Beyond Final Products: Multi-Dimensional Essay Scoring Using Keystroke Logs and Deep Learning

Xinyun He
University of Science and Technology of China
Hefei, China
hxy1520726975@mail.ustc.edu.cn

Qi Shu
University of Science and Technology of China
Hefei, China
shuqi0314@mail.ustc.edu.cn

Mo Zhang
Educational Testing Service
Princeton, NJ, USA
mzhang@ets.org

Wei Huang
National Education Examinations Authority
Beijing, China
huangw@mail.neea.edu.cn

Han Zhao
University of Science and Technology of China
Hefei, China
ximang@mail.ustc.edu.cn

Mengxiao Zhu*
University of Science and Technology of China, Anhui Province Key Laboratory of Science Education and Communication
Hefei, China
mxzhu@ustc.edu.cn

Abstract

Essay assessment plays a crucial role in evaluating students' abilities in logical reasoning, critical thinking, and creativity. However, traditional manual scoring methods often suffer from inefficiencies due to fatigue, bias, and emotional factors, compromising objectivity. Automated Essay Scoring (AES) systems offer a more efficient and impartial alternative, yet most existing systems focus primarily on evaluating the final written product, overlooking the valuable data captured during the writing process. To address this issue, we introduce an innovative model called KAES, which explores the potential of integrating writing process data to enhance AES performance. The KAES model leverages a variety of data sources, including text content, prompts, keystroke dynamics, and manually extracted features, to extract meaningful insights. It employs a multi-task learning approach to evaluate essays across both linguistic and argumentative dimensions. Extensive experiments on the real-world CBAL dataset demonstrate that the KAES model significantly outperforms traditional baseline models, highlighting the effectiveness of incorporating writing process data into AES tasks.

CCS Concepts

• **Applied computing** → *Computer-assisted instruction*; • **Computing methodologies** → *Natural language processing*.

Keywords

Automated Essay Scoring, Keystroke Logs, Deep learning, Feature Engineering, Multi-Task Learning

*Corresponding author

LAK 2025, March 03–07, 2025, Dublin, Ireland

ACM ISBN 979-8-4007-0701-8/25/03
https://doi.org/10.1145/3706468.3706548

ACM Reference Format:
Xinyun He, Qi Shu, Mo Zhang, Wei Huang, Han Zhao, and Mengxiao Zhu. 2025. Beyond Final Products: Multi-Dimensional Essay Scoring Using Keystroke Logs and Deep Learning. In *LAK25: The 15th International Learning Analytics and Knowledge Conference (LAK 2025), March 03–07, 2025, Dublin, Ireland*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3706468.3706548

1 INTRODUCTION

The evaluation of student learning outcomes is a fundamental aspect of the educational process. The assessment of written essays represents a crucial methodology for the measurement of students' abilities to think logically, critically, and creatively [1, 16, 35]. Nevertheless, traditional manual grading approaches are inherently constrained. When confronted with a substantial corpus of essays, graders frequently demonstrate inconsistencies due to fatigue, subjective biases, and emotional factors. This results in time-consuming and inefficient grading processes that compromise accuracy and fairness[18, 21, 37].

In response to the growing demand for more efficient and objective grading methods, automated essay scoring (AES) systems have been developed. These systems employ computational analysis of essay content, structure, and language features to assign scores to essays written in response to specific prompts[21]. The available evidence indicates that AES systems can enhance the efficiency and accuracy of scoring, reduce the workload on human raters, and provide students with immediate feedback, which in turn can facilitate the development of their writing skills[14]. As technology advances, a variety of AES systems have been designed to improve both holistic and trait-specific scoring for essays in prompt-specific or cross-prompt settings[10, 36].

Despite ongoing advancements in AES technology, current research reveals certain shortcomings. The majority of existing systems rely exclusively on the final written product for scoring predictions, thereby failing to recognize the potential value of data pertaining to the writing process. In practical educational applications, these systems typically provide feedback on the final written

product, such as grammar and rhetoric, without supporting analysis of the writing process, including editing and revision behaviors. Consequently, they are unable to facilitate improvements in students' writing habits and strategies. To address this limitation, we consider the advantages of keystroke logging, which provides fine-grained insights into the writer's process[22], and aim to explore keystroke features within students' keystroke data and incorporate them into the AES model.

In our study, we propose a novel hybrid model named KAES, which incorporates writing process features from keystroke logs into a deep neural network to enhance AES performance and provide students with more detailed feedback. Given that keystroke logs capture a comprehensive record of the writing process and contain valuable insights[4, 32], their integration holds great potential for improving the accuracy of essay evaluation. The KAES model leverages a state-of-the-art hybrid framework, combining the strengths of feature engineering and deep neural networks. This enables a thorough exploration of the critical role of writing process data in AES systems from multiple modeling perspectives. Specifically, we first design and extract writing product and process features from students' keystroke logs according to feature engineering methods. Next, we perform embedding representations on multiple data sources (including keystroke sequences, text content, and prompts) and utilize deep learning methods such as convolutional neural networks, LSTM networks, and attention mechanisms to generate representations crucial for essay scoring, including keystroke sequence representations, text representations, and prompt representations. Finally, we fuse these representations with manually designed writing product and process features and employ a trait attention mechanism along with a multi-task architecture to simultaneously predict scores on language and argumentation traits. To validate the KAES model and assess the importance of writing process data in essay scoring, we conduct extensive experiments on the real-world CBAL dataset. The results indicate that our proposed KAES model significantly outperforms various strong baseline methods. Compared to the most effective baseline model[10], KAES improves the average Quadratic Weighted Kappa (QWK) scores in the language and argumentation dimensions by 4.54% and 2.68%, respectively. Meanwhile, these results demonstrate the immense value of writing process data in AES systems and confirm that the KAES model effectively utilizes it.

In summary, the contributions of this paper are as follows:

- To the best of our knowledge, this is the first study to apply writing process data to AES tasks using deep neural networks. This approach enhances the accuracy and interpretability of automated essay scoring and provides new insights for future research in essay scoring.
- We propose a novel automated essay scoring model named KAES, which employs a widely recognized hybrid architecture. Through the integration of feature engineering and deep learning sequence modeling methods, the model thoroughly explores the valuable potential of writing process data, highlighting its key role in AES tasks.
- The findings presented in this study are supported by extensive experiments on the real-world CABL dataset, which show that the proposed method significantly outperforms various strong baseline methods. Furthermore, KAES model code[1] has been released to advance future research in this domain.

2 RELATED WORK

2.1 Automated Essay Scoring

To address the inefficiencies and subjectivity inherent in manual essay scoring, Page et al.[25, 26] pioneered the development of the Project Essay Grader (PEG) system, which serves as the foundation for subsequent research into AES systems. Compared to manual grading, AES systems can efficiently process a large number of essays in a relatively short period and reduce the potential for subjective bias. For over a decade, the Educational Testing Service (ETS) has advanced AES research and developed the E-Rater engine, which evaluates student essays by providing detailed feedback on aspects such as grammar, structure, word usage and complexity, style, and organization. The E-Rater system has been widely adopted in the Test of English as a Foreign Language (TOEFL), Graduate Record Examination (GRE), and other online writing assessments[3, 29]. Notably, substantial evidence exists to support the accuracy of AES systems, showing high consistency with human raters[13].

