systems

MDPI

Article

Learning to Score: A Coding System for Constructed Response Items via Interactive Clustering

Lingjing Luo [1,2], Hang Yang [3], Zhiwu Li [3,4,*] and Witold Pedrycz [2,5]

1 School of Marxism, University of Electronic Science and Technology of China, Chengdu 611731, China; lingjingluo@uestc.edu.cn
2 Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB T6R 2V4, Canada; wpedrycz@ualberta.ca
3 Macau Institute of Systems Engineering, Macau University of Science and Technology, Macau, China; 3240007361@student.must.edu.mo
4 School of Electro-Mechanical Engineering, Xidian University, Xi'an 710071, China
5 Systems Research Institute, Polish Academy of Sciences, 01-447 Warsaw, Poland
* Correspondence: zwli@must.edu.mo

Abstract: Constructed response items that require the student to give more detailed and elaborate responses are widely applied in large-scale assessments. However, the hand-craft scoring with a rubric for massive responses is labor-intensive and impractical due to rater subjectivity and answer variability. The automatic response coding method, such as the automatic scoring of short answers, has become a critical component of the learning and assessment system. In this paper, we propose an interactive coding system called ASSIST to efficiently score student responses with expert knowledge and then generate an automatic score classifier. First, the ungraded responses are clustered to generate specific codes, representative responses, and indicator words. The constraint set based on feedback from experts is taken as training data in metric learning to compensate for machine bias. Meanwhile, the classifier from responses to code is trained according to the clustering results. Second, the experts review each coded cluster with the representative responses and indicator words to score a rating. The coded cluster and score pairs will be validated to ensure inter-rater reliability. Finally, the classifier is available for scoring a new response with out-of-distribution detection, which is based on the similarity between response representation and class proxy, i.e., the weight of class in the last linear layer of the classifier. The originality of the system developed stems from the interactive response clustering procedure, which involves expert feedback and an adaptive automatic classifier that can identify new response classes. The proposed system is evaluated on our real-world assessment dataset. The results of the experiments demonstrate the effectiveness of the proposed system in saving human effort and improving scoring performance. The average improvements in clustering quality and scoring accuracy are 14.48% and 18.94%, respectively. Additionally, we reported the inter-rater reliability, out-of-distribution rate, and cluster statistics, before and after interaction.

Keywords: constructed response items; response coding; text clustering; large-scale assessment

Citation: Luo, L.; Yang, H.; Li, Z.; Pedrycz, W. Learning to Score: A Coding System for Constructed Response Items via Interactive Clustering. *Systems* **2024**, *12*, 380. https://doi.org/10.3390/systems12090380

Academic Editor: William T. Scherer

Received: 21 August 2024
Revised: 17 September 2024
Accepted: 18 September 2024
Published: 21 September 2024

1. Introduction

Constructed response items, also known as "open-response" items, have been popular in various assessments for a long time due to their ability to gauge a student's depth of understanding and critical thinking skills [1]. Despite their effectiveness and flexibility, the scoring of these responses using rubrics is plagued by labor-intensive processes. The diversity of answers makes scoring tiring and inefficient. For example, the semantics of student responses vary even within the same score. As a solution to these challenges, the automatic response coding system [2,3] has emerged as a critical facet within the realm of education assessment, which benefits many applications such as cognitive diagnosis [4] and automatic feedback [5].

The automatic response coding systems consider the existing test log as the training set, i.e., (response, code) pairs set, and transfer the learnable knowledge into new responses. The construction of ground truth in the training stage mainly falls into two categories. One is manually coding all existing responses, which is referred to as the fully-supervised method. Fine-tuning large language models has been widely used recently because of its performance in capturing complex semantics, especially in relatively small domain-specific datasets [6]. Another method is coding the most representative responses and propagating them to similar ones, which is referred to as the semi-supervised method. The pioneering semi-supervised work integrated text semantic representation and clustering to improve scoring efficiency on the OECD PISA 2012 items [7].

A key challenge for semi-supervised automatic coding is clustering, i.e., bringing together responses that look different but are essentially the same, where each cluster is called an *equivalent* response class. (Equivalent responses here mean semantic similarity responses instead of the exact same responses.) Clustering is also useful in fully supervised methods for creating scoring rubrics after machine learning [8]. In this paper, we adopt the latter perspective (i.e., semi-supervised labeling for the training set) and develop an interactive clustering method incorporating metric learning to correct the clustering result with the constrained set.

Although relevant research has made significant progress in semi-supervised methods such as measuring and improving intra-class homogeneity with statistics technique, the clustering fine-tuning, which most directly affects model performance, has been neglected. There is a lack of error correction mechanisms for clustering. For example, one response in a cluster is obviously different from the others and needs to be separated, or an equivalent response class is mistakenly divided into two clusters, which need to be merged. The possible wrong clustering assignment is an implicit risk in the training set construction process. The separated response clustering and expert coding without interaction may mislead the classifier with inductive bias and cause concerns about fairness.

To this end, we propose a coding system with interactive supervised clustering called ASSIST. As shown in Figure 1 in this paper, coding is a process that first categorizes student-generated responses into several discrete classes and then labels the scoring associated with the specific classes. Figure 2 demonstrates the proposed scoring procedure. First, pre-trained language models are used to convert textual responses into representation vectors. Then, the representation is dimensionally reduced and clustered into several equivalent response classes. We integrate an explanation model into the clustering results to show the indicator words for each equivalent class. The indicator words are helpful for experts to understand clustering and discover possible mistakes. To correct the mistakes in response clustering, a study in [9] introduces the interactive supervised clustering where experts can provide two types of designated constraints in clustering—cannot-link and must-link—corresponding to the separation and merging of clusters, respectively. After the refinement of clustering, the classifier is trained to map responses to codes.

So far, the score of each equivalent response class remains unknown. Therefore, experts review each class, including indicator words and representative responses, to assign scores accordingly. Then, to ensure fair and consistent scoring, the proposed method produces the scorer reliability and re-score classes with insufficient reliability. After that, the trained classifier of code and associated score is deployed to judge new responses automatically. Note that the new response may not belong to any existing equivalent class, i.e., not similar to any historical response as mentioned in [10]. To discover these fresh responses that need expert scoring, we consider prediction confidence. For example, while considering output probability as confidence level, responses with "flat" classes probability will be treated as out-of-distribution samples [11].

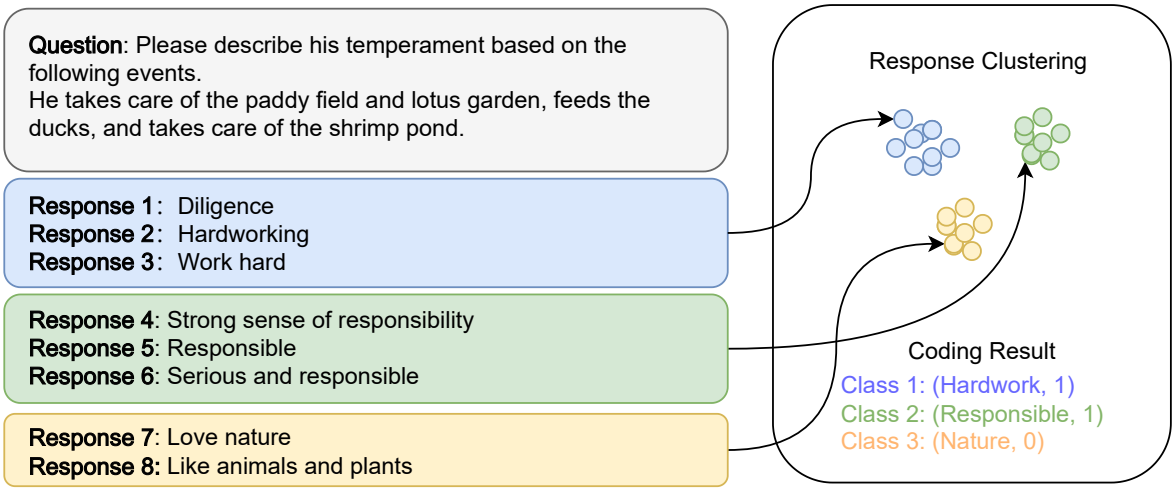

Figure 1. An example of coding for constructed response items. The example question and responses are translated from Mandarin.

