Intelligent Automation & Soft Computing
DOI: 10.32604/iasc.2023.026235
Article

Tech Science Press

Language-Independent Text Tokenization Using Unsupervised Deep Learning

Hanan A. Hosni Mahmoud[1], Alaaeldin M. Hafez[2] and Eatedal Alabdulkreem[1,*]

[1]Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh, 11671, Saudi Arabia
[2]Department of Information Systems, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia
*Corresponding Author: Eatedal Alabdulkreem. Email: eaalabdulkareem@pnu.edu.sa
Received: 19 December 2021; Accepted: 24 February 2022

Abstract: Languages–independent text tokenization can aid in classification of languages with few sources. There is a global research effort to generate text classification for any language. Human text classification is a slow procedure. Consequently, the text summary generation of different languages, using machine text classification, has been considered in recent years. There is no research on the machine text classification for many languages such as Czech, Rome, Urdu. This research proposes a cross-language text tokenization model using a Transformer technique. The proposed Transformer employs an encoder that has ten layers with self-attention encoding and a feedforward sublayer. This model improves the efficiency of text classification by providing a draft text classification for a number of documents. We also propose a novel Sub-Word tokenization model with frequent vocabulary usage in the documents. The Sub-Word Byte-Pair Tokenization technique (SBPT) utilizes the sharing of the vocabulary of one sentence with other sentences. The Sub-Word tokenization model enhances the performance of other Sub-Word tokenization models such pair encoding model by +10% using precision metric.

Keywords: Text classification; language-independent tokenization; sub word tokenization

1 Introduction

Recently, a great amount of data became available electronically in digital forms. This introduced a great chance to be retrieved for analysis and processing. However, manual analysis or processing of such huge content is costly and time-consuming. Hence, several computerized models were proposed to automatically process this data to deliver classification. Text classification models usually choose key points in texts to generate comprehensible classification target documents.

In general, text classification models attempt to analyze a document by picking the main topics that constitute the documents and identifying the relevant ideas of these topics. Therefore, current models attempt to enhance the classification process performance in identifying the document key points by allowing all the themes exist in it.

Text Classification is an important topic especially by using machine or deep learning. Text classification models assign text documents to different classes utilizing text contents. This study presents a language-independent classification model utilizing deep learning. Deep Learning techniques are employed extensively in many fields such as natural language processing [1–4]. Deep learning models possess multiple processing and learning layers to process large data sets in an efficient way [5]. neural network (CNN) is the best utilized deep learning models in automated Text Classification [6], and text sentiment analysis [7–9]. Nevertheless, the results were not acceptable because of the lack of pre-processing and parameter initializing procedures. In fact, the CNN and other deep learning methods can get held in a local optimum due to the random assigning of weights. These models can result in better performance if the weights were appropriately initialized [10–12].

The key problem that challenges text classification task is the generalization process. There are two main models for computerized text classification, namely abstractive feature extraction and tokenization models. Abstractive feature extraction models implement a deeper text analysis; they integrate semantic analysis and processing. The generated output contains new phrases not found in the original source text. Thus, phrases may be reformulated with a dissimilar meaning that is away from the intent of the author. On the other hand, tokenization models utilize superficial analysis and processing of the text-documents and convey only the syntactic level, where the output contains words and phrase from the original text only [13–15].

Our research presents a model for text tokenization classification using transformer method that substitutes each instance (y_i,Z_i) with |Zi| examples (yi,λj), for each λj element in Zi . Dubbed weight algorithm is an addition phase to the copy transformer model. It utilizes 1/|Zi| weight for each new instance. Previous models substituted each Zi by one of the members, and the model extracted the token with the most frequency.

In this paper, we realized that developing such models is a very significant task. The model will help people in rare languages to acquire cross-language classification in their own language with less time and cost.

This paper is structured as follows: Section 2 surveys the text documents classification techniques. Section 3 introduces problem definition and the new proposed methodology. In Section 4, experimental results are demonstrated. Section 5 depicts conclusion and future work.

2 Related Work

The related work section discusses two main divisions. Text Classification techniques using deep learning are presented. Then, research that utilize genetic algorithms and CNN are presented.

2.1 Deep Learning Text Classification Models

Deep learning methods, that are characterized by their high classification accuracy, are applied in many arears. The authors in [15] presented a model of multi-polar attention technique for text tokenization (BiLSTM). Compared with other model that employ single attention technique, BiLSTM could differentiate between tokens and their sentiment propensities. The authors in [16] presented a neural network architecture for emotional analysis.

In [17], the authors performed text classification utilizing three models, namely: Deep Belief encoder, Deep Auto network, and Recursive Auto network. For vectors, they applied the lexicon processor, and for the pre-processing, they did not utilize stop word removal or stemming. The Arabic Text Bank dataset contained 1,180 phrases. The training subset contained 944 phrases, while the testing subset contained 236 phrases. The models attained 55.5% accuracy for the Deep Belief encoder, 60.5% for Deep Auto

network, and 74.3% for Recursive Auto network. This reason for these low accuracy rates is the lack of the preprocessing phase.

