LETTER

A Partial Matching Convolution Neural Network for Source Retrieval of Plagiarism Detection

Leilei KONG[†], **Yong HAN**[†], **Haoliang QI**[†a)], ***Nonmembers***, ***and*** **Zhongyuan HAN**[†], ***Member***

SUMMARY Source retrieval is the primary task of plagiarism detection. It searches the documents that may be the sources of plagiarism to a suspicious document. The state-of-the-art approaches usually rely on the classical information retrieval models, such as the probability model or vector space model, to get the plagiarism sources. However, the goal of source retrieval is to obtain the source documents that contain the plagiarism parts of the suspicious document, rather than to rank the documents relevant to the whole suspicious document. To model the "partial matching" between documents, this paper proposes a *Partial Matching Convolution Neural Network* (PMCNN) for source retrieval. In detail, PMCNN exploits a sequential convolution neural network to extract the plagiarism patterns of contiguous text segments. The experimental results on PAN 2013 and PAN 2014 plagiarism source retrieval corpus show that PMCNN boosts the performance of source retrieval significantly, outperforming other state-of-the-art document models.

key words: *plagiarism detection, source retrieval, partial matching, convolution neural network*

1. Introduction

Source retrieval (SR) is one of the most important tasks of plagiarism detection. It can be described as: given a suspicious document d_{plg} that may contain plagiarized passages and a document set D, source retrieval identifies a small collection of candidate source documents $D_{src} \subseteq D$ that are likely sources for plagiarism regarding d_{plg} [1], [2].

Existing SR methods usually take source retrieval as an issue of information retrieval (IR) [3]. IR-based methods usually split the suspicious document into text segments at first to obtain the possible plagiarism parts of a suspicious document. Then some queries are extracted from these segments using some pre-defined rules and submitted to a search engine to retrieve the relevant documents [1]–[3]. Using the information returned by the search engine (such as the BM25 score of the search result, the number of words in the retrieved result, or whatever) or the snippets of search results to learn a classifier, these relevant documents are compared with the suspicious document to obtain the candidate source documents [1], [2], [6].

However, such methods do not give sufficient thought to the difference between source retrieval and information retrieval. The goal of information retrieval is to rank the documents according to the relevance between documents and query [4], [5]. But in source retrieval, suspicious documents are generally not full-text plagiarism, but only plagiarize some text segments of the source documents. The goal of source retrieval is to retrieve the source documents that match the plagiarism parts of a suspicious document, rather than search the ones that are relevant to the whole suspicious document. Therefore, one challenging problem for source retrieval lies in modeling the "partial matching" between documents, not the "entire relevance".

Addressing the partial matching in source retrieval, we propose PMCNN (Partial Matching Convolution Neural Network), a deep neural network architecture based on sequential convolution for source retrieval, shown in Fig. 1. In PMCNN, the sequential convolution operations are introduced to capture the local similarities of continuous text segments with different sizes to decide the candidate source documents.

We evaluate PMCNN on the PAN 2013 and PAN 2014 Plagiarism Source Retrieval Corpus [1], [2]. To established baselines, the experimental results demonstrate that PMCNN yields statistically significant improvements over the baselines.

2. Partial Matching Convolution Neural Network for Source Retrieval

PMCNN consists of three components: (1) an interaction matrix to represent the text segment interactions between two documents; (2) the sequential convolutions on the interaction matrix to obtain the partial plagiarism patterns; (3) a linear scoring function to decide the final candidate source documents.

2.1 Interaction Matrix

Given a suspicious documents d_{plg} and a document $d_{src} \in D$, for modeling the interactions between d_{plg} and d_{src}, we represent the input of their text segments as an interaction matrix M_0, with each element $x_{p,q}$ standing for the basic interaction, i.e. similarity between text segments s_p and s_q, shown in Eq. (1). Here for convenience, s_p is the p-th fixed-length text segment of d_{plg} and s_q is the q-th fixed-length text segment of d_{src}. s_p and s_q are all made up of t words. $\otimes$ stands for a general operator to obtain the similarity.

$$x_{p,q} = s_p \otimes s_q \tag{1}$$

In this paper, for simplicity, we adopt cosine similarity to compute the interaction score of s_p and s_q as follows:

