Improving Scientific Literature Classification: A Parameter-Efficient Transformer-Based Approach

Original Scientific Paper

Mohammad Munzir Ahanger
University of Kashmir
Faculty of Applied Sciences, Department of Computer Sciences, Srinagar, India
munzir.scholar@kashmiruniversity.net

M. Arif Wani
University of Kashmir
Faculty of Applied Sciences, Department of Computer Sciences, Srinagar, India
awani@uok.edu.in

Abstract *– Transformer-based models have been utilized in natural language processing (NLP) for a wide variety of tasks like summarization, translation, and conversational agents. These models can capture long-term dependencies within the input, so they have significantly more representational capabilities than Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN). Nevertheless, these models require significant computational resources in terms of high memory usage, and extensive training time. In this paper, we propose a novel document categorization model, with improved parameter efficiency that encodes text using a single, lightweight, multiheaded attention encoder block. The model also uses a hybrid word and position embedding to represent input tokens. The proposed model is evaluated for the Scientific Literature Classification task (SLC) and is compared with state-of-the-art models that have previously been applied to the task. Ten datasets of varying sizes and class distributions have been employed in the experiments. The proposed model shows significant performance improvements, with a high level of efficiency in terms of parameter and computation resource requirements as compared to other transformer-based models, and outperforms previously used methods.*

Keywords*: deep learning, document categorization, text classification, scientific literature classification*

1. INTRODUCTION

An ever-increasing amount of textual information in the form of research articles, books, conference proceedings, patents, and theses is produced and published every year. PubMed, a biomedical and life science literature search engine, lists more than 30,000 journals and more than 35 million citations [1]. As far back as 2009, the number of published journal articles surpassed 50 million [2] The value and utility of scientific literature depends upon the automatic organization and categorization into different subjects, domains, and themes. Text classification has proven to be an indispensable tool for the organization, curation, and retrieval of such textual data repositories.

Deep learning algorithms such as Convolution neural network (CNN) [3-6] and Recurrent neural network (RNN) [7, 8] based models have been used for text classification. The applications of these supervised deep learning models are numerous and varied, ranging from biometrics such as face recognition [9] and fingerprint recognition [10] to medical science [11, 12] and time series forecasting [13].

Recently more complex, transformer-based models have been applied [14-19]. These models outperform the other simpler models for tasks involving text classification. However, the performance improvements are at the cost of increased model size and complexity. Such complex models are required for good results in tasks such as translation and summarization. However, these models are inefficient when used in comparatively simpler tasks such as text classification. This inefficiency and model complexity result in issues such as higher computational demands, complex fine-tuning, model space complexity, interpretability issues, latency, and data requirements. Besides they may not be suitable for small datasets, consume substantial resources, and lack transparency.

In this study, we address the efficiency and model complexity issues associated with the transformer-based models in dealing with the text classification problem. The architecture choices for the transformer-based model are reconsidered to achieve parameter efficiency for text classification. A new efficient model is proposed and evaluated on the task of Scientific Literature Classification (SLC) and is compared with previously used models.

2. RELATED WORK

Deep learning has been applied in a variety of different applications of supervised learning [20]. Different approaches have been employed in text classification tasks [3-8]. Convolutional Neural network (CNN) based models apply filters of varying sizes on the input text to extract useful features. Such models vary in input representation, the number of CNN layers, and the number and size of filters in each layer. The model Text-CNN [3] trains a CNN over text represented as a matrix comprising pre-trained word vectors. The model uses both finetuned as well as pre-trained word embeddings. The authors in [21] demonstrate the advantages of using pre-trained word embeddings in text classification tasks. Multi-group norm constraint CNN [22] uses multiple such word embeddings to improve the performance of CNN-based algorithms. The authors apply the model to text classification tasks such as sentiment analysis, irony detection, and question type detection. Self-attention mechanisms can be used to achieve improvements for such algorithms [23]. Inspired by the success of large models used in image classification tasks, a deeper CNN network that uses a character-based input text representation is employed by some authors [24, 25]. The model VDCNN [25] uses up to 30 layers to extract features from the input text. For text classification, the use of such deep models is costly and not necessary [26]. Set-CNN employs semantic extension and multi-channel convolution to improve classification performance [27].

Recurrent neural network-based models have been used in text classification tasks [7, 28]. Enhanced recurrent models such as BLSTM-2DCNN try to combine the best features of CNN and RNNs to obtain a better input representation [7, 8].

