Qayyem: A Real-time Platform for Scoring Proficiency of Arabic Essays

Hoor Elbahnasawi, Marwan Sayed, Sohaila Eltanbouly, Fatima Brahamia, Tamer Elsayed
Computer Science and Engineering Department, Qatar University
{he2006903, me2104862, se1403101, fb2205644, telsayed}@qu.edu.qa

Abstract

Over the past years, Automated Essay Scoring (AES) systems have gained increasing attention as scalable and consistent solutions for assessing the proficiency of student writing. Despite recent progress, support for Arabic AES remains limited due to linguistic complexity and the scarcity of large publicly-available annotated datasets. In this work, we present *Qayyem*, a Web-based platform designed to support Arabic AES by providing an integrated workflow for assignment creation, batch essay upload, scoring configuration, and per-trait essay evaluation. *Qayyem* abstracts the technical complexity of interacting with scoring server APIs, allowing instructors to access advanced scoring services through a user-friendly interface. The platform deploys a number of state-of-the-art Arabic essay scoring models with different effectiveness and efficiency figures.

1 Introduction

Automated Essay Scoring (AES) systems provide a scalable solution that supports teachers in academic writing assessments. They typically adopt holistic scoring, which assigns a single overall quality score to the essay (Xie et al., 2022; Zhang et al., 2025), or trait-specific scoring, which evaluates it on distinct writing traits, such as grammar or organization (Kumar et al., 2022; Ormerod, 2022). Developing such systems typically follows one of two paradigms: prompt-specific and cross-prompt. *Prompt-specific* AES involves training and testing models on essays written for the same prompt, often achieving high performance due to the model's specialization (Taghipour and Ng, 2016; Dong et al., 2017). In contrast, *cross-prompt* AES models are tested on unseen prompts, enabling realistic and broader applicability, but presenting greater challenges due to increased topical variability (Ridley et al., 2021).

Despite the growing effort toward developing generalizable AES systems for English essays, research on Arabic AES remains limited, mainly due to the scarcity of large-scale Arabic datasets, which has constrained both methodological progress and system development. As a result, Arabic AES tools remain underexplored, particularly in terms of deployable software that supports realistic academic assessment workflows. Existing Arabic writing technologies primarily focus on writing assistance, such as Qalam,[1] rather than automated essay scoring. ARWI (Chirkunov et al., 2025) represents the only known deployed software that targets essay scoring; however, it operates in a prompt-specific setting and does not support multi-trait scoring.

In this work, we introduce *Qayyem*,[2] *the first* cross-prompt multi-trait Arabic AES platform designed for deployment in academic settings. *Qayyem* is delivered through a web-based interface and supports a comprehensive assessment workflow. It allows teachers to create writing assignments, define custom rubrics, upload student essays, select specific writing traits for evaluating them, then choose from a set of scoring models on a per-trait and/or per-assignment basis.

Qayyem supports both fully automated scoring and decision-support scenarios, allowing human oversight when required. It is designed to generalize to new assignments without retraining, addressing a key limitation of prompt-specific AES systems. The models provided within *Qayyem* are trained on LAILA, the largest available multi-trait Arabic AES corpus (Bashendy et al., 2026), enabling robust cross-prompt performance across diverse academic writing prompts.

Our contribution is four-fold: (1) we introduce *Qayyem*,[3] the first cross-prompt multi-trait Arabic AES online system, (2) we provide a Web-based interface supporting end-to-end assessment workflows for realistic educational settings, (3) we de-

[1] https://qalam.ai/en

[2] Pronounced in Arabic as "قَيِّم" which means "grade" as a verb and "valuable" as a noun in English.

[3] https://qayyem.qu.edu.qa/

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), pages 536–545

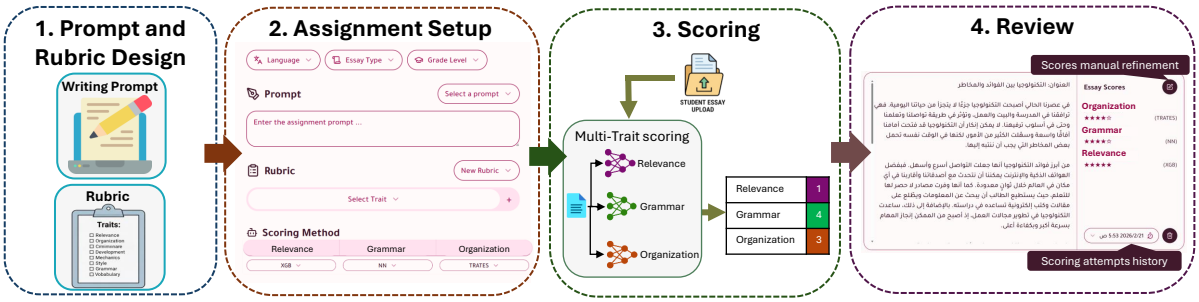

Figure 1: *Qayyem* workflow: (1) Prompt and Rubric Design, where the writing prompt and scoring rubric are defined; (2) Assignment Setup, where scoring traits and evaluation settings are configured; (3) Scoring, where essays are auto-evaluated across selected traits; and (4) Review, where results are inspected and optionally refined.

ploy a range of state-of-the-art (SOTA) Arabic AES models, and (4) we provide public API access to the deployed models, enabling developers and researchers to query them directly.

