Behaviormetrika (2021) 48:459–484
https://doi.org/10.1007/s41237-021-00142-y

REVIEW PAPER

A review of deep-neural automated essay scoring models

Masaki Uto[1]

Received: 18 June 2021 / Accepted: 8 July 2021 / Published online: 20 July 2021

Abstract
Automated essay scoring (AES) is the task of automatically assigning scores to essays as an alternative to grading by humans. Although traditional AES models typically rely on manually designed features, deep neural network (DNN)-based AES models that obviate the need for feature engineering have recently attracted increased attention. Various DNN-AES models with different characteristics have been proposed over the past few years. To our knowledge, however, no study has provided a comprehensive review of DNN-AES models while introducing each model in detail. Therefore, this review presents a comprehensive survey of DNN-AES models, describing the main idea and detailed architecture of each model. We classify the AES task into four types and introduce existing DNN-AES models according to this classification.

Keywords Automated essay scoring · Deep neural networks · Natural language processing · Educational/psychological measurement

1 Introduction

Essay-writing tests have attracted much attention as a means to measuring practical and higher-order abilities such as logical thinking, critical reasoning, and creative thinking in various assessment fields (Abosalem 2016; Bernardin et al. 2016; Liu et al. 2014; Rosen and Tager 2014; Schendel and Tolmie 2017). In essay-writing tests, examinees write an essay about a given topic, and human raters grade those essays. However, essay grading is an expensive and time-consuming process, especially when there are many examinees (Hussein et al. 2019; Ke and Ng 2019). In addition, grading by human raters is not always consistent among and within raters (Eckes 2015; Hua and Wind 2019; Kassim 2011; Myford

Communicated by Kazuo Shigemasu.

✉ Masaki Uto
uto@ai.lab.uec.ac.jp

[1] The University of Electro-Communications, Tokyo, Japan

and Wolfe 2003; Rahman et al. 2017; Uto and Ueno 2018a). One approach to resolving this problem is automated essay scoring (AES), which utilizes natural language processing (NLP) and machine learning techniques to automatically grade essays.

Many AES models have been developed over recent decades, and these can generally be classified as feature-engineering or automatic feature extraction approaches (Hussein et al. 2019; Ke and Ng 2019).

AES models based on the feature-engineering approach predict scores using textual features that are manually designed by human experts (e.g., Dascalu et al. 2017; Mark and Shermis 2016; Nguyen and Litman 2018). Typical features include essay length and the number of grammatical and spelling errors. The AES model first calculates these types of textual features from a target essay, then inputs the feature vector into a regression or classification model and outputs a score. Various models based on this approach have long been proposed (e.g., Nguyen and Litman 2018; Attali and Burstein 2006; Phandi et al. 2015; Beigman Klebanov et al. 2016; Cozma et al. 2018). For example, e-rater (Attali and Burstein 2006) is a representative model that was developed and has been used by the Educational Testing Service. Another recent popular model is the Enhanced AI Scoring Engine (Phandi et al. 2015), which achieved high performance in the Automated Student Assessment Prize (ASAP) competition run by Kaggle.

The advantages of feature-engineering approach models include interpretability and explainability. However, this approach generally requires extensive effort in engineering and tuning features to achieve high scoring accuracy for a target collection of essays. To obviate the need for feature engineering, automatic feature extraction approach models based on deep neural networks (DNNs) have recently attracted attention. Many DNN-AES models have been proposed over the last five years and have achieved state-of-the-art accuracy (e.g., Alikaniotis et al. 2016; Taghipour and Ng 2016; Dasgupta et al. 2018; Farag et al. 2018; Jin et al. 2018; Mesgar and Strube 2018; Wang et al. 2018; Mim et al. 2019; Nadeem et al. 2019; Uto et al. 2020; Ridley et al. 2021). The purpose of this paper is to review these DNN-AES models.

Several recent studies have reviewed AES models (Ke and Ng 2019; Hussein et al. 2019; Borade and Netak 2021). For example, Ke and Ng (2019) reviewed various AES models, including both feature-engineering approach models and DNN-AES models. However, because the purpose of their study was to present an overview of major milestones reached in AES research since its inception, they provided only a short summary of each DNN-AES model. Another review (Hussein et al. 2019) explained some DNN-AES models in detail, but only a few models were introduced. Borade and Netak (2021) also reviewed AES models, but they focused on feature-engineering approach models.

To our knowledge, no study has provided a comprehensive review of DNN-AES models while introducing each model in detail. Therefore, this review presents a comprehensive survey of DNN-AES models, describing the main idea and detailed architecture of each model. We classify AES tasks into four types according to recent findings (Li et al. 2020; Ridley et al. 2021), and introduce existing DNN-AES models according to this classification.

2 Automated essay scoring tasks

AES tasks are generally classified into the following four types (Li et al. 2020; Ridley et al. 2021).

1. *Prompt-specific holistic scoring* This is the most common AES task type, whereby an AES model is trained using rated essays that have holistic scores and have been written for a prompt. This trained model is used to predict the scores of essays written for the same prompt. Note that a prompt refers to an essay topic or a writing task that generally consists of reading materials and a task instruction.
2. *Prompt-specific trait scoring* This task involves predicting multiple trait-specific scores for each essay in a prompt-specific setting in which essays used for model training and unrated target essays are written for the same prompt. Such scoring is often required when an analytic rubric is used to provide more detailed feedback for educational purposes.
3. *Cross-prompt holistic scoring* In this task, an AES model is trained using rated essays with holistic scores written for non-target prompts and the trained model is transferred to a target prompt. This task has recently attracted attention because it is difficult to obtain a sufficient number of rated essays written for a target prompt in practice. This task includes a zero-shot setting in which rated essays written for a target prompt do not exist, and another setting in which a relatively small number of rated essays written for a target prompt can be used. The cross-prompt AES task relates to domain adaptation and transfer learning tasks, which are widely studied in machine learning fields.
4. *Cross-prompt trait scoring* This task involves predicting multiple trait-specific scores for each essay in a cross-prompt setting in which essays written for non-target prompts are used to train an AES model.

In the following section, we review representative DNN-AES models for each task type. Table 1 summarizes the models introduced in this paper.

3 Prompt-specific holistic scoring

This section introduces DNN-AES models for prompt-specific holistic scoring.

3.1 RNN-based model

One of the first DNN-AES models was a recurrent neural network (RNN)-based model proposed by Taghipour and Ng (2016). This model predicts a score for a given essay, defined as a sequence of words, by following multi-layered neural networks whose architecture is shown in Fig. 1.

Table 1 Summary of DNN-AES models classified into four task types

1. Prompt-specific holistic scoring
 –RNN-based models (Taghipour and Ng 2016; Alikaniotis et al. 2016)
 –Hierarchical representation models (Dong and Zhang 2016; Dong et al. 2017),
 –Coherence models (Tay et al. 2018; Li et al. 2018; Farag et al. 2018; Mesgar and Strube 2018; Yang and Zhong 2021),
 –BERT-based models (Nadeem et al. 2019; Uto et al. 2020; Rodriguez et al. 2019; Yang et al. 2020; Mayfield and Black 2020),
 –Hybrid models (Dasgupta et al. 2018; Uto et al. 2020)
 –Robust model (Uto and Okano 2020)
 –Integrating multiple AES models (Aomi et al. 2021)
2. Prompt-specific trait scoring
 –Multiple trait-specific models (Mathias et al. 2020)
 –Model with multiple output modules (Hussein et al. 2020)
3. Cross-prompt holistic scoring
 –Two-stage learning models (Jin et al. 2018; Li et al. 2020),
 –Multi-stage pre-training approach model (Song et al. 2020)
 –Model with self-supervised learning (Cao et al. 2020)
4. Cross-prompt trait scoring
 –Multiple trait-specific models with self-supervised learning (Mim et al. 2019)
 –Model with multiple output modules (Ridley et al. 2021)

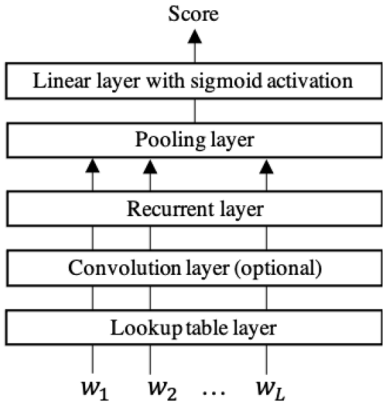

Fig. 1 Architecture of RNN-based model

– *Lookup table layer* This layer transforms each word in a given essay into a G-dimensional word-embedding representation. Word-embedding representation is a real-valued fixed-length vector of a word, in which words with similar meaning have similar vectors. Suppose $\mathcal{V}$ is a vocabulary list for essay collection, $\boldsymbol{w}_t$ represents a $|\mathcal{V}|$-dimensional one-hot representation of t-th word w_t in a given essay, and $\boldsymbol{A}$ represents a $G \times |\mathcal{V}|$-dimensional trainable embeddings matrix. Then, the embedding representation $\tilde{\boldsymbol{w}}_t$ corresponding to w_t is calculable as a dot product $\tilde{\boldsymbol{w}}_t = \boldsymbol{A} \cdot \boldsymbol{w}_t$.

