Gated POS-Level Language Model for Authorship Verification

Linshu Ouyang , **Yongzheng Zhang**[*] , **Hui Liu** , **Yige Chen** and **Yipeng Wang**
Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{ouyanglinshu, zhangyongzheng, liuhui, chenyige, wangyipeng}@iie.ac.cn,

Abstract

Authorship verification is an important problem that has many applications. The state-of-the-art deep authorship verification methods typically leverage character-level language models to encode author-specific writing styles. However, they often fail to capture syntactic level patterns, leading to sub-optimal accuracy in cross-topic scenarios. Also, due to imperfect cross-author parameter sharing, it's difficult for them to distinguish author-specific writing style from common patterns, leading to data-inefficient learning.

This paper introduces a novel POS-level (Part of Speech) gated RNN based language model to effectively learn the author-specific syntactic styles. The author-agnostic syntactic information obtained from the POS tagger pre-trained on large external datasets greatly reduces the number of effective parameters of our model, enabling the model to learn accurate author-specific syntactic styles with limited training data. We also utilize a gated architecture to learn the common syntactic writing styles with a small set of shared parameters and let the author-specific parameters focus on each author's special syntactic styles. Extensive experimental results show that our method achieves significantly better accuracy than state-of-the-art competing methods, especially in cross-topic scenarios (over 5% in terms of AUC-ROC).

1 Introduction

With a set of documents written by a known author, how can we determine whether another document is written by this author? This is the authorship verification problem [Koppel and Schler, 2004], which has important applications in broad domains, e.g., to detect fraud or phish in cybersecurity, to detect plagiarism in research, and to assist forensic investigation for the judiciary.

Numerous authorship verification methods have been introduced to tackle this problem [Stamatatos, 2009]. Most of these approaches focus on finding features that can effectively characterize writing styles [Abbasi and Chen, 2008]. Character level tri-grams is shown to be effective for authorship analysis [Bevendorff *et al.*, 2019b]. There are also works trying to address this problem from the perspective of compression [Khmelev and Teahan, 2003; Halvani *et al.*, 2017] and the distribution of function words [Koppel and Schler, 2004]. [Zhao *et al.*, 2006] demonstrates that the POS tag is effective to characterize the writing styles from the perspective of syntax. However, they only use simple distributions to represent the pattern of each author, which is insufficient to capture the local correlations in the sequence.

Recently, deep learning has achieved exceptional successes in many natural language processing problems. However, applying deep learning on the authorship verification problem faces a major challenge: the amount of training texts for each author is extremely limited, which poses significant challenges to the training of neural networks that have a large number of parameters.

[Bagnall, 2015] is one of the few in applying deep learning to the authorship verification problem. They attempt to address the above challenge by building a character-level language model with a shared RNN layer and a separate output layer for each author to encode the author-specific writing styles. Their method sits at the first place in the PAN-15 authorship verification competition and demonstrates the great potential of deep learning for authorship verification. However, we find experimentally that character-level RNN has difficulties in learning syntactic information with a limited number of samples. In addition, although their model uses a shared RNN layer, the separate fully-connected output layers for each author have to learn from scratch independently, which may lead to sub-optimal language models. This also makes it difficult for their model to distinguish author-specific writing styles from common writing patterns shared by everyone.

In this paper, we introduce a novel POS-level gated language model to capture the author-specific syntactic writing styles in a data-efficient way. As shown in Figure 2, we utilize a POS tagger pre-trained on a large external data set to convert the raw inputs into sequences of POS tokens. The external author-agnostic syntactic information comes with the POS tagger limits the parameter space of our language model to the vicinity of common syntactic patterns. This greatly reduces the number of the effective parameters and enables the

