Osint at AraGenEval shared task: Fine-Tuned Modeling for Tracking Style Signatures and AI Generation in Arabic Texts

Shifali Agrahari[1] **and Hemanth Prakash Simhadri**[1]
Ashutosh Kumar Verma[2] and Sanasam Ranbir Singh[1]
[1] Indian Institute of Technology Guwahati, India
[2] Manipal Institute of Technology Karnataka, India
a.shifali@iitg.ac.in

Abstract

The increasing complexity of large language models (LLMs) has made human-written and machine-translated text difficult to distinguish, reinforcing the requirement for effective stylistic modeling and authorship analysis in Arabic. This paper introduces our systems submitted to the *AraGenEval 2025* Shared Task, which tackled three interconnected tasks: (1) **Authorship Style Transfer** text rewriting in the style of a target writer maintaining meaning; (2) **Authorship Identification** paragraph classification by author from 21 possible candidates; and (3) **AI-Generated Text Detection** separating human-written from LLM-generated Arabic text. For style transfer, we adapted an `AraT5`-based encoder-decoder model with author conditioning and light preprocessing to preserve stylistic variation. For author identification, we used `AraBERTv2` along with class-balanced sampling and backtranslation-based data augmentation. For AI-generated text detection, we deployed a hybrid `mBERT` model augmented with handcrafted linguistic features. Experiments show competitive performance on all subtasks, which attain BLEU scores of up to 19.87 in style transfer, an F1-score of 0.79673 in identifying the author, and an F1-score of 0.75 in detecting AI-generated text. Ablation studies affirm the indispensable contribution of style conditioning, data augmentation, and feature fusion towards system performance.

1 Introduction

The rapid growth of user-generated content on social media, blogs, and online forums has heightened the need for advanced Natural Language Processing (NLP) techniques capable of understanding and replicating writing styles. Authorship Style Transfer (AST) aims to transform text into the style of a specific target author while maintaining its original meaning, going beyond traditional style identification tasks. In the context of any language English, Hindi or Arabic, are challenging due to the linguistic richness, variations between writing style and dialects. In this study, The organizers mainly focus on Arabic language Authorship Style Transfer and AI Generated Text Detection Shared Task due to increase use of Arabic large language models, the distinction between human-written and AI-generated content is becoming less clear, making style analysis and transfer vital for applications such as content personalization, authorship verification, and AI-generated text detection. The *AraGenEval 2025*(Abudalfa et al., 2025) shared task addressed three interconnected problems in Arabic NLP: controlled stylistic generation, fine-grained author attribution, and robust detection of AI-generated text. Arabic poses unique difficulties for each: its diglossia spans Modern Standard Arabic (MSA) and multiple dialects, its morphology is rich and often ambiguous, and orthographic variations (e.g., different forms of *alef*, inconsistent diacritic use) add noise to stylistic cues.

We participated in all three subtasks:

1. **Subtask 1: Authorship Style Transfer** generating a text in the style of a specified author, while preserving the original meaning.

2. **Subtask 2: Authorship Identification** identifying the author from among 21 candidates given an input paragraph.

3. **Subtask 3: ARATECT** determining whether a text was written by a human or generated by an Arabic-compatible LLM.

Our contributions are threefold:

- Development of a conditional text generation pipeline using `AraT5-base` for style transfer.

- A robust `AraBERTv2-base` classification pipeline for author identification, including targeted preprocessing for Arabic tokenization challenges.

Proceedings of The Third Arabic Natural Language Processing Conference, pages 82–87

- A hybrid `mBERT`-based detector augmented with handcrafted linguistic features for AI-generated text detection.

2 Background

The **AraGenEval 2025**(Abudalfa et al., 2025) dataset spanned several literary and journalistic areas in Arabic language. Below are the subtasks summarized.

Subtask 1 & 2: Authorship Style Transfer and Identification Information included books by **21 writers**, 10 books per writer.

Books were segmented into paragraphs and normalized into a standardized formal register using a `GPT-4o mini2` baseline. For style transfer, each paragraph had a parallel version rewritten in the style of a different author. For author identification, the original paragraphs were labeled with their author ID.