– *Convolution layer* This layer captures local textual dependencies using convolution neural networks (CNNs) from a sequence of word-embedding vectors. Given an input sequence $\{\tilde{\boldsymbol{w}}_1, \tilde{\boldsymbol{w}}_2, \ldots, \tilde{\boldsymbol{w}}_L\}$ (where L is the number of words in a given essay), this layer is applied to a window of c words to capture local textual dependencies among c-gram words. Concretely, the t-th output of this layer is calculable as follows.

$$f(\mathbf{W}_c \cdot [\tilde{\boldsymbol{w}}_t, \tilde{\boldsymbol{w}}_{t+1}, \ldots, \tilde{\boldsymbol{w}}_{t+c-1}] + b_c), \tag{1}$$

where $\mathbf{W_c}$ and b_c are trainable weight and bias parameters, and $[\cdot, \cdot]$ means the concatenation of the given elements. Zero padding is applied to outputs from this layer to preserve the input and output sequence lengths. This is an optional layer that has often been omitted in recent studies.

– *Recurrent layer* This layer generally uses a long short-term memory (LSTM) network, a representative RNN, that outputs a vector at each timestep while capturing time series dependencies in an input sequence. A single-layer unidirectional LSTM is generally used, but bidirectional or multilayered LSTMs are also often used.

– *Pooling layer* This layer transforms the output hidden vector sequence of the recurrent layer $\{\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_L\}$ (where $\boldsymbol{h}_t$ represents the hidden vector of the t-th output of the recurrent layer) into an aggregated fixed-length hidden vector. Mean-over-time pooling, which calculates an average vector

$$\tilde{\boldsymbol{h}} = \frac{1}{L} \sum_{t=1}^{L} \boldsymbol{h}_t, \tag{2}$$

is generally used because it tends to provide stable accuracy. Other frequently used pooling methods include the last pool (Alikaniotis et al. 2016), which uses the last output of the recurrent layer $\boldsymbol{h}_L$, and an attention pooling layer (Dong et al. 2017), which we explain later in the present study.

– *Linear layer with sigmoid activation* This layer projects a pooling layer output onto a scalar value in the range [0, 1] by utilizing the sigmoid function as

$$\sigma(\mathbf{W}_o \cdot \tilde{\boldsymbol{h}} + b_o), \tag{3}$$

where $\mathbf{W}_o$ is a weight matrix and b_o represents bias parameters. $\sigma()$ represents the sigmoid function.

For model training, the mean-squared error (MSE) between predicted and gold-standard scores is generally used as the loss function. Specifically, letting y_n be the gold-standard score for n-th essay and letting $\hat{y}_n$ be the predicted score, the MSE loss function is defined as

$$\frac{1}{N} \sum_{n=1}^{N} (y_n - \hat{y}_n)^2, \tag{4}$$

where N is the number of essays. Note that the model training is conducted after normalizing gold standard scores to [0, 1], but the predicted scores are linearly rescaled to the original score range in the prediction phase.

3.2 RNN-based model with score-specific word embedding

Alikaniotis et al. (2016) also proposed a similar RNN-based model consisting of three layers, namely, a lookup table layer, a recurrent layer, and a pooling layer. The model uses a bidirectional LSTM for the recurrent layer and the last pooling for the pooling layer. The unique feature of this model is the use of score-specific word embedding (SSWE), which is an extension of Collobert & Weston (C&W) word-embedding (Collobert and Weston 2008), in the lookup table layer.

Suppose we train a representation for a target word w_t within a sequence of one-hot encoded words $\boldsymbol{S} = \{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_t, \ldots \boldsymbol{w}_L\}$. To derive this representation, the C&W word-embedding model learns to distinguish between the original sequence $\boldsymbol{S}$ and an artificially created noisy sequence $\boldsymbol{S}'$ in which the target word is substituted for a randomly selected word. Given a trainable embedding matrix $\boldsymbol{A}$, the model concatenates the embedding representation vectors of the words in the sequence, that is, $\tilde{\boldsymbol{S}} = [\boldsymbol{A} \cdot \boldsymbol{w}_1, \boldsymbol{A} \cdot \boldsymbol{w}_2, \ldots, \boldsymbol{A} \cdot \boldsymbol{w}_L]$. Using the vector, the C&W word-embedding model predicts whether the given word sequence $\boldsymbol{S}$ is the original sequence or a noisy one based on the following function.

$$f(\boldsymbol{S}) = \mathbf{W}_1 \cdot \text{htanh}\left(\mathbf{W}_2 \cdot \tilde{\boldsymbol{S}} + b_2\right) + b_1, \tag{5}$$

where $\mathbf{W}_1$, $\mathbf{W}_2$, b_1, and b_2 are the trainable parameters, and htanh() is the hard hyperbolic tangent function.

The SSWE model extends the C&W word-embedding model by adding another output layer that predicts essay scores as follows.

$$f_{\text{score}}(\boldsymbol{S}) = \mathbf{W}'_1 \cdot \text{htanh}\left(\mathbf{W}'_2 \cdot \tilde{\boldsymbol{S}} + b'_2\right) + b'_1, \tag{6}$$

where $\mathbf{W}'_1$, $\mathbf{W}'_2$, b'_1, and b'_2 are the trainable parameters. The SSWE model is trained while minimizing a weighted linear combination of two error loss functions, namely, a classification loss function based on Eq. (5) and a scoring error loss function based on Eq. (6).

The SSWE model provides a more effective word-embedding representation to distinguish essay qualities than does the C&W word-embedding model. Thus, Alikaniotis et al. (2016) proposed using the embedding matrix $\boldsymbol{A}$ trained by the SSWE model in the lookup table layer.

3.3 Hierarchical representation models

The models introduced above handle an essay as a linear sequence of words. Dong and Zhang (2016), however, proposed modeling the hierarchical structure of a text. Concretely, they assumed that an essay is constructed as a sequence of sentences defined as word sequences. Accordingly, they introduced a two-level hierarchical representation model consisting of a word-level CNN and a sentence-level CNN, as shown in Fig. 2. Each CNN works as explained below.

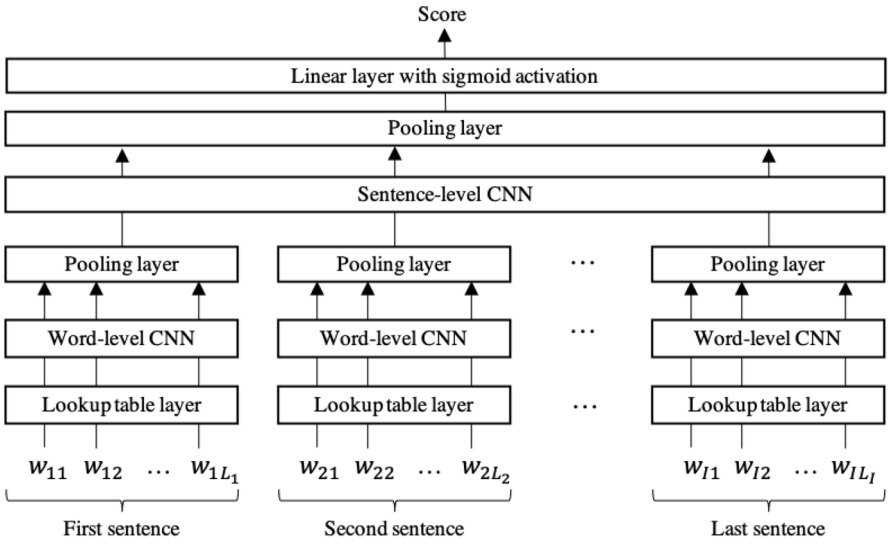

Fig. 2 Architecture of hierarchical representation model

- *Word-level CNN* The sequence of words in each sentence is processed and an aggregated vector is output, which can be taken as an embedding representation of a sentence. Suppose an essay consists of I sentences $\{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_I\}$, and each sentence is defined as a sequence of words as $\boldsymbol{s}_i = \{w_{i1}, \ldots, w_{iL_i}\}$ (where w_{it} is the t-th word in i-th sentence, and L_i is the number of words in i-th sentence). For each sentence $\boldsymbol{s}_i$, the lookup table layer transforms each word into an embedding representation, and then the word-level CNN processes the sequence of word-embedding vectors. The operation of the word-level CNN is the same as that of the convolution layer explained in Subsection 3.1. The output sequence of the word-level CNN is transformed into an aggregated fixed-length hidden vector $\tilde{\boldsymbol{h}}_{s_i}$ through a pooling layer.
- *Sentence-level CNN* This CNN takes the sequence of sentence vectors $\{\tilde{\boldsymbol{h}}_{s_1}, \ldots, \tilde{\boldsymbol{h}}_{s_I}\}$ as input and extracts n-gram level features over the sentence sequence. Then, a pooling layer transforms the CNN output sequence into an aggregated fixed-length hidden vector $\tilde{\boldsymbol{h}}$. Finally, the linear layer with sigmoid activation maps vector $\tilde{\boldsymbol{h}}$ to a score.

