Linking Writing Processes to Writing Quality

Kiwuuwa Balikuddembe
University of Twente
k.j.balikuddembe@student.utwente.nl

Davide Berasi
University of Twente
d.berasi@student.utwente.nl

Samuel Coste
University of Twente
s.f.coste@student.utwente.nl

Marti Jimenez
University of Twente
m.jimenez@student.utwente.nl

Zerida Namuyomba
University of Twente
a.z.namuyomba@student.utwente.nl

ABSTRACT

When composing an essay, each individual goes trough a different process to obtain the final text. Studying how writing habits affect essay quality is an interesting and challenging problem. Keystroke logs offer insights into the writing process by recording timing and typing patterns. In our project, we propose a classification framework to predict writing quality directly from keystroke logs, without using information on the content of the final text. To do it, we combine a convolutional network with the vanilla transformer encoder. Thanks to its representational power, our model addresses the problem without the need of heavy feature engeneering.

keywords: Keystroke logs, sequence classification, transformer

1 INTRODUCTION

Defining the process of writing can be a rather confusing and ambiguous activity as no two people can approach writing in the same way. However, a lot of interest is directed towards the study of writing behaviors because, when writing a text, the production process could contain a lot of information on the quality of the final result.

Keystroke log data can provide us with an insight into how an individual planned and revised its work, as it demonstrates distinct pause patterns or strategic time allocation throughout the writing process [6]. For this reason, keystroke logging programs have been designed to observe writing processes on a computer. These programs keep track of the time stamps and keystroke activity to reconstruct and describe the text production process [3], [11].

In our project, we investigated if typing behavior affects the outcome of an essay. To do it, we came up with a solution to the competion "Linking Writing Processes to Writing Quality" hosted on Kaggle [1]. In this challenge, we were requested to develop a model capable of predicting the score of an essay from keystroke logs data in a supervised manner.

Most of the solutions to the competition tackle the problem by extracting a large amount of handcrafted features from the input sequence and then they train a classifier on the feature space. This approach leads to very good results but requires a lot effort for the design of the features. For this reason, we tackle the problem in a completely different way. In our solution, we combine a convolutional network with a vanilla transformer adapted for sequence classification. Thus, the research question that we aim to answer is: **Can a transformer-based model be effectively utilized for direct classification of keystroke sequence logs to predict writing quality, eliminating the need for heavy feature engineering?**

2 RELATED WORK

In this section, we will provide an overview of existing works that have been written on similar topics or methods. The aim is to position our study within the broader research landscape and show how it what way our approach is novel.

Research into writing processes is not something recent. In his 1972 dissertation [9], Charles K. Stallard concluded that "good student writers write slowly, taking time to read segments of their work at intervals during the writing process. During these hesitations to read, the good writers make numerous revisions. These are usually word choice revisions". This is one of numerous studies that have found patterns in the writing process and writing quality, but these kinds of studies were performed by watching students write in a controlled environment and conducting interviews.

Nowadays much deeper analysis can be conducted by using keylogger programs that reveal exactly what the writing process has been. Studying keylogs can provide a lot of insight in the writing process [5], [10], [4], but especially for longer essays this represents a lot of data that has to be processed. As mentioned by Stallard there are certain features in the writing process that discern good writers from the rest. As such, the most natural way to solve the problem consists of extracting a fixed number of features from the log data sequence and then using a classifier to predict the score from the feature representation. In this Kaggle competition all of the top entries adopted this method which requires a heavy work of feature engineering; for example in the submission [2] by M. Chan, 230 features are extracted from the logs data and an ensemble of six tree-based methods is used to reach a *RMSE* of 0.586.

Our approach differs from this by adopting techniques used in Automated Essay Scoring (AES) and applying them to the keylog data rather than the actual essay itself. The most recent studies on AES [7], [8] have implemented a combination of convolutional neural networks and transformer-based approaches. To our knowledge, such models have never been used for the specific problem of writing quality prediction from keylog data.

