# PAA: Persian Author Attribution Using Dense and Recursive Connection

Maryam Najafi * and Saeide Sadidpur

Posted Date: 20 May 2024

doi: 10.20944/preprints202405.1258.v1

Keywords: author attribution;Word2Vec; skip connections; attention mechanism

 doi:10.20944/preprints202405.1258.v1



*Article*

# PAA: Persian Author Attribution Using Dense and Recursive Connection

**Maryam Najafi and Saeide Sadidpur**

Department of Mathematics and Computer Science, Malek-Ashtar University of Technology , Tehran, Iran; najafi@mut.ac.ir

* Correspondence: sadidpur@mut.ac.ir

**Abstract:** Author attribution refers to identifying the author of a document by distinguishing between texts written by various authors. Since the author's writing style is a straightforward concept of his writing habits, an author's attribution can be used in various sections of society, including legal work, court cases, and plagiarism. This paper studies and compares a deep architecture that automatically identifies Persian writers through their writing styles. For this purpose, since there were no existing datasets suitable for author identification in Persian, the first step was to collect three different datasets: Novels, News, and Twitter. Afterward, the Word2Vec algorithm is trained on many Persian datasets and used as word embedding. To have syntactic features besides semantic ones, Part Of Speech (POS) embedding has also been used. In the next step, an attention mechanism and Long Short-Term Memory (LSTM) are coupled with residual connections. Thus, the current layer input becomes the output of all previous layers rather than the top one. The model implemented is superior in performance for all three types of data compared to earlier models in text classification and author attribution. Absolute accuracies of 90.22%, 85.10%, and 71.39% were achieved among the three Novels, News, and Tweets datasets.

**Keywords:** author attribution; Word2Vec; skip connections; attention mechanism

Author attribution refers to identifying the author of a document by distinguishing between texts written by various authors. Since the author's writing style is a straightforward concept of his writing habits, an author's attribution can be used in various sections of society, including legal work, court cases, and plagiarism. This paper studies and compares a deep architecture that automatically identifies Persian writers through their writing styles. For this purpose, since there were no existing datasets suitable for author identification in Persian, the first step was to collect three different datasets: Novels, News, and Twitter. Afterward, the Word2Vec algorithm is trained on many Persian datasets and used as word embedding. To have syntactic features besides semantic ones, POS embedding has also been used. In the next step, an attention mechanism and LSTM are coupled with residual connections. Thus, the current layer input becomes the output of all previous layers rather than the top one. The model implemented is superior in performance for all three types of data compared to earlier models in text classification and author attribution. Absolute accuracies of 90.22%, 85.10%, and 71.39% were achieved among the three Novels, News, and Tweets datasets.

## 1. Introduction

Authorship attribution (AA) is the task of identifying the author of a given text. In addition to its many applications to identify the author of anonymous texts, it has also been used in other investigations such as phishing emails and plagiarism detection. Personality traits such as the author's text, genre, temperament, sentiment, native language, and gender can be determined by stylistic features. In other words, the goal is to classify the authors according to their writing style. These were then categorized based on machine learning algorithms for each author. In [1], the AA task is done using a character-based N-gram with a machine learning algorithm. Stylometry is defined as a lexical, syntactic, or structural feature. A document's linguistic features represent the author's vocabulary and word choice, while syntactic features show how the sentences are constructed. Stylometry is the

 doi:10.20944/preprints202405.1258.v1



first field that has gone through some computational work. In most studies of AA, great emphasis is placed on feature engineering and extracting the writing style features intrinsic to the author[2–4]. A wide variety of techniques have been introduced to solve the AA problem. The Mosteller method, which applies 30 functions to words, is one of the essential components of AA research [5]. Using the frequency of single words, such as prepositions, Mosteller and Wallace examined the authorship detection in Federalist Articles [5]. However, this research does not end with manual calculations of textual features. The present paper will take advantage of deep architectures in artificial intelligence to learn input tokens features automatically. It is possible to identify the author in two ways. The first method is close-AA, which is applied when a candidate author exists for the text. In other words, if the train and test authors are the same, this method will be used. A second method, known as open-AA, involves a mechanism in which the actual author is not even among the candidates. In other words, the train and test authors are not necessarily from the same source. Most stylistic features are consistent across different documents written by the same author. Moreover, modern algorithms like neural networks require labeled data. Therefore, most Natural Language Processing (NLP) tasks are primarily focused on data gathering. This study includes the collected corpus in Persian since the existing corpus does not meet the needs of researchers in this area. This work is seminal in the nascent field of Persian AA and addresses the limitations of data, embedding, and model, and has broad applicability in identifying authors' writing styles. It is worth mentioning that there are many ways in which the proposed system may be improved or extended. The contributions of this paper are summarized as follows. Section 2 introduces previous attempts in the dataset, Author Attribution, and text classification. The dataset is presented in detail in Section 3. Section 4 offers deep learning-based methods implemented and run on this corpus. Section contains the models of experimental information. In Section 6, the model results and discussions are reported. Finally, some conclusions and possible paths for future research are provided in Section 7.

## 2. Related Works

Text classification and author identification are among the emerging concepts in NLP, but fortunately, it comes with practical results. Many research studies were conducted to identify the writing style of a text as deep learning. Due to the similarity in the nature of this task, it can be claimed that AA is considered a kind of text classification method. Thus, both types of text classification research and AA research were investigated.