The authors in [18] used Markov clustering Deep Belief encoders. The model started with the preprocessing phase that removed punctuation and conjunctions marks. This phase also utilized the roots of the words. The second phase applied the clustering algorithm and the Markov statistics technique. In the final stage, the training was done by utilizing a CNN network. They performed testing on Arabic Press dataset containing 6,000 phrases. The model achieved precision of 91.2% and F1-measure of 91%.

In [19], the model performed Sentiment Analysis utilizing Neural Tensor. In their research, they developed the Arabic treebank. They performed training and testing using dataset named QALB that has 550,273 phrases. They used a morphological analyzer that excerpts orthographic features with high precision. Their model outperformed the results of other classifiers such as support vector machine (SVM), Recursive encoders and Long Short-Term encoders by 8.6%, 4.2%, and 2.6% respectively.

The research performed by the authors in [20] employed Recurrent Neural architecture for sentence Classification. They compared the results of their model against the Support Vector Machine. The authors utilized a data set of Hotels' booking reviews that has 24,028 labeled instances. They utilized Morphological and syntactic features. The experiments proved that this model outpaced their model with precision reaching 95.3% *vs.* 87.4%. But their model was twice as fast as the SVM.

The authors in [21] improved document classification by utilizing CNN. The introduced model has several phases. The model first employed preprocessing phase to clean the text and eliminate the stop words. In [22], they represented texts with Bag of Words. They utilized term frequencies for dimension reduction to discover the important words. At the last phase, they trained the CNN model. The results of this model realized an accuracy of 92.44%.

In [23], the authors built a Deep Averaging CNN for text classification. They constructed a cross-language text classification. They utilized the English with the voluminous lexical sets to discover text summary from other languages. They used the CNN model with rectified linear activation unit (ReLU), along with 35 learning iterations. They executed the CNN utilizing a machine learning model named PyTorch. The model utilized many Arabic datasets both annotated and unlabeled instances. This model achieved an accuracy of 84.54% for text summary and classification.

In [24], they presented a text summary and classification deep model with a Multi-Kernel word embedding process. The proposed model enhanced the sentence representation through embedding. the model employed ten Arabic domains and compared the results to machine learning architectures. The precision achieved a value of 92.37%, that was better than most results at that time.

The model in [25] proposed a sentiment classification model, based on CNN, for Arabic text. A narrow CNN with only three convolutional layers was proposed. The model utilized row data without lexical or lexicon processing. This model performed better than the existing research and reached a better recall rate.

2.2 Genetic Algorithm for Text Classification

Genetic Algorithm is one of the best optimization algorithms in terms of reliability. In [26], the proposed model was able to distinguish human actions. They optimized the weights of the neural network using genetic algorithm. The seed weights were used to define the genetic chromosome. The fitness was characterized by the prediction algorithm precision. The neural network was accomplished by the gradient descent technique. 128 chromosomes were used in this technique. The first 127 chromosomes encoded the three convolutional matrix masks for blurring, and the last chromosome defined the seed. The masks ranged from –120 to +120 and the seed value ranged from 0 to 7000. The model employed

the crossover with a probability of 0.9. After several epochs, the best solution was selected in testing phase. The model was validated and it achieved an accuracy of 93.48%.

The authors in [27] utilized neural network with a genetic algorithm to optimize the weight for crack parameters in the text documents. The chromosomes were randomly selected and were alternating between 0 and 127, while the Bias was alternating between −64 and 64. The generated solution was utilized in the training phase to generate a classification decision. This output was matched with the ground truth to compute the solution. The most fit one was nominated utilizing a dynamic selection algorithm, with a 3% transmutation ratio. The crossover was designated within the values of zero and the chromosome size. The model achieved an accuracy of 92.8% for crack classification of 100 document.

3 The Proposed Text Classification Model

In this section, we are proposing a Cross-Language-independent classification model. The next subsections will represent the phases of the proposed technique.

3.1 Sub-Word Tokenization

Sub-Word tokenization is an important preprocessing phase that targets to represent a word by partitioning a word into Sub-Words. There are several cases where a single word is a collection of a several significant Sub-Words. Usually, Byte-Pair Encoding [28] is employed for Sub-Word tokenization in many research articles. Byte-Pair Encoding is a model that cuts vocabulary size by partitioning whole sentences into Sub-Word components using byte pair technique [29]. This allows the unidentified token to be well handled. An example is depicted in Fig. 1.

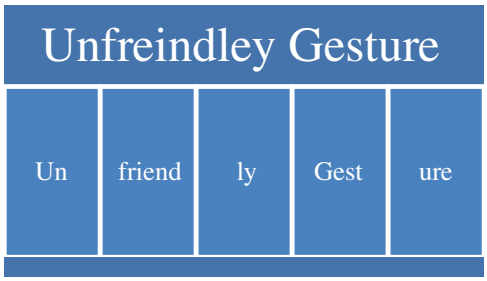

Figure 1: An example of sub word tokenization

One important issue in text classification is how well entities are represented. Text classifications show that proper names and place names exist in most of the sentences. Also, the data about entities is present with same importance. To account for those properties in text classification, data about entities is limited in the Sub-Word tokenization phase. Also, for vocabulary, we propose the training of the Byte-Pair Encoding algorithm by connecting extracted vocabulary. This will enhance the Sub-Word partition consistency and will yield a better performance.