Hybrid models that combine CNN and RNN in different ways have been proposed, as those used in [29, 30, 31]. These models use a CNN layer to extract useful features followed by an RNN layer to obtain an enriched contextual representation. The authors in [29] use BiGRU (Bidirectional Gated Recurrent unit), a gated RNN while [31] uses LSTM (Long Short-Term Memory). An alternate approach is to use RNN followed by CNN. This approach has been employed by [7, 32]. The authors in [7] use two-dimensional max-pooling to the output of LSTM. The model BiLSTM-C uses two layers of LSTM followed by a layer of CNN [32]. In such models, the LSTM layer outputs token representations based on the previously seen tokens. This representation is input into the convolutional layer(s) for feature extraction. Similar models are used in [33, 34, 35]. The authors in [36] propose enhancing RNN models by modifying the activation functions used.

To address the problems associated with CNN and RNN-based methods, specifically, the lack of interpretability and intuition, Attention-based models were proposed in [37]. Attention-based models have been employed for machine translation [37], visualizing important parts of a text [38]. HAN (Hierarchical Attention Network) proposed in [38] encodes a sentence by focusing on the constituent words differentially. Another encoder is used to encode the whole document by attending to the constituent sentence representations [39]. Transformer-based models similar to the original transformer model [15] such as BERT (Bidirectional Encoder Representation for Text) [14] use a stack of attention-based encoders. Similar variations have been proposed. These include the models proposed in [16-19]. These models have been applied for tasks such as translation [40], summarization [41] as well as text categorization [42].

This paper proposes a new multi-headed attention-based model for text classification that reconsiders the architectural choices in transformer-based models when used for tasks involving text classification. The model demonstrates that significantly more parameter-efficient models can be designed and used for such tasks. This reduces memory requirements as well as training and inference times. Besides the carbon footprint associated with large pre-trained language models is also minimized [43]. The paper applies the model to the task of classifying scientific literature. The model outperforms previously applied methods.

3. PROPOSED MODEL

3.1. OVERVIEW

Transformer-based models such as BERT [14] have the drawback of having large parameter spaces (BERT-base has 108 M parameters), making them slow to train and run. For relatively simple NLP tasks such as text classification, this makes their use inefficient. The proposed model addresses this by reconsidering the architectural choices in such large and complex transformer-based models.

We employ a single efficient and lightweight encoder block with 12 attention heads. The encoder block consists of a Self-Attention subunit and a fully connected neural network (FCN) subunit. Our experiments show that this simplification of the architecture of the encoder block suffices to capture textual semantics without a significant loss in performance.

3.2. DETAILED DESCRIPTION OF THE PROPOSED MODEL

This subsection presents a detailed description of the proposed approach and explains the different steps involved using mathematical equations.

3.2.1. Embeddings Block

The proposed model (Fig. 1) uses two types of embeddings, word embedding and positional embedding. We use token embeddings of size 100 initialized using GloVe [44]. The word embedding outputs a dense word representation as a linear projection of one-hot encoded words as shown in (Eq. 1).

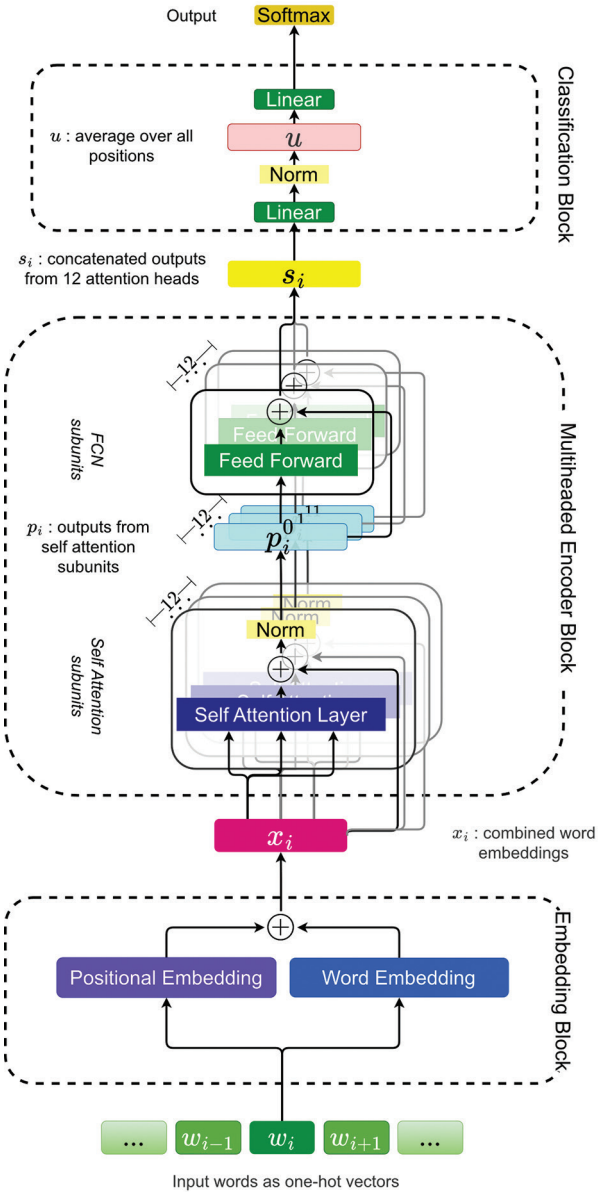

Fig.1. Proposed model architecture

$$x_i^{word} = W^{word} \times w_i^T \tag{1}$$

Here x_i^{word} is the dense word vector for the i^{th} word in the input, w_i. W^{word} is the word embedding matrix initialized using GloVe [44].

