DAAB: Deep Authorship Attribution in Bengali

Atish Kumar Dipongkor*, Md. Saiful Islam†, Humayun Kayesh†, Md Shafaeat Hossain‡, Adnan Anwar §, Khandaker Abir Rahman ¶, Imran Razzak ‖
*Jashore University of Science and Technology, Bangladesh
Email: atish.cse@just.edu.bd
†School of Information and Communication Technology, Griffith University, Australia
Emails: {saiful.islam, h.kayesh}@griffith.edu.au
‡Southern Connecticut State University, USA
Email: hossainm3@southernct.edu
§Centre for Cyber Security Research and Innovation (CSRI), Deakin University, Australia
Email: adnan.anwar@deakin.edu.au
‖School of Information Technology, Deakin University, Australia
Email: imran.razzak@deakin.edu.au
¶ Saginaw Valley State University, Michigan, USA
Email: krahman@svsu.edu

***Abstract*—Authorship attribution identifies the true author of an unknown document. Authorship attribution plays a crucial role in plagiarism detection and blackmailer identification, however, the existing studies on authorship attribution in Bengali are limited. In this paper, we propose an instance-based deep authorship attribution model, called DAAB, to identify authors in Bengali. Our DAAB model fuses features from convolutional neural networks and another set of features from an artificial neural network to learn the stylometry of an author for authorship attribution. Extensive experiments with three real benchmark datasets such as Bengali-Quora and two online Bengali Corpus demonstrate the superiority of our authorship attribution model.**

***Index Terms*—Authorship Attribution, Bengali, Convolutional Neural Network, Artificial Neural Network, Deep Learning**

I. Introduction

In this day and age, the number of internet users and their digital contents are growing exponentially over time. As a result, plagiarism detection and blackmailers identification have become crucial. Authorship Attribution (AA) aims to determine the original author of an unknown document [1]. Fig. 1 depicts the workflow of how plagiarism detection and blackmailers identification is performed using AA system. The other applications of AA systems are digital forensics investigation, author profiling, authorship verification or characterization, and detection of stylistic inconsistencies [2]. The main hypothesis behind this idea is that every author has his/her distinct writing style which is known as author's stylometry or stylo-features [3]. Although the writing style of authors may change from subject to subject, some habits or styles remain uncontrolled over time. From this persistent stylometry, the automated approaches like Machine Learning (ML) and Deep Learning (DL) models can learn and measure author's stylometry.

In ML or DL, AA is considered as a multi-class text classification problem where authors play the roles of different classes. However, the appropriate feature selection makes the major difference between AA and other text classification problems. For instance, sentiment analysis or news classification makes use of syntactic features like Parts of Speech (POS) tags (e.g., noun, adjective, and verb), punctuation (e.g., !;:?) and counts of function words (e.g., for, of). On the contrary, AA system requires stylistic features and it needs to deal with the huge number of documents having different lengths and numerous classes or authors.

Due to the exponential growth of digital content, AA is evolving as a great interest of research. Surveying the existing studies, it is found that several classical methods are applied to solve the AA problem such as Support Vector Machine (SVM) [4], Naive Bayes (NB) [5], *K*-Nearest Neighbor (KNN) [6], AdaBoost [7] and Recurrent Neural Network (RNN) [8]. In addition, these techniques are mostly applied on the English language [9], [10], [5], [11]. The other languages are also drawing attention such as Urdu [12], [13], Greek [14], [15], Arabic [10], [7], Dutch [16], [17] and Portuguese [18], [19]. Bengali is a member of the Indo-Aryan language and it is one of the most spoken languages in South Asia. Over 228 million people use Bengali in their daily lives which tends to produce a huge number of electronic contents [20]. However, AA in Bengali has received very limited attention compared to other languages. The Bengali AA problems were tried to solve using traditional ML models with some hand-crafted features [21], [22], [23]. Moreover, the experiments of these studies were not conducted under controlled environments. For example, the number of authors and documents per author was not sufficient in the experiments. Thus, a new model is essential to expedite the issues of Bengali AA. However, the challenge of developing a new model using DL for Bengali is two-folded: (i) there is a scarcity of benchmark datasets for conducting experiments and (ii) insufficient prior knowledge because DL is rarely used to solve the Bengali AA problems.

In this paper, we propose a DL model coined as DAAB for Bengali Authorship Attribution that leverages convolutional

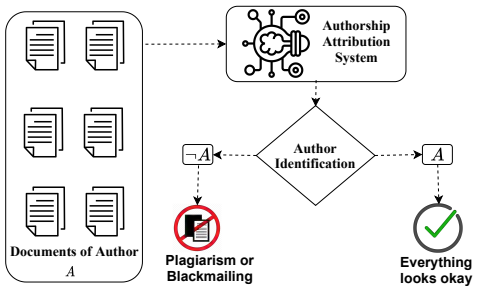


Fig. 1: Application of AA in plagiarism detection and blackmailers identification

and stylometric features from the author's writings or documents. The main architecture of our model comprises two modules such as Convolutional Neural Network (CNN) and Artificial Neural Network (ANN) modules. Here, both modules work simultaneously to acquire the highest performance. The objective of the CNN module is to capture the contextual or semantic features using convolutional filters. On the other hand, the ANN module aims to capture repeating patterns using *n*-grams. We fuse these two kinds of features in our model to learn the stylometry of an author. The experimental results show that DAAB can outperform other techniques due to considering both convolutional and stylometric features. The main contributions of this paper are listed below.

- We develop a DL model for the Bengali Authorship Attribution using convolutional and stylometric features.
- We prepare a heterogeneous dataset for conducting experiments and evaluating the benchmark models.
- We conduct extensive experiments to validate the robustness of our proposed model against other approaches.

We organise the rest of the paper as follows. Section II describes notable existing works on AA. In this section, we discuss how our approach is different compared to the techniques of other languages and Bengali. Section III describes the materials and method we used to solve the Bengali AA problem. This section is started by describing the AA problem statement (III-A). Then, we describe the dataset (III-B) and its preprocessing (III-C). In the next Sub-Section III-D, we describe our proposed model thoroughly. This Sub-Section is followed by our experiments (IV), environment (IV-A), model training (IV-C), results (IV-D) and discussion (IV-E). We conclude our study in the Section V with future directions.

II. Related Work

In this section, we review prior works on Authorship Attribution (AA) including AA in Bengali. The existing works on AA can be grouped into two categories such as (a) AA using Machine Learning (ML) based approaches and (b) AA using Deep Learning (DL) based approaches. We review here notable and recent works from each category.

