MISSION at AraGenEval Shared Task: Enhanced Arabic Authority Classification

THAMER MASEER ALHARBI
tamr4947@gmail.com

Abstract

This paper describes the approach developed for the AraGenEval shared task, with a focus on Arabic authorship identification and AI-generated text detection. Transformer-based models, including ALLaM-7B-Instruct-preview for Subtask 2 and AraModernBERT for Subtask 3, were fine-tuned using both the official and additional datasets. Prompt engineering and transfer learning techniques were adapted to address challenges specific to the Arabic language. Competitive performance was achieved on both subtasks, and all code and resources have been made publicly available to facilitate reproducibility.

Arabic NLP, Authorship Identification, AI-generated Text Detection, Transformer Models, Prompt Engineering, ALLaM, AraModernBERT

1 Introduction

This paper is prepared for the *AraGenEval: Arabic Authorship Style Transfer and AI-Generated Text Detection* shared task (Abudalfa et al., 2025) and presents our approach to **Subtask 2: Authorship Identification** and **Subtask 3: ARATECT – Arabic AI-Generated Text Detection**. Subtask 2 is formulated as a multi-class classification problem, where the goal is to predict the author of a given Arabic text from a predefined set of candidates. Subtask 3 is framed as a binary classification problem, aiming to distinguish between human-written and machine-generated Arabic text. Both subtasks are conducted entirely in Arabic, posing unique linguistic and modeling challenges. To address these tasks, we employed two transformer-based models pretrained on large-scale Arabic corpora (Bari et al., 2025; NAMAA, 2025). Each model was fine-tuned for its respective subtask to adapt to the target domains and maximize performance. Our approach achieved competitive results in the official evaluation, ranking **4th** in Subtask 2 and **3rd** in Subtask 3. All training and inference code is publicly available on Kaggle.

2 Datasets

This work uses datasets provided as part of the *AraGenEval* shared task, which focus on Arabic authorship and AI-generated text detection challenges(Abudalfa et al., 2025). For **Subtask 2** (Authorship Identification), the dataset consists of Arabic texts labeled with their respective authors. This dataset was provided by the shared task organizers (Abudalfa et al., 2025) and includes training, development, and test splits with a diverse set of authors, allowing for a multi-class classification setup. For **Subtask 3** (ARATECT), the task involves distinguishing human-written from machine-generated Arabic texts. We combined the dataset provided by the organizers (Abudalfa et al., 2025) with an additional publicly available Arabic AI-generated text dataset Al-Shaibani and Ahmed's (2025) to enhance the model's robustness. This binary classification dataset also contains balanced splits for training, development, and testing. Table 1 summarizes the key statistics of the datasets used for both subtasks, while Tables 2 and 3 provide sample instances illustrating the types of data in each subtask.

Task	Dev	Train	Test
AID entries	4157	35122	8413
ARATECT entries	500	17604	500

Table 1: Data Statistics.

Proceedings of The Third Arabic Natural Language Processing Conference, pages 14–17

text_in_author_style	author
الشتيم : العابس. الخديم : الخادم.	أحمـد شوقي
فلا يحجر على الفكر غير الفكر، ولا قوة تصد العقيدة غير العقيدة.	عباس محمود العقاد

Table 2: Example of Author Text in Arabic for subtask2.

content	Class
تقرير وليد العطار	human
رامي مخلوق يثير الجدل باستجدائه الأسد جدولة ضرائب على شركاته.	machine

Table 3: Example of human/machine text in Arabic.

3 System Overview and Experimental Setup

3.1 Hardware

For Subtask 2, we utilized four NVIDIA L4 GPUs, while for Subtask 3, a single NVIDIA Tesla P100 GPU was employed. All experiments were conducted on the Kaggle platform.

