

# How Much Does Hyperparameter Tuning Actually Help? An Efficiency Survey for Fine-Tuning Transformers to Score Mathematics-Explanation Items

Gregory M. Jacobs   Ahmed H. Bediwy   Martha Bellows

Pearson Education

{greg.jacobs, ahmed.bediwy, martha.bellows}@pearson.com

## Abstract

Fine-tuning pre-trained transformer models for constructed-response items often begins with a hyperparameter grid search using  $k$ -fold cross-validation. We study how much that search actually helps by fine-tuning two encoders—MathBERT, a smaller math-focused model, and DeBERTa-v3-large, a larger general-purpose model—to score 18 math-explanation items from a state-wide assessment. Crossing learning rate, weight decay, and label smoothing over two epochs and five folds (1,800 model-fold runs), we compare each model’s tuning gain directly against fold-to-fold noise via a gain-to-noise ratio. Gains were small relative to fold noise for both encoders: MathBERT’s ratio fell below one (0.90), and DeBERTa’s nominally higher ratio (1.42) traced to a handful of divergent fits rather than an informative search landscape. Weight decay and label smoothing were effectively inert, leaving learning rate as the only hyperparameter worth checking—though even for learning rate the best configurations offered small performance gains over a reasonable default. We accordingly recommend a lean workflow that fixes the inert hyperparameters to sensible defaults, runs a narrow learning-rate search extended modestly upward, and increases the epoch budget with early stopping, substantially reducing training compute without sacrificing accuracy.

## 1 Introduction

Fine-tuning a pre-trained transformer model for scoring a constructed-response item conventionally begins with hyperparameter (HP) tuning. A common approach to HP is a grid search wrapped in  $k$ -fold cross-validation (Morris et al., 2025). The approach is so standard that it is rarely scrutinized. Most experiments simply enumerate a grid across a set of parameters, evaluate each configuration by an appropriate cross-validated metric, and keep the highest performing parameter values.

For an operational program that includes dozens of mathematics-explanation items at multiple grades for every administration cycle, this search can be quite resource intensive because the number of training runs scales at a rate of items  $\times$  configurations  $\times$  folds. For example, even a modest grid of a dozen configurations over five folds already encompasses sixty fine-tuning runs *per item* before a single production model is trained.

Yet the assumption that exhaustive tuning is *necessary* is seldom scrutinized. One of the main challenges of this process is that each cross-validated QWK estimate is calculated on finite validation folds and therefore carries its own sampling noise. Refit the same configuration on different folds and the estimate often moves. When the apparent benefit of searching (the gap between the best configuration and a reasonable default) is smaller than this run-to-run variability, the search is effectively selecting noise. It will report a “winner,” but that winner is not reliably reproducible and uncovers no dependable operational advantage. Quantifying this tradeoff could be significant when designing a fine-tuning pipeline that is both accurate and efficient at scale for a state-wide assessment program.

Conversely, if a particular model hyperparameter value genuinely needs tuning, a program must know which one so that it does not implement an overly lean pipeline that can result in degraded models. The goal therefore is to measure, per model and per hyperparameter, *when* tuning helps and *when* it does not, and to turn that into a concrete recipe.

This paper surveys how effective HP tuning actually is for automated scoring of a set of math-explanation items, using two encoders: MathBERT, a smaller, math-focused pretrained model (Shen et al., 2021), and DeBERTa-v3-large, a larger general-purpose model (He et al., 2021, 2023). Through this experimentation, we seek to answer the following research questions: (RQ1) Which hy-

hyperparameters, if any, improve performance when fine-tuning a smaller, math-focused encoder or a larger, general-purpose encoder to score math-explanation items, and (RQ2) what approach to hyperparameter tuning, if any, should be implemented when designing a training pipeline to fine-tune encoders for performing operational scoring of math-explanation items at scale?

## 2 Related Work

**Hyperparameter optimization.** The theory of hyperparameter optimization is well-developed. [Bergstra and Bengio \(2012\)](#) showed that random search is often as effective as grid search and far cheaper because only a few hyperparameters typically matter. Our study is complementary and specific to automated scoring. Rather than proposing a better search strategy, we ask whether searching is worthwhile at all once the resulting gains are compared against the sampling noise of the very metric used to select configurations.

