Science, Engineering and Technology Vol. 5, No. 1, pp. 55-65
www.setjournal.com *https://doi.org/10.54327/set2025/v5.i1.224*

Fine-tuning AraGPT2 for Hierarchical Arabic Text Classification

Djelloul BOUCHIHA[1], Abdelghani BOUZIANE[1], Noureddine DOUMI[2], Benamar HAMZAOUI[1]

[1]*Department of Computer Science, Institute of Science, University Centre of Naama, EEDIS Lab., UDL-SBA, Algeria.*
[2]*Department of Computer Science, University of Saida, Algeria.*

Abstract

Text classification consists in attributing a text to its corresponding category. It is a crucial task in natural language processing (NLP), with applications spanning content recommendation, spam detection, sentiment analysis, and topic categorization. While significant advancements have been made in text classification for widely spoken languages, Arabic remains underrepresented despite its large and diverse speaker base. Another challenge is that, unlike flat classification, hierarchical text classification involves categorizing texts into a multi-level taxonomy. This adds layers of complexity, particularly in distinguishing between closely related categories within the same super-class. To tackle these challenges, we propose a novel approach using AraGPT2, a variant of the Generative Pre-trained Transformer 2 (GPT-2) model adapted specifically for Arabic. Fine-tuning AraGPT2 for hierarchical text classification leverages the model's pre-existing linguistic knowledge and adapts it to recognize and classify Arabic text according to hierarchical structures. Fine-tuning, in this context, refers to the process of training a pre-trained model on a specific task or dataset to improve its performance on that task. Our experiments and comparative study demonstrate the efficiency of our solution. The fine-tuned AraGPT2 classifier achieves a hierarchical HF score of 80.64%, outperforming the machine learning-based classifier, which scores 41.90%.

Keywords*: Arabic Text Classification, Hierarchical Classification, AraGPT2, GPT-2 Fine-tuning, Generative Pre-trained Transformer, Large Language Model.*

1. Introduction

Text classification, a crucial subtask of natural language processing (NLP), involves assigning texts to predefined categories. This process underpins many applications, including sentiment analysis, spam detection, content recommendation, and topic categorization [1]. Progress in artificial intelligence (AI), especially through the creation of large language models (LLMs), has greatly improved the effectiveness of text classification. Recent studies have demonstrated the efficiency of LLMs in various text classification settings, particularly through transfer learning, where a pre-trained model is adapted to a new domain or task [2], [3]. LLMs, such as GPT-2, leverage deep learning to process and generate human-like text, making them powerful tools for various NLP tasks [4].

Large Language Models (LLMs) mark a significant advancement in AI, distinguished by their capacity to understand and produce text with a high degree of relevance and coherence [5]. GPT-2 (Generative Pre-trained Transformer 2), an LLM developed by OpenAI, is pre-trained on extensive text data and can be fine-tuned for particular tasks [5]. AraGPT2 is a specialized adaptation of GPT-2 tailored for the Arabic language, allowing it to handle the unique linguistic features and nuances of Arabic [6]. Deep learning, the backbone of these models, involves training neural networks with many layers to learn complex patterns in data, which is

Corresponding author: *Djelloul BOUCHIHA (bouchiha@cuniv-naama.dz)*
Received: *15 October 2024*; Revised: *4 March 2025*; Accepted: *11 March 2025*; Published: *17 March 2025*

crucial for the success of LLMs in various NLP applications [7].

This paper focuses on Arabic language processing due to the notable lack of resources and tools for Arabic NLP, despite its extensive and diverse speaker base. Arabic's rich morphology, diverse dialects, and unique script present additional challenges that require specialized approaches [8]. The hierarchical classification of Arabic texts adds further complexity, as it involves navigating nested categories with subtle differences, especially within the same super-class.

In this paper, we investigate the application of GPT-2 for hierarchical classification of Arabic text. Hierarchical classification involves categorizing texts into multi-level taxonomies, which is more complex than flat classification due to the need to distinguish between closely related categories under the same parent category. To address these challenges, we fine-tune AraGPT2, adapting its pre-existing knowledge to accurately classify Arabic texts within a hierarchical structure. Fine-tuning involves adjusting a pre-trained model by training it on a specific dataset or task to enhance its performance for that specific use case. Our experimental results demonstrate that fine-tuning AraGPT2 for hierarchical classification substantially outperforms traditional machine learning-based methods.

This paper is organized as follows: Section 2 reviews related work in Arabic text classification and hierarchical classification methods. Section 3 details the methodology, including the used dataset, the system architecture, and the fine-tuning process. Section 4 presents the experiments and results, followed by a comparative study in Section 5. Finally, Section 6 wraps up the paper and proposes areas for future research.