### *2.1. Dataset*

PAN at CLEF[1] is a competition that evaluates and advances approaches to computational text analysis tasks. One of the main tasks is author attribution, where algorithms are developed to identify the author of a given document based on various features. The competition has been running since 2006 and has stimulated the development of various techniques such as stylometry and machine learning. Each year, the competition releases a dataset of anonymized texts written by multiple authors, and participants are tasked with developing algorithms that can accurately attribute the texts to their respective authors. The evaluation is based on various metrics, including precision, recall, and F1-score, and the best-performing algorithms are recognized and awarded. Overall, Pan at CLEF has played an important role in advancing the field of author attribution and computational text analysis. In the study presented by [6], a dataset of 3000 passages from three Bengali authors was introduced for authorship attribution (AA) statistics. Shanta Phani proposed a system for author classification, which included a feature selection process and a feature ranking mechanism, along with a learning curve to assess the accuracy correlation between train and test data. This article was one of the first to

[1] pan.webis.de,pan.webis.de/data.html

 doi:10.20944/preprints202405.1258.v1



be published in the field of Bengali language AA. Given the limited number of Bengali AA datasets available, the study attempted to collect more data to facilitate future research in this area.

### 2.2. Author Attribution

Most AA tasks are performed using SVM [7–10] and random forests[11,12]. As a feature, vocabulary richness, sample length, stop words, and symbols like punctuation and syntactic information have been used [10,13,14]. For example, based on the Bengali dataset, Tasmim and Ismail analyzed four authors for features such as word repetition frequency, word length, and the number of conjunctions and suffixes [15]. Using n-grams at various levels is one of the most popular ways to analyze the author's writing style [1,3,4,16] In [6], There is another method developed by Phani which uses a multi-layer perceptron for AA tasks in the Bengali dataset. An experimental study by Jafari et al. in Feb2019 [17] looked at the long-term and short-term effects of POS lexical dependencies. Experimental results from this paper in the PAN 2012 indicate that syntactically recurrent neural networks (RNN) achieve approximately 14% better accuracy than lexical models with the same architecture . In [17], embedded POS encoder (CNN or LSTM) learns how to synthesize sentences from the embedded POS layer output, represented by a sequence of POS tags. The sentence encoder learns how to express a document syntactically from the sequence of sentences displayed by the POS encoder. Attention is used to focus on the more critical words in predicting [17]. In a paper published a few months later, Jafari et al. presented a new style-aware neural network for encoding document information from three stylistic levels [18]. As a first step, they propose a joint encoding of syntactic and lexical information about sentences. Consequently, attention-based hierarchical neural networks were implemented to encode syntactic and semantic structures in documents [18]. The transformer-based models [19–22] are also discussed in this paper. Recent studies have shown that models designed using this network are successful in many aspects of natural language processing. [22] has improved the accuracy of sentiment analysis, emotion recognition, news classification and AA by 5-29 percent In 2020, Shaheen et al.[23], like Alam et al. [22], who investigated text classification, used the transformer architecture introduced in [24] in many recent models.

### 2.3. Text Classification

Finding the connection between the context of words in a sentence is the main challenge for classification and one of the main problems with RNNs is their tendency to vanish/explode gradients. [25,26] were published on classification based on Attention and LSTM to improve the performance when dealing with long sentences. In 2018, A new multi-layer RNN model was presented by Zixiang Ding et al.[27] Each layer is a combination of its hidden state and all the hidden states of the previous layers. The article's main idea is similar to that of DenseNet in 2016 [28]. Algorithms like CNN and RNN heavily influence text classification. CNN, however, considers only local information between consecutive words and ignores content-based information between long-distance words. A model in [29] integrates Bi-LSTM with an attention mechanism and multi-channel convolution layer to improve text representation. In the self-attention layer, each hidden vector is assigned a weight by computing the dot product attention. Using different convolution kernels, higher-level document features are obtained at the convolutional layer [29]. Gang et al. presented a study based on a Bi-LSTM, CNN, and attention mechanism, whose main application is text classification [30]. ). In this paper, convolutional layers extract higher-level phrase representations from the word embedding vectors. At the same time, Bi-LSTM is used to access the preceding and following context representations. As [25,26], there is an attention mechanism to give different emphasis to the information outputted by Bi-LSTM. Alexis Conneau et al. presented a new architecture in 2017 [31] based on character-level convolution and pooling. With a convolutional layer depth of 29 layers, the performance of this model increases significantly. [32] proposes a recurrent convolutional neural network for capturing semantic information in text classification. in [33], Nasiri et al. proposed the attention-based CNN-RNN sentiment analysis model.

 doi:10.20944/preprints202405.1258.v1



## 3. Dataset

AA is still a nascent field in natural language processing. Hence, the main concern in this field is to have appropriate data. It was decided to gather data so that it is possible to identify the author at this point. There is no specific data type to which AA should be applied since it is a general issue. The collected data has been divided into three major categories:

- **Novels** to handle the lack of long and informal data
- **News** to handle the lack of long and formal data
- **Tweets** to handle the lack of short and formal, and informal data

Each type of data will be discussed in the following sections.

### *3.1. Novel Dataset*

A collection of 35 Persian authors whose novels are famous was investigated. Since having information about how many sentences were in the corpus was the main criterion, the number of samples for each author was calculated. High variance in the number of samples from each author causes the results to flow in favor of the labels with more information. To achieve balanced data, the novel database was derived from 30 authors with adequate samples. It should be noted that since the data with a single sentence structure does not represent the stylometry of the author, in this paper, the samples have chunk size M instead of a single sentence. There are phrases between 175 and 250 characters in each sample for each author. It should be acknowledged that to prevent information loss, data imbalances remain yet. There are several methods to address this issue. Class weighting techniques will be discussed in Section 5.3, which helps reduce the impact of data imbalance on a model result by allowing the model to give more weight to minority group examples than majority group examples. Finally, 12% of the total data was allocated for test data, while 15% of the remaining 85% was assigned to evaluation data. The rest of the data was considered as train data in Table 1.