2 Related Work

Automated writing support systems assist learners by providing feedback or evaluative scores. This section reviews related work on Arabic AES and automated support tools for different languages.

2.1 Automated Essay Scoring for Arabic

Early Arabic AES relied on rule-based methods and feature engineering (Alqahtani and Alsaif, 2020; Alsanie et al., 2022), which were limited by manual feature design and struggled to generalize across prompts. Recently, AraBERT has become a popular backbone for Arabic AES, where fine-tuning achieved strong performance (Ghazawi and Simpson, 2024), enhancements with handcrafted features improved relevance assessment (Machhout and Zribi, 2024), and parameter-efficient tuning further optimized AraBERT-based AES (Mahmoud et al., 2024). In addition, Sayed et al. (2025) introduced 816 handcrafted features spanning surface, lexical, syntactic, readability, and semantic aspects, and used them to study cross-prompt AES. More recently, large language models (LLMs) like ChatGPT and LLaMA have been evaluated for Arabic under zero-shot, in-context, and fine-tuning settings (Ghazawi and Simpson, 2025). Alongside these advances, Bashendy et al. (2026) introduced LAILA, a large-scale dataset comprising 7,859 essays across eight prompts with holistic and trait-level annotations, establishing the first large-scale standardized benchmark for Arabic AES.

2.2 Writing Support Tools

Several automated writing support systems combine scoring with feedback. Criterion^{SM} (Burstein et al., 2003) is a commercial, cross-prompt AES system that produces holistic scores and feedback on grammar, usage, style, and discourse. Among non-commercial systems, eRevise+RF (Liu et al., 2025) is a prompt-specific writing evaluation system that provides formative feedback; however, its models and data are not publicly released. Essay-CBM (Chaudhary et al., 2025) uses a rubric-aligned concept bottleneck model to predict concept-level scores and aggregate them into a final grade, providing actionable feedback with human-in-the-loop support via an interactive web application.

For other languages, Hirao et al. (2020) introduced a publicly available prompt-specific system for non-native Japanese learners, generating both holistic and trait-specific scores with a BERT-based model. IFlyEA (Gong et al., 2021) is a cross-prompt Chinese AES system that analyzes grammar, rhetoric, and discourse analysis, providing explainable scoring, feedback, and visualizations.

For Arabic, ARWI (Chirkunov et al., 2025) is the first publicly available prompt-specific AES system, supporting grammar checks and CEFR-aligned scoring. While foundational for research, it lacks multi-trait scoring and cross-prompt generalization, limiting applicability in diverse educational settings. *Qayyem* addresses these limitations by providing cross-prompt models that support multi-trait scoring and generalization to unseen prompts.

3 System Workflow

This section presents the overall workflow of *Qayyem*, as depicted in Figure 1. The user first defines the writing prompts and rubrics (§3.1), which are incorporated into the assignment setup along