3 DATASET

To train the model, Kaggle provides a large dataset of keystroke logs that have captured writing process features. The dataset comprises around 2500 sequences of events registered during the composition of an essay. The sequences have an average length of about 5000 elements. For each event (keystrokes or mouse click), the start time, end time and some categorical features are reported.

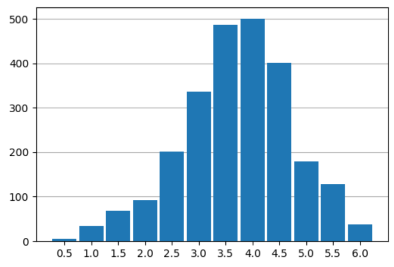

Figure 1: Scores distribution in the dataset.

To prevent reproduction of the essay text, all alphanumeric character inputs have been replaced with the anonymous character "q". Punctuation and other special characters have not been anonymised. Each essay was scored on a scale from 0 to 6 with half grades (that is, there are 13 possible scores). For this reason, the prediction task can be considered as a classification problem. The dataset has two properties that make the challenge particularly difficult:

- the dataset is highly unbalanced, in the sense that the distribution of the scores is far from being uniform (*Fig.* 1).
- the sequences of logs data of different users have different lengths;

4 METHODOLOGY

In this section we present the methodology we adopted in our work. We first describe the data-cleaning techniques employed to prepare the dataset, then we describe the architecture of the proposed framework and, finally, we define the evaluation metric used to measure the performance of the model.

4.1 Data Pre-Processing

The preprocessing steps are performed to prepare the keystroke logs data for model training. The following steps outline the data processing pipeline:

4.1.1 Action Time Processing. The action time values in the keystroke logs are winsorized to mitigate the effect of outliers. Winsorization limits the extreme values to a specified percentile range. For this process, the action time column is winsorized with a lower limit of 5% and an upper limit of 95%. Outliers beyond these limits are identified and flagged using binary columns. Finally, the winsorized action time values are scaled using Min-Max scaling to ensure uniformity across the dataset.

4.1.2 Event Processing. The distribution of down events in the keystroke logs is analyzed to understand the frequency of different events. Based on the distribution, a subset of events is selected for one-hot encoding. The selected events include alphanumeric characters ('q'), space, backspace, shift, arrow keys, mouse clicks, and punctuation marks. Other events are categorized as 'Other' for encoding. One-hot encoding is performed on the selected events to represent them as binary features.

4.1.3 Cursor Position Processing. The cursor position values are normalized using Min-Max scaling to bring them within a standardized range. Min-Max scaling ensures that all cursor position values lie between 0 and 1, preserving the relative differences between them.

4.1.4 Text Change Processing. Similar to event processing, the distribution of text change events is analyzed to identify the most frequent events. The selected events include alphanumeric characters ("q"), space, and 'NoChange' indicating no text modification. Other events are categorized as 'Other' for encoding. One-hot encoding is applied to represent these events as binary features.

4.1.5 Activity Processing. The distribution of activity types in the keystroke logs is examined to identify predominant activities. The selected activity types include input, remove/cut, and non-production activities. Other activities are categorized as 'Other' for encoding. One-hot encoding is utilized to represent these activities as binary features.

4.1.6 Word Count Processing. Word count values are normalized using Min-Max scaling to ensure consistency across different essays.

4.2 Model architecture

The transformer encoder can be adapted for the classification task by adding a pooling layer followed by a linear layer. The role of the pooling layer is to remove dependency on the input sequence length as the inputs to the linear classifier needs to have fixed dimension. We refer to our model as the ConvFormer as we also use some convolution layers to reduce the size of the input sequences. Figure 2 summarises the proposed architecture.

4.2.1 Convolution layers. Since the attention mechanism has quadratic complexity, feeding the input sequences directly to the encoder would require large computational resources. We solve the problem by adding a convolutional block before the encoder. Its role is to reduce the input size while extracting local information. Specifically, we use 5 convolutional layers with kernel size 5 and stride 2. In this way, if the input sequence has length n, the output of the convolution block is a sequence of length $\approx n/32$.