Dong et al. (2017) proposed another hierarchical representation model that extends the above model by using an *attention* mechanism (Bahdanau et al. 2014) to automatically identify important words and sentences. The attention mechanism is a neural architecture that enables to dynamically focus on relevant regions of input data to make predictions. The main idea of the attention mechanism is to compute a weight distribution on the input data, assigning higher values to more relevant regions. (Dong et al. 2017) uses attention-based pooling in the pooling layers. Letting the input sequence for the pooling layer be $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_J\}$, where J

indicates the sequence length, the attention mechanism aggregates the input sequence into a fixed-length vector $\tilde{\boldsymbol{x}}$ by performing the following operations.

$$\tilde{\boldsymbol{x}}_j = \tanh\left(\mathbf{W}_{a_1} \cdot \boldsymbol{x}_j + b\right) \tag{7}$$

$$a_j = \frac{\exp\left(\mathbf{W}_{a_2} \cdot \tilde{\boldsymbol{x}}_j\right)}{\sum_{j'=1}^{J} \exp\left(\mathbf{W}_{a_2} \cdot \tilde{\boldsymbol{x}}_{j'}\right)} \tag{8}$$

$$\tilde{\boldsymbol{x}} = \sum_{j=1}^{J} a_j \boldsymbol{x}_j \tag{9}$$

In these equations, $\mathbf{W}_{a_1}$, $\mathbf{W}_{a_2}$, and b are trainable parameters. $\tilde{\boldsymbol{x}}_j$ and a_j are called *an attention vector* and *an attention weight* for j-th input, respectively.

In addition to the incorporation of the attention mechanism, Dong et al. (2017) proposed adding a character-level CNN before the word-level CNN and using LSTM as an alternative to the sentence-level CNN.

3.4 Coherence modeling

Coherence is an important criterion for evaluating the quality of essays. However, the RNN-based models introduced above are known to have difficulty capturing the relationships between multiple regions in an essay because they compress a word sequence within a fixed-length hidden vector in the order they are inputted. To resolve this difficulty, several DNN-AES models that consider coherence features have been proposed (Tay et al. 2018; Li et al. 2018; Farag et al. 2018; Mesgar and Strube 2018; Yang and Zhong 2021). This subsection introduces two representative models.

3.4.1 SKIPFLOW model

Tay et al. (2018) proposed SKIPFLOW, which learns coherence features explicitly using a neural network architecture. The model is the RNN-based model with a neural tensor layer as shown in Fig. 3. The neural tensor layer takes two positional outputs of the recurrent layer that are collected from different time steps as input and computes the similarity between each of these pairs of positional outputs. Concretely, for a recurrent layer output sequence $\{\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_L\}$, the model first selects a pair of sequential outputs of width δ, that is, $\{(\boldsymbol{h}_1, \boldsymbol{h}_\delta), (\boldsymbol{h}_{\delta+1}, \boldsymbol{h}_{2\delta}), \ldots, (\boldsymbol{h}_{t\delta+1}, \boldsymbol{h}_{(t+1)\delta}), \ldots\}$. Then, each pair of hidden vectors $(\boldsymbol{h}_{t\delta+1}, \boldsymbol{h}_{(t+1)\delta})$ is input into the following neural tensor layer to return a similarity score as

$$sim\left(\boldsymbol{h}_{t\delta+1}, \boldsymbol{h}_{(t+1)\delta}\right) = \sigma\left(\mathbf{W}_u \cdot \tanh\left(\boldsymbol{h}_{t\delta+1} \cdot \mathbf{M} \cdot \boldsymbol{h}_{(t+1)\delta} + \mathbf{V} \cdot [\boldsymbol{h}_{t\delta+1}, \boldsymbol{h}_{(t+1)\delta}] + \mathbf{b}_u\right)\right),$$

where $\mathbf{W}_u$, $\mathbf{V}$, and $\mathbf{b}_u$ are the weight and bias vectors and $\mathbf{M}$ is a three-dimensional tensor. These are trainable parameters.

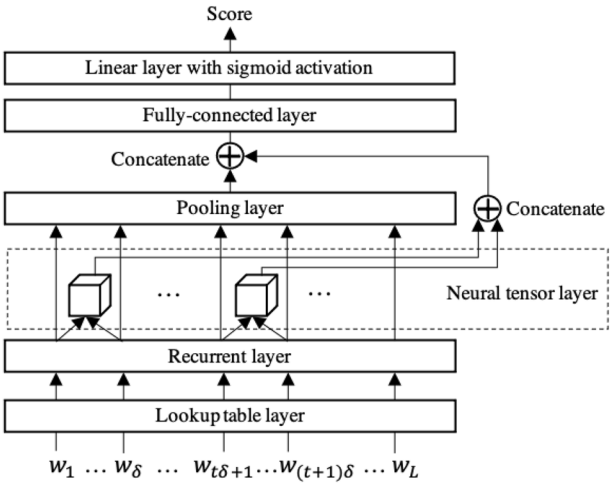

Fig. 3 Architecture of SKIPFLOW model

The similarity scores for all the pairs are concatenated with the polling layer output vector $\tilde{\boldsymbol{h}}$ as

$$\left[\tilde{\boldsymbol{h}}, sim(\boldsymbol{h}_1, \boldsymbol{h}_\delta), sim(\boldsymbol{h}_{\delta+1}, \boldsymbol{h}_{2\delta}), \ldots, sim(\boldsymbol{h}_{t\delta+1}, \boldsymbol{h}_{(t+1)\delta}), \ldots\right],$$

and the resulting vector is mapped to a score through a fully connected neural network layer and a linear layer with sigmoid activation.

3.4.2 Self-attention-based model

Li et al. (2018) proposed another model using a self-attention mechanism to capture relationships between multiple points in an essay. Self-attention mechanisms have been shown to be able to capture long-distance relationships between words in a sequence and have recently been used in various NLP tasks.

Figure 4 shows the model architecture. This model first transforms each word into an embedding representation through a lookup table layer with a position encoding, and then inputs the sequence into a multi-head self-attention model that combines multiple self-attention models in parallel. See Vaswani et al. (2017) for details of the lookup table layer with the position encoding and the multi-head self-attention architecture. The self-attention output sequence is input into a recurrent layer, a pooling layer, and a linear layer with sigmoid activation to produce an essay score.

3.5 BERT-based models

Bidirectional encoder representations from transformers (BERT), a pre-trained language model released by the Google AI Language team in 2018, has achieved state-of-the-art results in various NLP tasks (Devlin et al. 2019). Since then, BERT has

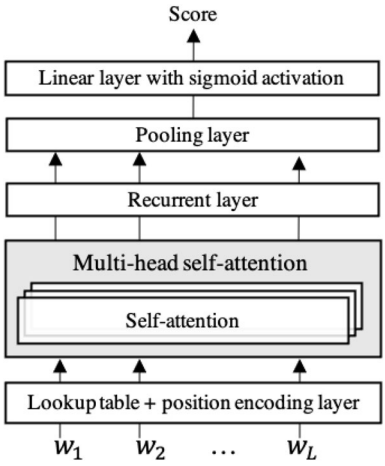

Fig. 4 Architecture of self-attention-based model

also been applied to automated text scoring tasks, including AES (Nadeem et al. 2019; Uto et al. 2020; Rodriguez et al. 2019; Yang et al. 2020; Mayfield and Black 2020) and automated short-answer grading (Liu et al. 2019; Lun et al. 2020; Sung et al. 2019), and has shown good performance.

BERT is defined as a multilayer bidirectional transformer network (Vaswani et al. 2017). Transformers are a neural network architecture designed to handle ordered sequences of data using an attention mechanism. Specifically, transformers consist of multiple layers (called transformer blocks), each containing a multi-head self-attention network and a position-wise fully connected feed-forward network. See (Vaswani et al. 2017) for details of this architecture.

BERT is trained in pre-training and fine-tuning steps. Pre-training is conducted on huge amounts of unlabeled text data over two tasks, namely, masked language modeling and next-sentence prediction. Masked language modeling is the task that predicts the identities of words that have been masked out of the input text. Next-sequence prediction is the task that predicts whether two given sentences are adjacent.

