Continual Transfer Learning With Progress Prompt for Multi-Author Writing Style Analysis

Zhanhong Ye[1,†], Yutong Zhong[1], Chen Huang[2] and Leilei Kong[1,†]

[1]Foshan University, Foshan, Guangdong, China
[2]Zhongnan University of Economics and Law, Wuhan, Hubei, China

Abstract

This paper introduces a method utilizing forward knowledge transfer in continual learning to address the Multi-Author Writing Style Analysis 2024. The motivation is to transfer knowledge of varying difficulty levels to the current training task. Therefore, we employ the method of continual learning and forward knowledge transfer to train the model on task sequences composed of datasets with varying difficulty levels. This approach allows us to gradually transfer knowledge of different difficulties to the current training task. We then evaluated the Multi-Author Writing Style Analysis datasets provided by PAN. Finally, we selected model weights with the best validation set performance from each sequence. We achieved F1 scores of 0.993, 0.830, and 0.832 on each of the three difficulty levels of the test sets.

Keywords

PAN 2024, Multi-Author Writing Style Analysis 2024, continual learning, transfer learning

1. Introduction

Multi-author style identification involves determining whether the writing styles of two authors are consistent. Specifically, the style change detection task aims to determine whether the writing style changes between two consecutive paragraphs in a given multi-author document. Multi-Author Writing Style is extensively applied in plagiarism detection and author identification [1]. Furthermore, style change detection can aid in uncovering anonymous authorships, verifying claimed authorships, or developing new technologies for writing support.

Recent studies [2, 3] have employed the MTL (Multiple task learning) method, which involves solving multiple tasks jointly. However, one of the biggest challenges in MTL is to balance the convergence schedule across tasks. Differences in task difficulties can result in faster convergence on some tasks over others. As a result, when access to all datasets are available, three difficult datasets provided by PAN [4, 5] can be accessed simultaneously, it is sub-optimal to directly utilize the MTL method for mixing the data from the three difficulty levels of datasets together [2].

Progress prompts [6] differ from traditional MTL in that they transform multiple tasks into a sequential learning process. This approach effectively avoids the sub-optimal outcomes often associated with the simultaneous training of tasks in MTL. Hence Progress prompt methods are better solutions than simply adding together the losses of all tasks. Adding together the losses of all tasks is typically sub-optimal [2]. Especially, when each dataset is evaluated independently, rather than evaluating the performance of all datasets simultaneously.

Progress prompts [6] combine prompt tuning [7] with continual learning [8], retain a learnable soft prompt [9] for each incoming task, and sequentially concatenate it with previously trained soft prompts. The purpose of this approach is to facilitate forward knowledge transfer [10], focusing on learning multiple tasks sequentially rather than simultaneously.

In this paper, we leverage the progress prompts method mentioned in the study [7] to transfer knowledge

CLEF 2024: Conference and Labs of the Evaluation Forum, September 09–12, 2024, Grenoble, France

† corresponding author

chinwang.yip@gmail.com (Z. Ye); yutongz115@gmail.com (Y. Zhong); 1141460892@qq.com (C. Huang); kongleilei@fosu.edu.cn (L. Kong)

0009-0001-4094-006X (Z. Ye); 0009-0003-1694-9800 (Y. Zhong); 0009-0008-5942-0006 (C. Huang); 0000-0002-4636-3507 (L. Kong)

of varying difficulty levels from previous tasks to the current task using learnable soft prompts. Different from the MTL method, we employ the progress prompts method, which involves training a soft prompt for the current task and concatenating it with previously trained soft prompts. This allows for the transfer of knowledge across datasets of varying difficulty levels. Regarding the model architecture, the model has three parts. The first part involves soft prompt parameters that are combined with the parameters of the soft prompt for the current task and the parameters of the soft prompt for the previous task. The second part consists of the deberta-v3-base [11] model, which handles the current task. The third part is the classifier with classification loss.

2. Network Architecture

First, let T_i be the dataset, where $i \in 1..3$. T_i consists of a binary classification task for a style change. We convert the easy, medium, and hard difficulties datasets in the Multi-Author Writing Style Analysis task [4, 5] into binary classification tasks. This means that in any dataset T_i, the data input is called S_j, meaning the paragraph pair, with an output of 0 or 1. 0 indicates that the paragraph pair has no style change, while 1 indicates that the paragraph pair has a style change. We then form a sequence of tasks with easy, medium, and hard datasets, (T_1, T_2, T_3).
The goal is to utilize the DeBERTa-v3 model to sequentially implement this binary classification for a style change task on the task sequence. After training on T_i, we obtain the model's classification performance on T_i and then proceed to the next classification task. The core feature of the method is the progress prompts method, which involves learning a distinct soft prompt [9] P_i for each task T_i, $i \in 1..3$. Note that the soft prompt P_i has parameters provided by the embedding layer of the pre-trained language model. In addition, we not only learn a soft prompt P_i for each task T_i but also concatenate it with all previously trained soft prompts P_k; $k < i \leq 3$. According to the model shown in Figure, it consists of an encoder block, classification, and soft prompt parameters. The first is the encoder block. We use the deberta-v3 [11] model to encode the input, which consists of pairs of paragraphs from the current difficulty dataset. Next comes the classification part, where we use linear layers as classifiers to classify the encoded content, making it possible to complete the current downstream task. Then, concerning the soft prompt parameters, they are initialized using the parameters from the embedding layer of the deberta-v3 model. The details of the progress prompts are in section 2.1. Overall, the primary loss function $\mathcal{L}$ for training task T_i can be defined as follows.

