

Beyond the Grid: Auditing Code-Editing Search for Automated Essay Scoring

Julius Frost

Cambium Assessment

julius.frost@cambiumassessment.com

Abstract

Code-editing language-model agents can change training choices beyond the hyperparameter grids tested in automated essay scoring. Comparing these approaches requires distinguishing gains from a broader search space from evidence of a better search procedure. We compare code-editing agents with grid-restricted search, including random search, in two studies on ASAP-AES with nominally matched 12-hour search budgets. Code-editing produced the configuration with the highest test quadratic-weighted kappa (QWK) point estimate in each primary comparison. However, random search over a grid built afterwards around the Study 2 agent's backbone, input length and head rule recovered most of its gain over BERT. Requiring a minimum validation-score improvement to accept a trial also left the accepted configuration's validation score below the highest recorded valid validation score in every run that accepted a trial under this rule. The primary comparisons use one search run per condition, and some test folds reused essays involved in configuration selection, so these results compare selected configurations without establishing search-procedure superiority. These findings motivate reporting the available search choices and both accepted and best-scoring trials, and evaluating search procedures through repeated runs on data independent of configuration selection.

1 Introduction

Comparisons of code-editing search in automated essay scoring depend on the harness: its editable search space, acceptance rule, and evaluation procedure determine what an agent can propose and what the comparison can establish. We compare the test QWK of configurations selected by code-editing and grid-restricted search under nominally

matched 12-hour budgets, and examine what acceptance and evaluation records allow us to conclude. Our automated essay scoring (AES) base grids fix the backbone and loss; hyperparameter optimization (HPO) can also search these choices. Large language models (LLMs) can edit training code. Karpathy's `autoresearch` repository¹ set out the pattern: a single editable training file, a protected evaluator with one metric, and an accept-or-reject decision per trial. Ferreira et al. (2026) compared fixed-space HPO and LLM search, including source editing, on language-model pre-training.

Essay scoring's ordinal objective gives code-editing search choices beyond numeric hyperparameters: the loss, head, sampler, and score discretization can all change how the model learns or predicts scores. We compare search conditions on ASAP-AES from two starting configurations: one untuned to this dataset, the other adapted from an AES literature baseline. Screening selects configurations using validation scores on a prompt subset; we then evaluate them across prompts and folds.

Contributions. First, the code-editing configuration has the highest test-QWK point estimate in each primary comparison (Table 1). Random search over Study 2's grid, built afterwards around the code-editing configuration's backbone, input length and head rule, recovers most of that configuration's gain over BERT. Second, requiring a minimum improvement margin left accepted trials below the highest valid screening QWK in every run that accepted a trial under this rule; we report both recorded scores and trial identities (§4.2). Third, the primary comparisons use one search run per condition; on screened prompts, other folds' test splits reuse selection essays. The comparisons therefore concern selected configura-

