Authorship Attribution using Capsule-based Fusion Approach

Chanchal Suman[1], Rohit Kumar[1], Sriparna Saha[1], and Pushpak Bhattacharyya[2]

[1] Dept. of Computer Science & Engineering, Indian Institute of Technology Patna
{1821cs11,rohit.cs17,sriparna}@iitp.ac.in
[2] Dept. of Computer Science & Engineering, Indian Institute of Technology Bombay
pushpakbh@gmail.com

Abstract. Authorship attribution is an important task, as it identifies the author of a written text from a set of suspect authors. Different methodologies of anonymous writing, have been discovered with the rising usage of social media. Authorship attribution helps to find the writer of a suspect text from a set of suspects. Different social media platforms such as Twitter, Facebook, Instagram, etc. are used regularly by the users for sharing their daily life activities. Finding the writer of micro-texts is considered the toughest task, due to the shorter length of the suspect piece of text. We present a fusion based convolutional Neural Network model, which works in two parts i) feature extraction, and ii) classification. Firstly, three different types of features are extracted from the input tweet samples. Three different deep-learning based techniques, namely capsule, LSTM, and GRU are used to extract different sets of features. These learnt features are combined together to represent the latent features for the authorship attribution task. Finally the softmax is used for predicting the class labels. Heat-maps for different models, illustrate the relevant text fragments for the prediction task. This enhances the explain-ability of the developed system. A standard Twitter dataset is used for evaluating the performance of the developed systems. The experimental evaluation shows that proposed fusion based network is able to outperform previous methods. The source codes are available at `https://github.com/chanchalIITP/AuthorIdentificationFusion`.

Keywords: Authorship Identification · Capsule · Fusion · LSTM · GRU

1 Introduction

Forensic authorship analysis is the process of examining the characteristics of a questioned text in order to draw conclusions on its authorship. Its application involves analyzing long fraud document, terrorist conspiracy texts, short letters, blog posts, emails, SMS, Twitter streams or Facebook status updates to check the authenticity and identify fraudulence. Authorship analysis can be carried out in different ways: 1) authorship attribution, 2) authorship verification, and 3) authorship profiling [7]. In authorship profiling, characteristics (e.g., age, gender, native language, race, personality) of an author are determined after analyzing different texts written by the author [3]. Authorship verification is the task of assessing whether a specific individual writes a suspicious text [7]. In authorship attribution (AA), given the examples of the writings of a number of authors, for an anonymous text, the author is determined. These days, social media play a vital role in our life. People write about their daily life activities via different social media platforms like Twitter, Facebook etc. Most of the data created on these social media applications are micro text. A micro or short text message could be a tweet or a comment which is around 140 characters or less. The authorship analysis of a micro-text is challenging due to the smaller length of text [14].

The traditional strategy of developing the authorship attribution (AA) model, deals with extraction of different features from the text data, and then feeding the generated vector to different available machine learning classifiers (mainly Support Vector Machine) [22,8]. Convolutional neural networks (CNNs) also perform well in this area [5,23,4,10]. The recently proposed n-gram based CNN model with fastText word embeddings has set the state-of-the-art results on the Twitter dataset released by [4]. Pooling layer of CNN reduces the computational complexity of convolution operations. It captures the invariance of local features. But, pooling operations loose information regarding spatial relationships, which causes the mis-classification of objects based on their orientation or proportion. Capsules consider the spatial relationships between entities and learn these relationships via dynamic routing [16]. This has motivated us to use capsule networks for the attribution task. Some recent works have shown the efficacy of a deep learning system using a combination of features learnt from different deep learning models [2]. Following these concepts, in addition to the capsules, we have fused features extracted from different modules for better representation of text features.

Our developed system mainly consists of two parts, i) feature extraction, and ii) classification. The feature extraction part is based on an ensemble technique. Different hidden representations are learnt from three different deep learning models, i.e., Convolutional neural network with capsule, ii) Long Short-Term Memory (LSTM), and iii) Gated Recurrent Unit (GRU). These representations learn the higher-level features from the given text sample. We subsequently fuse these features for representing the latent feature for the AA task. Finally, the fused vector is fed to the softmax layer for the final classification. In this way, the proposed system extracts the authorship information using different features learnt from multiple networks.