**Stability of fine-tuning.** Parallel literature documents that fine-tuning large pretrained encoders can be unstable, with substantial variance across random seeds, data orders, and learning rates ([Dodge et al., 2020](#); [Mosbach et al., 2021](#)). This instability is precisely the fold noise our diagnostics seek to quantify. If run-to-run variability is large relative to between-configuration differences, a grid search will ineffectively chase variance. We connect this general finding to the operational scoring setting and to a specific observed failure—the occasional divergence of the larger encoder at higher learning rates.

**Cross-validation variance.** Selecting the configuration with the highest cross-validated score amounts to maximizing over noisy performance estimates, making the selected configuration’s apparent advantage susceptible to selection-induced optimism ([Cawley and Talbot, 2010](#); [Varma and Simon, 2006](#)). The problem becomes more pronounced when the model-selection criterion has substantial variance, since model selection can itself overfit the cross-validation criterion ([Cawley and Talbot, 2010](#)). Moreover,  $k$ -fold cross-validation has nontrivial variance arising from the dependence among folds, and this uncertainty is not generally estimable without bias ([Bengio and Grandvalet, 2004](#)). Our gain-to-noise ratio provides a direct diagnostic of whether the observed tuning

gain is large relative to this estimation noise. When the tuning gain is at or below the fold-noise floor, the “winner” may, to a first approximation, be the configuration that benefited most from validation noise rather than the configuration that generalizes best. Thus, the methodological question shifts from “which configuration wins?” to “is the observed win distinguishable from cross-validation noise?”.

**Automated scoring and evaluation.** Automated scoring is evaluated against human-to-human reliability using ordinal agreement statistics ([Cohen, 1968](#); [Pearson and ETS, 2015](#); [Williamson et al., 2012](#); [Yan et al., 2020](#)), and the 2023 NAEP Mathematics Automated Scoring Challenge showed that fine-tuned transformers can reach human-like agreement on such items ([Crossley et al., 2023](#); [NAEP Research and Development Program, 2023](#)). Those systems emphasized peak accuracy through elaborate augmentation and ensembling. We instead study the *efficiency* of the tuning stage for operational deployment, treating compute as a constraint.

**The structure of fine-tuning models for operational scoring.** The resources required to fine-tune automated scoring models scale with the number of training runs, which a hyperparameter grid search multiplies by configurations and folds. Because these runs can reoccur every administration and across many items, even a constant-factor reduction compounds. Prior work on scoring accuracy sometimes fails to account for this, but we instead place it on the same compute-versus-accuracy footing as the rest of the pipeline.

## 3 Corpus

### 3.1 Data

Every statistic in this paper is computed from a single administration of 18 math-explanation items from one statewide summative assessment. The items span from grades 3 through 8 and apply rubrics with scorepoint ranges of either four (0-3) or five (0-4) integer values.

### 3.2 Human scoring

Responses were scored under an operational double-read procedure whereby 10% of responses received a second score to measure human-to-human reliability. For both training and evaluation, we take the target label to be `best_human`: the program’s final reported score, determined by a

predefined hierarchy in which an adjudication or resolution score takes precedence over the first and second reads. Human-human reliability is taken from the first and second reads on the subset of responses that received a second human score.

### 3.3 Response pre-processing

Operational responses arrive in a semi-structured delivery format rather than as clean text, and they carry markup that a language model should not see verbatim. Before fine-tuning, we apply a deterministic cleaning pipeline: we unwrap the JSON delivery format, decode HTML entities and strip presentational tags, convert embedded MathML into a plain-text rendering so that symbols, fractions, and operators survive as readable tokens, and normalize Unicode. The pipeline is identical across all items and both encoders, so no difference between models can be attributed to pre-processing.