¹<https://github.com/karpathy/autoresearch>

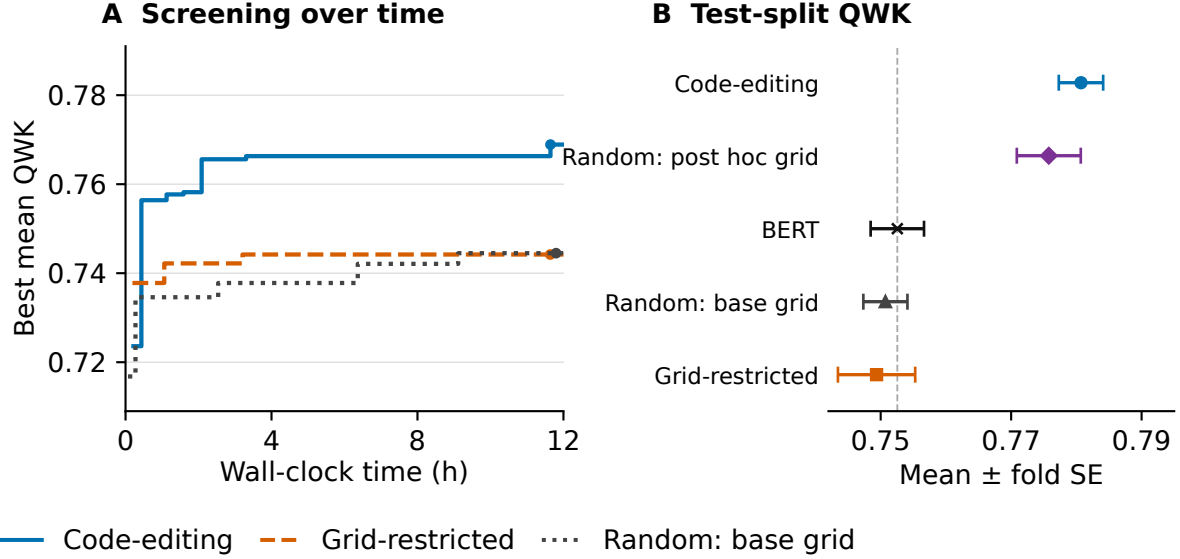


Figure 1: Code-editing has the highest test-QWK point estimate in Study 2’s primary comparison; its fold-SE interval overlaps that of post hoc random search (Table 1). A: single-seed best screening QWK with unequal starting scores; condition-specific time origins preclude speed comparisons (§4.3.2). B: corrected test QWK, mean $\pm$ fold standard error (SE) over 40 prompt-fold cells; dashed line, BERT. The code-editing result evaluates its accepted trial. The post hoc grid was built afterwards around that trial’s backbone, input length and head rule. Each search ran once; screened-prompt test folds reuse selection essays.

rations (§3.1.1; §3.4), not search-procedure performance. These findings motivate reporting permitted search choices and both accepted and best-scoring trials; they do not establish which harness design improves search quality.

2 Related Work

Agents that do machine-learning work. MLAGentBench (Huang et al., 2024) evaluates LLM agents on end-to-end ML experimentation tasks and MLE-bench (Chan et al., 2025) scores them against Kaggle leaderboards. In AIDE (Jiang et al., 2025) the agent is a tree search over solution scripts; Toledo et al. (2025) model such agents as search policies with explicit operator sets, comparing greedy, Monte Carlo tree search and evolutionary policies on MLE-bench lite. These works evaluate agents through task outcomes or search-policy comparisons. We compare code-editing search with grid-restricted HPO under nominally matched time budgets, examining how edit permissions and acceptance rules affect the comparison.

LLMs as optimizers. OPRO (Yang et al., 2024) uses an LLM to propose candidates for black-box optimization; Liu et al. (2025) apply the idea to hyperparameters, and OPT-BENCH (Li et al., 2026) benchmarks agents on large search

spaces. Code modification lets GrowthHacker (Wu et al., 2026) tune off-policy evaluators; van Stein et al. (2025) separate LLM-generated heuristic code from in-loop HPO. Closest to us, Ferreira et al. (2026) compared LLM agents with covariance matrix adaptation evolution strategy (CMA-ES) and the tree-structured Parzen estimator (TPE) inside Karpathy’s loop on a pretraining loss. The HPO methods performed better on a fixed hyperparameter space. Source editing narrowed the gap without closing it; an LLM–CMA-ES hybrid sharing the optimizer’s search state had the best mean. Their comparison clock excludes LLM inference, unlike ours. We examine ordinal scoring and use a post hoc grid to assess how much of the selected configuration’s gain random search recovers. We also distinguish accepted trials from those with the highest screening scores.

Li (2026) reports no benefit from simulated annealing over greedy hill climbing in two tested tasks with LLM proposals. Our acceptance-rule comparison (§3) reports individual search-run outcomes, not an estimate of policy equivalence (§5).

