

Transformer-Aided Detection of Gaming in Constructed-Response English Language Assessments

Melanie Sharif¹, Scott Hellman^{2*}, Martha Bellows¹, Susan Lottridge¹

¹Pearson plc ²Macmillan Learning

Abstract

Gaming responses in constructed-response English language assessments can threaten score validity, making robust detection an important function for assessment organizations. We compare handcrafted-feature baselines with frozen transformer embeddings and embedding-feature ensembles for gaming detection across three constructed-response item types under familiar-prompt and unseen-prompt evaluation regimes, and we test whether a single pooled model can replace item-type specialists. Transformer representations improve agreement with gaming labels in most conditions, and an embedding-feature ensemble is the best configuration in every item-type/regime split, or statistically indistinguishable from it, with the best configuration reaching ROC-AUC = 0.95 (Cohen's κ = 0.75). In the lowest-prevalence item type under unseen prompts, however, the ensemble provides no significant gain over handcrafted features, and frozen embeddings alone are markedly weaker there. A pooled model performs comparably to item-type specialists in the familiar-prompt regime. We also show that these results are bounded by the label-generation process: review process labels record gaming that was detected, while missed gaming is recorded as negative. Reported metrics therefore estimate agreement with the existing review process rather than construct-level detection accuracy. We argue for supplementing standard agreement metrics with reference-conditioned recall and targeted blind re-review of candidate misses.

Keywords: automated scoring, gaming detection, constructed response, transformer embeddings, label quality, validity, label noise

1 Introduction

1.1 Gaming as a validity threat

Automated scoring in constructed-response language assessment relies on the assumption that produced responses provide evidence of the target construct (Mislevy et al., 2003; Williamson et al., 2012). When test-takers substitute memorized templates, copied prompt or passage material, filler, and/or generic text for a construct-relevant response, fooling content models, this is called gaming by practitioners (Hughes and Clesham, 2024). It is also collectively known as construct-irrelevant response strategies, or CIRS (Bejar et al., 2014).

Gaming is thus a direct attack on test validity, as automated scoring systems can reward surface features that resemble valid responses even when the underlying construct evidence is weak or absent (Higgins and Heilman, 2014; Ding et al., 2020). Gaming detection therefore becomes a validity and quality-assurance function. Some detectors rely on handcrafted indicators such as template overlap and prompt or passage copying; these features encode known failure modes but can miss strategies not anticipated in advance (Samant et al., 2025).

Transformer encoders offer a complementary approach by representing semantic and syntactic structure without requiring each gaming pattern to be manually specified, motivating the comparison in this study.

1.2 Label noise and interpretive constraints

Gaming detection differs from many automated scoring tasks because its value lies in identifying gaming that the current review process misses, not merely reproducing existing labels. This creates a measurement problem: a response receives a positive gaming label only if it has already been caught, while gaming missed by review is recorded as negative. The observed labels therefore contain one-sided noise concentrated in the negative class.

* Work done while at Pearson plc

This is a positive–unlabeled setting: positives are observed only when detection succeeds, and the negative class mixes true negatives with undetected positives (Elkan and Noto, 2008; Bekker and Davis, 2020). A model that flags previously missed gaming may be scored as producing a false positive, even when the flag is useful.

Importantly, conventional accuracy, agreement, precision, and recall consequently estimate agreement with the existing review process rather than pure detection accuracy against the construct. We therefore interpret all performance figures as label-agreement estimates and supplement them with metrics aimed at the deployment question: whether a model recovers positives the reference detector missed. Coverage of missed gaming is the quantity of interest, while κ and AUC serve to rank model quality.

Because test-takers adapt to current detection methods, the target category is adversarial and non-stationary. A detector model’s ability to catch latent gaming features is therefore crucial for providing support to human reviewers. Highly sensitive models can help scorers better identify gaming over time, which can gradually improve model training sets and close the coverage gap.