A. Authorship Attribution using ML-based Approaches

To capture authors' writing styles, these approaches emphasized on punctuation, lexical and syntactic features, capitalization information, and other write prints [24], [25], [26], [27].

Olga *et al.* [5] fused different language models for AA. For building language models, they used different units of the documents, such as characters, words, or POS tags trigrams. They conducted experiments on the different types of heterogeneous datasets, e.g., movie reviews and tweets to prove the robustness of their fusion-based model. Although they insisted on the fusion of language models, the fusion-based model did not perform consistently in all datasets. For instance, the combination of characters, words, and POS trigrams performed well (96% Accuracy) in the movie review dataset but it resulted in very poor performance (52% Accuracy) in the twitter dataset.

Anderson *et al.* [4] provided a comprehensive study of the AA methods that can be applied to the short text for social media forensics. They generated *n*-grams of different lengths and applied five classification strategies such as Power Mean SVM (PMSVM), Weibull-based SVM (W-SVM), Random Forests, Source-code Author Profile, and Compression-based Attribution to select a benchmark approach. According to their experiments, PMSVM performed (70% Accuracy) better with 4-gram as a feature. Although this study reported promising results for short-sized texts using 4-grams, we find that 1-gram and 2-gram repeat frequently more than other *n*-grams in large documents. In another work, Laura [28] applied graph2vec [29], a graph embedding technique for Authorship Recognition in short text using SVM. However, it may arise high computational complexities to apply graph embedding techniques for large documents. Thus, it might not be effective for large documents to use the same features of short texts.

Waheed *et al.* [13] presented a Latent Dirichlet Allocation (LDA) based approach for authorship identification in English and Urdu. In this work, LDA is used to handle a high-dimensional document term matrix. However, applying LDA in the Bengali document is not straightforward because it requires two types of preprocessing such as stemming and POS tagging. The Bengali NLP is yet to find effective Stemmer and POS tagger tools like other languages [30]. For these issues, we use a different matrix decomposition technique called Singular Value Decomposition (SVD) to handle high-dimensional sparse data in our approach.

In Bengali, ML techniques are mostly applied to literary texts to attribute the authors of those texts [21], [23], [22]. In these studies, they used a homogeneous corpus, and the number of authors was limited. The choice of different features (e.g., semantic and stylistic) makes a significant difference between our work and existing Bengali AA works. Moreover, we evaluate our approach on heterogeneous datasets (documents covering diverse topics) with a large number of authors.

B. Authorship Attribution using DL-based Approaches

To solve AA problems of different fields (source code, news, and documents), existing studies mostly used Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) based DL models. Among these works, we summarize recent works from each field in this section.

TABLE I: Dataset statistics

Dataset	Source	#Words	#Docs	# Authors	Docs/Author
BQC	Bengali-Quora	6.54 M	19924	95	100-1122
OBC1	Online [35]	2.3 M	2100	6	350
OBC2	Online [36]	13.4 M	17966	16	185-4518

Farhan *et al.* [31] proposed Program Dependence Graph with Deep Learning (PDGDL) methodology to identify authors from different programming source codes. In this work, Program Dependence Graph (PDG) is used to identify the coding styles of the authors. Then, a DL model with a dropout and dense layer is trained to predict the author of source codes. Although this approach achieved better accuracy in identifying authors from different programming languages (C++, Java, and C#), it cannot be adopted to identify the authors of large documents. The reason is: coding style of an author largely depends on the control and data dependencies of the programs whereas the writing style depends on the stylometric markers.

Liuyu *et al.* [8] applied RNN to identify the authors of a news corpus. Using RNN, they tried to capture the word or sentence sequence information for classifying the authorship of news articles. However, the word or sentence sequence information did not perform well to capture stylometric features. According to the experimental results of this study, RNN has achieved 0.6 f1-score so far. Prasha *et al.* [32] presented a CNN model to perform AA over short text. Initially, this model takes a sequence of character *n*-grams as input. Then, these *n*-grams are processed by three modules sequentially such as character embedding, convolutional, and fully connected softmax module. Although this model outperformed traditional ML models over short text, CNN based on character *n*-grams did not perform up to the mark over large documents [33].

For Bengali AA, Hemayet *et al.* [34] performed a comparative analysis of several word embedding techniques such as fastText (skip-gram & CBOW), word2vec (skip-gram & CBOW), and glove. In their work, these word embeddings were used by the deep layers of different neural networks, e.g., ANN, RNN, and CNN. To analyze the performance of these embeddings, they trained ANN, RNN, and CNN models with 80% of 2100 documents, and the 20% documents are used for performance testing. According to their experiments, fastText (skip-gram) embedding with CNN performed better (92.9% Accuracy) than other embeddings. In our work, we achieve higher performance (95.28% Accuracy) on the same 2100 documents by applying our model even after using 10-fold cross-validation.

III. Materials and Methods

This section formally states the authorship attribution problem, datasets used, data preprocessing techniques used in this study, and the proposed deep authorship attribution model.

A. Problem Statement

Assume authors $\mathcal{A} = \{A_1, A_2, ..., A_i\}$ own the documents $\mathcal{D} = \{D_{11}, D_{22}, ..., D_{ij}\}$, where D_{ij} denotes the jth document of the ith author. Given a document D'_{ij} whose author is unknown, we aim to identify the author A'_i for D'_{ij}. It should be noted that the document D'_{ij} has the similarities in terms of stylometry with some of the documents D_{ij} in the collection $\mathcal{D}$ and the author A'_i of D'_{ij} is one of the authors in $\mathcal{A}$.

TABLE II: Number of unique {1,2,3}-grams in each dataset

Dataset	#1-grams	#2-grams	#3-grams
BQC	465971	3517465	5755598
OBC1	229306	1426315	2025840
OBC2	590660	6149483	11534258

B. Dataset

We evaluate authorship attribution models by conducting experiments on three benchmark datasets. Table I presents the different statistics of the datasets. Among these datasets, we prepare one of them, and the other two are publicly available. A brief overview of these datasets is provided below.

- **Bengali-Quora Corpus (BQC):** We prepare this dataset from Bengali-Quora[1]. In Quora platform, plagiarism is strongly prohibited. Since plagiarism detection is one of the major applications of Authorship Attribution (AA), we consider Bengali-Quora as a viable source of AA data. Moreover, it contains different types of writings (history, traveling, education, politics and etc.) of the same author. For collecting data from this source, we implement a custom web-scrapper using Python[2]. This dataset contains 6.54 million words and 0.57 million sentences of 95 authors. The minimum and maximum documents per author vary between 100 to 1122.
- **Online Bengali Corpus ONE (OBC1) [35]:** This dataset was prepared from online blogs. It consists of writings from 6 authors. Each author has the same number of documents which is 350 per author. The total word count in this dataset is 2.3+ million.
- **Online Bengali Corpus TWO (OBC2) [36]:** This dataset contains writings of 16 Bengali prominent authors having 13.4 million words. It contains 17966 documents in total, and each document has an equal word length which is 750 words per document. The number of documents is between 185 to 4518 per author.