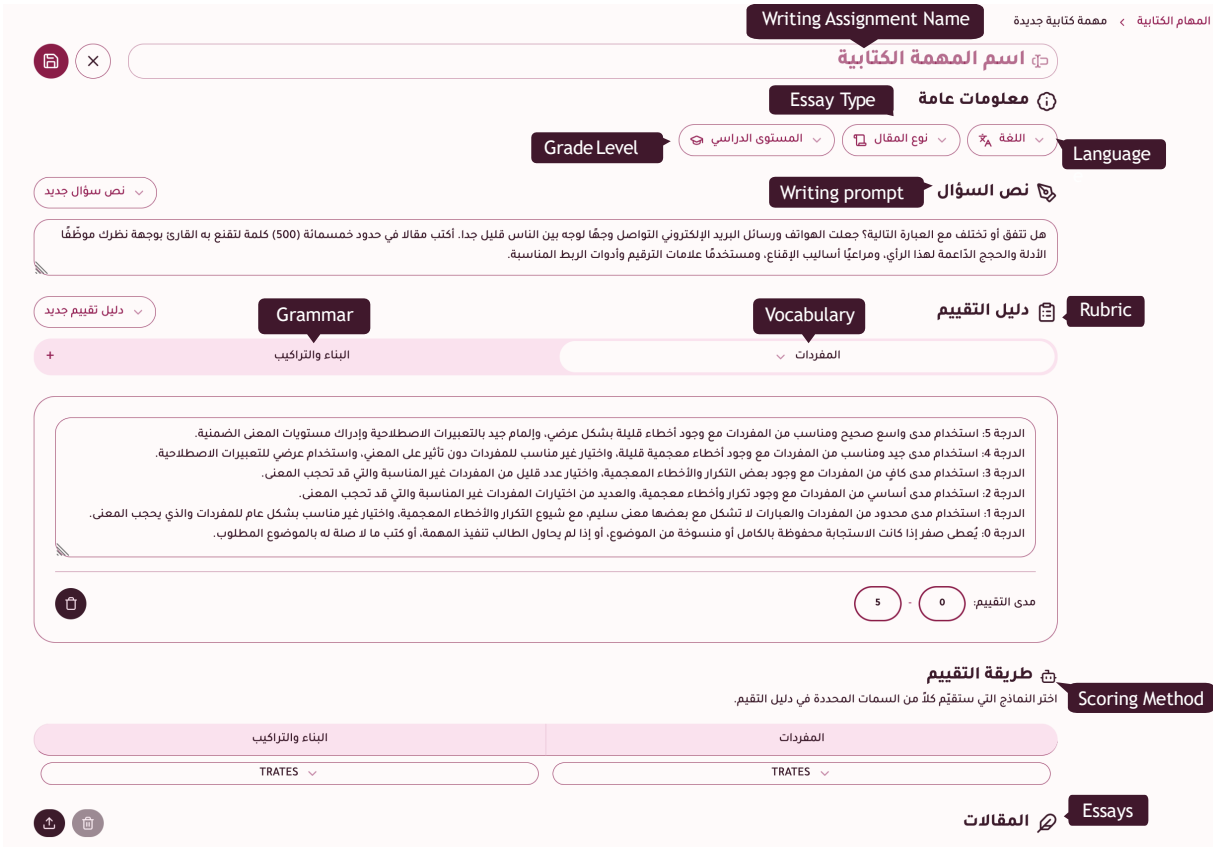

Figure 2: The assignment setup interface in Arabic, with English header translations. English translations of the prompt and rubric are provided in Appendix A.

with other supplementary information (§3.2). Essays are then uploaded and evaluated for the selected traits by the chosen scoring models (§3.3). The resulting scores can be reviewed and, if needed, manually refined by the user (§3.4).

3.1 Prompt and Rubric Design

The first step in designing a writing assignment is to create the *writing prompt* and the corresponding *trait rubrics*. The writing prompt refers to the instructions and expected topic to be discussed in the essays. The rubric defines the scoring criteria for evaluating a specific writing trait over a corresponding score range. Adding prompts and rubrics on *Qayyem* enables their reuse in subsequent assignments. An instructor may, for example, have a common rubric to score 8th-grade student essays, but the prompt changes for different assignments.

3.2 Assignment Setup

Figure 2 illustrates the assignment creation interface (in Arabic, with English header translation). The assignment is constructed by specifying several needed information. This first includes defining the **general information**: title, language of the assignment, essay type (e.g., persuasive, explanatory, or argumentative), and target grade level. Additionally, a **writing prompt** and **trait rubrics** are either selected from previously saved ones, edited, or newly created, allowing for flexible reuse and customization. Finally, **scoring models** are selected, one per designated trait. Each scoring model is accompanied by a short description and a 0-5 star performance rating to provide users with an overview of its characteristics. For usability, each trait is automatically associated with a default model, selected based on its performance.

3.3 Scoring

After an assignment is created, essays can be uploaded in batches using a predefined format, ensuring consistent and efficient processing. The user can then select essays to be assessed and traits to be scored. The system processes the selected essays and returns scores in real time, as illustrated in Figure 6 in Appendix B, with all results stored to allow optional review or refinement at any time. Multiple scoring runs can be performed using different models, with each run tracked independently.

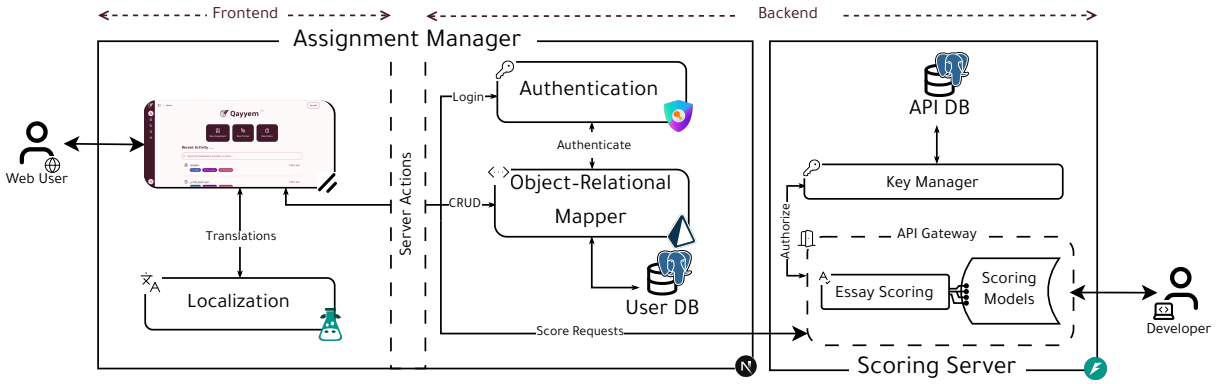

Figure 3: *Qayyem*'s High-level Architecture Diagram.

