Psychological Test and Assessment Modeling, Volume 64, 2022 (4), 495-526

Mapping Between Hidden States and Features to Validate Automated Essay Scoring Using DeBERTa Models

Christopher Michael Ormerod[1]

Abstract

We introduce a regression-based framework to explore the dependence that global features have on score predictions from pretrained transformer-based language models used for Automated Essay Scoring (AES). We demonstrate that neural networks use approximations of rubric-relevant global features to determine a score prediction. By considering linear models on the hidden states, we can approximate global features and measure their importance to score predictions. This study uses DeBERTa models trained on overall scores and trait-level scores to demonstrate this framework with a specific focus on convention errors, which are errors in the use of language, encompassing spelling, grammar, and punctuation errors. This introduces a new form of explainability and provides evidence of validity for Language Model based AES.

Keywords: Automated Essay Scoring, Transformer, DeBERTa Model, Language Models, Explainability, Overall Essay Scores, Trait-Level Essay Scores

[1] Cambium Assessment *Correspondence concerning this article should be addressed to:* Christopher Michael Ormerod, Cambium Assessment, Inc. 1000 Thomas Jefferson St., N.W. Washington, D.C. 20007, USA. christopher.ormerod@cambiumassessment.com.

Introduction

Automated Text Scoring (ATS) is an application of statistical models to approximate how a human rater might assess constructed text responses. ATS can be classified into two categories: Automated Essay Scoring (AES) and Automated Short-Answer Scoring (ASAS). AES has had a long-distinguished history dating back to 1968 with Project Essay Grade (PEG) by Ellis Page (Page, 2003). Most current AES engines in production use a mix of frequency-based methods and expert crafted features (Attali & Burstein, 2006). While AES allows us to provide instant and consistent feedback at substantially lower cost than human assigned scores under the condition that the models meet the technical standards and the scores are valid and reflect the rubrics. Work on the validity of AES helps address these concerns (Attali, 2013). There have been numerous studies on the performance using Artificial Intelligence (AI) based AES. This includes a range of neural network-based AES models such as traditional convolutional and recurrent networks (Dong et al., 2017; Taghipour & Ng, 2016) and more recent pretrained transformer-based language models (Ormerod et al., 2021; Rodriguez et al., 2019; Uto & Uchida, 2020; Yang et al., 2020). While traditional AES explicitly depend on hand-crafted features (Attali & Burstein, 2006), the paradigm for neural network-based AES is that the models learn a set of features implicitly that are conducive to a statistically high agreement in the training process.

To address the validity issues in AES, two major questions are raised related to what features have been learned in this training process and how these features are related to the original rubric. This study introduces a novel method of testing whether a given feature has been learned in the training process and provides a measure of the features importance to score predictions. We use this method to provide evidence of validity to neural network-based AES.

From a computer science standpoint, this study is relevant to explainable Artificial Intelligence (xAI; Vilone & Longo, 2020), which concerns methods designed to help human understand the predictions made by AI. Within xAI, we address explainability in terms of local features, which focuses on the importance of particular words/tokens to a score, and global explainability, which focuses on the features of an essay as a whole. For example, a local feature could be a key word in a student response that is relevant to the prompt, whereas an example of global feature is the number of adjectives used in a student response. Most work in xAI concerns local features rather than global features (Smilkov et al., 2017; Sundararajan et al., 2017) and is usually centered around numerical values of importance derived by removing or masking tokens and observing the change in the models' hidden states and outputs. Such an approach is not sufficient to address the validity of the scores from AES because the properties relevant to scoring go beyond single tokens. In this way, we believe our proposed approach is novel in that it is focused on global properties of a response. It is important to view explainability in terms both local and global features. In this way, saliency maps based on the approaches like integrated gradients (Sundararajan et al., 2017) provides a complement to our global approach. Saliency maps (Sundararajan et al.,

2017) isolate potential tokens that may be detrimental to a given score within a response and heat maps are a way of visualizing the output. The combination of the methods introduced in this paper and saliency maps provides evidence of validity at both the local and global levels.

From a psychometric standpoint, it is important to be able to test whether features have been used in the scoring process as validity evidence to support the explanation of the scores from AES. Currently, most models are defended purely from a statistical viewpoint in that they have high inter-rater agreement with human raters (Williamson et al., 2012). The agreement between computer and human scores does not indicate that the scores assigned are valid interpretations of the rubrics. It is important to know that neural networks are not just isolating spurious random correlations unrelated to the rubrics in order to produce high agreements. The validity evidence collected based on our proposed approach is whether the information used for score assignment is relevant to the rubrics. Little literature investigates how global attributes, such as length or total number of spelling or grammatical errors may affect score point determinations within a neural network-based AES engine.

Comprehensive discussions on the score validity of AES are too ambitious a goal to be covered in one paper. Our goal is to provide a simple proof of concept based around the dependence on the correct use of language in the rubrics. That is, we demonstrate that AES models use approximations of the number of convention errors per sentence an essay contains to determine scores. By convention errors, we mean spelling, grammar, and punctuation errors. Furthermore, the quality of these approximations provides a measure of the features importance.

Unlike many other traits, we have a well-defined relation between misspellings and grammatical errors to the rubrics concerning conventions. Thus, convention errors serve as our starting point for a discussion on score validity of AES. If our hypothesis is valid, rubrics that have a stronger dependence on conventions should yield models with better approximations for the number of convention errors. Another reason to start with conventions, as opposed to some other traits, is that it is notoriously difficult to write down features for other traits, such as organizational structure or argumentation quality (Wingate, 2012).

To illustrate the proposed approach, we first present how neural network-based AES engines embed information in score point determinations. We then present an analysis of the data used in this study followed by the proposed methods to approximate the number of convention errors in the responses. We describe the modelling details for the overall scores, trait-level scores, and feature approximations subsequently.

Class States and Implicit Features

Text classification engines are predominantly developed hierarchically; a text is tokenized, and the tokens are fed into a vector space model, which is an algebraic model that maps the document to a vector space (Salton et al., 1975). The vector space is defined as a set of hidden states that are used as input into a traditional classifier to determine scores. These hidden states are important because they encode all possible information used by the engine to assign scores. We reference these hidden states as the class states and the vector as the vector of the class states as input into the traditional classifier. In general, the class states are functions of the entire text and the interpretations of the class states are expressed in terms of global features.

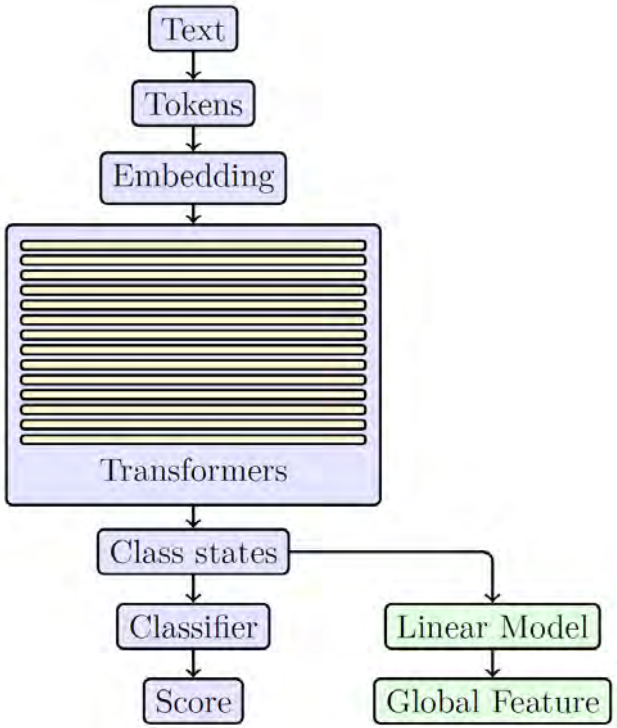
In AES, the class states encode elements pertaining to simple global features like length, but also high-level global features like organizational structure, argumentation quality, style, word choice, and voice. Interpreting the class states in terms of high-level characteristics such as argumentation quality, for example, would require an associated quantifiable features conducive to argumentation.

To illustrate class states, we present two examples. The first example is the traditional bag-of-words (BOW) model, where the class states are the union of frequency-based terms, typically elements of a term frequency-inverse document frequency (TF-IDF) vector, and a vector containing hand-crafted features. This vector is commonly used as input in a logistic regression or random forest classifier.