In order to show the effectiveness of our developed system, the twitter dataset released by [21] is used for experimentation. An accuracy of 85.35% is achieved for 1000 tweets per author (for 50 authors), using the character unigram representation. The results show that, our developed systems outperform the previous state-of-the-art models. Using heat maps for different models, the working of our developed system is shown. Below, we have listed the contributions of this work.

1. To the best of our knowledge, fusion of different features extracted from capsule, LSTM, and GRU is carried out for the first time for solving the task of authorship attribution.
2. We set new state-of-the-art values, by outperforming the previous ones.
3. A detailed ablation study of different features extracted from the tweet samples has also been performed.
4. The workings of different models are shown using heat-map on the test data, which helps in finding the relevant text-fragments of the suspect text. This gives explain-ability to our developed system.

The paper is organized as follows: Previous works are discussed in Section 2. In Section 3, we discussed our proposed approach. The dataset used for implementation is discussed in Section 4. In Section 5, we have discussed the results achieved using our model. The performance comparison of our developed model with other previous approaches is presented in Section 6. A short analysis of heat-maps generated from different system is presented in Section 7. Finally, we conclude this work in Section 8.

2 Related Work

Traditional methods for dealing with authorship attribution task, involve extraction of features related to content and style. Term-Frequency Inverse-Document Frequency at character or word n-gram level, and Bag-of-Words are mostly used as content features. Whereas usage of punctuation, capital letters, POS tags, digits represents the stylistic features of an author [24]. Different ML classifiers are trained on these features, for the final classification task. Mainly logistics regression, and support vector machines are used as the classifier [1,17]. Some of the recent works are using different deep learning techniques like convolutional neural networks, and siamese networks too [20,19,23,14,13,4]. A comprehensive literature review on the AA task, is presented in [15], focusing on dark sides of AA.

Table 1: Some Recent Works on Twitter Dataset

Approach	Dataset
Character n-gram (n=1,2) on a CNN architecture [23]	Standard Twitter Dataset [21]
combination of character and word n-grams with flexible pattern, and sub-word embedding above MLP [14]	Standard Twitter Dataset [21]
text representation and tag representation (posting style) fed to CNN [12]	standard Twitter Dataset [21]
pre-trained word embedding and character bigram on a multi channel CNN [4]	Standard Twitter Dataset [21]
User representation is learnt from siamese network [13]	Standard Twitter Dataset [21]

Word and character level CNNs are explored for AA in [20]. It is noticed that character level CNNs outperform the traditional simple approaches based on support vector machines and logistic regression. In [19], character n-grams (n=3,4,5) are fed to a multi-layer CNN approach, with max-pooling. Character level n-grams have been used in [23], for authorship attribution in short texts. The results show that CNN based prediction outperforms the techniques based on LSTM and hand-crafted feature extraction. For better visualisation of model learning capacity, saliency score has been used to highlight the text modules responsible for the classification. Authors in [14], utilized character n-grams, flexible patterns, word n-grams, and sub-word embedding on a multi-layer perceptron network to observe its effect on the Twitter dataset [21]. In [4], authors have showed that the combination of embedding layers captures different stylometric features. Authors in [13], have shown that siamese networks are useful in learning the user tweets, with small amount of data only. We have also tabulated some of the important works on the Twitter authorship data in Table 1.

The previous studies show that mainly CNN is working effectively on short texts like tweets, messages etc. This has motivated us to use CNN as our basic framework, on top of which different features extracted from various modules are combined. We have discussed the developed framework below.