C. Data Preprocessing

According to the existing studies [23], [13], [4], rigorous data preprocessing is not essential for the automated Authorship Attribution. For instance, frequent usages stop words, grammatical mistakes, and usages of inflected[3] words are often considered as authors' writing style. Thus, eliminating stop words, fixing grammatical errors, and stemming words are supposed to reduce stylometric information from the authors' documents. To this end, we perform very minimal data preprocessing which is summarized below.

1) **Tokenization:** Tokenization is a required step for word embedding. In this study, we perform word-level tokenization for each document by ignoring white spaces

[1]https://bn.quora.com
[2]https://www.python.org
[3]Words with suffix or prefix

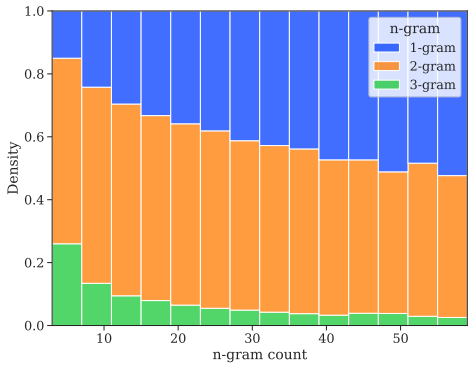


Fig. 2: $\{1,2,3\}$-gram distributions in Bengali-Quora Corpus

between two words. Here, no stemming or lemmatization is performed on the words. Moreover, we did not remove any punctuation marks before tokenization.

2) ***n*-gram Generation:** *n*-gram is considered as a very useful technique for capturing authors' writing structure [13], [37] because it can isolate repeating patterns from documents. There are two types of word *n*-grams such as (i) overlapping and (ii) non-overlapping. In this study, we use the overlapping word *n*-gram. Table II demonstrates the number of unique $\{1,2,3\}$-grams from each dataset. The value of n while *n*-gram generation should be chosen carefully because the too small or too large value of n may fail to capture the important patterns. Fig. 2 illustrates $\{1,2,3\}$-gram distributions in Bengali-Quora Corpus where X-axis represents the count of different *n*-grams and Y-axis represents the density of those *n*-grams. From this figure, it is evident that 1-gram and 2-gram repeat frequently compared to 3-gram. This study uses 1- and 2-grams for isolating writing patterns of the authors due to the repeating characteristics of $\{1,2\}$-grams.
3) **TF-IDF Vectorization:** Term Frequency-Inverse Document Frequency (TF-IDF) discriminates each document by providing different weights to the words. When it is applied to the Authorship Attribution, significant terms or *n*-grams of a particular author receive more importance than others. For this reason, we vectorize documents using the TF-IDF values of *n*-grams.
4) **Document-Term Matrix:** The documents in a dataset or corpus can be represented by a document-term matrix whose rows correspond to documents and columns correspond to the distinct terms from all documents. As far as this study is concerned, columns represent the unique *n*-grams instead of terms or words. Formally, let $M_{DT} \in \mathbb{R}^{j\times t}$ is a document-term matrix with dimension $j \times t$ where j is the number of documents and t is the number of distinct *n*-grams. If document j contains *n*-gram t then $M_{DT}[j,t] = b$ where b indicates the TF-IDF value of *n*-gram t. It is worth noting here that same *n*-gram does not occur in all documents. For instance, *n*-gram t frequently occurs in the author A_i's documents but not in A_i's and which turns M_{DT} into a sparse matrix.
5) **Dimensionality Reduction:** In deep or machine learning, handling higher dimensional sparse matrix is not a trivial task. For example, if we consider 1-grams and 2-grams as the features for BQC dataset, the dimension of M_{DT} will be $19924 \times 3.98M$. For this reason, we consider to reduce the dimension of M_{DT} using a well-known matrix factorization technique, Singular Value Decomposition (SVD). According to SVD, M_{DT} can be factored as $U^{j\times r} \cdot S^{r\times r} \cdot V^{r\times t}$. Then, we reduce the dimension of M_{DT} by choosing t' components from $S^{r\times r}$ and t' columns from $U^{j\times r}$ where t' is always less than r. Mathematically, it can be written as $M'_{DT} = U^{j\times t'} \cdot S^{t'\times t'}$ where the dimension of M'_{DT} is $j\times t'$. After this SVD operation, the jth row of M'_{DT} basically represents a vector pointing to the document D_{ij}.

Although we conduct this study on the Bengali dataset, we find that some Bengali authors quote English sentences in their writings. Moreover, they provide one or more hyperlinks in their documents. Thus, we lowercase English words and remove hyperlinks from the datasets to make them consistent.

D. Proposed Deep Authorship Attribution Model

Our proposed deep authorship attribution model for Bengali (DAAB for short) aims to solve authorship attribution problem using Deep Learning (DL) for Bengali. Initially, it takes a document as input and captures semantic and stylistic features using Convolutional Neural Network (CNN) and Artificial Neural Network (ANN) module, respectively. Additionally, it has an Information Fusion (IF) layer that basically concatenates the outputs of CNN and ANN layers. Then, the yields of the IF layer is sent to a dense layer. Finally, a softmax layer is used to determine the author of a given document. Overall illustrative architecture of our model is shown in Fig. 3.

The CNN module in our model consists of the followings.