**Table 1.** The number of Novel data

| Data Type | Number of Samples |
|---|---|
| Train | 19473 |
| Evaluation | 3125 |
| Test | 3437 |

### *3.2. News Dataset*

News data were collected to associate the News with its author on social media. News data has been automatically crawled from News on the online news site. There were 234 authors in total. Fifty authors with the most data were selected to analyze subsequent samples due to the unequal data distribution discussed in the previous section. As mentioned earlier, given that the data with a single sentence structure cannot capture the author's writing style features, samples with chunk size M are selected instead of a single sentence. The dataset is published with M=5,15 for short and long author detection tasks. Table 2 indicates the number of samples used for train, evaluation, and test in M= 5, 15. Table 2 indicates the number of samples used for train, evaluation and test in M= 5, 15.

**Table 2.** The number of News data

| Data Type . | Number of Samples $M = 5$ | $M = 15$ |
|---|---|---|
| Train | 66702 | 22242 |
| Evaluation | 10702 | 3569 |
| Test | 11772 | 3926 |

 doi:10.20944/preprints202405.1258.v1



### 3.3. Twitter Dataset

Tweet datasets from Twitter have been generated by crawling data for a specific community, including followers of an influential person. The crawling result collected about 2,000 users, of which 50 users with the most significant number of posts were selected. The Twint library performs the crawling action. Each tweet was considered to be one sample in the database. There were 303,732 tweets in the dataset. Table 3 specifies the number of samples divided into the train, evaluation, and test samples.

**Table 3.** The number of Twitter data

| Data Type | Number of Samples |
|---|---|
| Train | 231727 |
| Evaluation | 34288 |
| Test | 37717 |

The next step was to prepare a database containing tweets of famous peoples such as actors, athletes, journalists, and politicians to have relevant and attractive information on social media. This corpus is consists of 149297 tweet samples from 30 famous people.Number of samples for train, evaluation and test data is specified in Table 4 .

**Table 4.** The number of Famous Twitter data

| Data Type | Number of Samples |
|---|---|
| Train | 107866 |
| Evaluation | 19036 |
| Test | 22395 |

## 4. Model

A deep residual base neural network is presented for encoding the patterns of writing style in a document. As a first step, each sentence is represented by a sequence of words and their POS tags. Each word is encoded into a vector of 300 dimensions. Then the POS tag is embedded into a low dimensional vector. There is also a concatenation of word embeddings and POS embeddings in the embedding section. These syntactic and semantic representations are then aggregated into three successive layers of LSTMs. In this part, sentences contributing more to label prediction get rewards with an attention mechanism. The probability distribution over class labels is then computed using a softmax classifier. Figure 1 shows the overall network architecture. The main components of the model are presented in the subsequent sections.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2024 doi:10.20944/preprints202405.1258.v1