## 4 Hyperparameter Optimization Methodology

### 4.1 Broad Hyperparameter Search Grid.

The exploratory survey varied learning rate, weight decay, and label-smoothing  $\alpha$  (Table 1). MathBERT crossed two values of each hyperparameter (8 configurations); DeBERTa-v3-large used three rates and two values for the other hyperparameters (12). Each configuration received a two-epoch budget and was evaluated by five-fold stratified cross-validation, yielding 1,800 model-fold runs (720 for MathBERT and 1,080 for DeBERTa). Because we are modeling math explanation items, the hyperparameters chosen to survey, and the breadth of their values included in the grid search, were based on the experimentation reported out from the 2023 NAEP Mathematics Automated Scoring Challenge (Crossley et al., 2023; NAEP Research and Development Program, 2023). Table 1 outlines the grid search values used for each encoder.

For each hyperparameter configuration, an item’s 5,000 response training set was randomly divided into five 1,000 response folds for cross-validation. For each configuration/fold combination, a model is trained on four folds and validated on the held-out fold, using QWK as the validation metric. We then calculated the mean QWK value across the five folds. Figure 1 visually displays this five-fold cross-validation process.

| Hyperparameter     | Values searched                                        |
|--------------------|--------------------------------------------------------|
| Learning rate      | 1e-5, 2e-5 (MathBERT);<br>1e-5, 1.5e-5, 2e-5 (DeBERTa) |
| Weight decay       | 0.01, 0.05 (MathBERT);<br>0.1, 0.3 (DeBERTa)           |
| Smoothing $\alpha$ | 0.5, 0.7                                               |
| Batch size         | 4 (grad. accumulation)                                 |
| Epochs             | 2                                                      |
| Scheduler          | linear with warmup                                     |

Table 1: Hyperparameter grid search space. Grids yielded 8 (MathBERT) and 12 (DeBERTa-v3-large) configurations over 5 folds.

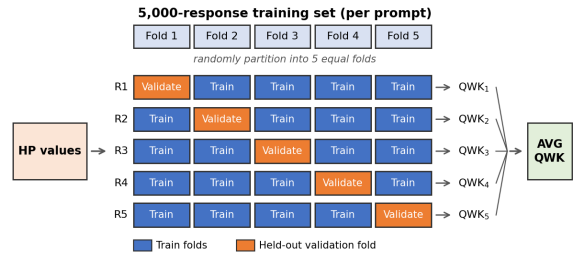


Figure 1: Five-fold cross-validated grid search. For a given hyperparameter configuration, the training set is partitioned into five folds; the model is trained on four folds and validated on the held-out fold, and the five QWK scores are averaged. This is repeated for every configuration in the grid.

### 4.2 Evaluation

For every prompt  $\times$  model  $\times$  configuration we compute the mean and standard deviation (SD) of validation QWK across the five folds (Figure 1). From these we derive four diagnostics.

**Tuning gain** the best-configuration mean QWK minus the average QWK across all configurations. This is what a search buys relative to picking a configuration at random from the grid.

**Tuning range** the best minus the worst configuration mean QWK. This is the most a search could possibly buy, i.e. the spread the grid exposes.

**Fold noise** the SD of the best configuration’s QWK across its five folds. This is ordinary run-to-run variability at fixed hyperparameters.

**Gain-to-noise ratio** tuning gain divided by fold noise. A ratio below 1.0 means the benefit of searching is smaller than ordinary run-to-run variation and is therefore not reliably reproducible.

Formally, configurations by  $c \in \{1, \dots, C\}$  and folds by  $j \in \{1, \dots, 5\}$ , and let  $q_{cj}$  be the validation QWK of configuration  $c$  on fold  $j$ . Write  $\bar{q}_c = \frac{1}{5} \sum_j q_{cj}$  for a configuration’s mean and  $c^* = \arg \max_c \bar{q}_c$  for the selected configuration. Then, for a prompt,

$$\text{gain} = \bar{q}_{c^*} - \frac{1}{C} \sum_c \bar{q}_c, \quad (1)$$

$$\text{range} = \max_c \bar{q}_c - \min_c \bar{q}_c, \quad (2)$$

$$\text{noise} = \left( \frac{1}{4} \sum_j (q_{c^*j} - \bar{q}_{c^*})^2 \right)^{1/2}, \quad (3)$$

$$\text{gain-to-noise} = \text{gain}/\text{noise}. \quad (4)$$

We report these per prompt and average across prompts within a model.