Over the past several decades, an increasing number of AES systems have been developed, which can generally be categorized into three main types:

Manually Crafted Feature-Based Systems. These systems utilize manually designed discrete features and frequently employ techniques such as natural language processing, Bayesian networks, and latent semantic analysis[24, 28, 39]. Feature engineering models frequently exhibit a high degree of alignment with human raters, a consequence of their reliance on meticulously selected features. However, these systems can also be characterized by a lack of robustness and a tendency to be time-consuming[18].

Automatic Feature Extraction Systems. These systems utilize neural networks and deep learning techniques, such as CNNs, RNNs, and transformer networks[2, 23, 33]. DNN-AES models do not require manual feature design and can automatically learn features from data. For example, Dong et al.[12] have developed a hierarchical sentence-to-document model that represents essays by combining RNNs and CNNs. This model employs an attention mechanism to determine the relative importance of words and sentences and has demonstrated competitive accuracy[11].

Hybrid Models. These models integrate manual features with deep neural networks. Ridley et al. (2020) [30] designed the PAES model, incorporating essay quality features such as length, readability, textual complexity, textual variation, and sentiment. This model emphasizes general essay quality features, which are not prompt-specific, to address semantic variations in cross-prompt scoring settings. A year later, they proposed the leading model for cross-prompt essay trait scoring (CTS)[31], extending Dong et al.'s HiATT model by introducing multiple trait-specific layers and concatenated features. Building on this work, Do et al.[10] proposed a new architecture to enhance CTS by considering essay topic coherence. Their model, the prompt and trait relation-aware cross-prompt essay trait scorer (ProTACT), achieved the best results across all prompts and features in the ASAP and ASAP++

[1] https://anonymous.4open.science/r/KAES-328E/

datasets. In this paper, we adopt a similar hybrid model framework to enhance performance in essay scoring predictions.

However, as our research reveals, much like manual scoring, most AES studies focus solely on predicting scores based on the final written product, whether for holistic or trait-specific scoring. Consequently, these systems generally provide feedback on issues related to the final context of the essay, such as grammar and phrasing, but do not support the analysis of the writing process, including editing and revision behaviors. To effectively address these shortcomings, we propose incorporating students' keystroke data into AES models as a viable solution.

2.2 Keystroke Logging

Keystroke logging is a technique used to capture data entered by a computer user via the keyboard. With advances in computer technology and the increasing prevalence of online assessments, this method has garnered growing attention. Keystroke logging automatically collects data in a minimally invasive and scalable manner, without disrupting the writer's workflow. Moreover, it provides real-time, fine-grained insights into fundamental aspects of writing activities, such as writing speed, pauses, and edits[22].

Keystroke logs typically record the following detailed keyboard data[43]:

- **Timestamp**: Each keystroke event is marked with a timestamp, recording the exact time the event occurred.
- **Type**: The type of key action is recorded, including deletion, insertion, and content replacement.
- **Position**: The location in the text where the content is generated or modified is tracked.
- **Content**: The actual content generated or deleted, such as letters, numbers, and punctuation marks, is logged.

Keystroke logs provide a comprehensive record of students' writing processes, capturing valuable data with significant potential for enhancing AES models and offering more detailed feedback. Previous research has shown that writing process aspects, such as total writing time, pre-writing pause duration, text burst length, and the extent of text editing and revisions, correlate with the quality of the final written product[4, 7, 32]. Metrics like DIRatio (the ratio of deletion to insertion operations) and IKI (inter-key interval) can be analyzed to better understand writing behaviors and assist students in identifying their strengths and weaknesses[40, 42], particularly addressing the needs of lower-performing groups. Feedback on keystroke features enhances learners' awareness of effective writing strategies and practices, fostering improvements in their overall writing proficiency throughout the process.

3 PRELIMINARY

3.1 Dataset

We use the dataset from a scenario-based assessment (SBA)[6] of writing, collected as part of a larger data collection of middle-school students participating in the 2013 CBAL Writing Study[38]. The dataset was designed using SBA principles[9], and scenarios and source reading materials on a uniform theme were used throughout the assessment. In this study, we consider six persuasive and argumentative writing tasks, totaling 4,309 essays. Each essay was scored by a human rater on two dimensions, using detailed scoring criteria for each dimension:

- **Language:** The first scoring dimension is based on discourse-level features in the multi-paragraph text-grammar, word usage, spelling, syntactic variety, coherence, structure, etc.
- **Argumentation:** The second scoring dimension is based on genre-specific skills for writing argumentative/persuasive texts-quality of reasoning, mastery of argument structure, etc.

Both dimensions were scored on a scale from 0 to 5. For training and evaluating deep learning models, we divide the dataset into training, validation, and testing sets in a ratio of approximately 8:1:1. Detailed statistical information about the dataset is provided in Table 1.

3.2 Feature Engineering

Features extracted from the CBAL dataset are categorized into two main types based on their source and construction method: (a)***Writing Product Features*** derived from the final written product and (b)***Writing Process Features*** generated from keystroke logging data.

3.2.1 Writing Product Features. Writing product features are derived from the final output of the writing process, offering insights into the static characteristics of the completed text. These features capture aspects of writing quality, structure, and linguistic style, providing a retrospective view of the writing process. The writing product features used in this study are adapted from the system developed by Ridley et al.[30] and can be classified into five categories:

- **Length-based features**: These include metrics such as document length, sentence length, number of paragraphs, and word count, capturing the basic structure of the text.
- **Readability features**: These mainly include vocabulary complexity, sentence structure, paragraph length, and grammatical accuracy, all of which collectively affect the ease of understanding for readers.
- **Text complexity features**: These include measures of lexical diversity, the use of complex words (e.g., polysyllabic or rare words), and syntactic complexity.
- **Text variation features**: These assess diversity in the text, such as lexical variation, word frequency distribution, and synonym usage, reflecting the richness of expression.
- **Sentiment features**: These features analyze the sentiment of the text—positive, negative, or neutral—based on sentiment lexicons or classifiers, using NLP techniques.

3.2.2 Writing Process Features. Writing process features are derived from the dynamic behaviors exhibited during the writing process, capturing real-time changes that reflect strategies, cognitive load, time management, and other relevant factors. These features are generated by monitoring and recording writing activities, thereby revealing temporal dependencies and behavioral patterns. In this study, the writing process features include:

- **Keystroke records**: Logs of all keystrokes made during the writing process.

Prompt	1	2	3	4	5	6
Topic	Culture Fair	Generous Gift	Service Earning	Ban Ads	Cash For Grades	Social Networking
Num of Essays	761	745	502	560	877	864

Table 1: The statistics of the dataset.

