Nojoom.AI at AraGenEval Shared Task: Arabic Authorship Style Transfer

Hafsa Kara Achira
Nojoom.AI
ih_karaachira@esi.dz

Mourad Bouache
Nojoom.AI
bouache@nojoom.ai

Mourad Dahmane
Nojoom.AI
mdahmane@gmail.com

Abstract

This paper presents our approach and findings for Subtask 1 (Authorship Style Transfer) of the AraGenEval2025 shared task. We explore methods to transform neutral Arabic text into the distinctive style of a specified author while preserving its original meaning. Our work details a two-phase development: an initial baseline model leveraging few-shot prompting with Gemini and K-means clustering, followed by fine-tuning of pre-trained seq2seq models that support Arabic, including representatives from the mT5 and mBART model families. We evaluated our models using BLEU and chrF metrics, demonstrating significant improvements in fine-tuning, particularly in capturing Arabic-specific stylistic nuances. To complement these surface-level overlap metrics, we incorporate BERTScore to assess semantic preservation across style transfer. Additionally, we introduce a style classifier to quantify author-specific style transfer strength. We discuss the challenges encountered, including Arabic linguistic complexities, handling long Arabic text, and hardware constraints, and outline future directions for enhancing Arabic Authorship Style Transfer.

1 Introduction

The proliferation of digital content requires advanced natural language processing (NLP) techniques for text manipulation. Authorship style transfer (AST) is a challenging yet key task aiming to convert a given text into the writing style of a target author while maintaining its semantic content. This differs from traditional stylistic analysis, focusing solely on identifying and characterizing an author's style. The increasing sophistication of AI-generated content, particularly in Arabic, further highlights the need for robust AST models, as style identification can contribute to detecting synthetic texts.

Despite its importance, Arabic AST remains a relatively underexplored area compared to other languages. The Arabic language presents unique linguistic challenges, including significant morphological variations, rich affixation, diverse dialects, and complex reordering phenomena, all of which impact style transfer. Furthermore, the scarcity of large-scale labeled datasets for Arabic AST poses a significant hurdle. This complexity is further exacerbated by the high inflectional nature of Arabic, which introduces tokenization difficulties, especially when dealing with long texts and paragraph-level inputs.

The AraGenEval2025 shared task, hosted with the Arabic Natural Language Processing (ArabicNLP 2025) Conference (Abudalfa et al., 2025), aims to foster research in this domain. Our participation focuses on Subtask 1: Authorship Style Transfer, where the objective is to transform a formal input text into a specified author's style. This paper details our methodology, experimental setup, evaluation, and the insights gained throughout the project, and concludes with perspectives for future works.

Our system entails a two-stage strategy: an initial baseline using few-shot prompting with *Gemini*, supported by K-means clustering, followed by fine-tuning of Arabic-supporting seq2seq models from the *mT5*, *AraT5*, and *mBART* families. The resulting models achieved **24.46%** and **59.33%** in BLEU and chrF, respectively, reflecting word- and character-level surface overlap with reference texts. Meaning preservation across style transfer was measured at **92.01%** using *BERTScore*. The stylistization precision per author reached **86.12%**, as assessed using the style classifier. Implementation is available at[1].

2 Background

While Arabic AST remains relatively underexplored, two recent approaches (Shao et al., 2024) and (Hu et al., 2022) provide valuable foundations. Both generate pseudo-parallel neutral↔stylized pairs using GPT and fine-tune a seq2seq model on sentence-level data. (Shao et al., 2024) focuses on general purpose style transfer and has been applied to well-defined styles such as Shakespeare, rap lyrics, and Chinese literature. It leverages

[1] https://github.com/nojoom-ai/AraGenEval2025

Proceedings of The Third Arabic Natural Language Processing Conference, pages 18–25

English- and Chinese-centric tokenizers and pre-trained BART models. Stylized samples are selected using K-means clustering and augmented bidirectionally to train a BART-based model.(Hu et al., 2022), on the contrary, targets a few-shot style transfer with low-resource authors. It applies GPT-based neutralization followed by supervised fine-tuning and introduces a reward model to guide output refinement through preference-based policy optimization.

