ÉCOLE NATIONALE SUPÉRIEURE D'ARTS ET MÉTIERS
UNIVERSITÉ HASSAN II DE CASABLANCA

Exploring the Impact of Contextual Attention on Arabic Text Classification

Submitted by:

Mr. OUAAZIZ Mouhcine

Supervisor:

Mr. BAHASSINE Said

ENSAM Casablanca
Master Big Data and IoT
2023-2024

Figure Table

Table of Content

I. Introduction

The ever-growing field of Natural Language Processing (NLP) relies heavily on accurate text classification for tasks like sentiment analysis, topic modeling, and spam filtering. Arabic text classification presents a unique challenge due to its right-to-left writing system, complex morphology (word structure), and lack of extensive resources compared to other languages.

Classification is one of the key tasks to enable automation in text processing. In text classification, the model actively used today is Long-Short Term Memory (LSTM). LSTM is a type of a Recurrent Neural Network (RNN) that can learn long-term dependencies used in the field of deep learning. LSTM networks are commonly used for time series data. Therefore, it is also well known in text data processing, which we can consider as a time series. However, one of the biggest disadvantages of LSTM is that it takes up a lot of memory and time during processing. Secondly, these models have processing problems if there is a large distance between the point to be reached and the current point [1].

The transformer model has been developed to overcome the disadvantages of RNN models and to keep up with today's data density. Most previous methods perform language processing using a recurrent approach. Contrary to other methods, the transformer avoids the recurrent approach by using multi-head self-attention and reaches to a result more quickly and successfully [2]. In this study, we investigate the importance of the attention mechanism to better classify news articles from a public dataset founded on a public website. We compared the AraBERT with the LSTM and CNN models in the text data classification we obtained.

In the second section, related works are explored. We describe our research methodology including data preparation and the chosen models in the third section. The experimental results will be discussed in the fourth section followed by interpretation of results found and concluded with a conclusion in the last section.

II. Related Works

In recent research exploring the impact of contextual attention on Arabic text classification, several notable studies have emerged. *Shammur A Chowdhury et al. (2020)* [3] presents a study on Arabic influencer Twitter datasets, comparing the effectiveness of transformer models trained on a mixture of formal and informal text to models trained exclusively on formal text for text classification. The results show that pre-training a BERT model on a mixture of formal and informal text leads to greater generalization power and improved classification effectiveness in Arabic text categorization. In their 2021 paper, *Soyalp et al.* [4] explore the efficacy of Transformer models in enhancing text classification tasks, demonstrating notable improvements through advanced attention mechanisms and word embeddings. In his systematic review, *Alammary* (2022) [5] evaluates various BERT models tailored for Arabic text classification, comparing their performances and highlighting areas for further improvement and future research. These studies collectively underscore the importance of contextual attention in advancing the field of Arabic text classification.

III. Methodology

1. Architecture Adapted

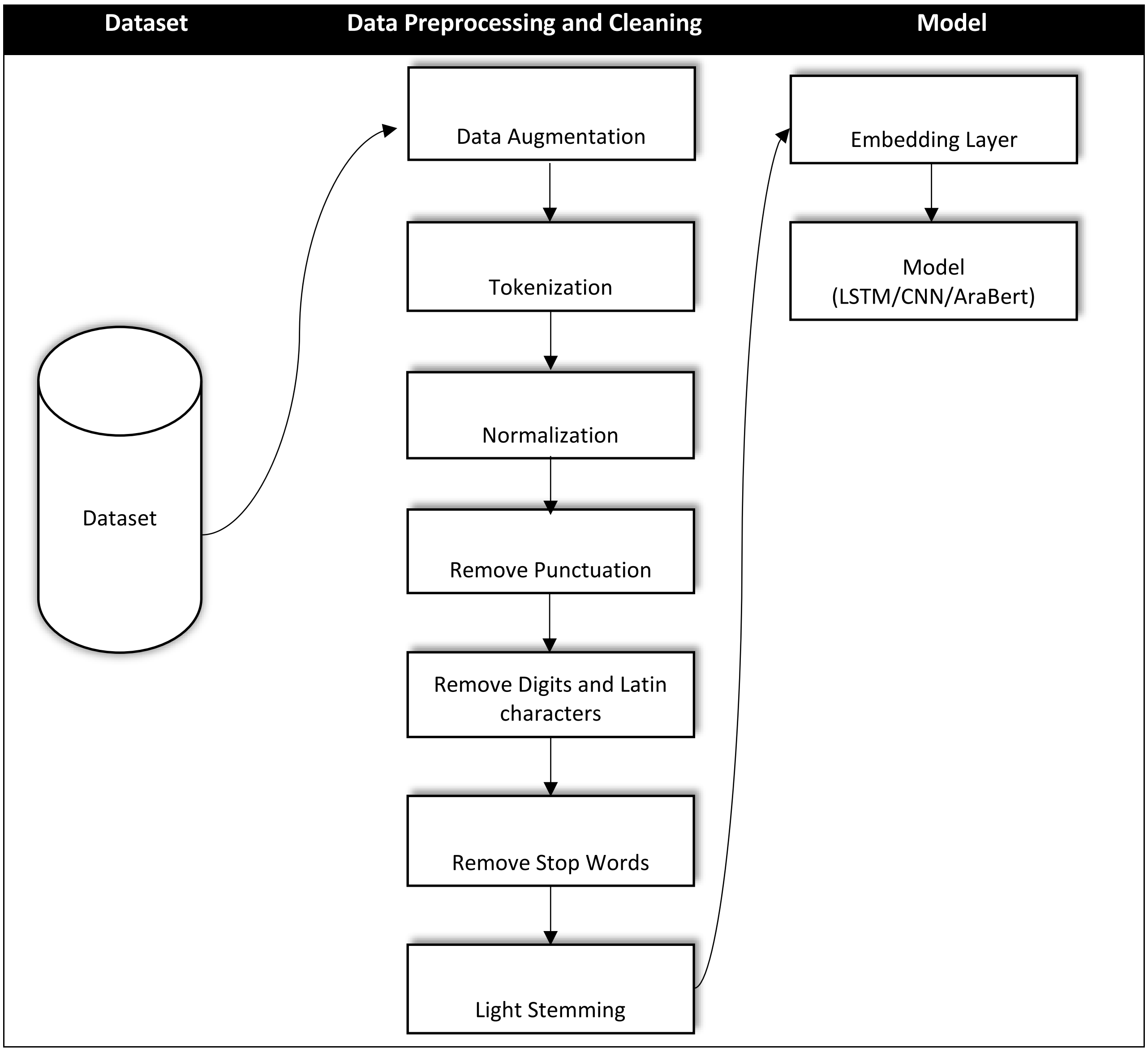


Figure 1 Architecture adapted

2. Dataset

In our study, we utilized two datasets, the first being an Arabic news dataset and the other is an Arabic sentiment analysis one. The former dataset is sourced from online sources, specifically focusing on Arabic news articles from BBC Arabic and CNN Arabic categorized into folders named after their respective categories. The dataset for BBC news was obtained from the "OSAC: Open-Source Arabic Corpus," as documented in the paper by Motaz K. Saad and Wesam Ashour, presented at the 6th ArchEng International Symposiums, EEECS’10 the 6th International Symposium on Electrical and Electronics Engineering and Computer Science, held at the European University of Lefke, Cyprus, in 2010. Similarly, the dataset for CNN news was also acquired from the same source and publication. This dataset served as the foundation for our analysis and experimentation, enabling us to explore and evaluate the performance of various models for classifying Arabic text.