- **Operation num**: The number of operations, such as insertions, deletions, copying, pasting, jumping, and replacing, reflecting the writer's revision activity and content refinement.
- **DIRatio**[42]: The ratio of deletion to insertion operations, indicating the frequency of revisions.
- **IKI**[41]: The average time between consecutive keystrokes, reflecting cognitive pauses during writing.
- **Keyboarding state**: The keystrokes are categorized into six states: InWord, BetweenWord, BackSpace, Edit, BetweenSentence, and BetweenParagraph. Frequency reveals the writer's habits and text-handling behavior.
- **Preparation time**: The time spent before starting to write, from the planning phase to the first keystroke.
- **Writing duration**: The total time from the start to the completion of writing, reflecting the writer's speed and mastery of the task.
- **Efficiency**: The ratio of total operations to writing time, indicating the overall efficiency of the writing process.

3.2.3 Feature Analysis. To assess the effectiveness of these features in predicting scores, we use the LightGBM model for feature selection and ranking. LightGBM[20] is a highly efficient gradient-boosting framework, well-suited for large datasets and numerous features. Feature importance is analyzed through the following steps:

Data Preprocessing. Before model training, the data is cleaned and normalized. Missing values are imputed (e.g., mean imputation for numerical features), and outliers are addressed. Min-max normalization is applied to scale features within the $[0, 1]$ range, though LightGBM is relatively insensitive to feature scaling.

Model Training. We use two dimensions of writing scores (Language and Argumentation) as target variables, with the writing product and process features serving as input. The dataset is divided into a training set (80%) and a test set (20%), and five-fold cross-validation is applied to prevent overfitting. Hyperparameters are tuned via grid search, and performance is evaluated using Root Mean Squared Error (RMSE) and QWK.

Feature Importance Evaluation. To assess the importance of each feature in predicting writing scores, we use LightGBM's built-in feature importance mechanism. Feature importance in LightGBM is calculated by counting how often a feature is used to split data across all tree models and weighting this by the improvement in model accuracy that each split brings. This provides a ranked list of features according to their contribution to the overall model performance. We generate bar charts of feature importance (see Figure 1), offering a visual reference for the contributions of all features. Due to space constraints, we only present the results for the top ten features ranked by average importance.

Feature Selection. Based on feature importance scores, further feature selection is conducted to retain the most relevant features and enhance model interpretability. Features with low importance are removed, with the selection threshold ensuring that the retained features account for approximately 90% of the total importance.

3.3 Problem Setup

In cross-prompt AES, training samples come from non-target prompts (source domain $E_S = \{x_S^i,\ Y_S^i\}_{i=1}^{N_S}$), with N_S representing the number of essays in E_S. During testing, essays belong to the target prompt (target domain $E_T = \{x_T^i,\ Y_T^i\}_{i=1}^{N_T}$), with N_T denoting the number of essays in E_T. The goal is to predict M different dimension scores for each essay i , represented as a set $Y_i = \{y_1^i,\ y_2^i,\ ...,\ y_M^i\}$.The AES tasks in this paper are formally defined as follows.

The input consists of:

- **Essay**: The complete text written by the student.
- **Prompt**: The topic and writing requirements of the essay.
- **Manual Features**: Manually constructed features from the CBAL dataset, including both writing product and process features.
- **Keystroke sequence**: The operational log recorded during the writing process.

The output consists of two score dimensions:

- **Language score**: Evaluates grammar, vocabulary usage, and other linguistic features.
- **Argumentation score**: Assesses the quality of reasoning and the structure of arguments in the essay.

4 METHOD

Based on our previous analysis, we propose a novel model named KAES, whose architecture is illustrated in Figure 2. The KAES model is built upon ProTACT[10], which is one of the most advanced methods in cross-prompt AES systems. Our approach has been streamlined into a one-step process, detailed as follows:

4.1 Keystroke Sequence Modeling

Numerous studies have shown that information about the writing process can be used to improve the accuracy of essay quality predictions[8, 15, 32]. Therefore, we encode the keystroke sequence for each essay and direct the model's attention to the data. First, we extract the operation sequence, $\{k_1, k_2, \ldots, k_n\}$, from the keystroke log, where k_i represents the i-th keystroke operation (including key presses, deletions, cursor movements, etc.). This sequence reflects user behavior during the writing process, which is likely correlated with essay quality. Subsequently, we use an embedding matrix $\mathbf{E}^K$ to convert each keystroke operation k_i into an embedding vector,

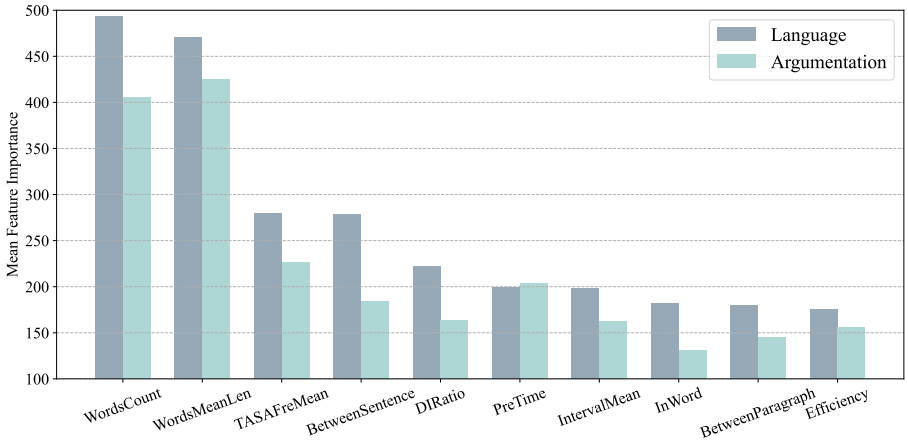


Figure 1: Top ten handcrafted features by average importance across all prompts.

