Arabic News Text Classification Using Deep Learning Models with Dynamic N-grams

Ahmed I.Taloba[1], George Samy Rady[2], Khaled F. Hussain[3]
Department of Information Systems-Faculty of Computer and Information, Assiut University, Egypt[1]
Department of Information Technology-Faculty of Computers and Information Technology,
The Egyptian E-Learning University, Assiut, Egypt[2]
Department of Computer Science-Faculty of Computers and Information, Assiut University, Assiut, Egypt[3]

***Abstract*—The complexity and morphological richness of the Arabic language pose significant challenges in natural language processing (NLP), including issues with contextual understanding and feature extraction. Traditional deep learning architectures such as CNNs, LSTMs, and GRUs often struggle to effectively model these linguistic intricacies, limiting their performance on Arabic text analysis tasks. To address these limitations, the study integrates parallel multi-kernel word-level convolutional features into conventional and hybrid deep learning models. The convolutional windows capture short contextual relationships among neighboring word tokens, while recurrent components model longer sequential evidence; the framework does not directly analyze roots, affixes, or other within-word morphological structures. These enhancements are integrated into CNN, LSTM, GRU, and hybrid architectures such as LSTM-CNN and GRU-CNN. A comprehensive evaluation was conducted across varying learning rates to assess the impact of the enhanced configurations on model performance. The results indicate competitive performance within the evaluated architectures and dataset variants, although the magnitude of improvement depends on the model and learning rate. Under their best settings, the Dynamic N-gram LSTM-CNN achieved an accuracy of 93.32%, while the Dynamic N-gram LSTM achieved 93.57%. Because previously published studies use different corpora, class configurations, preprocessing pipelines, and evaluation protocols, these results are not presented as evidence of state-of-the-art superiority. Instead, the study provides a systematic within-study assessment of model sensitivity to architecture, preprocessing, and learning-rate selection. Future directions include character- and subword-level modeling, transformer-based architectures, and domain-specific tasks such as sentiment analysis and information retrieval.**

***Keywords*—Dynamic n-grams; GRU-CNN hybrid; Arabic NLP; sequential dependencies; text classification**

I. Introduction

The consumption and production of online news have greatly increased, as never before, due to technological advancements and the easy accessibility of the internet in the digital age. The Arabic language market, with a total of more than 420 million people, has witnessed a huge increase in the production of online news [1], [2]. Arabic news websites, including Al-Arabiya, Al Jazeera, and Youm7, reach a broad audience, providing information in many fields, like politics, sports, culture, and technology. The occurrence of this great reservoir of information has altered the aspect through which Arabic-speaking groups dwell on contemporary issues, and the necessity of digital channels in the formation of opinion and transfer of knowledge cannot be overstressed [3], [4]. Nevertheless, the increase in Arabic news material poses some difficulties as regards obtaining accessibility and classification of information. It is common among users to have difficulties progressing through the flood of content in search of the news of their interest [5], [6]. The necessity of automated classification systems has become very evident. Automated text classification would make it easier to organize the news articles in predefined categories and thereby allow a user to access the relevant articles. Such functionality is essential to end-users and makers of the content of the news, which will add to the user experience and increase the searchability of the articles [7], [8]. Besides the need to automate the increasing amount of news content, the special linguistic complexity of the Arabic language also necessitates the development of strong text classification systems specifically deployed in Arabic news management. These shall be described as complexities that require advanced machine learning and deep learning to be adopted.

The Arabic language presents distinctive challenges for automated text classification because of its morphological, syntactic, and lexical properties. Arabic is highly inflected, and surface forms may vary according to tense, gender, number, and case [9], [10]. A single root can also generate many derived forms through prefixes, suffixes, infixes, and templatic patterns, increasing word-level sparsity and making contextual interpretation important. Arabic additionally has a rich vocabulary, many near-synonyms, and context-dependent meanings. Modern Standard Arabic coexists with regional dialects that differ in syntax and vocabulary [11], [12], and digital news may contain variation across sources. Relatively flexible word order and the frequent absence of diacritics can further increase ambiguity. Accordingly, methods that combine local token context, longer sequence modeling, and controlled comparisons of stemmed and non-stemmed representations may help evaluate how these challenges affect classification. The present word-level models are not morphological analyzers and do not directly decompose Arabic roots or affixes.

A. Problem Statement

With the fast rate of expansion of Arabic news on the internet, users have found it difficult to find something of interest with ease. The current models of text classification are mainly developed to support English or less expressive

languages, which cannot be applied to the specifics of the Arabic language, i.e., the abundance of morphology, the flexibility of syntax, and the existence of different types of dialects [13], [14]. Moreover, the unavailability of customized sets and approaches curbs the functionality of classical machine learning and deep learning algorithms. Consequently, there is an urgent need to establish a strong system of categorizations dealing with the linguistic heaviness of Arabic and the categorization of news into sensible clusters. The mentioned gaps can be filled to increase usability among users, create real-time classifications, and cover various kinds of applications, including sentiment analysis and recommendation of personalized content.

B. Motivation of the Study

The sheer growth of the Arabic digital news sector has brought up the need to ensure efficient content organization to enhance access and relevance. The existing automated classification is not able to address this need because the language itself is as complex as it is morphologically rich and contextually dependent. In addition, the current developments in deep learning, particularly in hybrid models such as CNNs and LSTMs, demonstrate potential to overcome these challenges. This research derives interest in the possibility of using these sophisticated methods to develop a more powerful Arabic news classification system. This study will help to fill the divide between linguistic complexity and technological capacity and thus may contribute to the improvement of user experience and the practicality of automated Arabic text classification.

C. Significance of the Study

This study is relevant because Arabic text classification remains less extensively researched than English and other high-resource languages. The proposed research evaluates deep learning methods, including Dynamic N-grams and hybrid models, for Arabic news classification. It can support content accessibility, personalization, sentiment analysis, and trend detection. The findings provide additional empirical evidence on the behavior of deep learning architectures under different preprocessing and learning-rate settings for Arabic text classification. Finally, this research can contribute to more organized and accessible digital news environments for Arabic-speaking communities.

D. Key Contribution of the Study

- Dynamic Hybrid Model: Proposed a dynamic model integrated with dynamic n-gram and CNN, LSTM, and GRU in improving the classification of Arabic news.
- Optimized Learning Rates: Optimized the best learning rates that enhance the performance and the stability of the model.
- Systematic Evaluation of Dataset Variants: Systematically evaluated the six existing representations supplied with the Ultimate Arabic News Dataset to examine how provider-applied cleaning, stop-word removal, and stemming affect the evaluated deep learning architectures.
- Model Comparison: Conducted a controlled within-study comparison between baseline and Dynamic N-gram-enhanced architectures across multiple learning rates and dataset variants; the results are interpreted within this experimental setting rather than as a direct ranking against studies using different corpora.
- Development of Arabic NLP: Addressing the Arabic-specific issues helped in the development of improved tools in text analysis and classification.

Structure of the Study: The study includes an introduction (Section I), related works (Section II), methodology (dataset, preprocessing, model architectures, learning rate experiments) (Section III), results and analysis (Section IV), discussion, and conclusion with future directions for Arabic text classification (Section V).

II. Related Works

With the rapid rise of Arabic tweets, Alzanin et al. [15] suggest an automatic classification model that classifies tweets into five categories based on linguistic features and content. It compares two text representations: Word2Vec and stemmed tf-idf, and tests three classifiers: SVM, GNB, and RF. A manually annotated 35,600-tweet dataset was employed. Results indicate that RF and SVM using tf-idf and stemming obtained the best macro-F1 values (98.09%–98.14%), while RF obtained 98.95% accuracy, 99.54% precision, 96.58% recall, and 97.99% F1. However, GNB with Word2Vec did not work well. One major limitation is the difficulty presented by short, ambiguous, and mixed-dialect tweets.

Hossain et al. [16] propose a hybrid Attention-Based Transformer Model (ABTM) for classifying Arabic news that blends deep learning with conventional text representations such as TF-IDF and Bag-of-Words. An effective preprocessing pipeline involving cleaning, tokenization, lemmatization, and data augmentation was used to handle the richness of Arabic news text. The model combines contextual and statistical information and utilizes LIME for interpretability. Compared to AraBERT models, ABTM resulted in 97.69% accuracy, 97.13% precision, 97.11% recall, and 97.10% F1-score with various news categories. Although the model indicates robust performance and interpretability, its limitation resides in handling subtle dialectal variations and possible data imbalance between categories.

Zamzami et al. [17] report on the increasing necessity of determining the origin country of Arabic news articles to enable users to follow up on credible, pertinent news. It suggests a variety of classification models with both machine learning and deep learning methods, trained on varied feature extraction techniques. The models accurately classify articles into origin categories with a maximum accuracy of 94%, with little variation in performance between ML and DL methodologies. This shows the potential for application to regional news filtering. But there is a limitation in the difficulty of dealing with overlapping regional material and subtle linguistic variations, which could impact classification accuracy in some situations.

Imad et al. [18] introduce an Arabic text classification method using NLP methods and deep learning, with TF-IDF, Bag-of-Words, Word Embedding, and a suggested CNN model. The model was tested on three datasets—CNN, BBC, and OSAC—with high accuracy: 94.47% (One Hot), 93.98% (BOW), and 93.58% (TF-IDF) on CNN; 90.97%, 87.19%, and 86.35% respectively on BBC; and more than 98.8% on OSAC. The results validate the model's performance on diverse datasets. Yet, there are limitations such as performance differences on the basis of dataset specifications and possibly challenges in generalizability to diverse Arabic dialects and content structures in real-world use.

Othman et al. [19] tackle the emerging issue of Arabic fake news in the media by suggesting a hybrid approach using Arabic pre-trained BERT models (AraBERT, GigaBERT, MARBERT) augmented with 1D or 2D CNNs for efficient feature extraction and classification. Evaluated on three datasets—ANS, AraNews, and Covid19Fakes—the top-performing architecture (AraBERT + 2D-CNN) achieved F1-scores of 0.6188, 0.7837, and 0.8009, and surpassed current models on the ANS dataset with 71.42% accuracy, 75.6% precision, and 52.38% recall. The model also minimized training time. A primary limitation, though, is the relatively low recall on the ANS dataset, reflecting difficulty in detecting all occurrences of fake news.