The second dataset is a dataset that has been collected in April 2019. It contains 58K Arabic tweets annotated in positive and negative labels. The dataset is balanced and collected using positive and negative emojis lexicon.

3. Data Cleaning and Preprocessing

Before performing any data cleaning or preprocessing techniques, we had first to turn the dataset itself into a more suitable and easy-to-work-with form. The dataset comprises a collection of Arabic news articles categorized into folders named after their respective categories. We had to turn it into a tabular form, specifically a .CSV form.

Text pre-processing, which is the next step, converts the Arabic text to a form that is suitable for a text classification system. These pre-processing tasks include Punctuations removal, Latin Characters removal, Stop-Words removal, Digits removal, Tokenization, Normalization, Light Stemming, and Data Augmentation.

The pre-processing step consists of the following steps:

- Data Augmentation

To address the imbalance in specific categories like sport, business, and entertainment in our merged BBC News and CNN News datasets, we employed a data augmentation strategy. Initially, we fused the datasets to consolidate our training data. Recognizing the need for more samples in certain categories, we leveraged a two-step process. First, we summarized each article using a pre-trained model known as csebuetnlp/mT5_multilingual_XLSum. Then, we utilized the output of this summarization as input for another pre-trained model, bigscience/bloom-1b1. This approach allowed us to effectively augment our dataset by generating additional instances for underrepresented categories, thereby mitigating the imbalance, and enhancing the robustness of our training data.

The idea of augmenting our data emerged after conducting exploratory analysis on our dataset and discovering startling results: a single class dominated over half of the dataset. Additionally, after training and testing two deep learning models, LSTM and CNN, we observed significant instability in the results, with performance metrics varying widely between runs. Accuracy, for instance, fluctuated from 70% to 99%. To address the imbalance issue and achieve a more balanced dataset, we decided to implement data augmentation, aiming for improved and more consistent results.

- Tokenization

Tokenization is a method for dividing texts into tokens; Words are often separated from each other by blanks (white space, semicolons, commas, quotes, and periods). These tokens could be individual words (noun, verb, pronoun, and article, conjunction, preposition, punctuation, numbers, and alphanumeric) that are converted without understanding their meaning or relationships. The list of tokens becomes an input for further processing. In this work, we use “one_hot” from Keras, which is the Python Deep Learning library.

- Stop Word Removal

Stop word removal involves the elimination of insignificant words, such as (since/ منذ) and (for/ لأجل) and (so/ لذا).

Other examples of these insignificant words are articles, conjunctions, pronouns (such as he/هو and she/ هي and they/ هم), prepositions (such as from/ من, to/إلى, in/في, and about/حول), demonstratives, (such as this/هذا, these/هؤلاء, and there/هناك), and interrogatives (such as where/أين, when/متى, and whom/من). Moreover, Arabic circumstantial nouns indicating time and place (such as after/بعد, above/فوق, and beside/بجانب).

- Punctuations Removal

Punctuations Removal aims to remove the punctuations symbols such as {#,-,_,.,,,;,:,‘}, because these symbols are not useful in our approach.

- Latin Characters Removal and Digits Removal

In addition, we remove the Latin characters and digits, which appear in texts and do not have any meaning or indications.

- Word Normalization

Normalization aims to normalize certain letters that have different forms in the same word to one form.

- Light Stemming

Light stemming is the affix removal approach that refers to a process of stripping off a small set of prefixes and/or suffixes to find the root of the word. In this work, we used the Information Science Research Institute’s (ISRI) stemmer. It uses a similar algorithm to word rooting of the Khoja stemmer. However, it does not employ a root dictionary for lookup. In addition, if a word cannot be rooted, it is normalized by the ISRI stemmer (e.g., removing certain determinants and end patterns) instead of leaving the word unchanged. Furthermore, it defines a set of diacritical marks and affix classes. For Example : (ضم - للضمان).

4. Model Creation

During the project, we aimed to classify Arabic text using three distinct models: Long Short-Term Memory (LSTM), Convolutional Neural Network (CNN), and AraBERT. Our primary objective was to investigate the influence of attention mechanisms on the classification of Arabic text. By employing these models, we sought to discern how attention mechanisms enhance the accuracy and efficacy of text classification in Arabic, thereby shedding light on the most effective approach for this task.

Long short-term memory (LSTM) is a type of recurrent neural network (RNN) architecture designed to address the vanishing gradient problem encountered in traditional RNNs. LSTMs can learn long-term dependencies in sequential data by incorporating memory cells and gates to regulate the flow of information. These memory cells enable LSTMs to retain information over extended time periods, making them particularly effective for tasks involving sequential data such as natural language processing, time series prediction, and speech recognition.

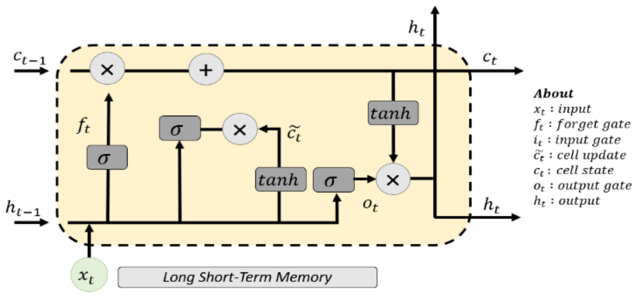


Figure 2 LSTM

Convolutional Neural Networks (CNN) are a class of deep learning models commonly used for tasks involving image recognition, classification, and computer vision. CNNs are inspired by the organization of the visual cortex in animals and are designed to learn spatial hierarchies of features automatically and adaptively from input images. They employ convolutional layers to extract local features, pooling layers to reduce dimensionality and preserve spatial invariance, and fully connected layers for classification. CNNs have demonstrated remarkable success in various computer vision tasks, including object detection, image segmentation, and facial recognition.