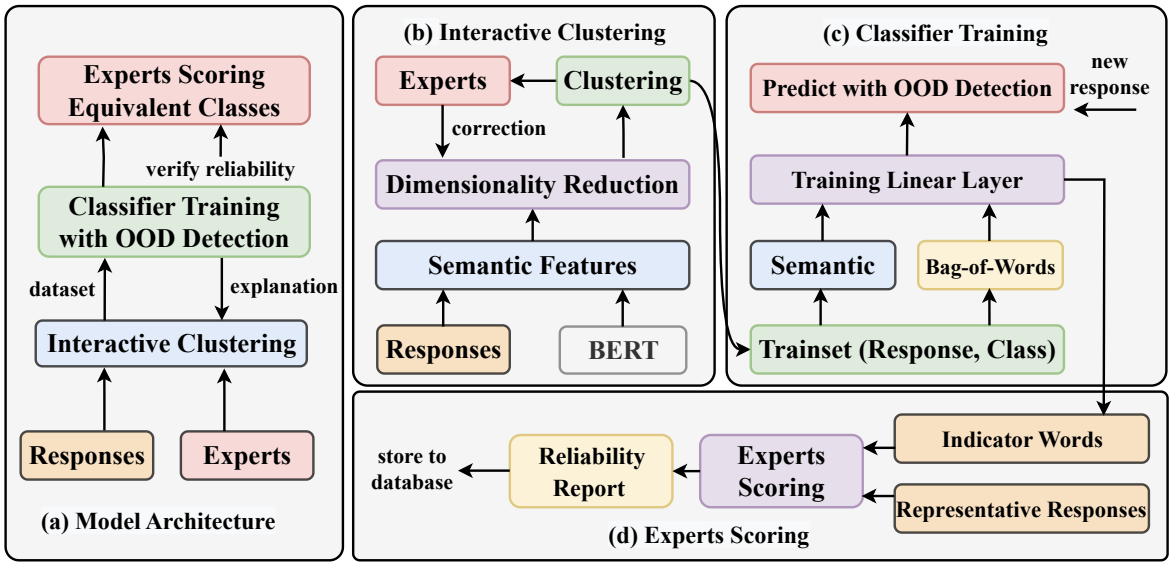

Figure 2. Architecture of ASSIST system.

To evaluate the proposed ASSIST system, we conduct experiments using both real-world assessment datasets and open-sourced benchmark datasets. Furthermore, we carry out comprehensive hyper-parameter sensitivity studies to showcase the resilience and efficacy of the proposed interactive supervised clustering approach in enhancing response representations. The results demonstrate a significant reduction in workforce requirements while concurrently improving scoring performance and automatic scoring accuracy.

The main contributions of this research work are as follows:

1. We propose a novel and practical response coding approach to obtain better scoring results for constructed response items in a large-scale assessment setting.
2. State-of-the-art algorithms are integrated into the proposed ASSIST system for clustering and classifier learning. We propose an interactive clustering method that allows experts to actively review and refine the code-discovering process. We incorporate a classifier with out-of-distribution detection for learning to score while explaining clustering results.
3. We conduct experiments on real-world assessment datasets. The performance and visualization results highlighted the effectiveness of the developed model, including clustering quality and automatic scoring accuracy.

This research work is based on the three-step computational grounded theory methodological framework [12]. First, unsupervised clustering is used to help human experts find patterns in answers. Then, the experts review and commit feedback to correct the clustering. Finally, the feedback is then fed into a metric learning loss to fine-tune the feature extractor and generate identified response codes.

The research work is organized as follows. The literature review is summarized in Section 2. The problem definition and research goal are formally given in Section 3.1, and the methodology details are described in the rest of Section 3. Experimental studies are reported in Section 4. Conclusions are offered in Section 5. Limitations and future works are given in Section 6.

2. Related Works

This section provides an overview of the related works in the area of automatic response coding systems and related machine-learning techniques, such as text clustering. We provide a brief introduction to recent advances and establish connections to our research work in the context of educational automatic scoring. Text clustering is described in the categories of representation learning, dimensionality reduction, and clustering.

2.1. Automatic Response Coding System

After the transition from traditional paper-and-pencil tests to computer-based testing, assessment has become more efficient with data mining and Natural Language Processing (NLP) techniques. The field of automatic response coding systems, especially for fully-supervised short-answer grading, has been researched for over three decades [13]. Early works relied on the prior rules such as concept matching and syntax analysis. The c-Rater [14] of Educational Testing Service (ETS) recognized the main points or concepts of answers after linguistic processing and automatic score based on pre-prepared rules. Recent data-driven works leverage statistical learning to learn from large-scale student responses. Sultan et al. [15] built a random forest classifier using features such as text similarity and term weight based on it or embedded. Marvaniya et al. [8] considered class-specific representatives obtained after text clustering as a better scoring rubric. Tan et al. [16] took the graph neural network as the feature extractor to learn more complex patterns. Zhu et al. [6] compared the student answers and reference answers with the BERT model to predict the very most possible score. Nowadays, the large language model is also explored because of its generalizability for the few-shot datasets. Schneider et al. [17] compare the GPT-3.5 and humans in assessing bachelor-level German and master-level English short answers. Chang et al. [18] designed prompts of ChatGPT for scoring Finnish short answers under zero-shot and one-shot settings. Additionally, some efforts address the automatic evaluation issues in other materials, such as instructional videos [19].

Another branch is semi-supervised automatic coding for reducing human efforts. Zesch et al. [20] formulated the clustering approach and analyzed the semi-supervised performance in the short-answer and essay annotations. Andersen et al. [21] proposed a strategically querying method that can achieve satisfied intra-cluster homogeneity under a statistics guarantee. The originality of this research work comes from the fact that the response clustering procedure is interactive with experts, and the automatic classifier can identify new response classes adaptively, which differs from the existing work in the literature.

2.2. Text Clustering

Representation learning. Text representation is a crucial task in the fields of text mining and NLP, aiming to transform textual data into a format that computers can comprehend and process. The bag-of-words (BoW) model is the most intuitive and effective text representation. In the BoW model representation, the focus is solely on the frequency of each word in a text within the corpus, while much of the structural information in the input text is disregarded [22]. While the BoW model is simple and easy to understand,

it neglects semantic relationships between words, resulting in a loss of information in the representation. To address the limitations of the BoW model, the TF-IDF model is proposed by considering the importance of vocabulary by calculating the product of term frequency and inverse document frequency [23]. This assigns lower weights to high-frequency words, capturing key information in the text more effectively. However, TF-IDF still falls short in capturing semantic relationships between words, especially when dealing with complex natural language tasks. With the advancement of deep learning, pre-trained language models have emerged as a promising force in the field of text representation. Bidirectional encoder representation from transformers (BERT) is a notable example [24]. Unlike traditional models, BERT undergoes pre-training on large corpora, learning rich semantic relationships between words. Through bidirectional encoding, BERT not only considers context information but also comprehends the context of words within a sentence more effectively. The InferSent model is a sentence embedding model designed for natural language understanding tasks [25]. The InferSent model employs a BiLSTM network as its core architecture. The bidirectional nature allows the model to capture contextual information from both the left and right sides of a given word in a sentence.

Dimensionality reduction. After representation learning, dimensionality reduction is a necessary step aimed at capturing the essential information while reducing computational complexity and avoiding the curse of dimensionality. At early stages, principal component analysis (PCA) is a generic and long-established method that linearly transforms high-dimensional data into a lower-dimensional space. After that, t-SNE is a non-linear dimensionality reduction technique that emphasizes preserving the pairwise similarities between data points in the lower-dimensional space [26]. It is particularly effective in capturing local structures and revealing clusters in the data and is widely used in data visualization. Uniform manifold approximation and projection (UMAP) is a non-linear dimensionality reduction technique that aims to preserve both local and global structures of the data. It is known for its efficiency and ability to capture complex relationships in high-dimensional spaces [27]. Semi-supervised methods combine both partially labeled and unlabeled data during dimension reduction to leverage the benefits of metric learning [28]. These methods can enhance the quality of the reduced representation by incorporating labeled information.

Clustering. Clustering is performed in the reduced low-dimension space. K-Means is a widely used unsupervised clustering algorithm that partitions documents into K clusters based on their feature representations. The algorithm iteratively refines cluster centroids to minimize the within-cluster sum of squares [29]. Latent Dirichlet Allocation (LDA) is a probabilistic model that represents documents as mixtures of topics. It assumes that documents are generated from a set of topics, and a Dirichlet distribution of words characterizes each topic. LDA has been used for topic modeling and document clustering [30]. Density-based spatial clustering of applications with noise (DBSCAN) is a density-based clustering algorithm that is particularly effective in identifying clusters of arbitrary shapes in spatial data, which can effectively identify and ignore noisy data points, making it suitable for datasets with outliers [31]. Hierarchical clustering organizes documents into a tree-like structure, where the leaves represent individual documents, and internal nodes represent clusters at different levels of granularity [32]. Hierarchical density-based spatial clustering of applications with noise (HDBSCAN) is an extension of the traditional DBSCAN algorithm that introduces a hierarchical approach to clustering, which offers flexibility in exploring clusters at different levels of granularity [33]. This hard clustering divides data into a specific cluster, while fuzzy clustering data points may belong to more than one cluster [34–36]. Interactive clustering involves a collaborative process where a user interacts with the clustering algorithm, providing feedback and guidance to refine and improve the clustering results. This interactive approach aims to incorporate human expertise and domain knowledge to enhance the quality of clustering outcomes, making the process more transparent and adaptable to user requirements [9]. In other modalities, such as graph-level clustering, there are many effective methods recently, which are sum-

marized in this survey [37]. Contrastive learning [38], cluster-enhanced self-supervised learning [39], and redundancy-free self-supervised relational learning [40] are used to learn the graph representation better.