Jamaleddyn et al. [20] investigate improving Arabic text classification by utilizing machine learning algorithms with hyperparameter tuning methodologies, i.e., grid search and random search, for model performance optimization. NLP techniques were incorporated to enhance text representation and classification accuracy. The suggested method was benchmarked with the CNN Arabic dataset, resulting in high performance: 95.16% accuracy, 94.64% recall, 94.04% precision, and 94.31% F1-score. Comparison with prior work validated the strength of the model. Still, one limitation is that it relies on tuning techniques, which are computationally intensive and dataset-dependent and may compromise scalability and generalizability to diverse Arabic language corpora.

Alzaidi et al. [21] introduce the TCAODL-ANA method—an improved Arabic news categorization model that integrates FastText word embedding, attention-based bidirectional GRU (ABiGRU), and Aquila Optimizer (AO) for hyperparameter optimization. Targeted at classifying news into seven categories, the model utilizes deep learning's contextual advantages and AO's optimization to enhance performance. Tested using simulations, TCAODL-ANA recorded 95.48% accuracy, 84.21% precision, 84.18% recall, and 84.17% F1-score, surpassing current methods. Although effective, the limitations of the model are relatively moderate precision and recall, implying possible difficulties with ambiguous or overlapped news topics, and heavy reliance on large computational resources for optimization.

Alroobaea [22] considers Arabic news classification based on 5,000 news articles in six categories: business, entertainment, health, politics, sports, and technology. Different preprocessing methods, word embeddings, and deep learning models—CNN, LSTM, and a CNN-LSTM hybrid—were considered. The hybrid model yielded the best accuracy of 93.15% and provided competitive results on other Arabic news datasets. Model architecture and preprocessing decisions are critical to the findings. Nonetheless, a limitation is the comparatively small size of the dataset, which can impact generalization, and the performance of the model can differ for diverse dialects or less-organized Arabic news outlets. The summarization of the related works is shown in Table I.

TABLE I. SUMMARIZATION OF THE RELATED WORKS

Author (s)	Approach	Dataset(s)	Best Results	Limitations
Alzani n et al., [15]	Word2Vec & tf-idf with SVM, RF, GNB	35,600 Arabic tweets	RF: Acc 98.95%, Prec 99.54%, Recall 96.58%, F1 97.99%	Poor GNB performance; ambiguity and dialectal mixing in short tweets
Hossai n et al., [16]	ABTM (Attention-Based Transformer Model) + TF-IDF/BOW + LIME	Benchma rk with AraBER T models	Acc 97.69%, Prec 97.13%, Recall 97.11%, F1 97.10%	Dialectal variation & potential data imbalance
Zamza mi et al., [17]	ML and DL models with various feature extraction methods	Multiple Arabic news datasets	Accuracy up to 94%	Overlapping regional content and linguistic subtleties may affect precision.
Imad et al., [18]	TF-IDF, BOW, Word Embedding + CNN	CNN, BBC, OSAC	CNN: Acc 94.47%, BBC: 90.97%, OSAC: >98.8%	Performance varies across datasets; dialect/genera lization concerns
Othma n et al., [19]	Hybrid: AraBERT/GigaB ERT/MARBERT + 1D/2D CNN	ANS, AraNews , Covid19 Fakes	F1: 0.6188 (ANS), 0.7837 (AraNews), 0.8009 (Covid19Fake s); Acc: 71.42% (ANS)	Low recall on ANS; difficult to capture all fake news cases
Jamale ddyn et al., [20]	ML + Grid/Random Search for hyperparameter tuning	CNN Arabic dataset	Acc 95.16%, Recall 94.64%, Prec 94.04%, F1 94.31%	Tuning is resource-intensive; the model may not generalize across datasets
Alzaidi et al., [21]	TCAODL-ANA: FastText + ABiGRU + Aquila Optimizer	Simulate d Arabic news dataset	Acc 95.48%, Prec 84.21%, Recall 84.18%, F1 84.17%	Moderate precision/reca ll; computationa lly demanding due to AO
Alroob aea [22]	Preprocessing + Word Embeddings + CNN, LSTM, CNN-LSTM	5,000 Arabic news articles	Acc 93.15% (Hybrid CNN-LSTM)	Small dataset; may not generalize well; performance may vary by dialect or news type

III. Framework for Arabic News Text Classification

The proposed study establishes a framework for Arabic news classification using deep learning models and the existing preprocessing variants supplied with the Ultimate Arabic News Dataset. The dataset representations are used as provided by the original contributor to examine how stop-word removal and stemming influence model performance; the study does not claim to release a newly enhanced corpus. The framework integrates Dynamic N-gram extraction with CNN, LSTM, and GRU architectures, evaluated individually and in hybrid combinations to capture local textual patterns and long-range dependencies. Five learning rates are examined to assess convergence and performance sensitivity. The block diagram of the proposed work is shown in Fig. 1.

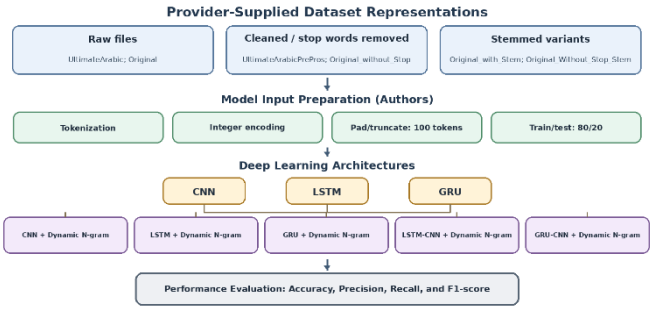

Fig. 1. Block diagram for the proposed study.

A. Linguistic Motivation and Research Hypotheses

Arabic news classification is influenced by high surface-form variation arising from inflection and derivation, clitic attachment, relatively flexible word order, absent diacritics, and lexical ambiguity [9]-[12]. These properties motivate architectures that can combine local contextual evidence with information distributed across a longer sequence. The present implementation uses word-level tokenization; therefore, its convolutional branches operate over neighboring word tokens and do not directly segment or analyze Arabic roots, patterns, prefixes, suffixes, or other within-word morphological units. Morphological variation is addressed only indirectly through learned word embeddings, surrounding context, and comparison of the provider-supplied stemmed and non-stemmed dataset variants.

Kernel sizes 2, 3, and 4 were used as controlled local context scales to detect short token sequences and category-bearing phrases. The recurrent LSTM and GRU components were included to retain evidence across longer portions of an article, which may be useful when semantically related cues are separated by flexible word order or intervening text. The hybrid architectures were consequently designed to test whether combining local phrase extraction with sequential context provides complementary information. These choices are theoretical expectations rather than assumptions of uniform superiority, and the results are interpreted against the following hypotheses.

- H1 (local-context hypothesis): Multi-kernel convolution is expected to improve selected models when category labels are signaled by short neighboring-token patterns; any benefit is expected to depend on the architecture and training configuration rather than occur universally.
- H2 (sequential-context hypothesis): LSTM or GRU layers are expected to be useful when classification requires contextual evidence extending beyond a fixed local convolutional window.
- H3 (complementarity hypothesis): Hybrid convolutional-recurrent models are expected to benefit when local lexical cues and longer sequential evidence contribute jointly to the classification decision.
- H4 (preprocessing-interaction hypothesis): The relative value of each architecture is expected to change across the provider-supplied preprocessing variants because stemming and stop-word removal may reduce lexical sparsity while also removing potentially discriminative lexical or syntactic cues.

B. Data Collection

The data used in this research are from the Ultimate Arabic News Dataset [23], downloaded from Mendeley Data, Version 2 (DOI: 10.17632/jz56k5wxz7.2), published on July 4, 2022, by Ahmed Hashim Al-Dulaimi. The collection contains more than 193,000 single-label Arabic news texts gathered from sources including Al-Arabiya, Al-Youm Al-Sabea (Youm7), Google News, and other news websites. All dataset representations evaluated in this study were used as supplied by the original contributor.

- UltimateArabic: The primary file containing the original Arabic news texts without preprocessing; the texts may include words, numbers, and symbols.
- UltimateArabicPrePros: The provider-preprocessed counterpart in which stop words, non-Arabic words, symbols, and numbers were removed for direct use in Arabic NLP tasks.

The repository also provides two supplementary folders that document the collection process and offer four alternative dataset representations:

- Sample: Two source-specific subsets, Sample_Youm7_Politic and Sample_alarabiya_Sport, illustrate news collected from Youm7 and Al-Arabiya in the Politics and Sport categories, respectively.
- Dataset Versions: Four provider-supplied representations are available: Original (raw text), Original_without_Stop (cleaned text with stop words removed), Original_with_Stem (cleaned and stemmed text), and Original_Without_Stop_Stem (cleaned text with both stop-word removal and stemming).

The corpus is divided into ten single-label categories: Culture, Diverse, Economy, Sport, Politics, Art, Society, Technology, Medical, and Religion. In total, the study evaluates the two primary files and the four existing dataset-version files. The authors did not create or release an independently enhanced version of the corpus; the methodological contribution is the systematic evaluation of

these existing representations. The dataset counts are shown in Table II.

TABLE II. DATASET SAMPLE

Category	Original	Without Stem	Without Stop	Without Stop+ Stem	Ultimate Arabic	UltimateArabicPrePros
Art	3,000	3,000	3,000	3,000	14,390	14,388
Culture	3,000	3,000	3,000	3,000	6,806	6,806
Economy	2,995	2,995	2,995	2,995	26,117	26,038
Medical	3,000	3,000	3,000	3,000	7,294	7,294
Politics	3,000	3,000	3,000	3,000	45,369	45,349
Religion	3,000	3,000	3,000	3,000	6,500	6,500
Sport	3,000	3,000	3,000	3,000	59,998	57,151
Technology	3,000	3,000	3,000	3,000	11,518	11,518
Diverse	3,000	3,000	3,000	3,000	17,203	17,203
Society	0	0	0	0	1,084	1,084
Total	26,995	26,995	26,995	26,995	196,279	193,331

C. Dataset Variants and Model Input Preparation

The corpus-level transformations were not newly introduced by the present authors. The six representations were selected and used as provided in Version 2 of the dataset. The provider-supplied transformations are distinguished below from the subsequent model-input operations. Beyond selecting the appropriate file for each experiment, the authors do not claim additional corpus cleaning, stop-word removal, stemming, or the release of a new processed dataset.