The second example is a transformer-based language model, such as the Bidirectional Encoding by Representations using Transformers (BERT; Devlin et al., 2019), fine-tuned for text-classification. This general structure for a transformer-based classifier is presented in Figure 1. The class states in this case are specified as the context-dependent representation of the first token that has been fine tuned to summarize the global properties of the text. This means the class states are the output of an embedding (word and positional) followed by multiple layers of attention-based transformer units (Wolf et al., 2020). Figure 1 presents the general structure of text classification models where an auxiliary linear model appended to determine the importance of a global feature.

Figure 1

Text Classifier Structure



Many studies in explainability consider tracking the changes in output through each of the attention-based transformer layers. Recall that attention is a model of relative importance between tokens (Devlin et al., 2019). The original BERT system of Devlin et al. (2019), and many other derivative architectures, are based on 12 to 24 layers of attention based transformer units, where attention is applied at each layer. When attention is applied successively, the relevant mechanism is called self-attention. Self-attention not only encodes the relationships between words, but also the importance of the relationships to other relationships between words. Although this provides the language model with a more complete context for each word, it is difficult to endow the final outputs with a concrete intuitive meaning with 24 layers of self-attention after fine-tuning is applied.

An analysis of the vector of class states considers this problem in the opposite direction. We take known properties of text and observe how the vector of class states vary while we change those properties. We know that, depending on the architecture, the

vector of class states is used as input into a final linear layer that is optimized with the other layers to produce the final score probabilities.

Let us demonstrate how the class states encode global information by taking length as an example. For the BOW models, length is typically used explicitly as a hand-crafted feature. This means length would be a class state and defines a single dimension in the vector of class states. Depending on the final simple classifier used, the importance of length to score determination can be inferred from the parameters that define the classifier. For neural networks, there are other ways length can appear in the class states. In contrast with the BOW models, the neural network uses length implicitly to determine scores if we are able to form a linear function of the class states that approximates length.

We can test whether the neural network is using a given feature implicitly by considering linear models on the vector of class states that approximate this feature. The way in which linear models approximate a feature is also shown in Figure 1. We determine how well the neural network has learned a feature by how well linear models on the vector of class states can approximate the feature. If our linear models are good approximations of this feature, the neural network uses it implicitly to determine scores. That is, a feature used in score determinations can be any numerical quantity approximated well as a linear combination of the class states. Our argument for validity is that neural networks determine score based on appropriate information if the features that are more relevant to the rubrics are modelled better than those that are not.

We can measure the quality of an approximation by using the Pearson correlation coefficient or the Spearman rank-order correlation coefficient. We use the Spearman rank-order correlation coefficient more frequently because the functional dependence between the feature and the score may be nonlinear. However, in determining the coefficients of the linear models approximating a given feature, we must optimize with respect to the Pearson correlation coefficient as a proxy for the Spearman rank-order correlation coefficient. This is because, if we maximize the correlation coefficient as a function of the coefficients of the linear model, we can only employ high-dimensional optimization techniques if the function is differentiable.

Given a feature and a linear model approximating this feature from the class states, if the Pearson correlation coefficient is high enough, augmenting the class states with this feature is algebraically equivalent to appending a row in a matrix that is a linear combination of the other rows. When we consider the features appended to the BERT models specified by Uto and Uchida (2020), appending features that are approximated well, such as the number of nouns or the number of syllables, only endows the neural network with redundant information.

We can also provide validity evidence through the use of other datasets and models built on those datasets when the dataset has a relevant property. Many higher-level features are not directly defined by rules, but by data. We use Corpus of Linguistic Acceptability (CoLA) as a dataset that is the most relevant to linguistic conventions (Warstadt et al., 2019). Our rationale for using the CoLA and models trained on the

CoLA is that the output probabilities from the model should define a measure of linguistic acceptability for language more generally, whereas models trained on conventions are typically only valid on student responses to a particular prompt. By applying a model trained on the CoLA to student responses, the outputs should be similar to a regression-based convention score. If the models trained have learned linguistic acceptability as a feature, then the outputs of a model trained on the CoLA datasets should be better approximated by linear models on class states when the rubrics depend on conventions.

Methods

Automated Essay Scoring Data

To illustrate our approach for addressing the score validity from AES, this study used the original essays in the Kaggle ASAP dataset (Shermis, 2014), which contains eight essay prompts. These essay prompts are classified into four types; persuasive, expository, descriptive, and narrative. The expository and descriptive essays in the Kaggle ASAP dataset are classified as "source dependent". The original data contains the rubrics which specify criteria that an essay needs to satisfy for each score point.

To understand the nature of the text responses, we need to know what the essays measure (Cohen et al., 1996). In a well-designed writing assessment, the rubrics for an essay prompt align with the standards. The scores assigned are expected to adhere to those standards. The writing standards in general specify traits associated with writing at different proficiency levels. The quality of the essay can be evaluated in an overall score using holistic scoring rubrics or trait-level scores using analytic scoring rubrics specific to each facet in writing.

Only two of the eight essay prompts, numbers 7 and 8, came with a complete set of trait level scores (Shermis, 2014), while one essay prompt, number 2, came with just scores for two traits. After the initial release of the data, an independent effort was made to provide trait-level scores for all prompts (Mathias & Bhattacharyya, 2018). This study used both overall scores and trait scores for each prompt. The rubrics for each trait are provided in both sources (Mathias & Bhattacharyya, 2018; Shermis, 2014). A summary of this data is presented in Table 1.

Table 1

A summary of the Essay Prompts

Es-say	Grade	Type	Training Samples	Average Length	Overall Range	Number of Traits	Trait Range
1	8	P	1783	350	2-12	5	1-6
2	10	P	1800	350	1-6	5+1	1-6
3	10	E	1726	150	0-3	4	0-3
4	10	E	1772	150	0-3	4	0-3
5	8	D	1805	150	0-4	4	0-4
6	10	D	1800	150	0-4	4	0-4
7	7	N	1569	250	2-24	4	0-3
8	10	N	723	650	10-60	6	1-6

Note*:* P for Persuasive, E for Expository, D for Descriptive, and N for Narrative.

For the trait-level scores, each type of essay assesses a different set of skills. In particular, the Common Core standards for writing associated with persuasive and narrative essays assess critical thinking. These tasks often place much emphasis on how the essay is organized, how well the arguments are supported, and the correct use of language. Expository and descriptive essays usually assess comprehension with more emphasis on style and how well the student relates their argument to a prompt text. The traits assessed by each prompt are listed in Table 2. It is noted that while language is important to all essays, persuasive and narrative prompts rely on the adherence to conventional rules of language more than expository and descriptive prompts.

Table 2

The Trait-Level Scores Assigned for Each Essay Prompt

Essay	Traits Assessed by Each Prompt
1	Ideas & Content, Organization, Word Choice, Sentence Fluency, Conventions.
2	Ideas & Content, Organization, Word Choice, Sentence Fluency, Conventions, Language Conventions
3	Content, Prompt Adherence, Language, Narrativity.
4	Content, Prompt Adherence, Language, Narrativity.
5	Content, Prompt Adherence, Language, Narrativity.
6	Content, Prompt Adherence, Language, Narrativity.
7	Ideas, Organization, Style, Conventions.
8	Ideas & Content, Organization, Voice, Word Choice, Sentence Fluency, Conventions.

When trait-level scores are available, the overall score can be derived from the addition of the scores for each trait (Shermis, 2014). This is the case for essay prompts 7 and 8. For the remaining prompts, the trait-level scores were provided after the overall scores were assigned by a different set of raters (Mathias & Bhattacharyya, 2018). One of the difficulties in this study is that the trait-level scores for the ASAP++ dataset are much more highly correlated than usual. Although usually positive correlations are observed between independent traits, it is very unusual for independent trait scores to be as highly correlated as they are in the ASAP++ data. This means it is more difficult to ascertain what properties are being measured that are unique to each trait. We list the inter-trait Spearman rank-order correlation coefficients in Table 3. The average (Avg.) for each trait is calculated over all essay prompts for that trait.