Therefore, this paper presents the Sub-Word Byte-Pair Tokenization technique (SBPT). SBPT utilizes the sharing of the vocabulary of one sentence with other sentences. Sharing vocabulary decreases the insertion of characters when repeating entity data. This technique is suitable for abstract classification where entity data is very important.

The entities are extracted from the source documents. They can be realized from the data structure and can be extracted from source data automatically.

The specific phases of the Sub-Word Byte-Pair Tokenization technique (SBPT) are as follows.

a) Prepare the source *corpus*.

b) Associate every two corpora to perform Sub-Word Tokenization.

c) Fix the Sub-Word token vocabulary size. In our proposed model, we set it to 35,000 Sub-Word based on statistical methods.

d) Divide the words into sequence characters.

e) Merge adjacent pairs of characters with the highest frequencies.

f) Repeat step e several times until the predefined vocabulary size is reached.

3.2 Transformer Model

After we build the training data using the Sub-Word Byte-Pair tokenization technique, training is started employing a sequence-to-sequence method. The proposed Transformer employs an encoder that has ten layers with self-attention encoding and a feedforward sublayer. The decoder is composed of eight layers with masked self-attention decoding, and encoder–decoder intermediate self-attention, and feedforward sublayer.

3.2.1 Encoder Details

When an unknown language sentence is input to the system, it is vectored using word embedding. The encoder learning scheme is started by learning expressions from one language sentence and finding matches of them in another language by utilizing the decoder.

The basic attention process allows the decoder to expect the output word and states of the input phrases in the encoder. Nevertheless, not every input phrase is intentioned with the same reference weight. As an alternative, the portion of the input phrase, that is associated with the word to be predicted, gets more focus. The self-attention process allows the system to learn the association between the current word and other preceding words using key vectors. This procedure is a dot product attention and is described as follows:

a) Create three vectors (query vector, key vector, and value vector).

b) Multiply the query key vectors to calculate the score.

c) Advance with Scale to store key vector size to 64 divided by the value square root to create a stable gradient.

d) Compute the Softmax value, which is equal to the ratio of the number of words in the to-position of the current position.

e) Multiply the results generated from all the steps up to step e by the value vector. Both extreme values of the Softmax value will be ignored (the highest and lowest values).

f) Chain the resultant vectors to form the self-attention output vector.

The multithread attention mechanism is a process that computes the attention outputs for eight weight matrices. The formula for computing the attention values is as follows,

$$A(q,\ k,\ v) = softmax\ \frac{(q \times k^T)}{\sqrt{d_k}}\ v \tag{1}$$

where, A indicates ancient attention. q, k and v indicate query vector, key vector, and value vector respectively.

$$multi - H(A(q,\ k,\ v)) = contact(h_1\ ,\ h_2,\ \ldots,\ h_n) \times Wiegt \tag{2}$$

where, $h_i = A\left(qW_i^q,\ kW_i^k,\ vW_i^v\right)$.

3.2.2 The Decoder Details

The decoder completes the back-transformation using the information generated from the encoder. The attention layer, in the decoder, resembles the previous position of the attentional word in the output phrase sequence. The decoder's output is sequentially produced. Also, the query matrix is extracted from the key and value vector that is part of the encoder output. That is, the attention between the source sentence and the target sentence is processed at the decoder.

We compute the target language attention matrix as follows:

$$AT(q(t),\ k(t),\ v(t)) = softmax\ \frac{\left(q(t) \times k(t)^T\right)}{\sqrt{d_{k(t)}}}\ v(t) \tag{3}$$

Correspondences are built using the decoder. Softmax will produce the optimal output of succeeding token probabilities. In conclusion, the joint training process of the encoder and the decoder will maximize the conditional correspondence–likelihood. The maximization equation is as follows:

$$max_{\propto} = 1/N(\sum_{n=1}^{N} \log P_{\propto}\left(\frac{T_n}{S_n}\right) \tag{4}$$

where α is the model-parameter set and each of (S_n , T_n) is the (source sentence, target sentence) pair in the training data. Fig. 2 shows the overall process.