Despite their strengths, both approaches are limited to short-form inputs, rely heavily on English-centric infrastructure, and employ evaluation setups that do not account for Arabic's morphological complexity or long-form stylistic variation. Our work addresses these limitations by extending AST to paragraph-level Arabic inputs, explicitly managing tokenization challenges caused by high inflection. We fine-tune Arabic-supporting seq2seq models and propose a broader evaluation protocol, inspired by (Shao et al., 2024) and (Hu et al., 2022).

3 System Overview

The system comprises two stages, inspired by the (Shao et al., 2024) and (Hu et al., 2022) approaches. First, we develop a baseline model that serves as a reference for comparison (see Fig. 2). Next, we fine-tune several Arabic-supporting pre-trained models.

3.1 Baseline Model

Our initial approach utilizes few-shot prompting with *Gemini 2.5 Flash*. The process involves:

- **K-means Clustering** (Fig. 2 Step 1.a): We performed exploratory data analysis (EDA) on embedding representations of training samples, using the elbow method and silhouette scores to determine that $k = 2-3$ clusters are optimal for most authors. We then applied K-means to select the top $K = 3$ representative examples per author.
- **Prompt Construction**: We construct a prompt by concatenating the selected exemplars with the neutral input text.
- **Styled Output Generation** (Fig. 2 Step 1.b): *Gemini 2.5 Flash* generates the stylized output based on the constructed prompt.

3.2 Pre-trained Models Fine-Tuning

To address the limitations of the few-shot baseline, we implemented a fine-tuning pipeline for pre-trained seq2seq models (phase 2):

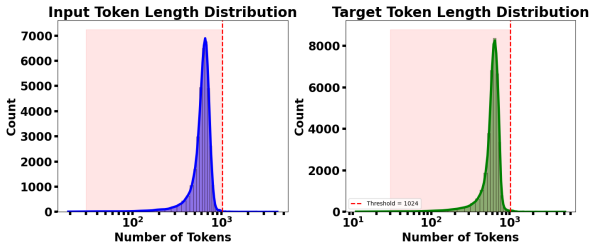

Figure 1: Token-length distributions for training dataset input (blue) and target (green).

%	In	Tgt	%	In	Tgt
0	19	11	95	781	765
5	433	419	98	822	825
11	509	501	99	870	934
50	644	635	Q3+1.5·IQR	877	864
90	748	735	100	4248	5094

Table 1: Training Set Input and target token-length statistics. Q3+1.5·IQR indicates the statistical outlier upper threshold.

- **Input Preparation**: For each training sample, we prepend an author tag to the neutral text. The corresponding stylized text is used as the target sequence.
- **Tokenization** (Fig. 2 Step 2.a): Arabic morphology is highly inflected and rich in prefixes and suffixes, resulting in a higher subword token count per word compared to English. (Rust et al., 2021) shows that Arabic typically yields 1.1–1.8 subword tokens per word, compared to 1.2–1.3 in English. Since VRAM usage scales roughly with the square of sequence length, we selected our token-length caps to balance dataset coverage and hardware constraints.

  We analyze token-length distributions across training and validation sets using the *mBART50* tokenizer (Fig. 1). A maximum length of 750 tokens covers $\approx$ 90% of the samples, while 1024 tokens cover $\approx$ 99.6% (see Table 1). The final tokenization limits were chosen based on the available hardware and pre-trained model sizes.
- **Fine-Tuning** (Fig. 2 Step 2.b): The pre-trained model weights (*mT5*, *AraT5*, *mBART*) were fine-tuned on the prepared dataset, with intermediate checkpoints saved to handle long training sessions.
- **LoRA Injection** (Fig. 2 Step 2.c): To improve