[*] Contact Author

Known Author	Unknown Author	Same?
A beautiful lake lies at the foot of the hill . DET ADJ NOUN VERB ADP DET NOUN ADP DET NOUN PUNCT	At the foot of the hill lies a beautiful lake . ADP DET NOUN ADP DET NOUN VERB DET ADJ NOUN PUNCT	NO
This is the second article that I have written . DET AUX DET ADJ NOUN DET PRON AUX VERB PUNCT	It is the best film that he has ever seen . PRON AUX DET ADJ NOUN SCONJ PRON AUX ADV VERB PUNCT	YES

Figure 1: Two toy authorship verification cases. Note that they are unrealistically simplified since typically it requires at least a few hundred words to decently characterize an author's writing style. The two sentences in the first case express similar meanings, but their syntactic styles are completely different. In the second case, the syntactic styles of the two sentences are similar, but the topics are different. The PoS tagset used here is the universal PoS tagset.

model to focus on the details of each author's special syntactic styles.

Also, we use a shared fully-connected layer and a set of independent fully-connected layers to respectively learn the shared syntactic patterns and each author's special syntactic writing styles. A gate unit shared between authors is responsible for deciding whether to use author-specific information at different positions of the texts. The rationale behind this design is that we assume that only a small part of the patterns appeared in the POS sequences comes from author-specific writing styles, while most patterns come from the standard syntax. This structure has two major benefits: (i) the gate can prevent the model from overfitting to each author's rather limited training data since it makes the decision based on the entire training set, and (ii) learning common syntactic patterns with a small set of shared parameters makes the language model more data-efficient and accurate.

To summarize, our main contributions are as follows:

- We propose a novel POS-level language model to effectively encode the topic-agnostic syntactic writing style for each author. As far as we know, we are the first to conduct authorship verification by capturing the author-specific correlations in the sequences of POS tokens with language models.
- We propose a novel gated architecture to learn the author-specific language models by utilizing a relatively small group of parameters to learn the common syntactic patterns shared across all authors. The model is data-efficient and less prone to overfitting.
- Extensive empirical results on four publicly available datasets demonstrate that our model can effectively learn each author's unique syntactic writing style and achieve significant accuracy improvements compared to other state-of-the-art methods, especially in cross-topic scenarios (over 5% in terms of AUC-ROC).

2 Preliminaries

2.1 Problem Statement

Suppose there is a group of authors A, for each author $a_i \in A$ we have a small set of texts $S_i = \{s_1^i, s_2^i, \cdots, s_k^i\}$ that is known to be written by a_i. Now there is also a text of unknown author denoted as q_t, the purpose of authorship verification is to determine whether q_t is written by author a_i. $\{a_i, q_t\}$ is called an authorship verification case. Figure 1 shows two toy cases.

2.2 Language Model for Authorship Verification

The core idea of neural language model [Bengio *et al.*, 2003] is learning a neural network to model the conditional probability $P(w_i|w_0, \cdots, w_{i-1})$ for each sequence $w_0, \cdots, w_N$. $w_0, \cdots, w_{i-1}$ is called context and w_i is a token that is either a character or a word. Typically, this model consists of two components: a recurrent neural network unit f synthesize the correlations from the context into a hidden representation $h_i = f(h_{i-1}, w_{i-1})$; and a fully-connected decoding layer g to map the hidden representation to the probability distribution of next token: $P(w_i|w_0, \cdots, w_{i-1}) = g(h_i)$.

To conduct authorship verification, a set of author-specific language models can be learned for each author respectively. Each author's special writing habits will lead to differences in learned conditional probability $P(w_i|w_0, \cdots, w_{i-1})$, and a text is expected to have high probability with respect to its sincere author's language model.

3 Methods

Figure 2 gives an overview of the gated POS-level author-specific language model. First, we convert the raw texts into the sequences of POS tokens. These sequences are then embedded and fed into a vanilla RNN shared by all authors to capture the context information. On top of that, we apply gated decoding to predict the next token in the sequence.