3.2 Subtask 2: Authorship Identification

For Subtask 2, We built upon the pipeline proposed by ducnh279 [1], which achieved first place in the KAChallenges Series 1: Classifying Math Problems competition [2]. Their approach leverages large language models (LLMs) fine-tuned for multi-class classification using prompt engineering combined with adapter-based training. Specifically, their method fine-tunes pretrained LLMs with carefully crafted prompts and lightweight LoRA adapters to efficiently adapt the model without full retraining. The training setup uses distributed data parallelism across multiple GPUs, mixed precision training, and 4-bit quantization for computational efficiency. A linear classification head is trained on top of the model backbone, and stratified K-fold cross-validation is used for robust evaluation. The model is trained with weighted cross-entropy loss to address class imbalance, and micro F1-score is used for validation. Our approach retains the core training framework, including distributed training, mixed precision, LoRA adapters, and quantization. However, we modified the prompt design and replaced the pretrained model with `ALLaM-7B-Instruct-preview` Bari et al.'s (2025) to better align with the authorship identification task. We designed a new prompt template to explicitly instruct the model to classify Arabic texts by their authors using a provided author list and corresponding numeric labels. The prompt template is shown in Figure 1. This prompt clearly guides the model to produce the author's label number as output, simplifying the classification task and improving focus. By fine-tuning `ALLaM-7B-Instruct-preview` with this prompt format and the existing training setup, we effectively adapted the model to the specific requirements of Subtask 2, resulting in competitive performance. Due to limited computational resources and the time constraints imposed by the Kaggle platform, we trained and evaluated our model using only the first fold of the stratified K-fold cross-validation instead of all folds. Despite this limitation, the model demonstrated strong performance. More details on our implementation and training code are publicly available in the accompanying Kaggle notebook[3].

```
<|im_start|>user
### Instruction:
صنف النص التالي حسب مؤلفه من القائمة المرفقة أدناه.
أجب برقم المؤلف فقط.
قائمة المؤلفين (مع الرقم المقابل):
٠: المؤلف١
١: المؤلف٢
...
ن: المؤلف
### Input:
النص
### Response:
```

Figure 1: Example of an Arabic prompt formatted for model input.

3.3 Subtask 3: Arabic AI-Generated Text Detection

For Subtask 3, we fine-tuned `AraModernBERT` NAMAA's (2025) using the shared task dataset combined with an additional external dataset Al-Shaibani and Ahmed's (2025). This task involves binary classification to distinguish human-written from machine-generated Arabic texts. We began by preprocessing the data, removing any miss-

[1] https://www.kaggle.com/code/ducnh279/kacs1-fine-tuning-qwen3-14b/notebook

[2] https://www.kaggle.com/competitions/classification-of-math-problems-by-kasut-academy

[3] https://www.kaggle.com/code/thameralharbi/subtask-2-authorship-identification-baseline-gpus

ing entries. The labels were encoded as integers, mapping `human` to 0 and `machine` to 1. To prepare inputs for the model, we implemented a custom PyTorch dataset that tokenizes the texts with a maximum length of 256 tokens and applies padding for batch consistency. The pretrained `AraModernBERT-Base-V1.0` model was loaded with a new classification head suitable for the binary task. Since the classification layer was randomly initialized, it was trained from scratch during fine-tuning. Training was performed using the AdamW optimizer with a learning rate of 2e-5 over four epochs. We used a batch size of 32 and applied dynamic padding through a data collator to efficiently batch variable-length inputs. Our approach effectively adapts a state-of-the-art Arabic pretrained model to the specific AI-generated text detection task, leveraging additional data to enhance performance. The full implementation and training scripts are publicly available on Kaggle[4].

4 Results

Metrics. The **Macro-F1** score was used as the primary evaluation metric. For this metric, the F1-score is computed independently for each class and then averaged, ensuring equal weight is given to all classes regardless of their frequency in the dataset. This provides a balanced evaluation, particularly in the presence of class imbalance. **Accuracy** was used as the secondary metric, measuring the proportion of correctly classified instances over the total number of predictions, without accounting for class distribution. As presented in the results tables, the system was ranked **4th** in **Subtask 2** and **3rd** in **Subtask 3**, with Macro-F1 scores of 84% and 80%, and accuracies of 89% and 79%, respectively (Tables 4[5] and 5[6]).