3.4 Review

The Review interface allows inspection and management of previously generated scoring results. For each essay, it displays the full essay text alongside evaluated traits, scoring models, and assigned scores. *Qayyem* maintains a history of all scoring attempts, which can be browsed, compared, or deleted as needed, enabling traceability across different models and assignment configurations. Trait scores can also be *manually refined* when necessary. Finally, a summary report can be generated with the finalized scores for convenient reference.

4 System Design and Implementation

This section presents the architecture of *Qayyem*, illustrated in Figure 3, which consists of two main components: the assignment manager (§4.1) and the scoring server (§4.2). The assignment manager provides the Web interface for managing assignments, while the scoring server hosts the models and returns the scores. The section then details the system's deployment (§4.3) and the specific scoring models used (§4.4).

4.1 Assignment Manager

The assignment manager is *Qayyem*'s primary interface, presented as a website through which instructors interact with the system. As shown in Figure 3, it is responsible for managing assignments and coordinating communication between users and the scoring server. It supports both Arabic and English user interfaces (UIs) and allows users to create, edit, and delete prompts, rubrics, and assignments, providing a centralized workspace for essay grading.

The assignment manager is implemented as a full-stack Web application using Next.js,[4] enabling tight integration between frontend interactions and backend logic. Actions performed through the Web interface are handled via server-side actions that enforce access control and trigger appropriate system workflows. This design allows instructors to access the essay scoring functionality without interacting directly with the scoring server or its API.

On the backend, the assignment manager handles user authentication, authorization, and persistent data management. Access to the system requires a valid user account. Assignment-related data, including users, prompts, rubrics, assignments, and scores, are stored in a PostgreSQL database[5] and accessed via an object-relational mapping layer implemented with Prisma,[6] ensuring a structured and consistent data model.

When essays are uploaded and submitted for scoring, the assignment manager sends an authorized scoring request to the scoring server through its API. During scoring, it tracks progress and updates the UI accordingly. Once completed, scores are stored in the database and made available through the UI.

4.2 Scoring Server

The scoring server is responsible for the core functionality of *Qayyem*, handling automated scoring requests submitted by both website users and external developers. As shown in Figure 3, it operates as a standalone backend service that exposes a unified API for accessing the available scoring models.

The scoring server is implemented using FastAPI[7] to manage its API endpoints. Upon startup, it reads from centralized configuration files that specify (among others) the available scoring

[4] https://nextjs.org/

[5] https://www.postgresql.org/

[6] https://www.prisma.io/

[7] https://fastapi.tiangolo.com/

models, including which models to enable and load, along with their associated metadata. This approach allows system administrators to add or disable models without modifying the core server logic, as all models adhere to a shared interface.

During execution, the scoring server manages active scoring requests and provides real-time progress updates as essays are evaluated. These updates are streamed to clients using Server-Sent Events (SSEs). Once scoring is completed, the final results are returned through the API.

Access to the scoring server is protected through API key–based authorization. External requests are routed through an API gateway and require API keys, which are managed by a dedicated key manager, responsible for access control and rate limiting. The API supports both REST and Server-Sent Events (SSE) endpoints, enabling synchronous configuration queries and real-time streaming of scoring progress. API keys are requested via a dedicated form.[8] Detailed API documentation, including endpoint specifications, request formats, and code examples is available online.[9]

4.3 Deployment

Qayyem is deployed on a single Linux-based server with NVIDIA A10 GPU and AMD EPYC 74F3 24-core processor. NGINX[10] is used as a reverse proxy to handle incoming requests and route traffic to the appropriate system components. The assignment manager is deployed as a persistent Node.js service using PM2,[11] while the scoring server runs as a standalone backend service using Uvicorn.[12]

The scoring server relies on external configuration files to define the supported languages, grade levels, essay types, traits, and available scoring models. This allows the system to be adapted or extended without changes to the deployment setup, and enables controlled updates and maintenance. During deployment or restart, the scoring server loads the active configuration into memory, making the supported traits and models immediately available to the API and assignment manager. Models may be initialized at startup or activated on demand, depending on their configuration. This design allows the system to manage computational resources while remaining responsive to scoring requests.

As new scoring models emerge, they can be deployed by adding their corresponding configuration and model support files, without disrupting existing deployments. Previously-deployed models are retained or disabled when no longer needed, ensuring consistency between the scoring server and the assignment manager, particularly with respect to database records and the tracking of historical scores, where stable model identifiers are required to preserve and reference past results.

4.4 Scoring Models

The scoring models deployed on *Qayyem* are chosen to balance SOTA performance with inference efficiency, providing a diverse set of models for Arabic AES. The implementation details for all the models are provided in Appendix C. We note that although the currently deployed models support Arabic, the system's bilingual UI enables seamless integration of English AES models.