Table 3

Trait-Level Correlations for the ASAP and ASAP++ Data

1	Cont	Org	WC	SF
Org.	0.810	-	-	-
WC	0.806	0.761	-	-
SF	0.796	0.779	0.856	-
Conv.	0.788	0.782	0.848	0.950

2	Cont	Org	WC	SF	Conv.
Org.	0.883	-	-	-	-
WC	0.851	0.836	-	-	-
SF	0825	0.817	0.872	-	-
Conv.	0.778	0.774	0.842	0.836	-
LC	0.592	0.608	0.629	0.606	0.644

3	Cont	PA	Lang
PA	0.900	-	-
Lang	0.790	0.808	-
Nar	0.838	0.859	0.895

4	Cont	PA	Lang
PA	0.954	-	-
Lang	0.791	0.792	-
Nar	0.899	0.904	0.861

5	Cont	PA	Lang
PA	0.852	-	-
Lang	0.712	0.711	-
Nar	0.779	0.780	0.784

6	Cont	PA	Lang
PA	0.917	-	-
Lang	0.690	0.721	-
Nar	0.750	0.773	0.839

7	Cont	Org	Style
Org.	0.797	-	-
Style	0.610	0.635	-
Conv.	0.494	0.567	0.626

8	Cont	Org	Voice	WC	SF
Org.	0.801	-	-	-	-
Voice	0.755	0.705	-	-	-
WC	0.727	0.707	0.762	-	-
SF	0.687	0.719	0.660	0.752	-
Conv	0.643	0.665	0.583	0.627	0.739

Note: The traits are abbreviated as Content (Cont.), Organization (Org.), Word Choice (WC), Sentence Fluency (SF), Conventions (Conv.), Language Conventions (LC), Prompt Adherence (PA), Language (Lang.), Narrativity (Narr.), Style and Voice.

One of the important considerations is the quality of the data. Unlike many other areas of machine learning where classifications can be clear-cut, overall scores and trait-level scores have a level of subjectivity. There are a few standard practices in compiling datasets used for training AES engines for production purposes. These standards require that each response is scored by two independent raters (Williamson et al., 2012). These standards were not adhered to in the creation of the ASAP++ dataset (Mathias & Bhattacharyya, 2018).

The advantage of using two raters is that the agreement between the two raters becomes a benchmark for evaluating scoring engines trained on that data. The most commonly used benchmark for agreement is the quadratic weighted kappa (QWK) score given by

$$\kappa = 1 - \frac{\sum\sum w_{ij}\, x_{ij}}{\sum\sum w_{ij}\, m_{ij}}, \tag{1}$$

where x_{ij} is the observed probability of a rater first assigning a score of i and the rater assigning a score of j, m_{ij} is the expected agreements between raters, given by

$$m_{ij} = \left(\sum_k x_{ik}\right)\left(\sum_k x_{kj}\right), \tag{2}$$

where w_{ij} is the quadratic weight given by

$$w_{ij} = \frac{(i-j)^2}{(k-1)^2}, \tag{3}$$

where k is the number of classes (Cohen, 1960). This was the benchmark used in the competition and in evaluating the performance of AES more generally (Williamson et al., 2012). Different approaches could improve QWK including a more clearly defined scoring rubrics to distinguish between what constitutes each score point, the construction of good rater training material, and well-implemented backreads for quality assurance. In addition, two other metrics are often used including scoring accuracy, which is the probability of exact agreement between two raters, and standardized mean difference (SMD) given by

$$SMD = \frac{|\mu(y_1) - \mu(y_2)|}{\sqrt{\frac{\sigma(y_1)^2}{2} + \frac{\sigma(y_2)^2}{2}}}, \tag{4}$$

where y_1 and y_2 denote rater 1 and 2's scores and μ and σ denote the mean and standard deviation of their respective assigned scores.

Table 4 lists the QWK for all the traits with human double-scores computed from the ASAP dataset (Shermis, 2014). Note that the conventions score for prompt 2 refers to the second domain score presented in Shermis (2014) and not the trait-level scores presented in Mathias and Bhattacharyya (2018). Each response was assigned a score by a single rater for trait-level scores for prompts 1 to 6 (Mathias & Bhattacharyya,

2018), without information on QWK. Consequently, we do not have any benchmark for evaluating the quality of the trait scores. Further, the quality of the scores predicted by the automated scoring system cannot be assessed either. In general, the inter-rater agreements for trait-level scores presented in Table 4 are low by production standards (Williamson et al., 2012). For double-scored data for AES, a QWK of around 0.7 is often considered as a lower-bound. By this standard, the inter-rater agreements of the trait-level scores are lower than optimal.

Table 4

QWK for Trait-Level Scores

	Overall	Cont	Org	Style	Conv.	Voice	WC	SF
1	0.721	-	-	-	-	-	-	-
2	0.814	-	-	-	0.801	-	-	-
3	0.769	-	-	-	-	-	-	-
4	0.850	-	-	-	-	-	-	-
5	0.752	-	-	-	-	-	-	-
6	0.776	-	-	-	-	-	-	-
7	0.721	0.695	0.576	0.544	0.567			
8	0.624	0.531	0.542	-	0.546	0.467	0.481	0.507

Note: The traits are abbreviated as Content (Cont.), Organization (Org.), Word Choice (WC), Sentence Fluency (SF), Conventions (Conv.), Prompt Adherence (PA), Language (Lang.), Narrativity (Narr.), Style and Voice.

The original five-fold cross validation splits of Taghipour and Ng (2016) are used in this study. Each of the five folds is a non-overlapping set of the same size for each prompt where three sets are used for training, one set is used as a development set, and the remaining set is used as a test set. Each essay is in the test set for one fold and the performance of an AES engine is averaged over each test set. Regarding nomenclature, in some literature the training, validation, and test sets are used to label the splits. However, it is common in machine learning literature to use the term development set for the set used to optimize parameter and architecture choices and a test set to report results. Development and test sets were the terms used in Taghipour and Ng

(2016) which defined the splits for almost all subsequent studies of the ASAP dataset, and these are the naming conventions we follow in this study.

Counting the Number of Convention Errors

The second part of our data collection or processing is to calculate the number of convention errors. Since the information on the convention errors annotated by human raters was not available, we employ methods to approximate the counts. Grammatical Error Correction (GEC) is the process of transforming grammatically incorrect sentences to grammatically correct ones. The neural network architectures used for this task are the same architectures used to perform neural machine translation (Sutskever et al., 2014). These architectures typically contain an encoder, which transforms a grammatically incorrect sentence into a fixed dimensional vector, then a decoder interprets the vector to produce a grammatically correct variation of the incorrect sentence. Essentially, the most frequently used natural architecture is a sequence-to-sequence model of Vaswani et al. (2017). The model used in this study to perform GEC is the openly available pretrained model of Rothe et al. (2021) which is based on the T5 architecture (Raffel et al., 2020). This architecture uses the encoder and decoder structure of Vaswani et al. (2017) with 12 layers of transformers in both the encoder and decoder. The model leverages pretraining on a large corpus of synthetic data (Xue et al., 2021) and is then fine-tuned on high-quality human-annotated data (Rothe et al., 2021).

Due to the inherent length limitations in transformer-based architectures, the essays were split into sentences using the spaCy library[2] (Srinivasa-Desikan, 2018) and the GEC model of Rothe et al. (2021) was applied to each sentence. This provides us with a source sentence and a grammatically correct version of the sentence. We then apply the grammatical ERRor ANnotation Toolkit (ERRANT), which compares the original sentence to the corrected sentence, to classify the types of errors as being either spelling or grammatical (Bryant et al., 2017; Felice et al., 2016). We used LangTool[3] to calculate punctuation errors because there are no inherent length limitations. The punctuation errors can be more accurately identified using the rule-based methods. Using these two approaches, the total number of convention errors was calculated. Because the total number of errors is closely related to sentence length, we used the average number of errors per sentence to develop a quantity that is independent of length. The average number of errors per sentence in each of the essays for each prompt are presented in Table 5. The average number of errors per sentence in each of the essay prompts was approximated by combining the outputs from the GEC model used, LangTool, and the ERRANT system. Prompt 7 contains the most total errors per sentence while Prompt 6 contains the least total errors per sentence. This

[2] See https://spacy.io/ for more information.

[3] See https://languagetool.org/ for more information.

table also lists the average number of sentences ranging from 4.73 in Prompt 4 to 35.6 in Prompt 8. An evaluation of the quality of these approximations using GEC benchmarks is presented in Appendix A.

