REGLAT at AbjadGenEval Shared Task: Multi-Model Ensemble Approach for Arabic AI-Generated Text Detection

Mariam Labib[1,2] **Nsrin Ashraf**[1,3] **Ahmed M. Fetouh**[3] and **Hamada Nayel**[3,4]
[1]Computer Engineering, Elsewedy University of Technology, Cairo, Egypt
[2]Department of Electronics and Communications Engineering, Faculty of Engineering, Mansoura University, Egypt.
[3]Department of Computer Science, Faculty of Computers and AI, Benha University, Egypt
[4]Department of Computer Engineering and Information, College of Engineering, Prince Sattam Bin Abdulaziz University, Al-Kharj 16273, Saudi Arabia

Abstract

The rapid advancement of large language models necessitates robust methods for detecting AI-generated Arabic text. This paper presents our system for distinguishing human-written from machine-generated Arabic content. We propose a weighted ensemble combining AraBERTv2 and BERT-base-arabic, trained via 5-fold stratified cross-validation with class-balanced loss functions. Our methodology incorporates Arabic text normalization, strategic data augmentation using 16,678 samples from external scientific abstracts, and threshold optimization prioritizing recall. On the official test set, our system achieved an F1-score of 0.763, an accuracy of 0.695, a precision of 0.624, and a recall of 0.980, demonstrating strong detection of machine-generated texts with minimal false negatives at the cost of elevated false positives. Analysis reveals critical insights into precision-recall trade-offs and challenges in cross-domain generalization for Arabic AI text detection.

1 Introduction

The spread of Large Language Models (LLMs) has fundamentally transformed the creation of digital content, with models now capable of generating highly fluent Arabic text that challenges human discrimination capabilities (Al Katat et al., 2024). The detection of AI-generated Arabic text presents unique challenges due to the language's rich morphological complexity, extensive orthographic variations, diverse dialectal forms, and sophisticated syntactic structures distinct from Indo-European languages (Alayba, 2025; Abudalfa et al., 2025).

These linguistic properties create multifaceted challenges where both human writers and AI systems navigate complexity through fundamentally different processes (Labib et al., 2025). Recent research has shown that fine-tuned transformer models outperform traditional statistical approaches for synthetic text detection (Labib et al., 2025). For Arabic specifically, pre-trained models like AraBERT and its variants have demonstrated strong performance across NLP tasks (Karajeh et al., 2023). Early synthetic text detection approaches relied on statistical features and classical machine learning methods. The field has shifted decisively toward fine-tuning pre-trained language models, which demonstrate superior performance across detection tasks (Alharthi, 2025). Recent work explores zero-shot detection methods using probability curvature and watermarking techniques. However, these approaches primarily target high-resource languages, with limited exploration of morphologically rich languages like Arabic (Wang and Qu, 2025).

Detection of AI-generated Arabic text remains significantly underexplored compared to English. Recent work benchmarked various detection approaches on Arabic, revealing that fine-tuned Arabic BERT models outperform multilingual alternatives (Yaquine et al., 2025). The development of Arabic-specific transformers has significantly advanced Arabic NLP capabilities. AraBERT (Karajeh et al., 2023) pioneered BERT-based models for Arabic with segment-level pre-training and Farasa segmentation. Subsequent variants include ARBERT and MARBERT (Ezzeldin et al., 2025), which explore different tokenization strategies and corpus compositions. CAMeL-BERT investigated dialectal variations across Arabic varieties (Aljomah et al., 2025). These models offer complementary strengths AraBERT's linguistic preprocessing versus BERT-base-arabic's larger diverse corpus motivating our ensemble approach. Ensemble learning has proven effective for improving robustness and generalization in text classification (Zhang and Shafiq, 2024).

Recent work indicates that ensembles of models with heterogeneous architectures or pre-training strategies capture complementary features, yielding improved performance on challenging tasks such as authorship detection. In addition, validation-based weighted averaging outperforms simple voting. We extend these approaches to Arabic AI-generated text detection through principled model selection and threshold optimization.