3. Methodology

3.1. Problem Definition

The proposed ASSIST system aims to assist experts in efficiently coding responses for a constructed response item. Suppose that there are M experts and N raw responses $\{\boldsymbol{r}_i\}_{i=1}^{N}$ generated by students.

Definition 1 (Response Coding)**.** *Response coding refers to the systematic process of assigning numerical or category values to student responses to constructed response items. In this research, we define the code as a doublet (class c, score s). The class is a category variable that represents response semantics. The score is a numerical variable ranging from zero to full points, indicating how correct the response is.*

Responses are transformed into some categories using clustering. After that, experts review and modify clustering results interactively.

Definition 2 (Interactive Clustering)**.** *Assume that N responses are clustered into C equivalent response classes. In order to achieve better clustering performance, the expert can specify that response* $\boldsymbol{r_i}$ *belongs to category* c_i*. The whole constraint set is denoted as* $\{(\boldsymbol{r_i}, c_i)\}_{i=1}^{K}$*, where K is the number of constraints. The algorithm is realized by taking the constraint set as a training set in metrics learning, see Section 3.3. Note that the number of clusters does change with the feedback. The feedback is the set of (response_ID, correction_cluster_ID) pairs. If all the responses in a cluster are reassigned to other clusters, this cluster will be vanished. If some responses from different clusters are reassigned to a new cluster ID, then the representations of these responses will be pulled closer and form the new clusters. For the sake of simplicity, there is only one epoch of feedback aggregating from all experts.*

Then, these equivalent response classes are scored, and the scoring results are evaluated for reliability, such as consistency. Scores that meet the reliability requirements will be stored in the database. Finally, the classifier with out-of-distribution detection is trained for future assessments. The classifier is a mapping from response $\boldsymbol{r}$ to coding (c, s). The out-of-distribution samples refer to responses that do not belong to any existing class.

3.2. System Architecture

The architecture of ASSIST involves multiple stages, as shown in Figure 2, including clustering of ungraded responses, training a response-to-code classifier, expert review, and automatic scoring. The outline is as follows.

1. Semantic Feature Extraction. After text pre-processing, word embedding and pre-trained BERT are used to extract semantic features from the raw responses. Then, to solve the distance failure phenomenon caused by the high dimensionality of the feature, dimensionality reduction is performed on the feature. For text pre-processing, see Section 4.2.
2. Interactive Clustering. The dimension-reduced features are clustered, and each cluster represents an equivalent response class. Experts' feedback helps correct individual errors; see Section 3.3.
3. Classifier Training. The input of the classifier contains two parts, namely semantic features and one-hot bag-of-words vector. The BoW model is used to capture the word similarity between responses and is used to explain the clustering results. The output of the classifier is the equivalent response class; see Section 3.4. The backbone of the classifier is a linear layer. For detailed parameter settings, see Section 4.2.

4. Scoring and Reliability Report. Representative responses and indicator words, which are extracted from clustering and classifier, are provided to each expert to score each equivalent response class. Then, all experts' scoring will be jointly calculated for reliability to ensure the objectivity and consistency of the ratings, see Section 3.5.

3.3. Interactive Clustering

We embed the raw response $\boldsymbol{r}_i$ to create representations in vector space that reflect its semantics. We assume that responses in the same equivalent response classes are semantically similar. The ASSIST uses the pre-trained language model to extract the semantic representation. First, we segment the original response text and remove stop words to generate the token series. Then, the tokens are replaced with the d-dimensional word vectors. Finally, the initial word vector matrix $\boldsymbol{r}_i' \in R^{d \times n_t}$ is obtained, where n_t is the number of tokens. Since the features input to the model must be of fixed length N_t, the representation vector needs to be modified. If the number of tokens $n_t < N_t$, the vector is filled with $N_t - n_t$ blank vectors, i.e., zero-filling. Otherwise, if $n_t > N_t$, the excess $n - N$ word vectors will be removed. After the above steps, the word vector matrix $\boldsymbol{r}_i'' \in R^{d \times N_t}$ can be obtained. Finally, the sentence representation can be extracted with any text representation model. In the following section, the word vector and pre-trained language model is based on an open-source (https://huggingface.co/DMetaSoul/sbert-chinese-general-v2, accessed on 20 December 2023). Sentence-BERT model [41] and fine-tuned with the education question and response corpus collected from a Chinese education platform (https://www.zhixue.com, accessed on 20 December 2023). The semantic sentence representation generated by Sentence-BERT with frozen parameters is denoted by $\boldsymbol{x}_i$.

The higher the dimension, the more the distances between samples tend to be equal, so it is impossible to measure the distance relationship between different samples through distance in the original semantic vector space [42]. Therefore, a dimension reduction technique is needed to post-process the semantic feature. The uniform manifold approximation and projection (UMAP), a general non-linear dimensionality reduction algorithm, is used in the proposed system [27]. There are two steps of UMAP: (1) Learn the manifold structure in high-dimensional space; (2) Find a low-dimension representation of the manifold. In step one, the nearest-neighbor-descent algorithm is used, and then the graph is constructed by connecting the nearest neighbors. The weight in the high-dimension space of edge e between nodes is denoted as $\mu(e)$. The computational details of $\mu(e)$ can be found in [43]. In step two, the UMAP wants the topological structure in a low-dimension space to be preserved in a similar way as possible. Therefore, assume that the weight in low-dimension space is $\nu(e)$ and the edge set is E. The cross-entropy loss function is:

$$\mathcal{L}_{CE}^{DR} = \sum_{e \in E} \mu(e) \log\left(\frac{\mu(e)}{\nu(e)}\right) + (1 - \mu(e)) \log\left(\frac{1 - \mu(e)}{1 - \nu(e)}\right), \tag{1}$$

where $\nu(e) = \phi(||e||_L)$, $\phi(x; a, b) = \left(1 + ax^{2b}\right)^{-1}$, and a, b are hyper-parameters of UMAP.

Assume that the nodes connected with edge e are $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, and $||e||_L$ is the distance in low-dimension space, i.e., $||e||_L = ||u(\boldsymbol{x}_i) - u(\boldsymbol{x}_j)||$, where u is the project function.

The reduced semantic features $\boldsymbol{x}' = u(\boldsymbol{x}_i)$ are clustered with HDBSCAN, a hierarchical density-based clustering algorithm, which considers noise samples as outliers to prevent unrelated responses from being assigned to any cluster [44]. After initial clustering, every response $\boldsymbol{x}_i'$ is assigned to an equivalent class c_i. As mentioned in Definition 2, with the constraint set $\{(\boldsymbol{r_i}, c_i)\}_{i=1}^{K}$, the project function u is further trained. The intuition here is to push the feature $\boldsymbol{x}_i'$ to the centroid $\bar{\boldsymbol{x}}_i'$ of the class c_i. The loss function is:

$$\mathcal{L}_{DIS}^{DR} = \sum_{i=1}^{K} ||u(\boldsymbol{x}_i) - \frac{1}{n_{c_i}} \sum_{j=1}^{n_c} u(\boldsymbol{x}_j^{(c_i)})||, \tag{2}$$

where n_{c_i} is the number of samples, $\boldsymbol{x}_j^{(c_i)}$ is the j-th sample of class c_i, and Euclidean distance is used here.

Then, the overall metric learning loss is:

$$\mathcal{L}^{DR} = \mathcal{L}_{CE}^{DR} + \lambda \mathcal{L}_{DIS}^{DR}, \tag{3}$$

where λ is the trade-off hyper-parameter.

3.4. Classifier Training

The corrected clustering result is denoted as $\{(\boldsymbol{x}_i', c_i')\}_{i=1}^N$, where c_i' is the corrected assignment with feedback. The corrected clustering results are used in classifier learning. Here, the BoW model is used to represent the word-level information. For response $\boldsymbol{r}_i$, the feature $\boldsymbol{w}_i$ is the one-hot bag-of-words feature where the length of $\boldsymbol{w}_i$ is the size of dictionary D, and $\boldsymbol{w}_{ij} = 1$ if response $\boldsymbol{r}_i$ contains the j-th word of the dictionary, and vice versa. Therefore, the augmented feature of $\boldsymbol{r}_i$ is $\boldsymbol{x}_i' \oplus \boldsymbol{w}_i$. A linear layer with softmax is used to predict the class of response according to $\boldsymbol{x}_i' \oplus \boldsymbol{w}_i$, i.e.,

$$\hat{\boldsymbol{c}}_i = \text{softmax}(\boldsymbol{W} \times (\boldsymbol{x}_i' \oplus \boldsymbol{w}_i) + \boldsymbol{b}), \tag{4}$$

where $\boldsymbol{W}$ and $\boldsymbol{b}$ are trainable parameters of linear layer and $\text{softmax}(\boldsymbol{x}_i) = \frac{\exp(\boldsymbol{x}_i)}{\sum_j \exp(\boldsymbol{x}_j)}$.