1) Provider-supplied character cleaning: In the cleaned variants, the dataset contributor removed non-Arabic words, symbols, and numbers from the original text.

$$D\_clean = Clean(D) \quad (1)$$

2) Provider-supplied stop-word removal: Stop-word removal is already represented in UltimateArabicPrePros, Original_without_Stop, and Original_Without_Stop_Stem; it was not introduced as a new dataset contribution in this study.

$$D\_stop = RemoveStopWords(D\_clean) \quad (2)$$

3) Provider-supplied stemming: Stemming is already represented in Original_with_Stem and Original_Without_Stop_Stem, as supplied by the dataset contributor.

$$D\_stem = Stem(D\_clean) \quad (3)$$

4) Provider-supplied combined variant: Original_Without_Stop_Stem combines the contributor's cleaning, stop-word removal, and stemming operations in a single representation.

$$D\_both = Stem(RemoveStopWords(D\_clean)) \quad (4)$$

5) Tokenization: For model input, each selected text representation was segmented into tokens.

$$T = Tokenize(D\_selected) \quad (5)$$

6) Integer sequence encoding: The resulting tokens were mapped to integer sequences so they could be processed by the embedding layer.

$$X = Encode(T) \quad (6)$$

7) Fixed-length sequence preparation: Sequences were standardized to the reported length of 100 tokens by padding shorter sequences and truncating longer sequences.

$$X\_100 = PadOrTruncate(X, 100) \quad (7)$$

8) Data partitioning: For each evaluated representation, the prepared samples were divided into the reported 80% training and 20% testing partitions.

$$(D\_train, D\_test) = Split(D\_selected, 80/20) \quad (8)$$

D. Baseline Models

The baseline models provide the basis for assessing and comprehending the efficacy of text classification methods for Arabic news data. The models are suitable for coping with distinctive linguistic patterns in Arabic text, and they also shed light on the performance of classical deep learning architectures.

1) Convolutional Neural Network (CNN): CNNs are commonly employed in natural language processing (NLP) due to their capacity to effectively extract local patterns like n-grams and key phrases from text. The hierarchical structure processes input data hierarchically to ensure that significant features are captured at every level. The structure of the CNN is shown in Fig. 2.

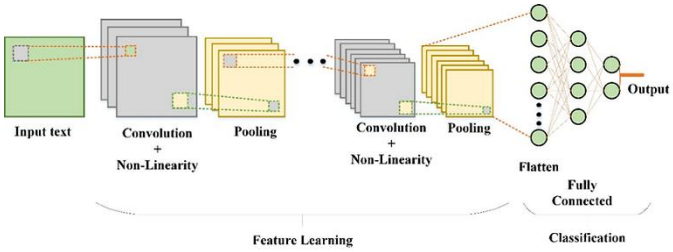

Fig. 2. CNN architecture.

a) Embedding layer: Input sequences of integer-encoded tokens are converted into dense vector representations by this layer. Every token, denoted by t_i, is mapped to an embedding vector $e_i \epsilon R^d$, denoted by d, where d is the embedding dimension. Word semantic links are preserved thanks to the embedding. It is shown in Eq. (9).

$$E = Embedding(T) \quad (9)$$

where, T is the token sequence and E is the embedding matrix.

b) Conv1D layers: To find patterns in fixed-length contexts, one-dimensional convolutional filters are applied over the embeddings. To create feature maps, a filter $w \epsilon R^{k \times d}$ with kernel size k slides over the embedding matrix. It is shown in Eq. (10).

$$h_i = \sigma(w.E_{i:i+k-1} + b) \quad (10)$$

The bias component is denoted by b, the activation function is denoted by σ, and the activation for the i-th position is represented by h_i.

c) Global max pooling: The layer reduces dimensions and retains the most salient characteristics by selecting the maximum activation of each feature map. It is shown in Eq. (11).

$$h_{max} = \max_i(h_i) \quad (11)$$

d) Dense layers: Fully connected layers handle the pooled features and apply softmax activation to support multi-class classification. It is shown in Eq. (12).

$$\hat{y} = Softmax(W.h_{max} + b) \quad (12)$$

e) Rationale: CNNs are well suited to detecting local textual patterns, including short token sequences and category-bearing phrases. In this study, convolution is applied to word-token sequences; consequently, it captures contextual relationships among neighboring words rather than prefixes, suffixes, roots, or patterns within individual words.

2) Long Short-Term Memory (LSTM): LSTM networks are geared to learn long-term dependencies in sequential data by solving the vanishing gradient problem in regular recurrent neural networks (RNNs). This positions LSTMs well to capture contextual and sequential patterns in Arabic text. The LSTM structure is shown in Fig. 3.

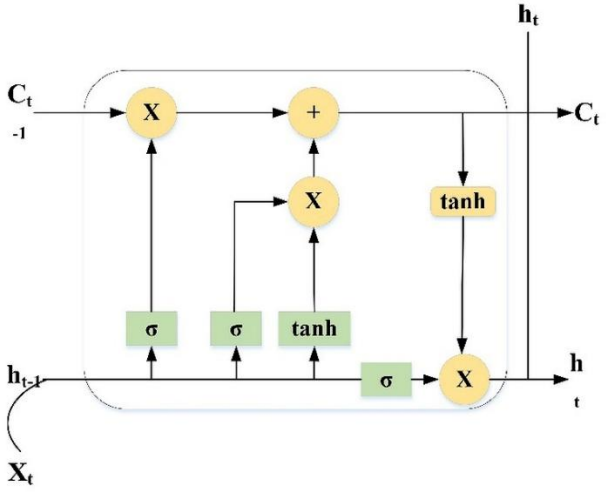

Fig. 3. LSTM structure.

a) Embedding layer: Functions as in CNN, mapping tokens to dense vectors.

b) LSTM layer: The following equations define the model's central component, an LSTM cell at time step t, shown in Eq. (13)-(18).

$$f_t = \sigma(W_f[h_{t-1}, X_t] + b_f) \quad (13)$$

$$i_t = \sigma(W_i[h_{t-1}, X_t] + b_i) \quad (14)$$

$$g_t = tanh(W_g[h_{t-1}, X_t] + b_g) \quad (15)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (16)$$

$$o_t = \sigma(W_o[h_{t-1}, X_t] + b_o) \quad (17)$$

$$h_t = o_t \odot \tanh(c_t) \quad (18)$$

Here, the cell state is represented by c_t, the hidden state by h_t, and the forget, input, and output gates are represented by f_t, i_t, and o_t, respectively.

c) Dense layers: Like CNNs, dense layers project the final output of LSTM to classification probabilities via softmax.

d) Rationale: LSTMs learn long dependencies and word order, which are necessary for realizing complex Arabic sentence structure and syntax.

3) Gated Recurrent Unit (GRU): GRUs are a recurrent architecture that uses update and reset gates to model sequential information with a simpler gating structure than LSTMs. The GRU structure is shown in Fig. 4.

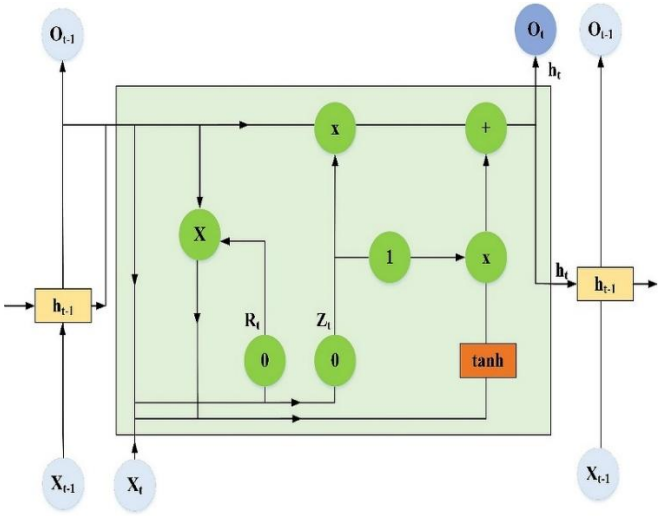

Fig. 4. GRU structure.

a) Embedding layer: Same as in CNN and LSTM.

b) GRU Layer: By merging the input and forget gates into an update gate, GRU streamlines the architecture. The equations at the time step t are shown in Eq. (19)-(21).

$$z_t = \sigma(W_z[h_{t-1}, X_t] + b_z) \quad (19)$$

$$r_t = \sigma(W_r[h_{t-1}, X_t] + b_r) \quad (20)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tanh(W_h[r_t \odot h_{t+1}, x_t] + b_h \quad (21)$$

If h_t is the new hidden state, z_t is the update gate, and r_t This is the reset gate.

c) Dense layers: As in the case of LSTM, the final output is passed to fully connected layers with softmax for classification.

d) Rationale: GRUs provide an alternative recurrent mechanism for capturing contextual and sequential dependencies in Arabic text. In this study, their suitability is assessed through classification metrics only; computational efficiency was not benchmarked.

By using such baseline models, the research comprehensively examines the strengths and limitations of various architectures for Arabic news classification. These models also serve as a benchmark for measuring the performance of sophisticated hybrid solutions.

E. Hybrid and Advanced Models

To further improve classification accuracy, hybrid and advanced versions of the models were used by leveraging the strengths of Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM), Gated Recurrent Units (GRUs), and dynamic n-gram feature extraction methods. These models seek to leverage both local patterns and sequential dependencies, providing an enhanced feature representation. The following is a detailed overview of each hybrid model, its architecture, and the significance of the model with respect to Arabic text classification.