3 The Proposed Approach

The Convolution operator in a CNN is represented by the weighted sum of lower layers, thus it is difficult to carry out these features into upper layers in case of complex objects. In this way, it can be said that CNNs do not consider hierarchical relationships [16]. To overcome these shortcomings, pooling layers are introduced. Pooling can reduce the computational complexity of convolution operations and capture the invariance of local features. However, pooling operations loose information regarding spatial relationships and are likely to misclassify objects based on their orientation or proportion. The capsule network is a structured model, which solves the problems of CNNs. To learn the existence of visual entities and encoding them into vectors, there are locally invariant groups which are known as capsules. Capsule networks use a non-linear function called as squashing for grouping of neurons [16].

Long Short Term Memory (LSTM) [11] and Gated Recurrent Unit (GRU) [9] are variants of recurrent neural network (RNN) designed to solve the issue of learning long-term dependencies. For capturing the style of an user, we need information from the past and next part of the writing/future context. Thus, for capturing the contexts from both the past and the future, we have considered bidirectional LSTM and GRU for the purpose of feature extraction.

Our developed system predicts the author of the written sample in two steps, i) feature extraction, and ii) author identification. Below, the complete system is discussed in detail.

3.1 Feature Extraction

The proposed system contains three sister networks for learning three different sets of features, namely capsule network extracted features, LSTM network extracted features, and GRU network extracted features. The three networks are called as sister networks, because the error is back propagated in all of them.

1. Capsule features: The capsule network learns temporal as well as spatial features from the convolutional layer. This hybrid combination of features helps in learning the feature maps of lower and higher level representations. At first, the tweets are fed to convolutional layers for learning lower-level text features.
$$t_{c1} = tweet(conv_1) \tag{1}$$
$$t_{c11} = t_{c1}(conv_1) \tag{2}$$
The convoluted vector is then fed to the capsule layer for learning the hybrid features representing the higher-level text features.
$$t_{Capsule} = t_{c11}(Capsule) \tag{3}$$
The output vector $t_{Capsule}$ represents the capsule features obtained from the text sample.
2. LSTM features: Similar to the above network, the tweet samples are fed to an LSTM network. LSTM helps in learning the long-term dependency of text samples.
$$t_{c2} = tweet(conv_2) \tag{4}$$
$$t_{c12} = t_{c2}(conv_2) \tag{5}$$
The tweet samples are first fed to two convolutional networks for representing texts into vector forms. The convoluted vector is then passed to LSTM network for getting LSTM features.
$$t_{LSTM} = t_{c12}(LSTM) \tag{6}$$
3. GRU features: GRU and LSTM are used for solving the long-term dependency problem of RNN. Thus, we have used GRU as well for learning the text features.
$$t_{c3} = tweet(conv_3) \tag{7}$$
$$t_{c13} = t_{c3}(conv_3) \tag{8}$$
$$t_{GRU} = t_{c13}(GRU) \tag{9}$$

In each of the networks, tweets are fed to a separate convolutional layer via input layer for the convolution operation. The convoluted output vector is then passed through another convolution layer for learning higher-level features. The learnt features are then passed to capsule layer. Similarly, the convoluted features are also passed to the LSTM and GRU layers for learning three different categories of features. Now, these three extracted features are fused and passed to classification layer for final prediction.

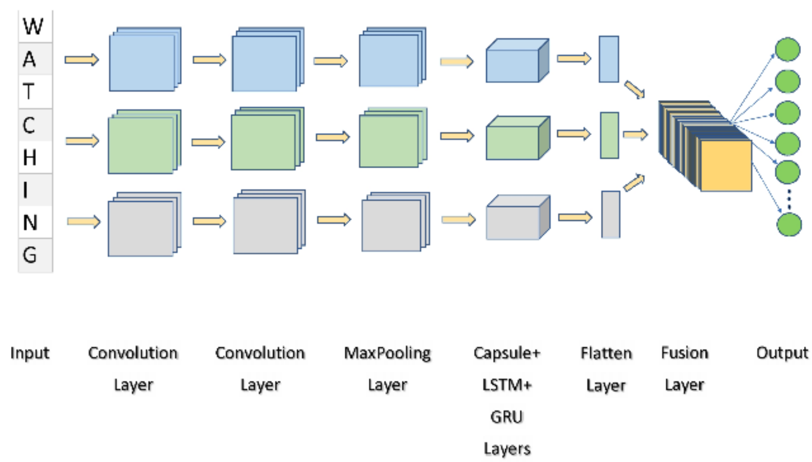


Fig. 1: The Proposed Fusion-based Model

3.2 Classification

Finally all these three features are flattened together. The fused feature vector F_{tweet} is the final vector representing the input tweet. In this way, the final text vector represents all the three features extracted from the feature extraction layers.