3.1 POS-Level Language Model

Instead of the traditional character-level language model, we propose to use a novel POS-level language model to learn author-specific syntactic writing styles. Specifically, we utilize a POS tagger pre-trained on a large external data set [Matthew and Ines, 2015; Qi *et al.*, 2018] to convert the raw texts into sequences of POS tokens. Then we learn author-specific language models upon these sequences, that is, predict the next POS token based on previously observed POS sequence. The POS-level language model can effectively capture the syntactic writing styles with limited training data while the character-level language model often fails to do so.

Why Is Syntactic Writing Style Important for Authorship Verification?

The syntactic writing style is important because it is topic-agnostic. Consider the two toy examples in Figure 1. The two sentences in the first case convey similar meaning with exactly the same words, but the arrangement of their POS token sequences are completely different. For the second case, the two sentences have different topics but similar POS token sequences. It is evident that the syntactic writing style represented by the arrangement pattern of POS token sequences is more generalizable for authorship verification in cross-topic scenarios.

Why Can POS Level Language Model Better Learn Syntactic Writing Style?

The language model essentially models the joint probability distribution of a sequence of discrete random variables. This distribution has an exponentially sized sample space, thus the language model will need a large amount of training data to explore this huge sample space and accurately fit the distribution. However, in the authorship verification problem, there are typically not enough training data to do this. As a result, the char-level language model is likely to overfit to some special local correlations in the training set.

To address this problem, a potential solution is to leverage external knowledge to constraints the size of the sample space. We can consider that there is a subspace consisting of all the ”correct” sentences obeying the common syntactic rules in the sample space. Each author will have a different probability distribution in this subspace according to their syntactic writing style. With this intuition, we leverage the general author-independent syntactic rules obtained from POS taggers trained on external large-scale data sets to limit the sample space of the language model to the vicinity of ”correct” sentences, enabling the author-specific language models to focus on each author’s subtle syntactic style. From another perspective, the vocabulary size of the POS is only 19, compared to the character (80) and the word (several thousand). This greatly reduces the number of effective parameters of the model while retaining its expressive power. Besides POS tags, we also explored the possibility of using more explicit syntactic information obtained from dependency parsing. However, it performs consistently worse than POS tags.

3.2 Gated Neural Language Model

Given a sequence of POS tokens $P = \{w_t\}_{t=1}^{m}$ known to be written by author $a_i \in A$, we first convert them to embeddings $\{\mathbf{e}_t\}_{t=1}^{m}$ with a embedding layer. Then we generate the representation of context $\{\mathbf{h}_t\}_{t=1}^{m}$ utilizing a vanilla RNN unit to capture the correlations in the POS sequence:

$$\mathbf{h_t} = \mathrm{RNN}(\mathbf{h}_{t-1}, \mathbf{e}_{t-1}) \tag{1}$$

The parameters of the embedding layer and the RNN are shared between all authors. We choose vanilla RNN rather than more powerful variance such as LSTM because we find experimentally that LSTM typically provides rather limited performance improvements. We speculate that this might be due to the limited training data which prevents the LSTM to learn meaningful long-range correlations.

Then, the representation of context is fed into a decoder to generate the estimation of the distribution over the next token. This decoder consists of three parts:

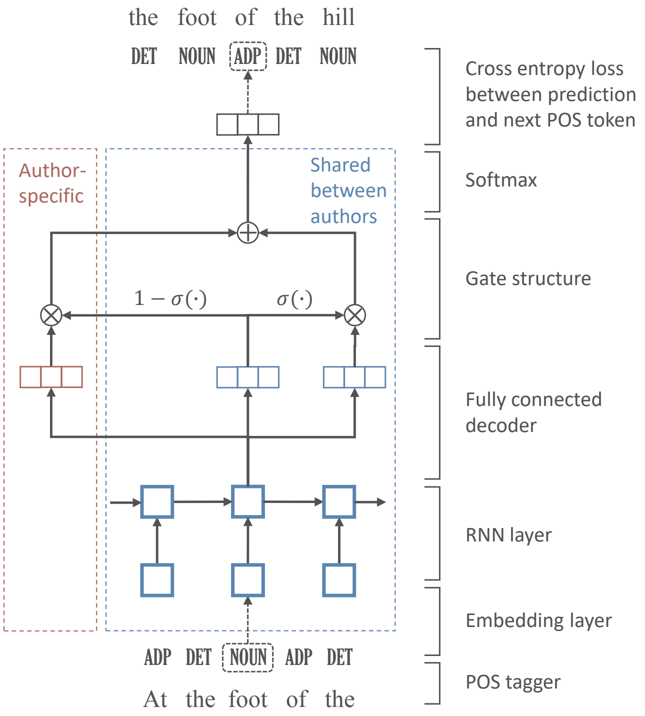

Figure 2: Gated neural language model structure overview.

(i) A shared fully-connected layer with parameters $\{\mathbf{W}^{share}, \mathbf{b}^{share}\}$ to learn the common syntactic writing patterns among all authors:

$$\mathbf{s}_t^{share} = \mathbf{W}^{share}\mathbf{h}_t + \mathbf{b}^{share} \tag{2}$$

(ii) A separate fully-connected layer with parameters $\{\mathbf{W}^{a_i}, \mathbf{b}^{a_i}\}$ for each author $a_i \in A$ to learn special syntactic writing styles:

$$\mathbf{s}_t^{a_i} = \mathbf{W}^{a_i}\mathbf{h}_t + \mathbf{b}^{a_i} \tag{3}$$

(iii) The final output is the linear weighted average of the outputs of the previous two parts performed by a shared gate unit parameterized by $\{\mathbf{W}^{gate}, \mathbf{b}^{gate}\}$:

$$\mathbf{s}_t^{gate} = \sigma(\mathbf{W}^{gate}\mathbf{h}_t + \mathbf{b}^{gate}) \tag{4}$$

$$\mathrm{logits}_t^{a_i} = \mathbf{s}_t^{gate} \otimes \mathbf{s}_t^{share} + (1 - \mathbf{s}_t^{gate}) \otimes \mathbf{s}_t^{a_i} \tag{5}$$

Finally, a softmax function is applied to the raw logits output to obtain the author-specific estimated distribution of the next token:

$$\hat{\mathrm{P}}^{a_i}(w_t | \{w_k\}_{k=1}^{t-1}) = \mathrm{softmax}\left(\mathrm{logits}_t^{a_i}\right) \tag{6}$$

To train the network, we minimize the sum of the cross-entropy loss between the estimated distribution and the ground-truth next token on every step t.

Dataset	Genre	#Train	#Test	Char Level Length			Word Level Length		
				Min.	Max.	Avg.	Min.	Max.	Avg.
PAN-15	Dialog in plays	100	500	1304	5024	2566	283	1407	638
Gutenberg	Fiction	182	80	5783	25972	21835	1158	5889	4809
Enron	Email	16	64	1894	5428	3958	378	1108	797
Reddit	Social news	200	800	4989	7496	6978	356	1958	1485

Table 1: The statistics of the four publicly available datasets used in the experiments.

Why Does Our Proposed Architecture Learn Author-specific Syntactic Writing Style More Accurately?

The assumption behind this design choice is that there is a large intersection between the writing patterns of the authors, and only a small part of special writing patterns can be utilized to distinguish these authors. With this architecture, we learn the shared writing patterns with a small set of shared parameters, and the author-specific parameters can focus on each author's special writing styles.

Also, note that the parameters of the gate are shared, thus the ratio to include author-specific information at each step of the text sequence is determined by all training data. As a result, it will be less likely for the model to overfit to the training data of each author.