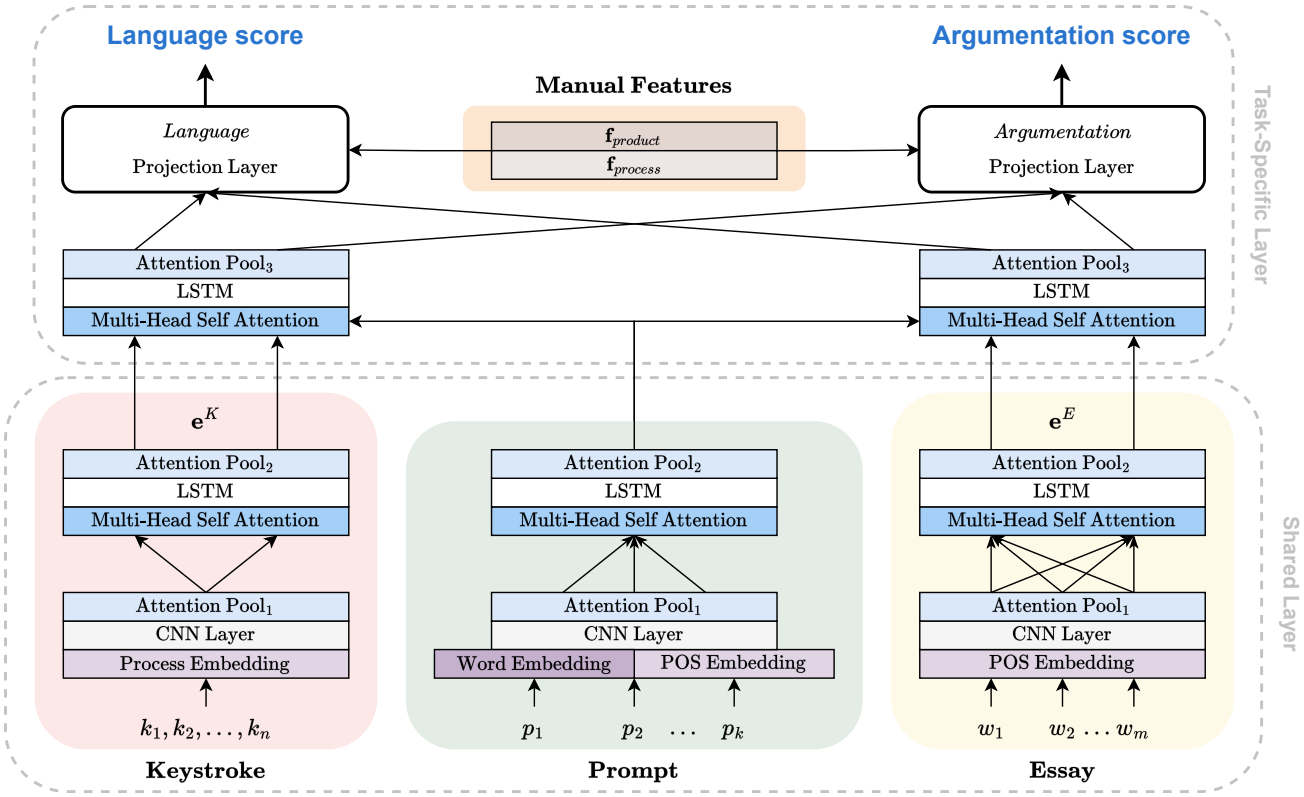


Figure 2: KAES model architecture.

which maps each keystroke operation to the corresponding vector:

$$\mathbf{e}_i^k = \mathbf{E}^K(k_i), \ i = 1, 2, ..., n. \tag{1}$$

Here, $\mathbf{e}_i^k$ represents the embedding of the keystroke operation k_i. Through the embedding layer, the original keystroke sequence is transformed into a vector sequence $\{\mathbf{e}_1^k, \mathbf{e}_2^k, \ldots, \mathbf{e}_n^k\}$. The embedded keystroke sequence is then passed to a convolutional layer, where a one-dimensional convolutional neural network (1D CNN) extracts local features and captures the local dependencies between keystroke sequences:

$$\boldsymbol{c}_i = f\left(\boldsymbol{W}_c \cdot \left[\mathbf{e}_i^k : \mathbf{e}_{i+h_w-1}^k\right] + \boldsymbol{b}_c\right). \tag{2}$$

Here, $\boldsymbol{c}_i$ represents the feature representation after the convolutional layer, $\boldsymbol{W}_c$ is a trainable weight matrix, $\boldsymbol{b}_c$ is the bias vector,

and h_w is the window size of the convolutional layer. For ease of subsequent use and description, we denote the 1D convolution operation in Equation (2) as CNN $(\cdot)$. To enhance the model's ability to focus on critical information, we introduce an attention pooling mechanism following the convolutional layer[12]. The KAES model assigns different weights to various keystroke operations based on attention weights, thus performing weighted pooling of the output:

$$\boldsymbol{a}_i = \tanh\left(\boldsymbol{W}_a \cdot \boldsymbol{c}_i + \boldsymbol{b}_a\right), \tag{3}$$

$$u_i = \frac{exp\left(\boldsymbol{r}_u \cdot \boldsymbol{a}_i\right)}{\sum_j exp\left(\boldsymbol{r}_u \cdot \boldsymbol{a}_j\right)}, \tag{4}$$

$$\boldsymbol{k}_s = \sum_i u_i \cdot \boldsymbol{c}_i. \tag{5}$$

Here, $\boldsymbol{a}_i$ and u_i represent the attention vector and attention weight of the i-th keystroke operation, respectively. $\boldsymbol{W}_a$ and $\boldsymbol{r}_u$ are the weight matrix and weight vector, and $\boldsymbol{b}_a$ is the bias vector. $\boldsymbol{k}_s$ represents the keystroke sequence corresponding to the completion of a full sentence by the author. We denote the attention pooling operation represented by Equations (3), (4), and (5) as AttentionPool $(\cdot)$. Next, we use the generated sentence-level keystroke sequence representation as input and apply multi-head self-attention to produce an essay-level representation. Here, taking the j-th trait score prediction task as an example, its representation is obtained through the multi-head attention mechanism:

$$\boldsymbol{H}_i^j = \text{Att}(X\boldsymbol{W}_{i,j}^K, X\boldsymbol{W}_{i,j}^Q, X\boldsymbol{W}_{i,j}^V), \tag{6}$$

$$\boldsymbol{m}^j = \text{Concat}(\boldsymbol{H}_1^j, ..., \boldsymbol{H}_h^j)\boldsymbol{W}_j^O. \tag{7}$$

Here, $Att\,(\cdot)$ and H_i represent the scaled dot-product attention operation and the i-th head, respectively. X is the sentence-level keystroke sequence representation matrix. $\boldsymbol{W}_{i,j}^K$, $\boldsymbol{W}_{i,j}^Q$, and $\boldsymbol{W}_{i,j}^V$ are the parameter matrices. $\boldsymbol{W}_j^O$ is the linear transformation matrix for the output. We denote the multi-head attention mechanism represented by Equations (6) and (7) as MultiHead $(\cdot)$.

Due to the strong learning capability of LSTM in handling sequential data, it is able to capture the long-term dependencies within the keystroke sequence. Next, we apply the recurrent layer of the LSTM to capture the sequential connections of keystroke operation features, generating hidden representations over time steps. Finally, we apply attention pooling to these hidden representations to obtain the global representation of the keystroke sequence during the writing process, denoted as $\mathbf{e}^K$:

$$\mathbf{h}_t^j = \text{LSTM}(\mathbf{m}_{t-1}^j, \mathbf{m}_t^j), \tag{8}$$

$$\mathbf{e}^K = \text{AttentionPool}_2\left(\mathbf{h}_t^j\right). \tag{9}$$

Here, $\mathbf{h}_t^j$ represents the hidden representation at time step t for the j-th task, and $\boldsymbol{m}^j$ is the concatenated output of the multi-head attention layer.