2. Literature Review

In this section, we examine the literature on Arabic text classification, hierarchical text classification, and the use of large language models (LLMs) in NLP. We highlight previous work and methodologies relevant to our study and position our approach in the context of existing research.

2.1. Arabic text classification

A significant effort in this domain is by El-Halees, who surveyed various machine learning techniques, including Decision Trees and Support Vector Machines (SVMs), applied to Arabic text classification (ATC), and discussed the specific challenges posed by the Arabic language [9]. While methods have shown effectiveness, they often require extensive feature engineering to address the rich morphology and syntax of Arabic. Notably, none of the works cited in this survey deal explicitly with hierarchical Arabic text classification; all of them are flat classification solutions.

Deep learning models have been progressively utilized for Arabic text classification. For example, AraBERT, a transformer-based model specifically pre-trained for Arabic, has shown substantial improvements over traditional methods [10]. AraBERT leverages the transformer architecture to capture the syntactic and semantic nuances of Arabic, leading to better performance in tasks like named entity recognition. However, AraBERT has not yet been explicitly used for hierarchical Arabic text classification.

Wahdan et al. [11] conducted a systematic review to provide a thorough overview of the current advancements in Arabic text classification, emphasize current challenges and explore major trends in large-scale research. Among 2875 studies, 60 met the eligibility criteria and were widely analyzed. Notably, none of these studies focused on fine-tuning GPT for Arabic text classification.

2.2. Hierarchical text classification

Hierarchical text classification (HTC) involves organizing documents into a hierarchical taxonomy, which is more complex than flat classification due to the need to handle multiple levels of categories. HTC is particularly useful in domains where documents are naturally organized in a hierarchy, such as news articles, medical records, and product categories [12].

Early approaches to HTC used extensions of flat classification algorithms, such as hierarchical versions of SVMs or decision trees [12]. However, these methods struggled with the complexity of deep hierarchies and the interdependencies between categories.

Zangari et al. [13] offer a comprehensive overview of recent research in hierarchical text classification. They begin by situating the task within the larger field of text classification and discussing key concepts such as text representation. Following this, they describe traditional methods and review contemporary approaches, emphasizing their key contributions. They also provide statistics on commonly used datasets and evaluate the advantages of metrics designed for hierarchical contexts. Lastly, they compare recent methods to non-hierarchical baselines using five publicly available domain-specific datasets.

In the realm of the Arabic language, hierarchical text classification (HTC) has received less attention than flat classification. Most research has focused on flat classification, with few studies addressing the specific challenges of HTC in Arabic. Aljedani et al. [14] effectively addressed the HTC challenge by introducing a hierarchical multi-label Arabic text categorization (HMATC) model that integrates machine learning techniques. Although some other studies employed multi-labeling for texts, they did not specifically address Arabic hierarchical text classification (HTC) [15] [16] [17] [18] [19]. The complexity of Arabic's morphology and syntax, combined with the hierarchical structure of documents, makes HTC a particularly challenging problem in this language.

2.3. Large Language Models and GPT

Large Language Models (LLMs) are sophisticated AI models that understand and produce human-like text by identifying patterns within vast datasets [5]. Characterized by billions of parameters and typically using transformers [20], this architecture enables them to capture long-range dependencies in text, making them especially effective for tasks that need understanding context and sequence. LLMs are pre-trained on vast corpora to grasp general language patterns and fine-tuned for specific tasks, making them powerful tools in natural language processing applications. Examples of LLMs include GPT-2 [5] and GPT-4 [21].

LLMs have demonstrated exceptional performance across a variety of NLP tasks, such as annotation [22-25], data generation [26], and Text-to-SQL [27]. Notably, they excel in flat text classification using zero-shot or few-shot prompting [28]. However, their use in hierarchical contexts with extensive and structured label spaces continues to pose challenges, even for the English language. To address this, Zhang et al. [29] introduced TELEClass, a method for taxonomy enrichment and employing LLMs in weakly-supervised hierarchical text classification. TELEClass enhances the label taxonomy with terms indicative of specific classes to improve classifier training and employs LLMs to annotate and create data, specifically tailored to the hierarchical label structure.

Leveraging LLMs, we propose in this paper to fine-tune AraGPT2, an adaptation of GPT-2, for the hierarchical classification of Arabic texts. AraGPT2 is pre-trained on a vast corpus of Arabic text, enabling it to manage the linguistic features and nuances specific to Arabic. Fine-tuning is an essential method for adapting LLMs to specific tasks. It entails training a pre-trained model on a smaller, task-specific dataset to enhance its performance. This process enables the model to utilize its existing knowledge while adapting to the unique requirements of the task.