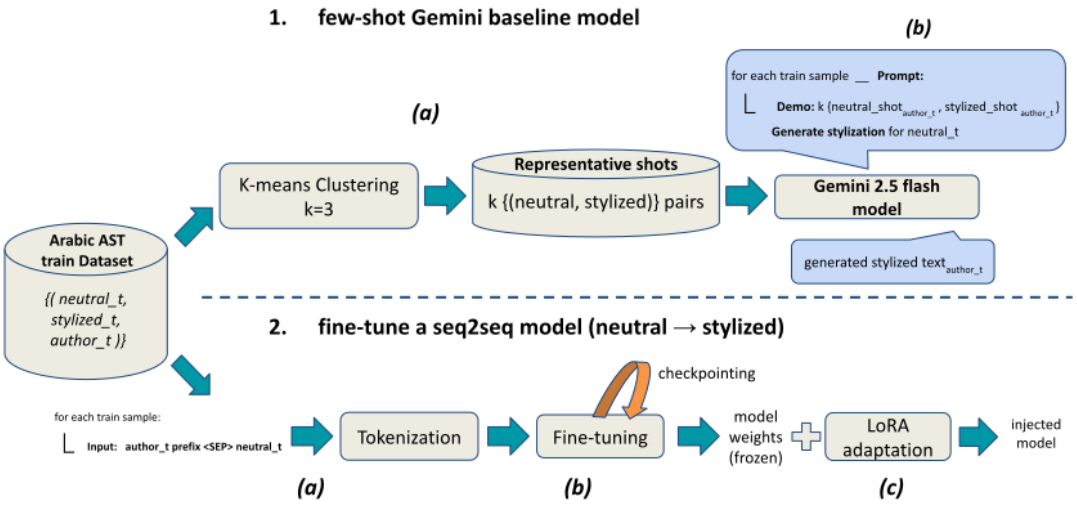

Figure 2: Arabic AST Model developement pipeline

performance under hardware constraints, we injected Low-Rank Adaptation (LoRA) modules (Hu et al., 2021) into the fine-tuned models and conducted additional training on the training dataset.This enabled further optimization over more epochs while keeping the base model weights frozen.

4 Experimental Setup

4.1 Data Splits

We use the official *AraGenEval2025* dataset, consisting of 35,122 paragraph-level samples for training (72.1%), 4,157 for validation (8.5%), and 8,143 for testing (19.3%). The test set labels are withheld by the organizers and used only for final evaluation. Tokenized input lengths reach up to 3,361 tokens, with 99.66% of samples under 1,024 tokens (Fig. 4).

4.2 Preprocessing

Each neutral input is prepended with an author tag in the format: `<AUTHOR> | <NEUTRAL_TEXT>`. Tokenization is performed using the corresponding *AutoTokenizer* for each model.

4.3 Hardware and Environments

All experiments were conducted on cloud-based platforms with varying GPU configurations; full details are provided in Appendix B.1.

4.4 Evaluation Metrics

We report the two official competition metrics - **BLEU** and **chrF** to assess word- and character-level surface overlap. In addition, we include two complementary metrics: **BERTScore (BS)**, for measuring semantic preservation, and **Style Classifier Accuracy (SC)**, to assess author-specific style transfer strength. Details of the style classifier development are provided in Appendix B.6.

5 Results

This section presents the empirical evaluation of our AST models, detailing their performance across various metrics, and providing per-author insights. Our models were evaluated on validation dataset. The best performing models were then used on the final test data set evaluation.

5.1 Overall Performance Comparison

Table 5 summarizes the performance of the Few-Shot baseline and various fine-tuned models. Overall, fine-tuning yields substantial gains: BLEU improves from 11.66 to 24.46 ($\Delta = +11.26$) and chrF from 48.12 to 59.33 ($\Delta = +11.21$), confirming improved stylistic alignment. BS remains consistently high ($\sim$ 0.91–0.93), indicating strong meaning preservation across models. SC aligns well with other metrics, supporting its usefulness in quantifying stylistic strength.