1) Dynamic N-gram + CNN

a) Dynamic N-gram block: The main innovation consists of parallel Conv1D layers that capture bi-, tri-, and four-grams with different kernel sizes k (e.g., 2, 3, 4). The convolution process for an embedding matrix, where n is the sequence length and d is the embedding dimension. It is shown in Eq. (22).

$$h_{i,j} = \sigma(w_j . E_{i:i+k_j-1} + b_j) \quad (22)$$

where, the bias is represented by b_j, the filter is represented by w_j, the activation function by σ, and the kernel size is indicated by σ.

b) Concatenation: The outputs of the Conv1D layers are concatenated to form a feature-rich representation. It is shown in Eq. (23).

$$H_{concat} = [H_1, H_2, H_3] \quad (23)$$

where, the feature map from the j-th kernel is represented by H_j.

c) Global max pooling: Reduces dimensionality by choosing the maximum activation for every feature map. It is shown in Eq. (24).

$$h_{max} = \max_i(H_{concat}) \quad (24)$$

d) Dense layers: Fully connected layers process the pooled features and offer classification probabilities using a softmax layer. Dynamic N-gram + CNN is shown in Fig. 5.

e) Rationale: The use of multiple kernel sizes allows this model to capture different local textual patterns. The present study evaluates the effect of this configuration on classification performance; training speed and computational cost were not measured.

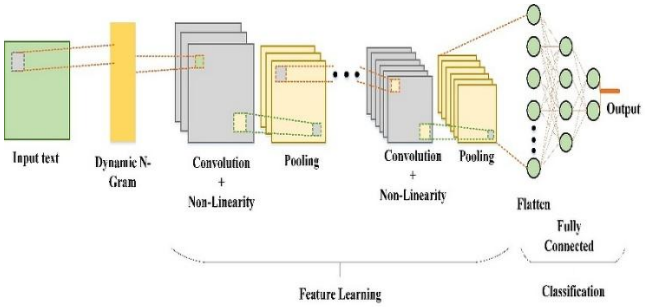

Fig. 5. Dynamic N-gram + CNN.

2) Dynamic N-gram + LSTM

a) Dynamic N-gram block: This block, similar to the above model, captures n-gram patterns using parallel Conv1D layers.

b) LSTM layer: Concatenated n-gram features are fed to an LSTM layer to capture long dependencies and context relationships. The LSTM handles input sequentially with both hidden and cell states. It is shown in Eq. (25).

$$h_t, c_t = LSTM(x_t, h_{t-1}, c_{t-1}) \quad (25)$$

Here, the concealed state is denoted by h_t, the cell state by c_t, and the input at time t by x_t.

c) Dense layers: Final layers project the LSTM output to class probabilities. Dynamic N-gram + LSTM is shown in Fig. 6.

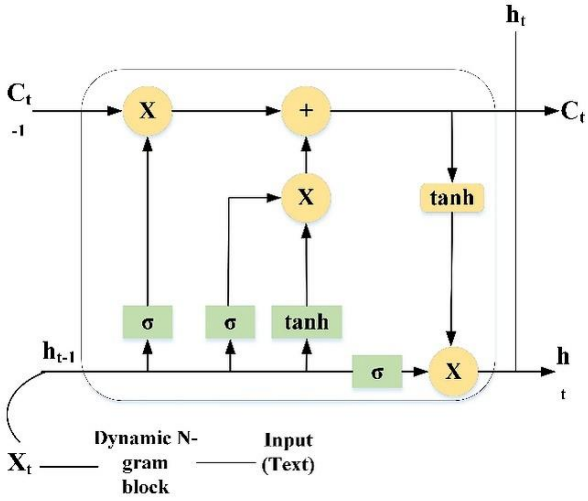

Fig. 6. Dynamic N-gram + LSTM.

d) Rationale: This model brings together the CNN's power in extracting strong local features and the LSTM's capacity for sequential dependency capture, offering a comprehensive solution to Arabic text classification.

3) Dynamic N-gram + GRU

a) Dynamic N-gram block: As defined earlier, this block extracts bi-grams, tri-grams, and more with Conv1D layers.

b) GRU layer: The GRU layer substitutes the LSTM and models sequential relations through update and reset gates. The GRU cell can be described as shown in Eq. (26)-(28).

$$z_t = \sigma(W_z[h_{t-1}, x_t] + b_z) \quad (26)$$

$$r_t = \sigma(W_r[h_{t-1}, x_t] + b_r) \quad (27)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tanh(W_h[r_t \odot h_{t+1}, x_t] + b_h \quad (28)$$

c) Dense layers: Final classification is done by fully connected layers with softmax activation. Dynamic N-gram + GRU is shown in Fig. 7.

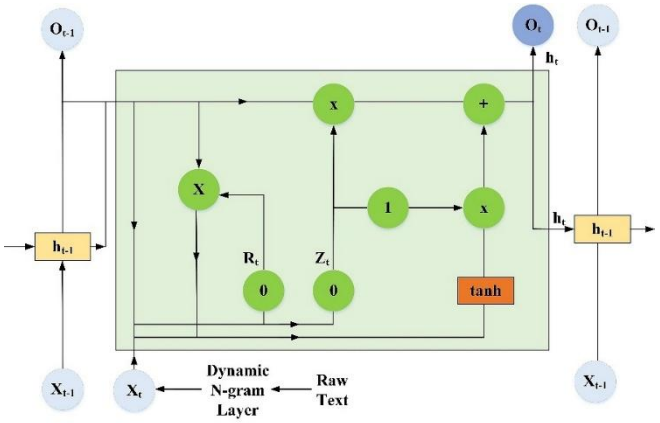

Fig. 7. Dynamic N-gram + GRU.

d) Rationale: The integration of GRU and Dynamic N-gram extraction combines multi-scale local features with recurrent sequence modeling. Its classification behavior is evaluated experimentally, while computational efficiency and suitability for resource-limited deployment are not assessed in this study.

4) Dynamic N-gram + LSTM + CNN

a) Dynamic N-gram block: Parallel Conv1D layers extract varied local patterns, similar to earlier models.

b) LSTM layer: Parameterized with return_sequences=True, the LSTM layer processes n-gram features, extracting sequential dependencies without losing temporal structure for subsequent processing. It is shown in Eq. (29).

$$h_t = LSTM(H_{concat}) \quad (29)$$

c) Additional CNN layer: The LSTM outputs are passed through another Conv1D layer to fine-tune local patterns. It is shown in Eq. (30).

$$h_{refine} = Conv1D(h_t) \quad (30)$$

d) Global max pooling: Combines fine-tuned features and dimension reduction.

e) Dense layers: Fully connected layers finalize the architecture, ending with a final softmax layer for classification. Dynamic N-gram + LSTM + CNN is shown in Fig. 8.

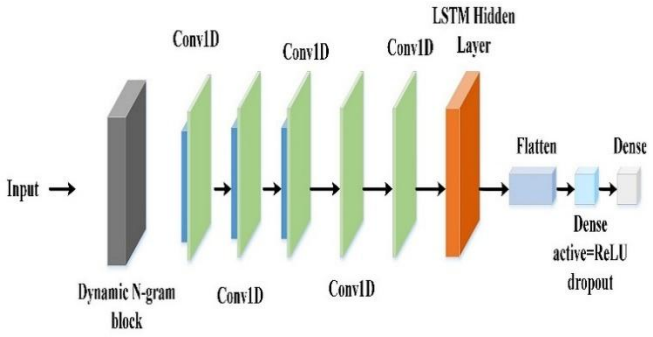

Fig. 8. Dynamic N-gram + LSTM + CNN.

f) Rationale: This hybrid model extracts fine-grained local features using CNNs, formulates their temporal relationships using LSTM, and further refines the resultant patterns with a second CNN layer. This multi-step processing guarantees excellent feature representation and classification performance.

5) Dynamic N-gram + GRU + CNN

a) Dynamic n-gram block: Parallel Conv1D layers: Derive varied local patterns from the input text data, such as in the Dynamic N-gram + LSTM + CNN architecture.

- Output: Integrated local features for sequential processing.

b) GRU layer: Set up with return_sequences=True, the GRU layer processes the n-gram features while retaining temporal relationships for the subsequent layer. It is shown in Eq. (31).

$$h_t = GRU(H_{concat}) \quad (31)$$

c) Additional CNN layer: The GRU outputs are passed through a Conv1D layer to further refine local patterns so that feature extraction at multiple levels is possible. It is shown in Eq. (32).

$$h_{refine} = Conv1D(h_t) \quad (32)$$

d) Global max pooling: Combines refined features and performs dimensionality reduction by taking the most important feature values from each feature map.

e) Dense layers: The aggregated features are processed by fully connected layers, leading to a softmax layer for multi-class classification. Dynamic N-gram + GRU + CNN is shown in Fig. 9.

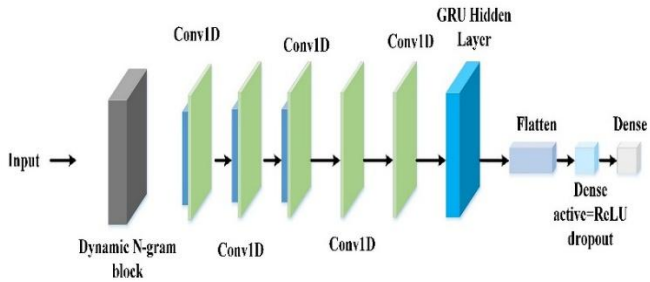

Fig. 9. Dynamic N-gram + GRU + CNN.

This fusion architecture combines CNN-based local feature extraction with GRU-based sequential dependency modeling. The additional CNN layer further refines the temporal representations produced by the GRU, while the Dynamic N-gram block supplies features derived from multiple kernel sizes. Because runtime and resource measurements were not collected, no conclusion is drawn regarding a computational advantage or suitability for resource-constrained deployment.

The advanced and hybrid models are evaluated because Arabic news classification may depend on both local token patterns and longer contextual relations. Combining multi-kernel word-level convolution with sequence modeling tests whether these two forms of evidence are complementary across the supplied preprocessing variants. The architecture does not itself solve Arabic morphology or directly analyze within-word structure (Algorithm 1).