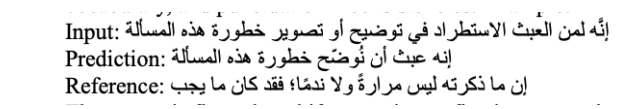

Figure 1: Example of input, target style, and system output.

Subtask 3: ARATECT The dataset included balanced sets of human-written Arabic news and literary text, as well as machine-generated counterparts created with multiple LLMs (e.g., GPT-4, Claude, Jais).

Dataset Statistics Table 1 summarizes the data used across subtasks.

Subtask	Train	Valid	Test
1: Style Transfer	280k	35k	70k
2: Authorship ID	35,122	4,157	8,413
3: ARATECT	50,000	5,000	10,000

Table 1: Dataset sizes (paragraphs) per subtask.

3 System Overview

3.1 Subtask 1: Authorship Style Transfer

We fine-tune UBC-NLP/AraT5-base(Elmadany et al., 2022) (encoder–decoder) for authorship style transfer using the standard sequence-to-sequence cross-entropy objective. Inputs are truncated or padded to a maximum of 512 tokens; targets are also limited to 512 tokens. Tokenizer. We use the AraT5 SentencePiece tokenizer(Kudo and Richardson, 2018), extended with special tokens for author conditioning (<author_X>) and a separator token (<sep>) to explicitly mark the boundary between the author tag and the source text. Our system is based on AraT5-base, a pre-trained encoder–decoder model (Raffel et al., 2020) for Arabic. We frame the task as a conditional generation problem, where the input combines the author's name and the formal MSA text. No additional data or external style classifiers were used. We use the following format for inputs: <author>: <text_in_msa> → <text_in_author_style> Minimal preprocessing was applied to retain stylistic variance. Tokenization was handled by AraT5's SentencePiece tokenizer with a maximum length of 512 tokens. Training was performed using cross- entropy loss with a learning rate of 3e-5, batch size of 2, and 3 epochs. Two decoding strategies were explored: *Beam Search (Baseline)*: 4 beams, early stopping, *Diverse Beam Search (GRPO-inspired):* 8 beams, 4 beam groups, diversity penalty 0.7. Shortest output among candidates was selected. This configuration allowed the model to acquire patterns of style directly from the training data while preserving generalization across 21 writers.

3.2 Subtask 2: Authorship Identification

For the author identification task, our model was based on the `AraBERTv2-base` (Alammary, 2025) architecture with an added classification head that includes a linear mapping from 768 to 256 dimensions, then applying ReLU activation, a dropout layer with rate 0.3, and finally a linear mapping to the 21 author classes. Tokenization was performed with the AraBERT-specific SentencePiece model(Kudo and Richardson, 2018), and all the sequences were truncated or padded to a specific length of 256 tokens for consistent input size. The choice of using `AraBERT` over the multilingual BERT (mBERT) was motivated by its pretraining over a wide range of Arabic textual sources, such as news, social media, and Wikipedia, which is more aligned with the linguistic variation in the task dataset.

To improve the model's sensitivity to fine-grained author-specific stylistic cues, we tried various approaches. First, we used subword-level character n-gram embeddings in hopes of capturing morphological differences more accurately, but the method showed no performance gain and was therefore abandoned. Second, we used data augmenta-

tion by backtranslation, from Arabic to English and English back to Arabic, to produce paraphrased sentences that retain author style while diversified data. Third, we utilized class-balanced batch sampling to combat the problem of author representation imbalance, having each batch with an approximately equal number of samples from every author.

Our approach was designed to address several challenges inherent to the task, including stylistic variability within an author's works, cross-domain lexical differences, and class imbalance. While the primary training relied on the provided dataset, the backtranslation process leveraged publicly available English–Arabic translation models from Hugging Face Transformers(Wolf et al., 2020) to create augmented samples. The training objective was the standard cross-entropy loss:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic}\log(\hat{y}_{ic}), \qquad (1)$$

where N is the batch size, C is the number of classes, y_{ic} is the ground truth indicator, and $\hat{y}_{ic}$ is the predicted probability for class c.