1.3 A note on scoring policies

The corpus does not support an inter-rater reliability (IRR) or “human ceiling” comparison: no response was scored blind by two humans. Instead, the second read for responses in this corpus is a read-behind of the first, so IRR is un-estimated.

The reference labels are also not independent of the systems under evaluation. Adjudication is triggered by a disagreement of more than two score points on a content trait between any two reads, where a positive automated gaming label functions as a content-trait read of score 0. Since the reference detector participates in producing the labels against which new models are scored, agreement with those labels partly reflects shared machinery rather than independent construct evidence (§1.2).

1.4 Paper Contributions

The main contributions of this paper are as follows. First, we evaluate transformer-aided modeling, embedding–feature ensembles, and the handcrafted-feature approach on the gaming detection task across three constructed-response item types and two generalization regimes: familiar prompts and unseen prompts. Second, we evaluate

pooled models across item types alongside specialist models to explore whether a pooled model is a defensible choice.

Third, we present an evaluation protocol for labels that are detection outcomes rather than ground truth. This includes reference-conditioned recall that distinguishes between gaming the reference detector caught versus gaming it missed, and prioritizing prevalence-robust agreement metrics (accuracy, Cohen’s κ , recall, and ROC-AUC), with ROC-AUC as the primary criterion for both model selection and reporting; precision and F1 are class-dependent, are reported on the balanced condition only, and are not used for selection or cross-condition comparison (§4.1, §4.4).

Finally, we explain why gaming detection is difficult to evaluate under existing labeling conditions and propose concrete, inexpensive remedies that can move this area toward better measurement practice.

2 Related work

2.1 Gaming detection in automated scoring

Automated essay scoring (AES) systems infer a latent writing construct from surface features of a response, and a substantial literature has shown that this inference can be subverted by construct-irrelevant response strategies (CIRS) collectively termed gaming (Bejar et al., 2013, 2014; Higgins and Heilman, 2014; Ding et al., 2020; Hughes and Clesham, 2024; Samant et al., 2025). One of the most conspicuous examples remains the BABEL generator, whose grammatical gibberish nonetheless earned near-perfect scores from production scoring engines (Perelman et al., 2014). This work prompted dedicated detection advisories in response (Cahill et al., 2018), which Perelman (2020) argued did not fully resolve the validity problem. Crucially, the shift from feature-based to deep learning models did not resolve this fragility: neural scorers have been characterized as both overstable and oversensitive, assigning high scores to adversarial inputs produced by repeating, shuffling, deleting, or injecting content and by inserting off-topic or nonsensical text (Lottridge et al., 2020; Ding et al., 2020; Kabra et al., 2020; Singla et al., 2023). As a result of these persistent vulnerabilities, gaming detection remains a vital asset for assessment organizations.

Recent work has focused on detecting templated text as a proxy for inauthentic responses – an ad-

Item type	N	Human-labelled
Type A	111,362	107,789
Type B	637,162	586,661
Type C	167,000	166,564
All	915,524	861,014

Table 1: Corpus size and number of human labels by item type.

versarial strategy distinct from previous ones documented in the literature (Samant et al., 2025). Despite this targeted approach, a gap remains between what handcrafted features like templates and text patterns can catch and the scope of adversarial strategies in use. The sheer volume of linguistic features present in large transformer models, combined with the strength of handcrafted features, could offer a substantial next step in detection.

3 Data

3.1 Assessment context and corpus

Data are drawn from an adult English language proficiency assessment with three constructed-response item types (Type A, Type B, and Type C).

In addition to content scores, every response across item types is subject to an automated gaming review that produces a ternary label — High-Confidence Gaming (GH), Low-Confidence Gaming (GL), and No Gaming (NG). The ternary labels have been collapsed to binary labels for the sake of this study.

3.2 Corpus size

We use a corpus of 915,524 responses, of which 861,014 (94.0%) carry a positive or negative human gaming label. The remaining 6% of responses without a human label are removed from the response pool. For prompt counts and train/test prompt splits, see §4.1.