Algorithm 1: Dynamic N-gram + GRU + CNN for Arabic Text Classification

Input: Arabic text dataset D

Output: Classified labels L

Step 1: Preprocessing:

Clean text (remove special characters, stop words).

Generate Dynamic N-grams and encode using embeddings.

Step 2: Dynamic N-gram Block:

Apply parallel $Conv1D$ layers to capture diverse local patterns.

Concatenate outputs to form H_{ngram}

Step 3: GRU Processing:

Pass H_{ngram} to GRU (return_sequences=True) to model sequential dependencies.

Output: H_{gru} CNN Refinement:

Step 4: CNN Refinement

Apply another $Conv1D$ layer on H_{gru} to refine features.

Output: H_{gru}

Step 5: Pooling:

Perform global max pooling to reduce dimensionality.

Result: H_{pool}

Step 6: Classification:

Pass H_{pool} through dense layers and a softmax layer for classification.

Step 7: Training:

Use cross-entropy loss and Adam optimizer with tuned learning rates.

Step 8: Evaluation:

Compute Accuracy, Precision, Recall, and F1-Score.

End

IV. Result and Discussion

The results and discussion section presents a detailed show of the performance of the models in different architectures and learning rates. To guarantee uniform assessment, each of the models trained in TensorFlow/Keras was presented with 10 epochs. The results of the performance were determined in the form of a weighted average to take into account the probable class imbalance. The metrics, such as the percentage value with two decimals, are classified in bordered tables to facilitate comparison and ease of calling. The aim of the evaluation is to determine what the best learning rate is per architecture and how the trade-off between the convergence rate and the performance is. The explanation is to focus on the effectiveness of the various architectures (e.g., CNN, LSTM, and hybrid models) in addressing the complexities of the dataset. Through combining the observations of these critiques, the following section will serve to emphasize the merits, weaknesses, and application to real life of the suggested methodologies applied to Arabic text classification, providing an authoritative basis to work on in the form of future developments. All experiments were conducted on a 64-bit workstation equipped with an AMD Ryzen 5 3600 six-core processor operating at 3.59 GHz and 64 GB of RAM. Model training and evaluation were performed exclusively on the CPU. Table III shows the simulation parameters.

TABLE III. Simulation Parameter

Parameter	Value
Dataset	Ultimate Arabic News Dataset
Number of Classes	10
Training Epochs	10
Batch Size	32
Learning Rates Tested	0.1, 0.01, 0.001, 0.0001, 0.00001
Evaluation Metrics	Accuracy, Precision, Recall, F1-Score
Optimizer	Adam
Activation Function	Softmax
Training/Testing Split	80% / 20%
Framework Used	TensorFlow/Keras
Dropout Rate	0.5
Kernel Sizes (CNN)	2, 3, 4
Sequence Length	100 tokens
Embedding Dimensions	128

A. Performance Metrics

Accuracy: Measures the proportion of correctly classified instances out of total instances. It is shown in Eq. (33).

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (33)$$

Precision: Indicates the proportion of true positives among predicted positives, showing correctness. It is shown in Eq. (34).

$$Precision = \frac{TP}{TP+FP} \quad (34)$$

Recall (Sensitivity): Reflects the proportion of true positives identified among actual positives. It is shown in Eq. (35).

$$Recall = \frac{TP}{TP+FN} \quad (35)$$

TABLE IV. PERFORMANCE METRICS: BASELINE MODELS VS. DYNAMIC N-GRAM ENHANCED MODELS

Model Type	Learning Rate	Baseline Model				Dynamic N-gram Model			
		Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
CNN	1.00E-05	52.45	62.58	52.45	51.42	59.21	66.25	59.21	60.33
	0.0001	89.55	89.65	89.55	89.57	89.07	89.15	89.07	89.09
	0.001	90.18	90.35	90.18	90.23	89	89.32	89	89.06
	0.01	82.92	88.51	82.92	84.32	74.75	87.81	74.75	76.83
	0.1	10.78	1.16	10.78	2.1	10.78	1.16	10.78	2.1
LSTM	1.00E-05	22.12	7.53	22.12	9.58	23.63	20.2	23.63	12.25
	0.0001	28.49	28.21	28.49	23.04	38.75	34.09	38.75	30.07
	0.001	87.76	88.11	87.76	87.82	88.37	88.6	88.37	88.39
	0.01	85.81	85.92	85.81	85.84	43.45	42.18	43.45	41.16
	0.1	10.76	6.95	10.76	2.13	10.78	1.16	10.78	2.1
GRU	1.00E-05	19.15	17.06	19.15	8.33	19.13	11.99	19.13	8.2
	0.0001	22.12	32.8	22.12	17.53	23.19	18.03	23.19	18.36
	0.001	26.32	38.8	26.32	23.43	27.52	39.44	27.52	25.46
	0.01	10.74	1.15	10.74	2.08	10.74	1.15	10.74	2.08
	0.1	10.78	1.16	10.78	2.1	11.06	1.22	11.06	2.2
GRU_CNN	1.00E-05	28.65	30.97	28.65	19.73	28.21	36.91	28.21	21.37
	0.0001	83.29	84.08	83.29	83.3	84.33	85.2	84.33	84.4
	0.001	90.24	90.36	90.24	90.25	81.87	82.85	81.87	81.82
	0.01	10.74	1.15	10.74	2.08	10.74	1.15	10.74	2.08
	0.1	10.78	1.16	10.78	2.1	11.06	1.22	11.06	2.2
LSTM_CNN	1.00E-05	28.08	17.75	28.08	18.04	30.64	26.33	30.64	24.01
	0.0001	84.81	85.23	84.81	84.92	83.76	83.97	83.76	83.82
	0.001	88.37	88.89	88.37	88.46	88.16	88.2	88.16	88.16
	0.01	88.85	89.18	88.85	88.87	81.81	82.23	81.81	81.79
	0.1	10.78	1.16	10.78	2.1	10.78	1.16	10.78	2.1

F1 Score: Balances precision and recall, providing a single performance measure. It is shown in Eq. (36).

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (36)$$

Table IV shows the comparison of baseline models (CNN, LSTM, GRU, and their ensemble) against their respective Dynamic N-gram-enhanced versions. The data distribution ensures even training and testing partitions to allow for fair assessment. Results are reported across different learning rates to gain holistic insights into model performance under various settings.

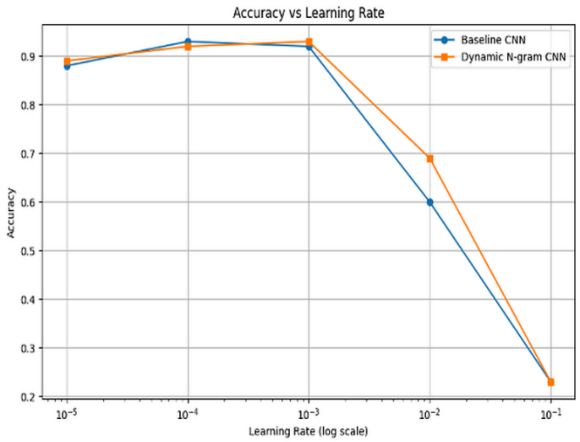

Fig. 10. Accuracy vs learning rate.

Fig. 10. presents the trend in the performance of deep learning models for varying learning rates. It indicates the reaction of baseline and Dynamic N-gram-augmented models to variations in learning rate, comparing the ideal range for successful classification. Dynamic models tend to exhibit enhanced accuracy at smaller learning rates, indicative of improved flexibility in learning fine-grained patterns of Arabic text. This visualization is critical to determine the learning rate that gives the optimal generalization performance and model stability. It also aids in comparing how different architectures (CNN, LSTM, etc.) perform while training, enabling improved model tuning approaches for Arabic NLP tasks.

Table V shows the comparison of baseline and Dynamic N-gram-augmented models over the "Original Without Stem" dataset. The dataset ensures a balanced distribution over categories so that training and testing are strong and unbiased. Performance of models is checked with different learning rates so that their effect on different metrics can be analyzed.

- CNN models outperform others at higher learning rates (e.g., 0.0001 and 0.001).
- Dynamic N-gram improvements demonstrate enhanced performance on lower learning rates for certain models (e.g., GRU, LSTM).
- GRU and GRU-CNN models are less effective compared to their counterparts across all metrics.

TABLE V. PERFORMANCE METRICS COMPARISON: ORIGINAL WITHOUT STEM – BASELINE VS. DYNAMIC N-GRAM MODELS

Model Type	Learning Rate	Baseline Model				Dynamic N-gram Model			
		Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
CNN	1.00E-05	57.92	64.86	57.92	57.24	65.16	66.56	65.16	65.48
	0.0001	89.74	89.84	89.74	89.75	88.5	88.59	88.5	88.53
	0.001	89.2	89.68	89.2	89.29	88.7	89.39	88.7	88.85
	0.01	79.98	87.3	79.98	81.04	75.7	85.21	75.7	77.8
	0.1	10.78	1.16	10.78	2.1	10.8	12.7	10.8	2.14
LSTM	1.00E-05	22.23	8.47	22.23	10.43	23.17	8.83	23.17	10.18
	0.0001	41.62	36.28	41.62	34.06	36.14	42.13	36.14	31.75
	0.001	88.74	89.17	88.74	88.83	89.04	89.43	89.04	89.11
	0.01	84.98	85.21	84.98	84.98	37.67	37.24	37.67	35.28
	0.1	10.78	9.82	10.78	2.2	10.78	1.16	10.78	2.1
GRU	1.00E-05	18.67	21.09	18.67	7.31	18.87	19.93	18.87	7.47
	0.0001	25.47	28	25.47	22.77	22.71	27.37	22.71	20.04
	0.001	28.54	38.98	28.54	27.9	22.15	33.93	22.15	19.66
	0.01	10.74	1.15	10.74	2.08	10.74	1.15	10.74	2.08
	0.1	10.78	1.16	10.78	2.1	11.06	1.22	11.06	2.2
GRU_CNN	1.00E-05	24.19	29.21	24.19	14.62	28.69	31.33	28.69	24.16
	0.0001	78.01	78.95	78.01	77.96	73.7	76.28	73.7	74.14
	0.001	87.37	87.46	87.37	87.33	89.09	89.29	89.09	89.12
	0.01	10.74	1.15	10.74	2.08	10.74	1.15	10.74	2.08
	0.1	10.78	1.16	10.78	2.1	11.06	1.22	11.06	2.2
LSTM_CNN	1.00E-05	26.58	17.87	26.58	18.08	30.52	26.41	30.52	23.1
	0.0001	86.22	86.45	86.22	86.23	88.37	88.47	88.37	88.39
	0.001	87.24	87.76	87.24	87.33	88.48	89.18	88.48	88.63
	0.01	87.39	87.74	87.39	87.37	83.92	84.72	83.92	83.81
	0.1	10.78	1.16	10.78	2.1	10.78	1.16	10.78	2.1

Tables VI and VII summarize performance differences between the baseline and Dynamic N-gram-augmented architectures across the evaluated learning rates.