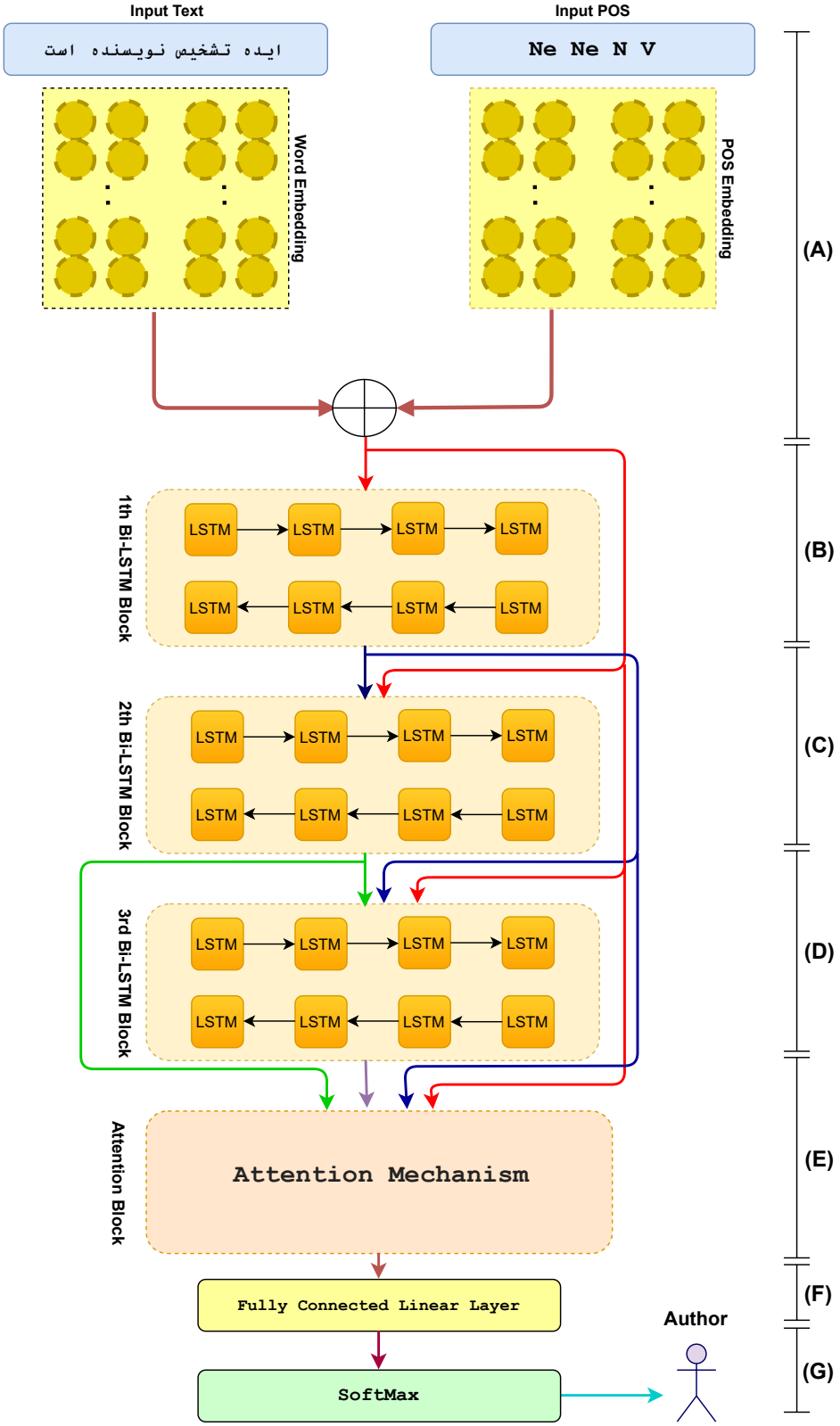


**Figure 1.** Overall architecture of the model

### *4.1. Embedding Layer*

The input elements of the network are typically a sequence of words representation in the k dimension. However, in the proposed model, the sequence of syntactic information is also fed

 doi:10.20944/preprints202405.1258.v1



to the network through POS. These syntactic patterns were obtained with POS from the Hazm library's tagger [2].Therefore, the input word embedding section includes both 300-dimensional and low-dimensional POSembedding sections, mapping each word to its corresponding vector space. An embedding of this POS in a low dimension vector is created through a lookup table. The concatenation ofword and POS embeddings demonstrate at section (A) of Figure 1 and Formulate as (8). consider that the sequence of word and sequence of POS as $\{emb(w_1), emb(w_2), \ldots, emb(w_s)\}$ and $\{ emb(P_1), emb(P_2), \ldots, emb(P_s)\}$ respectively, each input term is represented by dense matrix dimensions of 300 for word and 30 for POS embedding. Since grammatical writing styles, such as how verbs, nouns, adjectives, and adverbs are specific to each author, adding syntactic information by POS was suggested. Experiments showed that the model's performance improved by about 2% with both syntactic and semantic information. A sequence of the word and POS vectors are sent to the dense Bi-LSTM module as input.

### *4.2. Bi-LSTM layer*

The reason for using recurrent networks is that they are highly effective at capturing conceptual dependencies. LSTM encodes the sentence in several time-steps and takes the implicit vector of each step. Bi-LSTM is used to extract content for both the past and the future. Bi-LSTM unit can be demonstrated as below:

$$\boldsymbol{h_t = LSTM(h_{t-1}, e(c_t))} \tag{1}$$

hidden state compute by LSTM is h = $\{h_1, h_2, \ldots, h_s\}$. consider that $[emb(word_t); emb(POS_t)]$ as $e(c_t)$ and $h_{t-1}$ as LSTM's hidden state at time-step t-1 So forward and backward direction in Bi-LSTM is Shown as below:

$$\overrightarrow{\boldsymbol{h_t}} = \overrightarrow{\boldsymbol{LSTM}}\left(\overrightarrow{\boldsymbol{h_{t-1}}}, \boldsymbol{e(c_t)}\right) \tag{2}$$

$$\overleftarrow{\boldsymbol{h_t}} = \overleftarrow{\boldsymbol{LSTM}}\left(\overleftarrow{\boldsymbol{h_{t-1}}}, \boldsymbol{e(c_t)}\right) \tag{3}$$

then the concatenation of forward and backward hidden states is taken as the representation of each word

$$\boldsymbol{h_T} = [\overrightarrow{\boldsymbol{h_t}}; \overleftarrow{\boldsymbol{h_t}}] \tag{4}$$

### *4.3. Residual Connection*

These architectures are not without their downsides; however, due to the increase or decrease of gradient values when backpropagating a loss to the previous steps, the gradient disappears, which causes a loss of information when network dependencies are extended. LSTM can take variable lengths as inputs; however, the sentence information is encoded as a hidden vector with a fixed size, which compresses the long sentences data. As a result, the output vector in the last time-step cannot accurately convey the context of long samples. This paper aims to use the same DenseNet architecture and conduct research based on it. These dense connections were used to preserve information by concatenating all previous outputs with the current layer. As shown in sections (A, B, C) of Figure 1, in contrast to classical methods, where each LSTM block gets input from its predecessor, each block gets input from all its predecessors. The formula for these three successive layers is shown as formula (9), (10) and (11). One way to increase LSTM performance is to add a max-pooling layer to its output. The max-pooling layer can only return the maximum value but still cause significant loss. So, if each time-step's output can be used more effectively, better results can be acquired. The next section will present a self-attention mechanism.