4.2.2 Block-wise Average Pooling. A natural choice to pool the encoded sequence is to use a Global Average Pooling layer. However, important temporal dependencies would be lost with global aggregation. For this reason, we propose a Block-wise Average Pooling layer (BAP). This layer, partitions the input sequence in a fixed number b of blocks and computes the block-wise average. In this way, the dependency from the input sequence length is removed without losing too much relevant information. In our implementation we used $b = 8$.

4.2.3 Prediction. From the flattened pooled sequence, the logits for the score classes are computed with a linear layer. These values are normalised with a softmax to obtain the confidence probabilities. The prediction is the expected score given these probabilities.

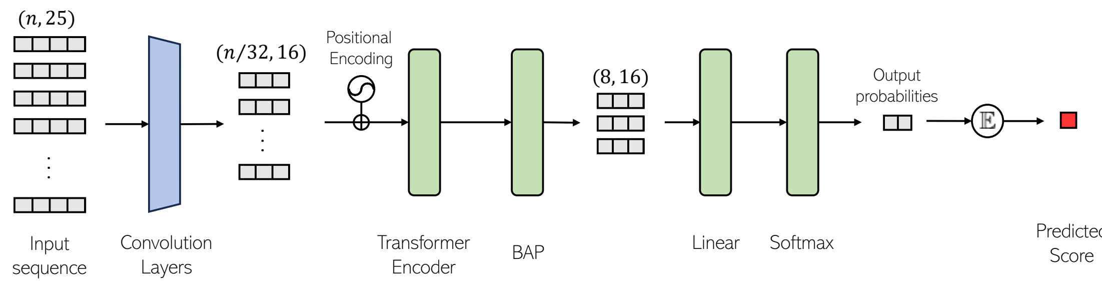

Figure 2: ConvFormer architecture.

4.3 Baseline

We implemented a simple baseline framework as a point of reference to compare the performance of the proposed architecture. This framework consists of representing the input sequence with 91 handcrafted features and then classifying the feature vector with a multi-layer perceptron.

4.4 Evaluation

As an evaluation metric, the Root Mean Squared Error is used, that is:

$$RMSE = \left(\frac{1}{n} \sum_{i=0}^{n} (y_i - \hat{y}_i)^2 \right)^{1/2}$$

where $\hat{y}_i$ is the predicted value and y_i is the original value for each instance i over n total instances.

5 RESULTS & DISCUSSION

Testing our proposed model we achieved an RMSE of 0.609, which is a significant improvement over the baseline framework which achieved a RMSE of 0.718. However, the top-score submissions to the Kaggle challenge achieve much better results. For example, the third place submission (the top score with information available) obtains an RMSE of 0.571. To achieve such a good score, the authors represent the input sequence with 1339 handcrafted features and they classify the feature vectors with an ensemble of 8 different models. In Table 1, we refer to this model as Kaggle3.

Model	RMSE	handcrafted features
baseline	0.718	91
ConvFormer	0.609	0
Kaggle3	0.571	1339

Table 1: RMSE and number of handcrafted features of the considered models.

To visualise our model's prediction accuracy, we plot the ground truth and predicted value of each test sample as a point on the xy-plane (Figure 3). The proximity of the point to the red line represents the accuracy of the prediction. It is easy to notice that the predicted values are all concentrated in the band between 2 and 5. This is a consequence of the high imbalance of the dataset. On the Kaggle competition discussion page, many participants argued that trying to solve the imbalance problem with techniques like up-sampling or down-sampling has not demonstrably improved model performance.