- **Embedding Layer:** After receiving a document D_{ij} as input, this layer transforms its text into an embedding matrix. Let, $v_e \in \mathbb{R}^d$ is a d dimensional vector that represents the eth word of D_{ij}. Now, the embedding matrix $M_{eb} \in \mathbb{R}^{n\times d}$ is obtained by concatenating all vectors that represent the words of D_{ij}. Here, n is the maximum number of words and d is the embedding dimension. If an input document does not contain the n number of words, padding is applied. Formally, $M_{eb} \in \mathbb{R}^{n\times d}$ can be written as follows.

$$M_{eb} = v_1 \oplus v_2 \oplus ... \oplus v_n \tag{1}$$

- **Convolution Filter Layer:** To extract semantic or contextual feature, this layer applies different filters $W \in \mathbb{R}^{h\times d}$ on M_{eb}. Here, h denotes the number of words to capture at a time which is also called window or kernel size. For example, if $h = 2$ is chosen, the filter $W \in \mathbb{R}^{2\times d}$ will capture the semantic relationship between two adjacent words. Similarly, applying filter $W \in \mathbb{R}^{h\times d}$ by sliding to the all possible window of words $[v_e : v_{e+h-1}]$ will generate a new feature s_j which can be denoted as follows:

$$s_j = g(W \cdot [v_e : v_{e+h-1}] + b_c) \tag{2}$$

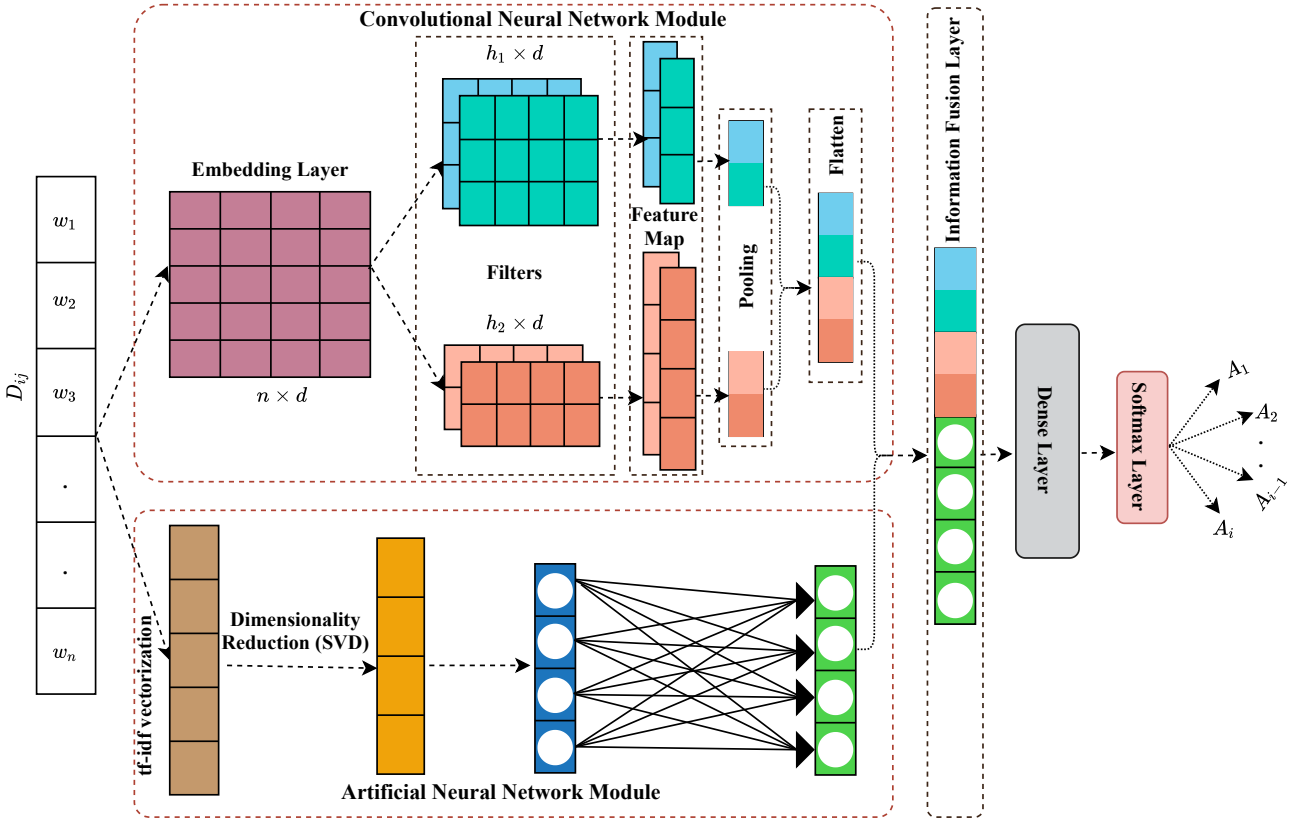


Fig. 3: Architecture of DAAB for Deep Authorship Attribution in Bengali

Here, $b_c \in \mathbb{R}$ is a bias term and g is non-linear function. For this layer, we use ReLU as non-linear function.

- **Feature Map Layer:** After applying s_j (Equation 2) to each possible window of M_{eb}, a feature map $C_j \in \mathbb{R}^{n-h+1}$ is obtained which can be expressed as follows.

$$C_j = [C_1, C_2, ...C_{n-h+1}] \tag{3}$$

- **Pooling Layer:** This layer further abstracts the features from feature maps. In this study, we use max-over-time pooling operation and select the maximum value $\hat{C}_j = max(C_j)$ from a particular filter s_j. Here, the idea is to capture the most important feature from C_j.
- **Flatten Layer:** In the convolution filter layer, various filters with different values of h are used to increase the feature coverage of the model. To that end, this layer aggregates all the max-pooled features, $V_{ccn} \in \mathbb{R}^p$ which can be treated as the final outcome of the CNN module. Here, p denotes the number of filters.

The ANN module consists of the following components.

- **Input Layer:** This layer takes a reduced TF-IDF vector $v_j \in \mathbb{R}^{t'}$ that represents the input document D_{ij}. Here, the $v_j \in \mathbb{R}^{t'}$ is obtained from the jth row of M'_{DT} (the reduced document-term matrix M'_{DT} is explained in Section III-C). In other words, this layer utilizes stylometric features from the author's documents.
- **Hidden Layer:** After receiving $v_j \in \mathbb{R}^{t'}$ from previous layer, this hidden layer assigns a weight vector $\mathcal{W}$ to $v_j \in \mathbb{R}^{t'}$. Formally, it can be expressed as follows.

$$x_i = f(\mathcal{W} \cdot [v_j^1 : v_j^{t'}] + b_a) \tag{4}$$

Here, $b_a \in \mathbb{R}$ is the bias and f is a non-linear function (e.g., ReLU). After assigning weight $\mathcal{W}$ to v_j using Equation 4, a new vector $V_{ann} \in \mathbb{R}^{t'}$ is obtained to capture the deep sytlometric features of an author in a document.

