduction, Related Work, or Format ($p > 0.05$). In the Abstract, Conclusion, and References sections, both graders assigned maximum scores to all submissions, resulting in trivially identical scores. However, GPT-4o assigns significantly lower scores than human evaluators in two sections: Approach ($p = 0.0083$) and Results ($p = 0.0020$). To control for Type I error from multiple comparisons, we apply the Holm–Bonferroni correction. After adjustment, the Approach ($p_{\text{adj}} = 0.0332$) and Results ($p_{\text{adj}} = 0.0102$) sections remain statistically significant. Overall, GPT-4o performs comparably to human grading, though it may be more conservative on technical or empirical sections.

To understand evaluation behavior differences, we categorize deduction instances from both GPT-4o (13 cases) and human (20 cases) during report grading. Table 3 shows the distribution of deduction reasons. GPT prioritizes empirical rigor, with insufficient quantitative results being the most frequent deduction (30.8%) compared to human (15%). Both similarly emphasize related work quality (GPT: 23.1%, human TA: 25%), penalizing literature reviews without critical analysis. However, while PhD student TAs dedicate attention to formatting and presentation (25% vs. GPT's 7.7%) and separately assess writing

quality (10%) and conclusion adequacy (10%). These categories are absent from GPT's deductions. Conversely, GPT uniquely penalizes missing limitations discussion (15.4%). These patterns reveal complementary evaluation approaches: GPT emphasizes analytical depth and empirical evidence, while human applies more holistic criteria including academic presentation standards. Tables 4 and 5 present the real examples of the GPT and human evaluation. The scores and the reason for the deduction are almost the same.

Table 2. Report Grading: Section-wise comparison of LLM and human grading using the Wilcoxon signed-rank test. Holm–Bonferroni adjusted *p*-values are shown to account for multiple comparisons. GPT-4o's section-level scores show strong alignment with those of the human grader.

Section	GPT Mean	TA Mean	Mean Diff	Holm *p*-Value	Significant (Holm)
Abstract	1.000	1.000	0.000	N/A	All scores identical
Introduction	0.986	0.979	0.007	1.0000	False
Related Work	0.943	0.943	0.000	1.0000	False
Approach	1.886	1.986	−0.100	0.0332	True
Results	1.750	1.957	−0.207	0.0102	True
Conclusion	1.000	1.000	0.000	N/A	All scores identical
References	0.500	0.500	0.000	N/A	All scores identical
Format	0.500	0.486	0.014	0.4719	False

Table 3. Distribution of deduction reasons across GPT-4o and human TA evaluations during project report grading. Each entry shows the raw count and percentage out of the total deduction instances per grader (GPT: n = 13; Human TA: n = 20). A dash (–) indicates that the grader did not apply deductions under that category.

Deduction Reason Category	GPT (n = 13)	Human TA (n = 20)
Insufficient quantitative results	4 (30.8%)	3 (15.0%)
Superficial Related Work	3 (23.1%)	5 (25.0%)
Missing limitations discussion	2 (15.4%)	–
Formatting issue	1 (7.7%)	5 (25.0%)
Weak novelty justification	1 (7.7%)	2 (10.0%)
Lack of detail in methods	1 (7.7%)	1 (5.0%)
Weak introduction or motivation	1 (7.7%)	–
Writing quality / clarity issue	–	2 (10.0%)
Missing/inadequate conclusion	–	2 (10.0%)

Table 4. Section-wise Comparison of GPT and Human Grades with Feedback for Team XYZ Report. The grades and the reason for deduction is identical.

Section	GPT	TA	GPT Feedback	TA Feedback
Abstract	1.0	1.0	The abstract is concise and informative, clearly summarizing the project's objectives, intellectual merit, and broader societal impact.	
Introduction	1.0	1.0	The introduction clearly states the project goals and motivation, providing a well-defined problem statement.	
Related Work	1.0	1.0	Reviews relevant prior research and distinguishes the proposed work from existing solutions.	
Approach	2.0	2.0	Clearly explained and technically sound; detailed descriptions of steps and justified novelty.	

Table 4. Cont.