4 Method

4.1 Splits and generalization regimes

We construct class balanced 50/50 train/test splits per item type, plus a pooled model trained across all three, under two regimes:

- **within_prompt** — random assignment; prompts appear in both train and test. This corresponds to deployment against a known item bank.

Regime	Type	train	test	Train gaming	Test gaming
within_prompt	Type A	6,000	2,000	3,000	1,000
within_prompt	Type B	1,756	390	878	195
within_prompt	Type C	6,000	2,000	3,000	1,000
within_prompt	all	5,756	1,722	2,878	861
cross_prompt	Type A	6,000	2,000	3,000	1,000
cross_prompt	Type B	1,620	394	810	197
cross_prompt	Type C	6,000	2,000	3,000	1,000
cross_prompt	all	5,558	1,746	2,779	873

Table 2: Train/test sizes and class label balance by item type and prompt regime.

- **cross_prompt** — prompt-disjoint, constructed within each item type; no prompt appearing in training appears in test. This corresponds to the cold-start case, a newly authored item before any of its responses have been reviewed.

Splits were requested at 6,000 train / 2,000 test. Type A and Type C achieve this; Type B and the pooled splits fall short because human-labelled positives run out, due to the extremely low prevalence of gaming in that item type (Table 2).

To ensure that enough human-labelled responses are available for inclusion in the test set, the test set is capped at the proportional share the train/test ratio implies, grouped by item type so that a pooled split cannot cover a shortfall in one type by over-drawing another.

Test sets are class balanced 50/50 by construction, standard practice for evaluating classifiers under severe class imbalance (He and Garcia, 2009). We apply the same 50/50 balancing to all three item types so that agreement metrics are computed under a common class distribution across item types and for the pooled model, keeping cross-type and pooled-versus-specialist comparisons on a common footing (Table A1). Prompt-disjoint (cross_prompt) prompt counts are Type A 14/6, Type B 67/14, Type C 20/7, and pooled 95/33 (train/test). Recall and ROC-AUC are class-invariant and transfer to deployment prevalence unchanged.

4.2 Models

There are two baseline model configurations per item type split. The baseline feature set comprises handcrafted features: for Types A and B, type-token ratio, comma density, minimum template distance, and authentic token percentage; for Type C, the number and percentage of authentic tokens over-

all, within templated text, and within prompt text, together with minimum template distance. The extended baseline set adds five runtime classical text features: passage similarity, bigram overlap, entity overlap, key-term coverage, and content-word ratio. Both feature sets are then used in an ElasticNet-regularized logistic regression model.

For the transformer-aided modeling, there are four encoder configurations: BGE-large-en-v1.5 (bi-encoder), longformer-base-4096, longformer-large-4096, and longformer-base-4096 used as a cross-encoder, taking the [CLS] state over a joint prompt–response input. The cross-encoder is included as a deliberate design contrast: the bi-encoders never see prompt and response together, so any gaming signal that consists of a mismatch between the two must be recovered downstream rather than attended to directly. Embeddings are reduced by PCA over {5, 10, 20, 50, 100, 200, 500}, skipping dimensions above the training matrix rank. Each encoder’s embeddings are then passed to three classifiers evaluated separately — ElasticNet logistic regression, random forest, and RBF SVM — so that each encoder–classifier pair is a distinct pipeline rather than a combined model.

An ensemble method is also tested by selecting the best embedding source, chosen by nested cross-validation, and concatenating with the handcrafted feature block. PCA is then applied to the embedding block only, and logistic regression is used as the classifier.

4.3 Selection protocol

Nested cross-validation is used for model selection: an inner loop searches hyperparameters within each outer training fold, and the outer loop estimates how the whole selection procedure generalizes (Cawley and Talbot, 2010). All preprocessing — PCA and scaling — is inside the pipeline and refit per fold. The test set is scored exactly once and is never consulted for selection. Model configurations are selected by mean inner-fold ROC-AUC, a threshold-free criterion aligned with the primary reporting metric (§4.4); F1 is not used for selection. The nested cross-validation uses five folds, and random seed 42 is fixed throughout.