Baselines and task. Random search (Bergstra and Bengio, 2012), the strongest tested HPO comparator in Study 1, supplies Study 2’s fixed-grid comparator and post hoc diagnostic. The selected

ordinal losses belong to established surrogate-loss families (Hou et al., 2017; de la Torre et al., 2018). Our AES model and loss choices also overlap prior regression-and-ranking work (Yang et al., 2020; Xie et al., 2022); those ingredients could inform a prospective categorical baseline. Our AES baselines follow Ormerod and Kwako (2024) and the BERT configuration of Rodriguez et al. (2019); splits follow Taghipour and Ng (2016). Ormerod and Kehat (2025) study long-context AES, including ModernBERT, on ASAP 2.0 rather than our eight-prompt corpus.

3 Methods

3.1 Shared protocol

A *harness* is the execution, scoring, and budget machinery surrounding the agent. The *autoresearch loop* is the agent’s propose, edit, evaluate, and accept-or-reject cycle. A *condition* specifies the compared search method, edit permissions, acceptance rule, and budget; a *search run* is one budgeted execution of that condition. A *trial* is a candidate’s training/evaluation attempt, while a *configuration* fixes the training choices. Trial counts need not equal the number of distinct configurations (Appendix A). *Screening* selects configurations using validation data on a prompt subset. *Post-search retraining and test-split evaluation* evaluates a selected configuration on rotating folds; it does not supply an outer holdout for search (§3.1.1).

Our harness retains Karpathy’s workflow (Appendix H). Each original search run has a 12-hour wall-clock budget, including thinking and trials. Study 1 fixes epochs, input length and data fraction except for halving (§3.2.1); Study 2 permits configuration changes. The agent selects on validation; post-search runs score selected configurations on test splits. In Study 2 the launcher also removes read permission on test files. This within-run access rule does not establish independence between selection and test data across folds.

We replace Karpathy’s any-improvement rule with a requirement that a trial’s screening QWK exceed the incumbent score (the accepted configuration’s screening QWK) plus a minimum improvement margin estimated from baseline training-seed reruns (Appendix G). This sum is the acceptance threshold; equality fails. The margin is not a significance threshold. An accepted trial becomes the starting configuration for the next pro-

posal. It need not have the highest valid screening QWK: better-scoring trials can fall short of the acceptance threshold.

Trials are recorded in an experiment tracker. The agent instructions also require a git commit stating each proposed change’s rationale. A trial counts only if it trained, evaluated the required split at the required fidelity, and covered all four screening prompts (§3.1.1). The agent instructions (Appendix H) name the metric, files, minimum improvement margin and exit-code contract, and state that no additional dataset-specific knowledge is supplied. They retain Karpathy’s preference for simpler code and improvements obtained by deleting code. The agent otherwise receives a single launch line.

3.1.1 Testbed and metric

ASAP-AES has eight essay prompts with reported dataset score ranges from 0–3 (prompt 3) to 10–60 (prompt 8) (Ormerod and Kwako, 2024). We use supplied local split files following the five-fold 60/20/20 protocol of Taghipour and Ng (2016), scoring each prompt by QWK on the real score scale and averaging the eight prompt QWKs with equal weight (macro-averaging). In both studies the screening objective is the mean validation QWK across four prompts {2, 3, 5, 8} spanning the reported score-range extremes. Post-search evaluation retrains selected configurations on each fold’s training split, selects an epoch on validation, and scores its test split. The configuration, not trained weights, is reused across eight prompts $\times$ five folds (40 prompt-fold cells per condition). Prompts {1, 4, 6, 7} were not screened. On screened prompts, rotating test folds include essays from the selection population, so the aggregate is not a fully independent test of search.

3.2 Editable search space

3.2.1 Study 1: configuration not tuned to ASAP-AES

Study 1 starts from an Ettin/LoRA configuration that had not been tuned to ASAP-AES before the benchmark, unlike Study 2’s dataset-informed literature baseline. All five conditions train an Ettin-encoder-400m model (Weller et al., 2026) with low-rank adaptation (LoRA; Hu et al., 2022) through the shared harness with the same screening objective. Screening runs at reduced fidelity (4 epochs, max length 640, half the training data, one fold, one seed), under the shared search budget.