$$F_{tweet} = [t_{capsule}; t_{LSTM}; t_{GRU}] \tag{10}$$

This tweet representation is then fed to the dense layer for the prediction of author labels.

$$Labels = F_{tweet}(Dense) \tag{11}$$

The proposed model is depicted in Figure 1. The kernel size is different in each of the sister networks, so those are treated as three different networks. Thus three different types of features are learnt.

4 Dataset Used and Implementation Details

We have used the same dataset released by [21], which is used by other previous approaches too [23,14,4]. The dataset contains a total of 7000 authors, out of which we selected 50 authors at random, where each author has 1000 tweets. The total number of words present in dataset are $101, 180, 659$, with $14, 454.38$ average words per author. There are a maximum of 97 words present in the tweet whereas the minimum sample size is 2, and the average sample size is 14.40.

The implementation details for the developed model are shown in Table 2.

Table 2: Hyperparameters for Neural Network Architecture

Layer	Number of Layers	Hyperparameters
Convolutional-capsule	1	filters:[500], kernel size:[3, 4]
Convolutional-LSTM	1	filters:[500], kernel size:[1, 3]
Convolutional-GRU	1	filters:[500], kernel size:[1, 3]
Capsules	1	no. of capsule:1, dim. of capsule:72
Dense	1	No of authors

Table 3: Performance of Different Input Types on Selected Methods

Input Type	Capsule-1	Capsule-2	LSTM	GRU	Max-Pooling
Unigram	84.03	84.35	67.40	68.98	62.52
Bigram	77.06	77.35	70.80	66.41	69.17
Trigram	76.98	75.94	62.53	61.93	69.04

5 Results and Discussion

Our developed system considers the fusion of capsule, LSTM and GRU features as input. Character n-grams have been used as the input in many of the previous AA tasks [23,14,4], thus we have also developed our model over character n-grams.

Firstly, the single models using capsule, LSTM, and GRU are developed over character n-grams. Accuracy values of 84.03%, 84.35%, 67.40%, and 68.98% are achieved using character uni-grams on capsule-1, capsule-2, LSTM, and GRU, respectively. Using character bi-grams and character tri-grams, the performances are lower than uni-grams. These results are shown in Table 3.

The existing approaches use max-pooling, so the results using max pooling are also shown. An accuracy value of 62.52% is achieved using character uni-grams over max-pooling settings. From the results, it is noticed that character uni-grams with capsule-2 is giving best results, followed by GRU and LSTM. Capsule-2 and capsule-1 are similar architectures, except the filter sizes. The filter size for capsule-2 is 4 and filter size for capsule-1 is 3. Thus, character uni-grams are considered for developing the fusion model. Bi-grams and tri-grams have also been explored in the fusion model, but the performance is not upto the mark, and results over character uni-grams are better. We have fused the capsule-2, LSTM, GRU in four different ways. They are, i) LSTM+GRU,

Table 4: Performance of The Fusion Models on Test Data

Method	Uni-gram	Bi-gram
LSTM+GRU	73.17	66.13
LSTM+Capsule-2	85.29	77.34
GRU+Capsule-2	85.33	77.84
Capsule-2+LSTM+GRU	85.35	78.22

ii) LSTM+Capsule-2, iii) GRU+Capsule-2, and iv) Capsule-2+LSTM+GRU. Accuracy values of 73.17%, 85.29%, 85.33%, and 85.35% are achieved using uni-grams on the above mentioned methods in the respective order. Using bi-grams, the respective accuracies are 66.13%, 77.34%, 77.84%, and 78.22%. The results are shown in Table 4. Thus, it can be concluded that the fusion of all the three features over character uni-grams performs better than others.