3.3 Verification

After learning a language model for each known author respectively, to conduct authorship verification on a test case $\{a_i, \{w_t\}_{t=1}^{m}\}$, we need to calculate the losses $\mathbf{z} = \{z_{a_i} | a_i \in A\}$ of this text with respect to the language model of each author respectively:

$$z_{a_i} = \mathrm{L}\left(a_i, \{w_t\}_{t=1}^{m}\right) \tag{7}$$

$$= \sum_{t=1}^{m} \left(\hat{\mathrm{P}}^{a_i}(w_t | \{w_k\}_{k=1}^{t-1}), w_t \right) \tag{8}$$

Then, we normalize these losses:

$$\text{score} = \frac{z_{a_i} - \text{mean}(\mathbf{z})}{\text{std}(\mathbf{z})} \tag{9}$$

This normalized loss for a_i can be used as a score for classification. The smaller the loss, the more likely this text is written by the author a_i.

The rationale behind this choice is that due to the inherent variability of neural language models, it's difficult to directly compare the losses of different texts with respect to the same language model. Instead, we can only compare the losses of the same text with respect to different author's language models.

4 Experiments

In this section, we conduct extensive experiments to evaluate the proposed method. First, we introduce the settings of the experiments, including the datasets used for evaluation, the baseline methods to compare, the implementation details of our method, and the metrics. Then we compare the performance of our proposed method with other state-of-the-art authorship verification methods. Finally, we conduct ablation studies to evaluate the contribution of each component of our model.

4.1 Data Sets

As shown in the Table 1, we used 4 publicly available authorship verification datasets, which were widely used by previous studies [Halvani *et al.*, 2017; Bevendorff *et al.*, 2019a; Bagnall, 2015; Halvani *et al.*, 2018], with different genre and sizes. Each case in these datasets consists of exactly one document of unknown authorship and at least one document from a known author. PAN-15 comes from a well-known authorship verification competition [Stamatatos *et al.*, 2015]. Recently, the authors of PAN-15 conduct a detailed analysis of the dataset and find several deficits such as special characters introduced during the processing of the dataset and accidental text overlap. Based on these insights, they constructed an improved authorship verification corpus named Gutenberg [Bevendorff *et al.*, 2019a]. This dataset eliminates the aforementioned bias and flaws, enabling the dataset to evaluate the performance of authorship verification methods more fairly. Therefore, this dataset is considered the most important one in our experiments. Nevertheless, we still included the PAN-15 dataset for completeness. The Enron data set comes from the public Enron emails dataset. The samples are manually selected and cleaned to make sure of the quality [Halvani *et al.*, 2018]. The Reddit dataset is obtained from [Halvani *et al.*, 2017].

These data sets are respectively divided into two parts, the development set, and the test set. The PAN-15 and Gutenberg datasets have been pre-divided by the provider. The other two datasets are randomly divided according to the ratio of 1: 4 following the common practice in the research of authorship verification [Halvani *et al.*, 2017].

4.2 Competing Methods

We compare our model with five other methods, including two baseline methods, BAFF [Bevendorff *et al.*, 2019a] and its variation that use POS tags as inputs, two non-deep learning methods GLAD [Hürlimann *et al.*, 2015] and Compression [Halvani *et al.*, 2017], and a deep learning based method MultiHead [Bagnall, 2015], which won the 1st place on the

Methods	PAN-15		Gutenberg		Enron		Reddit	
	AUC-ROC	AUC-PR	AUC-ROC	AUC-PR	AUC-ROC	AUC-PR	AUC-ROC	AUC-PR
BAFF	0.7445	0.7098	0.7026	0.6638	0.7439	0.6784	0.7420	0.6901
BAFF-POS	0.6813	0.6345	0.6742	0.6854	0.6961	0.6652	0.6407	0.6111
Compression	0.6400	0.6014	0.7633	0.7152	0.7810	0.7301	0.7689	0.7141
GLAD	0.6528	0.6621	0.8443	0.8615	0.7191	0.7491	0.6795	0.6721
MultiHead	0.8165	0.8244	0.7849	0.8084	0.9369	0.9618	0.8544	0.8737
GatedLM	**0.8434**	**0.8274**	**0.8912**	**0.8903**	**0.9497**	**0.9633**	**0.8714**	**0.8782**

Table 2: Authorship verification performance comparison.