[4] https://www.kaggle.com/code/thameralharbi/arageneval-subtask3-aratect

[5] https://www.codabench.org/competitions/8545/#/results-tab

[6] https://www.codabench.org/competitions/9120/#/results-tab

#	Participant	F1-Score	Accuracy
1	muhammad-helmy	**0.89886**	0.92416
2	batoolnajeh	**0.87163**	0.90859
3	moamin007	**0.85968**	0.89516
4	7h4m3r	**0.83753**	0.89053
5	jenin	**0.83468**	0.87377
6	omarnj	**0.83138**	0.87519
7	rafiulbiswas	**0.82824**	0.86497
8	mohamedsabaa	**0.82743**	0.88898
9	tasnim_meem	**0.82669**	0.86414
10	nlp_wizard	**0.81303**	0.85285
11	shifali	**0.79673**	0.83335
12	sabarinathan1	**0.36758**	0.67075
13	syedsaba	**0.00779**	0.03174

Table 4: Leaderboard results for Subtask 2.

#	Participant	F1 Score	Accuracy
1	kaoutar	**0.86**	0.87
2	deleted_user_25186	**0.81**	0.79
3	7h4m3r	**0.80**	0.79
4	tasneemduridi	**0.78**	0.74
5	alizain157	**0.77**	0.76
6	omarnj	**0.76**	0.79
7	deleted_user_27804	**0.76**	0.77
8	shifali	**0.75**	0.72
9	mutazay	**0.74**	0.71
10	nlp_wizard	**0.74**	0.70
11	jenin	**0.68**	0.55
12	sowravnath	**0.67**	0.53
13	tasnim_meem	**0.66**	0.70
14	Hedi	**0.65**	0.49
15	mariamlabib	**0.63**	0.65
16	sabarinathan1	**0.62**	0.53

Table 5: Leaderboard results for Subtask 3.

5 Conclusion

In this work, we presented our approach for the AraGenEval shared task, addressing both Subtask 2 (Authorship Identification) and Subtask 3 (AI-Generated Text Detection). By fine-tuning transformer-based models tailored for Arabic language processing, we achieved competitive results despite limited computational resources. Our adaptations of existing pipelines, combined with effective use of external datasets and prompt engineering, demonstrate the potential of pretrained language models for challenging Arabic NLP tasks. Future work will explore more advanced architectures and data augmentation strategies to further improve performance and robustness.

Acknowledgments

I would like to thank the organizers of the AraGenEval shared task and the Arabic NLP community for providing this valuable opportunity and platform for collaboration. Special thanks to everyone who contributed and shared their solutions to

help others and advance the community. Finally, I am deeply grateful to my parents for their continuous support and encouragement throughout this journey.

References

Shadi Abudalfa, Saad Ezzini, Ahmed Abdelali, Hamza Alami, Abdessamad Benlahbib, Salmane Chafik, Mo El-Haj, Abdelkader El Mahdaouy, Mustafa Jarrar, Salima Lamsiyah, and Hamzah Luqman. 2025. The AraGenEval Shared Task on Arabic Authorship Style Transfer and AI-Generated Text Detection. In *Proceedings of the Third Arabic Natural Language Processing Conference (ArabicNLP 2025)*, Association for Computational Linguistics.

Maged S Al-Shaibani and Moataz Ahmed. 2025. The arabic ai fingerprint: Stylometric analysis and detection of large language models text. *arXiv preprint arXiv:2505.23276*.

M Saiful Bari, Yazeed Alnumay, Norah A. Alzahrani, Nouf M. Alotaibi, Hisham Abdullah Alyahya, Sultan AlRashed, Faisal Abdulrahman Mirza, Shaykhah Z. Alsubaie, Hassan A. Alahmed, Ghadah Alabduljabbar, Raghad Alkhathran, Yousef Almushayqih, Raneem Alnajim, Salman Alsubaihi, Maryam Al Mansour, Saad Amin Hassan, Dr. Majed Alrubaian, Ali Alammari, Zaki Alawami, and 7 others. 2025. AL-Lam: Large language models for arabic and english. In *The Thirteenth International Conference on Learning Representations*.

NAMAA. 2025. Aramodernbert: Advanced arabic language model through trans-tokenization and modernbert architecture. https://huggingface.co/NAMAA-Space/AraModernBert-Base-V1.0. Accessed: 2025-03-02.