2.4. Comparative studies

Comparative studies in text classification often highlight the advantages and limitations of different approaches. Philip et al. [30] carried out a study comparing selective ML algorithms for analyzing English text documents. Their results showed that the SVM-based classifier performed best with a small feature set, while NB excelled with larger sets. Lee et al. [31] compared multiclass text classification in research proposals by using pre-trained Korean language models, finding that a BERT-based model trained on the latest Korean corpus achieved the highest performance. Koksal and Akgul [32] performed a comparative study on Turkish text classification, demonstrating substantial performance enhancements through the use of word embeddings with deep learning algorithms and optimized hyperparameters.

For the Arabic language, Berrimi et al. [33] conducted a comparative study on effective methods for flat Arabic text classification, finding that a Transformer-CNN architecture outperforms other neural network models. Bouchiha et al. [34] performed a comparative empirical study to identify the best combination of feature

extraction and machine learning algorithms for flat Arabic text classification, revealing that the TFIDF-Perceptron combination is the top performer among 160 classifiers. Melhem et al. [35] reviewed various published studies aimed at enhancing Arabic text classification and identified research opportunities in the field. Their findings indicated that Naive Bayes and SVM are the most commonly used classifiers for flat Arabic text classification.

The proposed approach in this paper, which involves fine-tuning AraGPT2 for hierarchical Arabic text classification, builds on these advancements. By adapting the pre-trained model to the specific requirements of hierarchical classification, we aim to tackle the difficulties arising from the complexity of the Arabic language and the hierarchical structure of the classification task.

3. Methodology: Fine-tuning AraGPT2

In this section, we detail our method to fine-tuning AraGPT2 for hierarchical Arabic text classification. This process is structured into four main layers (see Figure 1): Data preparation, Model, Training, and Evaluation.

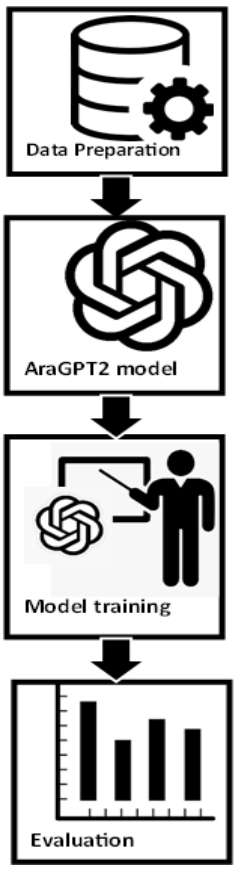

Figure 1. Fine-tuning AraGPT2.

3.1. Data preparation layer

The data preparation layer is fundamental for setting up the inputs required for fine-tuning. This layer involves two crucial steps:

- **Dataset input**: We start by reading the dataset containing Arabic texts along with their associated hierarchical labels, which include both category and super-category. These hierarchical labels are vital for understanding the text at different levels of granularity.
- **Train-Test split**: The dataset is divided into training and testing sets. This division enables us to train the model on one part of the data while assessing its performance on a separate, unseen portion.

3.2. Model layer

In the model layer, we focus on leveraging AraGPT2 [6], a pre-trained model tailored for Arabic language tasks. AraGPT2 is built on the GPT-2 architecture and is fine-tuned on a vast corpus of Arabic text. This model uses a tokenizer that converts Arabic text into tokens that the model can process effectively. Utilizing a pre-trained model like AraGPT2 provides a strong foundation of linguistic knowledge, significantly enhancing the model's performance on Arabic text.

3.3. Training layer

The training layer is where we adapt the pre-trained AraGPT2 model to our hierarchical classification task:

- **Training configuration**: We define the configuration for the training process, including parameters like the number of training epochs, the batch size, and the learning rate. These parameters control how the model learns from the training data (more details in section 4).
- **Dataset preparation for training**: The text data is formatted into a suitable structure for training and evaluation. This preparation guarantees that the model can effectively process and learn from the training data.
- **Training execution**: We then proceed with the actual training of the model. During this phase, we fine-tune the pre-trained weights of AraGPT2 to

align with the hierarchical classification task by using our specific dataset. This process involves iterative learning, where the model's parameters are refined to reduce the difference between predicted and actual labels.

3.4. Evaluation layer

Once the model is trained, we evaluate its performance using a variety of metrics designed for hierarchical classification:

- **Prediction generation**: We use the fine-tuned AraGPT2 model to predict hierarchical labels for texts in the test set. This involves determining both the category and super-category for each text, reflecting the hierarchical nature of our classification task.
- **Metrics calculation**: We assess the model's performance using several key metrics: Category Accuracy, Super-Category Accuracy, Category F1 Score, Super-Category F1 Score, and Hierarchical F1 Score (more details in section 4).