4.2 Essay Representation

The challenge in implementing cross-prompt AES lies in the semantic space differences between essays with different prompts. Inspired by previous work[30, 31], we do not use word embeddings, as they tend to encourage overfitting in cross-prompt settings. Instead, we represent essay text grammatically through part-of-speech (POS) embeddings, which provide a more generalizable representation. Specifically, we use Python's NLTK[2] package to perform POS tagging on each sentence of the essay and map it to dense vectors. The formula is expressed as follows:

$$\boldsymbol{x}_i = \text{POS}\left(\boldsymbol{w}_i\right),\ i = 1, 2, ..., m. \tag{10}$$

Here, $\boldsymbol{x}_i$ is the embedding vector for the i-th word, and POS $(\cdot)$ represents the part-of-speech embedding for the input words. $\boldsymbol{w}_i$ is the i-th word in the sentence. Then, to obtain the sentence-level representation, we apply a one-dimensional convolutional layer and attention pooling layer to the part-of-speech representations of each sentence:

$$\boldsymbol{s} = \text{AttentionPool}_1\left(\text{CNN}\left(\boldsymbol{x}_i : \boldsymbol{x}_{i+h_w-1}\right)\right). \tag{11}$$

Here, $\boldsymbol{s}$ is the final sentence representation. We then apply the multi-head self-attention mechanism, LSTM layer, and attention pooling layer to obtain the essay representation $\mathbf{e}^E$. The formulas are as follows:

$$\boldsymbol{g}^j = \text{MultiHead}\left(\boldsymbol{s}_1^j, ..., \boldsymbol{s}_n^j\right), \tag{12}$$

$$\mathbf{e}^E = \text{AttentionPool}_2\left(\text{LSTM}(\boldsymbol{g}_{t-1}^j, \boldsymbol{g}_t^j)\right). \tag{13}$$

Here, j represents the j-th feature score prediction task, and $\boldsymbol{s}_i$ denotes the embedding representation of the i-th sentence. $\boldsymbol{g}^j$ is the concatenated output of the multi-head attention layer.

4.3 Prompt Representation

In real-world human scoring scenarios, judges score essays based on the prompt. Similarly, for the AES tasks, the prompt plays a crucial role in guiding the writing direction and assessing text quality. Therefore, it is essential to ensure that the model can effectively capture and focus on the key information in the prompt to improve the accuracy and fairness of the scoring. Inspired by this, we encode the prompt instructions corresponding to each essay and direct the model's attention to it. Notably, to incorporate the content of the prompt, we additionally include pre-trained GloVe [27] word embeddings in the embedding layer of the prompt representation module:

$$\boldsymbol{p}_i^e = \text{GloVe}\left(\boldsymbol{p}_i\right) + \text{POS}\left(\boldsymbol{p}_i\right),\ i = 1, 2, ..., k. \tag{14}$$

Here, $\boldsymbol{p}_i^e$ and $\boldsymbol{p}_i$ represent the i-th word in the prompt and its corresponding embedding vector, respectively. GloVe $(\cdot)$ denotes the word embedding for the input words. Then, we sequentially apply the prompt's embedding representation to a one-dimensional convolutional layer, attention pooling layer, multi-head self-attention layer, LGBM layer, and a second attention pooling layer. Since these processes are essentially the same as those used to obtain the essay representation, we will not elaborate further.

4.4 Trait Score Prediction

To capture the relationship between the essay and the prompt, we use the generated prompt representation $\mathbf{e}^P$ and the essay representation $\mathbf{e}^E$ as inputs. We apply the multi-head self-attention

[2]https://www.nltk.org/

mechanism and LSTM layer to obtain the prompt-aware essay representation $\mathbf{T}^j$ for each j-th task. The specific processes are as follows:

$$\mathbf{Q} = \mathbf{W}_q \cdot \mathbf{e}^P, \quad \mathbf{K} = \mathbf{W}_k \cdot \mathbf{e}^E, \quad \mathbf{V} = \mathbf{W}_v \cdot \mathbf{e}^E, \tag{15}$$

$$\text{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}, \tag{16}$$

$$\mathbf{T}^j = \text{LSTM}(\text{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V})). \tag{17}$$

Here, $\mathbf{W}_q$, $\mathbf{W}_k$, and $\mathbf{W}_v$ are the weight matrices for the query, key, and value, respectively. d_k is the dimension of the key vector, used for scaling to ensure numerical stability. $\mathbf{T}^j$ represents the prompt-sensitive essay representation, used for scoring each j-th task.

For the final prediction, we concatenate $\mathbf{T}^j$ with the pre-designed writing product features $\mathbf{f}^j_{product}$ and writing process features $\mathbf{f}^j_{process}$ to form the feature vector $\mathbf{f}^* = \mathbf{T}^j || \mathbf{f}^j_{product} || \mathbf{f}^j_{process}$. Here, $||$ denotes the concatenation operation. Finally, we connect this with the essay representation $\mathbf{e}^E$ and the keystroke operation representation $\mathbf{e}^K$ to serve as input:

$$\mathbf{Input}^j = \text{Concat}(\mathbf{f}^*, \mathbf{e}^E, \mathbf{e}^K). \tag{18}$$

Then, the model performs the trait attention defined in [31]:

$$\mathbf{F} = [\mathbf{Input}^1, \ldots, \mathbf{Input}^M], \tag{19}$$

$$\mathbf{t}^j = \sum_i \left(\mathbf{F}_{-j,i} \cdot \frac{\exp(\text{score}(\mathbf{Input}^j, \mathbf{F}_{-j,i}))}{\sum_l \exp(\text{score}(\mathbf{Input}^j, \mathbf{F}_{-j,l}))} \right). \tag{20}$$

Here, $\mathbf{F}$ and $\mathbf{F}_{-j}$ represent the concatenation of the representations for each feature prediction and the masking of the target feature representation, respectively. $\mathbf{t}^j$ is the attention vector. The final representation for each trait prediction, $\mathbf{final}^j = \mathbf{Input}^j || \mathbf{t}^j$, is obtained by concatenating $\mathbf{Input}^j$ and $\mathbf{t}^j$. Finally, we obtain the final feature score $\hat{y}^j$ by applying a linear layer with a sigmoid function:

$$\hat{y}^j = \text{sigmoid}(\mathbf{w}^j_y \cdot \mathbf{final}^j + b^j_y). \tag{21}$$

Here, $\mathbf{w}^j_y$ is the weight vector, and b^j_y is the bias.

4.5 Model Optimization

The training of KAES uses the following loss function L_{total}:

$$\text{L}_{\text{total}}(y, \hat{y}) = \lambda \cdot \text{L}_{\text{mse}}(y, \hat{y}) + (1 - \lambda) \cdot \text{L}_{\text{ts}}(y, \hat{y}), \tag{22}$$

$$\text{L}_{\text{mse}}(y, \hat{y}) = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} (\hat{y}_{ij} - y_{ij})^2, \tag{23}$$

$$\text{L}_{\text{ts}}(y, \hat{y}) = \frac{1}{c} \sum_{j=2}^{M} \sum_{k=j+1}^{M} \text{TS}(\hat{\mathbf{y}}_j, \hat{\mathbf{y}}_k, \mathbf{y}_j, \mathbf{y}_k), \tag{24}$$

$$\text{TS} = \begin{cases} 1 - \cos(\hat{\mathbf{y}}_j, \hat{\mathbf{y}}_k) & \text{, if } \text{r}(\mathbf{y}_j, \mathbf{y}_k) \geq \delta \\ 0 & \text{, otherwise} \end{cases}. \tag{25}$$

Here, λ is a hyperparameter used to adjust the balance between the two losses. cos and r represent the cosine similarity and Pearson correlation coefficient, respectively. δ is the threshold, and c is the count of non-zero calculated TS. $\mathbf{y}_j$ is the j-th true feature vector, and $\hat{\mathbf{y}}_j$ is the predicted feature vector.