Feature-based Models Neural Networks (NN), Random Forest (RF), and XGBoost (XGB) were chosen as feature-based baselines for their strong performance among feature-based approaches (Li and Ng, 2024; Bashendy et al., 2026; Sayed et al., 2025),[13] and computational efficiency. They leverage 816 handcrafted features introduced by Sayed et al. (2025) to capture various linguistic aspects of essays, providing robust and interpretable scoring while serving as efficient complements to the more computationally-intensive deployed models.

State-of-the-Art (SOTA) Models We adapted two SOTA English AES models for Arabic. The first is TRATES (Eltanbouly et al., 2025), which generates trait-specific features using an LLM (Fanar (Team et al., 2025) in our adaptation), combined with linguistic features for scoring using a simple NN. The second is MOOSE (Chen et al., 2025), which combines mixture-of-experts for trait prediction, pairwise essay ranking, and essay–prompt relevance (using AraBERT encoder (Antoun et al., 2020) in our adaptation).

5 Experimental Evaluation

In this section, we report and discuss the performance of the deployed scoring models.

Experimental Setup To develop the deployed models, we utilize the newly-released dataset,

[8] https://qayyem.qu.edu.qa/api-request
[9] https://qayyem.qu.edu.qa/documentation
[10] https://nginx.org/
[11] https://pm2.keymetrics.io/
[12] https://uvicorn.dev/

[13] XGB achieves second-best performance on LAILA.

Model	REL	ORG	VOC	STY	DEV	MEC	GRM	HOL	AVG
RF	0.331	0.609	0.644	0.637	0.573	0.559	0.609	0.682	0.581
NN	0.353	0.609	0.621	0.631	0.566	0.565	0.597	0.651	0.574
XGB	0.360	0.645	0.641	0.641	0.583	0.577	0.619	0.679	0.593
MOOSE	0.411	0.627	0.642	0.649	0.585	0.586	0.623	0.649	0.597
TRATES	**0.557**	**0.696**	**0.657**	**0.664**	**0.652**	**0.608**	**0.643**	**0.744**	**0.653**

Table 1: Average QWK performance over all prompts for each trait. **Bold** and underlined values indicate the best and second-best performance per trait.

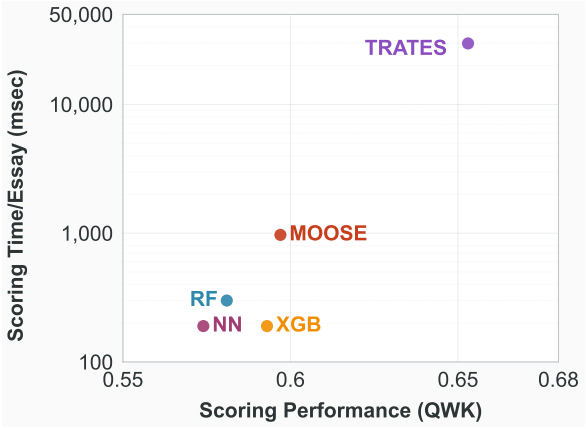

Figure 4: Scoring effectiveness (in QWK) vs. per-essay inference time (in msec) across deployed models.

LAILA (Bashendy et al., 2026), designed to advance robust Arabic AES research. It contains 7,859 Arabic essays written by students in grades 10-12 across 8 prompts (5 persuasive and 3 explanatory). The essays are annotated using a unified scoring rubric in 7 writing traits: Relevance (REL, 0–2), Organization (ORG, 0–5), Vocabulary (VOC, 0–5), Style (STY, 0–5), Development (DEV, 0–5), Mechanics (MEC, 0–5), and Grammar (GRM, 0–5), in addition to Holistic score (HOL) computed as the sum of all trait scores. To evaluate the models, we use Quadratic Weighted Kappa (QWK) (Cohen, 1968), a common measure for AES that assesses the agreement between the scores of two raters. All models are trained in a cross-prompt setting using the released splits of LAILA dataset.

Effectiveness Table 1 reports the average performance on LAILA over the 8 prompts for each writing trait. TRATES achieves the best overall performance across all traits, followed by MOOSE, highlighting the superiority of SOTA approaches over traditional feature-based models. Notably, TRATES outperforms MOOSE by 15 points on the relevance trait. For vocabulary and style, performance differences across models are relatively small, with a gap range of about 3.5 points. This gap increases to around 5 points for mechanics and grammar, and further widens to 9 points for holistic, development, and organization traits. However, these performance gains must be interpreted in light of the computational cost required to achieve them.