Among the models evaluated, *Facebook/mbart-large-mmt-50* attains the highest validation BLEU and chrF, while `UBC-NLP/AraT5-v2-1024` is highly competitive in both validation and test results given its parameter weight. LoRA injection on `UBC-NLP/AraT5-v2-1024` yielded modest gains where applied; overall improvements are primarily attributable to fine-tuning.

Although the gains are clear, chrF scores in the high 50s suggest remaining challenges in capturing

Validation Set Results				
Model	**BLEU**	**chrF**	**BS**	**SC**
Few-Shot Baseline	11.66	48.12	91.25	58.43
google/mt5-small	18.51	52.92	91.88	59.78
UBC-NLP/AraT5-base	21.24	57.13	92.02	62.20
agemagician/mlong-t5-tglobal-large	23.58	58.88	93.01	73.58
facebook/mbart-large-mmt-50	**24.56**	**59.92**	92.01	**85.86**
moussakam/AraBART	21.76	58.21	92.52	58.67
UBC-NLP/AraT5-v2-1024	23.80	59.27	91.63	73.90

Table 2: Validation set results for evaluated models.

Test Set Results			
Model	**BLEU**	**chrF**	**SC**
facebook/mbart-large-50	24.46	59.33	86.18
moussakam/AraBART	21.07	57.21	59.12
UBC-NLP/AraT5-v2-1024	24.07	59.48	74.31
UBC-NLP/AraT5-v2-1024 **+ LoRA**	24.22	59.53	75.42

Table 3: Test set results for selected models. LoRA was injected only where indicated.

Author	**Cnt**	**BLEU**			**chrF**			**Author**	**Cnt**	**BLEU**			**chrF**		
		B	**FT**	**Δ**	**B**	**FT**	**Δ**			**B**	**FT**	**Δ**	**B**	**FT**	**Δ**
A. G. Makawi	396	17.16	31.48	+14.32	55.07	66.64	+11.57	Ahmed Amin	246	9.47	18.77	+9.30	47.09	57.09	+10.00
Fouad Zakaria	125	17.10	27.02	+9.92	54.27	62.62	+8.35	A. M. Al-Aqqad	267	8.67	17.89	+9.22	44.76	54.15	+9.39
Naguib Mahfouz	327	15.21	25.49	+10.28	50.66	59.60	+8.94	Salama Moussa	119	8.05	14.53	+6.48	44.51	53.95	+9.44
Jurji Zaydan	327	14.39	21.48	+7.09	52.24	59.15	+6.91	Yusuf Idris	120	7.48	17.71	+10.23	42.79	55.08	+12.29
Robert Bar	82	13.75	19.16	+5.41	49.90	54.02	+4.12	**G. K. Gibran**	30	7.18	27.87	+20.69	45.35	61.44	+16.09
Tharwat Abaza	90	12.96	27.71	+14.75	50.15	59.93	+9.78	M. H. Heikal	260	6.07	14.31	+8.24	42.84	52.21	+9.37
Hassan Hanafi	548	12.93	25.04	+12.11	48.59	61.20	+12.61	Taha Hussein	255	5.68	14.54	+8.86	42.12	51.59	+9.47
Amin Al-Rihani	142	12.65	21.62	+8.97	51.12	59.93	+8.81	A. Teimur Pasha	57	3.76	17.74	+13.98	30.53	46.39	+15.86
W. Shakespeare	238	11.35	26.21	+14.86	48.08	61.02	+12.94	Kamel Kilani	25	2.43	13.38	+10.95	34.03	50.64	+16.61
N. El Saadawi	295	10.83	29.77	+18.94	48.28	65.90	+17.62	**Ahmed Shawqi**	58	1.91	19.34	+17.43	37.72	55.49	+17.77
Gustave Le Bon	150	9.60	18.60	+9.00	48.96	59.05	+10.09	**Overall**	4157	11.66	22.92	+11.26	48.12	59.13	+11.01

Table 4: Per-author performance comparison of the fine-tuned `UBC-NLP/AraT5-v2-1024` vs. the baseline models.