The positional embedding learns to encode the order of tokens in the input text as a linear transformation of the position within the input text as shown in (Eq. 2).

$$x_i^{pos} = W^{pos} \times w_i^T \tag{2}$$

Here x_i^{pos} is the positional embedding vector corresponding to the k^{th} word in the input, w_i. W^{pos} is the positional embedding matrix. The positional embedding matrix is initialized as used in [15].

The two embeddings are combined to obtain a better, position-aware word embedding. The combined word embeddings are obtained by calculating the sum of the corresponding word and positional embeddings as shown in (Eq. 3).

$$x_i = x_i^{word} + x_i^{pos} \tag{3}$$

Here, x_i is the combined word vector.

3.2.2. Light Weight encoder block

The proposed model approach uses a single efficient and lightweight encoder block to obtain token representations using the mechanism of self-attention. We project the word embeddings, to three different dense vector spaces using three projections – Query, Key, and Value which are linear transformations of the combined word vector (Eq. 3). This is shown in (Eq. 4), (Eq. 5), and (Eq. 6).

$$q_i = W_q \times x_i^T \tag{4}$$

$$k_i = W_k \times x_i^T \tag{5}$$

$$v_i = W_v \times x_i^T \tag{6}$$

Here, v_i is the key vector corresponding to the i^{th} word in the input text, x_i is the combined word embedding vector and W_v is the value projection matrix.

The encoder outputs an attention-based representation with the capability to attend to a specific piece of information from a potentially infinitely large context. The scaled dot product (a_{ij}) of a query vector (q_i) (representing the word being encoded) with a key vector (k_j) (representing another word) is treated as the attention score assigned to the keyword when interpreting the query word as shown in (Eq. 7).

$$a_{ij} = q_i^T k_j \tag{7}$$

$$score(q_i, k_j) = \frac{a_{ij}}{\sqrt{n}} \tag{8}$$

here, q_i is the query word (one that is being encoded) and k_j is a keyword (one of the words in the input) whose attention score against q_i is to be calculated. n is the length of the query and key vectors.

Next, *softmax* is applied over scores for all key vectors for a query vector. This scaled set of scores (α_i) is used to scale each value vector v_j. The scaled value vectors are added together to obtain z_i, the representation of the i^{th} word in the input text. This operation is shown in (Eq. 9).

$$z_i = \sum_j (\alpha_{i,j} \otimes v_j) \tag{9}$$

We use residual connections [45] across the attention mechanism in the proposed model, allowing gradients to travel through them directly. The unattended representations (x_i) are added to the representation obtained as a result of the attention mechanism (z_i) to obtain the input for the next step, layer normalization ($n_i \in R^{\text{embedding\_dim}}$). This is shown in (Eq. 10).

$$n_i = z_i + x_i \tag{10}$$

The proposed model uses layer normalization as a means of regularization to reduce training time [46]. This can be represented as shown in (Eq. 11)

$$p_i = \mathrm{LN}_{\gamma_1,\beta_1}(n_i) \quad (11)$$

The normalized word representations are input to two successive fully connected neural network layers to generate the attention head output. The output vectors obtained from multiple attention heads are concatenated together and are subjected to a linear transformation layer to output a combined multiheaded word representation. A residual connection is used that bypasses the output i.e., p_i is added to r_i to obtain a combined output s_i as shown in (Eq. 12).

$$s_i = p_i + r_i \quad (12)$$

3.2.3. Classification Block

The combined output from all the heads is passed to a single linear transformation layer, followed by a normalization layer that generates the normalized vectors t_i, as shown in (Eq. 13).

$$t_i = \mathrm{LN}_{\gamma_2,\beta_2}(W_0 \times s_i^T) \quad (13)$$

Finally, the average overall positions represent the entire text as shown in (Eq. 14).

$$\mathrm{u} = \frac{\sum_i^n t_i}{n} \quad (14)$$

The feature vector, u, is finally input into a *softmax* activated fully connected layer of neurons whose outputs are treated as class probabilities as shown in (Eq. 15) and (Eq. 16).

$$\mathrm{v} = W_{out} \times u^T \quad (15)$$

$$\mathrm{out_i} = \mathrm{softmax}(v_i) = \frac{e^{v_i}}{\sum_{j=1}^{nClasses} e^{v_j}} \quad (16)$$

4. EXPERIMENTS AND RESULTS

The proposed model was applied to ten different datasets. For comparison, seven other deep-learning-based text classification models were also tested on the task. The following subsections describe the datasets and the model evaluation method used.