Through this structured approach, we effectively leverage AraGPT2's capabilities, adapting it to the complex task of hierarchical Arabic text classification. Our approach illustrates the effectiveness of fine-tuning a pre-trained language model to attain superior performance in specialized tasks.

4. Experiments

In this section, we outline our experimental setup and present the results of our evaluation of the fine-tuned AraGPT2 model for hierarchical Arabic text classification.

4.1. Implemented tool

To support our methodology, we developed a tool using Python and several key libraries from both NLP and ML domains. Our implementation is accessible on GitHub[1] for the benefit of the AI and NLP communities.

The main libraries used in our implementation include:

- **pandas**: For data manipulation and analysis. pandas provides powerful tools for handling and processing structured data, making it a cornerstone for data analysis in Python [36].
- **sklearn**: For splitting the dataset and evaluating the model with various metrics. The sklearn library (Scikit-learn) is a widely-used toolkit for machine learning in Python, offering straightforward and effective tools for data mining and analysis [37].
- **transformers**: From the Hugging Face library, used for loading and working with pre-trained models like GPT-2. The transformers library provides state-of-the-art implementations of Transformer models, making it easier to integrate advanced NLP models into various applications [38].
- **torch**: For building and training deep learning models. torch (PyTorch) is an open-source deep learning framework that expedites the transition from research prototyping to production deployment [39].
- **datasets**: For handling the dataset during training and testing. The datasets library simplifies the process of accessing and working with large datasets for machine learning and NLP tasks [40].

These libraries provide the essential functions and classes needed to efficiently develop and fine-tune our model for hierarchical Arabic text classification.

4.2. Dataset

We used the WiHArD[2] (Wikipedia-based Hierarchical Arabic Dataset) for our experiments. This open-access dataset is available on the Elsevier Mendeley Data Platform and consists of more than 6000 texts organized hierarchically.

The dataset is structured into two levels of hierarchy:

- **Level 1 (Super-Categories)**: Culture (ثقافة), History (تاريخ), and Math (رياضيات).
- **Level 2 (Categories)**: Clothes (ملابس), Tourism (سياحة), Food_drinks (طعام و شراب), Events (أحداث), Monuments (أثار), Inventions (اختراعات), Algebra (جبر), Geometry (هندسة), and Analysis (تحليل).

These hierarchical classes provide a rich framework for testing the capabilities of our classification model.

[1] https://github.com/khouloud-1/AraGPT2vsML

[2] https://data.mendeley.com/datasets/kdkryh5rs2/2

4.3. Evaluation metrics

We evaluated our model using several key metrics tailored for hierarchical classification [41]. These metrics offer insight into the model's performance across various levels of the classification hierarchy:

- **Category Accuracy**: Measures the proportion of correctly predicted categories out of all predictions.

$$\textbf{\textit{Category Accuracy}} = \frac{Number\ of\ correct\ category\ predictions}{Total\ number\ of\ predictions} \quad (1)$$

- **Super-Category Accuracy**: Assesses the proportion of correctly predicted super-categories.

$$\textbf{\textit{Super\_Category Accuracy}} = \frac{Number\ of\ correct\ super\_category\ predictions}{Total\ number\ of\ predictions} \quad (2)$$

- **Category F1 Score**: Balances precision and recall for category predictions.

$$\textbf{\textit{F1 Score}} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (3)$$

- **Super-Category F1 Score**: Balances precision and recall for super-category predictions using the same F1 score formula applied at the super-category level.

$$\textbf{\textit{Super\_Category F1 Score}} = 2 \times \frac{Super\_Category\ Precision \times Super\_Category\ Recall}{Super\_Category\ Precision + Super\_Category\ Recall} \quad (4)$$

- **Hierarchical F1 Score**: Combines precision and recall across the hierarchical levels. This metric takes into account both the category and super-category levels to evaluate the overall hierarchical structure.

$$\textbf{\textit{Hierarchical F1 Score}} = \frac{1}{2} \times \left(\frac{Category\ Precision \times Category\ Recall}{Category\ Precision + Category\ Recall} + \frac{Super\_Category\ Precision \times Super\_Category\ Recall}{Super\_Category\ Precision + Super\_Category\ Recall} \right) \quad (5)$$

These metrics are essential for grasping the model's performance, particularly in the context of multi-level classification.