5 EXPERIMENT

5.1 Experiment Settings

5.1.1 Cross Validation. We perform six-prompt cross-validation, following the approach used in current prompt-specific AES models[10, 19, 30, 31]. For each iteration, essays from one prompt are used as the test data, while essays from the remaining prompts serve as the training data. This is repeated for each prompt, ensuring all prompts are evaluated.

5.1.2 Evaluation Metric. To assess the model's performance, we use the Quadratic Weighted Kappa (QWK) metric. QWK is widely regarded as the most reliable metric in both holistic and trait-specific scoring research[11, 12, 17, 19, 28, 33, 34], as it measures the agreement between system predictions and human raters. The QWK score ranges from -1 to 1, with values closer to 1 indicating better model performance, while lower values reflect weaker performance.

5.1.3 Baseline. To demonstrate the effectiveness of the KAES model, we compare it against seven strong baseline methods. These baselines, as mentioned earlier, can be categorized into three groups:

Feature Engineering Models

- **LGBM**[20]: This model is a high-performance machine learning model based on the gradient boosting framework. It accelerates training and reduces memory usage through histogram-based decision tree learning and is widely applied in both classification and regression tasks.

DNN-AES Models

- **Hi att**[12]: A leading prompt-specific overall scoring model that also excels in trait-specific scoring tasks.
- **AES_aug**[17]: This model builds upon the AES_T&N[33] framework by adding a linear layer for each trait, transforming the overall scoring model into a multi-task learning approach.

Hybrid Models

- **PAES**[30]: A cross-prompt holistic scoring model that leverages general essay quality features, which are not specific to any particular prompt, to handle semantic variations across different prompts.
- **CTS**[31]: A novel automatic cross-prompt trait scoring model, which incorporates a trait attention mechanism and a multi-task architecture. CTS has demonstrated superior performance compared to existing prompt-specific and cross-prompt AES methods.
- **CTS no att**[31]: This model uses the same multi-task architecture as CTS, but without the trait attention mechanism, focusing solely on shared and private layers.
- **ProTACT**[10]: The state-of-the-art cross-prompt AES model, specifically designed for trait scoring. ProTACT uses an essay-prompt attention mechanism and topic modeling to extract topic consistency features and employs a trait similarity loss function to enable multi-trait scoring even without labeled data.

5.1.4 Hyperparameter Settings. For a fair comparison, we use the official open-source code for the baseline models and carefully tune the hyperparameters to maximize performance. We train all

models for 50 epochs and select the epoch with the highest average QWK score for all traits in the development set for testing. In our implementation, we use the RMSprop algorithm for optimization, with a learning rate of 0.001. The CNN filter size and kernel size are set to 100 and 5, respectively. LSTM units are set to 100, and the part-of-speech embedding dimension is set to 50. We use a batch size of 10 and apply a dropout rate of 0.5 to the deep neural network layers to prevent overfitting. We run each model five times and the average scores represent the final scores.

5.2 Overall Results and Analyses

In this section, we report the performance comparisons between our KAES and all the baselines. The results are shown in Table 2 and Table 3, which show the performance of the prediction tasks for the language dimension and argumentation dimension. The bold numbers indicate the optimal values, while the underlined numbers represent the second-best values.

The data reveals that KAES consistently outperforms the best baseline model, ProTACT, with an average QWK improvement of 4.54% in the language dimension and 2.68% in the argumentation dimension. The integration of keystroke data into the traditional AES framework enables the KAES model to outperform most prompts, as measured by the QWK metric. For instance, in Prompt 1, KAES achieves a QWK score of 0.737 for the language dimension, improving by 2.07% over ProTACT (0.722). Similarly, in the argumentation dimension, KAES achieves a QWK score of 0.645, a 4.54% improvement over CTS (0.617). These significant performance improvements in automated scoring demonstrate the effectiveness of the KAES model. The ablation study, which will be discussed later, will further examine the contribution of each module in the KAES model.

Overall, hybrid models, including KAES, outperformed both DNN-AES models and feature engineering models, which aligns with previous studies[30, 31, 36]. This advantage arises from the hybrid model's ability to combine the automatic feature extraction power of deep learning with the targeted feature selection of traditional methods. Consequently, the hybrid model can leverage a broader range of data, leading to superior performance compared to purely DNN-AES models and feature-engineering-based approaches.

Among hybrid models, KAES achieves the best performance, attributed to its dual approach: it extracts static information from keystroke logs through feature engineering while also capturing temporal dependencies in keystroke sequences using a deep learning model. This enables KAES to track changes in students' writing processes effectively. The results confirm the practicality of incorporating writing process data into AES systems, with our model effectively leveraging this data.

5.3 Ablation Study

To explore the contribution of each module in the KAES model, we conduct experiments on several model variants:

- **−keystroke**: KAES without the keystroke sequence feature extraction module.
- **−feature**: KAES without the manual feature module.
- **−essay**: KAES without the essay text feature extraction module.

The results of the ablation study are presented in Figure 3. The full KAES model consistently outperforms its variants. Removing any module leads to a reduction in performance. This demonstrates the critical role each component plays in improving the system's accuracy and robustness.

Effect of essay texts: Removing the essay text module (-essay) results in the largest performance drop, underscoring the importance of essay content in scoring. This finding is broadly in line with educational practice: the core of writing scoring is the assessment of content quality[5].

Effect of writing product features & process features: Removing the manual feature module (-feature) also decreases performance, though the drop is less severe than with the essay text. This suggests that while manual features are valuable, their contribution is smaller than that of the deep learning-extracted product features.

Effect of Keystroke sequence feature extraction module: Removing the keystroke sequence feature extraction module (-keystroke) results in a noticeable, though smaller, performance decline. Although the decrease is not as significant as removing essay text features, the results may still indicate that keystroke sequence features play an important role in the scoring system. It is worth mentioning that "-keystroke" only represents results without the encoded keystroke vectors, the feature variables were still included in the model. If we completely remove all keystroke-related data, the model performance decreases by around 6%.

6 DISCUSSION AND FUTURE WORK

The Final Product Remains Key. The decline in model performance when the essay content module is removed highlights the critical role of content in writing evaluation. This finding aligns with the intuitive understanding that human raters primarily focus on the final written product.

Keystroke Data: A Window into the Writing Process. While the final written product remains paramount, the inclusion of process data significantly enhances the model's performance. Specifically, keystroke features provide valuable insights into the student's writing process, enabling educators to identify difficulties in real-time and deliver targeted feedback. Such process data is crucial for personalized instruction and early intervention.