4.1. DATASETS

In this study, we used ten datasets comprising abstracts of scientific papers. Three Web of Science (WOS) datasets created and used by [6] were employed. In addition to these, seven new SLC datasets were created and used. We used Python libraries like Urllib, Lxml, and Beautiful Soup to obtain data. The datasets contain labeled abstracts with their associated categories and subcategories.

The three *WOS* datasets [6] include abstracts from 46958 publications and are categorized into 134 categories and 7 domains. The three *COR* datasets are derived from the *ArXiv* metadata repository released by Cornell University. The *ArXiv* dataset was gathered from the *ArXiv* [47]. The collection is divided into seven domains and contains 146 areas. The *Nature* dataset was gathered from the scientific paper repository - *Nature* [48]. It is divided into eight domains and contains 102 areas. The *Springer* dataset comprises metadata about 116230 published papers available from *Springer* [49]. The collection is divided into 24 domains and contains 117 areas. The *Wiley* dataset was gathered from the *Wiley Online Library* [50]. There are 494 areas in the collection, which are divided into 74 categories.

Table 1 describes the features of the datasets used in the study. These datasets vary in the number of domains, training and testing samples, mean number of words and characters per sample, and vocabulary size. These characteristics can impact the performance of text classification models trained on these datasets. The larger and more diverse the dataset, the better the model's performance is likely to be. Table 2 lists the classes within each dataset.

Table 1. Summary of datasets used

Dataset	Dataset characteristics						
	Number of Domains	Number of Abstracts	Training samples	Testing samples	Words/sample (Mean)	Chars/sample (Mean)	Vocabulary Size
WOS5376	3	5736	4588	1148	209.03	1386.13	42306
WOS11967	7	11967	8017	3950	201.43	1340.19	57875
WOS46985	7	46985	31479	15506	205.27	1375.76	125968
ArXiv	7	40060	32048	8012	148.21	978.67	112452
Nature	8	49782	24891	24891	175.20	1206.56	84228
Springer	24	116230	92984	23246	167.92	1128.41	22254
Wiley	74	179953	143962	35991	170.26	1150.62	113534
COR-8883	3	8883	7106	1777	135.705	889.15	28836
COR-61033	4	61033	48826	12207	127.545	782.714	52053
COR-233962	6	233962	187169	46793	148.247	971.305	172954

Table 2. Classes in datasets

	Number	Labels
WOS5376	3	Elec. & Comm. Eng., Psychology, Biochemistry
WOS11967, WOS46985	7	Comp. Sci., Elec. & Comm. Eng., Psychology, Mech. & Aero Eng., Civil, Medical, Biochemistry
ArXiv	7	Comp Sci, Economics, EE&SS, Math, Physics, Q Biology, Q Finance
Nature	8	Bio Sci, Bus & Comm, Earth & Env Sci, Health Sci, Humanities, Phys Sci, Sci Community & Society, Soc Sci
Springer	24	Biomedicine, Bus. & Mgmt., Chemistry, Comp Sci., Earth Sci., Economics, Education, Engineering, Environment, Geography, History, Law, Life Sci., Literature, Mat. Sci., Math, Med. & Pub. Health, Pharmacy, Philosophy, Physics, Poli. Sci. & Intl. Relations, Psychology, Social Sci., Statistics
Wiley	74	Accounting, Agriculture, Allergy & Clin. Immunology, Analytical Chem., Anatomy & Physiology, ..., Religion & Theology, Social Policy & Welfare, Space & Planetary Sci., Statistics, Veterinary Medicine
COR-8883	3	Economics, Chaotic Dynamics, Algebraic Geometry
COR-61033	4	HEP - Experiment, HEP - Lattice, Nuclear Exp, Quantitative Finance
COR-233962	6	Gen. Relativity & Quantum Cosmology, Stats, Elec. Eng. & Sys. Science, Nuclear Theory, Math Physics, Quant. Biology

4.2. EXPERIMENTAL SETUP

This subsection describes the experimental setup used in this study. The experiments were performed using Python v3.9.1 [51] using Keras v2.3.1 [52] API for Tensorflow 2.0 [53]. Python libraries such as the lxml v4.6.4 [54], and beautiful soup v4.10.0 [55] were utilized to acquire some of the datasets from websites such as ArXiv, Wiley, Springer, and Nature

The datasets were cleaned by filtering out special characters. Tokenization was then done using a dictionary size of 20000. A constant length of 250 words was ensured using padding and truncation as needed. The datasets underwent a random shuffle to eliminate any ordering bias, and subsequently, a standard (80-20) rule was used to split them into training and testing subsets. In the proposed model, we utilized the Adam optimizer, with a learning rate of 0.01 and a batch size of 16. All the models were trained for 20 epochs. (Fig. 2) shows the training graphs for the proposed model on two different datasets.