The IF layer fuses both semantic (V_{cnn}) and stylistic (V_{ann}) feature obtained from the CNN and ANN layers in order to learn the stylometry in the fused feature space. Formally, the fused features V_{if} of a document D_{ij} is calculated as follows:

$$V_{if} = V_{cnn} \oplus V_{ann} \tag{5}$$

where, $\oplus$ denotes the concatenation. Next, the outcome of IF layer $V_{if} \in \mathbb{R}^{p+t'}$ is sent to a non-linear hidden layer for assigning weights to the each element of V_{if}. Finally, we use an output layer to convert the values of the above hidden layer into probabilities for classification. In our model, softmax activation is used in the output layer. The output layer assigns a different probability to each author but the original author will receive the highest probability. For example, D_{ij} is sent to our model for authorship attribution which is the jth document of ith author A_i. Then, the main target of the output layer is to assign the highest probability to A_i among all authors.

IV. Experiments

To evaluate our proposed model and compare it with the exiting benchmark models, we conduct extensive experiments. Initially, we apply our model and other models to all datasets (BQC, OBC1, and OBC2). During the training, we use k-fold cross-validation for evaluating the performance of our benchmark models. Since our dataset possesses heterogeneous characteristics, the train-test splitting method for performance evaluation may arise randomness in the results. Thus, we choose k-fold cross-validation to measure the performance of our benchmark models. Moreover, we tune different parameters to avoid over and under fittings of the models, which is also presented graphically in this section.

A. Environment

We conduct the experiments using 12 core 3.80 GHz, 64-bit Windows 10 operating system. This machine is equipped with 64 GB of memory and 8 GB of video memory. We implement the DL models using keras (keras.io) and traditional ML models using scikit-learn (scikit-learn.org).

B. Benchmark Models

To evaluate the robustness of our approach, we compare it against four traditional Machine Learning (ML) models such as Support Vector Machine (SVM), Naive Bayes (NB), Random Forest (RF), and Gradient Boosting (XGBoost). These ML models were frequently used in the previous works [4], [5], [7], [4], [23] to performs Authorship Attribution (AA). To train these models, we use vector $v_j \in \mathbb{R}^{t'}$ that represent a particular document D_{ij} of author A_i. It is noteworthy that traditional ML models are trained using stylometric features only since they are unable to learn from the heterogeneous features. Lastly, we create different variants of our DAAB model to perform an ablation study. A brief description of the DAAB variants is given below.

- **CNN2+CNN3+ANN:** This model comprises two CNN layers having different kernel sizes and an ANN layer as depicted in Fig. 3. An embedding layer is used as the input of these CNN layers. Here, the embedding matrix or layer is non-static, i.e., the vectors are modified during training using backpropagation. Then, a max-over-time pooling operation is applied to select the important features from CNN layers. The input of ANN layer is a TF-IDF vector that basically represents the stylometric features of a given document. Then, the size of that TF-IDF vector is reduced using SVD. After that, a fully connected dense layer assigns different weights to the reduced TF-IDF vector. Then, the IF layer fuses the outcome of the CNN and ANN layers, and the fused outcome is passed through a dense layer. Finally, a softmax layer predicts the author of an unknown document.
- **CNN2+CNN3:** This model is developed by removing the ANN layer from the above CNN2+CNN3+ANN model. After the max-pooling operation over two CNN layers, the outcome is processed via a dense and softmax layer.

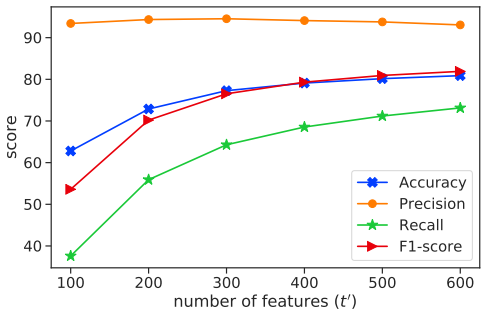


Fig. 4: Effect of t' on the ANN based AA model's performance such as Accuracy, Precision, Recall and F1-score

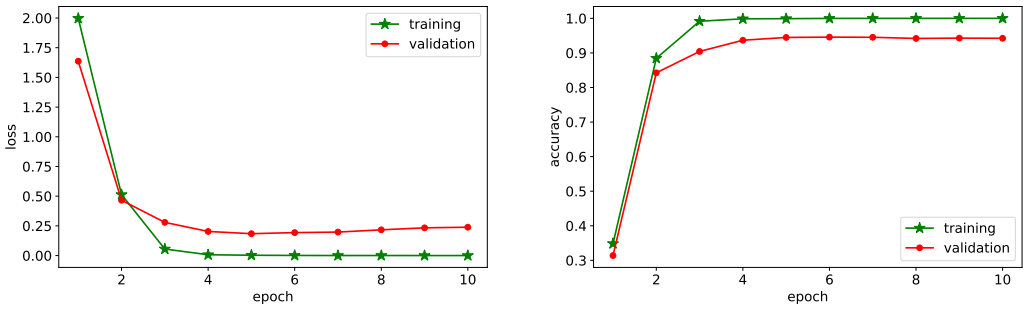
- **CNN2+ANN:** This model consists of a single CNN and ANN layer. The input of these layers is an embedding matrix and a reduced TF-IDF vector. The outcome of these two layers is fused in the IF layer, and which is followed by a dense and softmax layer for the final prediction.
- **ANN:** This model contains only a single hidden layer (no CNN layer is involved). The reduced TF-IDF stylometric vector is used as the input for the hidden layer. Lastly, the softmax layer determines the author of a given document.

C. Model Training and Evaluation

To train the variants of DAAB, we use different parameter settings. We train ANN variant using 600 features which means the value t' is 600 in other terms. Fig. 4 displays the effect of different values of t' on ANN's performance. It can be observed from Fig. 4 that the performance (F1-score) does not vary significantly between 500 and 600 features. In CNN2+ANN training, the ANN settings are the same as above. For CNN2, we use 256 filters over a 100-dimensional embedding vector. Each filter captures two words at a time which is the window or kernel size (i.e., 2×100) in our case. In order to train CNN2+CNN3, the settings of CNN2 are the same as that of the CNN2 of CNN2+ANN model. The CNN3 uses a different kernel size, which is 3 (i.e., 3×100 kernel) and other settings are the same as that of the CNN2 of CNN2+ANN model. Finally, we train CNN2+CNN3+ANN by combining the settings of CNN2+CNN3 and ANN. Apart from these settings, we use the same batch size (32), epochs (1 to 10), loss function (categorical crossentropy), and optimizer RMSprop to train all the variants of our DAAB model.

Fig. 5a and Fig. 5b display the loss and accuracy on training and validation, respectively for the epoch 1 to 10. From these figures, it is evident that we can obtain optimum performance in between epoch 4 to 8 and using other necessary settings. Moreover, it is also apparent from these figures that these settings are able to avoid over-fitting and under-fitting of the DAAB variants. Therefore, we do not use any dropout layer in the training. In traditional ML models training, we use the same 600 features as mentioned in ANN so that we can compare ML models with the variants of our DAAB model.