Effective writing assessment should consider both the final output and the writing process. The KAES model's robustness stems from its incorporation of diverse data sources—like the final written piece, prompt, keystroke records, and manual indicators—within a multi-task learning environment, enabling predictions across various dimensions. This amalgamation results in a more thorough insight into student writing challenges. For example, excessive reliance on copying and pasting might suggest issues with integrating external data or a lack of creativity; repeated halts and edits could point to difficulties in structuring or organizing arguments; and a high frequency of grammatical and spelling errors might reveal gaps in understanding grammar rules or insufficient vocabulary.

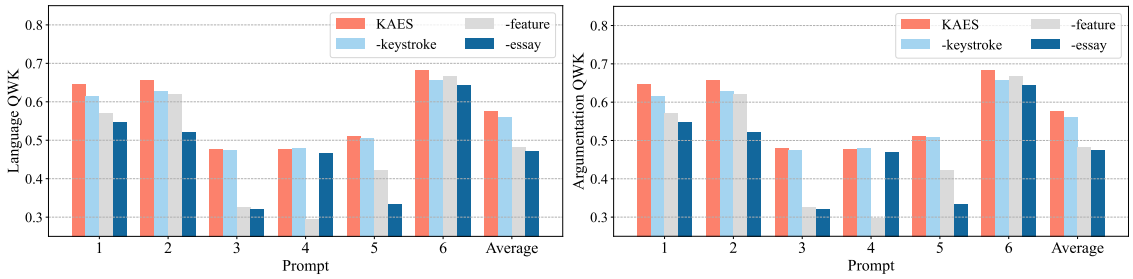
Exploring the Future. Several limitations of our current work warrant further exploration. First, while the pre-designed manual features provide interpretability, they lack additional engineering

Model	1	2	3	4	5	6	AVG
LGBM	0.521	0.642	0.658	0.476	0.596	0.415	0.551
Hi att	0.583	0.702	0.618	0.503	0.579	0.415	0.567
AES aug	0.557	0.709	0.627	0.496	0.574	0.451	0.569
PAES	0.639	0.703	0.662	0.447	0.583	0.385	0.570
CTS no att	0.637	**0.736**	0.623	0.514	0.604	0.408	0.587
CTS	0.663	0.706	0.715	0.494	0.615	0.432	0.604
ProTACT	0.722	0.726	0.636	0.485	0.617	0.516	0.617
KAES	**0.737**	0.731	**0.719**	**0.517**	**0.620**	**0.543**	**0.645**
improvement	**2.07%**	**-0.68%**	**0.56%**	**0.58%**	**0.49%**	**5.23%**	**4.54%**

Table 2: Mean QWK scores for each prompt on language dimension.

Model	1	2	3	4	5	6	AVG
LGBM	0.509	0.461	0.433	0.444	0.440	0.513	0.467
Hi att	0.572	0.557	0.427	0.486	0.417	0.565	0.504
AES aug	0.591	0.547	0.434	0.478	0.449	0.571	0.512
PAES	0.592	0.565	0.453	0.449	0.424	0.515	0.500
CTS no att	0.603	0.609	0.452	0.497	0.383	0.556	0.517
CTS	0.617	0.620	0.467	**0.507**	0.438	0.578	0.538
ProTACT	0.614	0.628	0.475	0.479	0.507	0.657	0.560
KAES	**0.645**	**0.656**	**0.478**	0.477	**0.510**	**0.682**	**0.575**
improvement	**4.54%**	**4.46%**	**0.63%**	**-5.92%**	**0.59%**	**3.80%**	**2.68%**

Table 3: Mean QWK scores for each prompt on argumentation dimension.



(a) QWK in the language dimension

(b) QWK in the argumentation dimension

Figure 3: Results of ablation experiments.

steps to fully justify their inclusion. Second, the feature extraction methods for process data and the final product are similar, potentially reducing the unique contribution of keystroke logs. Lastly, although our model demonstrates significant improvements on the CBAL dataset, it has not yet been validated on other datasets.

In future work, we plan to enhance the temporal modeling of keystroke sequences to capture more detailed temporal information. We will also optimize feature design to improve automated scoring systems. Additionally, we plan to develop interactive tools for real-time feedback and writing guidance using keystroke data and text analysis.

7 CONCLUSION

In this study, we investigate the use of keystroke logs combined with deep learning models for scoring argumentative essays. Our goal is to fully exploit the rich information embedded in keystroke data to enhance the effectiveness of AES tasks. To this end, we propose a novel AES model named KAES, which integrates data from multiple sources—essay content, writing prompts, keystroke sequences, and handcrafted features—within a multi-task learning framework. The KAES model predicts scores across both the language and argumentation dimensions of the essay. Experiments conducted on a real-world CABL dataset provided by ETS demonstrate that the

KAES model significantly outperforms various baseline models. Furthermore, ablation studies confirm the contribution and effectiveness of each individual module within the KAES framework.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (62177044).