With the corrected clustering result, the cross-entropy loss function of this multi-class classification task is as follows:

$$\mathcal{L}_{CE}^{MC} = \sum_{i=1}^{N} \sum_{j=1}^{C} \boldsymbol{c}_{ij}' \log \hat{\boldsymbol{c}}_{ij}, \tag{5}$$

where $\boldsymbol{c}_i'$ is the one-hot version of c_i', e.g., $\boldsymbol{c}_i' = [0, 0, 1, 0]$ means that $c_i' = 3$.

After training, the weight matrix $\boldsymbol{W}$ is used to discover the indicator words of each class as inspired by the study in [45]. Specifically, the weight matrix is decomposed into $\boldsymbol{W}_x$ and $\boldsymbol{W}_w$, corresponding to the semantic feature $\boldsymbol{x}'$ and bag-of-words feature $\boldsymbol{w}$, respectively. Therefore, we rewrite the weight matrix as follows:

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{W}_x^{(1)} & \cdots & \boldsymbol{W}_x^{(C)} \\ \boldsymbol{W}_w^{(1)} & \cdots & \boldsymbol{W}_w^{(C)} \end{bmatrix}. \tag{6}$$

The weight $\boldsymbol{W}_w^{(i)}$ with size $D \times 1$ is the word-level proxy of class i where each element represents the importance of the corresponding word. Therefore, for each class i, assign the top-n words with the highest absolute values in $\boldsymbol{W}_w^{(i)}$ as indicator words. These explanation results can then be used to measure the clustering quality and help experts review and score these equivalent response classes.

3.5. Expert Scoring for Equivalent Response Classes

Due to the indicator words being insufficient for experts to score, we select several representative responses for each equivalent class. In the dimension-reduced semantic space, the top-k samples closest to the centroid will be selected. For the equation of centroid, see Equation (2).

The indicator words and representative responses of class i are provided for M experts. After scoring, we report the inter-rater reliability. Only if the reliability requirement is satisfied can the score be said to be consistent and stored in the database. For each item, inter-rater reliability is measured at the rater level and the code level. At the rater level, the cohen's kappa is adopted between raters, and the average value of rater pairs is reported. At the code level, the variance of its scores is adopted, and the average value of codes is reported.

Finally, the coding response pool can help automatically score new responses. The stored average score can be applied directly if a response is similar to an existing equivalent response class. Only new responses that do not match any class in the response pool are assigned to experts. However, determining whether a response is entirely new is difficult. The out-of-distribution detection is used to tackle this challenge [46,47]. Specifically, for a response $\boldsymbol{r}_i$, assume that the predicted class is c. The similarity between the augmented feature $\boldsymbol{x}'_i \oplus \boldsymbol{w}_i$ and class proxy $\boldsymbol{W}_x^{(c)} \oplus \boldsymbol{W}_w^{(c)}$ is considered as the confidence:

$$\text{Sim} = \frac{< \boldsymbol{x}'_i \oplus \boldsymbol{w}_i, \boldsymbol{W}_x^{(c)} \oplus \boldsymbol{W}_w^{(c)} >}{||\boldsymbol{x}'_i \oplus \boldsymbol{w}_i|| \cdot ||\boldsymbol{W}_x^{(c)} \oplus \boldsymbol{W}_w^{(c)}||}, \tag{7}$$

where $< \cdot, \cdot >$ refers to dot product.

If the confidence of the multi-class classification is higher than the given threshold, they can be determined to be similar responses. In all, the pseudo-code of the ASSIST system is provided in Algorithm 1.

Algorithm 1 The pseudo-code of the ASSIST system.

Input: Raw responses
Output: Trained classifier

```
Pre-processing training responses, extract semantic features, reduce dimension, and clustering;
Experts review clustering result and provide feedback constrained set used in Equation (2);
Metric learning in UMAP as Equation (3);
Initialize training set as empty;
for cluster i: {x_j} do
    Extract the bag-of-words features {w_j} of {x_j};
    Add all samples {(w_j, i)} to the training set;
    Use training set to train the classifier as Equation (4);
    Extract the importance of each feature in the classifier and arrange it in descending order. See Section 3.4 and Equation (6);
    Take the word corresponding to the feature with the highest importance as the indication word of the cluster;
end for
Experts scoring and report inter-rater reliability;
Output the trained classifier, which is applied in new responses such as test set.
```

4. Experiments

In this section, we first introduce our real-world responses dataset, baseline methods, and evaluation metrics of experimental results are described. Then, we show the comparative results of different clustering settings. After that, we report the experimental results compared with other algorithms to demonstrate the effectiveness of the proposed method in improving scoring accuracy.

4.1. Dataset Description

We conduct experiments on our large-scale assessment dataset containing Chinese and History short-answer questions. The dataset comes from final exam questions and student responses from a high school. There are three Chinese questions and three History questions, named CHN1 to CHN3 and HIST1 to HIST3, see Figure 3 and Table 1. Among them, the CHN2, HIST1, and HIST2 are fill-in-blanks about specific knowledge points, and the CHN1, CHN3, and HIST3 are more complex reading comprehension questions. In the experiment, 80% of the data were used as the training set, 10% as the validation set, and 10% as the test set. In the validation and the test stage, all answers were scored manually by eight teachers.

Table 1. Statistics description of our dataset.

	Num. of Samples	Num. of Avg Word	Score
CHN1	2480	8.42	0-3
CHN2	2391	4.19	0-1
CHN3	2579	15.07	0-5
HIST1	892	4.81	0-2
HIST2	876	3.69	0-2
HIST3	904	7.13	0-3

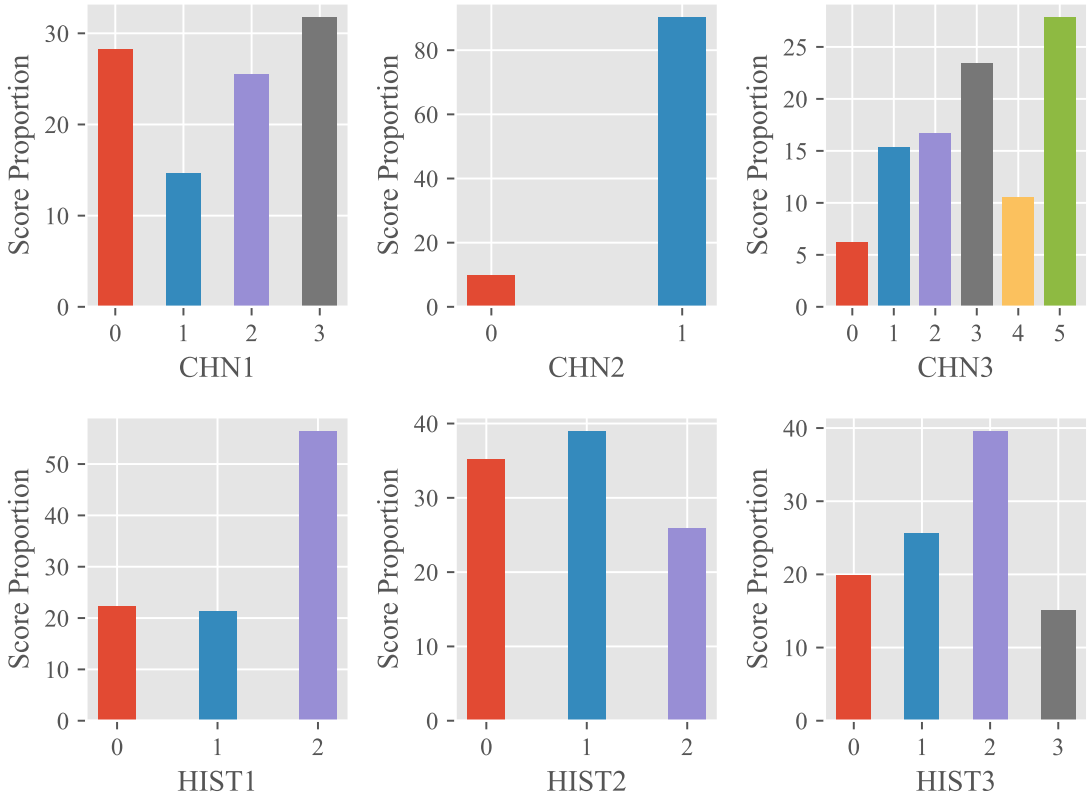

Figure 3. Score distribution of items. Orange colour refers to score 0, blue colour refers to score 1, purple colour refers to score 2, grey colour refers to score 3, yellow colour refers to score 4 and green colour refers to score 5.