Table 5: Comparison of Our Proposed Approach With Other Works, N: No. of Tweets Per User

N	Capsule-4	Fusion	Char-word-CNN	CNN-II	CNN-I	LSTM-W	CNN-W
50	58.63	47.18	47	42	31.2	39.6	45
100	67.23	57.18	57	46.8	31.8	45.4	50
200	73.86	69.69	64	53.9	37	49.9	53.5
500	80.06	80.54	73	63.0	45.7	53.9	62.2
1000	84.64	85.35	79	68.1	51.5	65.06	66.2

We have randomly selected 50 authors for performing the experiments as done in [21,23,4]. In order to analyse the behavior of our developed systems on different sets of authors, we have created 10 different sets of 50 authors having 100 tweets for each. These sets are created after randomly choosing authors from the dataset. The experiments are performed using our developed approach, Char-Word-CNN, CNN-II, and CNN-I, CNN-W, and LSTM-W models on these sets. From the results, it is noticed that, the performances are different for different sets of authors. Number of human-like authors and number of bot-like authors are also shown in the Table 6. These results clearly demonstrate the effects of bot-like and human-like authors. In [23], it is reported that nearly 30% of authors behave like automated bots. Bots are automated programs which pose as humans with the aim at influencing users with commercial, political or ideological purposes [18]. The set of authors, having less number of bot-like authors is having less accuracy in comparison to the set having more number of bot-like authors. Still in all the generated sets of authors, the performances of our developed model are better than the current SOTA ([4]). This analysis also shows the effectiveness of capsule based architectures.

6 Comparison with Other Works

We have compared our results with the previous state-of-the-art methods. The approaches proposed in [14,12] are mainly based on stylometric feature learning. Since, our main idea is to use deep learning methodologies directly, without any feature engineering thus we have not compared our results with those approaches. The architecture proposed in [4] is the current SOTA for the Twitter dataset [21]. All the methods used for comparison are described below.

Table 6: Accuracy (in %) Values for The Different Developed Models on Varying Set of 50 Authors {H: No. of Human-like Authors, B: No. of Bot-like Authors}

Set	H	B	Our Approach	Char-Word-CNN	CNN-II	CNN-I
A	42	8	47.19	46.4	16.4	16.4
B	40	10	51.80	31.6	25.6	20
C	39	11	53.07	34.8	38.8	30.4
D	37	13	55.06	35.6	21.2	17.6
E	37	13	47.46	32.8	27.6	18.4
F	37	13	58.67	42	27.2	25.6
G	37	13	50.54	24	25	24.2
H	36	14	55.60	39.2	36	29.6
I	32	18	51.80	30	30.4	20.4
J	30	20	61.03	29.6	46.8	31.8

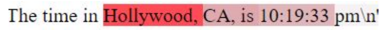
1. Char-Word-CNN: Character bi-grams and word embeddings generated from fastText ([6]) are used as inputs in a multi-channel CNN architecture. This multi-channel learns different stylometric features of an author [4]. This architecture is the current state-of-the-art system for the authorship attribution on the used dataset.
2. CNN-II: Character bi-grams are fed to a CNN architecture, for the classification task [23]. This architecture is the base model and is the first work on short texts using character n-grams over CNN architecture.
3. CNN-I: In this system, the character uni-grams are fed to a CNN architecture [23].

In Table 5, the performance of our fusion model is compared with the performances of other methods as listed above. The accuracy values achieved by our approach are better than the current state-of-the-art (Char-Word-CNN). Out of 7000 authors, only random 50 authors are chosen, and the results are reported in [23,4]. Thus, it is not fair to compare the reported accuracy values with those attained by our developed system. As the source code is also not available, thus, we have implemented their approaches, i.e., CNN-II, and Char-Word-CNN and executed them on our randomly selected authors. From the results, it is clear that, the performances of our model is better than the SOTA system.