4.4 Metrics

Since our labels are review process outcomes rather than an independent measure of the construct (§1.2, §1.3), conventional agreement metrics are necessary but not sufficient. Table 3 summarizes the

metrics reported throughout the study and the purpose each serves.

All agreement metrics are reported at each classifier’s default decision rule (0.5). No threshold tuning is applied, and no operating-point deployment claims are made. ROC-AUC is the primary metric reported throughout: it is threshold-free — scoring ranking quality across all possible decision thresholds rather than at a single cut — prevalence-robust, and is also the criterion used for model selection (§4.3), so selection and reporting use the same metric.

We additionally evaluate performance using Cohen’s κ , which scores each classifier at its default 0.5 decision rule rather than across thresholds. In test sets balanced 50/50 by construction (§4.1), chance agreement is fixed at 0.5 and κ reduces to a linear rescaling of accuracy ($\kappa = 2 \cdot \text{accuracy} - 1$). We report κ rather than raw accuracy because it is the standard in this literature and keeps comparisons legible across item types. Pooled item type slices are also class balanced (Table 7).

5 Results

5.1 Best configuration per split

Table 4 reports the best configuration for each item type and prompt regime, using held-out test data; the longformer-large encoder stood out, appearing in every winning configuration except Type A under familiar prompts. No cross-encoder configuration was selected in any split. The final column shows the change in kappa when moving from familiar prompts to unseen prompts. Table 5 then compares model families using ROC-AUC, the same threshold-free metric used for model selection.

5.2 Test set performance by model family

Across all eight splits, the embedding–feature ensemble improves ROC-AUC over the handcrafted baseline, with gains ranging from +0.05 (Type B, cross-prompt) to +0.22 (Type C, within-prompt). The ensemble is the best-performing configuration in every split except within-prompt Type B, where it is statistically indistinguishable from the best frozen embedding (Table 5).

5.3 Reference-conditioned recall

Recall on positives the reference detector already flagged is uniformly high (0.94–0.99), whereas recall on positives the reference detector missed is

Metric group	What is reported	Interpretive role
Conventional agreement	Cohen’s kappa, F1, precision, recall, and ROC-AUC with 95% percentile bootstrap confidence intervals from 2,000 resamples; McNemar’s test for paired model comparisons on identical test items.	Supports model-to-model comparisons within the observed label set. Label bias affects both arms of a paired comparison, so differences remain interpretable even when the labels are imperfect.
Reference-conditioned recall	Recall on positives the reference detector already flagged; recall on positives the reference detector did not flag.	Separates easy agreement with the reference detector from the more valuable case: recovering human-confirmed gaming the reference detector missed.
Unconfirmed flags	Model-positive / label-negative records. Traditional false positives.	Not treated as ordinary false positives, because this cell contains both model errors and possible genuine catches missed by the review process. This is the target population for the blind re-review proposed in §6.

Table 3: Study evaluation metrics by group along with reporting and interpretative information.

lower and more variable (0.57–0.84). The gap is modest under familiar prompts (roughly 0.13–0.17) but widens under unseen prompts for Type A and Type B (to 0.38 and 0.34); even so, the model recovers a nontrivial share of missed gaming in every condition (Table 6).

5.4 Pooled model vs. specialists

The pooled model matches or slightly trails the item-type specialists in every slice, with Cohen’s κ differences ranging from 0.000 (Type B, within-prompt) to -0.076 (Type A, within-prompt); the specialist estimate exceeds the pooled slice’s bootstrap interval only for within-prompt Type A (Table 7).

The full metric set behind this comparison is reported in Table A1 (Appendix A).

6 Discussion

The first finding is that transformer representations improve agreement with gaming labels in most item types and regimes, but not uniformly. For Type B under cross-prompt evaluation — the lowest-prevalence, most data-constrained condition — the ensemble is statistically indistinguishable from handcrafted features (McNemar $p = 0.65$). Frozen embeddings therefore provide no significant gain there. Frozen embeddings alone are also markedly weak in this condition (ROC-AUC = 0.7914; $\kappa \approx 0.21$), with the handcrafted block carrying the ensemble (Table 5, Table 6).