Arabic's morphological richness. These results emphasize the importance of both model architecture and input processing for effective style transfer.

5.2 Per-Author Insights

To gain deeper insights, we analyze per-author performance of the fine-tuned *UBC-NLP/AraT5-v2-base-1024* model (367M parameters) against the baseline. We chose it because of its strong performance compared to *mBART-large-50-mmt* at lower parameter cost, and because it better handles long inputs (full-sample tokenization); see Appendix A and §B.1. Table 4 reports BLEU and chrF per author with absolute changes (Δ).

The analysis shows consistent gains across most authors. Notable examples include *Gibran Khalil Gibran* (30 samples), which exhibits the largest increase ($\Delta_{\text{BLEU}} = +20.69$, $\Delta_{\text{chrF}} = +16.09$); *Ahmed Shawqi* (58 samples) also shows strong improvements ($+17.43$, $+17.77$); and *Nawal El Saadawi* (295 samples) with substantial gains ($+18.94$, $+17.62$). Overall, the model achieves a sizable overall uplift (BLEU $\uparrow$ 11.26, chrF $\uparrow$ 11.01), demonstrating that AraT5-v2-1024 effectively captures author-specific stylistic signals while handling longer inputs, and may surpass the model *mBART-large-50-mmt*, if a considerable share of long inputs (> 1024 tokens) were present in the evaluation sets.

Conclusion

Our participation in Subtask 1 of AraGenEval2025 demonstrates effective Authorship Style Transfer for Arabic. Building on a few-shot Gemini 2.5 Flash with shots selection through K-means clustering baseline, we fine-tuned arabic-supporting seq2seq models, achieving 24.46% BLEU, 59.33% chrF, 92.3% BS and 86% SC. Per-author results were consistently strong, with the lightweight *UBC-NLP/AraT5-v2-1024* (367 M parameters) matching or exceeding larger multilingual models, underscoring the value of Arabic-specific pre-training.

We identified several Arabic AST challenges , including rich morphology and affixation, dialectal variation, reordering, and long paragraph inputs. We tackled long training on limited hardware by injecting LoRA modules and using token-budgeted batching with CPU/GPU overlap to respect hardware limits while processing extended contexts.

Although chrF improvements indicate further room for capturing fine-grained character-level nuances, our approach lays a solid foundation. Future work will explore longer inputs handling, and integrate human-in-the-loop evaluation (e.g., Gemini judgment) to further enhance stylistic fidelity.

6 Acknowledgments

The authors gratefully acknowledge the invaluable support and resources provided by the ***Nojoom.AI*** team. Their dedication, technical guidance, and infrastructure were instrumental in the successful execution of this research and our participation in the *AraGenEval2025* shared task. We extend our sincere gratitude for their continuous encouragement and collaborative spirit.

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.

Moussa Kamal Eddine, Nadi Tomeh, Nizar Habash, Joseph Le Roux, and Michalis Vazirgiannis. 2022. Arabart: a pretrained arabic sequence-to-sequence model for abstractive summarization. arXiv preprint arXiv:2203.10945.

AbdelRahim Elmadany, El Moatez Billah Nagoudi, and Muhammad Abdul-Mageed. 2023. Octopus: A multitask model and toolkit for Arabic natural language generation. In *Proceedings of ArabicNLP 2023*, pages 232–243, Singapore (Hybrid). Association for Computational Linguistics.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint*, arXiv:2106.09685.

Zhiqiang Hu, Roy Ka-Wei Lee, Charu C. Aggarwal, and Aston Zhang. 2022. Text style transfer: A review and experimental evaluation. *ACM SIGKDD Explorations Newsletter*, 24(1):14–45.

El Moatez Billah Nagoudi, AbdelRahim Elmadany, and Muhammad Abdul-Mageed. 2022. Arat5: Text-to-text transformers for arabic language generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 628–647.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. 2021. How good is your tokenizer? on the monolingual performance of multilingual language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135, Online. Association for Computational Linguistics.