PAN 2015 author identification task [Stamatatos *et al.*, 2015]. We chose these methods because they are the state of the art in the field of author verification.

We reimplement the two baseline methods with Python, and MultiHead with PyTorch [Paszke *et al.*, 2019]. For other methods, we use codes obtained from their authors.

4.3 Implementation Details

We implement the proposed method with PyTorch and run all the experiments on a GPU with 11GB memory.

How to perform POS tagging is not the focus of this article, therefore we rather arbitrarily choose the popular open-source POS tagger Spacy [Matthew and Ines, 2015]. Other POS taggers such as StanfordNLP [Qi *et al.*, 2018] can achieve similar results.

In order to conduct fairly performance comparison to make sure the performance improvement of our method comes from the POS level language model and gated decoder, we use the vanilla RNN same as MultiHead.

The most important hyperparameters of our model are the size of the embedding layer and the size of the hidden layer. These hyperparameters can be effectively selected on the development set and do not vary much on different datasets. Except for the Enron dataset where the best hidden size is 16, on other datasets the model achieves the best results on the development set when the hidden size is 64.

In training, we follow the common practice to calculate the gradient with truncated backpropagation through time [Werbos and others, 1990; Graves, 2013]. Specifically, we truncated the gradient for 20 time-steps and perform gradient descent utilizing vanilla SGD optimizer combined with gradient clipping at 0.25. All models are trained for 100 epochs and the learning rate is set to 5. With these settings, we achieve stable performance on the development set.

4.4 Performance Evaluation Methods

We use two complementary indicators: AUC-ROC and AUC-PR to measure and compare model performance. The AUC-ROC is the area under the ROC curve, summing up true positives against false positives. The AUC-PR is the area under the curve of precision against the recall. We use average precision in [Manning *et al.*, 2008], a widely-used method, to calculate AUC-PR. We chose these two metrics because they are classification-threshold-invariant metrics. It allows us to focus on comparing the predictive power of each method. For a specific application, a threshold can be chosen to trade between precision and recall.

Due to the inherent variation of the algorithm, the experimental results reported in this paper comes from the average of 10 independently running with the same setting.

4.5 Performance Comparision

This section examines the performance of our method on aforementioned datasets. There are two major questions we aim to address:

- Is it necessary to utilize deep learning methods for authorship verification problem?
- Is our method more accurate than the state-of-the-art deep learning based method?

The AUC-ROC and AUC-PR performance of our method and other competing methods are listed in Table 2. It is evident that deep learning based methods MultiHead and our GatedLM outperform traditional models on all four datasets, which indicates that deep learning based methods can beat sophisticated manually designed features. Also, notice that the BAFF on POS perform worse than the raw BAFF on character. This implies that simply using distribution of POS tags is unable to unleash its full potential.

Comparing our method with MultiHead, we can obtain the main result that the POS level information and the gated neural language model enable our method to achieve significant performance improvement. In particular, our method achieves the most significant performance improvement in cross-topic scenario on Gutenberg dataset.

4.6 Ablation Study

In this section, we perform ablation studies on the two key components of our proposed method to examine their contribution to performance improvement respectively. We aim to answer the following three key questions:

- Is it necessary to share parameters between language models of authors?
- Does our gated neural language model achieve performance improvement over the multi-head structure?

- Does POS information have advantages over char information, and under what circumstances?

To answer the first question, we build a baseline model, which learns a completely independent language for each author. This model has no shared parameter.

From the experimental results listed in Table 3 we can find that both POS-level information and gated architecture make major contributions to the performance improvement. The baseline model without any parameter sharing performs significantly worse than multi-head and gated architecture, which indicates that it has difficulties in learning accurate language model for each author with extremely limited training data. Therefore, a well-designed cross-author parameter sharing is the key to learn more accurate author-specific language models by reducing the number of effective parameters.