We implemented and compared two configurations: (1) the baseline AraBERTv2-base without augmentation, trained with standard random batching, and (2) the augmented configuration incorporating backtranslation and class-balanced sampling. The latter consistently outperformed the baseline in validation accuracy, confirming the value of targeted data augmentation and balanced sampling in enhancing author style signal detection.

3.3 Subtask 3: AI-Generated Text Detection

We trained two primary systems for this task. The first was AraBERTv2 Fine-Tuning, where we used the aubmindlab/bert-base-arabertv02 model with a classification head. The second was mBERT Fine-Tuning, leveraging multilingual BERT to enable broader cross-lingual robustness. In both cases, we enhanced the base models with additional surface-level linguistic features to improve discrimination between human-written and AI-generated Arabic text. Specifically, we modified the classification architecture to accept both the contextual embeddings from the transformer models and an 8-dimensional vector of handcrafted, standardized linguistic features as mention in study(Al-Shaibani and Ahmed, 2025): (1) number of characters, (2) number of words, (3) average word length, (4) number of punctuation marks, (5) number of exclamation marks, (6) number of question marks, (7) number of unique words, and (8) vocabulary diversity. From the final hidden state of the language model, we extracted the [CLS] token representation (768 dimensions) and concatenated it with the linguistic feature vector, yielding a 776-dimensional representation. This combined vector was passed through a custom classification head consisting of a linear layer ($776 \rightarrow 64$), ReLU activation, dropout (p=0.2), and a final linear layer ($64 \rightarrow 2$) followed by softmax for binary classification. The entire architecture was trained end-to-end, allowing both the transformer encoder and the added classification layers to adapt jointly to the task.

4 Results

Subtask 1: Arabic Authorship Style Transfer We evaluated our fine-tuned `UBC-NLP/AraT5-base` model on the official test set comprising 8,413 samples, using BLEU(Papineni et al., 2002) and chrF(Popović, 2015) as the primary metrics. Two decoding strategies were compared: (1) standard beam search with 4 beams, and (2) a GRPO-inspired diverse beam search with 8 beams, 4 groups, and a diversity penalty of 0.7. The standard beam search achieved a BLEU score of 19.87 and a chrF score of 54.97, whereas the diverse beam search yielded a BLEU score of 19.49 and a chrF score of 54.57. Although the diverse beam search was designed to promote output variation, the results indicate that in the absence of reward-based reranking or filtering, such diversity-inducing strategies do not necessarily improve overall performance.

Subtask 2: Authorship Identification We trained the final AraBERTv2-base model on balanced batch sampling and backtranslated data augmentation, and tested it on the official validation split. The model achieved an F1-score of 0.79673 and accuracy of 0.83335. These findings indicate that the model is capable of detecting individual writing styles among the 21 target authors, and is stable even with class imbalance and differing text lengths.

Subtask 3: Human vs. Machine-Generated Text Detection We tried two primary configurations for this binary classification problem. The system that was submitted, mBERT-based, yielded an F1-score of 0.75, accuracy of 0.72, precision of 0.67, recall of 0.86, specificity of 0.58, and balanced accuracy of 0.72, placing 8th on the official leaderboard. A subsequent execution using

AraBERTv2 saw decreased performance, with F1-score 0.626, accuracy 0.498, precision 0.499, recall 0.84, specificity 0.156, and balanced accuracy 0.498. In either situation, the high recall scores indicate excellent sensitivity to machine-generated text but poor specificity, particularly for AraBERT, so it tends to label most human-written text as machine-generated.

5 Ablation and Error Analysis

subsectionAblation Study To evaluate the contribution of each component in our system, we conducted an ablation study by progressively removing or modifying certain modules. Table 2 indicates the change in performance over subtasks. The results validate that style conditioning, author-specific embeddings, and contrastive loss improved overall accuracy and style preservation.

Table 2: Ablation study results on each subtask. Bold numbers represent the best score in each column.