- Dynamic N-gram models tend to perform better than their baseline equivalents at low learning rates (e.g., 1.00E-05).
- At ideal learning rates (0.0001 and 0.001), basic models (e.g., CNN, GRU-CNN, and LSTM-CNN) provide greater accuracy.
- Stemming and stop-word removal slightly impact recall and precision positively for dynamic N-grams.
- GRU models have lower performance values with and without extensions.
- Dynamic N-gram CNN performs better than CNN using low learning rates (1.00E-05) but lags slightly at mid-range rates (0.0001).
- Dynamic N-gram LSTM always outperforms traditional LSTM in terms of accuracy and recall, especially at 0.0001 and 0.01.
- GRU models like Dynamic N-gram GRU have lower precision in all metrics, indicating minimal usefulness for this set of data.
- LSTM-CNN and Dynamic N-gram LSTM-CNN are equivalent, but Dynamic N-gram LSTM-CNN is marginally better when learning rates are high (0.001).

TABLE VI. PERFORMANCE METRICS COMPARISON: ORIGINAL WITHOUT STOP WORDS – BASELINE VS. DYNAMIC N-GRAM MODELS

Model Type	Learning Rate	Baseline Model				Dynamic N-gram Model			
		Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
CNN	1.00E-05	52.75	55.75	52.75	50.93	57.05	59.95	57.05	56.74
	0.0001	90.02	90.12	90.02	90.04	88.92	89.11	88.92	88.98
	0.001	89.72	90.12	89.72	89.82	88.41	88.84	88.41	88.51
	0.01	84.53	87.47	84.53	84.95	43.32	72.36	43.32	42.77
	0.1	10.78	1.16	10.78	2.1	10.78	1.16	10.78	2.1
LSTM	1.00E-05	23.5	21.45	23.5	14.47	22.99	15.03	22.99	12.9
	0.0001	32.38	32.61	32.38	26.4	34.84	30.4	34.84	27.24
	0.001	86.33	86.35	86.33	86.32	88.02	88.37	88.02	88.07
	0.01	83.02	83.34	83.02	82.99	53.08	55	53.08	52.05
	0.1	10.8	6.95	10.8	2.14	10.78	1.16	10.78	2.1
GRU	1.00E-05	20.91	7.72	20.91	8.81	19.65	9.61	19.65	8.08
	0.0001	23.1	37.43	23.1	19.15	21.45	30.44	21.45	16.01
	0.001	23.36	35.87	23.36	19.82	17.19	20.55	17.19	12.17
	0.01	10.74	1.15	10.74	2.08	10.74	1.15	10.74	2.08
	0.1	10.78	1.16	10.78	2.1	11.06	1.22	11.06	2.2
GRU_CNN	1.00E-05	28.08	36.72	28.08	20.07	24.56	19.6	24.56	16.51
	0.0001	82.18	82.78	82.18	82.3	78.64	79.96	78.64	78.88

	0.001	88.24	88.5	88.24	88.27	85.37	85.72	85.37	85.44
	0.01	10.74	1.15	10.74	2.08	10.74	1.15	10.74	2.08
	0.1	10.78	1.16	10.78	2.1	11.06	1.22	11.06	2.2
LSTM_CNN	1.00E-05	30.15	26.5	30.15	22.71	33.23	24.44	33.23	26.11
	0.0001	85.05	85.2	85.05	85.07	85.35	85.66	85.35	85.43
	0.001	88.42	88.72	88.42	88.46	88.05	88.55	88.05	88.17
	0.01	87.16	87.48	87.16	87.16	83.11	84.16	83.11	83.23
	0.1	10.78	1.16	10.78	2.1	10.78	1.16	10.78	2.1

TABLE VII. PERFORMANCE METRICS COMPARISON: ORIGINAL WITHOUT STOP WORDS AND STEMMING – BASELINE VS. DYNAMIC N-GRAM MODELS

Model Type	Learning Rate	Baseline Model				Dynamic N-gram Model			
		Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
CNN	1.00E-05	61.81	66.71	61.81	61.02	68.81	70.11	68.81	68.79
	0.0001	89.72	89.76	89.72	89.73	89.11	89.25	89.11	89.15
	0.001	89.42	90.01	89.42	89.53	87.85	88.41	87.85	87.95
	0.01	84.53	87.51	84.53	85.19	66.2	87.73	66.2	68.53
	0.1	10.78	1.16	10.78	2.1	10.78	1.16	10.78	2.1
LSTM	1.00E-05	22.12	5.78	22.12	8.97	23.23	21.97	23.23	14.43
	0.0001	33.58	31.94	33.58	28.63	35.62	29.5	35.62	29.14
	0.001	87.68	87.86	87.68	87.73	87.13	87.37	87.13	87.16
	0.01	85.66	85.8	85.66	85.65	63.96	64.93	63.96	62.63
	0.1	10.89	27.56	10.89	2.36	10.78	1.16	10.78	2.1
GRU	1.00E-05	19.73	9.4	19.73	8.16	20.26	6.01	20.26	8.34
	0.0001	24.15	25.71	24.15	20.08	22.86	20.19	22.86	17.93
	0.001	25.1	43.41	25.1	22.17	20.32	26.43	20.32	16.12
	0.01	10.74	1.15	10.74	2.08	10.74	1.15	10.74	2.08
	0.1	10.78	1.16	10.78	2.1	10.78	1.16	10.78	2.1
GRU_CNN	1.00E-05	25.17	30.39	25.17	18.65	27.34	34.11	27.34	22.15
	0.0001	81.14	82.5	81.14	81.37	83.35	83.88	83.35	83.44
	0.001	89.18	89.3	89.18	89.22	84.27	84.53	84.27	84.33
	0.01	10.74	1.15	10.74	2.08	10.74	1.15	10.74	2.08
	0.1	11.06	1.22	11.06	2.2	11.06	1.22	11.06	2.2
LSTM_CNN	1.00E-05	32.27	23.96	32.27	22.71	35.23	40.44	35.23	26.83
	0.0001	84.4	84.59	84.4	84.41	84.89	84.99	84.89	84.87
	0.001	88.11	88.61	88.11	88.21	88.41	88.75	88.41	88.45
	0.01	87.61	88.08	87.61	87.61	83.61	84.83	83.61	83.75
	0.1	10.78	1.16	10.78	2.1	10.78	1.16	10.78	2.1

Table VIII summarizes the performance patterns of the baseline and Dynamic N-gram configurations, particularly for the CNN- and LSTM-based models on this dataset representation.

- Dynamic N-gram improvements systematically enhance the performance of CNN, LSTM, and hybrid models, especially for smaller learning rates.
- GRU architectures do not perform well overall, with small improvements from the Dynamic N-gram method.
- Hybrid configurations such as LSTM-CNN and GRU-CNN are superior with Dynamic N-grams at best learning rates (0.001).

TABLE VIII. PERFORMANCE METRICS COMPARISON: ULTIMATE ARABIC RESULTS – BASELINE VS. DYNAMIC N-GRAM MODELS

Model Type	Learning Rate	Baseline Model				Dynamic N-gram Model			
		Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
CNN	1.00E-05	88.92	88.41	88.92	88.6	91.31	91.4	91.31	91.05
	0.0001	93.33	93.48	93.33	93.34	92.48	92.63	92.48	92.48
	0.001	92.67	92.92	92.67	92.68	92.8	92.9	92.8	92.79
	0.01	80.32	83.57	80.32	80.15	82.97	84.91	82.97	82.8
	0.1	30.51	9.31	30.51	14.26	30.51	9.31	30.51	14.26
LSTM	1.00E-05	39.6	20.92	39.6	24.33	54.73	51.32	54.73	45.53
	0.0001	83.1	82.34	83.1	82.08	93.57	93.56	93.57	93.55
	0.001	93.22	93.22	93.22	93.2	93.22	93.26	93.22	93.24
	0.01	43.27	41.66	43.27	38.78	57.17	54.23	57.17	49.48
	0.1	30.5	9.31	30.5	14.26	30.51	9.31	30.51	14.26
GRU	1.00E-05	33.11	29.89	33.11	27.7	32.28	25.6	32.28	19.39
	0.0001	47.48	51.33	47.48	43.13	40.9	55.86	40.9	38.46
	0.001	35.99	39.31	35.99	28.34	28.64	24.39	28.64	20.77
	0.01	30.51	9.31	30.51	14.26	30.51	9.31	30.51	14.26
	0.1	30.51	9.31	30.51	14.26	30.51	9.31	30.51	14.26
GRU_CNN	1.00E-05	79.34	77.86	79.34	76.78	62.36	61.73	62.36	56.77
	0.0001	93.68	93.6	93.68	93.53	90.06	90.02	90.06	89.9
	0.001	84.24	86.21	84.24	84.21	90.64	90.58	90.64	90.6

	0.01	30.51	9.31	30.51	14.26	30.51	9.31	30.51	14.26
	0.1	30.51	9.31	30.51	14.26	7.49	0.56	7.49	1.04
LSTM_CNN	1.00E-05	87.02	86.19	87.02	86.2	88.67	88.22	88.67	88.39
	0.0001	93.22	93.23	93.22	93.21	92.92	92.9	92.92	92.9
	0.001	92.85	92.95	92.85	92.83	93.32	93.43	93.32	93.33
	0.01	91.63	91.67	91.63	91.6	84.54	84.83	84.54	84.25
	0.1	30.51	9.31	30.51	14.26	30.51	9.31	30.51	14.26