Also, our gated architecture achieves higher accuracy compared to the multi-head when both trained on POS level inputs. This demonstrates that our gated decoder can indeed distinguish the author-specific writing styles from common patterns and learn the common patterns with shared parameters. As a result, our model is less prone to overfitting and can learn a more accurate language model.

Not surprisingly, when using the same model architecture, the POS-level language model outperforms the char-level language model by a large margin on the Gutenberg dataset but only achieves moderate performance improvement on the PAN-15 dataset. This is due to that the PAN-15 dataset contains special characters introduced during the process of the dataset. Character level models can easily overfit these special characters and obtain biased high accuracy.

In order to quantitatively compare the importance of the author-specific decoder and shared decoder, we calculate the mean absolute logits of them respectively. We also calculate the mean value of the gates. The experimental results listed in Table 4 demonstrate that the response amplitude of the shared decoder is much larger than the amplitude of the author-specific decoder on all datasets, which provide evidence to further support our claim that the model can learn common writing patterns by shared decoder and author-specific styles by stand-alone decoder for each author.

Model	Level	PAN-15	Gutenberg
GatedLM	**POS**	**0.8434**	**0.8912**
GatedLM	Char	0.8181	0.8199
MultiHead	POS	0.8321	0.8462
MultiHead	Char	0.8165	0.7849
Naive	POS	0.8063	0.8599
Naive	Char	0.7875	0.7161

Table 3: Ablation studies. (AUC)

Dataset	logits author	logits share	gate
PAN-15	0.5417	2.9452	0.5504
Gutenberg	0.7145	6.0114	0.3954
Enron	0.5287	3.2292	0.4794
Reddit	0.4022	3.6439	0.5716

Table 4: The contribution of author specific logits and shared logits.

5 Related Work

The authorship verification problem is first proposed and studied in [Koppel and Schler, 2004], and is closely related to authorship attribution problem. While being almost solved on long texts [Bevendorff *et al.*, 2019a], authorship verification is still a challenging task on short texts.

5.1 Traditional Authorship Verification

There are numerous authorship verification methods attempting to tackle this problem from different aspects [Stamatatos, 2009].

The very first work on authorship verification [Koppel and Schler, 2004] proposed an unmasking method that mainly focused on the analysis of most frequent function words. Writeprints [Abbasi and Chen, 2008] extract more than twenty stylometric features including lexical, syntactic, and structural text features to conduct authorship analysis. BAFF [Bevendorff *et al.*, 2019a] relies on several simple measures (e.g, TF-IDF) defined upon char level trigram. The distribution of POS tags can also be utilized to measure the style differences [Zhao *et al.*, 2006]. Compression is another well-studied path to authorship verification which is first proposed in [Khmelev and Teahan, 2003] and recently improved by [Halvani *et al.*, 2017].

5.2 Deep Authorship Analysis

Only a few studies have been done on deep authorship verification and deep authorship attribution. In [Bagnall, 2015], a character level language model is learned for each author to capture their unique writing styles. Texts will have a higher probability coming from the language model of its true author. Recently, several methods use convolution neural network to identify the authorship. Their main differences are the formats of the inputs. [Hitschler *et al.*, 2017] first transforms the raw texts into sequences of POS tags, then use a convolution neural network to predict the author. [Shrestha *et al.*, 2017] proposes to use character n-gram as the input of the convolution neural network. [Ruder *et al.*, 2016] proposes to combine the char-level input and the word-level input with a multi-channel neural network.

6 Conclusions

We have presented our novel POS-level (Part of Speech) gated RNN based language model that can learn the author-specific syntactic styles with high data efficiency. We first propose to use a POS-level language model to obtain the syntactic level information. Then we design a novel gated decoder architecture to improve parameter sharing and enable data-efficient learning of the author-specific language model. Extensive experimental results show that our model outperforms other state-of-the-art competing methods by a large margin in terms of accuracy.