Effectiveness–Efficiency Trade-off Figure 4 illustrates the scoring performance measured in QWK vs. average inference time per essay. Feature-based models are clustered in the low inference-time region, with NN, RF, and XGB requiring 0.2, 0.3, and 0.2 seconds per essay, respectively, exhibiting relatively lower QWK values than other models. MOOSE occupies an intermediate position, with an inference time of 1 second per essay and a bit higher QWK than feature-based models, reflecting its more complex architecture and increased computational requirements. In contrast, TRATES lies at the extreme end of the inference-time spectrum, requiring 30 seconds per essay while achieving the best QWK. This increased inference time is attributed to its trait-specific design, which involves a separate model call for each trait. In addition, it requires substantially greater computational resources, either through high-capacity GPUs for local inference or LLM-based API calls.

This comparison guides users in selecting models for trait-level essay scoring, highlighting the trade-off between effectiveness and efficiency.

6 Conclusion

In this paper, we introduced *Qayyem*, the first Arabic AES platform to support fully-automated essay scoring while providing instructors with a flexible and user-friendly interface. The platform allows instructors to create assignments, define writing prompts and trait-specific rubrics, and configure scoring methods. The current version of *Qayyem* supports five scoring models, enabling users to select and apply SOTA trait-based models according to their needs. By integrating real-time scoring, manual score refinement, and report generation, *Qayyem* provides a comprehensive solution for managing the full essay evaluation workflow.

As future work, we plan to extend *Qayyem* to support additional Arabic scoring models, incorporate English scoring models, integrate systems capable of generating formative feedback to students, support student user scenarios, and provide a tiered system for different types of users.

Acknowledgment

The work of Sohaila Eltanbouly was supported by GSRA grant# GSRA12-L-0413-250111, and the work of the other authors was supported by NPRP grant# NPRP14S-0402-210127, both from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors.

References

Abeer Alqahtani and Amal Alsaif. 2020. Automated Arabic essay evaluation. In *Proceedings of the 17th International Conference on Natural Language Processing (ICON)*, pages 181–190, Indian Institute of Technology Patna, Patna, India. NLP Association of India (NLPAI).

Waleed Alsanie, Mohamed I Alkanhal, Mohammed Alhamadi, and Abdulaziz O Alqabbany. 2022. Automatic scoring of Arabic essays over three linguistic levels. *Progress in Artificial Intelligence*, pages 1–13.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resources Association.

May Bashendy, Walid Massoud, Sohaila Eltanbouly, Salam Albatarni, Marwan Sayed, Abrar Abir, Houda Bouamor, and Tamer Elsayed. 2026. LAILA: A large trait-based dataset for Arabic automated essay scoring. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2026)*. Association for Computational Linguistics.

Jill Burstein, Martin Chodorow, and Claudia Leacock. 2003. Criterionsm online essay evaluation: An application for automated evaluation of student essays. In *Proceedings of the Fifteenth Annual Conference on Innovative Applications of Artificial Intelligence (IAAI)*, pages 3–10. Association for the Advancement of Artificial Intelligence. Reposted with permission on ETS.org.

Kumar Satvik Chaudhary, Chengshuai Zhao, Fan Zhang, Yung Hin Tse, Garima Agrawal, Yuli Deng, and Huan Liu. 2025. EssayCBM: Rubric-aligned concept bottleneck models for transparent essay grading. *arXiv preprint arXiv:2512.20817*.

Po-Kai Chen, Bo-Wei Tsai, Shao Kuan Wei, Chien-Yao Wang, Jia-Ching Wang, and Yi-Ting Huang. 2025. Mixture of ordered scoring experts for cross-prompt essay trait scoring. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18071–18084, Vienna, Austria. Association for Computational Linguistics.

Kirill Chirkunov, Bashar Alhafni, Chatrine Qwaider, Nizar Habash, and Ted Briscoe. 2025. ARWI: Arabic write and improve. In *Proceedings of the Fourth Workshop on Intelligent and Interactive Writing Assistants (In2Writing 2025)*, pages 11–18, Albuquerque, New Mexico, US. Association for Computational Linguistics.

Jacob Cohen. 1968. Weighted kappa: nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin*, 70(4):213.

Fei Dong, Yue Zhang, and Jie Yang. 2017. Attention-based recurrent convolutional neural network for automatic essay scoring. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 153–162, Vancouver, Canada. Association for Computational Linguistics.

Sohaila Eltanbouly, Salam Albatarni, and Tamer Elsayed. 2025. TRATES: Trait-specific rubric-assisted cross-prompt essay scoring. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20528–20543, Vienna, Austria. Association for Computational Linguistics.

Rayed Ghazawi and Edwin Simpson. 2024. Automated essay scoring in Arabic: a dataset and analysis of a BERT-based system. *arXiv preprint arXiv:2407.11212*.

Rayed Ghazawi and Edwin Simpson. 2025. How well can LLMs grade essays in Arabic? *Computers and Education: Artificial Intelligence*, page 100449.

Jiefu Gong, Xiao Hu, Wei Song, Ruiji Fu, Zhichao Sheng, Bo Zhu, Shijin Wang, and Ting Liu. 2021. IFlyEA: A Chinese essay assessment system with automated rating, review generation, and recommendation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 240–248, Online. Association for Computational Linguistics.