Manuscript received December 18, 2020.
Manuscript publicized March 3, 2021.
[†]The authors are with the Foshan University, China.
a) E-mail: qihaoliang@fosu.edu.cn (Corresponding author)
DOI: 10.1587/transinf.2020EDL8162

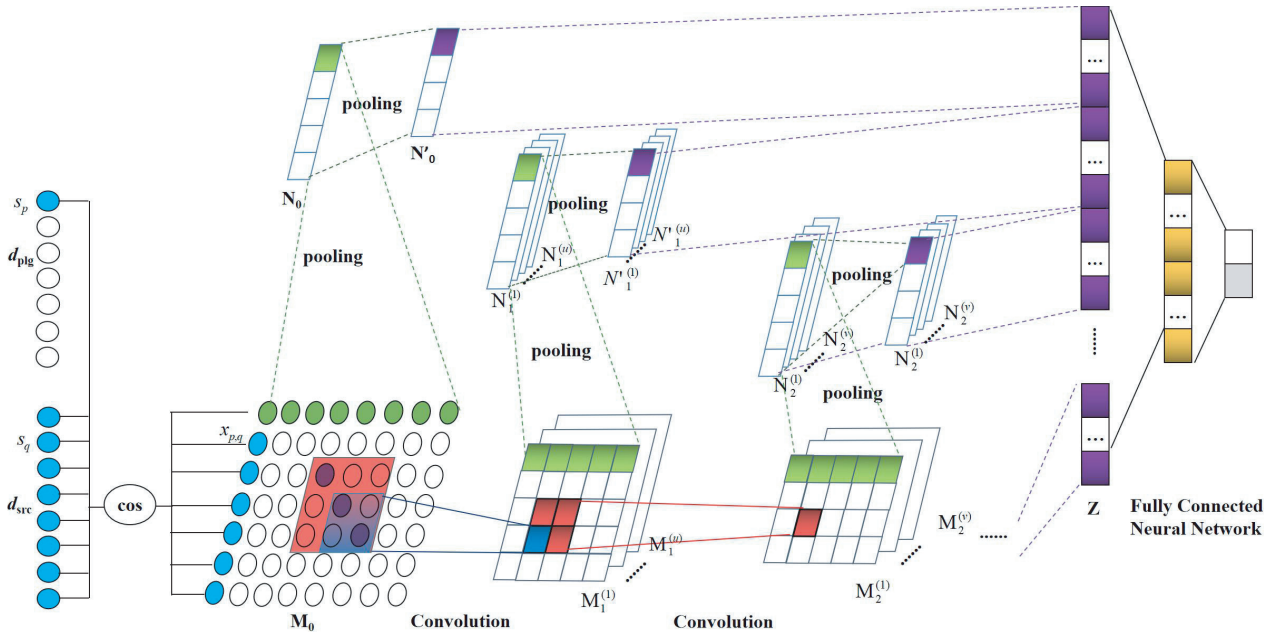

Fig. 1 Overview of partial matching convolution neural network for source retrieval of plagiarism detection

$$x_{p,q} = cosine(y_{s_p}, y_{s_q}) = \frac{{y_{s_p}}^T y_{s_q}}{||y_{s_p}||\,||y_{s_q}||} \tag{2}$$

where y_{s_p} and y_{s_q} are the vectors of s_p and s_q with tf-idf weighting, respectively.

Interaction matrix $M_0 \in \Re^{m \times n}$ has m rows and n columns. When $p > m$ or $q > n$, the interaction score for s_p and s_q is abandoned. If the number of text segments in d_{plg} or d_{src} is less than m or n, we set the corresponding cells to zero. t, m, and n are all the parameters to train.

2.2 Sequential Convolution

The body of PMCNN is a typical convolutional neural network, which is used to capture the partial plagiarism patterns of documents. Different from the research on using the convolutional neural network for extracting the text matching patterns [7], PMCNN designs a *sequential* convolution neural network structure to extract the plagiarism patterns of contiguous text segments.