The second finding is that unseen prompts are seemingly influenced by the default cutoff rather than ranking quality. ROC-AUC drops are modest and uniform across item types (Δ ROC-AUC ≈ -0.03 to -0.04). The larger drops in κ (-0.06

to -0.15 , largest for Type A) therefore appear to reflect miscalibration of the default 0.5 threshold under prompt shift rather than a comparable loss of ranking ability. This argues for item-type-specific decision thresholds at deployment.

The third finding is that a single pooled detector is plausible in the familiar-prompt regime. Specialist advantages are small and sit within the pooled model’s uncertainty intervals. Under cross-prompt evaluation, the pooled model also remains close to the specialists, but all item-type slices show some loss relative to specialist models, leaving a stronger case for specialization in that regime (Table 7, Table A1).

The main interpretive constraint is still the label-generation process. As established in §1.2, the reported metrics estimate agreement with the existing review process rather than construct-level accuracy. Reference-conditioned recall and unconfirmed flags make this blind spot visible, but they cannot determine how many model-positive, label-negative cases are genuine missed gaming. That requires new blind review data.

A targeted blind re-review is the most direct remedy. Rather than relabeling the corpus, a small stratified sample sourced from the model labels could be reviewed without model scores or prior labels, with two independent reviewers per response.

Oversampling the model-positive/label-negative cell would estimate the rate of genuine gaming among possible misses, while sampling true negatives and true positives would allow reweighting to the population. A sample in the low hundreds would be sufficient to distinguish whether a model’s additional flags are mostly noise or include substantial missed gaming, and would also provide

Item type	Regime	Winning configuration	ROC-AUC	κ	Δ
Type A	within-prompt	ensemble: longformer-base PCA=100 + handcrafted	0.8787 [0.863, 0.893]	0.6130 [0.580, 0.647]	—
Type A	cross-prompt	ensemble: longformer-large PCA=20 + handcrafted	0.8402 [0.823, 0.856]	0.4650 [0.427, 0.502]	−0.1480
Type B	within-prompt	ensemble: longformer-large PCA=200 + handcrafted	0.9481 [0.925, 0.968]	0.7487 [0.683, 0.810]	—
Type B	cross-prompt	ensemble: longformer-large PCA=10 + handcrafted	0.9105 [0.881, 0.938]	0.6548 [0.578, 0.727]	−0.0939
Type C	within-prompt	ensemble: longformer-large PCA=200 + handcrafted	0.9278 [0.917, 0.938]	0.6950 [0.663, 0.725]	—
Type C	cross-prompt	ensemble: longformer-large PCA=200 + handcrafted	0.8964 [0.883, 0.910]	0.6310 [0.597, 0.665]	−0.0640
Pooled	within-prompt	ensemble: longformer-large PCA=200 + handcrafted	0.8986 [0.883, 0.913]	0.6330 [0.595, 0.668]	—
Pooled	cross-prompt	ensemble: longformer-large PCA=50 + handcrafted	0.8597 [0.843, 0.876]	0.5590 [0.521, 0.596]	−0.0740

Table 4: Best-performing configuration per item type and prompt regime, selected by ROC-AUC, with κ and delta κ also reported. Bracketed values indicate 95% percentile bootstrap confidence intervals.

the reliability estimate the present corpus lacks.

Limitations

The study has three main limitations. First, no blind double-read is available, so we have no inter-rater reliability information, and the reference standard is not independent of the detector that helps produce it (§1.2, §1.3).

Second, recorded negatives may contain undetected positives, so precision and recall are relative to observed labels and should not be read as construct-level accuracy.

Third, cross-prompt uncertainty is understated by response-level bootstrap intervals because prompt-level variance is not fully captured, especially for item types with few held-out prompts.