$$\mathcal{L} = \mathcal{L}_{ce} \tag{1}$$

The loss $\mathcal{L}_{ce}$ means a cross-entropy loss to optimize the encoder block, classifier, and soft prompt parameters.

2.1. Progress prompt

Firstly, the PAN has provided three difficult datasets for Multi-Author Writing Style Analysis. Given a batch named B, which comes from the current training task T_i, the contents of B can be defined as $\{(S_1, y_1), (S_2, y_2) \ldots (S_j, y_j)\} \in B$, where S_j means the paragraph pair, and y_j is the corresponding label. Then we retain a learnable soft prompt P_i for each B and sequentially concatenate it with all previously learned soft prompts P_k; $k < i \leq 3$. The soft prompt is obtained through the embedding layer of the pre-trained language model. Specifically, we select the last m tokens from the pre-trained language model vocabulary V as pseudo tokens and then pass these pseudo tokens into the embedding layer of the pre-trained language model to obtain all soft prompts. Then, we combine P_i, P_k and $e(S_j)$ which is the current input embedding, sending them to the pre-trained model. The pre-trained model consists of the transformer [12] block, to obtain the corresponding hidden state $\mathcal{H}_j$. $e(S_j)$ represents the input S_j encoded by the embedding layer of the pre-trained language model.

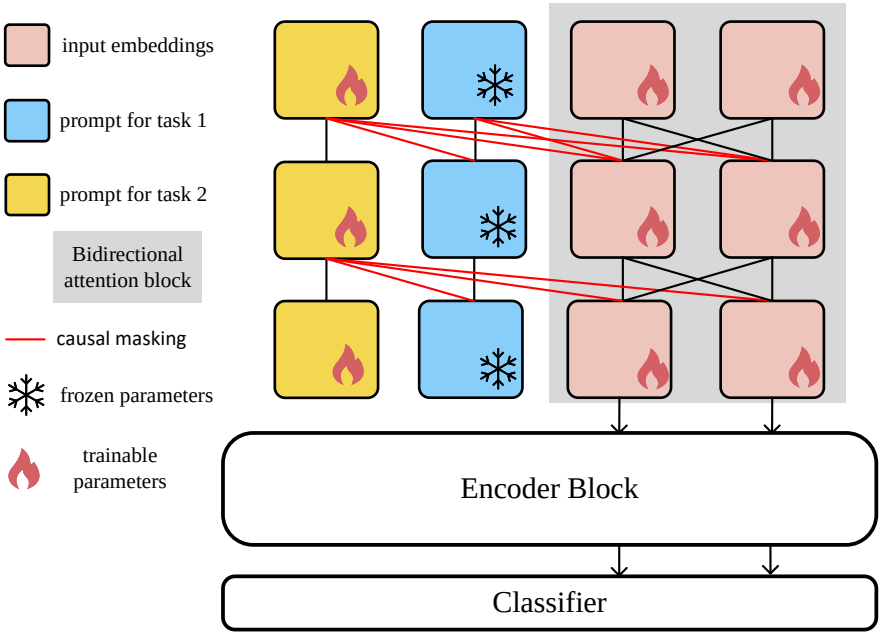

Figure 1: Model Architecture

$$\mathcal{H}_j = encoder(P_k, ..., P_1, S_j) \tag{2}$$

After obtaining the hidden state $\mathcal{H}_j$ we use a classifier to generate the soft labels for each category.