Using pre-trained BERT for a target NLP task, such as AES, requires fine-tuning (retraining), which is conducted from a task-specific supervised dataset after initializing model parameters to pre-trained values. When using BERT for AES, input essays require preprocessing, namely, adding a special token ("CLS") to the beginning of each input. BERT output corresponding to this token is used as the aggregate hidden representation for a given essay (Devlin et al. 2019). We can thus score an essay by inputting its representation into a linear layer with sigmoid activation, as illustrated in Fig. 5.

Furthermore, Yang et al. (2020) proposed fine-tuning the BERT model so that the essay scoring task and an essay ranking task are jointly resolved. As shown in Fig. 6, the proposed model is formulated as a BERT-based AES model with an additional output layer that predicts essay ranks. The model uses ListNet (Cao et al. 2007) for predicting the ranking list. This model is fine-tuned by minimizing a combination of the scoring MSE loss function and a ranking error loss function based on ListNet.

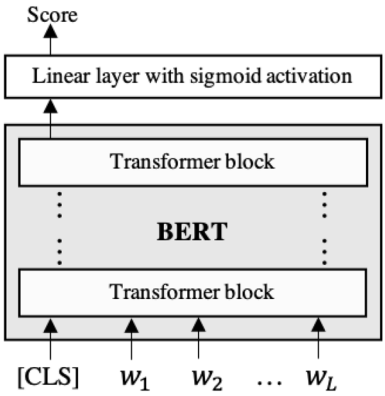

Fig. 5 Architecture of BERT-based model

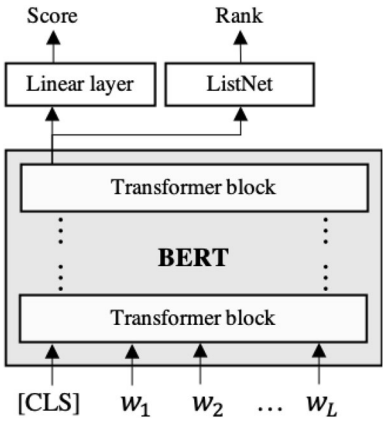

Fig. 6 Architecture of BERT-based model with ranking task

3.6 Hybrid models

The feature-engineering approach and the DNN-AES approach can be viewed as complementary rather than competing approaches (Ke and Ng 2019; Uto et al. 2020) because they provide different advantages. To receive both benefits, some hybrid models that integrate the two approaches have been proposed (Dasgupta et al. 2018; Uto et al. 2020).

One of the hybrid models is proposed by Dasgupta et al. (2018). Figure 7 shows the model architecture. As shown in the figure, it mainly consists of two DNNs. One processes word sequences in a given essay in the same way as the conventional RNN-based model (Taghipour and Ng 2016). Specifically, a word sequence is transformed into a fixed-length hidden vector $\tilde{\boldsymbol{h}}$ through a lookup table layer, a convolution layer, a recurrent layer, and a pooling layer. The other DNN processes a sequence of manually designed sentence-level features. Letting a given essay have I sentences, and letting $\boldsymbol{f}_i$ be a manually designed sentence-level feature vector for i-th sentence, the feature sequence $\{\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_I\}$ is transformed into a fixed-length hidden vector $\tilde{\boldsymbol{h}}_f$ through a convolution layer, a

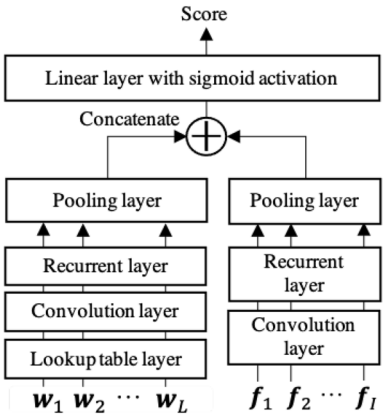

Fig. 7 Architecture of hybrid model with additional RNN for sentence-level features

recurrent layer, and a pooling layer. The model uses LSTM for the recurrent layer and attention pooling for the pooling layer. Finally, after concatenating the hidden vectors $[\tilde{\boldsymbol{h}}, \tilde{\boldsymbol{h}}_f]$, a linear layer with sigmoid activation maps it to a score.

Another hybrid model is formulated as a DNN-AES model incorporating manually designed essay-level features (Uto et al. 2020). Concretely, letting $\boldsymbol{F}$ be a manually designed essay-level feature vector, the model concatenates the feature vector with the hidden vector $\tilde{\boldsymbol{h}}$, which is obtained from a DNN-AES model. Then, a linear layer with sigmoid activation maps the concatenated vector $[\tilde{\boldsymbol{h}}, \boldsymbol{F}]$ to a score value. Figure 8 shows the architecture of this model. This hybrid model is easy to construct using various DNN-AES models.

3.7 Improving robustness for biased training data

DNN-AES models generally require a large dataset of essays graded by human raters as training data. When creating a training dataset, essay grading tasks are generally shared among many raters by assigning a few raters to each essay to lower the burden of assessment. However, in such cases, assigned scores are

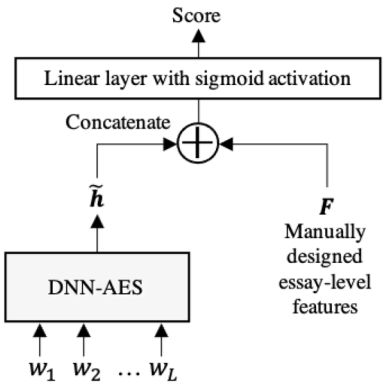

Fig. 8 Architecture of DNN-AES with handcrafted essay-level features

known to be biased owing to the effects of rater characteristics (Rahman et al. 2017; Amidei et al. 2020). The performance of AES models drops when biased data are used for model training because the resulting model reflects the bias effects (Amorim et al. 2018; Huang et al. 2019; Li et al. 2020).

To resolve this problem, Uto and Okano (2020) proposed an AES framework that integrates item response theory (IRT), a test theory based on mathematical models. Specifically, they used an IRT model incorporating parameters representing rater characteristics (e.g., Eckes 2015; Uto and Ueno 2016, 2018a) that can estimate essay scores while mitigating rater bias effects. The applied IRT model is the generalized many-facet Rasch model (Uto and Ueno 2018b, 2020) that defines the probability that rater r assigns score k to n-th essay for a prompt as

$$P_{rk}(\theta_n) = \frac{\exp \sum_{m=1}^{k} \left[\alpha_r(\theta_n - \beta_r - \beta_{rm})\right]}{\sum_{l=1}^{K} \exp \sum_{m=1}^{l} \left[\alpha_r(\theta_n - \beta_r - \beta_{rm})\right]}, \tag{10}$$

where α_r is the consistency of rater r, β_r is the severity of rater r, β_{rm} represents the strictness of rater r for category m, and K indicates the number of score categories. Furthermore, θ_n represents the latent scores for n-th essay, which removes the effects of the rater characteristics.

Using this IRT model, Uto and Okano (2020) proposed training an AES model through the following two steps. 1) Apply the IRT model to observed rating data to estimate the IRT-based score θ_n, which removes the effects of rater bias. 2) Train an AES model using the unbiased scores $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_N\}$ as the gold-standard scores based on the following loss function.

$$\frac{1}{N} \sum_{n=1}^{N} \left(\theta_n - \hat{\theta}_n\right)^2, \tag{11}$$

where $\hat{\theta}_n$ represents the AES's predicted score for n-th essay. Because the IRT-based scores are theoretically free from rater bias effects, the AES model will not reflect the bias effects.

In the prediction phase, the score for a new essay is calculated in two steps: (1) Predict the IRT score θ for the essay using a trained AES model. (2) Given θ and rater parameters, calculate the expected score, which corresponds to an unbiased original-scaled score (Uto 2019), as

$$\frac{1}{R} \sum_{r=1}^{R} \sum_{k=1}^{K} k \cdot P_{rk}(\theta), \tag{12}$$

where R indicates the number of raters who graded essays in the training data. The expected score is used as a predicted essay score, which is robust against rater biases.

3.8 Integration of AES models

Conventional AES models including those introduced above have different scoring characteristics. Therefore, integrating multiple AES models is expected to improve scoring accuracy. For these reasons, Aomi et al. (2021) proposed a framework that integrates multiple AES models while considering the characteristics of each model using IRT. In the framework, multiple AES models are first trained independently, and the trained models are used to produce prediction scores for target essays. Then, the generalized many-facet Rasch model introduced above is applied to the obtained prediction scores by regarding rater characteristic parameters, α_r, β_r, and β_{rm} as characteristic parameters of AES models. Given the estimated IRT score θ for the target essays, a predicted essay score is calculated as the expected score based on Eq. (12).