---

[2] https://github.com/sobhe/hazm

 doi:10.20944/preprints202405.1258.v1



### 4.4. Attention Mechanism

A key reason for using the attention mechanism is that attention is drawn mostly to the sentences that capture the style of writing most effectively. Attention have shown great success in many tasks, including question answering, machine translations, speech recognition, and image captioning. The mechanism is thought to pay attention to the tokens that help predict the labels. This module has also been updated by the output of the previous three LSTM blocks and one embedding block. So, self-attention is used to encode each hidden layer representation obtained through the Bi-LSTM layer. The result is a representation of a high-level feature. Assume H is a matrix containing all previous layer outputs as $h_{1,}h_2, h_3, \ldots, h_S$Where S is the length of the sentence S, and r is the representation of the sentence using the weighted sum of these output vectors. This section uses the batch matrix multiplied by the output from the previous layers to calculate similarity; the formula 5 explains that.

$$\mathbf{tanh}(H\boldsymbol{\alpha}^{L}) \tag{5}$$

where $\boldsymbol{\alpha} = \boldsymbol{softmax}(\boldsymbol{ww}^{L})$

### 4.5. Softmax

Finally, to determine the probability distribution of the labels, the softmax classification method is used. The module is a simple soft-max classifier that generates probabilities of distributions based on input features. A softmax classifier is used to predict a label $\hat{y}$ from a set of discrete classes for a sentence S. The classifier takes R as input:

$$\boldsymbol{P}(\boldsymbol{y} \mid \boldsymbol{S}) = \boldsymbol{softmax}(\boldsymbol{W}^{(S)}\boldsymbol{R} + \boldsymbol{b}^{(S)}) \tag{6}$$

$$\widehat{\boldsymbol{y}} = \boldsymbol{argmax}\boldsymbol{P}(\boldsymbol{y} \mid \boldsymbol{S}) \tag{7}$$

## 5. Experimental Setup

### 5.1. Data Pre-processing

An initial arrangement was made for pre-processing text based on the research carried out previously. There are usually incomplete phrases, miscellaneous phrases, and inappropriate sentences in texts. This is because acronyms, irregular grammar, inappropriate words, and non-verbal terms are frequently used. Unstructured textual data containing such noise can affect the deep learning model's performance. To reduce noise in the text, pre-processing is necessary before selecting a feature. The pre-processing steps are as follows:

- Remove non-ASCII and non-Persian characters
- Remove punctuation, including ".", ",", "!", "?" , etc.
- Remove All URL links.
- Remove the URL that does not contain any specific textual information.
- Remove numbers. These numbers usually do not contain specific textual information, so they are removed from text to optimize content.
- Remove Stop-words. Stop-words usually refer to the most common words in the language that do not contain semantic information

During text pre-processing, the above steps were taken in the present paper. A model was trained using normalized data, and the obtained accuracy shows that the score is about 20% lower than when only the Hazm normalizer was used as a pre-processing step instead of the above steps. In the pre-processing result, it was concluded that features like punctuation, number, stop-word, and how they were used somehow indicate each author's particular writing style. Therefore, removing them in pre-processing removed each author's stylometry. A more specific method of assessing an author's writing style is to observe the number of punctuation marks, and whether or not they are used, the

 doi:10.20944/preprints202405.1258.v1



type of stop words, and how many of them are used. Therefore, the only pre-processing done on the data in this section is the Hazm normalizer.

### *5.2. Embedding*

Word embeddings have constituted an active area of research for the past few decades. The attempt in this task is to obtain suitable vector representations for words. There are many datasets and embeddings in high-resource languages like English. Conversely, in low resource languages like Persian, a shortage of such linguistic resources affects the performance of tasks. Embeddings play an essential role in obtaining vector representations for words in low-resource languages. As compensation for this problem, the crawl of news sites provides a lot of raw data, including 350 million sentences from 5 million documents, of which 486775 have a unique word in them. Word embeddings of proposed data were provided to help Persian researchers. Data has been collected from Tabnak, Fars, IRNA, Mehr, ISNA, Tasnim, and online news websites. This article uses these data to train skip-gram and CBOW-based word embeddings.

### *5.3. Hyperparameter Tuning*

Word embedding was obtained by training an unsupervised word2vec model [34] on the 486775 unique words from 350 million sentences. The word embedding dimension is set to 300. To build an informative vocabulary, words that appear more than ten times were retained and replaced others with ”UNK” tokens. LSTM layers were configured in sequential mode, with each layer containing 128 memory units. In attention blocks, multi-head attention with 20 heads was used. The training was eventually conducted over ten epochs with a batch size of 16. As for regularization, dropout is applied to word embeddings and the output of LSTMs, and analysis shows that the model performs best with dropout rates of 0.5, 0.3, 0.3, 0.3, respectively. Cross-Entropy was used as the loss function and Adam as the optimizer for the network. Backward propagation was used for training by a learning rate of 0.005. Class weight was used with loss functions to help the model adapt to the imbalanced data. Class weight allows the model to pay more attention to examples from the minority class than the majority class in datasets with a severely skewed class distribution. Finally, experiments were conducted to obtain the most appropriate hyperparameter, including the number of hidden states, the number of Bi-LSTM layers, and the number of attention heads. The best of which can be seen in the Table below. Other values are not shown due to space limitations. Finally, it was found that increasing the number of neurons to only a certain extent seemed to help improve results.

 doi:10.20944/preprints202405.1258.v1



**Table 5.** Accuracy score in parameter tuning(%)

| Number of LSTM layers | | | |
|---|---|---|---|
| | **Novels** | **News** | **twitter** |
| **1** | 86.29 | 80.37 | 65.84 |
| **2** | 88.03 | 82.31 | 70.62 |
| **3** | **90.22** | **85.10** | **71.39** |
| **4** | 88.71 | 83.39 | 69.30 |
| **5** | 88.63 | 83.18 | 68.65 |
| **Number of Hidden units** | | | |
| | **Novels** | **News** | **twitter** |
| **64** | 88.91 | 83.76 | 70.08 |
| **128** | **90.22** | **85.10** | **71.39** |
| **256** | 89.54 | 84.47 | 70.57 |
| **Number of Attention heads** | | | |
| | **Novels** | **News** | **twitter** |
| **10** | 89.79 | 85.02 | 71.03 |
| **20** | **90.22** | **85.10** | **71.39** |
| **30** | 89.06 | 84.19 | 70.90 |