Section	GPT	TA	GPT Feedback	TA Feedback
Results	1.8	1.8	Evaluation metrics and benchmarks are described properly, but more quantitative data would help.	Section 5 should be improved. For instance, the report states that "ADAPT was evaluated using internal benchmarks," but no details are provided about these benchmarks or any results tables for the reader to reference. Including more detailed preliminary results would strengthen the section. (−0.2)
Conclusion	1.0	1.0	Summarizes findings and future work effectively, reflecting on project impact.	
References	0.5	0.5	References are complete, recent, and properly formatted.	
Format	0.5	0.5	The report is well-organized and easy to read.	
GPT Summary	-	-	ADAPT is an AI-powered platform designed to enhance email composition via large language models. It introduces a user-centric interface for context-aware revisions, integrating with email services for seamless communication. The project aims to improve accessibility and efficiency in digital communication, offering a novel approach to human-AI co-writing.	-

Table 5. Section-wise comparison of GPT and human grades with feedback for Team ABC's project report.

Section	GPT	TA	GPT Feedback	TA Feedback
Abstract	1.0	1.0	The abstract is concise and informative, clearly summarizing the project's objectives, intellectual merit, and societal impact.	
Introduction	1.0	1.0	Clearly states the project goals and motivation with originality and well-defined context.	
Related Work	1.0	1.0	Reviews prior research and distinguishes the proposed work from existing solutions effectively.	
Approach	1.8	2.0	The proposed method is clearly explained and technically sound, with detailed descriptions of steps, models, and algorithms. However, the novelty claim could be more explicitly justified with specific examples of how ATAP's approach differs from existing methods.	
Results	1.8	1.8	Preliminary results are presented with a clear evaluation plan, and metrics, datasets, and benchmarks are described properly. However, the discussion could benefit from more detailed analysis of the results and their implications.	The results show that the system achieved an overall precision of 92.31%, a perfect recall of 100%, and an F1 score of 96%. An error analysis (i.e., cases where system predictions were wrong) would have helped not only readers but also the team to improve. For instance, I see they tried to interpret the results, but including some examples (failed prediction cases) from the actual evaluation set would have been much more helpful.
Conclusion	1.0	1.0	Summarizes findings and reflects on impact effectively.	
References	0.5	0.5	References are complete, recent, and properly formatted.	

Table 5. Cont.

Section	GPT	TA	GPT Feedback	TA Feedback
Format	0.5	0.5	Well-organized and easy to read, with proper formatting.	
GPT Summary	-	-	ATAP is an innovative application tracking tool that automates email retrieval, content classification, and status updates. It addresses inequities and improves usability for students.	-

6. Discussion

We develop and release an open-source grading toolkit [16], which allows flexible configuration of model selection, number of questions, granularity, and maximum scores. Grading can be performed directly from PDFs, making the system practical for real course deployment. Our results show that GPT-4o achieves strong overall alignment with human graders, though with a consistent conservative bias—particularly in technical sections such as Approach and Results.

This bias may stem from GPT-4o's tendency to penalize missing quantitative evidence even when the overall argument is sound. The deduction pattern in Table 3 supports this: GPT most frequently deducted for insufficient quantitative results (30.8%), while TAs were more likely to flag formatting and writing quality issues. These complementary tendencies suggest a natural hybrid workflow, where LLM-generated grades serve as a first pass and human review focuses on the dimensions where automated evaluation is weakest.

Finally, even small systematic gaps between LLM and human grades could affect how students perceive fairness and whether they trust automated feedback. In practice, transparent communication about how grades are generated (combined with instructor review of borderline cases) would be important for responsible deployment.

7. Limitations and Future Work

This study has several limitations. First, the dataset is relatively small, collected from a single course at one institution, and our findings should be interpreted as preliminary and context-specific. Replication across larger cohorts, different disciplines, and multiple institutions will be necessary before broader conclusions can be drawn. Second, because the two TAs divide student responses between themselves rather than grading the same responses independently, we are unable to report a human-human inter-rater agreement baseline. This makes it harder to fully contextualize GPT-4o's alignment with human grading, and future work should incorporate overlapping annotation to address this gap. Third, although all GPT-4o API calls are made with temperature set to 0, we do not conduct repeated runs to assess output consistency, and future work should include stability analyses to strengthen reproducibility claims. Lastly, this work evaluates only GPT-4o; extending the comparison to other proprietary and open-source models would provide a more complete picture of the LLM-based grading.