This framework can integrate prediction scores from various AES models while considering the characteristics of each model. Subsequently, it provides scores that are more accurate than those obtained by simple averaging or a single AES model.

4 Prompt-specific trait scoring

This section introduces DNN-AES models for the prompt-specific trait scoring task. Although this task is important especially for educational purposes, only a limited number of models have been proposed for the task.

4.1 Use of multiple trait-specific models

Mathias et al. (2020) presents one of the first attempts to perform prompt-specific trait scoring based on a DNN-AES model. Their study used the hierarchical representation model with an attention mechanism (Dong et al. 2017), introduced in Sect. 3.3, to predict trait-specific scores for each essay. Concretely, in their study, the AES model was trained for each trait independently, and predicted scores using trait-specific models.

4.2 Model with multiple output modules

Hussein et al. (2020) proposed a model specialized in a prompt-specific trait scoring task that can predict multiple trait scores jointly. The model is formulated as a multi-output model based on the RNN-based model (Taghipour and Ng 2016), introduced in Sect. 3.1. Concretely, as shown in Fig. 9, they extended the RNN-based model by adding as many multiple output linear layers as the number of traits. Additionally, an optional fully connected neural network layer was added after the pooling layer. The loss function is defined as a linear combination of multiple MSE loss functions as follows.

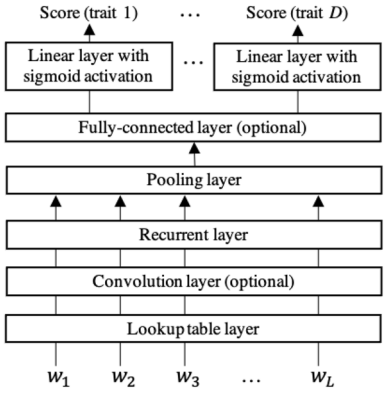

Fig. 9 Architecture of RNN-based model with multiple output layers for prompt-specific trait scoring

$$\frac{1}{ND}\left(\sum_{n=1}^{N}\sum_{d=1}^{D}(y_{nd}-\hat{y}_{nd})^2\right), \tag{13}$$

where D is the number of traits, y_{nd} and $\hat{y}_{nd}$ are the gold-standard score and predicted d-th trait score for n-th essay, respectively.

5 Cross-prompt holistic scoring

The prompt-specific scoring models introduced above assume situations in which rated training essays and unrated target essays are written for the same prompt. However, we often face situations in which we cannot use any rated essays or only a relatively small number of rated essays written for the target prompt in model training, even though we have many rated essays written for other non-target prompts. AES for such settings is generally called a cross-prompt scoring task. This section introduces cross-prompt holistic scoring models.

5.1 Two-stage learning models

One of the first cross-prompt holistic scoring models using DNN was proposed by Jin et al. (2018). The method is constructed as a two-stage DNN (TDNN) approach in which a prompt-independent scoring model is trained using rated essays for non-target prompts in the first stage, and is used to generate pseudo rating data for unrated essays in a target prompt. Then, using the pseudo rating data, a prompt-specific scoring model for the target prompt is trained in the second stage. The TDNN is detailed below.

– *First stage (Training a prompt-independent AES model)* In this stage, rated essays written for non-target prompts are used to train a prompt-independent AES model that uses manually designed prompt-independent shallow features, such as the number of typos, grammatical errors, and spelling errors.

Here, a ranking support vector machine (Joachims 2002) is used as the prompt-independent model.

– *Second stage (Training a prompt-specific AES model)* The trained prompt-independent AES model is used to produce the scores of unrated essays written for a target prompt, and the pseudo scores are used to train a prompt-specific scoring model. To train a prompt-specific scoring model, only confident essays with the highest and lowest pseudo scores are used, instead of using all the produced scores. The prompt-specific AES model in the study by Jin et al. (2018) used an extended model of the RNN-based model (Taghipour and Ng 2016) that can process three types of sequential inputs, namely, a sequence of words, part-of-speech (POS) tags, and syntactic tags.

Li et al. (2020) pointed out that the TDNN model uses a limited number of general linguistic features in the prompt-independent AES model, which may seriously affect the accuracy of the generated pseudo scores for essays in a target prompt. To extract more efficient features, they proposed another two-stage framework called a *shared and enhanced deep neural network (SEDNN)* model. The SEDNN model consists of two stages, described as follows.

- *First stage* As an alternative to a prompt-independent model with manually designed shallow linguistic features, the SEDNN uses a DNN-AES model that extends the hierarchical representation model with an attention mechanism (Dong et al. 2017), introduced in Sect. 3.3. Concretely, in the model, a new output layer is added to jointly solve the AES task and a binary classification task that distinguishes whether a given essay was written for the target prompt. The model is trained based on a combination of the loss functions for the essay scoring task and the prompt discrimination task using a dataset consisting of rated essays written for non-target prompts and the unrated essays written for the target prompt.
- *Second stage* As in the second stage of the TDNN model, scores of unrated essays written for a target prompt are generated by the prompt-independent AES model, and the pseudo scores are used to train a prompt-specific scoring model. The prompt-specific scoring model in the study by Li et al. (2020) is a Siamese network model that jointly uses the essay text and the text of the target prompt itself to learn prompt-dependent features more efficiently. In the model, an essay text is processed by a similar model to the SKIPFLOW (Tay et al. 2018) and is transformed into vector representations. The word sequence in the prompt text is also transformed into a fixed-length hidden vector representation by another neural architecture consisting of a lookup table layer, a convolution layer, a recurrent layer, and a mechanism that measures the relevance relation between the given essay and the target prompt text. After concatenating the two vector representations corresponding to an essay text and a prompt text, a linear layer with sigmoid activation maps it to a prediction score.

5.2 Multi-stage pre-training approach model

Another cross-prompt holistic scoring approach incorporates pre-training processes. In the approach, an AES model is developed by performing pre-training on a vast number of essays with or without scores written for non-target prompts, and then the model is fine-tuned using a limited number of rated essays written for a target prompt. The pre-training process enables a DNN model to capture a general language model for predicting essay quality. Thus, the use of a pre-trained model as an initial model helps in obtaining a model for a target scoring task. The BERT-based AES models explained in Sect. 3.5 are examples of the pre-training and fine-tuning approach models. In various NLP tasks, the use of pre-training has been popular and has achieved great success.

For cross-prompt holistic scoring, (Song et al. 2020) proposed training the hierarchical representation model with the attention mechanism (Dong et al. 2017), as explained in Sect. 3.3, through the following three pre-training and fine-tuning steps.

1. *Weakly supervised pre-training* The AES model is trained based on a vast number of roughly scored essays written for diverse prompts collected from the Web. The study by Song et al. (2020) assumed that binary scores are given to the essays; thus, this step is called weakly supervised. The objective of this pre-training step was to have the AES model learn a general language representation that can roughly distinguish essay quality.
2. *Cross-prompt supervised fine-tuning* If we have rated essays written for non-target prompts, the pre-trained model is fine-tuned using the data.
3. *Target-prompt supervised fine-tuning* The model obtained from the above steps is fine-tuned using rated essays written for the target prompt. The study by Song et al. (2020) reported that incorporating the above two-stage pre-training and fine-tuning improves the performance of the target-prompt scoring.

5.3 Model with self-supervised learning

Cao et al. (2020) proposed another cross-prompt holistic scoring model that was designed to solve the AES task with two prompt-independent self-supervised learning tasks jointly. The two self-supervised learning tasks, which are appended to efficiently extract prompt-independent common knowledge, are *a sentence reordering task* and *a noise identification task*, as explained bellow.

1. *Sentence reordering* In this task, each essay is divided into four parts and then shuffled according to a certain permutation order. The sentence reordering task predicts an appropriate permutation for each given essay.
2. *Noise identification* In this task, each essay is transformed into noisy data by performing random insertion, random swap, and random deletion operations on 10% of the words in the essay. The noise identification task predicts whether a given essay is noisy or not.

The above two self-supervised learning tasks are simultaneously trained with the AES task in a model. Figure 10 shows the model architecture. This model has a shared encoder that transforms an input word sequence into a fixed-length essay representation vector, and three task-specific output layers.