Table 5

Spelling, Grammatical, Punctuation, and Total Errors Per Sentence

Prompt	Spelling Errors Per Sentence	Grammatical Errors Per Sentence	Punctuation Errors Per Sentence	Total Errors Per Sentence	Average Number of Sentences
1	0.280	1.697	0.003	1.980	23.00
2	0.305	1.408	0.003	1.716	20.80
3	0.151	1.603	0.004	1.759	6.41
4	0.176	1.780	0.003	1.958	4.73
5	0.231	1.538	0.006	1.775	6.84
6	0.113	1.274	0.002	1.389	8.02
7	0.205	2.194	0.015	2.414	12.40
8	0.097	1.703	0.008	1.808	35.60

Modelling

This study explored the modeling of both overall and trait-level scores as well as the linear modeling of features using the vector of class states. There are a plethora of papers dedicated to modeling the overall scores in the Kaggle ASAP dataset (Dong et al., 2017; Ormerod et al., 2021; Rodriguez et al., 2019; Taghipour & Ng, 2016; Uto & Uchida, 2020). However, only a few papers addressed the modeling of trait-level scores (Mathias & Bhattacharyya, 2018, 2020; Ridley et al., 2021).

BERT has made a profound impact on NLP (Devlin et al., 2019). Researchers have sought to improve both the architectural aspects of BERT and the training procedures used to train BERT. Some have sought to simply scale up the parameters to obtain larger variants of the BERT architecture while others have created different language models by varying the corpus on which the models are trained on or modified the underlying structure of the model. Our model choice was guided by both the

methodological advance in BERT and the recent work on ASAS model performance on student data (Ormerod, 2022).

In this study, we used a variant of the DeBERTa model, which is a pretrained transformer-based language model utilizing disentangled attention and an adversarial training mechanism (He et al., 2021). DeBERTa differs from BERT in multiple aspects. First, a key difference in the architecture of DeBERTa is disentangled attention. BERT uses a positional embedding and a word embedding and adds the two vectors as members of the same vector space. On the other hand, DeBERTa represents the two embeddings as two different spaces, essentially operating on the disjoint union of vector spaces (He et al., 2021). Second, DeBERTa uses an adversarial training mechanism. Adversarial trained language models use a reinforcement learning training mechanism in which a generator produces outputs and a discriminator attempts to distinguish between the generated outputs and some ground truth (Clark et al., 2020; He et al., 2021). Adversarial trained models, such as DeBERTa and Electra (Clark et al., 2020), performed better than those trained simply as masked language models (Ormerod, 2022). These models are freely available in standard libraries[4] (Wolf et al., 2020). The DeBERTa models possess smaller variants for quick prototyping and possible use in production workflows.

The default language model-based classifier is one in which the vector of class states is used as input to a linear layer whose image is a vector of the same dimension as the number of score points (Wolf et al., 2020). This output is usually interpreted as log-probabilities, which is compared with the targets using the Cross-entropy loss function. This study does not adopt this approach for two reasons. First, cross-entropy applied to the log-probabilities treats each class equally regardless of all ordinal information. Second, the ASAP dataset contains prompts with large score point ranges with very few training samples for each score point. It is better to train a classifier using a regression-based approach. In a regression-based approach, the image of the classifier is a single floating-point number. By limiting the range of the linear layer using the sigmoid activation function, the image of the neural network is between 0 and 1.

We start by subdividing the interval [0,1] equally so that there is one sub-interval for each score point. Assume the maximum score is p and the minimum score is q, there are p-q+1 such intervals. We have one map in which each score is mapped to the midpoint of these intervals. Suppose we let

$$\boldsymbol{\delta = \frac{1}{p - q + 1}}, \tag{5}$$

then the mapping

[4] See https://huggingface.co/ for more information and implementation.

$$\boldsymbol{\mu(x)} \;=\; \boldsymbol{\delta(x-q)} \;+\; \frac{\boldsymbol{\delta}}{\boldsymbol{2}}, \tag{6}$$

satisfies this requirement. The inverse mapping is one in which we map each interval to the appropriate label. In a similar way, the inverse mapping is defined as

$$\boldsymbol{\mu^{-1}(x)} = \mathbf{round}\left(\mathbf{q} \;+\; \frac{\mathbf{x}}{\boldsymbol{\delta}} - \frac{\mathbf{1}}{\mathbf{2}}\right). \tag{7}$$

We can train the language model and the linear layer using regression by using the mean squared error (MSE) between the training targets and the output of the linear layers as a loss function. Using $\mu^{-1}(x)$ we can maximize QWK as an early stopping mechanism on the development set over 20 epochs. MSE is used as a proxy for the optimization of QWK.

We use the optimizer known as Adam with weight decay (Loshchilov & Hutter, 2019) with a learning rate of 2.5×10^{-5}. This method is a variation of the stochastic gradient descent (SGD) method with an adaptive step size. This method is an improvement upon the standard Adam optimizer by varying the weight decay mechanism (Loshchilov & Hutter, 2019). The chosen learning rate is a slightly lower than typically used because larger models train more robustly with slightly lower learning rates. This is coupled with a linear learning rate scheduler that tends to zero. In some cases, where DeBERTa failed to give expected performance (e.g., zero, or a significant drop compared with other folds), we ran the optimization twice and only chose the best performing model on the development set. The models failed rarely and their expected performance is known from other splits.

Once all the models had been trained, before the application to the test set, there was one additional optimization that was performed to obtain the final performance. One alternative interpretation of μ^{-1} is that μ^{-1} is defined by a series of cutoff points for the various scores. Thus, a sequence, c_k, is obtained with

$$\mathbf{0} \;<\; \mathbf{c_q} \;<\; \mathbf{c_{q+1}} \;<\ldots<\; \mathbf{c_{p-1}} \;<\; \mathbf{c_p} \;<\; \mathbf{1}, \tag{8}$$

given by $c_k = (q-k)\delta$ so that an alternative formulation of $\boldsymbol{\mu^{-1}}$ is given by

$$\boldsymbol{\mu^{-1}(x)} = \mathbf{q} \;+\; |\{\mathbf{x} \;>\; \mathbf{c_k} : \mathbf{q} \;<\; \mathbf{k} \;<\; \mathbf{p}\}|, \tag{9}$$

where |S| denotes the cardinality of a set. The goal of the additional optimization is to find a sequence, (c_q, …, c_p), satisfying the above constraint, that optimizes QWK on the development set. This is a constrained optimization problem with a non-differentiable target. We treat this as an unconstrained problem by taking a unconstrained vector and applying the composition of the softmax function, given by

$$\boldsymbol{\sigma(z_1,\cdots\ ,z_n)_i} = \frac{\mathbf{e^{z_i}}}{\sum_k \mathbf{e^{z_k}}} \tag{10}$$

and the cumulative sum allowing us to use a standard maximization algorithm with no constraints. The QWK as a function of the resulting scores is not a differentiable function, hence, many standard multidimensional optimization algorithms fail. Of the algorithms that do not require the underlying function to be differentiable, we chose Powell's method to optimize QWK due to its robustness (Powell, 1964).

The next part of modelling regards the approximations of features as linear functions of the class states. The first set of features explored is the total number of spelling errors, grammatical errors, and punctuation errors per sentence. The number of class states defined by the DeBERTa architecture is 1024, so for each split we have three matrices associated with the training, development, and test sets. Each of these has 1024 columns and one row for each element of the set.

Due to the possible non-linear nature of the relationship between score and the number of convention errors per sentence, our goal is to find the maximum Spearman rank-order correlation coefficient between the output of the linear models and our feature. However, the Spearman rank-order correlation coefficient is not differentiable. Given the size of the matrices, we need to employ methods that require differentiability. The approach we adopted was to optimize the Pearson correlation coefficients of the linear models on the training set using the Spearman correlation coefficient as an early stopping mechanism on the development set. The number of class states defined by the DeBERTa architecture is comparable to the number of training examples. To compensate, we used a linear layer with a dropout at the input level to regularize this network (Srivastava et al., 2014). This is also necessary because we typically expect many correlated class states. The result of this process is a linear approximation of the number of convention errors per sentence. This same process was applied to each feature used in the work of Uto and Uchida (2020).

In addition, this study considers the Corpus of Linguistic acceptability (CoLA). The CoLA was expertly annotated for general linguistic acceptability (Warstadt et al., 2019). This study uses the probability outputs of a pretrained distilled-BERT model fine-tuned on the CoLA (Sanh et al., 2020). Given the size and breadth of the CoLA, the output probabilities of this model, which are between 0 and 1, can be interpreted as an approximation of linguistic acceptability applied to language generally. If the model produces a value close to 0 for any given text, this should indicate the text is not linguistically acceptable and contains many linguistic errors, while a value close to 1 indicates the text contains relatively few linguistic errors. Given the model should apply to language more generally, it can also apply to the student responses. Applying the pretrained distilled-BERT model to each student response defines a new feature we call the linguistic acceptability of a response, which takes values between 0 and 1. The same process of finding linear approximations for the other features in terms of the class states was applied to linguistic acceptability, hence, we determine whether a form of linguistic acceptability is implicitly being used by the DeBERTa models trained for AES.