Zhonghui Shao, Jing Zhang, Haoyang Li, Xinmei Huang, Chao Zhou, Yuanchun Wang, Jibing Gong, Cuiping Li, and Hong Chen. 2024. Authorship style transfer with inverse transfer data augmentation. *AI Open*, 5:94–103.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning.

David Uthus, Santiago Ontañón, Joshua Ainslie, and Mandy Guo. 2023. mlongt5: A multilingual and efficient text-to-text transformer for longer sequences. arXiv preprint arXiv:2305.11129.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, pages 483–498.

A Appendix: Dataset Distribution Details

In seq2seq tasks, setting an appropriate maximum input length during tokenization is critical for reliable evaluation. Truncating long inputs can degrade performance by removing key information, especially for stylistic tasks that rely on paragraph-level context.

The tables and plots in this appendix provide a detailed overview of the input and target token length distributions for the validation and test sets. These statistics were used to determine safe maximum input lengths that cover at least 99% of the samples, ensuring high coverage without excessive memory consumption. Outlier thresholds based on the $Q3{+}1.5{\cdot}IQR$ rule are also reported to highlight extreme cases.

A.1 Validation Set Token-Length Distribution

%	In	%	In
0	21	75	693
5	432	90	736
10	500	95	768
25	574	Q3+1.5·IQR	872
50	639	100	1216

Table 5: Validation set input token-length statistics. Q3+1.5·IQR indicates the statistical outlier upper threshold.

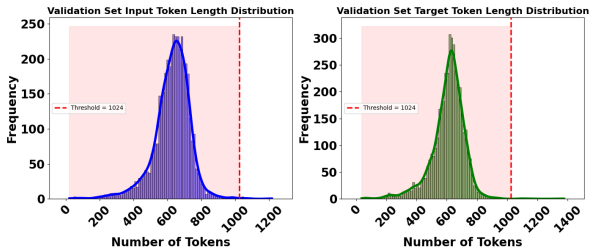

Figure 3: Token-length distributions for validation dataset input (blue) and target (green).

A.2 Test Set Input Token-Length Distribution

It is important to note that different model architectures impose different maximum input length constraints. **mBART-based models** such as `facebook/mbart-large-50-mmt` and `moussakam/AraBART` enforce a *hard limit* of 1,024 tokens due to their absolute positional embeddings. In contrast, **T5-based models** such as `google/mt5-small`, `UBC-NLP/AraT5-base`, and `UBC-NLP/AraT5-v2-1024` utilizes relative positional embeddings, which allow a *soft limit*—they can accept longer sequences as long as the available hardware permits.

%	In	%	In
0	30	75	702
5	433	95	747
10	514	99	855
25	587	Q3+1.5·IQR	877
50	650	100	3361

Table 6: Test set input token-length statistics. Q3+1.5·IQR indicates the statistical outlier upper threshold.

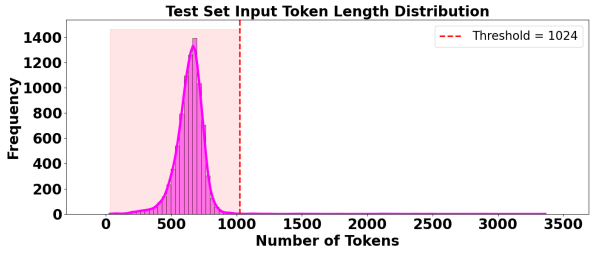

Figure 4: Token length distribution for test set inputs. Over 99.6% of samples fall under 1 024 tokens.

As shown in Table 7, the maximum input lengths used during training and evaluation were configured based on these architectural constraints and the available computing resources. For T5-based models, we set input length limits to 750 or 1,024 tokens to safely cover most validation and test samples without truncation.

B Appendix: Experimental Details

B.1 Model and Environment Details

Table 7 summarizes the models used, their parameter sizes, token length limits, and compute environments. T5-based models tolerate flexible input lengths (hardware permitting), while mBART-based models impose a strict 1024-token cap. Training was conducted on either Colab Pro+ (A100) or Kaggle (P100). CPU-only runs were reserved for small-scale evaluation like ChrF , BLEU and BERTScore calculations due to memory limitations.