4.3. RESULTS

We compare the performance of the proposed model with previously applied deep learning-based text classification methods based on classification accuracy percentages. This subsection presents the results of the experiments conducted to assess the efficacy of the proposed model and to compare its performance against other deep learning-based models that have previously been applied to the task of SLC.

Table 3 compares the accuracy of the proposed model with state of art models on seven different datasets. The performance can be depicted visually as shown in (Fig. 3). Model training times are listed in Table 4.

Table 3. Comparison with previously used models on classification accuracy (%) metrics

	Models							
Datasets	**TextCNN [3]**	**MGNCNN [22]**	**CharCNN [24]**	**RCNN [28]**	**VDCNN [25]**	**HDLTEX- CNN [6]**	**HDLTEX – RNN [6]**	**Proposed Model**
Parameters (M)	25.6	33.7	11.5	16.9	14.35	34.9	11.32	11.9
WOS-5736	49.46	97.41	88.48	97.07	82.9	96.47	97.82	99.13
WOS-11967	16.55	92.94	25.45	92.94	67.64	93.52	93.98	96.17
WOS-40896	31.13	89.33	76.59	88.2	75.76	88.67	90.45	95.71
ArXiv	26.26	85.35	80.53	83.32	66.11	85.84	84.26	87.11
Springer	27.7	100	61.6	85.0	63.0	99.98	100	100
Nature	33.79	57.23	51.23	52.21	53.44	60.12	61.83	61.56
Wiley	8.15	52.04	7.82	62.10	49.50	50.81	43.39	72.24
COR-8883	88.74	91.64	97.97	91.87	67.24	93.35	92.90	94.25
COR-61033	90.06	90.66	93.38	92.38	28.16	94.67	92.46	93.41
COR-233962	87.23	91.93	90.725	92.31	56.44	90.2	91.55	93.37

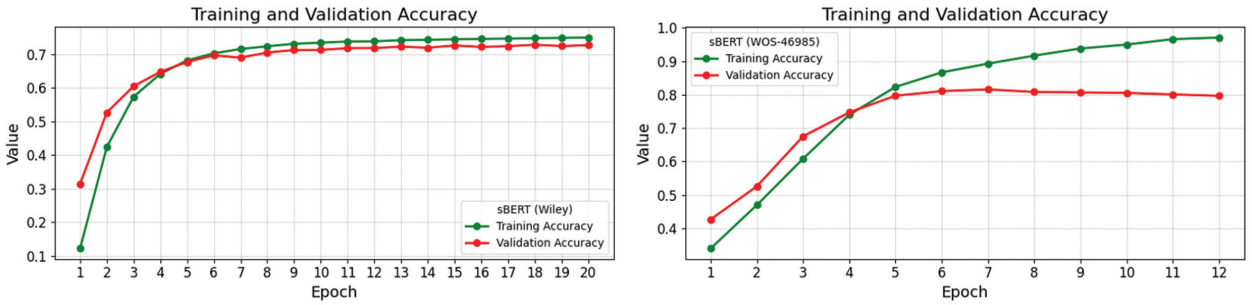

Fig. 2. Training graphs of the proposed model on two datasets WOS- 46985 and Wiley

Table 4. Comparison with training times with previously used models (in seconds)

	TextCNN	MGNCNN	CharCNN	RCNN	VDCNN	HDLTEX-CNN	HDLTEX – RNN	sBERT
WOS5736	73.15214	3631.116	158.1202	501.49	269.139	140.4653	166.1034	157.2688
WOS11967	106.2313	5498.224	221.2821	716.46	405.123	206.5897	708.5794	286.5306
WOS46985	333.4941	10805.61	771.9043	2644.6	1267.18	482.8745	2948.638	604.8373
arXiv	341.4121	15226.54	450.2403	771.23	687.927	330.3308	831.7495	500.1951
Nature	243.1619	10844.71	386.2782	532.34	407.972	271.8696	639.7405	361.6975
Springer	776.393	34626.14	880.3642	2096.1	788.691	549.7363	2251.08	721.5509
Wiley	1222.111	54504.59	1077.362	3002.8	8974.24	1845.265	3309.169	1333.135
COR-8883	94.99683	4236.737	186.051	457.67	555.342	127.8164	696.8934	179.0956
COR-61033	464.6511	20722.85	670.7038	1324.9	1891.57	434.1656	906.9122	527.6036
COR-233962	1509.796	67334.97	1773.456	4780.1	8909.46	1510.261	2864.248	1493.209