To estimate the performance of our benchmark models, we apply 10-fold cross-validation during the training of these



(a) Loss on training and validation

(b) Accuracy on training and validation

Fig. 5: Effect of epochs on the training and validation of our DAAB model (CNN2+ANN)

models. In each fold, we stratify the training and validation data. Moreover, we measure Accuracy, Precision, Recall, and F1-score metrics for each fold. After the last or 10th fold, we average these metrics to obtain the final scores of our benchmark models. To compare models between themselves, we use F1-score since it penalizes low Precision or Recall.

D. Results and Discussion

In this section, we present our experimental results of the traditional ML models and our DAAB variants which we obtain from the three benchmark datasets (BQC, OBC1, and OBC2) using the settings stated earlier (Section IV-C).

Table III displays the performance of traditional ML models (RF, SVM, XGBoost and NB) in terms of Accuracy, Precision, Recall and F1-score metrics. From this table, it is evident that SVM outperforms all other models in respect of all metrics. In BQC and OBC1 datasets, SVM beats other models notably, i.e., it achieved 74.72 ± 4.02 and 93.91 ± 2.07 F1-scores, respectively whereas its nearest competitor, XGBoost achieved 58.94 ± 4.17 and 91.75 ± 2.58 F1-scores, respectively. In OBC2 dataset, there is no significant difference between the F1-scores of SVM (99.78 ± 0.11), RF (99.49 ± 0.12) and XGBoost (99.30 ± 0.17). Amid these ML models, NB is seen to be less performed in terms of F1-scores in OBC1 (60.74 ± 4.44) and OBC2 (88.97 ± 0.56) datasets. In the BQC dataset, RF is found to be less performed. Since SVM has shown higher performance in terms of F1-score than other ML models, we consider it as a baseline for comparing the DAAB variants.

Table IV shows the performance of our DAAB variants (ANN, CNN2+ANN, CNN2+CNN3, and CNN2+CNN3+ANN) in terms of the metrics stated above. Moreover, this table also contains the average improvement in the F1-score of each DAAB variant compared to RF, SVM, XGBoost, and NB. From the last column of Table IV, it can be observed that the average improvement in F1-scores of all DAAB variants is remarkable in all datasets compared to RF, XGBoost, and NB. Apart from these non-baseline models, our DAAB variants achieve comparable performance in the OBC2 dataset compared to SVM. It is worth noting that the OBC2 dataset contains the writings of some Bengali veteran authors having distinctive writing styles. As a result, all ML models and our

TABLE III: Performance of traditional ML models

Dataset	Models	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
BQC	RF	52.84±3.29	54.51±3.00	52.84±3.29	49.14±3.14
	SVM	**74.29±4.31**	**78.94±3.26**	**74.29±4.31**	**74.72±4.02**
	XGBoost	60.24±4.28	60.13±4.08	60.24±4.28	58.94±4.17
	NB	47.39±3.29	60.72±2.80	47.39±3.29	50.00±3.44
OBC1	RF	90.76±3.08	91.35±2.91	90.76±3.08	90.74±3.15
	SVM	**93.90±2.08**	**94.20±1.98**	**93.90±2.08**	**93.91±2.07**
	XGBoost	91.76±2.50	92.11±2.45	91.76±2.50	91.75±2.58
	NB	61.81±4.27	65.16±4.49	61.81±4.27	60.74±4.44
OBC2	RF	99.49±0.12	99.49±0.12	99.49±0.12	99.49±0.12
	SVM	**99.78±0.11**	**99.78±0.10**	**99.78±0.11**	**99.78±0.11**
	XGBoost	99.30±0.17	99.31±0.17	99.30±0.17	99.30±0.17
	NB	88.91±0.58	89.69±0.51	88.91±0.58	88.97±0.56

DAAB variants have shown equal learning capabilities and resulted in similar performance (99%+ F1-score).

In the OBC1 dataset, two DAAB variants namely, CNN2+ANN and CNN2+CNN3+ANN slightly beat the baseline SVM. For instance, the average improvement in F1-scores of these models are 1.46% and 0.67%, respectively compared to SVM. Since OBC1 is a balanced dataset it is not difficult to learn the stylometry from these data. For instance, the F1-scores do not vary a lot between CNN2+ANN (95.28 ± 0.10), CNN2+CNN3+ANN (94.54 ± 0.16), SVM (93.90 ± 2.08), XGBoost (91.76 ± 2.50) and RF (90.76 ± 3.08).

In the BQC dataset, all DAAB variants have shown significant improvement compared to SVM which can be the main point of interest as BQC is an imbalanced dataset with a large number of authors. Among all variants of DAAB, CNN2+ANN and CNN2+CNN3+ANN are observed as the highly beating model of SVM in the BQC dataset. For example, their average improvement in F1-scores is 20.54% and 21.94% compared to SVM. In addition, they are not only the beating model of ML models, they perform better than other DAAB variants too. For instance, they achieve the highest Recalls and F1-scores in all datasets which indicates an expected outcome, i.e., they are very good at identifying the positive classes while classifying the documents. Due to this outstanding performance of CNN2+ANN and CNN2+CNN3+ANN, it validates the main idea of our research, i.e., information fusion. In ANN and CNN2+CNN3, there is no information fusion such as ANN uses *n*-grams or stylometric features only and CNN2+CNN3 uses convolutional or contextual features only. From Table IV, it can be observed that the models that

TABLE IV: Performance of the DAAB variants: ANN, CNN2+ANN, CNN2+CNN3, CNN2+CNN3+ANN