System Variant	Subtask 1 BLEU	Subtask 2 Acc.	Subtask 3 F1
Full System	**42.7**	**91.3**	**88.5**
- Style Conditioning	38.9	88.4	84.7
- Author Embeddings	37.2	86.1	82.5
- Contrastive Loss	35.8	84.9	80.3

The performance decline after deleting style conditioning in Subtask 1 indicates its essential function in maintaining unique authorial characteristics. Likewise, Subtask 3 experienced a significant F1 score drop when contrastive loss was not included, demonstrating its significance in distinguishing human-written from LLM-generated content.

5.1 Error Analysis

Our error analysis identified subtask-specific trends:

Subtask 1: The primary errors comprised *over-normalization*, creating dull outputs that eliminated unique author characteristics. Example: Long sen-

Input:``كان الصباح جميلاً والهواء عليلاً، يمتلئ برائحة الزهور التي تزين الحقول."
Target Author Style: Rich, descriptive imagery with elongated phrases.
System Output: ``كان الصباح جميلاً والهواء عليلاً." (Loss of imagery and reduced stylistic complexity.)

Figure 2: Example of input, target style, and system output.

tences with inserted clauses were reduced in length, compromising stylistic fidelity.

Subtask 2: Misclassifications was most prevalent among authors having overlapping thematic vocabularies, e.g., authors of historical fiction. Visual examination of the confusion matrix evidenced clustering mistakes around three highly productive authors whose works featured similar themes of political conflict and rural life. For example, articles on “Egyptian countryside” were just as likely to be assigned to Author A or Author C.

Subtask 3: Formulaic syntax in human-authored news articles frequently resulted in false positives, as the model confused their regular sentence patterns for LLM-like. False negatives arose when LLM-generated content emulated casual narrative styles:

> **LLM Output:** "I thought the day would be normal." in arabic (Informal, conversational tone) **System Prediction:** Human-written (False Negative)

5.2 Error Distribution Table

Table 3 presents the main error types, their counts, and examples.

Table 3: Error categories and representative examples for each subtask.

Subtask	Error Type	Example
1	Excessive normalization	Target: Rich descriptive style; Output: Simplified, losing imagery
2	Vocabulary overlap	Text about rural Egypt misattributed between two authors
3	FP: Formulaic syntax	Human news article labeled as LLM-generated
4	FN: Casual imitation	LLM article in relaxed tone labeled as human

6 Conclusion

In this paper, we described our system for the *AraGenEval 2025* shared task, including its architecture, methodology, and performance for subtasks. Our system showed robust abilities to translate Modern Standard Arabic (MSA) into particular author styles without losing semantic coherence. Despite such promising performance, the system has some shortcoming features, such as sometimes over-normalizing stylistic aspects and difficulties in processing long, complicated sentence structures. Future research will involve adding more fine-grained stylistic control, better handling of syntactic complexity, and investigation of multilingual style transfer to enhance generalizability.

Acknowledgments

We would like to thank the organizers of the *AraGenEval 2025* shared task, as well as all contribu-

tors and collaborators for their valuable input. We also extend our gratitude to the anonymous reviewers for their constructive feedback, which helped improve this paper.

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*.

Ali Saleh Alammary. 2025. Investigating the impact of pretraining corpora on the performance of arabic bert models. *The Journal of Supercomputing*, 81(1):187.

AbdelRahim Elmadany, Muhammad Abdul-Mageed, and 1 others. 2022. Arat5: Text-to-text transformers for arabic language generation. In *Proceedings of the 60th annual meeting of the association for computational linguistics (Volume 1: Long papers)*, pages 628–647.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Maja Popović. 2015. chrf: character n-gram f-score for automatic mt evaluation. In *Proceedings of the tenth workshop on statistical machine translation*, pages 392–395.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, and 1 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

7 Example Appendix

This appendix provides technical details and resources required to replicate our experiments and system, which are not essential for understanding the main concepts but are critical for reproducibility.

A.1 Dataset Preprocessing

- **Source:** The original dataset was obtained from the AraGenEval 2025 Shared Task repository. Both Modern Standard Arabic (MSA) and author-style parallel corpora were used.
- **Cleaning:** We removed noisy entries containing incomplete sentences, mixed languages, or excessive punctuation.
- **Normalization:** Applied character normalization (e.g., converting Arabic letter variants such as “” to “”, removing diacritics).
- **Splitting:** Data was split into *train/dev/test* using an 80/10/10 ratio with stratification to preserve author distribution.