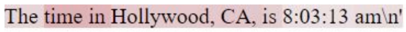
7 Analysis

A rigorous analysis of heat-maps generated through our developed system is performed. Heat-maps are helpful in identifying the words, which are important for prediction. As discussed in the earlier section, there are two types of authors (bot-like, and human-like). Thus, we have drawn heat-maps for one bot-like user, and one human-like user.

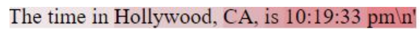
In Figures 2 and 3 the heat-maps generated for a sample of bot-like author, and human-like author are shown, respectively. The heat-maps are drawn for



(a) Heat-map of CNN-II for bot-like author-1: The model assigns more weight to the word “Hollywood” rather than focusing on writing style of author.



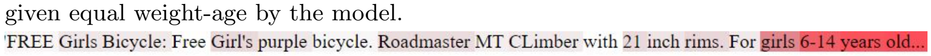
(b) Heat-map of CNN-I for bot-like author-1: As all the tweets are almost similar for this author, the model predicts on the basis of different words in the tweet. Rather than assigning focus on repeating words, this model focuses on the number at the last. This model assigns weights to all words that are common in the tweets.



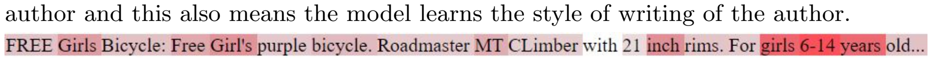
(c) Heat-map of fusion-model for bot-like author-1: Our model assigns more weight to the time pattern than the repeating words. This shows that, the developed system learns the writing style of the user more efficiently.

Fig. 2: Heat-maps Generated From Different Models For Bot-like Author

(a) Heat-map of CNN-I for Human-like author: The CNN-II model predicts the author on the basis of the words he/she uses frequently; here the word ”Girl” is assigned more weight as it is used by the author frequently but as mentioned above the author has repeated the word ”girl” three number of times in his tweets but all the words are not given equal weight-age by the model.



(b) Heat-map of CNN-I for Human-like author: Similar to the CNN-I, this model also assigns equal weight-ages to all the occurrences of the same word while predicting the author and this also means the model learns the style of writing of the author.



(c) Heat-map of fusion-model for Human-like author: The fusion-model correctly finds the unique characteristic of the author in repeating similar words in the tweet; in this tweet the ”Girl” word is used thrice and the model has assigned equal weight-age to all the occurrences of the word.

Fig. 3: Heat-maps Generated From Different Models For Human-like Author

CNN-II, CNN-I, and fusion-based model. It can be seen that the previous models CNN-II, and CNN-I assign more priority to the word ‘Hollywood’. On the other hand, our proposed fusion-based model assigns more priority to the time (‘10:19:33’). Hollywood and time both are present in most of the samples. But the numerical value for the time is different in different samples, only the pattern of writing time is similar. Similarly, in Figure 3 the heat-map drawn for a sample of human-like user is shown. From the above examples, it can be concluded that our developed system is capable of capturing the writing pattern of the user more effectively.

8 Conclusion and Future Work

The application area of the authorship attribution task includes analyzing long fraud document, terrorist conspiracy texts, short letters, blog posts, emails, SMS, Twitter streams or Facebook status updates to check the authenticity and identify fraudulence. In this work, a fusion-based capsule network is developed for solving the authorship attribution task. Character uni-grams are fed to a convolution layer for learning text representations. The convoluted vector is fed to different components such as LSTM, GRU, and capsule. Different features extracted from these components are fused together and then used for the final classification. Our work illustrates the effect of fusing these three sets of features, by achieving gain in performance. An accuracy of 85.35% is achieved using the fused model, and it outperforms the previous developed systems. With the help of heat-maps, we have also shown the relevant fragments of text sample, for solving the AA task.

In future, we would like to use different style based features with the neural network settings. Different discourse features can also be added to the developed system, which can lead to performance improvement.