For each individual hyperparameter we additionally tabulate (i) how often each grid value “wins” (is chosen in the best configuration), (ii) its marginal mean QWK averaged over the other knobs, and (iii) whether the winning value sits at a grid edge. A winning value at a grid edge suggests that the grid might benefit from being *extended* rather than that tuning has converged. Finally, we examine the number of training epochs by aggregating each model’s validation-QWK-versus-epoch curve.

**Gain versus range.** It is worth distinguishing the tuning *gain* from the tuning *range*. The range (best minus worst) can look large only because the grid includes a few genuinely bad configurations. But the relevant quantity is the gain over a *sensible* default, not the worst, which is what the gain diagnostic measures. A wide range with a small gain is the signature of a grid whose spread is skewed by avoidable bad settings rather than by meaningfully different good ones.

## 5 Results

Table 2 reports the tuning effectiveness by model. Across models the tuning gains were small and mostly of the same magnitude as fold noise. Table 3 reports the percentage each hyperparameter value resulted in the best performing configuration for each model. These results are analyzed for each model in the following two subsections.

### 5.1 MathBERT is robust to its hyperparameters

MathBERT was strikingly insensitive to the grid search parameters. The mean best QWK was 0.928, but the tuning gain was only 0.0053 against fold

| Model            | Mean best QWK | Tuning gain | Tuning range | Fold noise | Gain/noise |
|------------------|---------------|-------------|--------------|------------|------------|
| MathBERT         | 0.928         | 0.0053      | 0.0107       | 0.0058     | 0.90       |
| DeBERTa-v3-large | 0.931         | 0.0106      | 0.0629       | 0.0075     | 1.42       |

Table 2: Tuning effectiveness by model (grid search, validation QWK). A gain-to-noise ratio below 1.0 means the search bought less than ordinary run-to-run variation.

| Model    | Hyperparameter     | Winning value   | Win rate  |
|----------|--------------------|-----------------|-----------|
| MathBERT | Learning rate      | 2e-5 (grid max) | 100%      |
|          | Weight decay       | 0.01/0.05       | 0.61/0.39 |
|          | Smoothing $\alpha$ | 0.5             | 67%       |
| DeBERTa  | Learning rate      | 2e-5 (grid max) | 78%       |
|          | Weight decay       | 0.1/0.3         | no pref.  |
|          | Smoothing $\alpha$ | 0.5             | 56%       |

Table 3: Per-hyperparameter win rates. For DeBERTa-v3-large the winning learning rate sits at the top of the searched range, but the marginal differences across values are within fold noise.

noise of 0.0058. With a resulting gain-to-noise ratio of only 0.90, the average gain of searching the grid was *smaller* than the variation from re-running the best configuration on different folds.

Learning rate 2e-5 won on all 18 prompts, but with an edge of only about 0.007 QWK over the lower rate. So, even the “wrong” learning rate cost almost nothing. Weight decay was effectively inert: the two grid values split wins 0.61/0.39 with a negligible marginal difference, indicating no reliable preference. Label-smoothing  $\alpha = 0.5$  modestly outperformed  $\alpha = 0.7$  (winning 67% of prompts). Taken together with the sub-1.0 gain-to-noise ratio, the per-parameter picture is that no MathBERT hyperparameter reliably matters within the searched ranges. A single sensible default performs as well as the grid winner once fold noise is accounted for.

### 5.2 DeBERTa-v3-large is subject to occasional divergence

For DeBERTa-v3-large, the gain-to-noise ratio (1.42) in the aggregate indicates significance, but the per-parameter details show that this might not be the case. The marginal mean QWK of the three learning rates differs by less than 0.004—smaller than the fold noise—so no learning-rate value is reliably better on average. What inflates the tuning *range* (and consequently the gain over the cross-configuration average) is a small number of unstable fits: 6 of the 216 DeBERTa configurations collapsed to a QWK between 0.67 and 0.79, against roughly 0.92 for the rest. These collapses are not

tied to a single learning rate hyperparameter value. They occur at  $1.5e-5$  and  $2e-5$  but never at the lowest rate ( $1e-5$ ). This is consistent with the known instability of fine-tuning large encoders (Mosbach et al., 2021; Dodge et al., 2020). DeBERTa-v3-large’s favorable ratio is therefore evidence that one must always be cognizant of the occasional divergent fit when fine-tuning a larger encoder model.