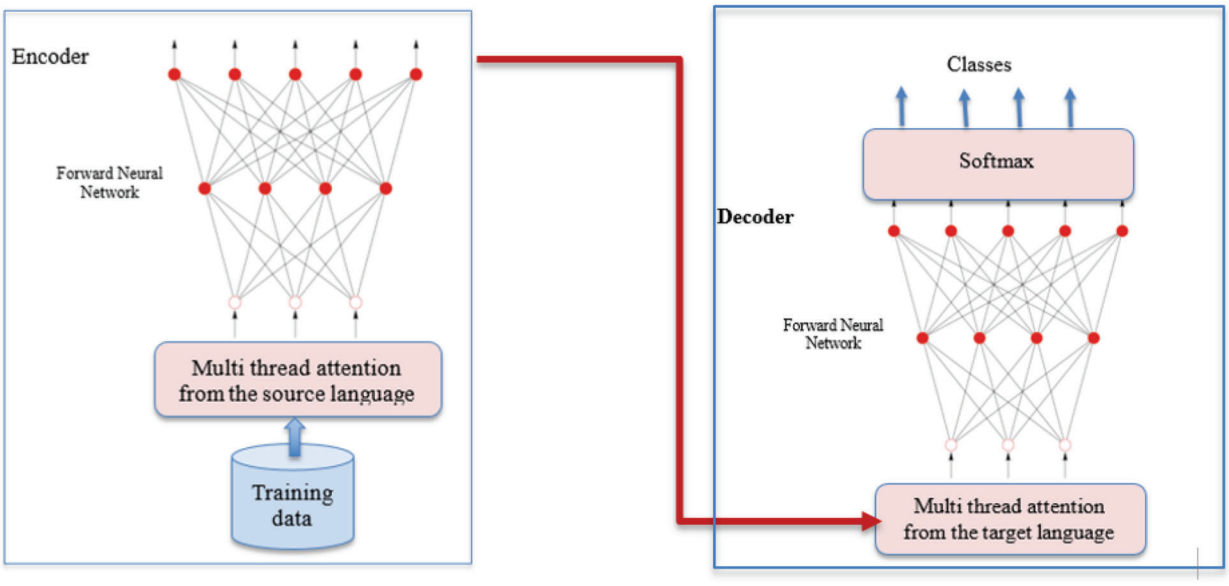

Figure 2: The encoder-decoder model

In Fig. 2, we depict the processing of a source-target parallel *corpus*. Also we show the extraction process from the source dataset and the computerized extraction of the entity data. This creates source

language Entity Dictionary. The proposed model performs the Sub-Word tokenization using the Sub-Word Byte-Pair Tokenization technique (SBPT). The produced results are utilized as the training data of the source language via the encoder. Embedding parameters are used by the encoder. The information of the source-language phrases is learned via multiple head attention model with a source forward neural network. At the end, the model will learn the association between source and target phrases. The source language is converted into the target language through the decoder.

4 Experiments

In this section, we are evaluating the accuracy of our cross language classification model. Datasets are depicted in Subsection 4.1. Evaluation metrics are discussed in Section 4.2. The results are reported and discussed in Section 4.3. Finally, we compare our model with other related models in Section 4.4.

4.1 The Dataset

It should be noted that acquiring training data for rare languages classification is very difficult. Our experiments employed a crawling procedure to collect data. We also performed data filtering with omitting phrases with less than 85 characters.

For multi-token dataset, token cardinality (Card) and density (Dens), for a token L, are used as metrics and they are defined as follows:

$$Card = \frac{1}{m}(\sum_{1}^{m} L_i) \tag{5}$$

$$Dens = \frac{1}{m}(\sum_{1}^{m} \frac{|L_i|}{q}) \quad (q \text{ is the total number of labels}) \tag{6}$$

where, m is defined as the total number of cases in the dataset. The density metric accounts for the total number of tokens in its calculation.

The dataset Mawdoo3 [30] is utilized for testing and is depicted in Tab. 1. Mawdoo3 is considered a benchmark set and is utilized in many multi-token testing methods. The size of the dataset, number of features and count of tokens are depicted in Tab. 1. We divided the dataset into 80% training and 20% testing partitions.

Table 1: The datasets statistics

Domain	Cases	Features	Tokens	Cardinality
Civil Engineering	2767	105	14	4.2
Science	8754	310	24	3.9
Cosmetics	6996	205	20	1.9
Children studies	1950	221	18	2.9
Literature	6710	276	19	4.4
Films	5300	156	8	2.7

4.2 Models and Parameters for Evaluation

We applied the Sub-Word Byte-Pair tokenization technique (SBPT) to Arabic language utilizing long and short-term attention. SBPT is a sequence-to-sequence transformer model. Also, we tested the performance using other Sub-Word tokenization.

For the SBPT attention algorithm, a 2-layer SBPT were employed. In addition, 600 layers were utilized as hidden layers. The dropout proportion was 0.23, with a batch of 256 sentences. The transformer utilized eight attention blocks in both the encoder and decoder, the feedforward network had 1,024 layers, the embedding had 256 layers. The model used twelve heads, and a Noam decay optimization technique with batch size of 2,046. The model collected a vocabulary of 32,000 Sub-Words, with entropy loss function. the model was executed on a GPU GTX 1040 system.

4.3 Evaluation Metrics

To compute the performance of the proposed SBPT methodology, a comparison was carried between the SBPT and another Sub-Word tokenization models.

Two metrics are usually used: the token cardinality, which is calculated as the average number of tokens per case in the test-set, and the token density, which is computed as the number of tokens per test-set divided by the total tokens, averaged over the test-set.

We have to encounter partially correct tests, as we cannot consider these tests as incorrect. The metric, utilized in single-token schemes, was also applied for multi-token metrics. This metric is called exact-match metric (multi-token subset accuracy). The precision of the model was computed as the ratio between the exact-match and the true set of tokens.