TABLE IX. PERFORMANCE METRICS COMPARISON: ULTIMATE ARABIC PREPOS – BASELINE VS. DYNAMIC N-GRAM MODELS

Model Type	Learning Rate	Baseline Model				Dynamic N-gram Model			
		Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
CNN	1.00E-05	89.4	88.88	89.4	89.1	89.99	89.45	89.99	89.69
	0.0001	93.08	93.08	93.08	93.06	92.2	92.27	92.2	92.17
	0.001	92.35	92.5	92.35	92.32	92.66	92.8	92.66	92.65
	0.01	60.36	63.71	60.36	56.9	69.48	65.03	69.48	65.73
	0.1	23.68	5.61	23.68	9.07	23.68	5.61	23.68	9.07
LSTM	1.00E-05	38.96	16.85	38.96	23.52	37.18	21.64	37.18	21.63
	0.0001	87.19	87.13	87.19	86.65	92.84	92.84	92.84	92.83
	0.001	92.84	92.87	92.84	92.84	92.93	92.99	92.93	92.95
	0.01	87.28	87.17	87.28	87.17	63.43	60.5	63.43	58.77
	0.1	23.68	11.52	23.68	9.07	23.68	5.61	23.68	9.07
GRU	1.00E-05	33.54	34.4	33.54	21.38	33.15	16.79	33.15	20.95
	0.0001	42.6	51.11	42.6	36.75	39.35	40.87	39.35	32.84
	0.001	33.03	36.05	33.03	24.58	27.69	33.46	27.69	19.78
	0.01	29.56	8.74	29.56	13.49	29.56	8.74	29.56	13.49
	0.1	23.68	5.61	23.68	9.07	23.68	5.61	23.68	9.07
GRU_CNN	1.00E-05	78.15	75.29	78.15	75.44	70.26	70.06	70.26	67.25
	0.0001	92.82	92.81	92.82	92.8	89.84	89.9	89.84	89.71
	0.001	84.68	85.18	84.68	84.71	90.44	90.49	90.44	90.39
	0.01	29.56	8.74	29.56	13.49	29.56	8.74	29.56	13.49
	0.1	23.68	5.61	23.68	9.07	7.59	0.58	7.59	1.07
LSTM_CNN	1.00E-05	75.99	74.68	75.99	74.4	86.13	85.93	86.13	85.9
	0.0001	92.48	92.48	92.48	92.47	92.3	92.31	92.3	92.27
	0.001	92.54	92.68	92.54	92.54	93.06	93.16	93.06	93.09
	0.01	29.56	8.74	29.56	13.49	84.46	84.99	84.46	84.19
	0.1	23.68	5.61	23.68	9.07	23.68	5.61	23.68	9.07

Table IX emphasizes the use of Dynamic N-gram methods in tuning model performance on Arabic datasets for various architectures.

Fig. 11 compares the baseline and Dynamic N-gram accuracy values shown for the evaluated architectures. The differences are architecture-dependent rather than uniformly favorable to the enhanced variants: the plotted Dynamic N-gram configurations improve CNN, LSTM, and LSTM-CNN, whereas the baseline GRU and GRU-CNN remain competitive or achieve higher values. The figure therefore illustrates variation across architectures and should not be interpreted as evidence of a consistent gain for every model.

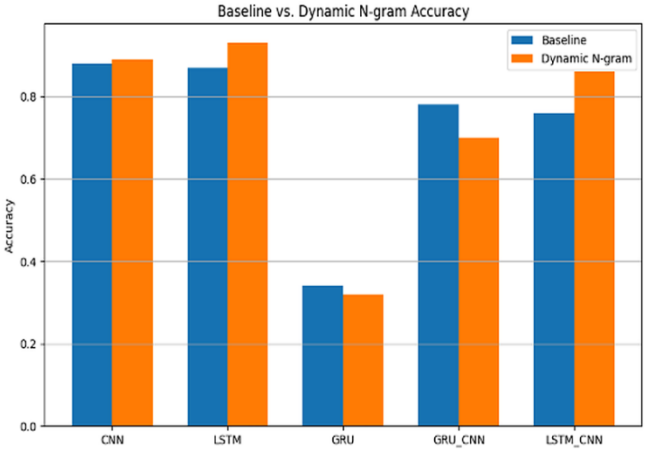

Fig. 11. Baseline vs. dynamic N-gram accuracy.

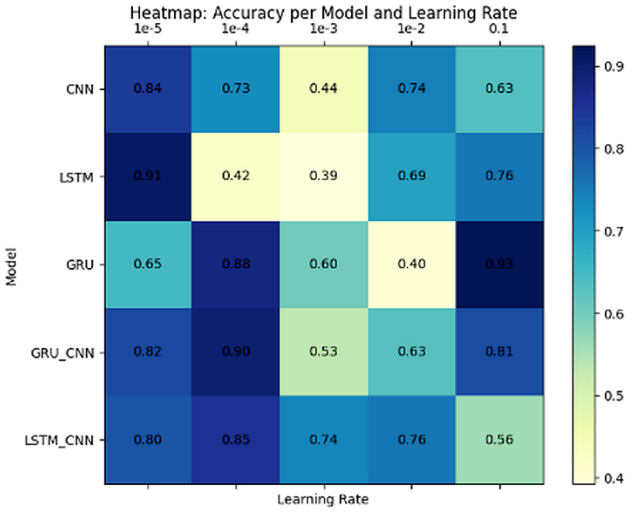

Fig. 12. Accuracy per model and learning rate.

Fig. 12 provides a matrix visualization of how model performance varies at various learning rates. Every cell plots a measure of performance (e.g., accuracy), with color saturation indicating the intensity of the value. The visual graph

immediately reveals good combinations of model types and learning rates. For example, lighter cells might reflect better accuracy, enabling researchers to identify optimal hyperparameter settings. It further allows cross-comparison between baseline and Dynamic N-gram models in a concise manner. By condensing large amounts of experimental data graphically, the heatmap facilitates making sensible decisions regarding model choice and training settings.

Fig. 13 presents the CNN accuracy, precision, recall, and F1-score in a compact multi-metric view. The overlap between the two profiles indicates that the differences are modest and specific to the selected configuration. The chart is used to inspect metric balance within this experiment, not to infer universal robustness or superiority beyond the evaluated dataset and settings.

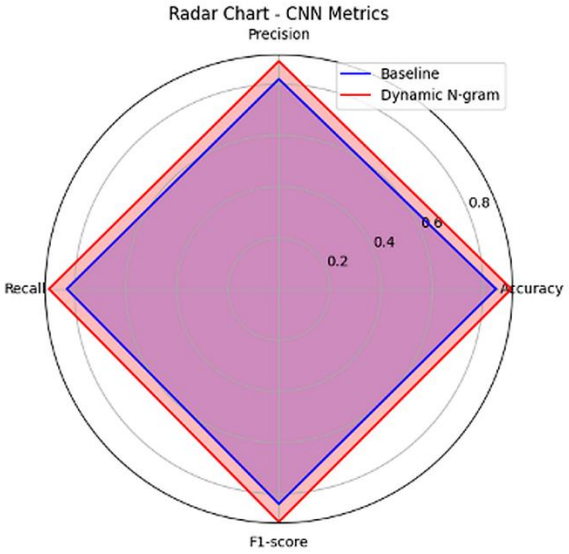

Fig. 13. Radar chart-CNN metrics.

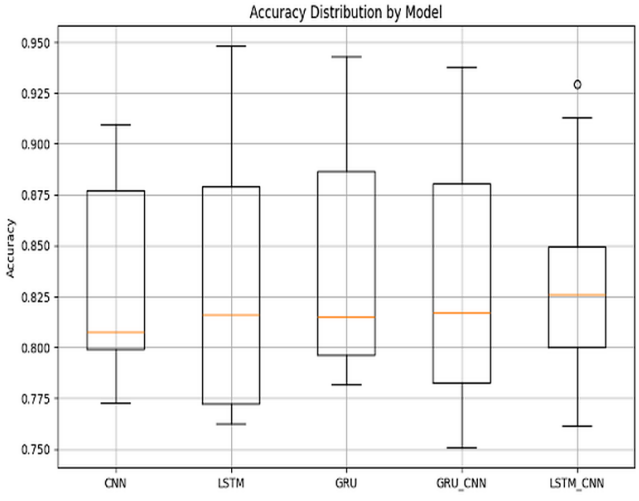

Fig. 14. Accuracy distribution by model.

Fig. 14 plots accuracy score distributions for every model, displaying central tendency and variability with multiple runs. Each box plots the interquartile range, with whiskers and outliers representing score spread. The plot is particularly useful for evaluating model stability and trustworthiness under various initializations or data splits. Models with more compact boxes and larger medians are perceived to be more consistent. It allows researchers to not only find the highest-performing model but also the most reliable one. This matters significantly in actual NLP tasks where consistency in performance over runs is as vital as getting high accuracy.

B. Discussion

The results are interpreted in relation to the four linguistic hypotheses stated in Section III. Because the architecture operates on word tokens, the analysis concerns local token context, longer sequential evidence, and interaction with preprocessing; it does not claim direct modeling of Arabic roots or affixes. The overall pattern is mixed: selected configurations are consistent with the hypotheses, whereas other settings show that the proposed components do not provide uniform gains.