4.2. Experiment Setup

4.2.1. Word Embeddings

For preprocessing, we use LAC (Lexical Analysis of Chinese) [48] word segmentation toolkit to segment words and remove stop words. We convert numbers, punctuation, and low-frequency words with a word frequency of less than 2 into specific symbols. Word embeddings and Sentence-BERT in the semantic feature extraction are fine-tuned on a large-scale word corpus. Word embeddings do not appear in the pre-trained words that were randomly initialized.

4.2.2. Parameter Settings

The word embedding dimensionality is 768, and the number of layers of the BERT is 12. In UMAP, the number of neighbors is 5. The effective scale of embedded points is 1, and the effective minimum distance between embedded points is 0.1. The hyper-parameters a and b are automatically determined by the above settings. The dimension of the space to embed into is 20. In HDBSCAN, the minimum number of clustering samples is 4. With

grid searching, the learning rate is chosen from [0.02, 0.01, 0.001, 0.0005], and λ is chosen from [0.6, 0.7, 0.8, 0.9]. After tuning on the validation set, the learning rate is 0.01, and λ in Equation (3) is 0.8 during training.

All network parameters are initialized using the Kaiming method [49], and models are trained with the Adam optimizer [50]. All models are implemented using the PyTorch library, and the experiments are conducted on a Linux server equipped with two Intel(R) Xeon(R) E5-2699 v4 CPUs and four Tesla V100 PCIe GPUs. The user interfaces of the ASSIST system are shown in Figures 4 and 5.

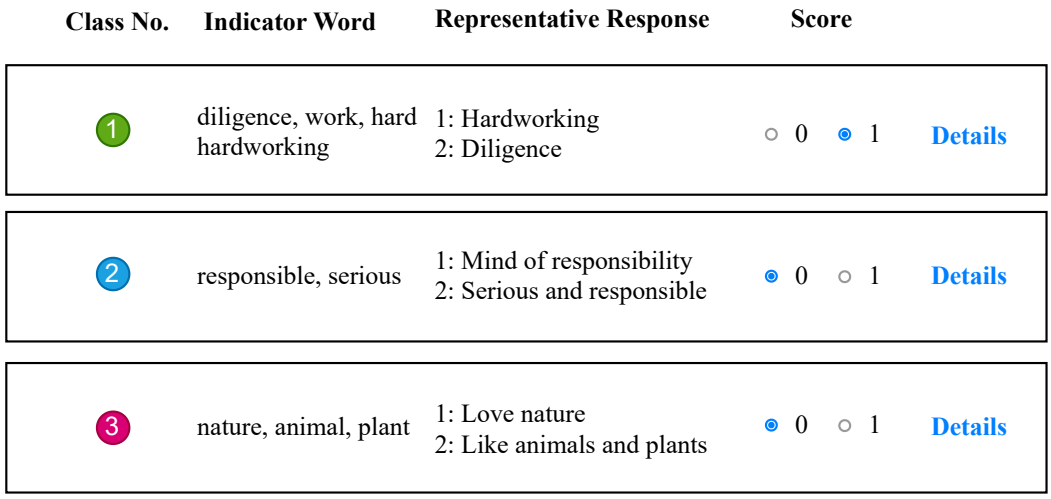

Figure 4. User interface of clustering results preview.

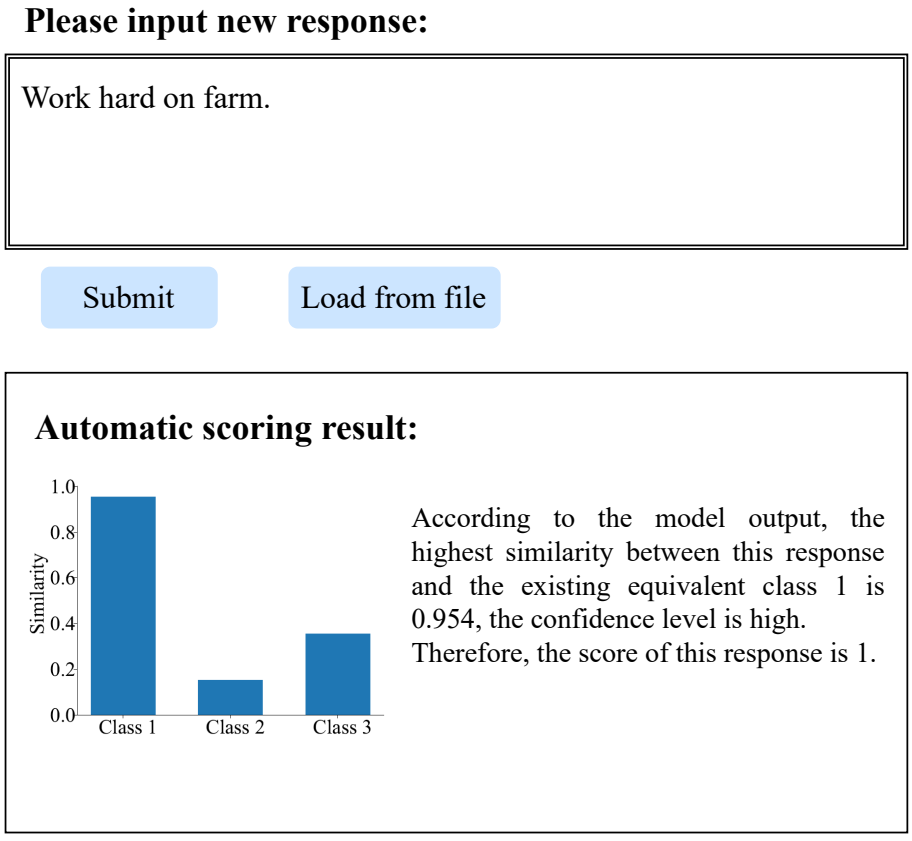

Figure 5. User interface of automatic scoring.

4.3. Performance on Response Clustering

We evaluate the response clustering performance by Silhouette Coefficient [51], which takes into account both the compactness (density) and separation (distance) of the clusters.

Baselines contain four clustering methods: K-Means, LDA, HDBSCAN, and interactive HDBSCAN. We compare the clustering effects of these four methods on the dataset with 10 different parameters, respectively. We adjust the parameter in the clustering algorithm to control the number of clusters from 5 to 90. The analysis of clustering is based on the whole dataset. The average of the clustering results of all questions is shown in Figure 6.

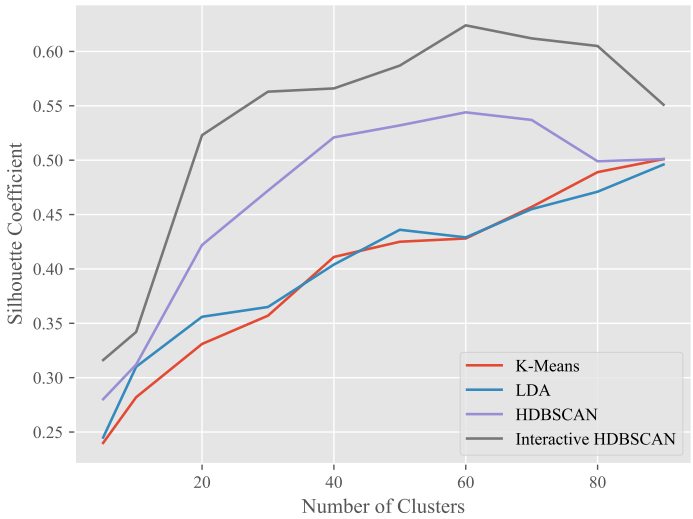

Figure 6. Comparative results of four response clustering methods.

As can be seen from Figure 6, the Silhouette Coefficient value of the interactive HDBSCAN on the dataset is much higher than other algorithms. Therefore, we choose interactive HDBSCAN as the clustering algorithm to cluster student responses. In addition, we can also find that for K-Means and LDA, the larger the number of clusters, the better the clustering effect will be, and the clustering effect is best when the number of clusters is around 60 for the HDBSCAN and Interactive HDBSCAN.

4.4. Performance on Automatic Scoring

We choose four code classification metrics and two score agreement metrics for the evaluation. The Precision is the fraction of True Positive items divided by the total number of positively predicted items. The Recall is the fraction of True Positive items divided by the total number of positively classified items. The micro-F1 combines Precision and recall into a single metric and is often used when significant class imbalances exist. The AUC (Area Under the ROC Curve) assesses the ability of a model to distinguish between the positive and negative classes across various probability thresholds. These reported classification metrics are the averaged values across all codes for a specific item. The RMSE (Root Mean Square Error) is the average magnitude of the errors between predicted scores and ground truth scores. The scores are normalized for the sake of comparability. Cohen's kappa $\hat{\kappa}_{hc}$ is the human-computer reliability computed in the test set.