Therefore, the concept of partially correctness is defined using the difference between the predicted tokens (P) and the actual true tokens (T), namely the token-based accuracy (TBA). TBA measures the closeness of P to T as the ratio between the number of correct tokens *vs.* all the returned tokens. TBA is computed as follows:

$$TBA = |T \cap P| / |T \; U \; P| \tag{7}$$

TBA is a joint measure of both the precision and the recall metrics. It combines both false positives (members in P that was should not be included) and false negatives (members that are missing in P). TBA is defined as the Jaccard metric [31].

The Hamming Loss metric is similar to the TBA, as it takes both the false positives and the false negatives predictions, and computes the symmetrical difference (logical XOR) between P and T.

Therefore, the precision and recall are calculated in such scenario as follows:

$$Precision = |T \cap P|/|P| \tag{8}$$

$$Recall = |T \cap P|/|T| \tag{9}$$

Sub-Accuracy is the number of the correct cases *vs.* the predicted ones [32]. It means that the predicted tokens are the same match of the true tokens. The accuracy is calculated as follows:

$$Sub - Accuracy = \frac{1}{m}\left(\sum_{1}^{m} \frac{|L_i = A_i|}{M}\right) \tag{10}$$

(L *is the predicted*, A *is the* actual token set, and M is all the cases)

Precision is the average number of the correctly predicted tokens *vs.* the count of tokens, for all instances. The precision is calculated as follows:

$$Precision = \frac{1}{m}\left(\sum_{1}^{m}\frac{|L_i \cap A_i|}{A_i}\right) \tag{11}$$

Recall is the average correctly predicted tokens *vs.* the count of predicted tokens, for all instances. The precision is calculated as follows:

$$Recall = \frac{1}{m}\left(\sum_{1}^{m}\frac{|L_i \cap A_i|}{L_i}\right) \tag{12}$$

F-measure is the harmonic average between the recall and the precision is calculated as follows:

$$F - measure = \frac{2 \times Recall \times Precision}{Recall + precision} \tag{13}$$

4.4 Experiment

The experiments are executed using 12-fold cross validation. Each dataset is iterated for 12 times. Each time, a single subset is used for testing, while 11 other subsets are used for training. Therefore, each fold will be a test set. The final results are averaged on all executions.

4.4.1 Comparison and Result Analysis

We compared our proposed Sub-Word Byte-Pair tokenization technique (SBPT) to the state-of-the-art models: ADAN [23] and ArabADAN [24]. Both models are Arabic language models and are described in Tab. 2.

Table 2: Description of our proposed CNN and other models (ADAN and ArabADAN)

Model	Hidden Layers	Att. Heads	Hidden size	Batch	Epochs	Layers	Activation function
Our proposed model	24	24	1024	Size: 20/32	10/20/50	10 for the encoder/ 8 for the decoder	Softmax
ADAN [23]	24	20	512	Size: 20/32	12	12	ReLU
ArabADAN [24]	20	12	512	Size: 256/64	29	12	Softmax

The comparison of the different models and our proposed model is depicted in Tab. 3. From the tabulated results, the hamming loss metric proves that our proposed model performed better than the ADAN algorithm, on the four of the six dataset domains in Tab. 1. and is comparable to ArabADAN for the literature and the Film domains.

Table 3: Hamming-loss comparison

Model	Civil Engineering	Science	Cosmetics	Children studies	Literature	Films
Our proposed model	0.207 ± 0.021	0.201 ± 0.023	0.199 ± 0.021	0.204 ± 0.021	0.220 ± 0.019	0.221 ± 0.027
ADAN [23]	0.197 ± 0.018	0.205 ± 0.019	0.209 ± 0.019	0.211 ± 0.019	0.219 ± 0.023	0.224 ± 0.021
ArabADAN [24]	0.215 ± 0.021	0.215 ± 0.019	0.204 ± 0.023	0.209 ± 0.021	0.218 ± 0.015	0.223 ± 0.021

Tab. 4 displays that our proposed model performs better in term of the precision measure. It outperforms ADAN systems and is comparable to ArabADAN model. Our model performs higher numbers in the recall metric for all datasets compared to the state-of-the-art in this text classification, as depicted in Tab. 5.

Table 4: Precision comparison

Model	Civil Engineering	Science	Cosmetics	Children studies	Literature	Films
Our proposed model	0.909 ± 0.098	0.908 ± 0.093	0.966 ± 0.098	0.920 ± 0.098	0.986 ± 0.089	0.998 ± 0.099
ADAN [23]	0.868 ± 0.083	0.808 ± 0.086	0.806 ± 0.082	0.888 ± 0.086	0.888 ± 0.083	0.884 ± 0.078
ArabADAN [24]	0.858 ± 0.088	0.868 ± 0.086	0.804 ± 0.083	0.806 ± 0.088	0.888 ± 0.078	0.883 ± 0.068