The shared encoder is formulated as a hierarchical representation DNN model such as that introduced in Sect. 3.3. In this model, a sequence of words corresponding to each sentence is transformed into a fixed-length sentence representation vector through a lookup table layer, a recurrent layer, a self-attention layer, a fusion gate, and a mean-over-time pooling layer. Here, the fusion gate is an operation that combines the input and output of the self-attention layer as follows.

$$\lambda_i = \sigma\left(\mathbf{W}_{g1} \cdot \boldsymbol{H}_{s_i} + \mathbf{W}_{g2} \cdot \tilde{\boldsymbol{H}}_{s_i}\right), \tag{14}$$

$$\hat{\boldsymbol{H}}_{s_i} = \lambda_i \boldsymbol{H}_{s_i} + (1 - \lambda_i)\tilde{\boldsymbol{H}}_{s_i}, \tag{15}$$

where $\boldsymbol{H}_{s_i}$ and $\tilde{\boldsymbol{H}}_{s_i}$ are the input and output vector sequences of the self-attention layer for i-th sentence, and $\hat{\boldsymbol{H}}_{s_i}$ is the fusion gate output. $\mathbf{W}_{g1}$ and $\mathbf{W}_{g2}$ are trainable parameters. The essay representation vector is calculated by averaging the obtained sentence vectors, and the vector is used for the AES and the two self-supervised learning tasks. This model is trained based on a weighted sum of the MSE loss function for the AES and error loss functions for the two self-supervised learning tasks.

Furthermore, Cao et al. (2020) proposed a technique to improve the adaptability of the model to a target prompt. Concretely, during the model training processes,

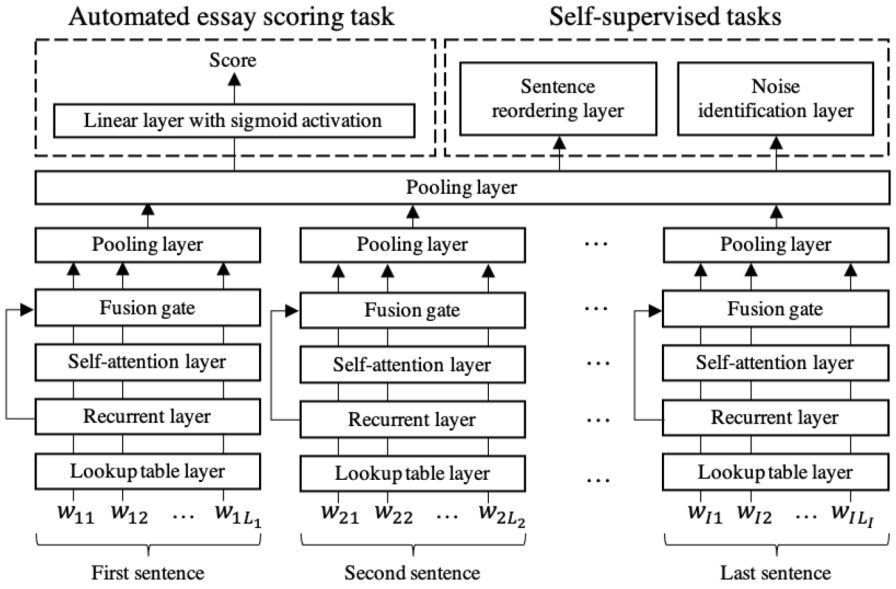

Fig. 10 Architecture of cross-prompt holistic scoring model with self-supervised learning

this technique calculates the averaged essay representation vector for each prompt and shifts the representation of each essay into the target prompt's averaged vector.

6 Cross-prompt trait scoring

This section introduces cross-prompt trait scoring models that predict multiple trait-specific scores for each essay in a cross-prompt setting.

6.1 Use of multiple trait-specific models with self-supervised learning

Mim et al. (2019) proposed a method to predict two trait scores, namely, *coherence* and *argument strength*, for each essay. They used a vast number of unrated essays for non-target prompts to pre-train a DNN model, and then the model was transferred to a target AES task. They used the RNN-based model (Taghipour and Ng 2016) introduced in Sect. 3.1 as the base model. The detailed processes are as follows.

1. *Pre-training based on self-supervised learning with non-target essays* In this step, the base model is trained using unrated essays written for non-target prompts based on a self-supervised learning task, which is a binary classification task that distinguishes artificially created incoherent essays. For the self-supervised learning task, incoherent essays are created by randomly shuffling sentences, discourse indicators, and paragraphs in the original essays. This pre-training is introduced to enable the base model to learn features for distinguishing logical text from illogical text.
2. *Pre-training based on self-supervised learning with target essays* The pre-trained model is retrained using essays written for the target prompt based on the same self-supervised task described above. This step is introduced to alleviate mismatch between essays written for non-target prompts and those written for the target prompt.
3. *Fine-tuning for AES* The pre-trained model is fine-tuned for the AES task using rated essays for the target prompt. Note that, for the AES task, the base model is extended by adding two RNN-based architectures that process a prompt text and a sequence of paragraph function labels (i.e., Introduction, Body, Rebuttal and Conclusion). The fine-tuning is conducted independently for two traits, namely, coherence and *argument strength*.

6.2 Model with multiple output modules

Ridley et al. (2021) proposed a model specialized in trait scoring that can predict multiple trait scores jointly. As shown in Fig. 11, the model is formulated as the following multi-output DNN model.

– *Shared layers* The model first processes an input through shared layers that is commonly used for predicting all trait scores. The shared layers consist of a

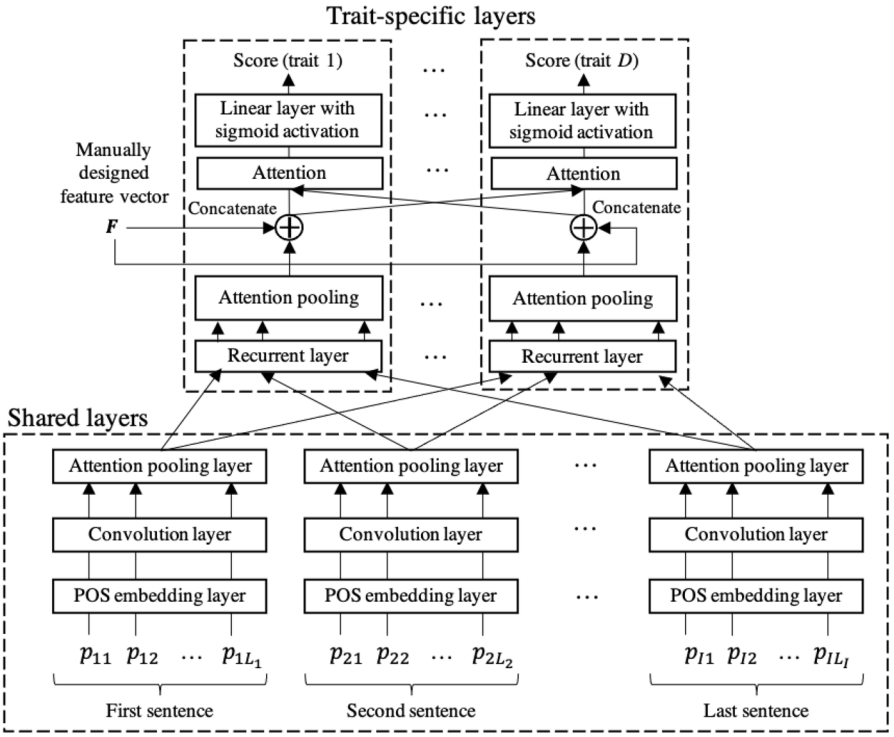

Fig. 11 Architecture of cross-prompt trait scoring model with multiple output layers

POS embedding layer, a convolutional layer, and an attention pooling layer, as explained below.

- *A POS embedding layer* takes a sequence of POS tags for words in a given essay and transforms it into embedding representations, using the same operations as in the lookup table layer. Note that this model uses a POS tag sequence as the input instead of a word sequence because word information depends strongly on a prompt, but POS information that represents syntactic information is more adaptable to different prompts.
- *A convolutional layer* extracts n-gram level features from a sequence of POS embeddings for each sentence in the same way as described in Sect. 3.1.
- *An attention pooling layer* applies an attention mechanism to produce a fixed-length vector representation for each sentence from the convolutional layer outputs.

– *Trait-specific layers* The sequence of sentence representations produced by the shared convolutional layer is input into trait-specific layers that are used for predicting each trait score through the following procedures.

1. The sentence representation sequence is transformed into a fixed-length vector corresponding to essay representation through a recurrent layer, and an attention pooling layer.
2. The essay representation vector is concatenated with prompt-independent manually designed features, similar to those used in the first stage of TDNN.
3. To obtain a final representation for each trait score, the model applies an attention mechanism so that each trait-specific layer can utilize the relevant information from the other trait-specific layers.
4. A linear layer with sigmoid activation maps the aggregated vector to a corresponding trait score.

The loss function for training this model is similar to Eq. (13). Note that, because different prompts are often designed to evaluate different trait scores, the model introduces a masking function. Concretely, letting $mask_{nd}$ be a variable that takes 1 if the prompt corresponding to n-th essay that has d-th trait score, and 0 otherwise, the loss function with the mask function is defined as follows.

$$\frac{1}{ND}\left(\sum_{n=1}^{N}\sum_{d=1}^{D} mask_{nd}(y_{nd}-\hat{y}_{nd})^2\right). \tag{16}$$

The mask function enables the loss values for the traits without the gold scores to be 0.