In Figure 2c, we replace the semantic feature extractor from pre-trained Sentence-BERT to TextCNN [52], BiLSTM [53], and InferSent [25], respectively.

1. BiLTSM and TextCNN. We take BiLSTM in the semantic feature extraction stage. BiLSTM stands for Bidirectional Long Short-Term Memory. It is a type of recurrent neural network (RNN) architecture designed to capture sequential dependencies in data. The bidirectional aspect makes them particularly effective in capturing dependency patterns in temporal sequences. TextCNN utilizes convolutional layers to capture local patterns and hierarchical representations in text. The BiLSTM and

TextCNN are widely used in automatic scoring, such as the rubric-aware model [54] and the attention-based scoring model [55].

2. BERT-Refine: This method adds a semantic refinement layer after the BERT backbone to refine the semantics of the BERT outputs, which consists of a Bi-LSTM network and a Capsule network with position information in parallel [6]. We reproduce the feature extractor to our pipeline but without the proposed triple-hot loss strategy because we model the automatic coding as classification instead of regression.
3. SR-Combined: This method proposes a feature extractor based on InferSent and incorporates class-specific representatives obtained after clustering [8]. We use the InferSent, combined token-sentence similarity, as mentioned in this work, to generate the predicted code while keeping the clustering as our result for the sake of comparability.
4. Sentence-BERT: As mentioned in Section III.C, the pre-trained sentence-BERT is used as our default feature extractor, and this model is also used in recent automatic scoring work such as [56].
5. w/o Semantic and w/o bag-of-words: Based on sentence-BERT, We ablate specific modules to show the effectiveness. The w/o semantic means that we only use bag-of-words features to train the classifier, and the w/o bag-of-words means that we only use semantic features to train the classifier.

We report the scoring results of responses using the baseline methods and the proposed method on the test sets of all Chinese and History questions, respectively. The experiments are repeated five times with different random seeds, and we count the mean value and standard derivation as Table 2. The reported performance on test-stage OOD detection and inter-rater reliability is shown in Table 3. The reported value in each discipline, as shown in Table 4, is averaged across three items. As can be seen from Table 4, the proposed method performs significantly better than the baseline, i.e., "w/o interact", on both Chinese and History questions, which shows that the feedback from experts can help to improve the final performance of automatic coding by fine-tuning the representation. Additionally, the pre-trained model-based methods outperform the BiLSTM and TextCNN, which show an improvement in feature quality for large-scale corpora. At the same time, it can also be seen that the proposed method of combining bag-of-words features and semantic features for classifier training outperforms methods of using only bag-of-words features or semantic features.

Table 2. The reported performance on automatic scoring and error analysis.

Item	Precision	Recall	micro-F1	AUC	RMSE	$\hat{\kappa}_{hc}$
CHN1	0.723 ± 0.074	0.644 ± 0.066	0.752 ± 0.079	0.671 ± 0.073	0.136 ± 0.021	0.773 ± 0.087
CHN2	0.755 ± 0.069	0.657 ± 0.064	0.741 ± 0.082	0.706 ± 0.068	0.114 ± 0.030	0.826 ± 0.053
CHN3	0.724 ± 0.082	0.685 ± 0.076	0.697 ± 0.069	0.705 ± 0.063	0.152 ± 0.033	0.807 ± 0.059
HIST1	0.787 ± 0.080	0.761 ± 0.072	0.740 ± 0.077	0.808 ± 0.049	0.185 ± 0.026	0.699 ± 0.091
HIST2	0.749 ± 0.087	0.794 ± 0.068	0.764 ± 0.092	0.779 ± 0.074	0.172 ± 0.019	0.724 ± 0.072
HIST3	0.759 ± 0.081	0.719 ± 0.058	0.803 ± 0.047	0.786 ± 0.078	0.139 ± 0.029	0.750 ± 0.083

Table 3. The reported performance on test-stage OOD detection and inter-rater reliability.

	OOD Detection on Test Set				Inter-Rater Reliability	
Item	**Precision**	**Recall**	**F1**	**OOD%**	$\bar{\kappa}_{hh}$	$\bar{v}_c$
CHN1	0.727	0.667	0.696	4.84%	0.815	0.127
CHN2	0.588	0.909	0.714	4.60%	0.793	0.218
CHN3	0.625	0.769	0.689	5.04%	0.921	0.262
HIST1	0.714	0.833	0.769	6.74%	0.672	0.301
HIST2	0.571	0.800	0.667	5.68%	0.843	0.179
HIST3	0.545	0.750	0.632	8.89%	0.758	0.245

Table 4. Comparison of the proposed method and other baseline methods on four average metrics in Chinese and History questions.

	CHN						HIST					
Method	**Precision**	**Recall**	**micro-F1**	**AUC**	**RMSE**	$\hat{\kappa}_{hc}$	**Precision**	**Recall**	**micro-F1**	**AUC**	**RMSE**	$\hat{\kappa}_{hc}$
BiLSTM	0.687	0.635	0.697	0.643	0.132	0.791	0.723	0.704	0.749	0.703	0.145	0.679
w/o interact	0.663	0.475	0.594	0.617	-	-	0.618	0.488	0.538	0.544	-	-
TextCNN	0.701	0.646	0.721	0.656	0.117	0.748	0.741	0.732	0.748	0.695	0.127	0.694
w/o interact	0.571	0.463	0.618	0.536	-	-	0.611	0.564	0.572	0.471	-	-
BERT-Refine[6]	0.735	0.656	0.725	0.707	0.171	0.804	0.753	0.769	0.762	0.750	0.183	0.742
w/o interact	0.436	0.378	0.461	0.533	-	-	0.485	0.558	0.506	0.524	-	-
SR-Combined [8]	0.719	0.655	0.738	0.680	0.153	0.810	0.745	0.732	0.772	0.764	0.153	0.706
w/o interact	0.738	0.844	0.639	0.513	-	-	0.742	0.581	0.696	0.650	-	-
Sentence-BERT	0.734	0.662	0.730	0.694	0.134	0.802	0.765	0.758	0.769	0.791	0.165	0.724
w/o interact	0.721	0.723	0.679	0.683	-	-	0.699	0.651	0.870	0.784	-	-
w/o semantic	0.711	0.617	0.703	0.657	0.182	0.663	0.752	0.733	0.752	0.827	0.123	0.676
w/o bag-of-words	0.679	0.625	0.669	0.641	0.126	0.775	0.676	0.711	0.934	0.787	0.141	0.768

4.5. Discussion on Reliability and Efficiency

To measure the inter-rater reliability, we report the Cohen's kappa $\bar{\kappa}_{hh}$ between raters, and the variance $\bar{v}_c$ of the ratings of all possible codes, as shown in Table 3. The reported value is the *average*. If there are M raters, the $\bar{\kappa}_{hh}$ is the average of all $\binom{M}{2} = \frac{M(M-1)}{2}$ pairs. If there are C codes, all scores of each code construct an array, and the $\bar{v}_c$ is the average of variances of all arrays. Then, all scores are normalized to 0 to 1 to achieve comparability between reported metrics.

We find the following: (1) The average Cohen's kappa between human raters is in the range of 0.672 to 0.921, and the average variance of scores is in the range of 0.127 to 0.301; (2) The higher Cohen's kappa between human raters, the higher the average variance of scores; (3) The inter-rater reliability of CHN dataset is generally higher than HIST dataset; (4) The inter-rater reliability is positively correlated with the number of clusters and the maximum score and negatively correlated with the average length of responses.

We report the clustering before and after expert correction, as shown in Table 5. The #Clusters means the number of clusters, the #Resp. means the number of responses in each cluster, and the Diff. means the edit distance of the cluster state before and after correction, i.e., the minimum number of single-response edits. For example, before correction, there are 3 clusters $C1, C2, C3$ and 5 responses $R1, R2, R3, R4, R5$, and the state is $C1 = \{R1, R2\}, C2 = \{R3, R4\}, C3 = \{R5\}$. After correction, there are 2 clusters $C1, C2$, and the state is $C1 = \{R1, R2, R3\}, C2 = \{R4, R5\}$. The minimal edit is moving the $R3$ from $C2$ to $C1$ and moving the $R5$ from $C3$ to $C2$, therefore, the edit distance is 2. We find that as the score increases, the number of clusters shows a decreasing trend for the same number of samples, i.e., the mean number of responses decreases with the score increases. As the possible score values increase, the difference between before and after correction increases.