References

[1] Yousef Abosalem. 2016. Assessment techniques and students' higher-order thinking skills. *International Journal of Secondary Education* 4, 1 (2016), 1–11.
[2] Dimitrios Alikaniotis, Helen Yannakoudakis, and Marek Rei. 2016. Automatic text scoring using neural networks. *arXiv preprint arXiv:1606.04289* (2016).
[3] Yigal Attali and Jill Burstein. 2006. Automated essay scoring with e-rater® V. 2. *The Journal of Technology, Learning and Assessment* 4, 3 (2006).
[4] Veerle M. Baaijen and David Galbraith. 2018. Discovery Through Writing: Relationships with Writing Processes and Text Quality. *Cognition and Instruction* 36, 3 (July 2018), 199–223. https://doi.org/10.1080/07370008.2018.1456431
[5] Jungok Bae, Peter M Bentler, and Yae-Sheik Lee. 2016. On the role of content in writing assessment. *Language Assessment Quarterly* 13, 4 (2016), 302–328.
[6] Randy E Bennett, Paul Deane, and Peter W. van Rijn. 2016. From cognitive-domain theory to assessment practice. *Educational Psychologist* 51, 1 (2016), 82–107.
[7] Iris Breetvelt, Huub Van den Bergh, and Gert Rijlaarsdam. 1994. Relations between writing processes and text quality: When and how? *Cognition and instruction* 12, 2 (1994), 103–123.
[8] Ikkyu Choi and Paul Deane. 2021. Evaluating writing process features in an adult EFL writing assessment context: A keystroke logging study. *Language Assessment Quarterly* 18, 2 (2021), 107–132.
[9] Paul Deane, Yi Song, Peter Van Rijn, Tenaha O'Reilly, Mary Fowles, Randy Bennett, John Sabatini, and Mo Zhang. 2019. The case for scenario-based assessment of written argumentation. *Reading and Writing* 32 (2019), 1575–1606.
[10] Heejin Do, Yunsu Kim, and Gary Geunbae Lee. 2023. Prompt- and Trait Relation-aware Cross-prompt Essay Trait Scoring. (July 2023).
[11] Fei Dong and Yue Zhang. 2016. Automatic features for essay scoring–an empirical study. In *Proceedings of the 2016 conference on empirical methods in natural language processing*. 1072–1077.
[12] Fei Dong, Yue Zhang, and Jie Yang. 2017. Attention-Based Recurrent Convolutional Neural Network for Automatic Essay Scoring. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*. Association for Computational Linguistics, Vancouver, Canada, 153–162. https://doi.org/10.18653/v1/K17-1017
[13] Mary K Enright and Thomas Quinlan. 2010. Complementing human judgment of essays written by English language learners with e-rater® scoring. *Language Testing* 27, 3 (2010), 317–334.
[14] Yuehchiu Fang. 2010. Perceptions of the computer-assisted writing program among EFL college learners. *Journal of Educational Technology & Society* 13, 3 (2010), 246–256.
[15] 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 (2018), 194–216.
[16] H., John, Bernardin, Stephanie, Thomason, M., Ronald, Buckley, Jeffrey, and S. 2016. Rater Rating-Level Bias and Accuracy in Performance Appraisals: The Impact OF Rater Personality, Performance Management Competence, and Rater Accountability. *Human Resource Management* 55, 2 (2016), 321–340.
[17] Mohamed A. Hussein, Hesham A., and Mohammad Nassef. 2020. A Trait-based Deep Learning Automated Essay Scoring System with Adaptive Feedback. *International Journal of Advanced Computer Science and Applications* 11, 5 (2020). https://doi.org/10.14569/IJACSA.2020.0110538
[18] Mohamed Abdellatif Hussein, Hesham Hassan, and Mohammad Nassef. 2019. Automated Language Essay Scoring Systems: A Literature Review. *PeerJ. Computer Science* 5 (2019), e208. https://doi.org/10.7717/peerj-cs.208
[19] Cancan Jin, Ben He, Kai Hui, and Le Sun. 2018. TDNN: A Two-stage Deep Neural Network for Prompt-independent Automated Essay Scoring. In *Meeting of the Association for Computational Linguistics*.
[20] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems* 30 (2017).
[21] Zixuan Ke and Vincent Ng. 2019. Automated Essay Scoring: A Survey of the State of the Art. (2019).
[22] Mariëlle Leijten and Luuk Van Waes. 2013. Keystroke logging in writing research: Using Inputlog to analyze and visualize writing processes. *Written Communication* 30, 3 (2013), 358–392.
[23] Elijah Mayfield and Alan W Black. 2020. Should you fine-tune BERT for automated essay scoring?. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*. 151–162.
[24] Huy Nguyen and Diane Litman. 2018. Argument mining for improving the automated scoring of persuasive essays. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.
[25] Ellis B Page. 1966. The imminence of... grading essays by computer. *The Phi Delta Kappan* 47, 5 (1966), 238–243.
[26] Ellis B. Page. 2003. Project Essay Grade: PEG. https://api.semanticscholar.org/CorpusID:57057724
[27] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.
[28] Peter Phandi, Kian Ming A Chai, and Hwee Tou Ng. 2015. Flexible domain adaptation for automated essay scoring using correlated linear regression. In *Proceedings of the 2015 conference on empirical methods in natural language processing*. 431–439.
[29] Donald E Powers, Jill C Burstein, Martin Chodorow, Mary E Fowles, and Karen Kukich. 2002. Stumping e-rater: challenging the validity of automated essay scoring. *Computers in Human Behavior* 18, 2 (2002), 103–134.
[30] Robert Ridley, Liang He, Xinyu Dai, Shujian Huang, and Jiajun Chen. 2020. Prompt Agnostic Essay Scorer: A Domain Generalization Approach to Cross-prompt Automated Essay Scoring. arXiv:2008.01441 [cs]
[31] Robert Ridley, Liang He, Xin-yu Dai, Shujian Huang, and Jiajun Chen. 2021. Automated Cross-Prompt Scoring of Essay Traits. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 15 (May 2021), 13745–13753. https://doi.org/10.1609/aaai.v35i15.17620
[32] Sandip Sinharay, Mo Zhang, and Paul Deane. 2019. Prediction of essay scores from writing process and product features using data mining methods. *Applied Measurement in Education* 32, 2 (2019), 116–137.
[33] Kaveh Taghipour and Hwee Tou Ng. 2016. A neural approach to automated essay scoring. In *Proceedings of the 2016 conference on empirical methods in natural language processing*. 1882–1891.
[34] Yi Tay, Minh Phan, Luu Anh Tuan, and Siu Cheung Hui. 2018. Skipflow: Incorporating neural coherence features for end-to-end automatic text scoring. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.
[35] Masaki Uto. 2021. A Review of Deep-Neural Automated Essay Scoring Models. *Behaviormetrika* 48, 2 (July 2021), 459–484. https://doi.org/10.1007/s41237-021-00142-y
[36] Masaki Uto, Itsuki Aomi, Emiko Tsutsumi, and Maomi Ueno. 2023. Integration of Prediction Scores From Various Automated Essay Scoring Models Using Item Response Theory. *IEEE Transactions on Learning Technologies* 16, 6 (Dec. 2023), 983–1000. https://doi.org/10.1109/TLT.2023.3253215
[37] Masaki Uto and Maomi Ueno. 2018. Empirical comparison of item response theory models with rater's parameters. *Heliyon* 4, 5 (2018).
[38] Peter Van Rijn, Jing Chen, and Yuen Yan-Koo. 2016. Statistical results from the 2013 CBAL English language arts multistate study: Parallel forms for policy recommendation writing. *Research Memorandum* (2016), 16–01.
[39] Yiqin Yang, Li Xia, and Qianchuan Zhao. 2019. An automated grader for Chinese essay combining shallow and deep semantic attributes. *IEEE Access* 7 (2019), 176306–176316.
[40] Mo Zhang, Hongwen Guo, and Xiang Liu. 2021. Using Keystroke Analytics to Understand Cognitive Processes during Writing. *International Educational Data Mining Society* (2021).
[41] Mo Zhang, Mengxiao Zhu, Paul Deane, and Hongwen Guo. 2019. Identifying and Comparing Writing Process Patterns Using Keystroke Logs. In *Quantitative Psychology*, Marie Wiberg, Steven Culpepper, Rianne Janssen, Jorge González, and Dylan Molenaar (Eds.). Vol. 265. Springer International Publishing, Cham, 367–381. https://doi.org/10.1007/978-3-030-01310-3_32
[42] Mengxiao Zhu, Mo Zhang, and Paul Deane. 2019. Analysis of Keystroke Sequences in Writing Logs. *ETS Research Report Series* 2019, 1 (2019), 1–16. https://doi.org/10.1002/ets2.12247
[43] Mengxiao Zhu, Mo Zhang, and Lin Gu. 2023. Insights into Editing and Revising in Writing Process Using Keystroke Logs. *Language Assessment Quarterly* 20, 4-5 (Oct. 2023), 445–468. https://doi.org/10.1080/15434303.2023.2291478