Reo Hirao, Mio Arai, Hiroki Shimanaka, Satoru Katsumata, and Mamoru Komachi. 2020. Automated essay scoring system for nonnative Japanese learners. In *Proceedings of the Twelfth Language Resources and Evaluation Conference (LREC 2020)*, pages 1250–1257, Marseille, France. European Language Resources Association (ELRA).

Rahul Kumar, Sandeep Mathias, Sriparna Saha, and Pushpak Bhattacharyya. 2022. Many hands make light work: Using essay traits to automatically score essays. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1485–1495, Seattle, United States. Association for Computational Linguistics.

Shengjie Li and Vincent Ng. 2024. Conundrums in cross-prompt automated essay scoring: Making sense of the state of the art. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7661–7681, Bangkok, Thailand. Association for Computational Linguistics.

Zhexiong Liu, Diane Litman, Elaine L Wang, Tianwen Li, Mason Gobat, Lindsay Clare Matsumura, and Richard Correnti. 2025. eRevise+RF: A writing evaluation system for assessing student essay revisions and providing formative feedback. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (System Demonstrations)*, pages 173–190, Albuquerque, New Mexico. Association for Computational Linguistics.

Rim Aroua Machhout and Chiraz Ben Othmane Zribi. 2024. Enhanced BERT approach to score Arabic essay's relevance to the prompt. *Communications of the IBIMA*, 2024.

Somaia Mahmoud, Emad Nabil, and Marwan Torki. 2024. Automatic scoring of Arabic essays: A parameter-efficient approach for grammatical assessment. *IEEE Access*.

Christopher Michael Ormerod. 2022. Mapping between hidden states and features to validate automated essay scoring using deberta models. *Psychological Test and Assessment Modeling*, 64(4):495–526.

Robert Ridley, Liang He, Xin-yu Dai, Shujian Huang, and Jiajun Chen. 2021. Automated cross-prompt scoring of essay traits. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 13745–13753.

Marwan Sayed, Sohaila Eltanbouly, May Bashendy, and Tamer Elsayed. 2025. Feature engineering is not dead: A step towards state of the art for Arabic automated essay scoring. In *Proceedings of the Arabic Natural Language Processing Conference (Arabic-NLP 2025)*, pages 231–245, China.

Kaveh Taghipour and Hwee Tou Ng. 2016. A neural approach to automated essay scoring. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1882–1891, Austin, Texas. Association for Computational Linguistics.

Fanar Team, Ummar Abbas, Mohammad Shahmeer Ahmad, Firoj Alam, Enes Altinisik, Ehsannedin Asgari, Yazan Boshmaf, Sabri Boughorbel, Sanjay Chawla, Shammur Chowdhury, and 1 others. 2025. Fanar: An Arabic-centric multimodal generative ai platform. *arXiv preprint arXiv:2501.13944*.

Jiayi Xie, Kaiwei Cai, Li Kong, Junsheng Zhou, and Weiguang Qu. 2022. Automated essay scoring via pairwise contrastive regression. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2724–2733, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Chunyun Zhang, Jiqin Deng, Xiaolin Dong, Hongyan Zhao, Kailin Liu, and Chaoran Cui. 2025. Pairwise dual-level alignment for cross-prompt automated essay scoring. *Expert Systems with Applications*, 265:125924.

A Writing Prompt and Rubrics Examples

Figure 5 and Table 2 illustrate the English translation of an example writing prompt and the corresponding rubric for the vocabulary trait drawn from LAILA dataset.

> Do you agree or disagree with the following statement? Telephones and emails have made face-to-face communication between people much less likely. Write a 500-word essay to persuade the reader of your point of view, employing supporting evidence and arguments, considering persuasive techniques, and using appropriate punctuation and linking words.

Figure 5: Example of a writing prompt from LAILA.

Score	Description
1	Use of a limited range of vocabulary and phrases that do not make sense together, with repetition and lexical errors, and generally inappropriate vocabulary that obscures meaning.
2	Use of a basic range of vocabulary, with repetition, lexical errors, and many inappropriate choices that may obscure meaning.
3	Use of a sufficient range of vocabulary, with some repetition and lexical errors, and a small number of inappropriate choices that may obscure meaning.
4	Use of a good and appropriate range of vocabulary with few lexical errors, occasional inappropriate choices that do not affect meaning, and occasional use of idiomatic expressions.
5	Use of a broad, correct, and appropriate range of vocabulary with few errors, demonstrating good knowledge of idiomatic expressions and awareness of implicit levels of meaning.

Table 2: Rubric for the **vocabulary** trait.

B Scoring Interface

Figure 6 shows the real-time scoring view after a batch of essays has been scored for three traits: development, relevance, and style. Each row corresponds to an essay, with columns showing the

Figure 6: Real-time scoring view of 5 essays evaluated across three traits: development, relevance, and style.