B.2 K-means Clustering for Few-Shot Samples Selection

To avoid suboptimal or noisy few-shot examples resulting from random selection, we apply clustering

Model	Params	Platform	Accel.	Training		Evaluation Max tok.	
				unit BS	Max tok.	Validation	Test
`google/mt5-small` (Xue et al., 2021)	310M	Kaggle	P100	1	750	750	/
`UBC-NLP/AraT5-base`(Nagoudi et al., 2022)	280M	Kaggle	P100	1	750	1500	/
`agemagician/mlong-t5-tglobal-large` (Uthus et al., 2023)	1768M	Colab Pro+	A100	4	1024	1500	/
`facebook/mbart-large-50-mmt` (Tang et al., 2020)	610M	Colab Pro+	A100	8	1024	1024	1024
`moussakam/AraBART` (Eddine et al., 2022)	139M	Kaggle	P100	16	1024	1024	1024
`UBC-NLP/AraT5-v2-1024` (Elmadany et al., 2023)	367M	Colab Pro+	A100	12	1024	1500	3500

Table 7: Compute platforms and sequence-length configurations across dataset splits.

of K-means on sentence embeddings to deterministically select representative neutral samples per author. The goal is to ensure that stylistically central examples are used in prompt-based evaluation, without model fine-tuning.

We encode each author's neutral training texts using the `all-MiniLM-L6-v2` model, then cluster the resulting embeddings and extract the closest samples to each cluster centroid as the selected few-shot examples.

Parameter	Value / Setting
Embedding model	`all-MiniLM-L6-v2`
Embedding dimension	384
Clustering method	K-means (per author)
Number of clusters (k)	3
Distance metric	Euclidean
Selection criterion	Centroid-nearest samples
Random seed	42

Table 8: K-means clustering setup for representative few-shot selection.

B.3 Training Configuration

Key hyperparameters (defaults unless otherwise noted):

Parameter	Value
Effective batch size	32
Gradient accumulation steps	8
Max sequence length	750 / 1024
Checkpoint interval	500 steps
Epochs	3
Optimizer	AdamW
Learning rate	5×10^{-5}

Table 9: Summary of training hyperparameters.

B.4 Evaluation Configuration

Inference is performed via a single-GPU, token-budgeted batching pipeline that overlaps CPU tokenization with GPU generation to maximize throughput and avoid OOMs. Inputs are first sorted by length on the CPU, grouped into batches whose total token count does not exceed a configurable budget, then transferred to the GPU for generation. If an OOM occurs, the budget is halved and the batch is retried in smaller splits.

Key parameters are summarized in Table 10.

Parameter	Value / Description
Token budget	10 000 total input tokens
VRAM Memory threshold	80 % of GPU VRAM
Budget increment	+1 000 tokens when VRAM<VRAM_THRESH
Budget update frequency	every 5 successful batches
Max input length	3 400 tokens (capped by model input handling)
Max generation length	4 000 tokens (capped by model input handling)

Table 10: Key settings for token-budgeted inference

This setup ensures that: (1) very long inputs are safely handled without silent truncation, and (2) GPU utilization remains high by feeding pre-tokenized batches as soon as memory permits.

B.5 LoRA Configuration

To enable lightweight and fast adaptation over limited resources, LoRA was injected into attention layers of a frozen *UBC-NLP/AraT5-v2-1024* base. This setup drastically reduces trainable parameters, making hyperparameter sweeps and multi-run experimentation feasible within constrained GPU environments. We used an aggressive injection configuration with moderately high rank and scaling