4.4. Configuration parameters

To optimize our model's performance, we considered several configuration parameters:

- **Training and test set sizes:** We split the dataset into 70% for training and 30% for testing. This split provides a balanced approach for model training and evaluation [42].
- **Block size:** We set the block size to 128, which determines the maximum length of each text sequence after tokenization. This length was chosen based on preliminary tests showing significant results.
- **Training arguments:**
  – num_train_epochs: The number of times the whole training dataset is fed through the model during training. We tested various values to determine the optimal number of epochs.
  – per_device_train_batch_size: The number of training examples handled together in a single forward and backward pass through the model for each training step, per device (GPU or CPU).
  – save_steps: The frequency, in terms of steps, at which model checkpoints are saved during training.
- **Post-processing step:** This involves handling and correcting any deformed outputs from the fine-tuned model. For example, ensuring the output follows the format "Text -> Category -> Super_Category" and not "Text -> Category -> Super_Category -> Super_Category".

4.5. Evaluation results

In this subsection, we present the evaluation results of our fine-tuned AraGPT2 model for hierarchical Arabic text classification. We conducted eight experiments with varying configurations of training parameters and the use of a post-processing step to evaluate the model's performance in a thorough manner. Table 1 details the configuration and results of each experiment.

Table 1. Hierarchical Arabic Text classification results.

Experiment number	Post-processing	Training Arguments			Results				
		num_train_epochs	per_device_train_batch_size	save_steps	Category Accuracy	Super-Category Accuracy	Category F1 Score	Super-Category F1 Score	Hierarchical F1 Score
1	Disabled	1	1	500	0.0238	0.0205	0.0436	0.0359	0.0137
2	Enabled	1	1	500	0.5998	0.6545	0.6364	0.7437	0.5975
3	Enabled	3	4	10000	0.7700	0.8115	0.7981	0.8653	0.7694
4	Enabled	5	7	12000	0.7750	0.8137	0.8042	0.8683	0.7745
5	Enabled	10	10	15000	0.7955	0.8286	0.8230	0.8813	0.7939
6	Enabled	15	15	17000	0.8004	0.8292	0.8294	0.8835	0.7984
7	Enabled	20	20	20000	0.8060	0.8325	0.8335	0.8858	0.8031
8	Enabled	25	25	25000	**0.8098**	**0.8364**	**0.8377**	**0.8905**	**0.8064**

The results from these experiments highlight several key findings:

- **Impact of post-processing:** The significant improvement in all metrics from Experiment 1 to Experiment 2 illustrates the importance of the post-processing step. Enabling post-processing helped in correcting hierarchical structure errors, which is crucial for hierarchical classification tasks.
- **Training duration and batch size:** Increasing the number of training epochs and the batch size consistently improved performance metrics across the experiments. This indicates that longer training and larger batch sizes allow the model to learn more effectively from the dataset.
- **Optimal configuration:** Experiment 8 achieved the highest overall scores, suggesting that extended training epochs, larger batch sizes, and appropriate checkpoint saving intervals contribute to better model performance.
- **Hierarchical metrics:** The Hierarchical F1 Score, which considers both category and super-category levels, reflects the model's effectiveness in maintaining the hierarchical relationships in the data. The steady improvement across experiments underscores the model's growing ability to handle the complexity of hierarchical classification.

These insights demonstrate the robustness of the fine-tuned AraGPT2 model in managing hierarchical text classification and its adaptability to various training configurations.

5. Comparative Study

To demonstrate the efficacy of our approach, we compared the performance of our fine-tuned AraGPT2 model to a traditional machine learning-based classifier.

5.1. Machine learning-based classifier

For comparison, we implemented a machine learning-based classification system using the same WiHArD dataset (70% training set, and 30% test set). The process involved several key steps:

a) **Preprocessing**: Each text was cleaned and tokenized to remove noise and prepare it for further processing [12].
b) **Doc2Vec representation**: We used the Doc2Vec algorithm to convert each text into a fixed-size vector representation. Doc2Vec is a powerful tool for capturing the semantic meaning of texts, enabling better input for classification models [43].
c) **Decision tree classifier**: A Decision Tree algorithm was employed for the classification task. Decision Trees are widely used in ML for

their interpretability and ability to handle both categorical and continuous data [44].

The Python code for the machine learning-based classifier is also available on GitHub[3] for the benefit of the AI and NLP communities.