As shown in Fig. 1, the u-th convolution kernel $w^{(1,u)}$ scans over the whole interaction matrix M_0 to generate a feature mapping matrix $M_1^{(u)}$, where u stands for the u-th convolution operation. For the feature $m_{i,j}^{(1,u)}$ on row i and column j in $M_1^{(u)}$, we define

$$m_{i,j}^{(1,u)} = \sigma\left(\sum_{s=0}^{r_{u-1}} \sum_{t=0}^{r_{u-1}} w_{s,t}^{(1,u)} \cdot m_{i+s,j+t}^{(0)} + b^{(1,u)}\right) \tag{3}$$

where r_u denotes the size of the u-th kernel and $m_{i,j}^{(0)}$ is the feature on row i and column j in M_0. In this paper, we use the square kernel with ReLU [8] as the active function σ. And the number of kernels for i-th layer convolution operation, denoted as $U^{(i)}$, is set as a parameter.

The way of sequential convolutions makes it possible to obtain the plagiarism patterns of text segments of various sizes. For example, if we use the 2×2 convolution kernel to scan over the interaction matrix M_0 to generate M_1, then each element $m_{i,j}^{(1,u)}$ in M_1 all maps a plagiarism feature of two adjacent text segments. Then, we use another 2×2 convolution kernel to scan over the feature mapping matrix M_1 to obtain the next feature mapping matrix M_2, then each element $m_{i,j}^{(2,u)}$ in M_2 all correspond to a further mapping on a block of 2×2 adjacent features in M_1. These 2×2 adjacent features in M_1 correspond to a block of 3×3 adjacent features in M_0. If we continue to perform the convolution operations, we can obtain the feature mapping of adjacent text segments with any size in M_0.

Based on the *sequential* convolutions, PMCNN model the partial plagiarism features to learn the plagiarism patterns of two documents.

2.3 Pooling

The sequential convolutions generate multiple feature mapping matrixes. Note that most of the segments in suspicious documents and source documents are not plagiarized. Hence, filtering out undesirable features is necessary. For this target, PMCNN utilizes the k-max pooling operations [9] to extract top k strongest partial matching features in the interaction matrix M_0 and the feature mapping matrix M_i.

Specifically for the first k-max pooling operation, each row of M_0 or M_i is scanned and the top k_1 values of each row are directly returned to form the vector N_i according to

the descending order. On N_i, we continue to perform the k-max pooling operation, and the top k_2 values of each N_i are returned to form a vector N_i'. Finally, these vectors are further concatenated to a single vector z.

2.4 Fully Connected Neural Network

Finally, we use a fully connected neural network to predict the score by aggregating partial plagiarism features filtered by the k-max pooling layers. Specifically, the feature vector z obtained by pooling is feed into a full connection hidden layer to obtain a higher-level representation. Then we use a linear transformation to the matching score:

$$(p_0, p_1)^T = \delta_2(\mathrm{W}_2\delta_1(\mathrm{W}_1 z + b_1) + b_2) \quad (4)$$

where p_0 and p_1 are the partial matching score of the corresponding class of plagiarism and non-plagiarism, z is the output of pooling, W_i stands for the weight of the i-th layer, b_i is the corresponding biases, and δ_1 and δ_2 represent the activation functions. ReLU activation is utilized for δ_1 and Softmax activation is applied for δ_2 to output the probability of belonging to each class.

2.5 Loss Function

We employ a discriminative training strategy with a cross-entropy loss function for training. During the training phase, model parameters of PMCNN are updated w.r.t. a cross-entropy loss between the predicted probabilities and the true answers:

$$Loss = -\frac{1}{N}\sum_{i=1}^{N}\left[y^{(i)}\log\left(p_1^{(i)}\right) + (1 - y^{(i)})\log\left(p_0^{(i)}\right)\right] \quad (5)$$

where $y^{(i)}$ is the label of the i-th training instance, N is the total number of training instances, p_k is defined in (4).

3. Experiment

The experiments are conducted on PAN 2013 [1] and PAN 2014 [2] Plagiarism Source Retrieval Corpus. Reference [10] gives a detailed description of the two datasets. Statistics for the experimental corpus is described in [11]. For the dataset of plagiarism source documents, we used ClueWeb09 (consists of 1,040,809,705 web pages).