## 6. Results

The first step was to apply recent models in text classification and author identification to the presented data. The next step was to implement the introduced model on the same data and compare the results. Two reasons have been given for the attempted implementation of text classification models based on AA tasks in this paper. Firstly, an AA task can be viewed as a text classification task since the author is classified according to his writing style. Secondly, there has been very little research done on identifying authors using deep learning. There are two different types of evaluations presented to evaluate the quality of the proposed model. Evaluations are carried out: accuracy and f-score. The evaluation of the test data shows that the proposed model has achieved accuracy and an F-score of 90.22%, 85.10%, and 71.39% in Novels, News, and Tweets, respectively. It can be started with the simplest models on Novel data. The accuracy of 76.94 for LSTM and 85.92 for CNN was deemed acceptable for the first attempt, as shown in Table 6. Several other studies were implemented. In Row three, Both LSTM and multi-channel CNN architectures are used followed by self-attention and max-pooling layer. A better result was expected because of the complexity of the architecture and the logical layer arrangement. It was, however, weaker than both the previous approaches, as shown in the Table. In row 4, CNN forms a very deep architecture(29 layers) based on VGG and Resnet's deep structure. This study produced inferior results to the previous three studies. The following steps are to implement the models using transformers. Many natural language processing tasks have been successfully conducted using networks of this type and their variants. The encoder of transformers has been used exclusively in this section. Table 6, row 5, shows the results of using the transformer. There were some acceptable but not satisfactory results. Therefore, research has shifted to architectures that focus on self-attention mechanisms. As such, the research results were examined based on attention in rows 7, 9, and 10 to evaluate the effect of self-attention implementation on AA and found that the self-attention mechanism significantly improved results. However, one of the best results in the proposed data is the R-CNN architecture, which encodes and outputs words by using Bi-LSTMs and the max-pooling layer. Next, the DC-Bi-LSTM architecture has been implemented, one of the inspirations depicted in this paper. The architecture used in this paper is based on residual connections.

$$\boldsymbol{Layer}_{(1)}: \begin{cases} \boldsymbol{Input}: \{w_1, w_2, \ldots, w_s\}, \{\ POS_1,\ POS_2, \ldots,\ POS_s)\} \\ \boldsymbol{ouput}:\ [\boldsymbol{emb}\,(\boldsymbol{word}_t)\,;\boldsymbol{emb}(\boldsymbol{POS}_t)] \end{cases} \tag{8}$$

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2024 doi:10.20944/preprints202405.1258.v1



$$Layer_{(2)}: \begin{cases} Input: \{e(c_1), e(c_2), \ldots, e(c_s)\} \\ ouput: \left\{h_1^1, h_2^1, \ldots, h_s^1\right\},\ h_t^1 = [\overrightarrow{h_t^1}; \overleftarrow{h_t^1}] \end{cases} \tag{9}$$

$$Layer_{(3)}: \begin{cases} Input: \{[e(c_1); h_1^1], [e(c_2); h_2^1], \ldots, [e(c_s); h_s^1]\} \\ ouput: \left\{h_1^2, h_2^2, \ldots, h_s^2\right\},\ h_t^2 = [\overrightarrow{h_t^2}; \overleftarrow{h_t^2}] \end{cases} \tag{10}$$

$$Layer_{(4)}: \begin{cases} Input: \{[e(c_1); h_1^1; h_1^2], [e(c_2); h_2^1; h_2^2], \ldots, [e(c_s); h_s^1; h_s^2]\} \\ ouput: \left\{h_1^3, h_2^3, \ldots, h_s^3\right\},\ h_t^3 = [\overrightarrow{h_t^3}; \overleftarrow{h_t^3}] \end{cases} \tag{11}$$

Figure 1 shows the overall architecture of the model. With this technique, there is almost no data loss between layers. Rows 6 and 8 in this Table display the results of these two architectures, which achieved the highest performance without using complex architectures. Hence, the proposed architecture in row 11 combines the Residual idea and an attention mechanism. This idea combines word and POS embeddings and three skip connections in the LSTM. The attention mechanism is finally being applied to the output of these layers. There is more information on the number of hyperparameters in Section 5.3. The results obtained on three datasets are shown in Table 6 Figure 2.