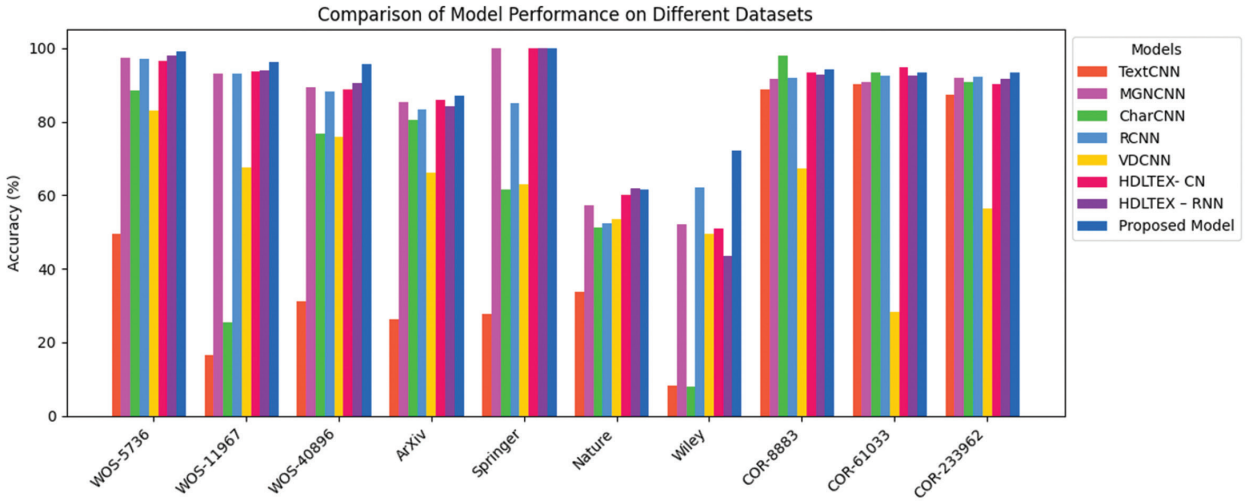

Fig. 3. Classification accuracy performance of different models

5. DISCUSSION

This section provides an analysis and discussion of the experimental results, comparing and contrasting the performance of the various models under consideration From the experimental results, it follows that models such as TextCNN capture local n-gram features and ignore long-range dependencies within the input, resulting in poor performance on the task. Models such as HDLTex CNN use a deeper network architecture with a wider range of filter sizes. Although this improves the results, the models still do not capture long-range dependencies within the text.

Models like VDCNN and CharCNN detect character n-grams rather than words and phrases within the text. The models need to be deeper, which increases the time required to train and infer. Models such as RCNN treat text as a sequence of tokens and attempt to capture long-range dependencies within the text by finding contextual representations. This is limited by the problem of vanishing gradients. HDLTex RNN uses LSTM to maintain an internal cell state and a system of gates to maintain information over an extended number of timesteps. These models are slow to train and infer because of their sequential nature.

Transformer-based models make it possible to attend to a potentially infinitely long context while encoding a given position within a text. These models, however, suffer from a large parameter space, leading to inefficiency in tasks like text classification.

The proposed model employs a single, parameter-efficient encoder block. The parameter efficiency achieved facilitates faster training and utilization. Moreover, the model significantly reduces resource requirements, making it feasible to train and deploy in low-resource environments. The proposed model outperforms previously used methods on the task.

6. CONCLUSION

Text classification, particularly in scientific literature, holds significant importance. While Transformer-based models have revolutionized NLP, applying them to text classification results in parameter-inefficient models with considerable space and time complexities. The proposed model addresses these issues by re-evaluating the architectural choices of Transformer-based models while prioritizing parameter efficiency. As a result, the model surpasses the performance of previously employed deep learning-based methods in the task of SLC, with only a fraction of the parameter space required by Transformer-based models like BERT. However, it is essential to recognize that our approach's applicability extends beyond scientific literature classification. To gauge its full potential and limitations, future research should involve its evaluation in other related tasks such as sentiment analysis, toxicity detection, and similar domains. Assessing the model's performance in diverse contexts will provide a comprehensive understanding of its capabilities and guide further refinements.

7. REFERENCES

[1] PubMed, "List of All Journals Cited in PubMed", nlm.nih.gov/bsd/serfile_addedinfo.html (accessed: 2022)

[2] M. Ware, M. Mabe, "The STM Report: An overview of scientific and scholarly journal publishing", International Association of Scientific, Technical and Medical Publishers, 2015.

[3] Y. Kim, "Convolutional neural networks for sentence classification", Proceedings of the Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, October 2014, pp. 1746-1751.

[4] S. V. Georgakopoulos, S. K. Tasoulis, A. G. Vrahatis, V. P. Plagianakos, "Convolutional Neural Networks for Toxic Comment Classification", Proceedings of the 10th Hellenic Conference on Artificial Intelligence, New York, NY, USA, July 2018, pp. 1-6.