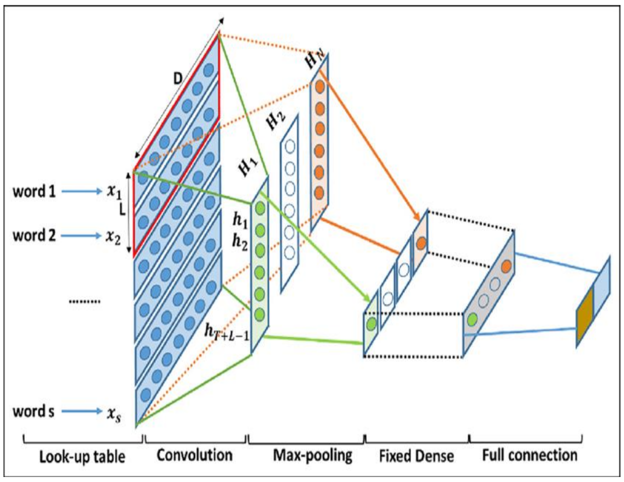


Figure 3 CNN

AraBert is a pre-trained language model specifically tailored for the Arabic language. It is based on the Bidirectional Encoder Representations from Transformers (BERT) architecture, which is renowned for its ability to capture contextual relationships in language. AraBERT is trained on a large corpus of Arabic text and can understand and generating Arabic text with high accuracy. This pre-trained model serves as a powerful foundation for various natural language processing tasks in Arabic, including text classification, named entity recognition, sentiment analysis, and machine translation. AraBERT has significantly advanced the state-of-the-art in Arabic natural language processing and is widely used in both research and industry applications.

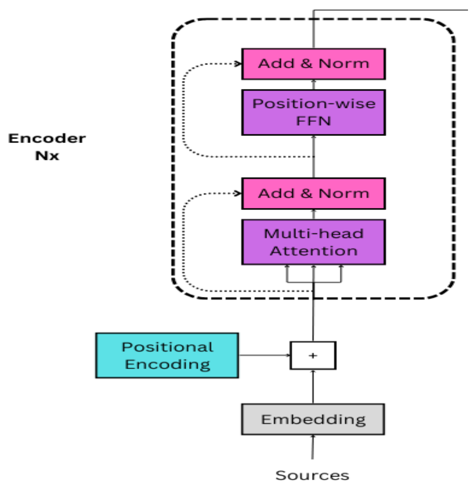


Figure 4 AraBert

5. Evaluation Metrics

To evaluate the performance of the news classification system, we have employed four well-known metrics: Accuracy, Precision, Recall, and F1-score. These metrics are widely used to measure the effectiveness of classification models, especially in multilabel scenarios where each news article can belong to multiple categories.

- **Accuracy:** The proportion of correctly classified instances among all instances.
- **Precision:** The proportion of relevant instances among the retrieved instances for each category.
- **Recall:** The proportion of relevant instances that were successfully retrieved for each category.
- **F1-score:** The harmonic mean of Precision and Recall, providing a single measure that balances both.

For multilabel classification, the metrics are calculated as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

Figure 5 Evaluation metrics formulas

- **True Positive (TP):** The number of instances correctly predicted as belonging to a specific category.
- **False Positive (FP):** The number of instances incorrectly predicted as belonging to a specific category.
- **True Negative (TN):** The number of instances correctly predicted as not belonging to a specific category.
- **False Negative (FN):** The number of instances incorrectly predicted as not belonging to a specific category.

IV. Implementation and Results

1. Data Splitting

Prior to model training, we partitioned the dataset into training and testing subsets, ensuring an 80-20 split for each category. This separation was achieved using the 'train_test_split' function from the scikit-learn library, maintaining the integrity of each category without mixing them during the splitting process. Specifically, 80% of the data from each category was designated for training purposes, while the remaining 20% was reserved for testing. This meticulous splitting strategy not only guarantees that the model receives enough training data but also provides a distinct evaluation set to assess its performance independently. <u>We opted for an 80-20 split, allocating 80% of the dataset for training and reserving 20% for testing, as these percentages are commonly considered optimal for balancing model performance and evaluation.</u>

2. LSTM

a) Model Architecture and Training Details

The LSTM model's architecture included an embedding layer, two bidirectional LSTM layers with dropout for regularization, and a dense layer with ‘softmax’ activation for classification. The model was compiled with categorical ‘crossentropy’ loss and the Adam optimizer, and trained for 25 epochs with a batch size of 32. We used ‘ModelCheckpoint’ to save the best model, ‘EarlyStopping’ to halt training when improvements ceased, and ‘ReduceLROnPlateau’ to reduce the learning rate upon performance plateaus. <u>Since we are utilizing callbacks during model training, specifically to halt training when performance stops improving, the selection of the number of epochs becomes less critical, provided it is not unreasonably low. Consequently, we have settled on 25 epochs as a suitable choice.</u>

The results of the LSTM model training are shown in the provided plots. The training and validation accuracy increased steadily over the epochs, reaching approximately 82% and 78% respectively. This indicates that the model was learning effectively without significant overfitting, as evidenced by the similar trends in both training and validation accuracy. The training and validation loss curves further support this observation, showing a general downward trend with some fluctuations, particularly in validation loss.