Table 5: Recall comparison

Model	Civil Engineering	Science	Cosmetics	Children studies	Literature	Films
Our proposed model	0.707 ± 0.076	0.706 ± 0.076	0.699 ± 0.076	0.699 ± 0.076	0.766 ± 0.067	0.776 ± 0.077
ADAN [23]	0.697 ± 0.068	0.707 ± 0.069	0.709 ± 0.067	0.766 ± 0.069	0.767 ± 0.076	0.774 ± 0.076
ArabADAN [24]	0.767 ± 0.076	0.767 ± 0.069	0.704 ± 0.076	0.709 ± 0.076	0.768 ± 0.067	0.776 ± 0.076

Sub-Accuracy is a strict measure because it depicts the exact matching extent between the predicted labels and the true labels. It also presumes some prediction penalty for subsets with almost correct or wrong prediction. In term of Sub-Accuracy, our approach does not perform well as depicted in Tab. 6. Due to the effect of high-cardinality dataset ratio by the ordering of the required target labels. But our model is still comparable to the two other models.

Table 6: Sub-accuracy comparison

Model	Civil Engineering	Science	Cosmetics	Children studies	Literature	Films
Our proposed model	0.948 ± 0.087	0.947 ± 0.083	0.966 ± 0.087	0.986 ± 0.077	0.876 ± 0.078	0.987 ± 0.088
ADAN [23]	0.914 ± 0.078	0.919 ± 0.076	0.926 ± 0.078	0.877 ± 0.076	0.878 ± 0.083	0.884 ± 0.087
ArabADAN [24]	0.927 ± 0.087	0.907 ± 0.076	0.904 ± 0.083	0.920 ± 0.087	0.878 ± 0.077	0.893 ± 0.087

4.4.2 Results of the Evaluation Measures

In this set of experiments, we employed a k-fold validation algorithm to assess the performance. We distributed the text data into k volumes of same size equal folds. CNN was trained in k consecutive iterations, where one fold was tested and k – 1 folds were trained. To have a recognized performance, text data underwent stratification, which is the movement of the words in the text instances in each fold to be a true reprehensive of the data.

Figs. 3–6. depict the results of the experiments in comparison with the models: ADAN and ArabADAN. They resembled our model in some aspects such as accounting for local information for tokenization. The results of the CNN, using *k*-fold validation with *k* ranges from 8 to 14 folds, are depicted. k = 10 was the best performance in all the metrics as depicted.

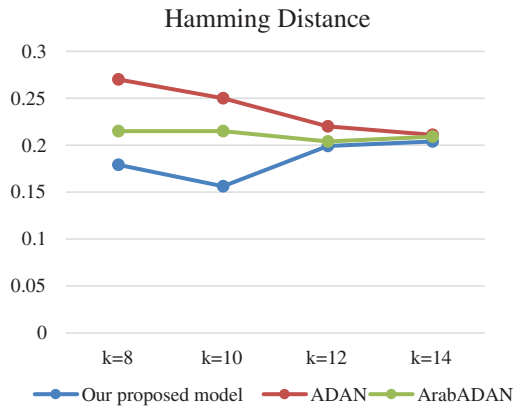

Figure 3: Hamming distance comparison for the CNN using *k*-fold validation with *k* ranges from 8 to 14 folds

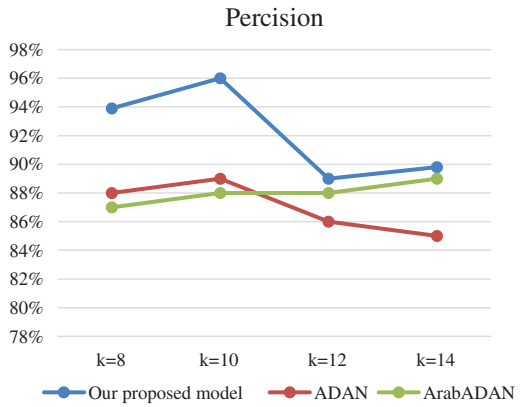

Figure 4: Precision comparison for the CNN using *k*-fold validation with *k* ranges from 8 to 14 folds

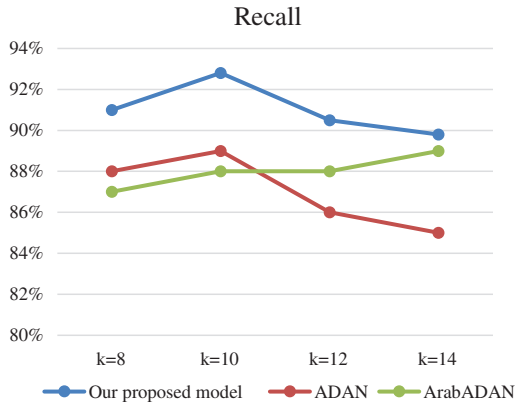

Figure 5: Recall comparison for the CNN using *k*-fold validation with *k* ranges from 8 to 14 folds