[5] M. Hughes, I. Li, S. Kotoulas, T. Suzumura, "Medical Text Classification Using Convolutional Neural Networks", Studies in Health Technology and Informatics, Vol. 235, 2017, pp. 246-250.

[6] K. Kowsari, D. E. Brown, M. Heidarysafa, K. Jafari Meimandi, M. S. Gerber, L. E. Barnes, "HDLTex: Hierarchical Deep Learning for Text Classification", Proceedings of the 16th IEEE International Conference on Machine Learning and Applications, Cancun, Mexico, 18-21 December 2017, pp. 364-371.

[7] P. Zhou, Z. Qi, S. Zheng, J. Xu, H. Bao, B. Xu, "Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling", Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers, Osaka, Japan, 11-17 December 2016, pp. 3485-3495.

[8] B. McCann, J. Bradbury, C. Xiong, R. Socher, "Learned in translation: Contextualized word vectors", Proceedings of the 31st International Conference on Neural Information Processing Systems, Red Hook, NY, USA, December 2017, pp. 6297-6308.

[9] M. A. Wani, F. A. Bhat, S. Afzal, A. I. Khan, "Supervised Deep Learning in Face Recognition", Studies in Big Data, Vol. 57, 2020, pp. 95-110.

[10] M. A. Wani, F. A. Bhat, S. Afzal, A. I. Khan, "Supervised Deep Learning in Fingerprint Recognition", Studies in Big Data, Vol. 57, 2020, pp. 111-132.

[11] M. L. Tlachac, R. Flores, E. Toto, E. Rundensteiner, "Early Mental Health Uncovering with Short Scripted and Unscripted Voice Recordings", Deep Learning Applications, Vol. 4, 2023, pp. 79-110.

[12] X. W. Gao et al. "Identification of Human Papillomavirus from Super-Resolution Microscopic Images Generated Using Deep Learning Architectures", Deep Learning Applications, Volume 4, Advances in Intelligent Systems and Computing, Springer Nature, 2023, pp. 1-20.

[13] K. Agarwal, L. Dheekollu, G. Dhama, A. Arora, S. Asthana, T. Bhowmik, "Deep Learning-Based Time Series Forecasting", Proceedings of the 19th IEEE International Conference on Machine Learning and Applications, December 2020, pp. 859-864.

[14] J. Devlin, M. W. Chang, K. Lee, K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding", Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, June 2019, pp. 4171-4186.

[15] A. Vaswani et al. "Attention Is All You Need", Proceedings of the 31st Conference on Neural Information Processing Systems, Long Beach, CA, USA, June 2017.

[16] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, Q. V. Le, "XLNet: Generalized autoregressive pretraining for language understanding", Proceedings of the 33rd International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 2019, pp. 5753-5763.

[17] Y. Liu et al. "RoBERTa: A robustly optimized BERT pretraining approach", arXiv:1907.11692, 2019.

[18] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, R. Soricut, "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations", Proceedings of the 8th International Conference on Learning Representations, Addis Ababa, Ethiopia, 26-30 April 2020.

[19] V. Sanh, L. Debut, J. Chaumond, T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter", arXiv:1910.01108, 2020.

[20] M. A. Wani, F. A. Bhat, S. Afzal, A. I. Khan, "Advances in Deep Learning, Vol. 57", Studies in Big Data, Vol. 57, 2019.

[21] A. Mandelbaum, A. Shalev, "Word Embeddings and Their Use in Sentence Classification Tasks", arXiv:1610.08229, 2016.

[22] Y. Zhang, S. Roller, B. C. Wallace, "MGNC-CNN: A Simple Approach to Exploiting Multiple Word Embeddings for Sentence Classification", Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, June 2016, pp. 1522-1527.

[23] H. L. Xin Wu, Yi Cai, QingLi, Jingyun Xu, "Combining Contextual Information by Self-attention Mechanism in Convolutional Neural Networks for Text Classification", Web Information Systems Engineering - WISE 2013 Workshops, Vol. 8182, 2014, pp. 453-467.

[24] X. Zhang, J. Zhao, Y. LeCun, "Character-level Convolutional Networks for Text Classification", Advances in Neural Information Processing Systems, 2015, pp. 649-657.

[25] A. Conneau, H. Schwenk, Y. le Cun, L. Barrault, "Very deep convolutional networks for text classification", Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, April 2017, pp. 1107-1116.

[26] R. Johnson, T. Zhang, "Convolutional Neural Networks for Text Categorization: Shallow Word-level vs. Deep Character-level", arXiv:1609.00718, 2016.

[27] Y. Zhou, J. Li, J. Chi, W. Tang, Y. Zheng, "Set-CNN: A text convolutional neural network based on semantic extension for short text classification", Knowledge-Based Systems, Vol. 257, 2022, p. 109948.