References

1. Aborisade, O., Anwar, M.: Classification for authorship of tweets by comparing logistic regression and naive bayes classifiers. In: 2018 IEEE International Conference on Information Reuse and Integration (IRI). pp. 269–276. IEEE (2018)
2. Akhtar, S., Ghosal, D., Ekbal, A., Bhattacharyya, P., Kurohashi, S.: All-in-one: Emotion, sentiment and intensity prediction using a multi-task ensemble framework. IEEE Transactions on Affective Computing (2019)
3. Al Marouf, A., Hasan, M.K., Mahmud, H.: Comparative analysis of feature selection algorithms for computational personality prediction from social media. IEEE Transactions on Computational Social Systems (2020)
4. Aykent, S., Dozier, G.: Author identification of micro-messages via multi-channel convolutional neural networks. In: 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC). pp. 675–681. IEEE (2020)
5. Bagnall, D.: Author identification using multi-headed recurrent neural networks. arXiv preprint arXiv:1506.04891 (2015)
6. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics **5**, 135–146 (2017)
7. Brocardo, M.L., Traore, I., Saad, S., Woungang, I.: Authorship verification for short messages using stylometry. In: 2013 International Conference on Computer, Information and Telecommunication Systems (CITS). pp. 1–6. IEEE (2013)
8. Castillo, E., Vilarino, D., Cervantes, O., Pinto, D.: Author attribution using a graph based representation. In: 2015 International Conference on Electronics, Communications and Computers (CONIELECOMP). pp. 135–142. IEEE (2015)

9. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014)
10. Fourkioti, O., Symeonidis, S., Arampatzis, A.: Language models and fusion for authorship attribution. Information Processing & Management **56**(6), 102061 (2019)
11. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
12. Huang, W., Su, R., Iwaihara, M.: Contribution of improved character embedding and latent posting styles to authorship attribution of short texts. In: Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data. pp. 261–269. Springer (2020)
13. Joshi, M., Singh, P., Zincir-Heywood, N.: Compromised tweet detection using siamese networks and fasttext representations. In: 2019 15th International Conference on Network and Service Management (CNSM). pp. 1–5. IEEE (2019)
14. Joshi, M., Zincir-Heywood, N.: Classification of micro-texts using subword embeddings. in RANLP (2019)
15. Juola, P.: Authorship studies and the dark side of social media analytics. J. UCS **26**(1), 156–170 (2020)
16. Kim, J., Jang, S., Park, E., Choi, S.: Text classification using capsules. Neurocomputing **376**, 214–221 (2020)
17. Madigan, D., Genkin, A., Lewis, D.D., Argamon, S., Fradkin, D., Ye, L., Consulting, D.D.L.: Author Identification on the Large Scale p. 20
18. Rangel, F., Rosso, P.: Overview of the 7th author profiling task at pan 2019: Bots and gender profiling in twitter. In: Proceedings of the CEUR Workshop, Lugano, Switzerland. pp. 1–36 (2019)
19. Rhodes, D.: Author Attribution with CNN's p. 8
20. Ruder, S., Ghaffari, P., Breslin, J.G.: Character-level and Multi-channel Convolutional Neural Networks for Large-scale Authorship Attribution. arXiv:1609.06686 [cs] (Sep 2016), `http://arxiv.org/abs/1609.06686`, arXiv: 1609.06686
21. Schwartz, R., Tsur, O., Rappoport, A., Koppel, M.: Authorship attribution of micro-messages. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. pp. 1880–1891 (2013)
22. Segarra, S., Eisen, M., Ribeiro, A.: Authorship attribution through function word adjacency networks. IEEE Transactions on Signal Processing **63**(20), 5464–5478 (2015)
23. Shrestha, P., Sierra, S., González, F.A., Montes, M., Rosso, P., Solorio, T.: Convolutional neural networks for authorship attribution of short texts. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers. pp. 669–674 (2017)
24. Stamatatos, E.: A survey of modern authorship attribution methods. Journal of the American Society for information Science and Technology **60**(3), 538–556 (2009)