Table 5. The details of interactive clustering.

		Before Correction			After Correction			
Item	**Score**	**#Cluster**	**mean (#Resp.)**	**max (#Resp.)**	**#Clusters**	**mean (#Resp.)**	**max (#Resp.)**	**Diff.**
CHN1	0	-	-	-	21	26.64	38	-
	1	-	-	-	16	18.10	51	-
	2	-	-	-	17	29.76	59	-
	3	-	-	-	9	69.88	106	-
	All	61	32.52	114	63	31.49	106	67

Table 5. *Cont.*

		Before Correction			After Correction			
Item	**Score**	**#Cluster**	**mean (#Resp.)**	**max (#Resp.)**	**#Clusters**	**mean (#Resp.)**	**max (#Resp.)**	**Diff.**
CHN2	0	-	-	-	19	9.96	14	-
	1	-	-	-	26	66.29	105	-
	All	47	40.70	94	45	42.51	105	53
CHN3	0	-	-	-	13	9.85	15	-
	1	-	-	-	18	17.55	28	-
	2	-	-	-	14	24.64	33	-
	3	-	-	-	23	21.01	30	-
	4	-	-	-	9	24.09	47	-
	5	-	-	-	10	57.41	98	-
	All	87	23.71	103	87	23.71	98	185
HIST1	0	-	-	-	15	10.61	27	-
	1	-	-	-	13	11.75	23	-
	2	-	-	-	26	15.45	31	-
	All	55	12.97	31	54	13.21	31	92
HIST2	0	-	-	-	18	5.65	14	-
	1	-	-	-	22	16.34	29	-
	2	-	-	-	13	18.42	27	-
	All	54	12.98	30	53	13.22	29	64
HIST3	0	-	-	-	17	8.46	17	-
	1	-	-	-	25	7.40	18	-
	2	-	-	-	19	15.02	26	-
	3	-	-	-	6	18.18	35	-
	All	64	11.30	36	67	10.79	35	89

5. Conclusions

This research proposes a novel interactive response coding system called ASSIST to address the challenge of scoring labor intensity in large-scale assessments. The proposed system employs a two-stage process: response clustering and classifier training. First, raw responses are clustered interactively to generate specific codes with the experts' feedback while ensuring that the clusters accurately reflect the diversity and complexity of student responses. This interactive clustering process involves the creation of representative responses for each cluster, which are critical for human scoring. Then, a classifier is trained based on the clustered data. The classifier not only learns to assign scores to new responses but also extracts indicator words for each response class, thus aiding in the interpretability and transparency of the scoring process. Experts use these indicator words and representative responses to score equivalent response classes, ensuring consistency and reliability in the scoring process. Moreover, the ASSIST system incorporates out-of-distribution detection to identify and handle responses that do not fit well into any of the pre-defined clusters, further enhancing the robustness and adaptability of the system.

This system not only reduces the workforce required for scoring but also improves the accuracy and consistency of the scores. We report the metrics on automatic scoring and comparison with ablation methods without interactive clustering. The experimental results of our dataset, concluding with six items from two disciplines and eight experts, demonstrate significant improvements in clustering quality and scoring accuracy, with average gains of 14.48% and 18.94%, respectively. Additionally, we report inter-rater reliability, out-of-distribution rates, and cluster statistics before and after interaction, confirming the effectiveness and reliability of the proposed ASSIST system.

6. Limitations and Future Works

The system developed in this research is only evaluated in two disciplines, Chinese and History, and all responses are texts without formulas or images. This limits our approach to semantic understanding of short texts. We believe the method can be extended to other language datasets with appropriate multilingual pre-trained language models. However, understanding logical information such as formulas remains difficult. Related work on the chain of thought [57] may be helpful. The chain of thought prompt enhances the LLMs' logical ability and is applied in automatic math problem-solving.

Moreover, due to human resource constraints, once the experts have scored it once, they will have experience with the responses of items. This results in many variables, such as the epochs of interactions, being difficult to control. It is also difficult to apply other automatic coding methods to conduct an adequate performance comparison since this requires repeated manual coding and scoring of the same responses. Additionally, our students and coding experts come from the same school and have similar backgrounds, which makes it challenging to conduct a complete analysis of endogenous variables.

In future research, we aim to refine and extend the developed system by exploring the following directions. First, we intend to enhance the models employed in the classifier, such as the Bayesian neural network, to improve further its capacity to handle diverse responses in a more explainable way. Second, we aim to introduce adaptive clustering strategies that can dynamically adjust to evolving patterns in student responses. Finally, designing prompts for LLMs such as GPT-4 with experts' feedback is also a potential direction.

Author Contributions: Conceptualization, L.L. and W.P.; methodology, L.L., H.Y. and W.P.; software, H.Y. and W.P.; validation, L.L., H.Y. and Z.L.; formal analysis, H.Y.; investigation, L.L.; resources, L.L.; data curation, H.Y.; writing—original draft preparation, L.L. and H.Y.; writing—review and editing, Z.L. and W.P.; visualization, H.Y.; supervision, W.P. and Z.L.; project administration, L.L.; funding acquisition, L.L. and Z.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the Science and Technology Development Fund (FDCT) [Grant number: 0029/2023/RIA1]; the National Social Science Foundation of China [Grant numbers: 22&ZD009, 22CDJ030]; and the China Scholarship Council [Grant number: 201807000122].

Data Availability Statement: The original contributions presented in the study are included in the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Bennett, R.E. On the meanings of constructed response. *ETS Res. Rep. Ser.* **1991**, *1991*, i–46. [CrossRef]
2. Gao, R.; Merzdorf, H.E.; Anwar, S.; Hipwell, M.C.; Srinivasa, A.R. Automatic assessment of text-based responses in post-secondary education: A systematic review. *Comput. Educ. Artif. Intell.* **2024**, *6*, 100206. [CrossRef]
3. del Gobbo, E.; Guarino, A.; Cafarelli, B.; Grilli, L.; Limone, P. Automatic evaluation of open-ended questions for online learning. A systematic mapping. *Stud. Educ. Eval.* **2023**, *77*, 101258. [CrossRef]
4. Wang, F.; Huang, Z.; Liu, Q.; Chen, E.; Yin, Y.; Ma, J.; Wang, S. Dynamic cognitive diagnosis: An educational priors-enhanced deep knowledge tracing perspective. *IEEE Trans. Learn. Technol.* **2023**, *16*, 306–323. [CrossRef]
5. Abbas, M.; van Rosmalen, P.; Kalz, M. A data-driven approach for the identification of features for automated feedback on academic essays. *IEEE Trans. Learn. Technol.* **2023**, *16*, 914–925. [CrossRef]
6. Zhu, X.; Wu, H.; Zhang, L. Automatic short-answer grading via Bert-based deep neural networks. *IEEE Trans. Learn. Technol.* **2022**, *15*, 364–375. [CrossRef]
7. Zehner, F.; Sälzer, C.; Goldhammer, F. Automatic coding of short text responses via clustering in educational assessment. *Educ. Psychol. Meas.* **2015**, *76*, 280–303. [CrossRef]
8. Marvaniya, S.; Saha, S.; Dhamecha, T.I.; Foltz, P.; Sindhgatta, R.; Sengupta, B. Creating scoring rubric from representative student answers for improved short answer grading. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, New York, NY, USA, 22–26 October 2018; pp. 993–1002. [CrossRef]
9. Bae, J.; Helldin, T.; Riveiro, M.; Nowaczyk, S.; Bouguelia, M.R.; Falkman, G. Interactive clustering: A comprehensive review. *ACM Comput. Surv.* **2020**, *53*, 1–39. [CrossRef]