In our future work, we plan to evaluate the toolkit on larger and more diverse student populations, and to explore how prompt design—such as rubric structure and the inclusion of reasoning examples—affects grading consistency across domains. While the toolkit is built for computational linguistics, the modular rubric design makes it straightforward to adapt for other courses through prompt modifications, and we intend to test this in future work.

8. Conclusions

In this study, we explore the use of LLMs to support grading in real-world classroom settings by applying them to an undergraduate computational linguistics course. Our

approach is effective for both short-answer quizzes and report assessments, demonstrating strong alignment with human grading across both task types. To support further research and practical adoption, we release our sample dataset and open-source the grading toolkit, LLM-as-a-TA. We aim to provide a practical foundation for future applications of LLMs in automated grading across educational contexts.

Author Contributions: Conceptualization, G.B., S.R. and J.D.C.; Methodology, G.B.; Software, G.B.; Formal analysis, G.B. and S.R.; Investigation, G.B. and S.R.; Data curation, G.B.; Writing—original draft preparation, G.B. and S.R.; Writing—review and editing, G.B., S.R. and J.D.C.; Visualization, G.B.; Supervision, J.D.C.; Project administration, J.D.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: The study exclusively involved secondary analysis of anonymized educational data collected during normal classroom activities, so it does not meet the criteria for human subjects research requiring Institutional Review Board approval under applicable institutional and federal guidelines. Therefore, formal Ethics Committee or Institutional Review Board approval was not required.

Informed Consent Statement: Formal written informed consent was not obtained, as the study involved the analysis of anonymized student responses collected as part of regular instructional activities. Students were informed that anonymized course data may be used for research purposes. No personally identifiable information was retained. Any examples included in this paper are either anonymized excerpts or synthetic illustrative samples generated by a large language model.

Data Availability Statement: No new publicly available datasets were generated or analyzed during this study. The code and sample data used in the experiments are available at https://github.com/emorynlp/LLM-Grading (accessed on 10 July 2025).

Acknowledgments: The authors would like to thank the reviewers and the editor of Information for their valuable time and constructive feedback that improved the quality of this manuscript.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

LLM	Large Language Model
TA	Teaching Assistant
NLP	Natural Language Processing
MCQ	Multiple-Choice Question
TF-IDF	Term Frequency-Inverse Document Frequency
BERT	Bidirectional Encoder Representations from Transformers

Appendix A. Short-Answer Quiz

Appendix A.1. Quiz Questions

Quiz 1: Text Processing

1. What is the difference between a word token and a word type?
2. How can we interpret the most frequent words in a text?
3. What is the difference between a word and a token?
4. All delimiters used in our implementation are punctuation marks. What types of tokens should not be split by such delimiters?
5. Our tokenizer uses hard-coded rules to handle specific cases. What would be a scalable approach to handling more diverse cases?

6. The use of a more advanced tokenizer mitigates the issue of sparsity. What exactly is the sparsity issue, and how can appropriate tokenization help alleviate it?
7. What is the difference between a lemmatizer and a stemmer?
8. What are the key differences between inflectional and derivational morphology?
9. In which tasks can lemmatization negatively impact performance?
10. What are the benefits and limitations of using regular expressions for tokenization vs. the rule-based tokenization approach discussed in the previous section?

Quiz 2: Language Models

1. What are the advantages of splitting "I" and "m" as two separate tokens, versus recognizing "I'm" as one token?
2. What advantages do unigram probabilities have over word frequencies?
3. What NLP tasks can benefit from bigram estimation over unigram estimation?
4. When applying Laplace smoothing, do unigram probabilities always decrease? If not, what conditions can cause a unigram's probability to increase?
5. What does the Laplace smoothed bigram probability of (w_{u-1}, w_u) represent when w_{u-1} is unknown, and what is a potential problem with this estimation?
6. Why is it problematic when bigram probabilities following a given word don't sum to 1?
7. What are the key differences between conditional and joint probabilities in sequence modeling, and how are they practically applied?
8. How do the Chain Rule and Markov Assumption simplify the estimation of sequence probability?
9. Is it worth considering the end of the text by introducing another artificial token, w_{n+1}, to improve last-word prediction by multiplying the above product with $P(w_{n+1}|w_n)$?
10. Why is logarithmic scale used to measure self-information in entropy calculations?
11. What indicates high entropy in a text corpus?
12. What is the relationship between corpus entropy and language model perplexity?