Conclusion

Transformer representations improve gaming detection over handcrafted features across most item types and both generalization regimes. A single pooled model may be defensible in the familiar-prompt regime, while frozen embeddings provide no significant advantage in the most data-constrained unseen-prompt condition. These findings support transformer-aided gaming detection, but only as estimates of agreement with review process labels.

The main constraint is the label-generation process. Review process labels identify gaming that was detected; they do not reveal how much gaming was missed. Positive-unlabeled class-prior estimation can sometimes estimate the share of hidden positives among unlabeled cases (Elkan and Noto, 2008; Bekker and Davis, 2020), but this requires assumptions that are unlikely to hold when the reference detector catches the most conspicuous cases. The missed-positive rate therefore cannot be recovered from observed labels alone.

Future studies should report how labels were produced, whether reviewers saw model output, whether review was routed by a reference detector, and whether any blind double-scoring was available. When such evidence is absent, reliability should be described as un-estimated rather than low. When feasible, targeted blind re-review of model-positive/label-negative cases should be added.

References

- Isaac I. Bejar, Michael Flor, Yoko Futagi, and Chaitanya Ramineni. 2014. On the vulnerability of automated scoring to construct-irrelevant response strategies (cirs): An illustration. *Assessing Writing*, 22:48–59.
- Isaac I. Bejar, Waverly Hester VanWinkle, Nitin Madnani, William Lewis, and Michael Steier. 2013. Length of textual response as a construct-irrelevant

Regime	Type	Baseline	Extended	Best frozen embedding	Ensemble
within_prompt	Type A	0.7421 [0.722, 0.764]	0.8052 [0.787, 0.825]	0.8721 [0.856, 0.887]	0.8787 [0.863, 0.893]
within_prompt	Type B	0.8468 [0.806, 0.886]	0.8962 [0.864, 0.926]	0.9491 [0.929, 0.968]	0.9481 [0.925, 0.968]
within_prompt	Type C	0.7057 [0.683, 0.728]	0.7757 [0.756, 0.795]	0.9266 [0.916, 0.937]	0.9278 [0.917, 0.938]
within_prompt	all	0.7431 [0.720, 0.765]	0.7781 [0.756, 0.799]	0.8894 [0.873, 0.905]	0.8986 [0.883, 0.913]
cross_prompt	Type A	0.7151 [0.692, 0.736]	0.7960 [0.775, 0.816]	0.8125 [0.794, 0.830]	0.8402 [0.823, 0.856]
cross_prompt	Type B	0.8589 [0.821, 0.894]	0.9100 [0.880, 0.937]	0.7914 [0.748, 0.832]	0.9105 [0.881, 0.938]
cross_prompt	Type C	0.7230 [0.701, 0.746]	0.7856 [0.766, 0.805]	0.8910 [0.877, 0.905]	0.8964 [0.883, 0.910]
cross_prompt	all	0.7467 [0.724, 0.769]	0.7786 [0.757, 0.800]	0.8463 [0.828, 0.863]	0.8597 [0.843, 0.876]

Table 5: Test-set ROC-AUC for handcrafted, frozen-embedding, and ensemble configurations, per item type and regime. Highest ROC-AUC values are bolded per row. Within each family, the reported configuration is the one selected by highest nested cross-validation ROC-AUC on training data.

Regime	Type	Recall (reference detector caught)	Recall (reference detector missed)	Unconfirmed flags (traditional false positives)
within_prompt	Type A	0.9487 (n=117)	0.7754 (n=846)	184
within_prompt	Type B	0.9355 (n=62)	0.7752 (n=129)	15
within_prompt	Type C	0.9684 (n=95)	0.8420 (n=861)	156
cross_prompt	Type A	0.9505 (n=101)	0.5672 (n=841)	141
cross_prompt	Type B	0.9881 (n=84)	0.6514 (n=109)	28
cross_prompt	Type C	0.9579 (n=95)	0.8237 (n=868)	205

Table 6: Recall on positives the reference detector flagged versus positives it missed, with unconfirmed flag counts. Higher recall on reference-caught than reference-missed positives indicates that the model partly reproduces reference-detector signal, while nontrivial missed-cell recall captures the potential added coverage.