A special case of this model that has a single output layer for the holistic score has also been proposed as a cross-prompt holistic scoring model (Ridley et al. 2020).

7 Conclusions and remarks

This review has presented a comprehensive survey of DNN-AES models. Concretely, we classified the AES task into four types, namely, (1) prompt-specific holistic scoring, (2) prompt-specific trait scoring, (3) cross-prompt holistic scoring, and (4) cross-prompt trait scoring, and introduced the main ideas and the architectures of representative DNN-AES models for each task type.

As shown in our study, earlier DNN-AES models focus mainly on the prompt-specific holistic scoring task. The commonly used baseline model is the RNN-based model (Taghipour and Ng 2016), which has been extended by incorporating an efficient word embedding representation, a hierarchical structure of a text, a coherence model, and manually designed features. We also described transformer-based models such as BERT that have recently been applied to AES with their widespread use in various machine learning research studies.

These prompt-specific holistic scoring models have been extended for prompt-specific trait scoring, which predicts multiple trait scores for each essay. Trait scoring is practically important, especially when we need to provide detailed feedback to examinees for educational purposes, although the number of papers for this task is still limited.

Although prompt-specific scoring tasks assume that we can use a sufficient number of rated essays for a target prompt, this assumption is not often satisfied in practice because collecting rated essays is an expensive and time-consuming task. To overcome this limitation, cross-prompt scoring models have provided frameworks that use a large number of essays for non-target prompts. Although the number of cross-prompt scoring models is still limited, this task is important for increasing the feasibility of applying DNN-AES models to practical situations.

We can use several corpora to develop and evaluate AES models. ASAP corpus, which was released as part of a Kaggle competition, has been commonly used in holistic scoring models. For the trait scoring models, the International Corpus of Learner English (Ke 2019) and ASAP++ corpus (Mathias et al. 2018) are available. See (Ke and Ng 2019) for a more detailed summary of these corpora.

A future direction of AES studies is developing efficient and accurate trait scoring models and cross-prompt models. As described above, although the number of studies for those DNN-AES models is limited, such studies are essential to the use of AES technologies in various situations. It is also important to develop methodologies that reduce costs and noise when training data are being created. Approaches to reducing rating costs include recently examined active learning approaches (e.g., Hellman et al. 2019). To reduce scoring noise or biases, the integration of statistical models such as the IRT models described in Sect. 3.7 would be a possible approach.

Another future direction is to analyze the quality of each essay test and the characteristics of an applied AES model based on test theory. From the perspective of test theory, evaluating the reliability and validity of a test and its scoring processes is important for discussing the appropriateness of the test as a measurement tool. Although AES studies tend to ignore these points, several works have considered the relationship between DNN-based AES tasks and test theory (e.g., Uysal and Doğan 2021; Uto and Uchida 2020; Ha et al. 2020).

The application of AES methods to various related domains is also desired. For example, AES methods would be applicable to various operations such as writing support systems (e.g., Ito et al. 2020; Tsai et al. 2020) and peer grading processes (Han et al. 2020).

Acknowledgements This work was supported by JSPS KAKENHI Grant Numbers 19H05663 and 21H00898.

Declarations

Conflict of interest The authors have no conflicts of interest directly relevant to the content of this article.

References

Abosalem Y (2016) Beyond translation: adapting a performance-task-based assessment of critical thinking ability for use in Rwanda. Int J Secondary Educ 4(1):1–11

Alikaniotis D, Yannakoudakis H, Rei M (2016) Automatic text scoring using neural networks. In: Proceedings of the annual meeting of the association for computational linguistics (pp. 715–725)

Amidei J, Piwek P, Willis A (2020) Identifying annotator bias: a new irt-based method for bias identification. In: Proceedings of the international conference on computational linguistics (pp. 4787–4797)

Amorim E, Cançado M, Veloso A (2018) Automated essay scoring in the presence of biased ratings. In: Proceedings of the annual conference of the north American chapter of the association for computational linguistics (pp. 229–237)

Aomi I, Tsutsumi E, Uto M, Ueno M (2021) Integration of automated essay scoring models using item response theory. In: Proceedings of the international conference on artificial intelligence in education (pp. 54–59)

Attali Y, Burstein J (2006) Automated essay scoring with e-rater v.2. J Technol, Learn Assessment 4(3):1–31

Bahdanau D, Cho K, Bengio Y (2014) Neural machine translation by jointly learning to align and translate. arXiv

Beigman Klebanov B, Flor M, Gyawali B (2016) Topicality-based indices for essay scoring. In: Proceedings of the workshop on innovative use of NLP for building educational applications (pp. 63–72)

Bernardin HJ, Thomason S, Buckley MR, Kane JS (2016) Rater rating-level bias and accuracy in performance appraisals: the impact of rater personality, performance management competence, and rater accountability. Hum Resour Manage 55(2):321–340

Borade JG, Netak LD (2021) Automated grading of essays: a review. In: Intelligent human computer interaction (vol. 12615, pp. 238–249), Springer International Publishing

Cao Y, Jin H, Wan X, Yu Z (2020) Domain-adaptive neural automated essay scoring. In: Proceedings of the international ACM SIGIR conference on research and development in information retrieval (pp. 1011–1020), Association for Computing Machinery

Cao Z, Qin T, Liu TY, Tsai MF, Li H (2007) Learning to rank: From pairwise approach to listwise approach. In: Proceedings of the international conference on machine learning (pp. 129–136), Association for Computing Machinery

Collobert R, Weston J (2008) A unified architecture for natural language processing: Deep neural networks with multitask learning. In: Proceedings of the international conference on machine learning (pp. 160–167), Association for Computing Machinery

Cozma M, Butnaru A, Ionescu RT (2018) Automated essay scoring with string kernels and word embeddings. In: Proceedings of the annual meeting of the association for computational linguistics (pp. 503–509)

Dascalu M, Westera W, Ruseti S, Trausan-Matu S, Kurvers H (2017) Readerbench learns Dutch: building a comprehensive automated essay scoring system for Dutch language. In: Proceedings of the international conference on artificial intelligence in education (pp. 52–63)

Dasgupta T, Naskar A, Dey L, Saha R (2018) Augmenting textual qualitative features in deep convolution recurrent neural network for automatic essay scoring. In: Proceedings of the workshop on natural language processing techniques for educational applications, association for computational linguistics (pp. 93–102)

Devlin J, Chang MW, Lee K, Toutanova K (2019) BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the annual conference of the north American chapter of the association for computational linguistics: Human language technologies (pp. 4171–4186)

Dong F, Zhang Y (2016) Automatic features for essay scoring—an empirical study. In: Proceedings of the conference on empirical methods in natural language processing (pp. 1072–1077), Association for Computational Linguistics

Dong F, Zhang Y, Yang J (2017) Attention-based recurrent convolutional neural network for automatic essay scoring. In: Proceedings of the conference on computational natural language learning (pp. 153–162), Association for Computational Linguistics

Eckes T (2015) Introduction to many-facet Rasch measurement: analyzing and evaluating rater-mediated assessments, Peter Lang Pub. Inc

Farag Y, Yannakoudakis H, Briscoe T (2018) Neural automated essay scoring and coherence modeling for adversarially crafted input. In: Proceedings of the annual conference of the north American chapter of the association for computational linguistics (pp. 263–271)

Ha LA, Yaneva V, Harik P, Pandian R, Morales A, Clauser B (2020) Automated prediction of examinee proficiency from short-answer questions. In: Proceedings of the international conference on computational linguistics (pp. 893–903)

Han Y, Wu W, Yan Y, Zhang L (2020) Human-machine hybrid peer grading in SPOCs. IEEE Access 8:220922–220934

Hellman S, Rosenstein M, Gorman A, Murray W, Becker L, Baikadi A, Foltz PW (2019) Scaling up writing in the curriculum: Batch mode active learning for automated essay scoring. In: Proceedings of the ACM conference on learning (pp. 1—10), Association for Computing Machinery

Hua C, Wind SA (2019) Exploring the psychometric properties of the mind-map scoring rubric. Behaviormetrika 46(1):73–99

Huang J, Qu L, Jia R, Zhao B (2019) O2U-Net: a simple noisy label detection approach for deep neural networks. In: Proceedings of the IEEE international conference on computer vision (pp. 3326–3334)

Hussein MA, Hassan HA, Nassef M (2019) Automated language essay scoring systems: a literature review. Peer J Comput Sci 5:e208

Hussein MA, Hassan HA, Nassef M (2020) A trait-based deep learning automated essay scoring system with adaptive feedback. Int J Adv Comput Sci Appl 11(5):287–293