This work presents a comprehensive ensemble system for the AbjadGenEval shared task on Arabic AI-generated text detection. We combine two Arabic BERT variants AraBERTv2 and BERT-base-arabic through weighted averaging with optimized decision thresholds. The main contributions of this model include:

- Development of a weighted ensemble architecture combining AraBERTv2 and BERT-base-arabic with empirically optimized thresholds, demonstrating effectiveness of model diversity for Arabic text authentication.
- Implementation of an effective data augmentation strategy leveraging 16,678 samples from external Arabic scientific abstracts.
- Empirical evidence of generalization challenges, with substantial performance gap between cross-validation (F1: 0.904-0.978) and test evaluation (F1: 0.763), highlighting domain adaptation needs.

The remainder of this paper is organized as follows: Section 2 reviews related work, Section 3 details our system architecture, Section 4 presents results, and Section 5 concludes with future directions.

2 Background

The AbjadGenEval shared task (Ezzini et al., 2026) requires binary classification of Arabic texts as human-written or machine-generated. The competition provides a training set of labeled Arabic texts and a test set of 200 unlabeled samples. Systems are evaluated using F1-score (primary metric), accuracy, precision, recall, and balanced accuracy. Key challenges include: (1) distinguishing subtle stylistic differences between human and AI-generated Arabic text, (2) generalizing across diverse domains, and (3) handling distribution shifts between training and test data.

3 System Overview

The proposed system consists of four stages: preprocessing, augmentation, training the base models, ensemble and evaluation.

3.1 Data Preprocessing

Arabic orthographic variability can hinder model performance. We implemented a comprehensive preprocessing pipeline:
Diacritic Removal: All Arabic diacritical marks (ḥarakāt) were removed, as they are inconsistently applied in digital text and increase vocabulary sparsity.
Character Normalization: We unified orthographic variants: Alef forms { أ ، إ ، آ }→ {ا}, Taa Marbuta {ه} → {ة}, Alef Maksura {ي} → {ى} and removed Tatweel { ـ }.
Whitespace Normalization: Multiple consecutive spaces collapsed to single spaces, with all leading and trailing whitespace removed.

This preprocessing reduced lexical variability while preserving semantic and syntactic properties essential for authorship detection.

3.2 Data Augmentation

To enhance model robustness, we augmented the competition training data with the KFUPM-JRCAI Arabic Generated Abstracts dataset[1]. This dataset contains parallel human-written and AI-generated scientific abstracts across three generation scenarios: (1) by polishing AI refinement of human abstracts, (2) from title generation from paper titles, and (3) from title and content generation from titles and paper content. We extracted human-written abstracts (labeled "`human`") and AI-generated abstracts from Allam and GPT-4 models (labeled "`machine`"). After removing duplicates and empty entries, this increased our training corpus from the original competition data to 16,678 samples, providing greater diversity in writing styles, exposure to multiple AI generation strategies, and more balanced representation of generation scenarios.

3.3 Model Selection

We selected two Arabic BERT models based on their architectural differences and pre-training characteristics:

[1]https://github.com/KFUPM-JRCAI/arabs-dataset

- AraBERTv2[2]: Pre-trained on 70 million Arabic sentences using segment-level pre-training with Farasa segmentation.
- BERT-base-arabic[3]: Trained on 8.2 billion Arabic tokens from diverse sources including Wikipedia, news, and web crawls.

The full hyperparameter configuration of both models is reported in Table 1.

Table 1: Shared fine-tuning hyperparameters for both LLMs

Hyperparameter	**Value**
Task	Binary classification
Classification head	$768 \rightarrow 2$
Hidden dropout	0.25
Attention dropout	0.25
Label smoothing	0.05

3.4 Training Methodology

We employed 5-fold stratified cross-validation with class distribution maintained across folds. All five folds' predictions were averaged for final test predictions. We computed balanced class weights (1.478 for human, 0.756 for machine) using scikit-learn's `compute_class_weight`, incorporated into weighted cross-entropy loss:

$$\mathcal{L} = -\sum_{i=1}^{N} w_{y_i} \log(p_{y_i})$$

Learning rate $1.5e-5$, warmup ratio 0.15, effective batch size 36 (12×3 gradient accumulation steps), weight decay 0.05, max gradient norm 1.0, max epochs 5, early stopping patience 2 epochs, `AdamW` optimizer and FP16 mixed precision. The complete setup of model configuration using `TrainingArguments` is illustrated in A.