- response strategy: The case of shell language. *ETS Research Report Series*, 2013(1):i–39.
- Jessa Bekker and Jesse Davis. 2020. Learning from positive and unlabeled data: A survey. *Machine Learning*, 109(4):719–760.
- Aoife Cahill, Martin Chodorow, and Michael Flor. 2018. Developing an e-rater advisory to detect babel-generated essays. *Journal of Writing Analytics*, 2(1):203–224.
- 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*, 11:2079–2107.
- Yuning Ding, Brian Riordan, Andrea Horbach, Aoife Cahill, and Torsten Zesch. 2020. Don’t take “nswvt-nvakgxp” for an answer: The surprising vulnerability of automatic content scoring systems to adversarial input. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 882–892.
- Charles Elkan and Keith Noto. 2008. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 213–220.
- Haibo He and Edwardo A. Garcia. 2009. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284.
- Derrick Higgins and Michael Heilman. 2014. Managing what we can measure: Quantifying the susceptibility of automated scoring systems to gaming behavior. *Educational Measurement: Issues and Practice*, 33(3):36–46.
- Sarah R. Hughes and Rose Clesham. 2024. Maintaining test integrity through integration of human judgement and ai systems. Technical report, Pearson.
- Anubha Kabra, Mehar Bhatia, Yaman Kumar Singla, Junyi Jessy Li, and Rajiv Ratn Shah. 2020. Calling out bluff: Attacking the robustness of automatic scor-

Regime	Type	n	Pooled model (slice)	Specialist (own test)	Δ
within_prompt	Type A	666	0.5375 [0.474, 0.604]	0.6130	−0.0755
within_prompt	Type B	390	0.7487 [0.677, 0.806]	0.7487	—
within_prompt	Type C	666	0.6607 [0.604, 0.718]	0.6950	−0.0343
cross_prompt	Type A	666	0.4565 [0.388, 0.523]	0.4650	−0.0085
cross_prompt	Type B	414	0.6280 [0.555, 0.700]	0.6548	−0.0268
cross_prompt	Type C	666	0.6186 [0.556, 0.674]	0.6310	−0.0124

Table 7: Pooled model vs. specialist model Cohen’s kappa scores by prompt type and regime. The pooled model is scored within each item-type slice of its own test set; each specialist is scored on its own test set. Bracketed figures show the 95% percentile bootstrap intervals (2,000 resamples) on the pooled slice; specialist figures are point estimates. Here, pooled slices do not easily allow for ROC-AUC to be computed, so κ is reported.

ing systems with simple adversarial testing. *arXiv preprint arXiv:2007.06796*.

A Supplementary Results

Susan Lottridge, B. Godek, A. Jafari, and M. Patel. 2020. Comparing the robustness of deep learning and classical automated scoring approaches to gaming strategies. White paper. <https://files.port.al.cambiumast.com/corporate-site/documents/CAI-Cambium-Comparing-Robustness-Automated-Scoring-Approaches.pdf>.

Robert J. Mislevy, Linda S. Steinberg, and Russell G. Almond. 2003. On the structure of educational assessments. *Measurement: Interdisciplinary Research and Perspectives*, 1(1):3–62.

Les Perelman. 2020. The babel generator and e-rater: 21st-century writing constructs and automated essay scoring (aes). *Journal of Writing Assessment*, 13(1).

Les Perelman, Louis Sobel, Milo Beckman, and Damien Jiang. 2014. Basic automatic b.s. essay language generator (babel). Software.

Yashad Samant, Lee Becker, Scott Hellman, Bradley Behan, Sarah Hughes, and Joshua Southerland. 2025. Automatic detection of inauthentic templated responses in english language assessments. *arXiv preprint arXiv:2509.08355*.

Yaman Kumar Singla, Swapnil Parekh, Somesh Singh, Junyi Jessy Li, Rajiv Ratn Shah, and Changyou Chen. 2023. Aes systems are both overstable and oversensitive: Explaining why and proposing defenses. *Dialogue & Discourse*, 14(1):1–33.