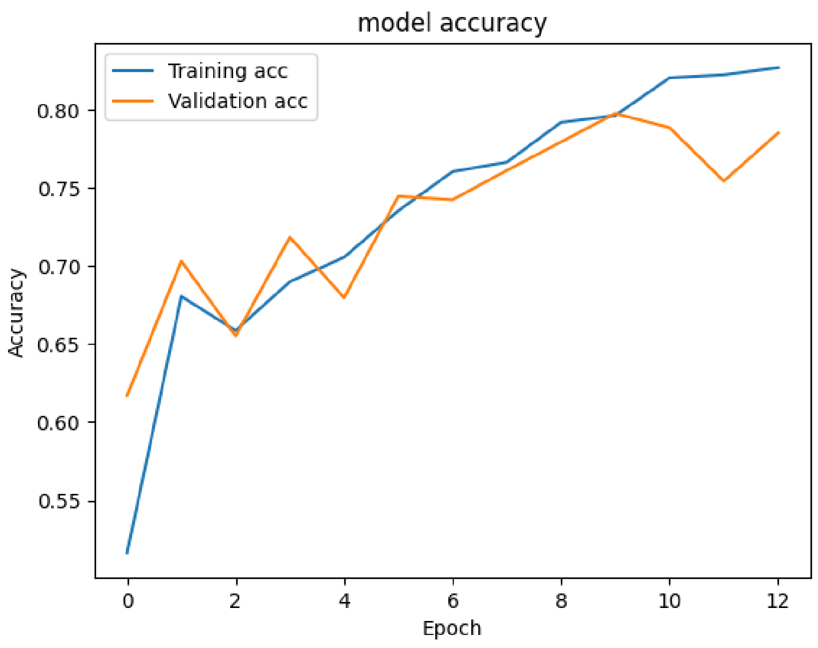


Figure 6 LSTM model accuracy over epochs

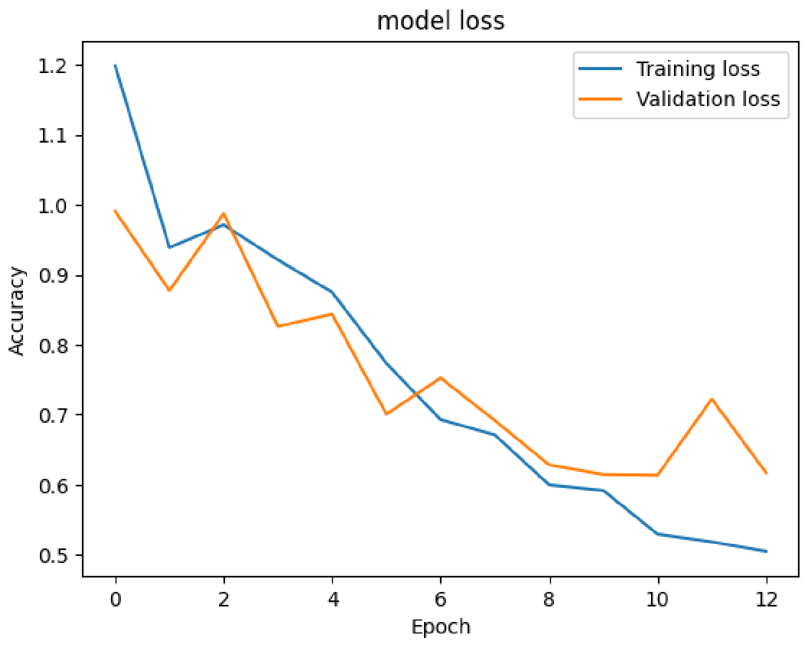


Figure 7 LSTM model loss over epochs

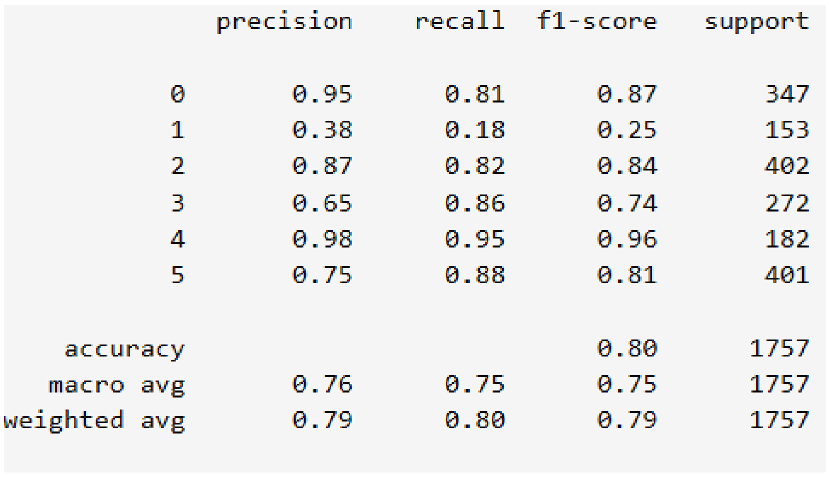
b) Model Performance Analysis

The confusion matrix and classification report provide a detailed breakdown of the model's performance across different categories. The overall accuracy of the LSTM model was 80%. The precision, recall, and F1-scores varied significantly across categories. For instance, the 'Business' and 'World' categories achieved high precision (0.95 and 0.98, respectively) and recall (0.81 and 0.95, respectively), indicating strong performance in these areas. In contrast, the 'Entertainment' category had much lower precision (0.38) and recall (0.18), suggesting that the model struggled to accurately classify these articles.

There are several potential reasons for the poor performance of the model on the 'Entertainment' category, including lexical overlap, the broad scope and ambiguous boundaries of the 'Entertainment' category, and dataset imbalance. Entertainment news often shares terms and phrases with other categories like world news and business, and the context in which these words appear may not be distinct enough for the model to differentiate. Additionally, the 'Entertainment' category encompasses a wide range of topics, from movies and music to celebrity gossip and cultural events, increasing the chances of semantic similarity with categories like 'World' or 'Business' news. Furthermore, the 'Entertainment' category has fewer samples compared to others, thus the model may not effectively learn its distinct features, contributing to the difficulty in accurate categorization.

Contrary to the 'Entertainment' category, the 'Sport' category is the least misclassified. This is because its terminology is unique and rarely found in other categories. For example, terms like "referee - حكم," "net - شباك," and "player - لاعب" are specific to sports and not commonly used elsewhere.

The macro average F1-score was 0.75, indicating moderate overall performance with some variability across categories. The weighted average F1-score was slightly higher at 0.79, reflecting the influence of categories with larger support (number of instances).

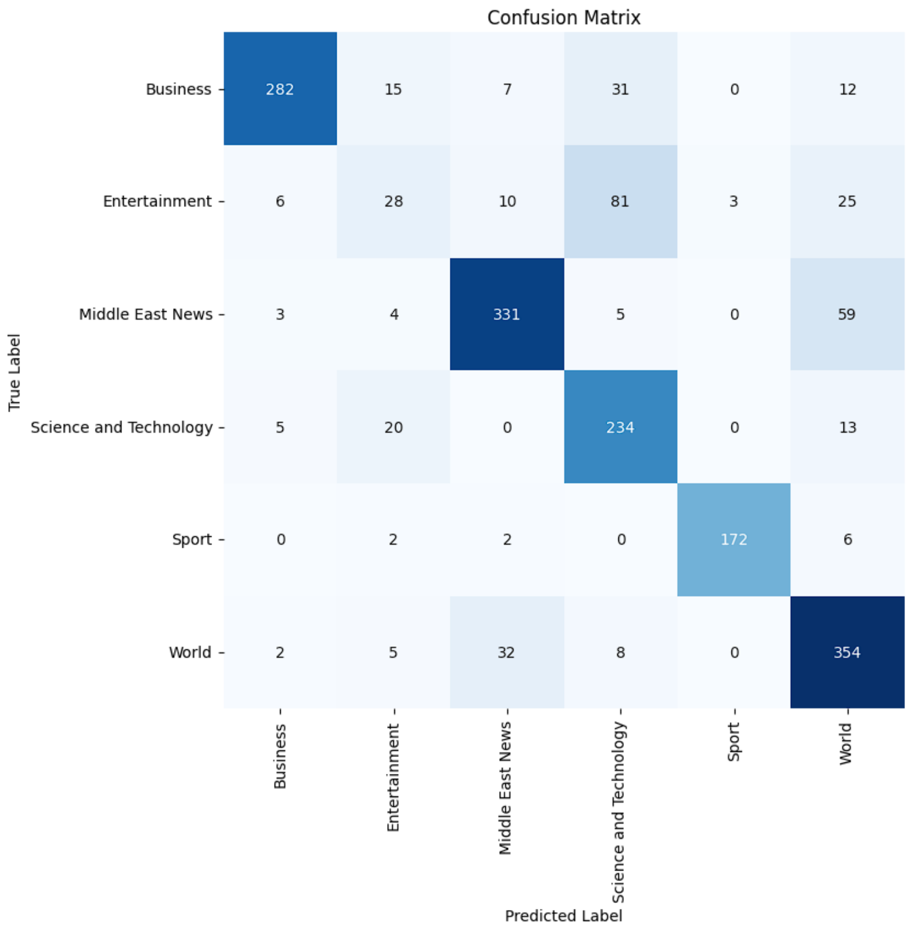


```
              precision    recall  f1-score   support

           0       0.95      0.81      0.87       347
           1       0.38      0.18      0.25       153
           2       0.87      0.82      0.84       402
           3       0.65      0.86      0.74       272
           4       0.98      0.95      0.96       182
           5       0.75      0.88      0.81       401

    accuracy                           0.80      1757
   macro avg       0.76      0.75      0.75      1757
weighted avg       0.79      0.80      0.79      1757
```