There are three baselines in our experiments: WilliamsLDA [6], [12], RankingSVM [13] and AggLR [11]. $\mathrm{Williams}_{\mathrm{LDA}}$ got the highest F-score in the evaluation of PAN 2013 and PAN 2014. RankingSVM significantly outperforms $\mathrm{Williams}_{\mathrm{LDA}}$ by using a ranking model. AggLR also used a ranking-based method to obtain the source documents by addressing the aggregation of search results,

For the baseline methods, we followed the parameter settings described in the original work. For PMCNN, the parameters on the test corpus used those optimized on the training corpus. All the parameters were learned on the training data in terms of optimizing the *F-score*. For the segment size t, we set 30. For the size of the interaction matrix, we set $m = 200$ and $n = 500$. For the k-max pooling operations, we set $k_1 = 10$ and $k_2 = 20$. PMCNN is built using Keras[†], with the network parameters in Eq. (3) and Eq. (4) initialized to their default values. The optimization uses the backpropagation algorithm [14] with the ADAM update rule [15].

Table 1 Experimental results on PAN 2013 and PAN 2014

	PAN 2013 Test Corpus			PAN 2014 Test Corpus		
	F-score	Precision	Recall	F-score	Precision	Recall
$\mathrm{Williams}_{\mathrm{LDA}}$	0.4798	0.4834	0.5338	0.4745	0.5906	0.5068
RankingSVM	0.5152	0.6069	0.4936	0.5054	0.5881	0.5458
AggSR	0.5817	0.5723	**0.7475**	0.5280	**0.6009**	0.5663
$\mathrm{PMCNN}_{\mathrm{2CNN\text{-}2\times2}}$	**0.6171**$^{*\#\&}$	**0.8993**	0.4994	**0.5474**$^{*\#\&}$	0.5982	**0.6173**

Table 2 Performance comparison of different number of the sequential convolutions layers

	PAN 2013 Test Corpus			PAN 2014 Test Corpus		
	F-score	Precision	Recall	F-score	Precision	Recall
$\mathrm{PMCNN}_{\mathrm{0cnn}}$	0.5579	0.8860	0.4219	0.4946	0.5001	0.6406
$\mathrm{PMCNN}_{\mathrm{1cnn\text{-}2\times2}}$	0.5860	0.8943	0.4576	0.5234	0.5413	0.6395
$\mathrm{PMCNN}_{\mathrm{2cnn\text{-}2\times2}}$	**0.6171**	0.8993	**0.4994**	**0.5474**	**0.5982**	0.6173
$\mathrm{PMCNN}_{\mathrm{3cnn\text{-}2\times2}}$	0.5895	**0.9084**	0.4576	0.5184	0.5302	0.6313
$\mathrm{PMCNN}_{\mathrm{4cnn\text{-}2\times2}}$	0.5878	0.9006	0.4582	0.5297	0.5500	**0.6402**

For comparison, the processes of source retrieval in our model and the baselines follow the Williams et al. Method [6], [12]. Following PAN 2013 and PAN 2014, we adopt the measures *Precision*, *Recall*, and *F-score* to evaluate the performance of source retrieval. Followed the baseline methods, *F-score* is used as the main evaluation measure [1], [2].

Table 1 shows the experimental results, where our model is denoted as $\mathrm{PMCNN}_{\mathrm{2CNN\text{-}2\times2}}$, which means PMCNN uses 2 sequential convolution layers with 2×2 convolution kernels. The bold values represent the best results per category and the superscripts *, $^\#$, and $^\&$ indicate the validity of the models on $\mathit{Williams}_{\mathit{LDA}}$, *RankingSVM*, and AggSR using a one-sided paired t-test at the $p < 0.05$ level.

The experimental results indicate that the partial matching patterns captured by PMCNN can better model the source retrieval task, yielding significantly better *F-score* over the baselines.

Sequential convolution plays a decisive role in PMCNN. Table 2 compares the performance with the different number of sequential convolution layers. We also use subscripts to denote the number of convolution layers and the size of convolution kernels.

[†]https://keras.io

Table 3 Performance comparison of different sizes of convolution kernel

	PAN 2013 Test Corpus			PAN 2014 Test Corpus		
	F-score	Precision	Recall	F-score	Precision	Recall
$PMCNN_{2cnn\text{-}2\times2}$	**0.6171**	0.8993	**0.4994**	**0.5474**	**0.5982**	0.6173
$PMCNN_{2cnn\text{-}3\times3}$	0.5958	**0.9330**	0.4576	0.5106	0.5172	**0.6449**
$PMCNN_{2cnn\text{-}4\times4}$	0.5927	0.9132	0.4576	0.5214	0.5400	0.6359

Table 2 shows that the model with too many sequential convolution layers does not receive a performance boost. We analyze the reason remains "partial matching". Too many sequential convolution layers will capture the larger and longer text segments. However, there are not that longer plagiarism text segments between the suspicious document and source document. Under our text segment size setting (30 words one segment), 2 convolution layers with 2×2 convolution kernel are the most appropriate choice. The same is true for larger convolution kernels, shown in Table 3.