Dataset	Models	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Avg. Improvement in F1-Sore (%)			
						RF	SVM	XGBoost	NB
BQC	ANN	**80.74±0.74**	93.16±0.70	73.01±1.00	81.86±0.82	> 66.59	> 09.56	> 38.89	> 63.72
	CNN2+ANN	74.08±1.50	98.05±0.03	83.30±0.24	90.07±0.14	> 83.29	> 20.54	> 52.82	> 80.14
	CNN2+CNN3	65.02±1.70	97.56±0.06	79.64±0.54	87.69±0.34	> 78.44	> 17.36	> 48.79	> 75.38
	CNN2+CNN3+ANN	75.78±1.10	**98.31±0.02**	**84.88±0.16**	**91.11±0.09**	> 85.41	> 21.94	> 54.85	> 82.22
OBC1	ANN	93.33±1.71	95.61±1.60	90.28±2.58	92.86±1.92	> 02.34	< 01.12	> 01.21	> 52.88
	CNN2+ANN	**96.71±1.46**	**99.57±0.09**	**91.36±0.20**	**95.28±0.10**	> 05.00	> 01.46	> 03.85	> 56.87
	CNN2+CNN3	95.14±1.40	98.82±0.09	86.65±0.70	92.33±0.40	> 01.75	< 01.68	> 00.63	> 52.01
	CNN2+CNN3+ANN	96.42±1.20	99.44±0.06	90.10±0.30	94.54±0.16	> 04.18	> 00.67	> 03.04	> 55.65
OBC2	ANN	**99.78±0.07**	**99.83±0.05**	99.68±0.10	**99.76±0.06**	> 00.27	< 00.02	> 00.46	> 12.13
	CNN2+ANN	99.76±0.13	99.76±0.12	**99.75±0.13**	**99.76±0.13**	> 00.27	< 00.02	> 00.46	> 12.13
	CNN2+CNN3	99.62±0.11	99.63±0.10	99.61±0.11	99.62±0.11	> 00.13	< 00.16	> 00.32	> 11.97
	CNN2+CNN3+ANN	99.75±0.12	99.75±0.12	99.74±0.12	99.74±0.12	> 00.25	< 00.04	> 00.44	> 12.12

utilize single information perform less. On the other hand, the models that utilize different types of features or fuse information perform remarkably. For example, CNN2+ANN and CNN2+CNN3+ANN leverage information fusion and achieve better performance than the other two models that do not fuse information namely, ANN and CNN2+CNN3.

E. Deep Error Debugging and Future Work

As it is observed that fusion-based models for Authorship Attribution (AA) perform significantly better than other models, we carry out extensive error debugging for one of them such as CNN2+ANN over the OBC1 dataset. Initially, we note all the incorrect author attributions of CNN2+ANN during the training and validation for each fold (1 to 10) and find that incorrect author attributions are coinciding. For instance, author A_1's documents have been incorrectly attributed as author A_4's documents 15 times and author A_4's documents have been incorrectly attributed as author A_1's documents 8 times by our CNN2+ANN model. So, the total incorrect coinciding between A_1 and A_4 is 23 times. Fig. 6a depicts the total number of times incorrect attributions coincide between two authors. From this figure, it is evident that the most coinciding authors in terms of incorrect attributions are $\{A_1, A_4\} = 23$, $\{A_2, A_6\} = 15$, $\{A_4, A_5\} = 15$, $\{A_4, A_6\} = 14$. To comprehend this phenomenon, we measure the stylometric similarities between authors. Let, D_i is a hyper document consisting of all documents of author A_i and is formed as follows.

$$D_i = D_{i1} \oplus D_{i2} \oplus ... \oplus D_{ij} \tag{6}$$

Then, we obtain a global document D^g by concatenating the documents of all authors which can be formalized as follows.

$$D^g = D_1 \oplus D_2 \oplus ... \oplus D_i \tag{7}$$

Then, we generate 1-gram and 2-gram for D^g and for all D_is. Using these n-grams or terms, we obtain author-terms matrix, $M_{AT} \in \mathbb{R}^{i \times t}$ where i and t represent the number of authors and unique n-grams, respectively. After that, we apply SVD to reduce the dimension of $M_{AT} \in \mathbb{R}^{i \times t}$ to $M_{AT} \in \mathbb{R}^{i \times t'}$ where the value of t' is very less than t. In OBC1 dataset, we find 1.66M unique n-grams and reduce its size to 600 for maintaining the accordance with other experiments. After the SVD operation, the ith row of $M_{AT} \in \mathbb{R}^{i \times t'}$ basically represents a document vector or profile of author A_i in the reduced dimension which can be denoted as A_i^v. Finally, we measure the stylometric similarities between authors using cosine similarity. For example, the stylometric similarity between two authors A_i and $A_{i'}$ is calculated using cosine similarity as given below.

$$\cos(A_i, A_{i'}) = \frac{A_i^v \cdot A_{i'}^v}{\|A_i^v\| \|A_{i'}^v\|} \tag{8}$$

Fig. 6b represents the stylometric similarities between authors which we measure using the equation 8. Now, if we look at the same cells of Fig. 6a and 6b, we find that higher stylometric similarity tends to yield a higher number of coinciding incorrect attributions. For example, the stylometric similarity between A_1 and A_4 is more than 89% which resulted in 23 times coinciding incorrect attributions between them. At this moment, we verify whether this relationship exists for all other authors. To this end, we measure the correlation between stylometric similarities and the total number of coinciding incorrect attributions. From Fig. 6a, it can be observed that these two elements have a moderate positive relationship as the value of the correlation coefficient is 0.65. In other terms, we can say that the stylometric similarities between authors moderately cause coinciding incorrect attributions between them. In our future studies, we aim to fix these issues and make our fusion-based models such as CNN2+ANN and CNN2+CNN3+ANN more robust.

V. Conclusion

In this work, we develop a deep learning model to perform Authorship Attribution (AA) in a low-resourced language, Bengali. Existing works in this domain utilize stylometric features only whereas our model fuses stylometric and convolutional features to perform AA. To validate that information fusion is a reliable way to perform AA in Bengali, we conduct extensive experiments over three datasets. Our experimental results show that the fusion-based model is able to beat the traditional machine learning models significantly. We also perform extensive deep error debugging of our model to find the root causes of incorrect author attributions. In our future studies, we aim to resolve the root causes of incorrect author attributions and apply our model to other low-resourced Asian languages to demonstrate its effectiveness.

(a) Coinciding incorrect attribution (b) Similarity between authors' documents (c) Correlation between Fig. 6b and Fig. 6a

Fig. 6: Overview of error debugging of CNN2+ANN model over OBC1 dataset: (a) Coinciding incorrect attribution between authors. Each cell represents the total number of times incorrect attributions coincide between two authors; (b) Similarity between authors' in terms of their all documents. Each cell represents the similarity between two authors and (c) Correlation between document similarities and the total number of times coinciding incorrect attributions between two authors

ACKNOWLEDGEMENT

We wish to appreciate the Department of Computer Science and Engineering (CSE), Jashore University of Science and Technology (JUST). We conduct all experiments in the Data Science and Machine Learning (DSML) lab of CSE, JUST.

REFERENCES

[1] P. Juola, "Authorship attribution," *Foundations and Trends® in Information Retrieval*, vol. 1, no. 3, pp. 233–334, 2008.

[2] E. Stamatatos, "A survey of modern authorship attribution methods," *Journal of the American Society for Information Science and Technology*, vol. 60, no. 3, pp. 538–556, 2009.