Saliency Methods and Local Explainability

The most common form of xAI is conducted at an input level (Linardatos et al., 2021). Common approaches include LIME (Ribeiro et al., 2016), SHAP (Lundberg & Lee, 2017), and integrated gradients (Sundararajan et al., 2017). The advantage of these approaches is that the output is easy to visualize which provides a straight-forward way of interpreting the results. Most applications of xAI for essays cause more concerns compared with short texts. These methods fail to give an adequate explanation of global properties, for example, how well an essay is organized. Thus, this study focuses on conventions first as we can consider convention errors at a global level, in terms of how well an essay generally adheres to the conventions of language, and a local level, in terms of specific errors like misspellings and grammatical errors.

The AllenNLP toolkit[5] offers an implementation of the Integrated Gradients method (Sundararajan et al., 2017) in the form of saliency maps. Saliency maps are typically applied to models in which the outputs are log-probabilities (Wallace et al., 2019), hence, we modified this code in order to apply the toolkit to classifications that use regression. For every token, the saliency map gives an attribution value, which is positive if inclusion of the token improves the score, and negative otherwise. If an engine is trained on good convention data, saliency maps applied to these models should give negative attributions when convention errors occur. Heat maps are a way to visualize the attributions by color-coding positive and negative attributions from a saliency map.

Results

We first present the results of modelling the scores in the ASAP and ASAP++ datasets. This includes both the results of the overall and trait-level scores. Secondly, we present the results of modelling global features as linear approximations of the class states. This provides us with a measure of the importance of these global features. Lastly, we present the results of using saliency, which complements the global features with local features that are important to score point determinations.

Modelling Overall and Trait-Level Scores

The values of QWK of modelling the overall scores from 8 models are presented in Table 6. The models featured in Table 6 include our proposed model and 7 other models analyzing the ASAP dataset. The methods of the compared papers were sufficiently distinct from each other and reflect the range of available methods used in the

[5] See https://github.com/allenai/allennlp.

literature. For the traditional BOW based approaches, the results of the EASE engine from Dong et al. (2017) are included. Further, models use the traditional recurrent and convolutional approaches with an attention mechanism was included as LSTM+CNN+Att. The QWK values from the first application of a single BERT model in Rodriguez et al. (2019) are listed in BERT (base), while the results of a regressive version of BERT from Yang et al. (2020) has been listed as R^2 BERT. The current state-of-the-art was achieved by appending hand-crafted features to BERT in Uto and Uchida (2020), and listed as BERT+Features. A computationally efficient ensemble from Ormerod et al. (2021), was listed as Efficient Ensemble in the table. The results of the modelling the overall scores from this study are presented as DeBERTa Large. The DeBERTa Large model performed similarly to the best performer for each prompt; the average QWK value is very close to the current state-of-the art. In general, the BERT+Features model (Uto & Uchida, 2020) performed the best on 5 of 8 prompts and the best on average. The human QWK reported in Table 6 is average inter-rater QWK over the five test sets. This is not to be confused by the QWK measured in Table 4, which is the QWK for the entire dataset. The averaged QWK over each fold and QWK for the entire dataset are not equal.

Table 6

QWK for the Overall Score Models on Each Prompt

QWK	1	2	3	4	5	6	7	8	Avg
Human	0.721	*0.812*	0.769	0.850	0.753	0.775	0.720	0.620	0.752
Ease	0.781	0.621	0.630	0.749	0.782	0.771	0.727	0.534	0.699
LSTM +CNN+Att	0.822	0.682	0.672	0.814	0.803	0.811	0.801	0.705	0.764
BERT (base)	0.792	0.680	0.715	0.801	0.806	0.805	0.785	0.596	0.758
R^2 BERT	0.817	0.719	0.698	0.845	0.841	*0.847*	0.839	0.744	0.794
BERT + Features	*0.852*	0.651	*0.804*	*0.888*	*0.885*	0.817	*0.864*	0.645	*0.801*
Efficient Ensemble	0.831	0.679	0.690	0.825	0.817	0.822	0.841	0.748	0.782
DeBERTa Large	0.832	0.713	0.699	0.835	0.826	0.834	0.851	*0.783*	0.797

The same procedure was applied to develop models for the trait-level scores. The results for the trait-score DeBERTa are presented in Table 7. A competing model of DeBERTa, labelled as CoNLL, uses the architecture specified by Dong et al. (2017) and is modeled in Mathias and Bhattacharyya (2020). The best QWK values for CoNLL from Mathias and Bhattacharyya (2020) are included for comparison of the DeBERTa model performance.

Table 7

QWK for the Trait-Level Score Models on Each Prompt

Prompt	Model	Cont	Org	WC	SF	Conv	PA	Lang	Narr	Style	Voice
1	CoNLL	0.703	0.664	0.675	0.648	0.638	-	-	-	-	-
	DeBERTa	0.739	0.712	0.717	0.742	0.725	-	-	-	-	-
2	CoNLL	0.617	0.623	0.630	0.603	0.601	-	-	-	-	-
	DeBERTa	0.720	0.712	0.756	0.743	0.749	-	-	-	-	-
3	CoNLL	0.673	-	-	-	-	0.683	0.612	0.684	-	-
	DeBERTa	0.739	-	-	-	-	0.751	0.714	0.749	-	-
4	CoNLL	0.751	-	-	-	-	0.738	0.645	0.722	-	-
	DeBERTa	0.807	-	-	-	-	0.806	0.749	0.796	-	-
5	CoNLL	0.738	-	-	-	-	0.719	0.638	0.700	-	-
	DeBERTa	0.733	-	-	-	-	0.725	0.688	0.689	-	-
6	CoNLL	0.820	-	-	-	-	0.783	0.664	0.690	-	-
	DeBERTa	0.849	-	-	-	-	0.802	0.720	0.723	-	-
7	CoNLL	0.771	0.676	-	-	0.621	-	-	-	0.659	-
	DeBERTa	0.722	0.588	-	-	0.585	-	-	-	0.580	-
8	CoNLL	0.586	0.632	0.559	0.586	0.558	-	-	-	-	0.544
	DeBERTa	0.568	0.592	0.561	0.588	0.599	-	-	-	-	0.551
Avg	CoNLL	0.707	0.649	0.621	0.612	0.605	0.731	0.640	0.699	0.659	0.544
	DeBERTa	0.735	0.651	0.675	0.682	0.661	0.771	0.718	0.739	0.580	0.551

Note: The traits are abbreviated as Content (Cont.), Organization (Org.), Word Choice (WC), Sentence Fluency (SF), Conventions (Conv.), Prompt Adherence (PA), Language (Lang.), Narrativity (Narr.), Style and Voice. In general, DeBERTa for each trait scores on each prompt performed better than CoNLL with higher QWK values than those from

CoNLL. The average QWK across all traits and prompts for the CoNLL model was 0.663, while for the DeBERTa models, the average QWK was 0.703, which is an improvement by 0.040 over the current best performing models known in accessible literature.

Modelling Features

Table 8 reports the average Spearman correlation coefficients over five test sets between the linear approximations of the errors per sentence and the number of errors identified. We could not produce a one-to-one correspondence between the 1024 class states and features given the information in the vector of class states.

Table 8

The Spearman's Rank-Order Correlation between the Linear Approximation and the Number of Convention Errors Per Sentence.

		Over.	Cont	Org	WC	SF	Conv	PA	Lang	Narr	Style/Voice
P	1	0.751	0.777	0.760	0.748	0.761	0.735	-	-	-	-
P	2	0.801	-	-	-	-	0.833	-	-	-	-
P	+	-	0.806	0.817	0.817	0.805	0.810	-	-	-	-
E	3	0.557	0.583	-	-	-	-	0.573	0.636	0.625	-
E	4	0.601	0.647	-	-	-	-	0.678	0.681	0.693	-
D	5	0.637	0.647	-	-	-	-	0.633	0.667	0.639	-
D	6	0.620	0.649	-	-	-	-	0.643	0.643	0.657	-
N	7	0.701	0.628	0.666	-	-	0.723	-	-	-	0.640
N	8	0.726	0.691	0.756	0.740	0.707	0.739	-	-	-	0.695

Note: The traits are abbreviated as Content (Cont.), Organization (Org.), Word Choice (WC), Sentence Fluency (SF), Conventions (Conv.), Prompt Adherence (PA), Language (Lang.), Narrativity (Narr.), Style and Voice. The row with a + refers to the ASAP++ data from (Mathias & Bhattacharyya, 2018). The values here are averages of the Spearman's Rank-Order Correlation over the 5 test sets.