Model	Hyperparameter	Value
RF	max depth	6
	max features	0.1
	max samples	0.1
	min samples split	5
	#estimators	200
	feature selection threshold	0.4
XGB	#estimators	300
	max depth	3
	learning rate	0.1
	subsample	0.9
	feature selection threshold	0.1
NN	#epochs	50
	#layers	2
	dropout	0.3
	loss	MSE
	optimizer	AdamW
	learning rate	1e-4
	batch size	16
	hidden layer widths	256
	feature selection threshold	0.5

Table 3: The hyperparameter values used to train the deployed RF, XGB, and NN models

scores assigned by the selected model for each trait. Scores are rendered as star ratings reflecting the score range of the trait: 0-2 for relevance and 0-5 for development and style. The State column indicates whether scoring is completed successfully, and the model used for each trait is shown beneath the trait name. From this view, the user can open an individual essay for inspection and possible manual edits of the scores, or return to the assignment to launch additional scoring batches.

C Implementation Details of Scoring Models

All models are trained on LAILA dataset using the released cross-prompt splits,[14] following a leave-one-prompt-out cross-validation setup. Hyperparameter tuning is performed, and the best configuration is selected based on development set performance, after which results are reported on the unseen target prompt. For training the deployment models, the models are retrained on the full dataset using the most frequently selected hyperparameter configurations across the individual folds.

C.1 Feature-based Models

The three feature-based models are trained in a multi-task setup, where all traits are predicted simultaneously. For model training, we adopt the 816 handcrafted features introduced by Sayed et al. (2025) for Arabic AES, covering surface, readability, lexical, semantic, and syntactic features. To reduce noise, we performed feature selection based on Pearson and Spearman correlations (Li and Ng, 2024), retaining features whose absolute correlation with any trait exceeded a predefined threshold. For the implementation of RF and XBG, we used the sklearn library[15] and the XGBoost library,[16] while the NN is implemented using Pytorch. Table 3 summarizes the hyperparameters used to train

[14] https://gitlab.com/bigirqu/laila
[15] https://scikit-learn.org/
[16] https://xgboost.readthedocs.io/en/stable/

	REL	ORG	VOC	STY	DEV	MEC	GRM	HOL
loss	WMSE	WMSE	WMSE	WMSE	WMSE	WMSE	WMSE	MSE
learning rate	0.001	0.0001	0.0001	0.01	0.001	0.01	0.0001	0.01
hidden layer number & widths	64	64	32	64	32, 16	64	64	32, 32
weight decay	1e-6	1e-4	1e-4	1e-4	1e-4	1e-6	1e-4	1e-5
dropout	0.1	0	0	0	0	0.2	0.2	0.2
feature selection threshold	0.5	0.5	0.4	0.3	0.5	0.5	0.5	0.5
#epochs	80	87	85	36	74	52	76	69

Table 4: The hyperparameter values used to train TRATES. WMSE refers to the weighted MSE.

Hyperparameter	Value
batch_size	4
#epoch	15
encoder	AraBERT
learning_rate	2e-5

Table 5: The hyperparameter values used to train the deployed MOOSE model.

the deployment models.

For deployment, 22 features that require pretrained transformer models for extraction are removed. These features were found to contribute minimally to overall performance while increasing scoring time; therefore, they are excluded to improve efficiency.

C.2 TRATES Model

TRATES pipeline consists of three stages (Eltanbouly et al., 2025). In the feature generation stage, we use `Fanar-1-9B-Instruct`[17] to generate trait-specific features from the rubrics. For holistic scoring, LAILA dataset does not provide a separate rubric for the holistic score. Therefore, to generate features for holistic scoring, Fanar is prompted with an aggregated list of all trait features and asked to generate a set of 10 features covering the main scoring aspects. In the feature extraction stage, Fanar is also used, with extraction optimized by querying all rubric-based features for a given essay in a single call rather than a separate call for each feature. In the final stage, which combines trait-specific features with engineered linguistic features, we use the same feature set adopted for the feature-based models, along with the same feature selection method. This stage trains trait-specific neural network regression models that take the combined feature set as input and predict the corresponding trait score. The hyperparameters used for the deployed models are presented in Table 4.

C.3 MOOSE Model

MOOSE (Chen et al., 2025) is a multi-trait, cross-prompt essay scoring model composed of three specialized experts: a scoring expert, which learns inherent scoring cues from essays; a ranking expert, which evaluates the relative quality across different essays; and an adherence expert, which estimates the degree of prompt adherence. To adapt MOOSE for Arabic, we replaced BERT with AraBERT and incorporated the engineered Arabic feature set for both essay and prompt representations (Sayed et al., 2025). For implementation, we used the official implementation released by the authors.[18] During hyperparameter tuning, the learning rate was explored over the values 1e-4 and 2e-5, while all other architectural parameters were kept consistent with those reported in the original study. The final hyperparameters are reported in Table 5.

[17] https://huggingface.co/QCRI/Fanar-1-9B-Instruct

[18] https://github.com/antslabtw/MOOSE-AES