4. Conclusion

This paper has proposed the neural network architecture for source retrieval, the partial matching convolution neural network for source retrieval, denoted as PMCNN. It is a new way of modeling the task of source retrieval. Unlike existing models, we focus on the partial matching between two documents rather than classical query-document relevance. In PMCNN, a neural network based on the sequential convolution is designed to capture the partial matching between two documents. Experimental results on the PAN 2013 and the PAN 2014 Source Retrieval Corpus demonstrate that the proposed models can capture the similarities of the text with different sizes using the sequential convolutions and boost the performance of source retrieval.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No.61806075 and No.61772177) and the Natural Science Foundation of Heilongjiang Province (No. F2018029).

References

[1] M. Potthast, M. Hagen, T. Gollub, M. Tippmann, J. Kiesel, P. Rosso, E. Stamatatos, and B. Stein, "Overview of the 5th international competition on plagiarism detection," Proc. CLEF 2013 Evaluation Labs and Workshop, Valencia, Spain, pp.301–331, 2013.
[2] M. Potthast, M. Hagen, A. Beyer, M. Busse, M. Tippmann, P. Rosso, and B. Stein, "Overview of the 6th international competition on plagiarism detection," Proc. CLEF 2014 Evaluation Labs and Workshop, Sheffield, United Kingdom, pp.845–876, 2014.
[3] M. Hagen, M. Potthast, P. Adineh, E. Fatehifar, and B. Stein, "Source retrieval for web-scale text reuse detection," Proc. 2017 ACM on Conference on Information and Knowledge Management, Singapore, pp.2091–2094, 2017.
[4] J. Guo, Y. Fan, Q. Ai, and W.B. Croft, "A deep relevance matching model for ad-hoc retrieval," Proc. 25th ACM International on Conference on Information and Knowledge Management, pp.55–64, 2016.
[5] T. Liu, Learning to Rank for Information Retrieval, Springer, 2011.
[6] K. Williams, H.H. Chen, and C.L. Giles, "Supervised ranking for plagiarism source retrieval," Proc. CLEF 2014 Evaluation Labs and Workshop, Sheffield, United Kingdom, pp.1021–1026, 2014.
[7] L. Pang, Y. Lan, J. Guo, et al., "Text matching as image recognition," Proc. 30th AAAI Conference on Artificial Intelligence, Phoenix, USA, pp.2793–2799, 2016.
[8] G.E. Dahl, T.N. Sainath, and G.E. Hinton, "Improving deep neural networks for lvcsr using rectified linear units and dropout," IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp.8609–8613, 2013.
[9] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," Proc. 52nd Annual Meeting of the Association for Computational Linguistics (ACL), pp.655–665, 2014.
[10] M. Potthast, M. Hagen, M. Volske, and B. Stein, "Crowdsourcing interaction logs to understand text reuse from the web," Proc. 51st Annual Meeting of the Association for Computational Linguistics, ACM, Sofia, Bulgaria, pp.1212–1221, 2013.
[11] L. Kong, Z. Han, H. Qi, and M. Yang, "Source retrieval model focused on aggregation for plagiarism detection," Information Science, vol.503, pp.336–350, 2019.
[12] K. Williams, H.H. Chen, S.R. Choudhury, and C.L. Giles, "Unsupervised ranking for plagiarism source retrieval," Proc. CLEF 2013 Evaluation Labs and Workshop, Valencia, Spain, pp.1–8, 2013.
[13] L. Kong, Z. Lu, Z. Han, and H. Qi, "A ranking approach to source retrieval of plagiarism detection," IEICE Trans. Inf. & Syst., vol.E100-D, no.1, pp.203–205, 2017.
[14] D. Williams and G. Hinton, "Learning representations by back-propagating errors," Nature, vol.323, no.6088, pp.533–538, 1986.
[15] D.P. Kingma and B.J. Adam, "A method for stochastic optimization," Computer Science, vol.3, pp.1–13, 2015.