5.2. Comparison results

The comparison of the Hierarchical F1 scores of both models underscores the superiority of our fine-tuned AraGPT2 approach:

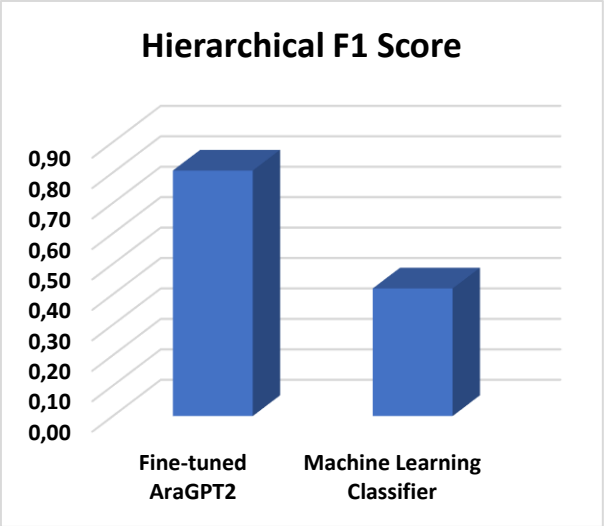

Figure 2. Comparison results (Fine-tuned AraGPT2 vs ML classifier).

These results, in Figure 2, clearly illustrate that the fine-tuned AraGPT2 model significantly outperforms the traditional machine learning-based classifier, demonstrating the advantages of leveraging large language models for complex classification tasks.

6. Conclusion and Perspectives

In this study, we introduced a novel approach for hierarchical Arabic text classification by fine-tuning the AraGPT2 model. Our methodology involved comprehensive data preparation, model adaptation, and rigorous evaluation using hierarchical metrics. The experimental results demonstrated that our fine-tuned AraGPT2 model significantly outperforms traditional machine learning classifiers, achieving high accuracy and F1 scores across different classification levels.

Our work makes several notable contributions:

- **Adaptation to hierarchical classification**: We successfully adapted the AraGPT2 model to perform hierarchical classification, a challenging task due to the need to categorize texts into multi-level taxonomies and differentiate between closely related categories within the same super-class.
- **Addressing underrepresentation in Arabic NLP:** We tackled the underrepresentation of the Arabic language in NLP research, providing a robust model that can classify Arabic texts with high precision.
- **Superior performance:** Our comparative evaluation against traditional machine learning-based classifier illustrates the superior performance of our approach in handling hierarchical classification effectively.
- **Model adaptation for Arabic:** By leveraging AraGPT2, a model specifically designed for Arabic, we ensured that our solution was well-suited to the linguistic characteristics of the Arabic language.

Looking ahead, there are multiple promising avenues for future research:

- **Extension to longer texts**: While our current study focused on short texts from the WiHArD dataset, we plan to extend our approach to classify longer texts and validate our model's performance across a broader range of datasets. This will help us better understand the model's capability to handle complex and lengthy textual content.
- **Inclusion of Arabic dialects**: We aim to extend our model to recognize and categorize texts written in various Arabic dialects, given their prevalent use in daily communication, especially on social media [45].
- **Comparison with other Deep Learning models**: We plan to compare our fine-tuned AraGPT2 [6] system with other deep learning-based classifiers, such as AraBERT [10] and BiLSTM (Bidirectional Long Short-Term

[3] *https://github.com/khouloud-1/AraGPT2vsML*

Memory) networks, to evaluate the relative performance and advantages of each approach.

- **Comparison with Machine Learning classifiers**: Further comparison with traditional machine learning-based classifiers, such as Naive Bayes and Logistic Regression, will help us highlight the strengths and potential areas for improvement in our hierarchical classification system [46].
- **Enhanced pre-processing and post-processing**: We intend to refine our classification outputs by integrating advanced pre-processing and post-processing techniques. These enhancements could involve more sophisticated text normalization methods and improved handling of model outputs to further boost accuracy.
- **Expansion of application domains**: Expanding the application domains of our model to various fields such as automated content moderation, targeted content delivery, and more nuanced topic categorization in Arabic content is another potential future direction.

In summary, our work provides a solid foundation for more advanced applications of hierarchical classification in Arabic NLP. By extending our focus to include Arabic dialects, comparing with other deep learning and traditional classifiers, and exploring additional future directions, we aim to contribute significantly to the field and offer valuable tools for managing the complexities of Arabic text in various contexts.

Conflicts Interest Statement

The authors declare that they have no known conflicts financial interests or personal relationships that could have influenced the work reported in this article.

Statement on the Ethical Use of AI Tools

AI tools were used in this work for text refinement and grammar checking. However, the core ideas, study design, methodology, experimental results, and conclusions are entirely the original work of the authors.

Data Availability Statement

To verify our proposed model, we used the WiHArD dataset (Wikipedia-based Hierarchical Arabic Dataset). This dataset is open-access and available on the Elsevier Mendeley Data Platform at: [https://data.mendeley.com/datasets/kdkryh5rs2/2].