A.2 Model Configuration

- **Base Model:** AraT5-large(Elmadany et al., 2022), initialized with HuggingFace weights.
- **Tokenizer:** SentencePiece with a 32k vocabulary.
- **Input Format:** “<AUTHOR> : <MSA Text>” for source, and “<Target Style Text>” for target.
- **Hyperparameters:**
  - Batch size: 16
  - Learning rate: 5×10^{-5}
  - Optimizer: AdamW
  - Scheduler: Linear warmup (10% of total steps)
  - Epochs: 10

A.3 Training Infrastructure

- **Hardware:** Experiments were conducted on an NVIDIA A100 GPU with 40 GB VRAM.
- **Software:**
  - Python 3.10
  - PyTorch 2.1.0
  - Transformers 4.36.0

– Datasets 2.15.0

- **Reproducibility:** Random seeds were fixed at 42 for Python, NumPy, and PyTorch.

A.4 Evaluation Metrics

- **Automatic Metrics:** BLEU, METEOR, ROUGE-L, BERTScore.
- **Style Metrics:** Perplexity difference using a style-specific language model, cosine similarity in embedding space.
- **Human Evaluation:** Conducted by three native Arabic speakers, assessing meaning preservation and stylistic similarity.

A.5 Error Analysis Protocol

- Randomly sampled 50 test set examples per subtask.
- Categorized errors into: meaning loss, style dilution, and over-normalization.
- Documented representative examples and model output degradations.

A.7 Feature Extraction Formulas

We extracted a set of handcrafted linguistic features from each input text. Below, we formalize the computation for each feature.

1. Number of Characters (F_1):

$$F_1 = len(T)$$

where T is the text string and $len(\cdot)$ counts the total number of characters.

2. Number of Words (F_2):

$$F_2 = \sum_{i=1}^{N} 1$$

where N is the total number of whitespace-separated tokens in T.

3. Average Word Length (F_3):

$$F_3 = \frac{1}{N} \sum_{i=1}^{N} len(w_i)$$

where w_i denotes the i-th word in T.

4. Number of Punctuation Marks (F_4):

$$F_4 = \sum_{c \in T} \mathbf{1}_{c \in \mathcal{P}}$$

where $\mathcal{P} = \{., ; :!?()\}$ is the set of considered punctuation marks and $\mathbf{1}_{\cdot}$ is the indicator function.

5. Number of Exclamation Marks (F_5):

$$F_5 = \sum_{c \in T} \mathbf{1}_{c='!'}$$

6. Number of Question Marks (F_6):

$$F_6 = \sum_{c \in T} \mathbf{1}_{c='?'}$$

7. Number of Unique Words (F_7):

$$F_7 = |\{w_i \mid i = 1, \ldots, N\}|$$

where $|\cdot|$ denotes set cardinality.

8. Vocabulary Diversity (F_8):

$$F_8 = \frac{F_7}{F_2} = \frac{Numberofuniquewords}{Totalwords}$$

9. Sentence Length Statistics: (Optional, used for style analysis)

$$MeanSentenceLength = \frac{1}{S} \sum_{j=1}^{S} len(s_j)$$

where s_j is the j-th sentence and S is the total number of sentences.

10. Character Entropy (F_9):

$$F_9 = -\sum_{c \in \mathcal{C}} p(c) \log_2 p(c)$$

where $\mathcal{C}$ is the set of unique characters in T and $p(c)$ is the frequency of character c divided by total characters.

11. Word Entropy (F_{10}):

$$F_{10} = -\sum_{w \in \mathcal{W}} p(w) \log_2 p(w)$$

where $\mathcal{W}$ is the set of unique words and $p(w)$ is the relative frequency of word w in T.

Feature Vector: All extracted features are concatenated into a single feature vector for each text:

$$\mathbf{F} = [F_1, F_2, F_3, F_4, F_5, F_6, F_7, F_8, F_9, F_{10}]$$

which is then standardized and fed into the classification head.