[28] S. Lai, L. Xu, K. Liu, J. Zhao, "Recurrent convolutional neural networks for text classification", Proceedings of the National Conference on Artificial Intelligence, January 2015, pp. 2267-2273.

[29] S. Gonçalves, P. Cortez, S. Moro, "A Deep Learning Approach for Sentence Classification of Scientific Abstracts", Lecture Notes in Computer Science, 2018, pp. 479-488.

[30] D. Jin, P. Szolovits, "Hierarchical Neural Networks for Sequential Sentence Classification in Medical Scientific Abstracts", Proceedings of the Conference on Empirical Methods in Natural Language Processing, August 2018, pp. 3100-3109.

[31] A. Hassan, A. Mahmood, "Convolutional Recurrent Deep Learning Model for Sentence Classification", IEEE Access, Vol. 6, 2018, pp. 13949-13957.

[32] Y. Senarath, U. Thayasivam, "DataSEARCH at IEST 2018: Multiple Word Embedding based Models for Implicit Emotion Classification of Tweets with Deep Learning", Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, Brussels, Belgium, October 2018, pp. 211-216.

[33] Y. Li, X. Wang, P. Xu, "Chinese Text Classification Model Based on Deep Learning", Future Internet, Vol. 10, No. 11, Nov. 2018, p. 113.

[34] T. Kim, J. Yang, "Abstractive text classification using sequence-to-convolution neural networks", arXiv:1805.07745, 2018.

[35] T. Liu, S. Yu, B. Xu, H. Yin, "Recurrent networks with attention and convolutional networks for sentence representation and classification", Applied Intelligence, Vol. 48, No. 10, 2018, pp. 3797-3806.

[36] W. Chen, J. Jin, D. Gerontitis, L. Qiu, J. Zhu, "Improved Recurrent Neural Networks for Text Classification and Dynamic Sylvester Equation Solving", Neural Processing Letters, Vol. 55, 2023, pp. 8755-8784.

[37] D. Bahdanau, K. H. Cho, Y. Bengio, "Neural machine translation by jointly learning to align and translate", Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 2015, pp. 1-15.

[38] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, E. Hovy, "Hierarchical Attention Networks", Proceedings of NAACL-HLT, June 2016, pp. 1480-1489.

[39] X. Zhou, X. Wan, J. Xiao, "Attention-based LSTM network for cross-lingual sentiment classification", Proceedings of the Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 2016, pp. 247-256.

[40] X. Wu, Y. Xia, J. Zhu, L. Wu, S. Xie, T. Qin, "A study of BERT for context-aware neural machine translation", Machine Learning, Vol. 111, No. 3, 2022, pp. 917-935.

[41] S. Abdel-Salam, A. Rafea, "Performance Study on Extractive Text Summarization Using BERT Models", Information, Vol. 13, No. 2, No. 2, 2022.

[42] M. Khadhraoui, H. Bellaaj, M. Ben Ammar, H. Hamam, M. Jmaiel, "Survey of BERT-Base Models for Scientific Text Classification: COVID-19 Case Study", Applied Sciences, Vol. 12, No. 6, 2022.

[43] J. Dodge et al. "Measuring the Carbon Intensity of AI in Cloud Instances", Proceedings of the ACM International Conference Proceeding Series, Vol. 22, June 2022, pp. 1877-1894.

[44] J. Pennington, R. Socher, C. D. Manning, "GloVe: Global vectors for word representation", Proceedings of the Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, October 2014, pp. 1532-1543.

[45] K. He, X. Zhang, S. Ren, J. Sun, "Deep Residual Learning for Image Recognition", Proceedings of the IEEE conference on computer vision and pattern recognition, Las Vegas, NV, USA, 27-30 June 2016, pp. 770-778.

[46] L. J. Ba, J. R. Kiros, G. E. Hinton, "Layer Normalization", arXiv:1607.06450, 2016.

[47] Arxiv, arxiv.org (accessed: 2020)

[48] Nature, Latest research and news by subject, www.nature.com/subjects (accessed: 2020)

[49] Springer, link.springer.com (accessed: 2020)

[50] Wiley, onlinelibrary.wiley.com (accessed: 2020)

[51] Python 3 Reference Manual, docs.python.org/3.9 (accessed: 2020)

[52] Keras, keras.io/api (accessed: 2020)

[53] Tensorflow, "A system for large-scale machine learning", www.tensorflow.org/api_docs (accessed: 2020)

[54] S. Behnel, M. Faassen, I. Bicking, "lxml: XML and HTML with Python", lxml.de (accessed: 2020)

[55] Beautifulsoup, "python library for HTML and XML", crummy.com/software/BeautifulSoup (accessed: 2020)