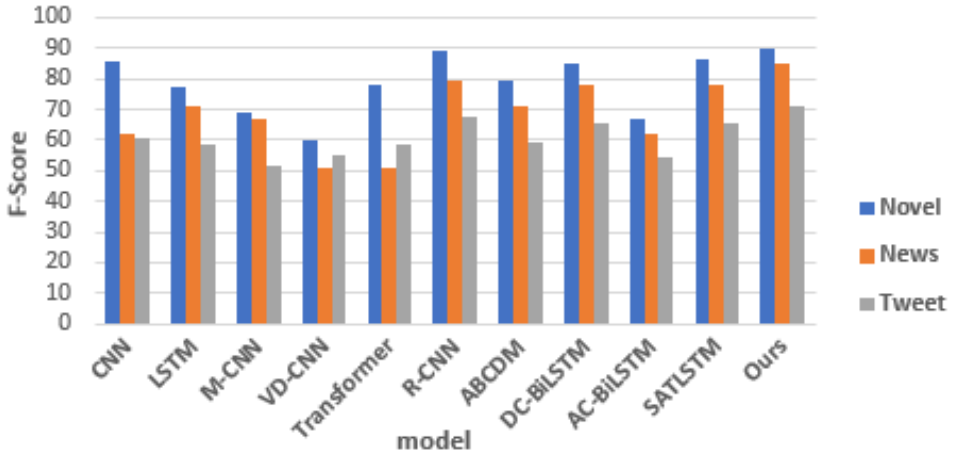


**Figure 2.** Results obtained from different models on proposed datasets

**Table 6.** Comparison of the results obtained from diffrent models on proposed datasets(%)

| ID | Models | Novel Data | | News Data | | Twitter Data | |
|---|---|---|---|---|---|---|---|
| | | Accuracy | F-score | Accuracy | F-score | Accuracy | F-score |
| 1 | **CNN** [35] | 85.92 | 85.92 | 63.34 | 62.15 | 60.70 | 60.50 |
| 2 | **LSTM**[36] | 76.94 | 77.39 | 70.41 | 71.13 | 57.85 | 58.29 |
| 3 | **M-CNN**[29] | 67.17 | 68.93 | 66.50 | 66.69 | 51.40 | 51.40 |
| 4 | **VD-CNN** [31] | 59.18 | 60.02 | 48.32 | 50.65 | 55.00 | 54.99 |
| 5 | **Transformer**[20] | 78.14 | 78.20 | 50.79 | 51.15 | 58.00 | 58.59 |
| 6 | **R-CNN**[32] | 89.14 | 89.14 | 79.77 | 79.76 | 67.22 | 67.82 |
| 7 | **ABCDM** [33] | 79.68 | 79.43 | 69.96 | 70.76 | 60.00 | 59.25 |
| 8 | **DC-Bi-LSTM** [27] | 85.15 | 85.06 | 78.01 | 78.21 | 65.78 | 65.74 |
| 9 | **AC-Bi-LSTM** [30] | 66.95 | 66.73 | 60.92 | 62.08 | 54.15 | 54.42 |
| 10 | **SATLSTM** [25] | 86.65 | 86.62 | 77.57 | 77.74 | 65.59 | 65.27 |
| **11** | **Ours** | **90.22** | **90.22** | **85.10** | **85.04** | **71.39** | **71.18** |

## 7. Conclusion and Future Work

There is likely not enough data for various tasks in languages with low resources. Hence, in this study, the first step was to collect the data. A total of three sources, Novels, News, and Twitter dataset, were presented following the data collection and analysis phase; a novel multi-layer RNN model with

 doi:10.20944/preprints202405.1258.v1



an attention mechanism that follows the dense connection method was presented. A network with dense connections is more unlikely to have disappearing gradients or over-fitting problems since the networks can be trained deeper than dozens of levels. The proposed model was evaluated on three presented datasets; experiments demonstrate that this model outperforms traditional Bi-LSTM and gets promising performance compared to the state-of-the-art approaches. As future work:

- **Data preparation**: In light of the numerous news resources and social media postings associated with each individual, it is possible to increase the number of authors and, consequently, the size of the data for each author.
- **Data processing**: As mentioned previously, since typical pre-processing such as removing stop words or punctuation reduces accuracy, we use Hazm normalizer. There needs to be more research in this area so that the best way to normalize the data can be accurately applied.
- **Embedding**: In future work, recent conceptual embeddings, like the encoder part of BERT and MT5, can be used.
- **Model architecture**: This architecture can be defined in other tasks such as NLI, Question Answering, Machine Translation, etc.

## References


1. Stamatatos, E. On the robustness of authorship attribution based on character n-gram features. *Journal of Law and Policy* **2013**, *21*, 421–439.
2. Potthast, M.; Braun, S.; Buz, T.; Duffhauss, F.; Friedrich, F.; Gülzow, J.; Köhler, J.; Loetzsch, W.; Müller, F.; Müller, M.; Paßmann, R.; Reinke, B.; Rettenmeier, L.; Rometsch, T.; Sommer, T.; Träger, M.; Wilhelm, S.; Stein, B.; Stamatatos, E.; Hagen, M. Who Wrote the Web? Revisiting Influential Author Identification Research Applicable to Information Retrieval. 2016, Vol. 9626, pp. 393–407. doi:10.1007/978-3-319-30671-1_29.
3. Sidorov, G.; Velasquez, F.; Stamatatos, E.; Gelbukh, A.; Chanona-Hernández, L. Syntactic N-grams as machine learning features for natural language processing. *Expert Syst. Appl.* **2014**, *41*, 853–860.
4. Gomez Adorno, H.; Sidorov, G.; Pinto, D.; Vilariño, D.; Gelbukh, A. Automatic Authorship Detection Using Textual Patterns Extracted from Integrated Syntactic Graphs. *Sensors* **2016**, *16*, 1374. doi:10.3390/s16091374.
5. Mosteller, F.; Wallace, D.L. Inference in an authorship problem: A comparative study of discrimination methods applied to the authorship of the disputed Federalist Papers. *Journal of the American Statistical Association* **1963**, *58*, 275–309.
6. Phani, S.; Lahiri, S.; Biswas, A. Authorship Attribution in Bengali Language. Proceedings of the 12th International Conference on Natural Language Processing; NLP Association of India: Trivandrum, India, 2015; pp. 100–105.
7. Diederich, J.; Kindermann, J.; Leopold, E.; Paass, G. Authorship attribution with support vector machines. *Applied intelligence* **2003**, *19*, 109–123.
8. Zheng, Q.; Tian, X.; Yang, M.; Wang, H. The email author identification system based on support vector machine (SVM) and analytic hierarchy process (AHP). *IAENG International journal of computer Science* **2019**, *46*, 178–191.
9. Ouamour, S.; Sayoud, H. Authorship attribution of ancient texts written by ten arabic travelers using a smo-svm classifier. 2012 International Conference on Communications and Information Technology (ICCIT). IEEE, 2012, pp. 44–47.
10. Martín-del Campo-Rodríguez, C.; Alvarez, D.A.P.; Sifuentes, C.E.M.; Sidorov, G.; Batyrshin, I.; Gelbukh, A. Authorship Attribution through Punctuation n-grams and Averaged Combination of SVM **2019**.
11. Khonji, M.; Iraqi, Y.; Jones, A. An evaluation of authorship attribution using random forests. 2015 International Conference on Information and Communication Technology Research (ICTRC). IEEE, 2015, pp. 68–71.
12. Maitra, P.; Ghosh, S.; Das, D. Authorship Verification-An Approach based on Random Forest. *arXiv preprint arXiv:1607.08885* **2016**.
13. Grieve, J. Quantitative authorship attribution: An evaluation of techniques. *Literary and linguistic computing* **2007**, *22*, 251–270.
14. Jin, M.; Jiang, M. Text clustering on authorship attribution based on the features of punctuations usage. 2012 IEEE 11th International Conference on Signal Processing. IEEE, 2012, Vol. 3, pp. 2175–2178.