References

[1] C. C. Aggarwal and C. Zhai, "A Survey of Text Classification Algorithms," in *Mining Text Data*, C. C. Aggarwal and C. Zhai, Eds., ed Boston, MA: Springer US, 2012, pp. 163-222.

[2] B. Hamzaoui, D. Bouchiha, and A. Bouziane, "A comprehensive survey on arabic text classification: progress, challenges, and techniques," *Brazilian Journal of Technology,* vol. 8, p. e77611, 2025.

[3] A. Kostina, M. D. Dikaiakos, D. Stefanidis, and G. Pallis, "Large Language Models For Text Classification: Case Study And Comprehensive Review," *arXiv preprint arXiv:2501.08457,* 2025.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, 2019, pp. 4171-4186.

[5] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners," *OpenAI blog,* vol. 1, p. 9, 2019.

[6] W. Antoun, F. Baly, and H. Hajj, "AraGPT2: Pre-Trained Transformer for Arabic Language Generation," in *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, Kyiv, Ukraine (Virtual), 2021, pp. 196-207.

[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature,* vol. 521, pp. 436-444, 2015/05/01 2015.

[8] N. Y. Habash, *Introduction to Arabic natural language processing*: Morgan & Claypool Publishers, 2010.

[9] A. M. El-Halees, "Arabic text classification using maximum entropy," *IUG Journal of Natural Studies,* vol. 15, 2015.

[10] W. Antoun, F. Baly, and H. Hajj, "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*, Marseille, France, 2020, pp. 9-15.

[11] A. Wahdan, M. Al-Emran, and K. Shaalan, "A systematic review of Arabic text classification: areas, applications, and future directions," *Soft Computing,* vol. 28, pp. 1545-1566, 2024/01/01 2024.

[12] C. N. Silla and A. A. Freitas, "A survey of hierarchical classification across different application domains," *Data Mining and Knowledge Discovery,* vol. 22, pp. 31-72, 2011/01/01 2011.

[13] A. Zangari, M. Marcuzzo, M. Rizzo, L. Giudice, A. Albarelli, and A. Gasparetto, "Hierarchical Text Classification and Its Foundations: A Review of Current Research," *Electronics,* vol. 13, p. 1199, 2024.

[14] N. Aljedani, R. Alotaibi, and M. Taileb, "HMATC: Hierarchical multi-label Arabic text classification model

using machine learning," *Egyptian Informatics Journal,* vol. 22, pp. 225-237, 2021/09/01/ 2021.

[15] H. El Rifai, L. Al Qadi, and A. Elnagar, "Arabic text classification: the need for multi-labeling systems," *Neural Computing and Applications,* vol. 34, pp. 1135-1159, 2022.

[16] B. Alsukhni, "Multi-Label Arabic Text Classification Based on Deep Learning," in *Proceedings of the 2021 12th International Conference on Information and Communication Systems, ICICS*, Valencia, Spain, 2021, pp. 475-477.

[17] F. Elghannam, "Multi-Label Annotation and Classification of Arabic Texts Based on Extracted Seed Keyphrases and Bi-Gram Alphabet Feed Forward Neural Networks Model," *ACM Transactions on Asian and Low-Resource Language Information Processing,* vol. 22, pp. 1-16, 2022.

[18] H. E. Rifai, L. Al Qadi, and A. Elnagar, "Arabic Multi-label Text Classification of News Articles," in *Advanced Machine Learning Technologies and Applications: Proceedings of AMLTA 2021*, Cairo, Egypt, 2021, pp. 431-444.

[19] A. M. Bdeir and F. Ibrahim, "A framework for arabic tweets multi-label classification using word embedding and neural networks algorithms," in *Proceedings of the 2020 2nd International Conference on Big Data Engineering*, Shanghai, China, 2020, pp. 105-112.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems,* vol. 30, 2017.

[21] OpenAI, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774,* 2023.

[22] B. Ding, C. Qin, L. Liu, Y. K. Chia, S. Joty, B. Li, and L. Bing, "Is gpt-3 a good data annotator?," *arXiv preprint arXiv:2212.10450,* 2022.

[23] X. He, Z. Lin, Y. Gong, A.-L. Jin, H. Zhang, C. Lin, J. Jiao, S. M. Yiu, N. Duan, and W. Chen, "AnnoLLM: Making Large Language Models to Be Better Crowdsourced Annotators," in *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, Mexico City, Mexico, 2024, pp. 165-190.

[24] X. Feng, X. Feng, L. Qin, B. Qin, and T. Liu, "Language Model as an Annotator: Exploring DialoGPT for Dialogue Summarization," 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)*, Online, 2021, pp. 1479-1491.