Interestingly, the winning learning rate sat at the top of the searched range ( $2e-5$ ) for 78% of prompts, yet the divergences also occurred at the two higher rates. The best runs accordingly occurred at the slightly higher rates, but that same region carries the divergence risk. The sensible response therefore is to keep the learning rate in the conservative  $1-2e-5$  band and retain a one- or two-point check to catch the rare collapse rather than to refine an interior grid.

## 6 Discussion

Our first research question asks which hyperparameters, if any, improve performance when fine-tuning a smaller, math-focused encoder or a larger, general-purpose encoder to score math-explanation items. We address this question, organized by hyperparameter, below.

### 6.1 Learning Rate

As noted in the previous section, tuning the learning rate hyperparameter on the grid values used in the experiment did not meaningfully impact model performance in the majority of circumstances for either encoder (with the exception of resulting in a rare divergence for DeBERTa models). It is noteworthy, however, that the best performing MathBERT models used the grid’s maximum learning rate value ( $2e-5$ ) for every model. The best performing DeBERTa models similarly used the grid’s maximum learning rate value ( $2e-5$ ) a majority of the time (78%). This suggests that it may be prudent to extend the grid search values upward for any future hyperparameter tuning of these models for math explanation items, even if it is unlikely to significantly improve performance over the default values.

### 6.2 Weight decay and label smoothing

Neither the weight decay or label smoothing behaved like a lever worth tuning. Weight decay was effectively inert for both encoders, splitting wins near evenly with a marginal effect negligible

relative to fold noise. Weight decay resulted in a slight preference of 0.01 (61%) for MathBERT models and no preference between 0.1 and 0.3 for DeBERTa models. Label-smoothing  $\alpha$  showed a mild, consistent preference for 0.5 over 0.7 for both models. These results indicate that both hyperparameter values can be *fixed* rather than searched in future hyperparameter tuning of encoder models for scoring math explanation items.

### 6.3 Number of training epochs

Both models show significant improvement between the first and second training epochs. MathBERT mean validation climbed from 0.889 at one epoch to 0.928 at two epochs. DeBERTa-v3-large rises from 0.878 at one epoch to 0.931 at two epochs. This increase between one and two epochs suggests that increasing the number of epochs beyond 2 could bear additional gains, though there does not appear to be much more to gain at mean QWK values exceeding 0.9 for both models.

## 7 Recommendations

Our second research question asks what approach to hyperparameter tuning, if any, should be implemented when designing a training pipeline to fine-tune encoders for performing operational scoring of math-explanation items at scale. The experimentation results support a workflow that includes a targeted learning rate grid search as the sole hyperparameter tuned for both smaller math-specific and larger general purpose encoders. We recommend the following model training approach when fine-tuning these models for scoring math-explanation items:

1. **Fix all non-learning rate hyperparameter values.** Set reasonable default values for weight-decay, smoothing, scheduler, and batch settings rather than including them in any hyperparameter grid search. Based on the winning values observed in this study, we suggest the following defaults:
  - **Weight decay:** 0.01 for MathBERT (won 61% of prompts); 0.1 for DeBERTa-v3-large (the two grid values were effectively tied, so the more conservative value is retained).
  - **Label smoothing  $\alpha$ :** 0.5 for both encoders (consistently preferred over 0.7).

- **Scheduler:** linear schedule with warmup for both encoders.

- **Batch size:** 4 with gradient accumulation of 4 for both encoders, resulting in an effective batch size of 16.

2. **Implement informed learning rate grid search.** Because the best configurations for both encoders overwhelmingly favored the top of the range of grid search values for learning rate ( $2e-5$ ), we suggest the following targeted learning-rate grids for future trainings:

- **MathBERT:**  $\{2, 3, 4, 5\}e-5$ , extending the grid search to capture any marginal gains that may exist at the higher rates, and
- **DeBERTa-v3-large:**  $\{1.5, 2, 3, 4\}e-5$ , kept in the conservative band to capture the best runs while retaining a low-rate anchor that guards against the occasional divergent fit at higher rates.