$$\varphi_j = \varphi(S_j) = (y_j^1, y_j^2) = (\frac{e^{(\phi(\mathcal{H}_j)^1)}}{\sum_{c=1}^{C} e^{(\phi(\mathcal{H}_j)^c)}}, \frac{e^{(\phi(\mathcal{H}_j)^2)}}{\sum_{c=1}^{C} e^{(\phi(H_j)^c)}}) \tag{3}$$

where $\varphi(\cdot)$ is the soft label of sample j, $\phi(\cdot)^c$ indicates the output of the linear layer for category c, and C represents the total number of categories. Then we calculate the cross-entropy loss for the classification

$$\mathcal{L}_{ce} = -\sum_{S_j, y_j \in T_i} logp(y_j | \varphi(S_j), \theta, \theta_{P_i}) \tag{4}$$

where θ refers to the model parameters of the encoder and classifier, and θ_{P_i} denotes the trainable parameters of the soft prompt for the i-th task in the embedding layer. By training with Equation (4), we obtain the final pre-trained language model M_i for the current task T_i. Since the PAN committee provides 3 datasets of varying difficulty for the Multi-Author Writing Style Analysis task, these datasets can be arranged in different combinations to form 6 task sequences with different orders. We will apply our proposed method to these 6 task sequences. Then, we will select and save the model weights that achieve the highest performance on the validation set for the easy, medium, and hard datasets from these 6 task sequences.

3. Experiments and Results

3.1. Data analysis

The PAN organizers have provided all data and the data is available in three difficulty levels: easy, medium, and hard. Each difficulty data set is divided into a training set, a validation set, and a test set. The distribution of each dataset is 70%, 15%, and 15%, respectively and the statistical analysis reveals that the token length of most entries is less than 512. Then we organize the data according to the method mentioned in section 2.1. In addition to this, when documents in the datasets of three difficulties are provided by only two authors (also given in the ground truth), it is possible to further analyze which author wrote each paragraph in the documents. Therefore, besides using consecutive pairs of paragraphs as paragraph pairs, we incorporate additional non-consecutive pairs of paragraphs into our paragraph pair set and assign them labels based on the inferred relationships between the authors. For example, if the same author is believed to have written both paragraphs, it is assumed that the style has not changed, and vice versa.

3.2. Experiment setting

In this work, the deberta-v3 base model is selected for classification. It concludes with 12 transformer encoder layers, its hidden size is 768. The three difficulty datasets are formed into six different task sequences, as Table 1 depicts. We train the model sequentially on datasets of varying task difficulties, following the given sequence. We set the early stopping to 10, the prompt length of 10 tokens for each difficulties dataset, and the learning rate to 5e-5, 3e-5, and 3e-5 for three datasets respectively. All experiments are conducted on NVIDIA A800 GPU with 80GB memory with a batch size of 64.

Table 1
Task sequences.

order	Task sequence
i	medium→hard→easy
ii	hard→medium→easy
iii	easy→hard→medium
iv	hard→easy→medium
v	easy→medium→hard
vi	medium→easy→hard

3.3. Results

We will conduct four experiments for validation datasets: the fine-tune method with deberta-v3, the best performance on the validation set, different datasets from all sequences with data augmentation and different datasets from all sequences including partial datasets with data augmentation or without augmentation. The results are presented in tables 2-5 respectively. We will then select the model weights that achieve the highest F1 scores on the validation sets corresponding to the difficulty levels across all sequences and submit these to the TIRA platform [13]. The final test set results are presented in Table 6.

Table 2
The fine-tune method with deberta-v3-base and our best performance with our methods on the validation set.

task	fine-tune	best validation set score with our methods
easy	96.9	99.6
medium	83.7	84.5
hard	83.4	84.1

Table 3
Different datasets from all sequences with data augmentation.

order	F1-score				
i	medium 83.9	→	hard 77.3	→	easy 97
ii	hard 76	→	medium 84.5	→	easy 96.7
iii	easy 98.3	→	hard 80.7	→	medium 82.7
iv	hard 75.2	→	easy 96	→	medium 83.4
v	easy 98.5	→	medium 82.8	→	hard 68.6
vi	medium 83.7	→	easy 97.7	→	hard 77.6

Table 4
Different datasets from all sequences without data augmentation.

order	F1-score				
i	medium 84.1	→	hard 83.4	→	easy 98.9
ii	hard 83.2	→	medium 82.9	→	easy 99.4
iii	easy 98.1	→	hard 81.4	→	medium 82.6
iv	hard 83.9	→	easy 99.6	→	medium 83.7
v	easy 97.5	→	medium 81.1	→	hard 82.2
vi	medium 83.8	→	easy 98.4	→	hard 83.7

Table 5
Different datasets from all sequences including partial datasets with data augmentation or without augmentation.
*with data augmentation

order	F1-score				
i	medium *83.3	→	hard 83	→	easy 98.4
ii	hard 83.2	→	medium *82.9	→	easy 99.4
iii	easy 97.3	→	hard 81.3	→	medium *82.5
iv	hard 83.3	→	easy 99.2	→	medium *83.7
v	easy 94.6	→	medium *80.6	→	hard 79.8
vi	medium *83.7	→	easy 98	→	hard 84.1

Table 6
Overview of the F1 accuracy for the multi-author writing style task in the test set. The word "combine" in the method refers to the performance of the test set obtained after adding the specified number of validation set data to the training set for training. The number added to the training set is the content after "combine-va".