Our linear approximations for the number of convention errors per sentence using the class states for the overall scores are better for persuasive and narrative essay prompts. This indicates that these scores depend on convention errors to a greater degree for persuasive and narrative essay prompts. These are also the set of essays in which convention is explicitly included in the rubric. On prompts 2 and 7, our best estimates for the convention errors per sentence arise from models on the trait of conventions. Prompt number 8 was an exception, perhaps due to the smallest number of training samples and the lowest inter-rater agreements.

The Spearman rank-order correlation coefficients between the linear approximations of features from Uto and Uchida (2020) and the features themselves averaged over the five test sets are summarized in Table 9. The average Spearman rank-order

correlation coefficients presented are for linear approximations of features using the class states for overall scores only. They were not computed for trait-level scores.

Table 9

The Spearman's Rank-Order Correlation Coefficients between the Traditional Features and Linear Approximations of the Features on the Class States for Overall Scores.

Feature Classes	Features	1	2	3	4	5	6	7	8
Length Related Features	# of Words	0.945	0.937	0.961	0.953	0.977	0.952	0.948	0.591
	# of Sentences	0.812	0.841	0.844	0.845	0.852	0.792	0.895	0.647
	# of Commas	0.769	0.704	0.718	0.751	0.730	0.695	0.654	0.734
	# of Question Marks	0.452	0.446	0.116	0.094	0.000	0.046	0.481	0.380
	# of Exclamation Marks	0.448	0.137	0.034	0.054	0.063	0.000	0.515	0.344
	Avg Word Length	0.623	0.661	0447	0.350	0.530	0.355	0.542	0.617
	Avg Sentence Length	0.373	0.068	0.392	0.398	0.420	0.255	0.527	0.458
Syntactic Features	# of Nouns	0.883	0.887	0.914	0.904	0.926	0.884	0.871	0.648
	# of Verbs	0.833	0.860	0.884	0.901	0.882	0.852	0.897	0.512
	# of Adjectives	0.776	0.803	0.778	0.778	0.780	0.807	0.735	0.691
	# of Adverbs	0.747	0.757	0.727	0.749	0.739	0.723	0.783	0.571
	# of Conjugations	0.663	0.711	0.693	0.751	0.738	0.524	0.690	0.550
Readability	Automated Readability	0.371	0.133	0.427	0.384	0.479	0.280	0.531	0.413
	Coleman-Liau	0.577	0.613	0.487	0.365	0.576	0.360	0.494	0.520
	Dale-Chall	0.673	0.312	0.492	0.325	0.447	0.386	0.545	0.545
	Ratio of Difficult Words	0.757	0.755	0.470	0.300	0.372	0.448	0.446	0.712
	Flesch Reading Ease	0.458	0.213	0.360	0.330	0.389	0.245	0.489	0.356
	Flesch Kancid Grade	0.370	0.130	0.394	0.366	0.425	0.246	0.518	0.418
	Gunning Fog	0.360	0.129	0.387	0.391	0.430	0.232	0.500	0.422
	Linsear Index	0.237	0.305	0.351	0.392	0.380	0.307	0.480	0.366
	Smog Index	0.353	0.318	0.581	0.738	0.580	0.432	0.364	0.348
	Syllable Count	0.945	0.933	0.958	0.955	0.967	0.941	0.947	0.613
Predicted Scores Based on CoLA	Linguistics Acceptability	0.647	0.614	0.196	0.334	0.285	0.421	0.634	0.541

In general, the correlations were high (usually above 0.7) for length related features. This includes the number of words, the number of sentences, the number of commas, the number of nouns, the number of verbs, the number of adjectives, and the number of adverbs as well as readability related features including syllable count. Length-based features and features that naturally scale with length are well approximated, indicating that length is implicitly used as a feature. The quality of these approximations also suggest that length is also important in score point determinations. Based on the average number of words in Table 1 and the correlations presented in Table 9, our approximation of length is better when the average response is shorter.

Average word length and the ratio of difficult words to the number of words are approximated well in essay prompts 1, 2, and 8, where word choice is a part of the rubrics. It should be noted that most readability measures are badly approximated

regardless of what traits are used, indicating that most readability indexes are not being used by neural networks.

Further, we considered the feature defined as linguistic acceptability, which is the output probabilities of a model trained on the CoLA (Sanh et al., 2020). The average of the Spearman's rank-order correlation coefficients between the set of linear models on the class states and the linguistic acceptability over the 5 tests sets is presented in Table 9 as well. The linear models approximating linguistic acceptability did not display high correlations generally, indicating that linguistic acceptability is generally not approximated well as a linear function of the class states. However, the highest correlation values were obtained for persuasive and narrative essay prompts. This may indicate that the models trained for persuasive and narrative prompts have a much stronger dependence on linguistic acceptability in their score predictions.

Saliency Methods and Local Explainability

We present the results of two responses chosen that are relatively short and contain a relatively high number of spelling and grammatical errors. The heat maps for the saliency values of two selected responses from the ASAP dataset using a DeBERTa model trained on conventions are presented in Figure 2. In these heat maps, the background colors were chosen between white and red, where words with white backgrounds had no negative effect on score while those with red backgrounds had a larger negative effect on score.

To clarify more general trends, we processed approximately 100 responses of approximately the same length using a convention model for prompt 7 and considered how words were used in assigned scores. This gave us an assessment of the contributions for approximately eighteen thousand words. We used a vocabulary of approximately 100k words to determine whether a word is in a vocabulary (In-Vocab) or not in the vocabulary (Out-of-Vocab). This count excluded punctuation and special tokens used in the competition, such as those beginning with ``\@" in Figure 2.

Table 10 summarizes the saliency statistics. The application of the saliency map gave us an evaluation of the contribution of each word in the responses tested. An average Z-score of -0.769 for Out-of-Vocab words indicates that the misspellings, on average, had a much more detrimental effect to scores than their correctly spelled versions.