Quiz 3: Vector Space Models

1. One limitation of the bag-of-words model is its inability to handle unknown words. Is there a method to enhance the bag-of-words model, allowing it to handle unknown words?
2. Another limitation of the bag-of-words model is its inability to capture word order. Is there a method to enhance the bag-of-words model, allowing it to preserve the word order?
3. If term frequency does not necessarily indicate semantic importance, what kind of significance does it convey?
4. Stop words can be filtered either during the creation of vocabulary dictionary or when generating the bag-of-words representations. Which approach is preferable and why?
5. What are the implications when a term has a high document frequency?
6. Why do both Euclidean Distance and Cosine Similarity metrics consider that D1 is more similar to D2 than to D3?
7. Why is Cosine Similarity generally preferred over Euclidean Distance in most NLP applications?
8. What potential problems might arise from the above data splitting approach, and what alternative method could mitigate these issues?
9. Why do we use only the training set to collect the vocabulary?
10. What are the primary weaknesses and limitations of the K-Nearest Neighbors (KNN) classification model when applied to document classification?

https://doi.org/10.3390/info17050505

Quiz 4: Distributional Semantics

1. Assuming that your corpus has only the following three sentences, what context would influence the meaning of the word "chair" according to the distributional hypothesis?
    1. I sat on a chair.
    2. I will chair the meeting.
    3. I am the chair of my department.
2. What are the drawbacks of using one-hot encoding to represent word vectors?
3. Why is the performance of `document_term_matrix()` significantly slower than `document_term_matrix_np()`?
4. What is the maximum number of topics that LSA can identify? What are the limitations associated with discovering topics using this approach?
5. By discarding the first topic, you can observe document embeddings that are opposite (e.g., documents 4 and 5). What are the characteristics of these documents that are opposite to each other?
6. Σ is not transposed in L3 of the above code. Should we use `S.transpose()` instead?
7. What role does the sigmoid function play in the logistic regression model?
8. Under what circumstances would the bias b be negative in the above example? Additionally, when might neutral terms such as "this" or "movie" exhibit non-neutral weights?
9. What is the role of the softmax function in the softmax regression model? How does it differ from the sigmoid function?
10. Notice that the above equation for MLP does not include bias terms. How are biases handled in light of this formulation?
11. What would be the weight assigned to the feature "truly" learned by softmax regression for the above example?
12. What are the limitations of a multilayer perceptron?
13. What are the advantages of using discriminative models like CBOW for constructing language models compared to generative models like n-gram models?
14. What are the advantages of CBOW models compared to Skip-gram models, and vice versa?
15. What are the implications of the weight matrices W_x and W_h in the Skip-gram model?
16. What limitations does the Word2Vec model have, and how can these limitations be addressed?

Quiz 5: Contextual Encoding

1. How can document-level vector representations be derived from Word2Vec word embeddings?
2. How did the embedding representation facilitate the adaption of Neural Networks in Natural Language Processing?
3. How are embedding representations for Natural Language Processing fundamentally different from ones for Computer Vision?
4. The EM algorithm stands as a classic method in unsupervised learning. What are the advantages of unsupervised learning over supervised learning, and which tasks align well with unsupervised learning?
5. What are the disadvantages of using BPE-based tokenization instead of rule-based tokenization? What are the potential issues with the implementation of BPE above?
6. How does each hidden state h_i in a RNN encode information relevant to sequence tagging tasks?
7. In text classification tasks, what specific information is captured by the final hidden state h_n of a RNN?

https://doi.org/10.3390/info17050505

8. What are the advantages and limitations of implementing bidirectional RNNs for text classification and sequence tagging tasks?
9. How does self-attention operate given an embedding matrix $W \in \mathbb{R}^{n \times d}$ representing a document, where n is the number of words and d is the embedding dimension?
10. Given an embedding matrix $W \in \mathbb{R}^{n \times d}$ representing a document, how does multi-head attention function? What advantages does multi-head attention offer over self-attention?
11. What are the outputs of each layer in the Transformer model? How do the embeddings learned in the upper layers of the Transformer differ from those in the lower layers?
12. How is a Masked Language Model used in training a language model with a transformer?
13. How can one train a document-level embedding using a transformer?
14. What are the advantages of embeddings generated by BERT compared to those generated by Word2Vec?