Ito T, Kuribayashi T, Hidaka M, Suzuki J, Inui K (2020) Langsmith: n interactive academic text revision system. In: Proceedings of conference on empirical methods in natural language processing (pp. 216–226), Association for Computational Linguistics

Jin C, He B, Hui K, Sun L (2018) TDNN: a two-stage deep neural network for prompt-independent automated essay scoring. In: Proceedings of the annual meeting of the association for computational linguistics (pp. 1088–1097)

Joachims T (2002) Optimizing search engines using clickthrough data. In: Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining (pp. 133–142), Association for Computing Machinery

Kassim NLA (2011) Judging behaviour and rater errors: an application of the many-facet Rasch model. GEMA Online J Lang Stud 11(3):179–197

Ke Z, Inamdar H, Lin H, Ng V (2019) Give me more feedback II: Annotating thesis strength and related attributes in student essays. In: Proceedings of the annual meeting of the association for computational linguistics (pp. 3994–4004)

Ke Z, Ng V (2019) Automated essay scoring: a survey of the state of the art. In: Proceedings of the international joint conference on artificial intelligence (pp. 6300–6308)

Li S, Ge S, Hua Y, Zhang C, Wen H, Liu T, Wang W (2020) Coupled-view deep classifier learning from multiple noisy annotators. In: Proceedings of the association for the advancement of artificial intelligence (pp. 4667–4674)

Li X, Chen M, Nie J, Liu Z, Feng Z, Cai Y (2018) Coherence-based automated essay scoring using self-attention. In: Chinese computational linguistics and natural language processing based on naturally annotated big data (pp. 386–397), Springer International Publishing

Li X, Chen M, Nie JY (2020) SEDNN: shared and enhanced deep neural network model for cross-prompt automated essay scoring. Knowl-Based Syst 210:106491

Liu OL, Frankel L, Roohr KC (2014) Assessing critical thinking in higher education: current state and directions for next-generation assessment. ETS Res Rep Series 1:1–23

Liu T, Ding W, Wang Z, Tang J, Huang GY, Liu Z (2019) Automatic short answer grading via multiway attention networks. In: Proceedings of the international conference on artificial intelligence in education (pp. 169–173)

Lun J, Zhu J, Tang Y, Yang M (2020) Multiple data augmentation strategies for improving performance on automatic short answer scoring. In: Proceedings of the association for the advancement of artificial intelligence (pp. 13389–13396)

Mark D, Shermis JCB (2016) Automated essay scoring: a cross-disciplinary perspective. Taylor & Francis

Mathias S, Bhattacharyya P (2018) ASAP++: enriching the ASAP automated essay grading dataset with essay attribute scores. In: Proceedings of the eleventh international conference on language resources and evaluation (pp. 1169–1173)

Mathias S, Bhattacharyya P (2020) Can neural networks automatically score essay traits? In: Proceedings of the workshop on innovative use of nlp for building educational applications (pp. 85–91), Association for Computational Linguistics

Mayfield E, Black AW (2020) Should you fine-tune BERT for automated essay scoring? In: Proceedings of the workshop on innovative use of nlp for building educational applications (pp. 151–162), Association for Computational Linguistics

Mesgar M, Strube M (2018) A neural local coherence model for text quality assessment. In: Proceedings of the conference on empirical methods in natural language processing (pp. 4328–4339)

Mim FS, Inoue N, Reisert P, Ouchi H, Inui K (2019) Unsupervised learning of discourse-aware text representation for essay scoring. In: Proceedings of the annual meeting of the association for computational linguistics: student research workshop (pp. 378–385)

Myford CM, Wolfe EW (2003) Detecting and measuring rater effects using many-facet Rasch measurement: part I. J Appl Meas 4:386–422

Nadeem F, Nguyen H, Liu Y, Ostendorf M (2019) Automated essay scoring with discourse-aware neural models. In: Proceedings of the workshop on innovative use of NLP for building educational applications, association for computational linguistics (pp. 484–493)

Nguyen HV, Litman DJ (2018) Argument mining for improving the automated scoring of persuasive essays. In: Proceedings of the association for the advancement of artificial intelligence (pp. 5892–5899)

Phandi P, Chai KMA, Ng HT (2015) Flexible domain adaptation for automated essay scoring using correlated linear regression. In: Proceedings of the conference on empirical methods in natural language processing (pp. 431–439)

Rahman AA, Ahmad J, Yasin RM, Hanafi NM (2017) Investigating central tendency in competency assessment of design electronic circuit: analysis using many facet Rasch measurement (MFRM). Int J Inf Educ Technol 7(7):525–528

Ridley R, He L, Dai X, Huang S, Chen J (2020) Prompt agnostic essay scorer: a domain generalization approach to cross-prompt automated essay scoring. arXiv

Ridley R, He L, yu Dai X, Huang S, Chen J (2021) Automated cross-prompt scoring of essay traits. In: Proceedings of the AAAI conference on artificial intelligence (vol 35, pp. 13745-13753)

Rodriguez PU, Jafari A, Ormerod CM (2019) Language models and automated essay scoring. arXiv

Rosen Y, Tager M (2014) Making student thinking visible through a concept map in computer-based assessment of critical thinking. J Educ Comput Res 50(2):249–270

Schendel R, Tolmie A (2017) Assessment techniques and students' higher-order thinking skills. Assess & Eval Higher Educ 42(5):673–689

Song W, Zhang K, Fu R, Liu L, Liu T, Cheng M (2020) Multi-stage pre-training for automated Chinese essay scoring. In: Proceedings of the conference on empirical methods in natural language processing (pp. 6723–6733), Association for Computational Linguistics

Sung C, Dhamecha TI, Mukhi N (2019) Improving short answer grading using transformer-based pre-training. In: Proceedings of the international conference on artificial intelligence in education (pp. 469–481)

Taghipour K, Ng HT (2016) A neural approach to automated essay scoring. In: Proceedings of the conference on empirical methods in natural language processing (pp. 1882–1891)

Tay Y, Phan MC, Tuan LA, Hui SC (2018) SKIPFLOW: Incorporating neural coherence features for end-to-end automatic text scoring. In: Proceedings of the AAAI conference on artificial intelligence (pp. 5948–5955)

Tsai CT, Chen JJ, Yang CY, Chang JS (2020) LinggleWrite: a coaching system for essay writing. In: Proceedings of annual meeting of the association for computational linguistics (pp. 127–133), Association for Computational Linguistics

Uto M (2019) Rater-effect IRT model integrating supervised LDA for accurate measurement of essay writing ability. In: Proceedings of the international conference on artificial intelligence in education (pp. 494–506)

Uto M, Okano M (2020) Robust neural automated essay scoring using item response theory. In: Proceedings of the international conference on artificial intelligence in education (pp. 549–561)

Uto M, Uchida Y (2020) Automated short-answer grading using deep neural networks and item response theory. In: Proceedings of the artificial intelligence in education (pp. 334–339)

Uto M, Ueno M (2016) Item response theory for peer assessment. IEEE Trans Learn Technol 9(2):157–170

Uto M, Ueno M (2018a) Empirical comparison of item response theory models with rater's parameters. Heliyon, Elsevier 4(5):1–32

Uto M, Ueno M (2018b) Item response theory without restriction of equal interval scale for rater's score. In: Proceedings of the international conference on artificial intelligence in education (pp. 363–368)

Uto M, Ueno M (2020) A generalized many-facet Rasch model and its Bayesian estimation using Hamiltonian Monte Carlo. Behaviormetrika, Springer 47(2):469–496

Uto M, Xie Y, Ueno M (2020) Neural automated essay scoring incorporating handcrafted features. In: Proceedings of the international conference on computational linguistics (pp. 6077–6088), International Committee on Computational Linguistics

Uysal İ, Doğan N (2021) Automated essay scoring effect on test equating errors in mixed-format test. Int J Assess Tools Educ 8:222–238

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Polosukhin I (2017) Attention is all you need. In: Proceedings of the international conference on advances in neural information processing systems (pp. 5998–6008)

Wang Y, Wei Z, Zhou Y, Huang X (2018) Automatic essay scoring incorporating rating schema via reinforcement learning. In: Proceedings of the conference on empirical methods in natural language processing (pp. 791–797)

Yang R, Cao J, Wen Z, Wu Y, He X (2020) Enhancing automated essay scoring performance via fine-tuning pre-trained language models with combination of regression and ranking. In: Findings of the association for computational linguistics: EMNLP 2020 (pp. 1560–1569), Association for Computational Linguistics

Yang Y, Zhong J (2021) Automated essay scoring via example-based learning. In: Brambilla M, Chbeir R, Frasincar F, Manolescu I (eds) Web engineering. Springer International Publishing, pp 201–208