 doi:10.20944/preprints202405.1258.v1




15. Das, P.; Tasmim, R.; Ismail, S. An experimental study of stylometry in bangla literature. 2015 2nd International Conference on Electrical Information and Communication Technologies (EICT). IEEE, 2015, pp. 575–580.
16. Sari, Y.; Vlachos, A.; Stevenson, M. Continuous n-gram representations for authorship attribution. Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, 2017, pp. 267–273.
17. Jafariakinabad, F.; Tarnpradab, S.; Hua, K.A. Syntactic recurrent neural network for authorship attribution. *arXiv preprint arXiv:1902.09723* **2019**.
18. Jafariakinabad, F.; Hua, K.A. Style-aware neural model with application in authorship attribution. 2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA). IEEE, 2019, pp. 325–328.
19. Shaheen, Z.; Wohlgenannt, G.; Filtz, E. Large scale legal text classification using transformer models. *arXiv preprint arXiv:2010.12871* **2020**.
20. Yüksel, A.E.; Türkmen, Y.A.; Özgür, A.; Altınel, B. Turkish tweet classification with transformer encoder. Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019), 2019, pp. 1380–1387.
21. González, J.Á.; Hurtado, L.F.; Pla, F. ELiRF-UPV at TASS 2019: Transformer Encoders for Twitter Sentiment Analysis in Spanish. IberLEF@ SEPLN, 2019, pp. 571–578.
22. Alam, T.; Khan, A.; Alam, F. Bangla Text Classification using Transformers. *arXiv preprint arXiv:2011.04446* **2020**.
23. Shaheen, Z.; Wohlgenannt, G.; Filtz, E. Large scale legal text classification using transformer models. *arXiv preprint arXiv:2010.12871* **2020**.
24. Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; others. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771* **2019**.
25. Zhou, P.; Shi, W.; Tian, J.; Qi, Z.; Li, B.; Hao, H.; Xu, B. Attention-based bidirectional long short-term memory networks for relation classification. Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers), 2016, pp. 207–212.
26. Jing, R. A self-attention based LSTM network for text classification. Journal of Physics: Conference Series. IOP Publishing, 2019, Vol. 1207, p. 012008.
27. Ding, Z.; Xia, R.; Yu, J.; Li, X.; Yang, J. Densely connected bidirectional lstm with applications to sentence classification. CCF International Conference on Natural Language Processing and Chinese Computing. Springer, 2018, pp. 278–287.
28. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4700–4708.
29. Zhu, X.; Yin, S.; Chen, Z. Attention based BiLSTM-MCNN for sentiment analysis. 2020 IEEE 5th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA). IEEE, 2020, pp. 170–174.
30. Liu, G.; Guo, J. Bidirectional LSTM with attention mechanism and convolutional layer for text classification. *Neurocomputing* **2019**, *337*, 325–338.
31. Conneau, A.; Schwenk, H.; Barrault, L.; Lecun, Y. Very deep convolutional networks for text classification. *arXiv preprint arXiv:1606.01781* **2016**.
32. Lai, S.; Xu, L.; Liu, K.; Zhao, J. Recurrent convolutional neural networks for text classification. Twenty-ninth AAAI conference on artificial intelligence, 2015.
33. Basiri, M.E.; Nemati, S.; Abdar, M.; Cambria, E.; Acharya, U.R. ABCDM: An attention-based bidirectional CNN-RNN deep model for sentiment analysis. *Future Generation Computer Systems* **2021**, *115*, 279–294.
34. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* **2013**.
35. Shrestha, P.; Sierra, S.; González, F.A.; Montes, M.; Rosso, P.; Solorio, T. Convolutional neural networks for authorship attribution of short texts. Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, 2017, pp. 669–674.
36. Wang, J.H.; Liu, T.W.; Luo, X.; Wang, L. An LSTM approach to short text sentiment classification with word embeddings. Proceedings of the 30th conference on computational linguistics and speech processing (ROCLING 2018), 2018, pp. 214–223.

 doi:10.20944/preprints202405.1258.v1