Figure 2

Saliency Map for Two Sample Responses

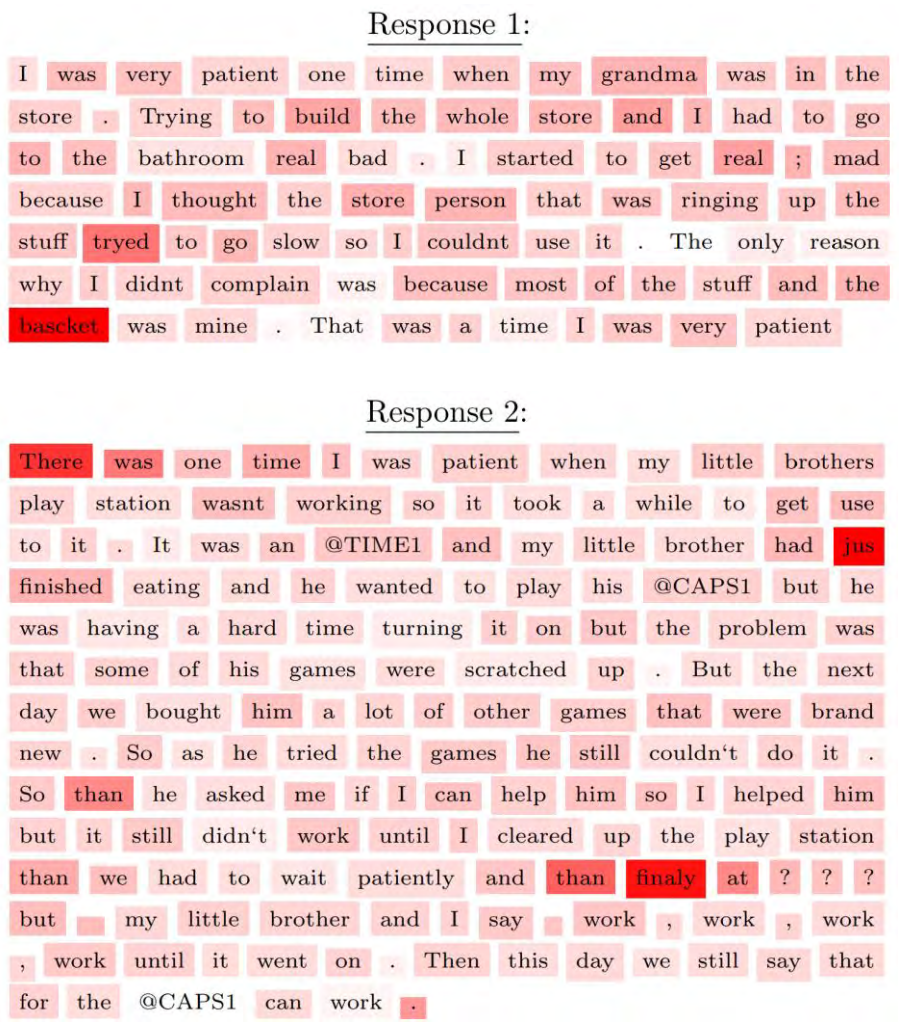


Table 10

Saliency Statistics

	Average Score	Standard Deviation	Z-Score	N
In Vocab	6.93×10^{-3}	8.87×10^{-3}	0.018	18608
Out of Vocab	-0.114×10^{-3}	0.01116	-0.769	422
Overall	6.80×10^{-3}	8.99×10^{-3}		18608

Summary and Discussion

This study demonstrated the development of deep neural networks based DeBERTa models for automated scoring of the overall and trait-level scores and the approximations of rubric-relevant information or features about a response within the class states. Overall, this study found that the number of errors per sentence was better approximated by the class states for overall scores and all traits when convention errors are part of the rubrics. This suggests that the neural networks may be using some version of this feature in its calculations for all traits in persuasive and narrative prompts. This could be as validity evidence of the scores automatically generated by these models. Further, length-based features are implicitly used by the DeBERTa models and are important to score point predictions even though one of the major criticisms of AES is that length is too important in score predictions (Perelman, 2013).

It is noted that a multitude of factors could have had a detrimental effect on the quality of these approximations. The two major issues are data quality and quantity. The inter-rater agreements at the trait level in the original convention data (Shermis, 2014), were low by production standards. As the ASAP++ data were obtained from one human rater, there was no adequate information to evaluate the scoring quality. Further, the trait-level scores were very highly correlated. Thus, it was difficult to determine whether the neural networks have used the non-rubric trait level information in score-point determinations. In the absence of high-quality annotated data, the GEC models only provide heuristics for the number of errors. This heuristic is potentially prone to the same types of agreement issues as hand-scoring. Further, the distribution of saliency values and their corresponding words shows more clearly that some spelling errors were penalized more than others. A raw total number of errors does not account for that variation. This is also true in hand-scoring. On the other hand, data quantity is a general issue in using the ASAP dataset. Typically, the datasets needed to train GEC models and other language models are generally much larger than the training sets used to train AES models (Mizumoto et al., 2011; Ng et al., 2014; Stahlberg & Kumar, 2021; Yannakoudakis et al., 2011). We should not expect the same quality of inferences from AES models trained on comparatively small quantities of data.

One approach we considered to ensure high correlations for the features was to add the loss function from the linear model on the class states to the loss function used to produce the scores. This modification would force the neural network to encode rubric

relevant information in the class states. This would be a novel way to implicitly incorporate features was originally proposed by Uto and Uchida (2020). This approach could address both validity issues and possibly increase performance though this would not address explainability for models more generally. This is a future research direction.

In general, the DeBERTa models trained on overall scores performed comparably with the current state-of-the-art. The trait-level AES models performed significantly better than those obtained in Mathias and Bhattacharyya (2018, 2020). One technique to improve the QWK of the models in this study is known as hyperparameter tuning. The literature on both AES and ASAS suggests that a very modest increase in final test QWK could be achieved by employing Tree-Parzan Estimator-based hyperparameter tuning (Ormerod, 2022; Ormerod et al., 2021; Snoek et al., 2012). Hyperparameters tuning involves training models at multiple batch sizes and learning rates for each trait and prompt in order to obtain optimal hyperparameters. While this would help to identify the optimal batch sizes and learning rates, hyperparameter tuning is difficult to implement without significantly more computational power.

The final optimization employed in this study with respect to using variable cutoffs is merely an extension of Yang et al. (2020). By optimizing the QWK as a function of variable cutoff values on the development set and applying those cutoff points to the test set gave an average increase in QWK of 0.004 on the test set compared with using fixed cutoff values. This method could have been utilized to address a different problem known in AES. By using the variable cutoff points to fit the distributions of scores instead of maximizing QWK, we potentially resolve the well-known issue that AES scores tend to regress to the mean more than human scores. More work needs to be done on the metrics used to gauge regression to the mean.

The models built are specific to these prompts in Shermis (2014), and hence, the results do not directly apply to generic essay grading or systems used in AWE systems. The results for an engine trained on a sufficiently large and broad corpus of essays would be more applicable to an AWE system.

References

Attali, Y. (2013). Validity and Reliability of Automated Essay Scoring. In *Handbook of Automated Essay Evaluation*. Routledge.

Attali, Y., & Burstein, J. (2006). Automated Essay Scoring With e-rater® V.2. *The Journal of Technology, Learning and Assessment*, *4*(3), Article 3. https://ejournals.bc.edu/index.php/jtla/article/view/1650

Bryant, C., Felice, M., & Briscoe, T. (2017). Automatic Annotation and Evaluation of Error Types for Grammatical Error Correction. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 793–805. https://doi.org/10.18653/v1/P17-1074

Clark, K., Luong, M.-T., Le, Q. V., & Manning, C. D. (2020). *ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators* (arXiv:2003.10555). arXiv. https://doi.org/10.48550/arXiv.2003.10555

Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, *20*(1), 37–46. https://doi.org/10.1177/001316446002000104

Cohen, R. J., Swerdlik, M. E., & Phillips, S. M. (1996). *Psychological testing and assessment: An introduction to tests and measurement, 3rd ed* (pp. xxviii, 798). Mayfield Publishing Co.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding* (arXiv:1810.04805). arXiv. https://doi.org/10.48550/arXiv.1810.04805

Dong, F., Zhang, Y., & Yang, J. (2017). Attention-based Recurrent Convolutional Neural Network for Automatic Essay Scoring. *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, 153–162. https://doi.org/10.18653/v1/K17-1017

Felice, M., Bryant, C., & Briscoe, T. (2016). Automatic Extraction of Learner Errors in ESL Sentences Using Linguistically Enhanced Alignments. *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 825–835. https://aclanthology.org/C16-1079

Ge, T., Wei, F., & Zhou, M. (2018). *Reaching Human-level Performance in Automatic Grammatical Error Correction: An Empirical Study* (arXiv:1807.01270). arXiv. http://arxiv.org/abs/1807.01270

He, P., Liu, X., Gao, J., & Chen, W. (2021). *DeBERTa: Decoding-enhanced BERT with Disentangled Attention* (arXiv:2006.03654). arXiv. https://doi.org/10.48550/arXiv.2006.03654

Linardatos, P., Papastefanopoulos, V., & Kotsiantis, S. (2021). Explainable AI: A Review of Machine Learning Interpretability Methods. *Entropy*, *23*(1), Article 1. https://doi.org/10.3390/e23010018

Loshchilov, I., & Hutter, F. (2019). *Decoupled Weight Decay Regularization* (arXiv:1711.05101). arXiv. https://doi.org/10.48550/arXiv.1711.05101

Lundberg, S. M., & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems*, *30*. https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html

Mathias, S., & Bhattacharyya, P. (2018, May). ASAP++: Enriching the ASAP Automated Essay Grading Dataset with Essay Attribute Scores. *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. LREC 2018, Miyazaki, Japan. https://aclanthology.org/L18-1187

Mathias, S., & Bhattacharyya, P. (2020). Can Neural Networks Automatically Score Essay Traits? *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, 85–91. https://doi.org/10.18653/v1/2020.bea-1.8

Mizumoto, T., Komachi, M., Nagata, M., & Matsumoto, Y. (2011). Mining Revision Log of Language Learning SNS for Automated Japanese Error Correction of Second Language Learners. *Proceedings of 5th International Joint Conference on Natural Language Processing*, 147–155. https://aclanthology.org/I11-1017

Napoles, C., Sakaguchi, K., & Tetreault, J. (2017). JFLEG: A Fluency Corpus and Benchmark for Grammatical Error Correction. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, 229–234. https://aclanthology.org/E17-2037

Ng, H. T., Wu, S. M., Briscoe, T., Hadiwinoto, C., Susanto, R. H., & Bryant, C. (2014). The CoNLL-2014 Shared Task on Grammatical Error Correction. *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, 1–14. https://doi.org/10.3115/v1/W14-1701

Ormerod, C. (2022). *Short-answer scoring with ensembles of pretrained language models* (arXiv:2202.11558). arXiv. https://doi.org/10.48550/arXiv.2202.11558

Ormerod, C. M., Malhotra, A., & Jafari, A. (2021). *Automated essay scoring using efficient transformer-based language models* (arXiv:2102.13136). arXiv. https://doi.org/10.48550/arXiv.2102.13136

Page, E. B. (2003). Project Essay Grade: PEG. In *Automated essay scoring: A cross-disciplinary perspective* (pp. 43–54). Lawrence Erlbaum Associates Publishers.