Grid search, random search, TPE (Bergstra et al., 2011), and halving search a fixed 864-point grid over seven LoRA and optimizer axes (Appendix F) with the starting cross-entropy `train.py`. The code-editing condition (Claude Opus 4.8, unattended, logged as `autoresearch`) may set the same grid values and edit five designated functions: loss, head, sampler, training-data transform, and score discretization. The minimum improvement margin is 0.02; its empirical basis is recorded in Appendix G. Each condition’s best screening configuration is retrained at full fidelity (10 epochs, max length 1536, all training data) and evaluated on test splits for all 40 cells. The condition logged as asynchronous successive halving (ASHA; Li et al., 2020) is labeled halving: the supplied scheduler uses synchronous successive halving (Jamieson and Talwalkar, 2016). Counts and the unauthenticated historical scheduler are described in Appendix A.

3.2.2 Study 2: AES literature baseline

The starting configuration adapts Rodriguez et al.’s BERT classifier (Rodriguez et al., 2019; Devlin et al., 2019). Implementation settings are in Appendix K.

Two conditions run Claude Sonnet 5 (Claude Code 2.1.261), each from the same baseline on its own L40S under the shared budget. They differ in edit permissions. The **code-editing** condition (GPU A) may edit `train.py` outside its protected `main()` entry point; the **grid-restricted** condition (GPU B) may only pass a configuration drawn from a 192-point grid over six optimizer axes (Appendix F), with `train.py` hash-pinned (checksum-verified each trial) and the driver refusing any other change. Each condition uses its own GPU’s baseline screening score. The minimum improvement margin is 0.036 (Appendix G). After search, accepted, highest-screening, and selected diagnostic configurations are retrained for evaluation on all 40 cells. Search-run counts and screening scores retain their historical values; the later audit does not revise them.

Two no-agent random-search conditions use the same budget: the base numeric grid and a post hoc grid built after observing the accepted configuration’s backbone, input-length, and head-rule choices. Prompt-dependent head selection uses classification for at most ten training-observed score levels and sigmoid regression otherwise. The post hoc grid restricts numeric choices; no

prospective categorical comparator was tested (Appendix F; Table 1).

3.3 Acceptance rule

The acceptance-rule ablation replaces the minimum improvement margin with any-improvement acceptance; its agent instructions otherwise match the Sonnet code-editing condition. Any-improvement acceptance requires screening QWK strictly above the incumbent score (§3). We distinguish reclassifying a recorded trajectory under a different rule from executing a search run with that rule (Table 2).

3.4 Evaluation outcomes

Selection in both studies is on QWK only. Yannakoudakis and Cummins (2015) caution against kappa because of its sensitivity to score distributions and scoring scales. We report six secondary outcomes for Study 2 (Table 9) and all except standardized mean difference for Study 1 (Table 8). The agreement and distribution conventions follow the operational-scoring framework of Williamson et al. (2012). Exact and adjacent (± 1) agreement measure the share of essays scored exactly right or within one point (0–1). Mean absolute error (MAE) is in raw score points. Standardized mean difference (SMD) is machine minus human mean divided by the square root of their mean variance (0 at parity). The predicted-to-human score standard deviation (SD) ratio is 1 at parity. Exact agreement on extreme-score essays uses each prompt/fold’s observed minimum and maximum human scores, weighted by essay count across those endpoints (0–1).

Fold standard errors describe variation conditional on a configuration; fold dependence complicates variance estimation (Bengio and Grandvalet, 2004). Shared data and overlapping training folds preclude treating folds as independent search replicates. We report point differences, not significance or equivalence tests. Study 1’s QWK SE propagates per-prompt population-SD estimates; Study 2’s uses sample SD of five fold means divided by $\sqrt{5}$ (Appendix G).