Figure 8 LSTM model classification report



True Label \ Predicted Label	Business	Entertainment	Middle East News	Science and Technology	Sport	World
Business	282	15	7	31	0	12
Entertainment	6	28	10	81	3	25
Middle East News	3	4	331	5	0	59
Science and Technology	5	20	0	234	0	13
Sport	0	2	2	0	172	6
World	2	5	32	8	0	354

Figure 9 LSTM model confusion matrix

c) Testing the Model on a Different Dataset

The LSTM model was trained on an Arabic sentiment analysis dataset to classify text as either positive or negative sentiment. The training and validation results over 20 epochs are presented as follows:

During the training process, the model's accuracy steadily improved, reaching 96.30% by the twentieth epoch. Concurrently, the loss decreased from 0.6707 to 0.0958, indicating convergence towards the optimal solution. However, the validation accuracy fluctuated between 60% and 64%, suggesting potential overfitting or limitations in the model's ability to generalize to unseen data.

The LSTM model achieved an overall accuracy of 64% on the sentiment analysis task, demonstrating moderate performance in distinguishing between positive and negative sentiments in Arabic text. Both precision and recall values for positive and negative sentiments were relatively balanced, with precision scores around 0.64 and recall scores of 0.69 and 0.58, respectively. This indicates that the model was equally effective in correctly classifying both sentiment categories.

Furthermore, the F1-scores, which combine precision and recall, were 0.66 for positive sentiment and 0.61 for negative sentiment. These scores suggest reasonable overall performance in capturing sentiment characteristics from the text data. However, there is still room for improvement.

In conclusion, The LSTM model demonstrated moderate success in classifying Arabic text sentiment, achieving an accuracy of 64%. While the model performed reasonably well in distinguishing between positive and negative sentiments, there is room for improvement, particularly in achieving more consistent validation accuracy over training epochs.

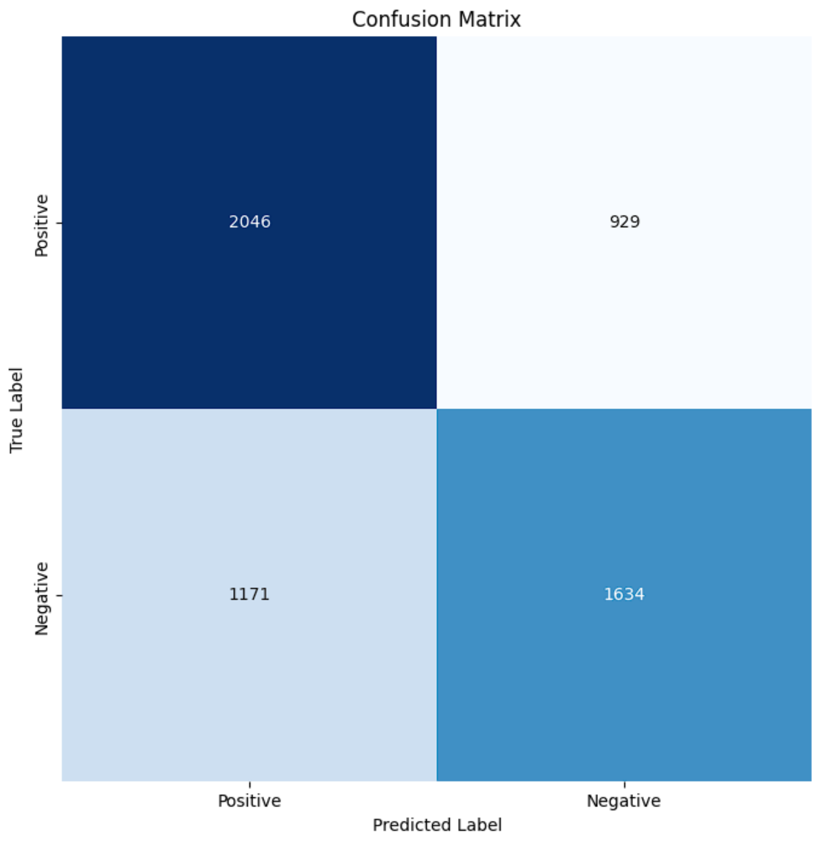


Figure 10 LSTM model confusion matrix (2)

3. CNN

a) Model Architecture and Training Details

The model consists of an Embedding layer, a 1D convolutional layer with max pooling, two dense layers with dropout, and a 'softmax' output layer. The model is compiled with the Adam optimizer, categorical cross-entropy loss, and accuracy metric. The training process involves fitting the model to the training data for 25 epochs, with a batch size of 64, using validation data for monitoring performance. Three callbacks are employed: Model Checkpoint for saving the best weights, Early Stopping for preventing overfitting, and Reduce Learning Rate on Plateau for adjusting the learning rate. The training history is stored for further analysis or visualization.

Epoch Performance:

- Epoch 1: The model started with a training accuracy of 32.19% and validation accuracy of 80.08%. This significant difference indicates that the model quickly learned some useful features from the data.
- Epochs 2-5: The model continued to improve, reaching a training accuracy of 52.90% and a validation accuracy of 87.19% by Epoch 5.
- Epochs 6-10: Although there were fluctuations, the model maintained a validation accuracy above 86% and achieved a peak validation accuracy of 90.15% at Epoch 7

b) Model Performance Analysis

The CNN model achieved an overall accuracy of 80% and a weighted average F1-score of 0.79, demonstrating strong performance across most categories. It excelled in classifying business-related texts (F1-score of 0.87), Middle East news (F1-score of 0.84), and global news (F1-score of 0.81). However, the model struggled with the 'Entertainment' category, achieving a low F1-score of 0.25 due to vocabulary overlap and misclassifications. The 'Science and Technology' category had a high recall but lower precision, resulting in an F1-score of 0.74, indicating some false positives. While the model performed well overall, the 'Entertainment' category remained challenging, likely due to its broad scope and ambiguous boundaries as well as having less data samples compared to other lasses.

We believe that the same things we have mentioned before regarding the performance of LSTM model could also be said on the CNN model, especially their poor performance on the ‘Entertainment’ category, and great performance on the ‘Sport’ one.