- H1 - Local-context hypothesis: The multi-kernel CNN configuration improved accuracy under selected low-learning-rate conditions, but the baseline CNN matched or exceeded it in several mid-range conditions. Thus, short multi-scale token context can be useful, but fixed windows of two to four words are not consistently superior across all settings.
- H2 - Sequential-context hypothesis: Dynamic N-gram LSTM achieved the strongest result among the enhanced models under its best configuration, which is consistent with the expectation that longer sequential context can complement local features. However, the advantage was not maintained at every learning rate, and both GRU configurations performed substantially below the strongest CNN- and LSTM-based settings. The evidence is therefore partial and architecture-dependent.
- H3 - Complementarity hypothesis: The hybrid results were also mixed. The enhanced LSTM-CNN reached a strong value at a selected learning rate, whereas baseline hybrid models matched or exceeded their enhanced counterparts in other configurations. These findings suggest that local and sequential representations can be complementary, but they do not establish a universal hybrid advantage or isolate the contribution of each component without an ablation study.
- H4 - Preprocessing-interaction hypothesis and learning-rate sensitivity: Relative model rankings changed across dataset representations and learning rates. This pattern is consistent with the possibility that stemming and stop-word removal alter the lexical and syntactic cues available to each architecture, but the present design cannot determine which specific linguistic information was retained or removed. Learning rate was also a major determinant of convergence: the largest rate generally produced unstable or near-chance results, while moderate and smaller rates yielded the strongest values for different architectures.

Taken together, the results support conditional rather than universal versions of the proposed hypotheses. Multi-scale local context, recurrent sequence modeling, and their combination can provide competitive benefits in selected settings, but the observed gains remain configuration-dependent. Because the implementation is word-based and no kernel-specific or component-level ablation was conducted, the results should not be interpreted as evidence of direct morphological analysis or as causal proof that any single architectural component produced the improvement.

Table X provides a contextual rather than a direct benchmark comparison. The reported values were obtained using different corpora, numbers of classes, preprocessing pipelines, data partitions, task definitions, and evaluation protocols. Therefore, a numerical ranking across these studies would be methodologically inappropriate. The present results are consequently described as competitive within the Ultimate Arabic News Dataset setting. A controlled comparison that reimplements external methods on the same corpus and split is required before any state-of-the-art claim can be supported.

TABLE X. CROSS-STUDY PERFORMANCE CONTEXT FOR ARABIC NEWS CLASSIFICATION

Study / Method	Dataset / Setting	Reported Accuracy / Comparison Note
Hossain et al. [16] — ABTM	Arabic news corpus; transformer with TF-IDF/BOW	97.69%; different corpus, preprocessing, and evaluation protocol
Zamzami et al. [17] — ML/DL	Multiple country-of-origin Arabic news datasets	Up to 94%; different target task and corpora
Imad et al. [18] — CNN	CNN, BBC, and OSAC datasets	94.47%, 90.97%, and >98.8%; dataset-dependent and not directly comparable
Jamaleddyn et al. [20] — tuned ML	CNN Arabic dataset	95.16%; different dataset and tuning protocol
Alzaidi et al. [21] — TCAODL-ANA	Simulated Arabic news dataset	95.48%; different dataset and optimization procedure
Alroobaea [22] — CNN-LSTM	5,000 Arabic news articles in six categories	93.15%; different dataset size, classes, and split
Present study — Dynamic N-gram LSTM	Ultimate Arabic News Dataset; 10 classes; 80/20 split	93.57%; competitive within the present protocol; no direct state-of-the-art claim

C. Limitations

Computational efficiency was not evaluated in the present experiments. Consequently, the study does not report parameter count, FLOPs, training time, inference latency or throughput, model size, or peak CPU memory consumption. The reported comparisons therefore support conclusions about classification performance only and cannot establish whether the observed accuracy differences justify any additional computational overhead in real-world deployment settings. In addition, the word-level tokenization used in the experiments does not directly model Arabic roots, patterns, clitics, or other within-word morphological structure. The linguistic explanations are therefore hypothesis-based interpretations of the architecture and observed performance patterns rather than causal evidence, and kernel-specific or component-level ablation experiments were not conducted.

V. CONCLUSION AND FUTURE WORKS

The study examined the usefulness of applying Dynamic N-gram augmentation to CNN, LSTM, GRU, and hybrid deep learning models for Arabic text classification. The findings showed that multi-kernel word-level context can improve selected architectures under particular learning-rate and preprocessing settings, while the magnitude and direction of the difference vary by model. Among the tested enhanced models, Dynamic N-gram LSTM and Dynamic N-gram LSTM-CNN achieved 93.57% and 93.32% accuracy, respectively, under their best configurations. Interpreted against the stated hypotheses, the results provide conditional support for combining short token-context features with longer sequential evidence, but they do not demonstrate uniform superiority or direct modeling of Arabic morphology. The regular GRU performed less effectively than other architectures, and its enhanced variant produced only limited gains. These findings demonstrate competitive performance within the present dataset and evaluation protocol. They should not be interpreted as state-of-the-art superiority because the studies reviewed in Section II used different corpora, class configurations, preprocessing pipelines, and experimental splits. Future work should reproduce representative external baselines, including transformer-based methods, on the same data partition and evaluation protocol. It should also evaluate character-level and subword tokenization, morphology-aware representations, and controlled ablations comparing individual kernel sizes 2, 3, and 4 with their combined configuration. Such experiments would test the linguistic hypotheses more directly and distinguish local-context effects from model-capacity effects. Future work should additionally conduct controlled computational profiling on common hardware by reporting parameter count, FLOPs, training time, inference latency and throughput, model size, and peak CPU memory consumption for each baseline and enhanced architecture. Such profiling is necessary to quantify the accuracy-efficiency trade-off and assess real-world deployment feasibility.

REFERENCES

[1] Z. Alyafeai, M. S. Al-shaibani, M. Ghaleb, and I. Ahmad, “Evaluating various tokenizers for Arabic text classification,” Neural Process. Lett., vol. 55, no. 3, pp. 2911–2933, 2023.

[2] H. El Rifai, L. Al Qadi, and A. Elnagar, “Arabic text classification: The need for multi-labeling systems,” Neural Comput. Appl., vol. 34, no. 2, pp. 1135–1159, 2022.

[3] M. Bounhas, B. Elayeb, A. Chouigui, A. Hussain, and E. Cambria, “Arabic text classification based on analogical proportions,” Expert Syst., vol. 41, no. 10, p. e13609, 2024.

[4] A. Y. Muaad et al., “An effective approach for Arabic document classification using machine learning,” Glob. Transit. Proc., vol. 3, no. 1, pp. 267–271, 2022.

[5] T. Sabri, S. Bahassine, O. El Beggar, and M. Kissi, “An improved Arabic text classification method using word embedding,” Int. J. Electr. Comput. Eng., vol. 14, no. 1, 2024.

[6] A. Shehata, M. N. Al-Suqri, N. E. Elshaiekh, F. Hamad, Y. N. AlHusaini, and A. Mahfouz, “ArabFake: A multitask deep learning framework for Arabic fake news detection, categorization, and risk prediction,” IEEE Access, 2024.

[7] H. Himdi, G. Weir, F. Assiri, and H. Al-Barhamtoshy, “Arabic fake news detection based on textual analysis,” Arab. J. Sci. Eng., vol. 47, no. 8, pp. 10453–10469, 2022.

[8] F. Fkih, M. Alsuhaibani, D. Rhouma, and A. M. Qamar, “Novel machine learning-based approach for Arabic text classification using

stylistic and semantic features," Comput. Mater. Contin., vol. 75, no. 3, 2023.

[9] O. Hamed and N. Zaidkilani, "Arabic topic classification corpus of the Nakba short stories," in Proc. 1st Int. Workshop Nakba Narratives as Language Resources, 2025, pp. 48–55.

[10] L. H. Baniata and S. Kang, "Transformer text classification model for Arabic dialects that utilizes inductive transfer," Mathematics, vol. 11, no. 24, p. 4960, 2023.

[11] M. Azzeh, A. Qusef, and O. Alabboushi, "Arabic fake news detection in social media context using word embeddings and pre-trained transformers," Arab. J. Sci. Eng., vol. 50, no. 2, pp. 923–936, 2025.

[12] D. Refai, S. Abu-Soud, and M. J. Abdel-Rahman, "Data augmentation using transformers and similarity measures for improving Arabic text classification," IEEE Access, vol. 11, pp. 132516–132531, 2023.

[13] A. M. E. Koshiry, E. H. I. Eliwa, T. Abd El-Hafeez, and A. Omar, "Arabic toxic tweet classification: Leveraging the AraBERT model," Big Data Cogn. Comput., vol. 7, no. 4, p. 170, 2023.

[14] H. A. G. Elsayed, S. Chaffar, S. B. Belhaouari, and H. Raissouli, "A two-level deep learning approach for emotion recognition in Arabic news headlines," Int. J. Comput. Appl., vol. 44, no. 7, pp. 604–613, 2022.

[15] S. M. Alzanin, A. M. Azmi, and H. A. Aboalsamh, "Short text classification for Arabic social media tweets," J. King Saud Univ. Comput. Inf. Sci., vol. 34, no. 9, pp. 6595–6604, 2022.

[16] M. M. Hossain, M. S. Hossain, M. Safran, S. Alfarhood, M. Alfarhood, and M. Mridha, "A hybrid attention-based transformer model for Arabic news classification using text embedding and deep learning," IEEE Access, 2024.

[17] N. Zamzami, H. Himdi, and S. F. Sabbeh, "Arabic news classification based on the country of origin using machine learning and deep learning techniques," Appl. Sci., vol. 13, no. 12, p. 7074, 2023.

[18] J. Imad, A. Rachid, and B. Mohamed, "Automated Arabic news classification using the convolutional neural network," Int. J. Electr. Eng. Inform., vol. 15, no. 2, 2023.

[19] N. A. Othman, D. S. Elzanfaly, and M. M. M. Elhawary, "Arabic fake news detection using deep learning," IEEE Access, 2024.

[20] I. Jamaleddyn, M. Biniz, and others, "An improved approach to Arabic news classification based on hyperparameter tuning of machine learning algorithms," J. Eng. Res., vol. 11, no. 2, p. 100061, 2023.

[21] M. S. A. Alzaidi et al., "Enhanced automated text categorization via Aquila optimizer with deep learning for Arabic news articles," Ain Shams Eng. J., vol. 16, no. 1, p. 103189, 2025.

[22] R. Alroobaea, "An empirical deep learning approach for Arabic news classification," Int. J. Adv. Comput. Sci. Appl., vol. 14, no. 6, 2023.

[23] A. H. Al-Dulaimi, "Ultimate Arabic News Dataset," vol. 2, Jul. 2022, doi: 10.17632/jz56k5wxz7.2.