3.4.1 Evaluation audit

For Study 2 we compute QWK from every prediction row on the contiguous integer score scale; the other six outcomes also include all rows. The audit compares these scores with the historical logs. `study2_metrics.py` reproduces

Study	Condition / evaluated configuration	Test QWK	SE
1: untuned Ettin/LoRA	code-editing (LLM), highest screening	0.771	0.0046
	random	0.752	0.0041
	halving	0.739	0.0073
	grid	0.715	0.0073
	TPE	0.654	0.0301
2: adapted BERT	BERT-base comparator	0.753	0.0041
	code-editing: str008 (accepted)	0.781	0.0034
	grid-restricted: num020 (highest-screening)	0.749	0.0059
	random, base grid	0.751	0.0034
	random, post hoc grid	0.776	0.0049
Derived contrast		QWK difference / share	SE
Study 1: code-editing minus random test QWK		+0.019	≈ 0.006
Study 2: post hoc random gain over BERT		+0.023	0.0059
Study 2: str008 residual over post hoc random		+0.005	0.0042
Study 2: post hoc share of str008 gain		82%	–

Table 1: Search space comparison (40 prompt-fold cells). SE: logged propagation (Study 1), sample fold SE (Study 2). Contrasts: quadrature ignoring covariance (Study 1), paired folds (Study 2; Appendix G). Share: unrounded BERT-relative gain ratio; dash: not applicable.

corrected means and contrasts without overwriting those logs (Appendix G; audit findings in §4.3.2). An outer holdout or nested search would be needed to assess the search procedure independently (Cawley and Talbot, 2010).

4 Results

4.1 Editable search space

4.1.1 Study 1: configuration not tuned to ASAP-AES

The code-editing condition had the highest screening and full-fidelity test QWK; random search was the strongest tested HPO comparator (Table 1; Appendix A). TPE’s screening rank reversed at full fidelity, with prompt-level declines (Table 13). The grid condition covered only part of its specified space; each condition is one search run.

The code-editing advantage over random search concentrates on prompts 7 and 8; the other six prompts’ mean contrast is near zero (Table 13). Study 1 predictions are unavailable to check the extreme-score omission found on these prompts in Study 2 (§4.3.2). The recorded configuration replaced cross-entropy with squared Earth Mover’s Distance loss and added a square-root inverse-frequency sampler (Table 15). The accepted trials changed the loss or sampler (Appendix I). The evaluated configuration, trial 005, doubled the last accepted configuration’s learning rate. It had the highest valid screening QWK but failed the acceptance threshold. The trial log associates the gains with these changes but does not isolate the loss or sampler’s contribution.

4.1.2 Study 2: AES literature baseline

The code-editing condition accepted a configuration beyond the base grid, while the grid-restricted condition accepted none (Tables 2 and 4). The accepted configuration combines ModernBERT-base (Warner et al., 2025), longer input windows, and prompt-dependent head selection (Table 16). Relative to BERT, the accepted configuration’s largest prompt-level test-QWK gain is on prompt 8; test QWK also rises on prompts 1 and 7, which were never screened (Table 14). The rejected diagnostic str001 also exceeds the BERT baseline (Appendix A).

Base-grid random search and the grid-restricted LLM both score below BERT; close point estimates do not establish equivalence. Post hoc random search recovers most of str008’s BERT-relative gain (Table 1). The comparison is retrospective, with different evaluated epoch counts and an unverified executed head threshold (Appendix K). The residual test-QWK difference is smaller than str008’s range across nominal-seed retrainings; these retrainings do not replicate search (Appendix B). Evaluation did not match configurations to GPU; the baseline offset remains uncontrolled (§4.3.2). Related post hoc diagnostics tune intermediate programs’ constants (Pelleriti et al., 2026); neither that comparison nor ours establishes that iterative discovery was necessary.

The additional code-editing search run accepted no trial, although its best trial exceeded BERT in evaluation (Appendix B). On the common ModernBERT backbone, component con-

Acceptance record	Minimum improvement margin	Any improvement
Trials / accepted	45 / 1	51 / 5
Baseline screening QWK	0.724	0.741
Last accepted screening QWK	0.766	0.772
Best screening QWK	0.769	0.772
Last accepted configuration	str008	k042
Last accepted test QWK (corrected)	0.781	0.776
Sample fold SE	0.0034	0.0028
Test QWK gain over BERT	+0.028	+0.023

Table 2: Acceptance-rule comparison. QWK: quadratic-weighted kappa. Screening scores are historical records. Corrected test outcomes use 40 prompt-fold cells per last accepted configuration; SE describes fold variation conditional on that configuration (Appendix G).

trasts favored head changes over input length (Appendix D). These are conditional contrasts; they do not isolate the BERT-relative gain.