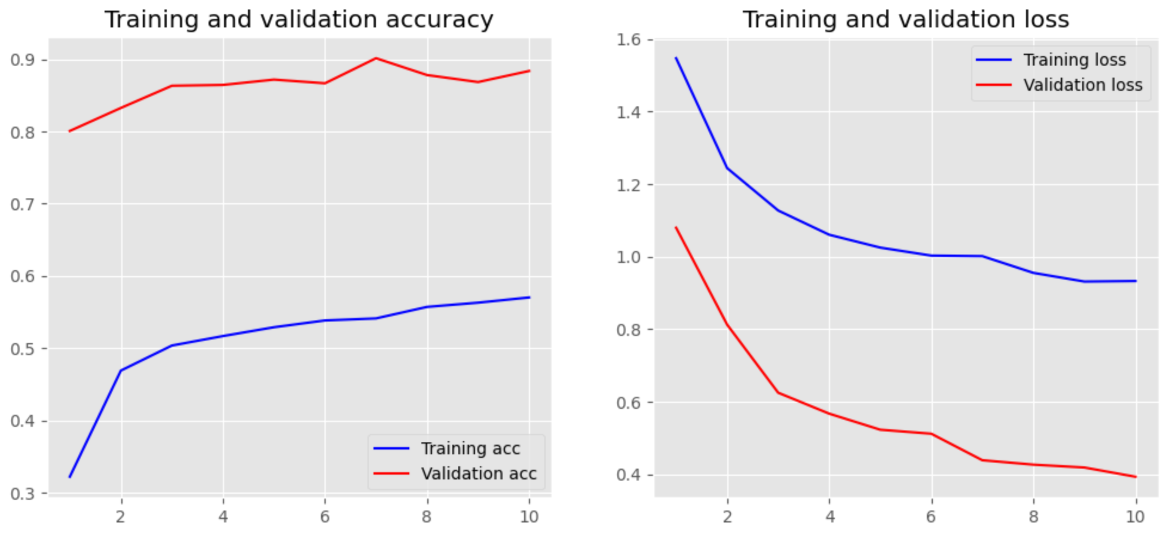


Figure 11 CNN model accuracy and loss over epochs

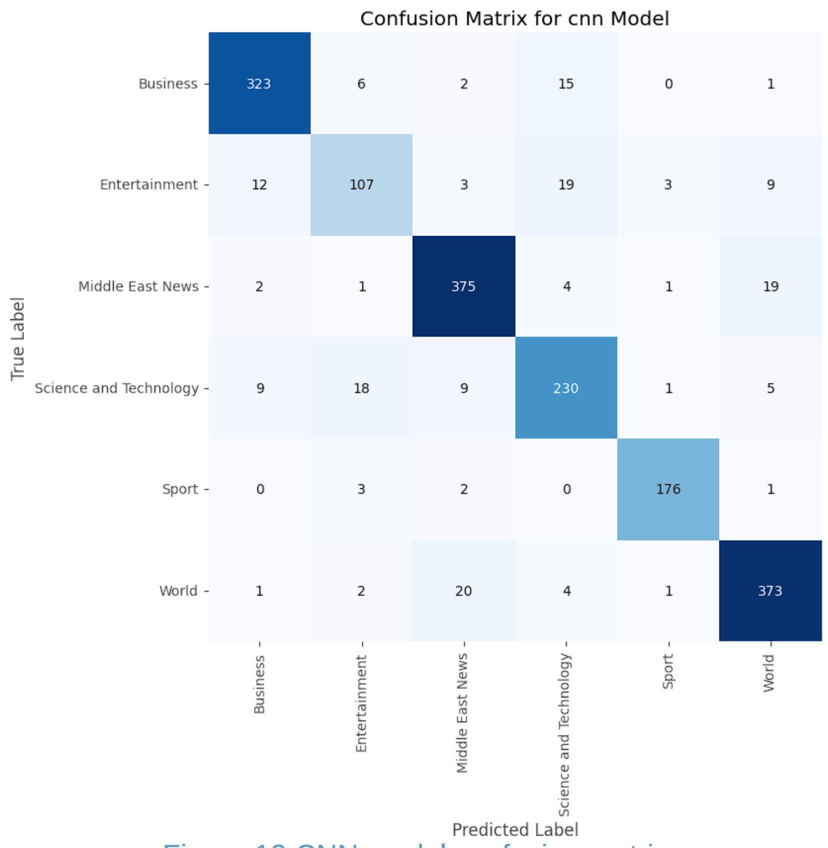


Figure 12 CNN model confusion matrix

c) Testing the Model on a Different Dataset

The CNN model was trained and tested on a new dataset for Arabic sentiment analysis. The objective was to classify Arabic text into positive or negative sentiments.

The model's training and validation performance showed initial improvements but indicated overfitting as training progressed. In the first epoch, the model achieved a training accuracy of 58.54% and a validation accuracy of 63.69%, with corresponding losses of 0.6656 and 0.6258. By the second epoch, both training and validation accuracies improved to 67.49% and 64.22%, respectively, with a notable reduction in loss. However, from the third epoch onwards, validation accuracy began to decline slightly, despite continued increases in training accuracy, reaching 85.81% by the fifth epoch. The divergence between training and validation performance, particularly the increase in validation loss up to 0.8847, clearly indicates overfitting, where the model performs well on training data but poorly on unseen validation data.

The classification report provides further insights into the model's performance across the two sentiment classes. For the negative sentiment class (Class 0), the model achieved a precision of 0.63, a recall of 0.76, and an F1-score of 0.69. In contrast, the positive sentiment class (Class 1) saw lower performance, with a precision of 0.67, a recall of 0.52, and an F1-score of 0.58. This disparity suggests that the model is more effective at identifying negative sentiments than positive ones, possibly due to the complexity of language used in positive expressions.

The overall accuracy of the model was 64%, which, while better than random guessing, indicates room for improvement. The macro and weighted averages for precision, recall, and F1-score were all approximately 0.64, reflecting balanced performance across both classes despite individual disparities. Higher precision in the positive sentiment class suggests that when the model predicts positive sentiment, it is likely to be correct, but the lower recall indicates that it misses many instances of positive sentiment, impacting overall performance.

In conclusion, the CNN model demonstrates a good understanding of negative sentiment but struggles with positive sentiment classification, likely due to distinct language structures.

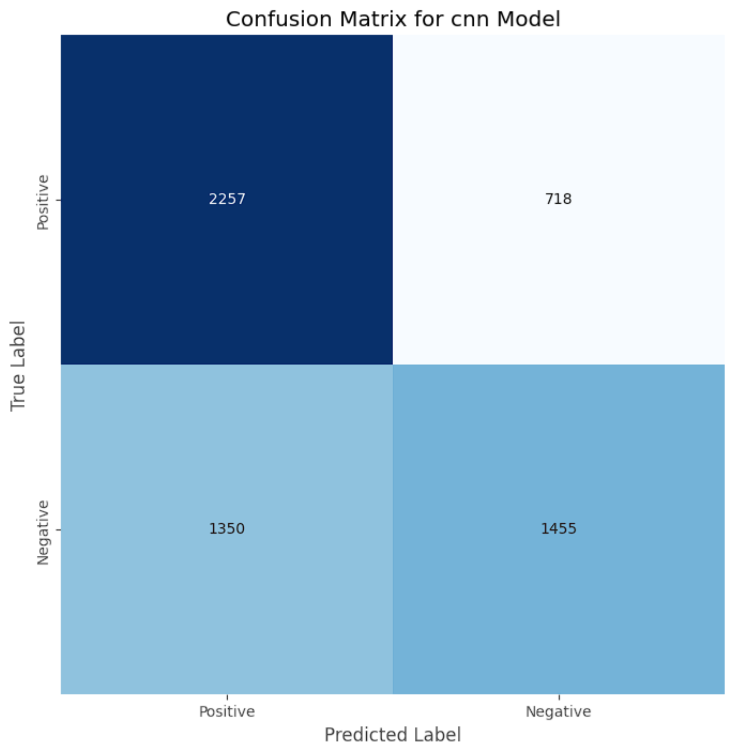


Figure 13 CNN model confusion matrix (2)

4. AraBert

a) Overview

The model was trained using a one-cycle policy with a maximum learning rate of 5*10-5. The choice of the maximum learning rate seems appropriate, contributing to the model's efficient learning without causing instability in the training process.