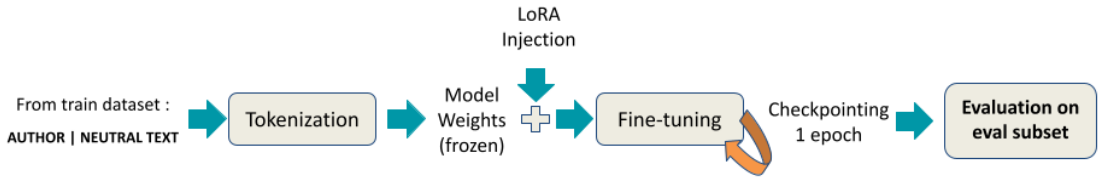

Figure 5: LoRA injection Development & Evaluation pipeline

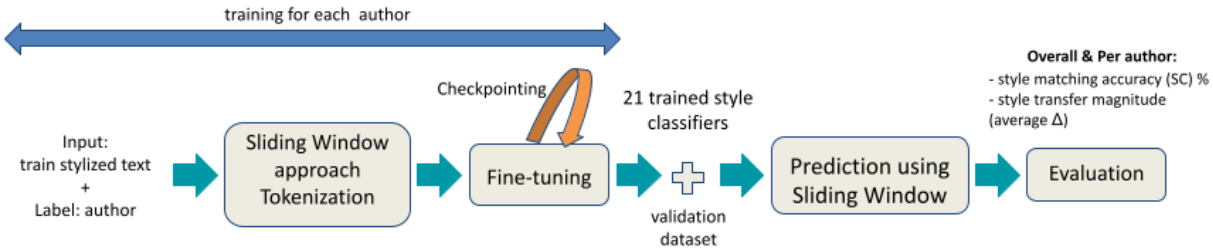

Figure 6: Arabic Style classifier Development & Evaluation pipeline

values. Checkpoints were saved each epoch, and the model with the best chrF score on a held-out validation subset was selected.

Component	Configuration
Base model	`UBC-NLP/AraT5-v2-1024`
Target modules	`q, v, k, fc1, fc2`
Injection layers	Encoder and decoder attention
Rank (r)	32
Scaling factor (α)	64
Dropout	0.1
Bias	None
Epochs	5
Eval subset	25% subset stratified from validation set
Checkpointing	Every epoch
Final model selection	Best checkpoint by chrF

Table 11: Summary of LoRA fine-tuning configuration.

B.6 Style Classifier

While BLEU and chrF quantify surface overlap, they do not directly measure whether the generated text truly mirrors an author's stylistic fingerprint. To address this, we train an author-specific binary classifier, based on `bert-base-arabic-camelbert-ca`, that learns the distinctive phrasing, vocabulary, and structural patterns of each author.

Unfortunately, no off-the-shelf Arabic style classifier supports long inputs beyond 512 tokens. Our options were to pre-train an English long-document model (e.g. Longformer) on Arabic data or to adopt a sliding-window approach. As shown in Fig. 6, we chose the latter: inputs are split into overlapping 512-token chunks (256-token stride), each classified separately, and results are aggregated. This ensures we capture stylistic cues from long paragraphs without truncation.

For each sample evaluated (from validation or test datasets), we compare the confidence of the classifier in the 'Author X' class on the neutral input versus the stylized output to calculate

$$\Delta = p_{\text{out}}(1) \; - \; p_{\text{in}}(1).$$

An instance is a *hit* if $\Delta > 0$, i.e. the generated output aligns more with the 'Author X' style than with the neutral text (that is,a successful style transfer). We report the hit rate as the SC metric.

Parameter	Setting
Base model	`bert-base-arabic-camelbert-ca`
Input length limit	512 tokens (sliding window)
overlap	256 tokens between chunks
Training epochs	5
Batch size	16
Learning rate	2×10^{-5}
Optimizer	AdamW
Scheduler	Linear warmup
Loss	Binary cross-entropy
Output metrics	Hit rate ($\Delta > 0$), mean Δ
Classifiers	One per author (21 total)

Table 12: Training Setup for each author style classifiers.

Future work should explore pre-training or adapting a native Arabic long-input classifier, rather than relying on sliding windows, to more seamlessly handle long input LLM generations evaluation.