10. Noorbehbahani, F.; Kardan, A. The automatic assessment of free text answers using a modified BLEU algorithm. *Comput. Educ.* **2011**, *56*, 337–345. [CrossRef]
11. Ren, J.; Liu, P.J.; Fertig, E.; Snoek, J.; Poplin, R.; Depristo, M.; Dillon, J.; Lakshminarayanan, B. Likelihood ratios for out-of-distribution detection. In Proceedings of the Advances in Neural Information Processing Systems 32: NeurIPS 2019, Vancouver, BC, Canada, 8–14 December 2019; pp. 1–12.
12. Nelson, L.K. Computational grounded theory: A methodological framework. *Sociol. Methods Res.* **2020**, *49*, 3–42. [CrossRef]
13. Burrows, S.; Gurevych, I.; Stein, B. The eras and trends of automatic short answer grading. *Int. J. Artif. Intell. Educ.* **2014**, *25*, 60–117. [CrossRef]
14. Sukkarieh, J.Z.; Blackmore, J. c-rater: Automatic content scoring for short constructed responses. In Proceedings of the Twenty-Second International FLAIRS Conference, Sanibel Island, FL, USA, 19–21 March 2009; pp. 290–295.
15. Sultan, M.A.; Salazar, C.; Sumner, T. Fast and easy short answer grading with high accuracy. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 12–17 June 2016; pp. 1070–1075. [CrossRef]
16. Tan, H.; Wang, C.; Duan, Q.; Lu, Y.; Zhang, H.; Li, R. Automatic short answer grading by encoding student responses via a graph convolutional network. *Interact. Learn. Environ.* **2023**, *31*, 1636–1650. [CrossRef]
17. Schneider, J.; Schenk, B.; Niklaus, C.; Vlachos, M. Towards llm-based auto-grading for short textual answers. In Proceedings of the 16th International Conference on Computer Supported Education, Angers, France, 2–4 May 2024. [CrossRef]
18. Chang, L.H.; Ginter, F. Automatic short answer grading for Finnish with ChatGPT. In Proceedings of the 39th AAAI Conference on Artificial Intelligence, vancouver, BC, Canada, 20–27 February 2024; Volume 38, pp. 23173–23181.
19. Min, Q.; Zhou, Z.; Li, Z.; Wu, M. Automatic evaluation of instructional videos based on video features and student watching experience. *IEEE Trans. Learn. Technol.* **2024**, *17*, 54–62. [CrossRef]
20. Zesch, T.; Heilman, M.; Cahill, A. Reducing annotation efforts in supervised short answer scoring. In Proceedings of the 10th Workshop on Innovative Use of NLP for Building Educational Applications, Denver, CO, USA, 4 June 2015; pp. 124–132.
21. Andersen, N.; Zehner, F.; Goldhammer, F. Semi-automatic coding of open-ended text responses in large-scale assessments. *J. Comput. Assist. Learn.* **2022**, *39*, 841–854. [CrossRef]
22. Zhang, Y.; Jin, R.; Zhou, Z.H. Understanding bag-of-words model: A statistical framework. *Int. J. Mach. Learn. Cybern.* **2010**, *1*, 43–52. [CrossRef]
23. Ramos, J. Using TF-IDF to determine word relevance in document queries. In Proceedings of the First Instructional Conference on Machine Learning, Piscataway, NJ, USA, 3–8 December 2003; pp. 29–48.
24. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186. [CrossRef]
25. Conneau, A.; Kiela, D.; Schwenk, H.; Barrault, L.; Bordes, A. Supervised learning of universal sentence representations from natural language inference data. *arXiv* **2017**, arXiv:1705.02364. [CrossRef].
26. Van der Maaten, L.; Hinton, G. Visualizing data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
27. McInnes, L.; Healy, J.; Melville, J. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv* **2018**, arXiv:1802.03426. [CrossRef].
28. Zhang, D.; Zhou, Z.H.; Chen, S. Semi-supervised dimensionality reduction. In Proceedings of the 2007 SIAM International Conference on Data Mining, Minneapolis, MN, USA, 26–28 April 2007; pp. 629–634. [CrossRef]
29. Erisoglu, M.; Calis, N.; Sakallioglu, S. A new algorithm for initial cluster centers in k-means algorithm. *Pattern Recognit. Lett.* **2011**, *32*, 1701–1705. [CrossRef]
30. Jelodar, H.; Wang, Y.; Yuan, C.; Feng, X.; Jiang, X.; Li, Y.; Zhao, L. Latent Dirichlet allocation (LDA) and topic modeling: Models, applications, a survey. *Multimed. Tools Appl.* **2019**, *78*, 15169–15211. [CrossRef]
31. Schubert, E.; Sander, J.; Ester, M.; Kriegel, H.P.; Xu, X. DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN. *ACM Trans. Database Syst.* **2017**, *42*, 1–21. [CrossRef]
32. Murtagh, F.; Contreras, P. Algorithms for hierarchical clustering: An overview. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2011**, *2*, 86–97. [CrossRef]
33. Malzer, C.; Baum, M. A hybrid approach to hierarchical density-based cluster selection. In Proceedings of the 2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), Karlsruhe, Germany, 14–16 September 2020; pp. 223–228. [CrossRef]
34. Pedrycz, W. Computing and clustering in the environment of order-2 information granules. *IEEE Trans. Cybern.* **2023**, *53*, 5414–5423. [CrossRef] [PubMed]
35. Pedrycz, W. Proximity-based clustering: A search for structural consistency in data with semantic blocks of features. *IEEE Trans. Fuzzy Syst.* **2013**, *21*, 978–982. [CrossRef]
36. Pedrycz, W. Collaborative fuzzy clustering. *Pattern Recognit. Lett.* **2002**, *23*, 1675–1686. [CrossRef]
37. Ju, W.; Yi, S.; Wang, Y.; Long, Q.; Luo, J.; Xiao, Z.; Zhang, M. A survey of data-efficient graph learning. In Proceedings of the 33rd International Joint Conference on Artificial Intelligence, Jeju, Republic of Korea, 3–9 August 2024; pp. 8104–8113.

38. Ju, W.; Gu, Y.; Chen, B.; Sun, G.; Qin, Y.; Liu, X.; Luo, X.; Zhang, M. GLCC: A general framework for graph-level clustering. In Proceedings of the AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2023; Volume 37, pp. 4391–4399.
39. Luo, X.; Ju, W.; Qu, M.; Gu, Y.; Chen, C.; Deng, M.; Hua, X.S.; Zhang, M. CLEAR: Cluster-enhanced contrast for self-supervised graph representation learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2024**, *35*, 899–912. [CrossRef]
40. Yi, S.; Ju, W.; Qin, Y.; Luo, X.; Liu, L.; Zhou, Y.; Zhang, M. Redundancy-free self-supervised relational learning for graph clustering. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, 1–15. [CrossRef]
41. Reimers, N.; Gurevych, I. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv* **2019**, arXiv:1908.10084. [CrossRef].
42. Aggarwal, C.C.; Hinneburg, A.; Keim, D.A. On the surprising behavior of distance metrics in high dimensional space. In Proceedings of the 8th International Conference on Database Theory: ICDT 2001, London, UK, 4–6 October 2001; pp. 420–434. [CrossRef]
43. Damrich, S.; Hamprecht, F.A. On UMAP's true loss function. In Proceedings of the 35th Conference on Neural Information Processing Systems: NeurIPS 2021, Virtual Conference, 6–14 December 2021; pp. 5798–5809.
44. McInnes, L.; Healy, J.; Astels, S. Hdbscan: Hierarchical density based clustering. *J. Open Source Softw.* **2017**, *2*, 205. [CrossRef]
45. Guan, R.; Zhang, H.; Liang, Y.; Giunchiglia, F.; Huang, L.; Feng, X. Deep feature-based text clustering and its explanation. *IEEE Trans. Knowl. Data Eng.* **2022**, *34*, 3669–3680. [CrossRef]
46. Zhang, Z.; Xiang, X. Decoupling maxlogit for out-of-distribution detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17–24 June 2023; pp. 3388–3397.
47. Sun, Y.; Ming, Y.; Zhu, X.; Li, Y. Out-of-distribution detection with deep nearest neighbors. In Proceedings of the 39th International Conference on Machine Learning, PMLR 162, Baltimore, MD, USA, 17–23 July 2022; pp. 20827–20840.
48. Jiao, Z.; Sun, S.; Sun, K. Chinese lexical analysis with deep Bi-GRU-CRF network. *arXiv* **2018**, arXiv:1807.01882. [CrossRef].
49. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1026–1034. [CrossRef]
50. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980. [CrossRef].
51. Rousseeuw, P.J. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.* **1987**, *20*, 53–65. [CrossRef]
52. Kim, Y. Convolutional neural networks for sentence classification. *arXiv* **2014**, arXiv:1408.5882. [CrossRef].
53. Zhou, P.; Shi, W.; Tian, J.; Qi, Z.; Li, B.; Hao, H.; Xu, B. Attention-based bidirectional long short-term memory networks for relation classification. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; pp. 207–212.
54. Wang, T.; Inoue, N.; Ouchi, H.; Mizumoto, T.; Inui, K. Inject rubrics into short answer grading system. In Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP: DeepLo 2019, Hong Kong, China, 3 November 2019; pp. 175–182.
55. Qi, H.; Wang, Y.; Dai, J.; Li, J.; Di, X. Attention-based hybrid model for automatic short answer scoring. In Proceedings of the 11th International Conference on Simulation Tools and Techniques: SIMUtools 2019, Chengdu, China, 8–10 July 2019; pp. 385–394.
56. Condor, A.; Litster, M.; Pardos, Z. Automatic short answer grading with SBERT on out-of-sample questions. In Proceedings of the 14th Iteration of the Conference, Educational Data Mining, Virtual Conference, 29 June 2021; pp. 1–8.
57. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q.V.; Zhou, D. Chain-of-thought prompting elicits reasoning in large language models. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 24824–24837.