[25] R. Zhang, Y. Li, Y. Ma, M. Zhou, and L. Zou, "Llmaaa: Making large language models as active annotators," *arXiv preprint arXiv:2310.19596,* 2023.

[26] Y. Yu, Y. Zhuang, J. Zhang, Y. Meng, A. J. Ratner, R. Krishna, J. Shen, and C. Zhang, "Large language model as attributed training data generator: A tale of diversity and bias," *Advances in neural information processing systems,* vol. 36, 2024.

[27] A. B. Kanburoğlu and F. B. Tek, "Text-to-SQL: A methodical review of challenges and models," *Turkish Journal of Electrical Engineering and Computer Sciences,* vol. 32, pp. 403-419, 2024.

[28] X. Sun, X. Li, J. Li, F. Wu, S. Guo, T. Zhang, and G. Wang, "Text classification via large language models," *arXiv preprint arXiv:2305.08377,* 2023.

[29] Y. Zhang, R. Yang, X. Xu, J. Xiao, J. Shen, and J. Han, "Teleclass: Taxonomy enrichment and llm-enhanced hierarchical text classification with minimal supervision," *arXiv preprint arXiv:2403.00165,* 2024.

[30] J. Philip, B. VeerasekharReddy, M. Harshini, I. V. S. L. Haritha, S. Patil, and S. K. Shareef, "A Comparative Study of Text Classification using Selective Machine Learning Algorithms," in *2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS)*, Madurai, India, 2023, pp. 482-484.

[31] E. Lee, C. Lee, and S. Ahn, "Comparative Study of Multiclass Text Classification in Research Proposals Using Pretrained Language Models," *Applied Sciences,* vol. 12, p. 4522, 2022.

[32] O. Koksal and O. Akgul, "A Comparative Text Classification Study with Deep Learning-Based Algorithms," in *2022 9th International Conference on Electrical and Electronics Engineering (ICEEE)*, Alanya, Turkey, 2022, pp. 387-391.

[33] M. Berrimi, M. Oussalah, A. Moussaoui, and M. Saidi, "A Comparative Study of Effective Approaches for Arabic Text Classification," *Social Science Research Network (SSRN),* 2023.

[34] D. Bouchiha, A. Bouziane, and N. Doumi, "Machine Learning for Arabic Text Classification: A Comparative Study," *Malaysian Journal of Science and Advanced Technology,* vol. 2, pp. 163-173, 2022.

[35] M. K. B. Melhem, L. Abualigah, R. A. Zitar, A. G. Hussien, and D. Oliva, "Comparative Study on Arabic Text Classification: Challenges and Opportunities," in *Classification Applications with Deep Learning and Machine Learning Technologies*, L. Abualigah, Ed., ed Cham: Springer International Publishing, 2023, pp. 217-224.

[36] W. McKinney, "Data structures for statistical computing in Python," in *Proceedings of the 9th Python in Science Conference, SciPy 2010*, 2010, pp. 51-56.

[37] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, and V. Dubourg, "Scikit-learn: Machine learning in Python," *the Journal of machine Learning research,* vol. 12, pp. 2825-2830, 2011.

[38] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, and M. Funtowicz, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 conference on*

empirical methods in natural language processing: system demonstrations, Online, 2020, pp. 38-45.

[39] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, and L. Antiga, "Pytorch: An imperative style, high-performance deep learning library," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS 2019)*, Vancouver, Canada, 2019, pp. 8026 - 8037.

[40] Q. Lhoest, A. V. Del Moral, Y. Jernite, A. Thakur, P. Von Platen, S. Patil, J. Chaumond, M. Drame, J. Plu, and L. Tunstall, "Datasets: A community library for natural language processing," *arXiv preprint arXiv:2109.02846,* 2021.

[41] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management,* vol. 45, pp. 427-437, 2009/07/01/ 2009.

[42] H. Jiawei and K. Micheline, *Data mining: concepts and techniques*: Morgan kaufmann, 2006.

[43] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proceedings of the 31st International Conference on Machine Learning*, Beijing, China, 2014, pp. 1188-1196.

[44] J. R. Quinlan, "Induction of decision trees," *Machine Learning,* vol. 1, pp. 81-106, 1986/03/01 1986.

[45] I. Hamed, F. Eryani, D. Palfreyman, and N. Habash, "ZAEBUC-Spoken: A Multilingual Multidialectal Arabic-English Speech Corpus," *arXiv preprint arXiv:2403.18182,* 2024.

[46] I. Rish, "An empirical study of the naive Bayes classifier," in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, Washington, USA, 2001, pp. 41-46.