Approach	Task 1	Task 2	Task 3
combine-va-1500-te-hard	—	—	0.828
combine-va-1500-te-medium	—	0.820	—
combine-va-1500-te-easy	0.988	—	—
combine-va-full-te-hard	—	—	0.826
combine-va-full-te-medium	—	0.825	—
combine-va-full-te-easy	0.989	—	—
combine-va-750-te-hard	—	—	0.817
combine-va-750-te-medium	—	0.824	—
combine-va-750-te-easy	0.993	—	—
hard-test-final-score	—	—	0.832
medium-test-final-score	—	0.830	—
easy-test-final-score	0.991	—	—
Baseline Predict 1	0.466	0.343	0.320
Baseline Predict 0	0.112	0.323	0.346

4. Conclusion

In this paper, we have completed the tasks set by PAN and have employed the progress prompt method to tackle the Multi-Author Writing Style Analysis task. Instead of using traditional MTL (Multi-Task Learning) techniques, we utilize the progress prompt method to transfer knowledge from datasets of varying difficulties to the current training dataset. The proposed method achieves scores of 0.993, 0.830, and 0.832 on three test datasets. These results validate the effectiveness of our proposed method in performing the Multi-Author Writing Style Analysis task.

Acknowledgments

This research was supported by the Natural Science Foundation of Guangdong Province, China (No.2022A1515011544)

References

[1] Z. Ye, C. Zhong, H. Qi, Y. Han, Supervised contrastive learning for multi-author writing style analysis, in: Conference and Labs of the Evaluation Forum, 2023.

[2] W. Liu, S. Rajagopalan, P. Nigam, J. Singh, X. Sun, Y. Xu, B. Zeng, T. Chilimbi, Asynchronous convergence in multi-task learning via knowledge distillation from converged tasks, in: Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track, 2022, pp. 149–159.

[3] A. Hashemi, W. Shi, Enhancing writing style change detection using transformer-based models and data augmentation, Working Notes of CLEF (2023).

[4] E. Zangerle, M. Mayerl, M. Potthast, B. Stein, Overview of the Multi-Author Writing Style Analysis Task at PAN 2024, in: G. Faggioli, N. Ferro, P. Galuščáková, A. G. S. de Herrera (Eds.), Working Notes of CLEF 2024 - Conference and Labs of the Evaluation Forum, CEUR-WS.org, 2024.

[5] J. Bevendorff, X. B. Casals, B. Chulvi, D. Dementieva, A. Elnagar, D. Freitag, M. Fröbe, D. Korenčić, M. Mayerl, A. Mukherjee, A. Panchenko, M. Potthast, F. Rangel, P. Rosso, A. Smirnova, E. Stamatatos, B. Stein, M. Taulé, D. Ustalov, M. Wiegmann, E. Zangerle, Overview of PAN 2024: Multi-Author Writing Style Analysis, Multilingual Text Detoxification, Oppositional Thinking

Analysis, and Generative AI Authorship Verification, in: Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Fourteenth International Conference of the CLEF Association (CLEF 2024), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2024.

[6] A. Razdaibiedina, Y. Mao, R. Hou, M. Khabsa, M. Lewis, A. Almahairi, Progressive prompts: Continual learning for language models, arXiv preprint arXiv:2301.12314 (2023).

[7] B. Lester, R. Al-Rfou, N. Constant, The power of scale for parameter-efficient prompt tuning, arXiv preprint arXiv:2104.08691 (2021).

[8] S. Thrun, Lifelong learning algorithms, in: Learning to learn, Springer, 1998, pp. 181–209.

[9] X. Liu, Y. Zheng, Z. Du, M. Ding, Y. Qian, Z. Yang, J. Tang, Gpt understands, too, AI Open (2023).

[10] Z. Ke, B. Liu, N. Ma, H. Xu, L. Shu, Achieving forgetting prevention and knowledge transfer in continual learning, Advances in Neural Information Processing Systems 34 (2021) 22443–22456.

[11] P. He, J. Gao, W. Chen, Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing, arXiv preprint arXiv:2111.09543 (2021).

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[13] M. Fröbe, M. Wiegmann, N. Kolyada, B. Grahm, T. Elstner, F. Loebe, M. Hagen, B. Stein, M. Potthast, Continuous Integration for Reproducible Shared Tasks with TIRA.io, in: J. Kamps, L. Goeuriot, F. Crestani, M. Maistro, H. Joho, B. Davis, C. Gurrin, U. Kruschwitz, A. Caputo (Eds.), Advances in Information Retrieval. 45th European Conference on IR Research (ECIR 2023), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2023, pp. 236–241. doi:10.1007/978-3-031-28241-6_20.