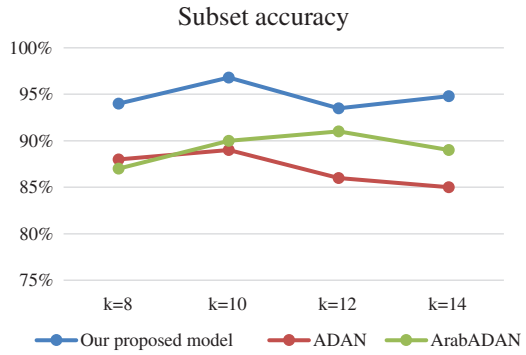

Figure 6: Subset accuracy comparison for the CNN using k-fold validation with k ranges from 8 to 14 folds

4.4.3 Execution Time and Limitations

The execution time needed for the proposed model training phase against other models is depicted in Tab. 7. The results show that our proposed model needs the least execution with an average of 0.43 milliseconds (Ms) for k = 8, while the ArabADAN model needs the highest execution time of 2.95 Ms. It has to be noted that time consumption for cross-validation for higher k, consumes more time, as expected, which can be overcome with the utilization of GPU processors.

Table 7: Comparison of the average execution time (Ms)

Model		Civil Engineering	Science	Cosmetics	Children studies	Literature	Films
Our proposed model	K = 8	0.43	0.51	0.32	0.45	0.29	0.4
ADAN [23]		0.86	0.80	0.95	1.54	0.98	1.27
ArabADAN [24]		2.85	2.96	2.12	3.08	2.88	2.83
Our proposed model	K = 10	0.67	0.61	0.72	0.56	0.49	0.5
ADAN [23]		0.96	1.34	1.45	1.84	1.58	1.97
ArabADAN [24]		5.75	5.96	5.15	5.07	5.77	5.75
Our proposed model	K = 12	1.67	1.61	1.72	1.56	1.70	1.5
ADAN [23]		1.06	1.37	1.75	1.87	1.58	1.07
ArabADAN [24]		5.75	5.06	5.15	5.17	5.77	5.75
Our proposed model	K = 14	2.67	2.62	2.72	2.86	2.70	2.8
ADAN [23]		4.06	4.39	4.98	4.89	4.88	4.09
ArabADAN [24]		8.78	8.06	8.28	8.27	8.77	8.78

5 Conclusion

In this research we proposed a cross-language text tokenization model using a novel Transformer. The proposed model improved the efficiency of text classification by providing a draft text classification for a number of documents. A novel tokenization model that shared frequent vocabulary usage in the documents was also proposed. We presented the Sub-Word Byte-Pair Tokenization technique (SBPT). SBPT utilized the sharing the vocabulary of one sentence with another sentences. Sharing vocabulary decreased the insertion of characters when repeating entity data. A Sub-Word Byte-Pair tokenization

technique was also proposed. The Sub-Word tokenization model enhanced the performance compared to pair encoding model by +10% using precision metric. we compared our proposed Sub-Word Byte-Pair tokenization technique (SBPT) to the state-of-the-art models: ADAN [23] and ArabADAN [24] Arabic language models. The experimental results depicted the superior performance of our model. Also, the execution time for the training phase was compared to other models for the same dataset. The comparison depicted that our proposed model needed the least execution with an average of 0.43 milliseconds (Ms) for k = 8, while the ArabADAN model needed the highest execution time of 2.95 Ms. In conclusion, our proposed model performed better in term of the precision measure. It outperformed ADAN and ArabADAN models in execution time. Our model performed higher rates in the recall metric for all datasets, compared to the state-of-the-art in this text classification.

Funding Statement: This research was funded by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R113), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] R. Qumsiyeh and Y. Ng, "Searching web text documents using a classification approach," *International Journal of Web Information Systems*, vol. 12, no. 1, pp. 83–101, 2019.

[2] A. Pal and D. Saha, "An approach to automatic language-independent text classification using simplified Lesk Algorithm and Wordnet," *International Journal of Control Theory and Computer Modeling (IJCTCM)*, vol. 3, no. 4/5, pp. 15–23, 2020.

[3] M. Al-Smadi, S. Al-Zboon, Y. Jararweh and P. Juola, "Transfer learning for Arabic named entity recognition with deep neural networks," *IEEE Access*, vol. 8, pp. 37736–37745, 2020.

[4] M. Othman, M. Al-Hagery and Y. Hashemi, "Arabic text processing model: Verbs roots and conjugation automation," *IEEE Access*, vol. 8, pp. 103919–103923, 2020.

[5] H. Almuzaini and A. Azmi, "Impact of stemming and word embedding on deep learning-based Arabic text categorization," *IEEE Access*, vol. 8, pp. 27919–27928, 2020.

[6] M. Alhawarat and A. Aseeri, "A superior Arabic text categorization deep model (SATCDM)," *IEEE Access*, vol. 9, pp. 24653–24661, 2020.

[7] S. Marie-Sainte and N. Alalyani, "Firefly algorithm based feature selection for Arabic text classification," *Journal of King Saud University Computer and Information Sciences*, vol. 32, no. 3, pp. 320–328, 2020.

[8] T. Sadad, A. Rehman, A. Munir and T. Saba, "Text tokens detection and multi-classification using advanced deep learning techniques," *Knowledge-Based Systems*, vol. 84, no. 6, pp. 1296–1908, 2021.