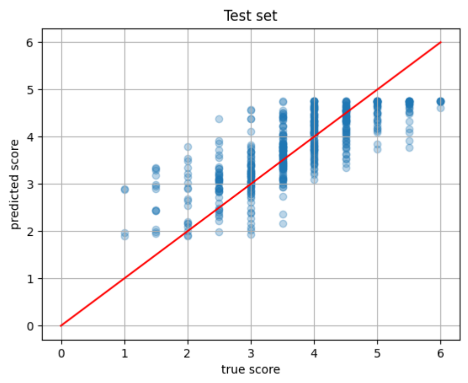

Figure 3: Predictions on the test set.

6 CONCLUSION

In this study, we explored the use of a transformer-based model to predict writing quality from keystroke log data. This approach represents a departure from the conventional methods that rely heavily on manual feature extraction.

Our model achieved a Root Mean Square Error of 0.609, which suggests that the transformer model can effectively interpret keystroke sequences for predicting writing quality, reducing the need for complex feature engineering. However, it is important to note that while our model shows promise, it still trails behind the top scores in the Kaggle competition that used extensive handcrafted features.

Overall, our research suggests that transformer models have potential applications in analyzing and evaluating writing processes. Future research could build upon our work by using a more balanced dataset, refining our model framework further and exploring their applications in educational settings for better understanding and enhancing writing skills.

REFERENCES

[1] [n. d.]. Linking Writing Processes to Writing Quality. https://www.kaggle.com/competitions/linking-writing-processes-to-writing-quality. Accessed: 2023-12-05.

[2] [n. d.]. Public submission by Marcus Chan. https://www.kaggle.com/code/mcpenguin/writing-processes-baseline-v2-lgbm-nn. Accessed: 2023-12-05.

[3] Muhammad M Abdel Latif. 2008. A STATE-OF-THE-ART REVIEW OF THE REAL-TIME COMPUTER-AIDED STUDY OF THE WRITING PROCESS. *International Journal of English Studies* 8 (06 2008), 29–50. https://doi.org/ijes/article/view/49081

[4] Tao Gong, Mo Zhang, and Chen Li. 2022. Association of keyboarding fluency and writing performance in online-delivered assessment. *Assessing Writing* 51 (2022), 100575. https://doi.org/10.1016/j.asw.2021.100575

[5] Hongwen Guo, Paul D. Deane, Peter W. van Rijn, Mo Zhang, and Randy E. Bennett. 2018. Modeling Basic Writing Processes From Keystroke Logs. *Journal of Educational Measurement* 55, 2 (June 2018). https://doi.org/10.1111/jedm.12172

[6] Van Waes L Leijten M. 2013. Keystroke Logging in Writing Research: Using Inputlog to Analyze and Visualize Writing Processes. *Written Communication* 30 (06 2013), 358–392. https://doi.org/doi.org/10.1177/0741088313491692

[7] S. Ludwig, C. Mayer, C. Hansen, K. Eilers, and S. Brandt. 2021. Automated Essay Scoring Using Transformer Models. *Psych* 3, 4 (2021), 897–915. https://doi.org/10.3390/psych3040056

[8] Angad Sethi and Kavinder Singh. 2022. Natural Language Processing based Automated Essay Scoring with Parameter-Efficient Transformer Approach. In *2022 6th International Conference on Computing Methodologies and Communication (ICCMC)*. 749–756. https://doi.org/10.1109/ICCMC53470.2022.9753760

[9] Jr. Stallard, Charles Kelson. 1972. An Analysis of the Writing Behavior of Good Student Writers. *University of Virginia* 7233385 (July 1972).

[10] M. Talebinamvar and F. Zarrabi. 2022. Clustering Students' Writing Behaviors Using Keystroke Logging: A Learning Analytic Approach in EFL Writing. *Language Testing in Asia* 12 (2022). Issue 6. https://doi.org/10.1186/s40468-021-00150-5

[11] Luuk Van Waes, Mariëlle Leijten, sa Wengelin, and Eva Lindgren. 2012. Logging tools to study digital writing processes. In *Past, present, and future contributions of cognitive writing research to cognitive psychology*. Psychology Press, 507–533.