3. **Increase number of epochs with early stopping.** Because validation QWK increased between one and two epoch trainings for both encoders, we recommend increasing the number of training epochs, particularly for the smaller MathBERT encoder, while using a more conservative increase for the larger DeBERTa-v3-large encoder. We also recommend implementing early stopping with the expectation that results could plateau within a few additional epochs given the large validation QWK values produced after two training epochs.

Because search cost scales as items  $\times$  configurations  $\times$  folds  $\times$  training epochs, streamlining the hyperparameter tuning approach as recommended above will result in a significant savings in training time and resources.

## 8 Limitations

This experiment includes a group of 18 constructed response math items from one administration, so the results are generalizable within a caveat of a fairly limited sample size and subject matter. Formal equivalence tests and cross-administration replication remain for future investigative work on this research.

## 9 Conclusion

This research surveyed how much hyperparameter tuning actually helps when fine-tuning MathBERT and DeBERTa-v3-large encoders to score math-explanation items. Across the 18 items, tuning gains were small relative to fold noise for both encoders: MathBERT’s gain-to-noise ratio fell below one (0.90), and DeBERTa’s nominally higher ratio (1.42) traced to mainly to a handful of divergent fits rather than an informative search landscape. Weight decay and label smoothing were effectively inert and can be fixed to sensible defaults, leaving learning rate as the only hyperparameter worth tuning using an informed and targeted grid search, though even for learning rate there the best configurations offered small performance gains over a reasonable default.

These findings support a lean, compute-efficient hyperparameter tuning approach: fix the non-learning-rate hyperparameters to defaults, run a narrow learning-rate grid search extended modestly upward, and increase the epoch budget with early stopping to capture any additional gains that may be attainable by training beyond two epochs. Because tuning cost scales with items, configurations, folds, and epochs, this streamlined approach substantially reduces training time and compute while preserving the small, prompt-specific benefits that tuning could provide. Formal equivalence testing and cross-administration replication remain promising directions for confirming these conclusions.

## References

- Yoshua Bengio and Yves Grandvalet. 2004. No unbiased estimator of the variance of k-fold cross-validation. *Journal of Machine Learning Research*, 5:1089–1105.
- James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305.
- Gavin C. Cawley and Nicola L. C. Talbot. 2010. On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11(70):2079–2107.
- Jacob Cohen. 1968. Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological Bulletin*, 70(4):213–220.
- Scott Crossley, Perpetual Baffour, Tian Yu, Alex Franklin, Meg Benner, and Ulrich Boser. 2023. A large-scale corpus for assessing written argumentation: PERSUADE 2.0.
- Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith.

2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping.
- Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. In *International Conference on Learning Representations (ICLR)*.
- Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced BERT with disentangled attention. In *International Conference on Learning Representations (ICLR)*.
- Wesley Morris, Langdon Holmes, Joon Suh Choi, and Scott Crossley. 2025. Automated scoring of constructed response items in math assessment using large language models. *International Journal of Artificial Intelligence in Education*, 35(2):559–586.
- Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning BERT: Misconceptions, explanations, and strong baselines. In *International Conference on Learning Representations (ICLR)*.
- NAEP Research and Development Program. 2023. [NAEP math automated scoring challenge results and preprint](#).
- Pearson and ETS. 2015. [Research results of parcc automated scoring proof of concept study](#). Technical report, Pearson and ETS. Accessed September 2026.
- Jia Tracy Shen, Michiharu Yamashita, Ethan Prihar, Neil Heffernan, Xintao Wu, Ben Graff, and Dongwon Lee. 2021. MathBERT: A pre-trained language model for general NLP tasks in mathematics education. *arXiv preprint arXiv:2106.07340*.
- Sudhir Varma and Richard Simon. 2006. Bias in error estimation when using cross-validation for model selection. *BMC Bioinformatics*, 7(1):91.
- David M. Williamson, Xiaoming Xi, and F. Jay Breyer. 2012. A framework for evaluation and use of automated scoring. *Educational Measurement: Issues and Practice*, 31(1):2–13.
- Duanli Yan, André A. Rupp, and Peter W. Foltz. 2020. Handbook of automated scoring: Theory into practice. *Chapman and Hall/CRC*.