[3] Y. Zhao, J. Zobel, and P. Vines, "Using relative entropy for authorship attribution," in *Asia Information Retrieval Symposium*. Springer, 2006, pp. 92–105.

[4] A. Rocha, W. J. Scheirer, C. W. Forstall, T. Cavalcante, A. Theophilo, B. Shen, A. R. Carvalho, and E. Stamatatos, "Authorship attribution for social media forensics," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 5–33, 2016.

[5] O. Fourkioti, S. Symeonidis, and A. Arampatzis, "Language models and fusion for authorship attribution," *Information Processing & Management*, vol. 56, no. 6, p. 102061, 2019.

[6] C. Akimushkin, D. R. Amancio, and O. N. Oliveira Jr, "On the role of words in the network structure of texts: Application to authorship attribution," *Physica A: Statistical Mechanics and its Applications*, vol. 495, pp. 49–58, 2018.

[7] M. Al-Sarem, F. Saeed, A. Alsaeedi, W. Boulila, and T. Al-Hadhrami, "Ensemble methods for instance-based arabic language authorship attribution," *IEEE Access*, vol. 8, pp. 17 331–17 345, 2020.

[8] L. Wang, "News authorship identification with deep learning," 2017.

[9] C. E. Chaski, "Empirical evaluations of language-based author identification techniques," *Forensic Linguistics*, vol. 8, pp. 1–65, 2001.

[10] A. Abbasi and H. Chen, "Applying authorship analysis to extremist-group web forum messages," *IEEE Intelligent Systems*, vol. 20, no. 5, pp. 67–75, 2005.

[11] M. Koppel, J. Schler, and S. Argamon, "Computational methods in authorship attribution," *Journal of the American Society for Information Science and Technology*, vol. 60, no. 1, pp. 9–26, 2009.

[12] A. A. Raza, A. Athar, and S. Nadeem, "N-gram based authorship attribution in urdu poetry," in *NAACL*, 2009, pp. 88–93.

[13] W. Anwar, I. S. Bajwa, and S. Ramzan, "Design and implementation of a machine learning-based authorship identification model," *Scientific Programming*, vol. 2019, 2019.

[14] E. Stamatatos, N. Fakotakis, and G. Kokkinakis, "Text genre detection using common word frequencies," in *COLING*, 2000.

[15] V. Kešelj, F. Peng, N. Cercone, and C. Thomas, "N-gram-based author profiles for authorship attribution," in *PACLING*, 2003, pp. 255–264.

[16] P. Juola and R. H. Baayen, "A controlled-corpus experiment in authorship identification by cross-entropy," *Literary and Linguistic Computing*, vol. 20, no. Suppl, pp. 59–67, 2005.

[17] P. Maitra, S. Ghosh, and D. Das, "Authorship verification-an approach based on random forest," *arXiv preprint arXiv:1607.08885*, 2016.

[18] R. S. Silva, G. Laboreiro, L. Sarmento, T. Grant, E. Oliveira, and B. Maia, "'twazn me!!!;('automatic authorship analysis of micro-blogging messages," in *NLDB*, 2011, pp. 161–168.

[19] I. Markov, J. Baptista, and O. Pichardo-Lagunas, "Authorship attribution in portuguese using character n-grams," *Acta Polytechnica Hungarica*, vol. 14, no. 3, pp. 59–78, 2017.

[20] Wikipedia, "List of languages by number of native speakers," https://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers, [Online], [Accessed 2020-02-07].

[21] S. Das and P. Mitra, "Author identification in bengali literary works," in *PReMI*. Springer, 2011, pp. 220–226.

[22] S. Phani, S. Lahiri, and A. Biswas, "Authorship attribution in bengali language," in *ICON*, 2015, pp. 100–105.

[23] ——, "A supervised learning approach for authorship attribution of bengali literary texts," *ACM Trans. Asian Low Resour. Lang. Inf. Process.*, vol. 16, no. 4, pp. 1–15, 2017.

[24] M. Kestemont, "Function words in authorship attribution. from black magic to theory?" in *CLFL*, 2014, pp. 59–66.

[25] M. Eder, "Does size matter? authorship attribution, small samples, big problem," *Digital Scholarship in the Humanities*, vol. 30, no. 2, pp. 167–182, 2015.

[26] J. Patchala and R. Bhatnagar, "Authorship attribution by consensus among multiple features," in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, pp. 2766–2777.

[27] T. Boran, M. Martinaj, and M. S. Hossain, "Authorship identification on limited samplings," *Computers & Security*, vol. 97, p. 101943, 2020.

[28] L. Cruz, "Authorship recognition with short-text using graph-based techniques," in *WiNLP*, 2019, pp. 153–156.

[29] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, "graph2vec: Learning distributed representations of graphs," *arXiv preprint arXiv:1707.05005*, 2017.

[30] A. K. Dipongkor, M. A. Nashiry, K. A. , Abdullah, and R. S. Ritu, "A study on bengali stemming and parts of speech tagging," in *IEMIS*, 2020, pp. 1–6.

[31] F. Ullah, J. Wang, S. Jabbar, F. Al-Turjman, and M. Alazab, "Source code authorship attribution using hybrid approach of program dependence graph and deep learning model," *IEEE Access*, vol. 7, pp. 141 987–141 999, 2019.

[32] P. Shrestha, S. Sierra, F. A. González, M. Montes, P. Rosso, and T. Solorio, "Convolutional neural networks for authorship attribution of short texts," in *EACL*, 2017, pp. 669–674.

[33] S. Ruder, P. Ghaffari, and J. G. Breslin, "Character-level and multi-channel convolutional neural networks for large-scale authorship attribution," *arXiv preprint arXiv:1609.06686*, 2016.

[34] H. A. Chowdhury, M. A. H. Imon, and M. S. Islam, "A comparative analysis of word embedding representations in authorship attribution of bengali literature," in *ICCIT*. IEEE, 2018, pp. 1–6.

[35] A. Khatun, A. Rahman, H. A. Chowdhury, and M. S. Islam, "Authorship Attribution Dataset in Bangla 2," Mendeley Data, V5, doi: 10.17632/w9wkd7g43f.5.

[36] A. Khatun, A. Rahman, and M. S. Islam, "Authorship Attribution Dataset in Bangla," Mendeley Data, V2, doi: 10.17632/6d9jrkgtvv.2.

[37] M. Eder, "Style-markers in authorship attribution: a cross-language study of the authorial fingerprint," *SPL*, vol. 6, no. 1, 2011.