Perelman, L. C. (2013). Critique of Mark D. Shermis & Ben Hamner, "Contrasting State-of-the-Art Automated Scoring of Essays: Analysis." *Journal of Writing Assessment*, *6*(1). https://escholarship.org/uc/item/7qh108bw

Powell, M. J. D. (1964). An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, *7*(2), 155–162. https://doi.org/10.1093/comjnl/7.2.155

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2020). *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer* (arXiv:1910.10683). arXiv. https://doi.org/10.48550/arXiv.1910.10683

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why Should I Trust You?": Explaining the Predictions of Any Classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135–1144. https://doi.org/10.1145/2939672.2939778

Ridley, R., He, L., Dai, X., Huang, S., & Chen, J. (2021). Automated Cross-prompt Scoring of Essay Traits. *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*(15), Article 15.

Rodriguez, P. U., Jafari, A., & Ormerod, C. M. (2019). *Language models and Automated Essay Scoring* (arXiv:1909.09482). arXiv. https://doi.org/10.48550/arXiv.1909.09482

Rothe, S., Mallinson, J., Malmi, E., Krause, S., & Severyn, A. (2021). *A Simple Recipe for Multilingual Grammatical Error Correction* (arXiv:2106.03830). arXiv. https://doi.org/10.48550/arXiv.2106.03830

Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, *18*(11), 613–620. https://doi.org/10.1145/361219.361220

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2020). *DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter* (arXiv:1910.01108). arXiv. https://doi.org/10.48550/arXiv.1910.01108

Shermis, M. D. (2014). State-of-the-art automated essay scoring: Competition, results, and future directions from a United States demonstration. *Assessing Writing*, *20*, 53–76. https://doi.org/10.1016/j.asw.2013.04.001

Smilkov, D., Thorat, N., Kim, B., Viégas, F., & Wattenberg, M. (2017). *SmoothGrad: Removing noise by adding noise* (arXiv:1706.03825). arXiv. https://doi.org/10.48550/arXiv.1706.03825

Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian Optimization of Machine Learning Algorithms. *Advances in Neural Information Processing Systems*, *25*. https://proceedings.neurips.cc/paper/2012/hash/05311655a15b75fab86956663e1819cd-Abstract.html

Srinivasa-Desikan, B. (2018). *Natural Language Processing and Computational Linguistics: A practical guide to text analysis with Python, Gensim, spaCy, and Keras*. Packt Publishing Ltd.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*. 30.

Stahlberg, F., & Kumar, S. (2021). *Synthetic Data Generation for Grammatical Error Correction with Tagged Corruption Models* (arXiv:2105.13318). arXiv. https://doi.org/10.48550/arXiv.2105.13318

Sundararajan, M., Taly, A., & Yan, Q. (2017). Axiomatic Attribution for Deep Networks. *Proceedings of the 34th International Conference on Machine Learning*, 3319–3328. https://proceedings.mlr.press/v70/sundararajan17a.html

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. *Advances in Neural Information Processing Systems*, *27*. https://proceedings.neurips.cc/paper/2014/hash/a14ac55a4f27472c5d894ec1c3c743d2-Abstract.html

Taghipour, K., & Ng, H. T. (2016). A Neural Approach to Automated Essay Scoring. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 1882–1891. https://doi.org/10.18653/v1/D16-1193

Uto, M., & Uchida, Y. (2020). Automated Short-Answer Grading Using Deep Neural Networks and Item Response Theory. *AIED*. https://doi.org/10.1007/978-3-030-52240-7_61

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is All you Need. *Advances in Neural Information Processing Systems*, *30*. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

Vilone, G., & Longo, L. (2020). *Explainable Artificial Intelligence: A Systematic Review* (arXiv:2006.00093). arXiv. https://doi.org/10.48550/arXiv.2006.00093

Wallace, E., Tuyls, J., Wang, J., Subramanian, S., Gardner, M., & Singh, S. (2019). *AllenNLP Interpret: A Framework for Explaining Predictions of NLP Models* (arXiv:1909.09251). arXiv. https://doi.org/10.48550/arXiv.1909.09251

Warstadt, A., Singh, A., & Bowman, S. R. (2019). *Neural Network Acceptability Judgments* (arXiv:1805.12471). arXiv. https://doi.org/10.48550/arXiv.1805.12471

Williamson, D. M., Xi, X., & Breyer, F. J. (2012). A Framework for Evaluation and Use of Automated Scoring. *Educational Measurement: Issues and Practice*, *31*(1), 2–13. https://doi.org/10.1111/j.1745-3992.2011.00223.x

Wingate, U. (2012). 'Argument!' helping students understand what essay writing is about. *Journal of English for Academic Purposes*, *11*(2), 145–154. https://doi.org/10.1016/j.jeap.2011.11.001

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., … Rush, A. M. (2020). *HuggingFace's Transformers: State-of-the-art Natural Language Processing* (arXiv:1910.03771). arXiv. https://doi.org/10.48550/arXiv.1910.03771

Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A., & Raffel, C. (2021). *mT5: A massively multilingual pre-trained text-to-text transformer* (arXiv:2010.11934). arXiv. https://doi.org/10.48550/arXiv.2010.11934

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. *Findings of the Association for Computational Linguistics: EMNLP 2020*, 1560–1569. https://doi.org/10.18653/v1/2020.findings-emnlp.141

Yannakoudakis, H., Briscoe, T., & Medlock, B. (2011). A New Dataset and Method for Automatically Grading ESOL Texts. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 180–189. https://aclanthology.org/P11-1019

Appendix A

Benchmarking Grammatical Error Correction

This appendix is dedicated to an evaluation of the quality of the counts we provide for convention errors. Of the several benchmarks, we use the JHU FLuency-Extended GUG (JFLEG) corpus (Napoles et al., 2017). The corpus is made of a test set and a development set with approximately 750 sentences each. The benchmark provided for a GEC engine is a GLEU score, which compares the output of a corrected sentence against four possible grammatically correct variations of each sentence in the corpus. The GLEU score for a GEC engine is interpreted as a measure of grammatical acceptability of the output. The results with no edits were used as a baseline model. The GLEU scores of the baseline model, the T5 model in this study, LangTool, and the best performing model on the test set as reported in Ge et al. (2018) are presented in Table A.1. The *Best Performing Model* of Ge et al. (2018), is a sequence-to-sequence model with a 7-layer convolutional encoder and decoder. This model is not publicly available. The *Baseline Model* contains the GLEU scores when the original sentences are unaltered and the *Human* reports the human level annotations reported by Ge et al. (2018). The relatively good performance of the model we use in this study ensures that the combination of the model output and ERRANT provide a reasonably accurate approximation of convention errors.

Table A.1

GLEU Scores for GEC Models

DataSets	Models	Variation 1	Variation 2	Variation 3	Variation 4	Average
Development DataSet	Baseline Model	0.340	0.320	0.411	0.460	0.383
	LangTool	0.426	0.388	0.493	0.556	0.466
	T5	0.508	0.475	0.565	0.618	0.541
Test DataSet	Baseline Model	0.434	0.452	0.397	0.334	0.404
	LangTool	0.542	0.560	0.493	0.556	0.506
	T5	0.628	0.637	0.589	0.535	0.597
	Best Performing Model	N/A	N/A	N/A	N/A	0.624
	Human	N/A	N/A	N/A	N/A	0.623

Note: The best performing model is the one on the test set from Ge et al. (2018).