The training process spanned 10 epochs, during which both training and validation loss and accuracy were monitored. This report summarizes the training progress, evaluates the performance of the model, and provides insights into the results.

b) Training and Validation Metrics

The following figure provides a detailed breakdown of the training and validation loss and accuracy for each epoch.

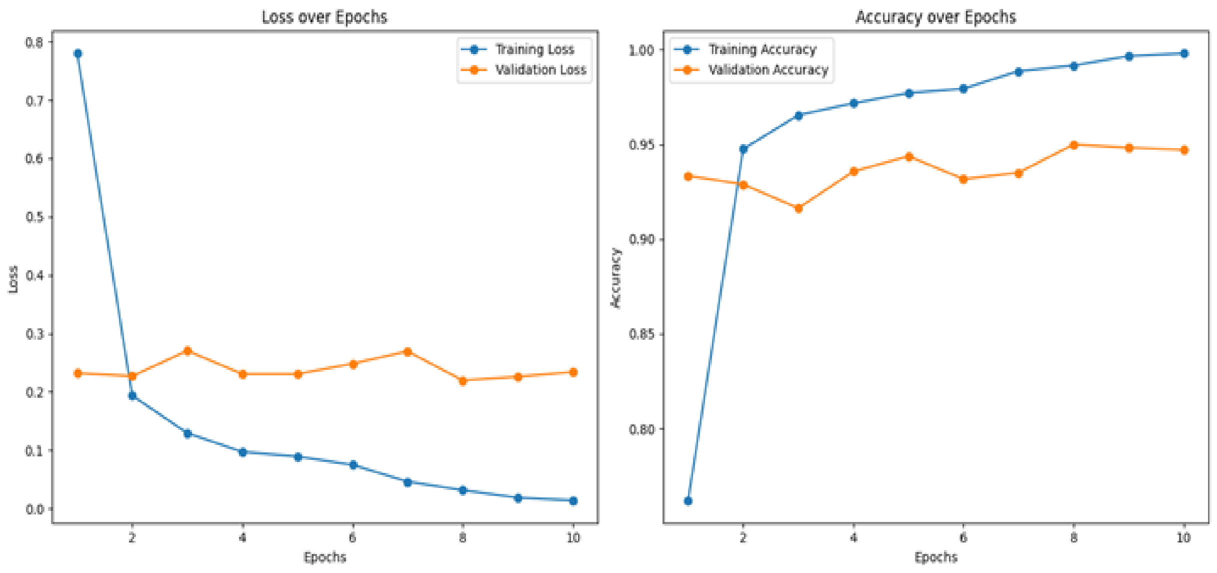


Figure 14 AraBert accuracy and loss over epochs

c) Model Performance Analysis

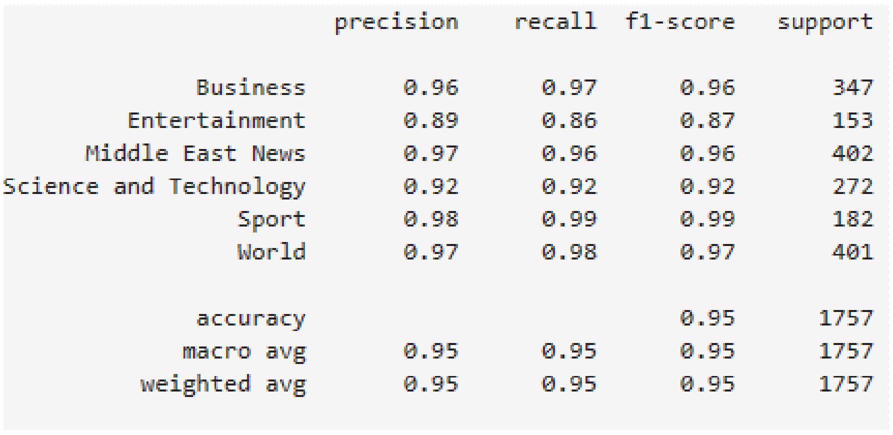
Training Loss: The training loss decreased consistently from 0.7799 in the first epoch to 0.0140 in the tenth epoch, indicating effective learning of the training data.

Training Accuracy: The training accuracy increased from 76.21% to an impressive 99.80% over the 10 epochs, demonstrating high proficiency in classifying the training samples correctly.

Validation Loss: The validation loss showed some fluctuations but generally remained low. It started at 0.2320 in the first epoch, fluctuated slightly, and ended at 0.2340 in the tenth epoch.

Validation Accuracy: The validation accuracy started at 93.34% and varied throughout the epochs, reaching as high as 94.99% and ending at 94.71%.

The detailed evaluation metrics on the validation dataset (Table 1) further reinforce the model's strong overall performance, with an accuracy of 0.95. The precision, recall, and F1-scores for most classes are above 0.9, indicating a high degree of correct classification while minimizing false positives and false negatives.



```
                         precision    recall  f1-score   support

              Business       0.96      0.97      0.96       347
         Entertainment       0.89      0.86      0.87       153
      Middle East News       0.97      0.96      0.96       402
Science and Technology       0.92      0.92      0.92       272
                 Sport       0.98      0.99      0.99       182
                 World       0.97      0.98      0.97       401

              accuracy                           0.95      1757
             macro avg       0.95      0.95      0.95      1757
          weighted avg       0.95      0.95      0.95      1757
```

Figure 15 AraBert classification report

The confusion matrix provides insights into the model's performance by showing the actual and predicted classes for the test dataset. The matrix reveals that the model performs exceptionally well in classifying the "Middle East News" and "Sport" categories, with high true positive rates along the diagonal.

The classification model exhibits some confusion between the 'Business' and 'Entertainment' classes, as well as between the 'Science and Technology' and 'World' classes. This confusion can be attributed to the presence of certain keywords in the news articles. For instance, some entertainment news articles contain words related to the Middle East (e.g., 'الإمارات العربية المتحدة', 'للإمارات'), which can lead to misclassification as 'Middle East' news. Similarly, some world news articles feature words like 'دبلوماسيين بريطانيين' (British diplomats) or 'الرئيس الأمريكي باراك أوباما' (Former US President Barack Obama), which may cause them to be misclassified as 'World News' or 'Business' news. Furthermore, the presence of terms such as 'البرنامج النووي' (nuclear program) or 'ناسا' (NASA) in entertainment news can result in confusion with the 'Science and Technology' class.