[9] M. El-Affendi, K. Alrajhi and A. Hussain, "A novel deep learning-based multilevel parallel attention neural (MPAN) model for multidomain Arabic sentiment analysis," *IEEE Access*, vol. 9, pp. 7508–7518, 2021.

[10] Y. LeCun, Y. Bengio and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2020.

[11] E. Ijjina and K. Chalavadi, "Human action recognition using genetic algorithms and convolutional neural networks," *Pattern Recognition*, vol. 59, no. 5, pp. 199–212, 2020.

[12] W. El-Kassas, C. Salama, A. Rafea and H. Mohamed, "Automatic text classification: A comprehensive survey," *Expert Systems Applications*, vol. 5, no. 3, pp. 112–121, 2021.

[13] M. Gomez, M. Rodríguez and C. Pérez, "Comparison of automatic methods for reducing the weighted-sum fit-front to a single solution applied to multi-text documents text classification," *Knowledge-Based Systems*, vol. 10, no. 1, pp. 173–196, 2019.

[14] A. Widyassari, S. Rustad, G. Hidik and A. Syukur, "Review of automatic language-independent text classification techniques & methods," *Journal King Saud University of Computer Information Sciences*, vol. 19, no. 1, pp. 133–142, 2020.

[15] M. Lins, G. Silva, F. Freitas and G. Cavalcanti, "Assessing phrase scoring techniques for tokenization text classification," *Expert Systems Applications*, vol. 40, no. 14, pp. 755–764, 2019.

[16] Y. Meena and D. Gopalani, "Efficient voting-based tokenization automatic text classification using prominent feature set," *IETE Journal of Reasoning*, vol. 62, no. 5, pp. 581–590, 2020.

[17] A. Al-Saleh and M. Menai, "Automatic Arabic text classification: A survey," *Artificial Intelligence Review*, vol. 45, no. 2, pp. 203–234, 2020.

[18] M. Fattah and F. Ren, "GA, MR, FFNN, PNN and GMM based models for automatic text summarization," *Computer Speech Languages*, vol. 23, no. 1, pp. 126–144, 2021.

[19] A. Al-Saleh and M. Menai, "Solving multi-text documents classification as an orienteering problem," *Algorithms*, vol. 11, no. 7, pp. 96–107, 2019.

[20] V. Patil, M. Krishnamoorthy, P. Oke and M. Kiruthika, "A statistical approach for text documents classification," *Knowledge-Based Systems*, vol. 9, pp. 173–196, 2020.

[21] H. Morita, T. Sakai and M. Okumura, "Query Pnowball: A co-occurrence-based approach to multi-text documents classification for question answering," *Information Media Technology*, vol. 7, no. 3, pp. 1124–1129, 2021.

[22] D. Patel, S. Shah and H. Chhinkaniwala, "Fuzzy logic based multi text documents classification with improved phrase scoring and redundancy removal technique," *Expert Systems Applications*, vol. 194, pp. 187–197, 2019.

[23] M. Safaya, A. Abdullatif and D. Yuret, "Yuret ADAN-CNN for offensive speech identification in social media," *Expert Systems*, vol. 4, no. 3, pp. 167–178, 2020.

[24] W. Antoun, F. Baly and H. Hajj, "ArabADAN: Transformer-based model for Arabic language understanding," *Information Technology*, vol. 4, no. 3, pp. 124–132, 2020.

[25] M. Fattah and F. Ren, "GA, MR, FFNN, PNN and GMM based models for automatic text summarization," *Computer Speech Languages*, vol. 23, no. 1, pp. 126–144, 2021.

[26] A. Qaroush, I. Farha, W. Ghanem and E. Maali, "An efficient single text documents Arabic language-independent text classification using a combination of statistical and semantic features," *Knowledge-Based Systems*, vol. 7, no. 2, pp. 173–186, 2019.

[27] V. Gupta and G. Lehal, "A survey of language-independent text classification tokenization techniques," *Journal Emerging Technology Web*, vol. 2, no. 3, pp. 258–268, 2020.

[28] I. Keskes, "Discourse analysis of Arabic text documents and application to automatic classification," *Algorithms*, vol. 12, no. 3, pp. 187–198, 2019.

[29] X. Cai, W. Li and R. Zhang, "Enhancing diversity and coverage of text documents summaries through subspace clustering and clustering-based optimization," *Information Science*, vol. 279, no. 1, pp. 764–775, 2021.

[30] W. Luo, F. Zhuang, Q. He and Z. Shi, "Exploiting relevance coverage and novelty for query-focused multi-text documents classification," *Knowledge-Based Systems*, vol. 46, no. 6, pp. 33–42, 2019.

[31] A. Zhou, B. Qu, H. Li and Q. Zhang, "Multiobjective evolutionary algorithms: A survey of the state of the art," *Swarm Evolutionary. Computation*, vol. 1, no. 1, pp. 32–49, 2020.

[32] M. Sanchez, M. Rodríguez and C. Pérez, "Tokenization multi-text documents text classification using a multi-objective artificial bee colony optimization approach," *Knowledge-Based Systems*, vol. 159, no. 1, pp. 1–8, 2020.