References

[Abbasi and Chen, 2008] Ahmed Abbasi and Hsinchun Chen. Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Transactions on Information Systems (TOIS)*, 26(2):7, 2008.

[Bagnall, 2015] Douglas Bagnall. Author identification using multi-headed recurrent neural networks. *arXiv preprint arXiv:1506.04891*, 2015.

[Bengio *et al.*, 2003] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[Bevendorff *et al.*, 2019a] Janek Bevendorff, Matthias Hagen, Benno Stein, and Martin Potthast. Bias analysis and mitigation in the evaluation of authorship verification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6301–6306, 2019.

[Bevendorff *et al.*, 2019b] Janek Bevendorff, Martin Potthast, Matthias Hagen, and Benno Stein. Heuristic authorship obfuscation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1098–1108, 2019.

[Graves, 2013] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[Halvani *et al.*, 2017] Oren Halvani, Christian Winter, and Lukas Graner. Authorship verification based on compression-models. *arXiv preprint arXiv:1706.00516*, 2017.

[Halvani *et al.*, 2018] Oren Halvani, Lukas Graner, and Inna Vogel. Authorship verification in the absence of explicit features and thresholds. In *European Conference on Information Retrieval*, pages 454–465. Springer, 2018.

[Hitschler *et al.*, 2017] Julian Hitschler, Esther van den Berg, and Ines Rehbein. Authorship attribution with convolutional neural networks and pos-eliding. In *Proceedings of the Workshop on Stylistic Variation*, pages 53–58, 2017.

[Hürlimann *et al.*, 2015] Manuela Hürlimann, Benno Weck, Esther van den Berg, Simon Suster, and Malvina Nissim. Glad: Groningen lightweight authorship detection. In *CLEF (Working Notes)*, 2015.

[Khmelev and Teahan, 2003] Dmitry V Khmelev and William J Teahan. A repetition based measure for verification of text collections and for text categorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 104–110. ACM, 2003.

[Koppel and Schler, 2004] Moshe Koppel and Jonathan Schler. Authorship verification as a one-class classification problem. In *Proceedings of the twenty-first international conference on Machine learning*, page 62. ACM, 2004.

[Manning *et al.*, 2008] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*. Cambridge university press, 2008.

[Matthew and Ines, 2015] Honnibal Matthew and Montani Ines. spacy: Industrial-strength nlp. https://spacy.io/, 2015. [Version=2.2.3].

[Paszke *et al.*, 2019] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019.

[Qi *et al.*, 2018] Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. Universal dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170, Brussels, Belgium, October 2018. Association for Computational Linguistics.

[Ruder *et al.*, 2016] Sebastian Ruder, Parsa Ghaffari, and John G Breslin. Character-level and multi-channel convolutional neural networks for large-scale authorship attribution. *arXiv preprint arXiv:1609.06686*, 2016.

[Shrestha *et al.*, 2017] Prasha Shrestha, Sebastian Sierra, Fabio A González, Manuel Montes, Paolo Rosso, and Thamar Solorio. 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*, pages 669–674, 2017.

[Stamatatos *et al.*, 2015] Efstathios Stamatatos, Walter Daelemans, Ben Verhoeven, Patrick Juola, Aurelio Lopez-Lopez, Martin Potthast, and Benno Stein. Overview of the author identification task at pan 2015. In *CLEF 2015 Working Notes Papers*, 2015.

[Stamatatos, 2009] Efstathios Stamatatos. A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, 60(3):538–556, 2009.

[Werbos and others, 1990] Paul J Werbos et al. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[Zhao *et al.*, 2006] Ying Zhao, Justin Zobel, and Phil Vines. Using relative entropy for authorship attribution. In *Asia Information Retrieval Symposium*, pages 92–105. Springer, 2006.