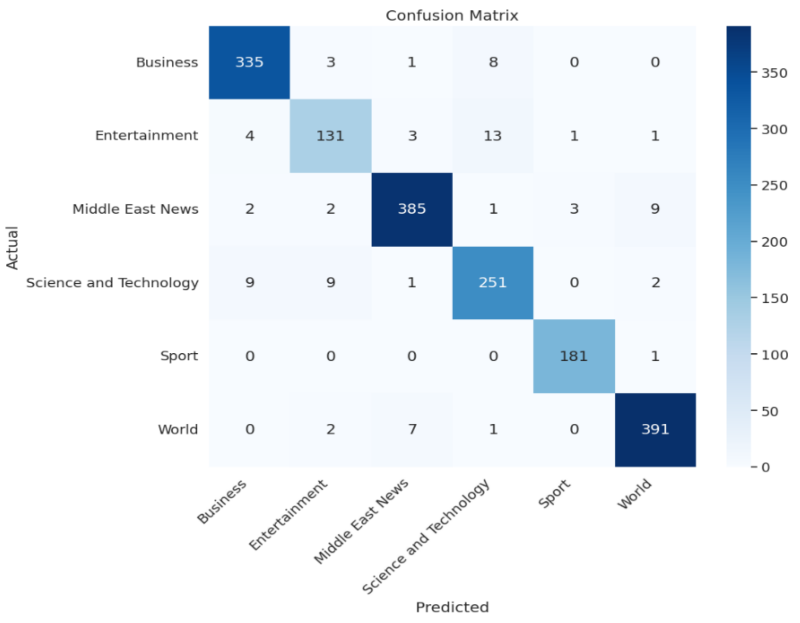


Figure 16 AraBert confusion matrix

d) Testing the Model on a Different Dataset

The model was evaluated on an unseen dataset of Arabic tweets for sentiment analysis. It performed well, achieving an accuracy of 0.78. This indicates that 78% of the tweets in the test set were correctly classified into the positive and negative sentiment categories, demonstrating the model's ability to generalize effectively to new data.

5. Discussion

The LSTM and CNN models achieved an overall accuracy of 80% on the Arabic text classification task, demonstrating their capability in handling this task. However, their performance varied significantly across categories, with the 'Entertainment' category being particularly challenging. Both models struggled to accurately classify entertainment-related articles, achieving low precision and recall scores for this category. This could be attributed to several factors, including lexical overlap with other categories, the broad scope, and ambiguous boundaries of the 'Entertainment' category, as well as being the least presented class in the dataset. The 'Sport' category on the other hand is the least misclassified. This is because its terminology is unique and rarely found in other categories.

In contrast, the AraBERT model, which incorporates the attention mechanism, exhibited superior performance on the same task. The model achieved an impressive overall accuracy of 95% on the validation dataset, with high precision, recall, and F1-scores across most categories. The attention mechanism in AraBERT helps the model identify and focus on the most relevant parts of the input text, which could be particularly beneficial for the 'Entertainment' category, where the context and language used may be more diverse and ambiguous.

When tested on a different dataset for Arabic sentiment analysis, the LSTM and CNN models demonstrated moderate success, achieving overall accuracies of 64%. While their performance was reasonably balanced across positive and negative sentiments, there was still room for improvement. The AraBERT model, on the other hand, achieved an accuracy of 78% on the sentiment analysis task, showcasing its ability to generalize effectively to new data.

The attention mechanism in AraBERT allows the model to dynamically focus on the most relevant parts of the input text, which is particularly advantageous for tasks like sentiment analysis, where the context and language patterns can vary significantly. By attending to the most informative words and phrases, the model can better capture the sentiment expressed in the text, leading to improved classification performance.

However, it is important to note that the attention mechanism alone is not a panacea for all text classification challenges. Even with AraBERT, some confusion was observed between certain categories, such as 'Business' and 'Entertainment', as well as 'Science and Technology' and 'World'. This could be due to the inherent overlap and ambiguity in the language used in these categories, which can be challenging for any model to disentangle.

Overall, the results suggest that the attention mechanism, as implemented in AraBERT, can have a significant positive impact on Arabic text classification tasks, particularly for categories with diverse and ambiguous language patterns. By allowing the model to dynamically focus on the most relevant parts of the input, the attention mechanism enhances the model's ability to capture contextual information and make more accurate classifications. However, other factors such as dataset quality, preprocessing techniques, and model architecture also play crucial roles in achieving optimal performance.

Conclusion

In conclusion, this study investigated the impact of the attention mechanism on Arabic text classification tasks. By comparing the performance of traditional models like LSTMs and CNNs with the more advanced AraBERT model, which incorporates the attention mechanism, several key findings emerged.

Firstly, the attention mechanism, as implemented in AraBERT, demonstrated a significant positive impact on the overall performance of Arabic text classification. The model achieved impressive results, with an overall accuracy of 95% on the validation dataset and an accuracy of 78% on the sentiment analysis task, outperforming the LSTM and CNN models. The attention mechanism's ability to dynamically focus on the most relevant parts of the input text allowed AraBERT to better capture contextual information and make more accurate classifications, particularly in challenging categories like 'Entertainment'.

However, it is important to note that the attention mechanism alone is not a silver bullet for all text classification challenges. Even with AraBERT, some confusion was observed between certain categories, likely due to inherent overlaps and ambiguities in the language used. Additionally, factors such as dataset quality, preprocessing techniques, and model architecture also play crucial roles in achieving optimal performance.

Moving forward, further research could explore the combination of attention mechanisms with other advanced techniques, such as transfer learning, few-shot learning, or multimodal approaches, to address the remaining challenges in Arabic text classification. Additionally, investigating the attention mechanism's impact on other natural language processing tasks, such as named entity recognition, machine translation, or summarization, could provide valuable insights, and potentially lead to further performance improvements.

Overall, this study highlights the potential of attention mechanisms to enhance the performance of Arabic text classification models, paving the way for more accurate and robust natural language processing applications in the Arabic language.

References

[1] Staudemeyer, R. C., & Morris, E. R. (2019). Understanding LSTM: A tutorial into Long Short-Term Memory Recurrent Neural Networks. Schmalkalden University of Applied Sciences. https://arxiv.org/pdf/1909.09586

[2] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS 2017), 6000-6010. https://doi.org/10.48550/arXiv.1706.03762

[3] Chowdhury, S. A., Abdelali, A., Darwish, K., Soon-Gyo, J., Salminen, J., & Jansen, B. J. (2020). Improving Arabic Text Categorization Using Transformer Training Diversification. In Proceedings of the Fifth Arabic Natural Language Processing Workshop (pp. 226–236). Barcelona, Spain (Online): Association for Computational Linguistics.

[4] Soyalp, G., Yildiz, B., Alar, A., & Ozkanli, K. (2021). Improving Text Classification with Transformer1. In 2021 6th International Conference on Computer Science and Engineering (UBMK). https://doi.org/10.1109/UBMK52708.2021.9558906

[5] Alammary, A.S. (2022)1. BERT Models for Arabic Text Classification: A Systematic Review23. Applied Sciences, 12(11), 5720. https://doi.org/10.3390/app1211572045