3.5 Ensemble Strategy and Threshold Optimization

We combined models using weighted averaging based on cross-validation performance:

$$P_{\text{ensemble}} = 0.6 \times P_{\text{AraBERTv2}} + 0.4 \times P_{\text{BERT-arabic}}$$

For threshold optimization, we evaluated F1-scores on aggregated validation predictions for thresholds from 0.35 to 0.70 in 0.01 increments:

$$\text{threshold} = \arg max_{\theta} F1(\theta)$$

where $\theta \in [0.35, 0.70]$. This identified optimal threshold 0.69 for both models, prioritizing recall to minimize false negatives in machine-generated text detection.

[2]aubmindlab/bert-base-arabertv2

[3]asafaya/bert-base-arabic

4 Experimental Results

Table 2 presents the cross-validation performance of both models. Results reported in Table 2, shows that BERT-base-arabic substantially outperformed AraBERTv2, demonstrating superior discrimination capabilities. Both models achieved near-perfect recall ($\geq$ 0.9982), but BERT-base-arabic showed significantly higher precision and more stable performance across folds (*lower standard deviation*). Our ensemble system achieved the following performance on the AbjadGenEval test set.

Table 2: Performance of AraBERTv2 and BERT-arabic Models Across 5-Fold Cross-Validation.

	AraBERTv2	**BERT-arabic**
P	0.8263 ± 0.0379	0.9568 ± 0.0100
R	1.0000 ± 0.0000	0.9994 ± 0.0007
F1	0.9044 ± 0.2320	0.9776 ± 0.0048
Acc.	0.8497 ± 0.0486	0.9693 ± 0.0078

The results reveal a distinct precision-recall trade-off. High recall indicates exceptional ability to identify machine-generated texts, missing only 2% of AI-generated samples. However, moderate precision indicates approximately 38% false positive rate human texts incorrectly classified as machine-generated. This pattern reflects our design decisions: (1) class-balanced training with higher weight on machine class, (2) threshold optimization favoring recall, and (3) cross-validation behavior showing perfect recall across folds. BERT-base-arabic's superior cross-validation performance suggests our ensemble weighting may have been suboptimal. The model's training on 8.2 billion diverse tokens likely provided richer representations compared to AraBERTv2's 70 million sentences, enabling better discrimination with higher precision while maintaining recall.

Table 3: Results of ensemble on test set.

Metric	**Ensemble**
P	0.6242
R	0.9800
F1	0.7626
Acc.	0.6950
Balanced Acc.	0.6950

5 Conclusion

This work presented an ensemble system combining two Arabic BERT variants for AI-generated text detection. Our contributions include:

(1) systematic comparison revealing BERT-base-arabic's superiority for Arabic authorship detection, (2) effective data augmentation expanding training data to 16,678 samples, (3) comprehensive analysis of precision-recall trade-offs in detection systems, and (4) empirical evidence for generalization challenges in Arabic AI text detection. The performance gap between validation and testing underscores the continuing challenge of building robust detection systems that generalize across diverse Arabic text distributions and generation scenarios. Our findings provide actionable insights for developing more robust Arabic AI text detection systems suitable for real-world deployment.

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: Shared Tasks*, pages 1–13, Suzhou, China. Association for Computational Linguistics.

Souha Al Katat, Chamseddine Zaki, Hussein Hazimeh, Ibrahim El Bitar, Rafael Angarita, and Lionel Trojman. 2024. Natural language processing for arabic sentiment analysis: A systematic literature review. *IEEE Transactions on Big Data*, 10(5):576–594.

Abdulaziz M. Alayba. 2025. Arabic natural language processing (nlp): A comprehensive review of challenges, techniques, and emerging trends. *Computers*, 14(11).

Haifa Alharthi. 2025. Investigation into the identification of ai-generated short dialectal arabic texts. *IEEE Access*, 13:85131–85138.