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.

Regime	Type	Metric	Pooled model (slice) [95% CI]	Specialist (own test)
within_prompt	Type A	kappa	0.538 [0.474, 0.604]	0.613
		F1	0.766 [0.730, 0.802]	0.805
		precision	0.775 [0.730, 0.820]	0.812
		recall	0.757 [0.711, 0.804]	0.797
		specificity	0.781 [0.737, 0.825]	0.816
within_prompt	Type B	kappa	0.749 [0.677, 0.806]	0.749
		F1	0.871 [0.831, 0.903]	0.868
		precision	0.897 [0.850, 0.939]	0.915
		recall	0.846 [0.795, 0.893]	0.826
		specificity	0.903 [0.858, 0.942]	0.923
within_prompt	Type C	kappa	0.661 [0.604, 0.718]	0.695
		F1	0.832 [0.800, 0.861]	0.848
		precision	0.824 [0.783, 0.863]	0.845
		recall	0.841 [0.801, 0.879]	0.851
		specificity	0.820 [0.778, 0.859]	0.844
cross_prompt	Type A	kappa	0.456 [0.388, 0.523]	0.465
		F1	0.746 [0.710, 0.780]	0.694
		precision	0.700 [0.655, 0.746]	0.811
		recall	0.799 [0.757, 0.843]	0.606
		specificity	0.658 [0.603, 0.707]	0.859
cross_prompt	Type B	kappa	0.628 [0.555, 0.700]	0.655
		F1	0.812 [0.770, 0.852]	0.822
		precision	0.822 [0.769, 0.874]	0.849
		recall	0.802 [0.746, 0.854]	0.797
		specificity	0.826 [0.775, 0.876]	0.858
cross_prompt	Type C	kappa	0.619 [0.556, 0.674]	0.631
		F1	0.816 [0.782, 0.846]	0.819
		precision	0.789 [0.745, 0.830]	0.803
		recall	0.844 [0.803, 0.882]	0.836
		specificity	0.775 [0.729, 0.818]	0.795

Table A1: Pooled model versus specialist (own-test) performance for all agreement metrics, by item type and prompt regime (both within- and cross-prompt), with 95% percentile bootstrap confidence intervals (2,000 resamples) on the pooled slice; specialist figures are point estimates. ROC-AUC is omitted because it is not computed on the pooled item-type slices; threshold-free ranking quality is reported per split in Table 4.

Split	Winner	vs. handcrafted	vs. frozen embedding
within_prompt/Type A	ensemble_longformer-base_PCA=100	$p=1.6\text{e-}12$ (270/128) **	$p=0.187$ (82/65)
within_prompt/Type B	ensemble_longformer-large_PCA=200	$p=0.016$ (34/16) *	$p=0.472$ (13/18)
within_prompt/Type C	ensemble_longformer-large_PCA=200	$p < 0.001$ (411/112) **	$p=0.775$ (26/23)
within_prompt/all	ensemble_longformer-large_PCA=200	$p < 0.001$ (295/92) **	$p=0.948$ (120/118)
cross_prompt/Type A	ensemble_longformer-large_PCA=20	$p=0.293$ (296/270)	$p=5.2\text{e-}13$ (121/31) **
cross_prompt/Type B	ensemble_longformer-large_PCA=10	$p=0.648$ (11/8)	$p=1.5\text{e-}06$ (85/32) **
cross_prompt/Type C	ensemble_longformer-large_PCA=200	$p < 0.001$ (356/152) **	$p=0.178$ (33/22)
cross_prompt/all	ensemble_longformer-large_PCA=50	$p=1.7\text{e-}11$ (269/133) **	$p=0.002$ (183/128) **

Table A2: McNemar paired comparisons between the winning configuration and each rival model family, by split. Parenthetical values give discordant-pair counts in the direction winner-only/rival-only. * $p < 0.05$; ** $p < 0.01$.