References

1. Maiti, P.; Goel, A.K. How Do Students Interact with an LLM-powered Virtual Teaching Assistant in Different Educational Settings? *arXiv* **2024**, arXiv.2407.17429. [CrossRef]
2. Chiang, C.H.; Chen, W.C.; Kuan, C.Y.; Yang, C.; Lee, H.Y. Large Language Model as an Assignment Evaluator: Insights, Feedback, and Challenges in a 1000+ Student Course. *arXiv* **2024**, arXiv:2407.05216. [CrossRef]
3. Chu, Y.; He, P.; Li, H.; Han, H.; Yang, K.; Xue, Y.; Li, T.; Krajcik, J.; Tang, J. Enhancing LLM-Based Short Answer Grading with Retrieval-Augmented Generation. *arXiv* **2025**, arXiv.2504.05276. [CrossRef]
4. Liu, J.; Jiang, B.; Wei, Y. LLMs as Promising Personalized Teaching Assistants: How Do They Ease Teaching Work? *ECNU Rev. Educ.* **2025**, *8*, 343–348. [CrossRef]
5. OpenAI. GPT-4o System Card. *arXiv* **2024**, arXiv:2410.21276. [CrossRef]
6. Jukiewicz, M. The Future of Grading Programming Assignments in Education: The Role of ChatGPT in Automating the Assessment and Feedback Process. *Think. Ski. Creat.* **2024**, *52*, 101522. [CrossRef]
7. Flodén, J. Grading exams using large language models: A comparison between human and AI grading of exams in higher education using ChatGPT. *Br. Educ. Res. J.* **2024**, *51*, 201–224. [CrossRef]
8. Xie, W.; Niu, J.; Xue, C.J.; Guan, N. Grade Like a Human: Rethinking Automated Assessment with Large Language Models. *arXiv* **2024**, arXiv:2405.19694. [CrossRef]
9. Yeung, C.; Yu, J.; Cheung, K.C.; Wong, T.W.; Chan, C.M.; Wong, K.C.; Fujii, K. A Zero-Shot LLM Framework for Automatic Assignment Grading in Higher Education. *arXiv* **2025**, arXiv:2501.14305. [CrossRef]
10. Miroyan, M.; Mitra, C.; Jain, R.; Ranade, G.; Norouzi, N. Analyzing Pedagogical Quality and Efficiency of LLM Responses with TA Feedback to Live Student Questions. In Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1, Pittsburgh, PA, USA, 26 February–1 March 2025; Association for Computing Machinery: New York, NY, USA, 2025; pp. 770–776. [CrossRef]
11. Golchin, S.; Garuda, N.; Impey, C.; Wenger, M. Grading Massive Open Online Courses Using Large Language Models. *arXiv* **2024**, arXiv:2406.11102. [CrossRef]
12. Lin, J.; Han, Z.; Thomas, D.R.; Gurung, A.; Gupta, S.; Aleven, V.; Koedinger, K.R. How Can I Get It Right? Using GPT to Rephrase Incorrect Trainee Responses. *arXiv* **2024**, arXiv:2405.00970. [CrossRef]
13. Grévisse, C. LLM-based automatic short answer grading in undergraduate medical education. *BMC Med. Educ.* **2024**, *24*, 1060. [CrossRef] [PubMed]
14. Poličar, P.G.; Špendl, M.; Curk, T.; Zupan, B. Automated Assignment Grading with Large Language Models: Insights From a Bioinformatics Course. *arXiv* **2025**, arXiv:2501.14499. [CrossRef] [PubMed]
15. Inc., A.S. PyMuPDF: Python Bindings for MuPDF. Available online: https://github.com/pymupdf/PyMuPDF (accessed on 9 July 2025).
16. EmoryNLP. LLM-grading: Automated Grading Framework Using LLMs (GitHub Repository). 2025. Available online: https://github.com/emorynlp/LLM-Grading (accessed on 10 July 2025).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.