Fay Aljomah, Lama Aldhafeeri, Maha Alfadel, Sultanh Alshahrani, Qaisar Abbas, and Sarah Alhumoud. 2025. Enhancing arabic sentiment analysis with pre-trained camelbert: A case study on noisy texts. *Computers, Materials and Continua*, 84(3):5317–5335.

Mohamed R. Ezzeldin, Gaber Sallam Salem Abdalla, and Abdoulie Faal. 2025. Enhancing arabic sentiment analysis via marbert: Domain adaptation with pseudo-labeling and contrastive learning. *Engineering Reports*, 7(12):e70528.

Saad Ezzini, Irfan Ahmed, Salmane Chafik, Shadi Abudalfa, Mo El-Haj, Ahmed Abdelali, Mustafa Jarrar, Nadir Durrani, Hassan Sajjad, and Farah Adeeba. 2026. Abjadgeneval: Abjad ai generated text detection shared task for languages using arabic script at abjadnlp 2026. In *Proceedings of the 2nd Workshop on NLP for Languages Using Arabic Script (AbjadNLP 2026), co-located with the 19th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2026)*, Rabat, Morocco.

Ola Karajeh, Mohammed N. Al-Kabi, and Edward A. Fox. 2023. Fusing arabert and graph neural networks for enhanced arabic text classification. In *2023 24th International Arab Conference on Information Technology (ACIT)*, pages 1–8.

Mariam Labib, Nsrin Ashraf, Mohammed Aldawsari, and Hamada Nayel. 2025. REGLAT at AraGenEval shared task: Morphology-aware AraBERT for detecting Arabic AI-generated text. In *Proceedings of The Third Arabic Natural Language Processing Conference: Shared Tasks*, pages 94–98, Suzhou, China. Association for Computational Linguistics.

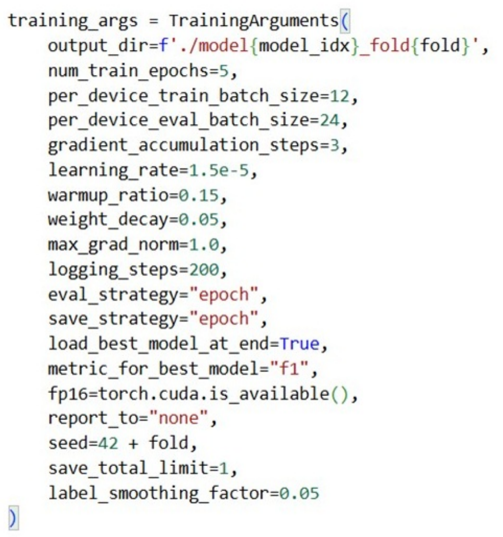
Yu Wang and Wen Qu. 2025. A tutorial on fine-tuning pretrained language models: Applications in social and behavioral science research. *Behavior Research Methods*, 57(12):336.

Saad Yaquine, Amine Hmimou, and Paolo Rosso. 2025. Magenta: Generating and detecting arabic machine-generated text in multiple domains. In *Arabic Language Processing: From Theory to Practice*, volume 2339, pages 151–159, Cham. Springer Nature Switzerland.

Hongzhi Zhang and M. Omair Shafiq. 2024. Survey of transformers and towards ensemble learning using transformers for natural language processing. *Journal of Big Data*, 11(1):25.

A Appendix

The complete setup of model configuration using `TrainingArguments` is illustrated below.



```
training_args = TrainingArguments(
    output_dir=f'./model{model_idx}_fold{fold}',
    num_train_epochs=5,
    per_device_train_batch_size=12,
    per_device_eval_batch_size=24,
    gradient_accumulation_steps=3,
    learning_rate=1.5e-5,
    warmup_ratio=0.15,
    weight_decay=0.05,
    max_grad_norm=1.0,
    logging_steps=200,
    eval_strategy="epoch",
    save_strategy="epoch",
    load_best_model_at_end=True,
    metric_for_best_model="f1",
    fp16=torch.cuda.is_available(),
    report_to="none",
    seed=42 + fold,
    save_total_limit=1,
    label_smoothing_factor=0.05
)
```