4.2 Acceptance rule

In Study 2’s Sonnet code-editing run, the margin rule accepted an early trial but rejected later improvements (Appendix G). Applying any-improvement acceptance to that recorded trajectory would accept str012 and str044; it does not reconstruct proposals under a different acceptance history.

The separately executed acceptance-rule ablation accepted more trials than the original Sonnet margin-rule run, but none of its recorded scores clears its initial baseline plus the original margin (Table 2). Only its last accepted configuration was evaluated. Its recovered source combines ModernBERT-base, longer input windows, and scalar regression with pairwise ranking loss (Table 16). Ranking has precedent in text assessment (Yannakoudakis et al., 2011); the exact historical execution remains unverified. Its final trials again adjust numeric constants. The ablation holds the clock and remaining agent instructions fixed but does not isolate the margin’s effect on later proposals or search quality. The other margin-rule runs that accepted trials are reported in Appendix I (Study 1) and Appendix C (Study 2 Opus).

4.3 Evaluation outcomes

4.3.1 Secondary scoring outcomes

The Study 1 LLM configuration has higher adjacent agreement and lower mean absolute error than random, halving, grid, and TPE, but lower exact agreement than grid and lower extreme-score agreement than random (Table 8). These comparisons do not isolate the cause of the tradeoff.

Relative to the BERT baseline in Study 2, the accepted code-editing configuration has higher adjacent and extreme-score agreement and lower mean absolute error (Table 9). Its SD ratio moves toward parity while SMD moves away (§3.4); exact agreement declines. The acceptance-rule ablation’s k042 also loses exact agreement relative to BERT despite improving QWK (Tables 9 and 2). Appendix E retains the reported exact-agreement contrasts and uncertainty; these estimates exclude search-level variation and do not establish equivalence.

4.3.2 Evaluation audit

Elapsed times in Figure 1A include search overhead and are measured from the recorded agent launch for the Sonnet conditions and from the cumulative sweep timer for random search. The original agent budget-clock origins are unavailable, so the trajectories do not support precise equal-budget timing comparisons.

The historical evaluators omit one extreme-score essay in each affected prompt/fold cell by restricting labels to the training-observed range (Table 11). The recovered so042 source uses test-fold essays as unlabeled pretraining text on three folds; its transductive diagnostic is reported separately and excluded from cross-condition rankings (Appendix K). Corrected Study 2 scores include every prediction row, but validation predictions are unavailable: the correction has not been shown to preserve epoch selection, trial ranking, or acceptance decisions. Study 1 predictions are unavailable for this check, so its QWK remains log-derived.

The logged and corrected BERT means differ from the published comparison (Tables 11 and 1); the records do not establish identical implementation. The between-GPU screening off-

set is within the baseline seed spread but its cause is not isolated. Against the higher grid-restricted baseline, the accepted code-editing trial would miss the margin (Table 2; Appendix G). Corrected QWK matches an independent all-essay calculation on the contiguous integer score scale for every manifest-listed cell (Table 11). Historical-versus-corrected discrepancies exceed four-decimal rounding.

Historical split assignments, information access and executed-code identities remain unauthenticated (Appendix K). The recovered starting script creates the model before seeding; nominal seed pairing therefore does not establish identical initialization.

5 Discussion

Search-space specification. The selected loss/sampler and backbone/input/head changes lie outside the tested base grids. Comparisons should state which choices each condition permits. The post hoc comparison does not establish that iterative discovery was necessary (Table 1). A prospective literature-informed categorical space or an initial proposal followed by HPO could be competitive; neither was tested. Different sampled settings and training variation also prevent assigning the remaining gaps to a particular mechanism.

Acceptance and evaluation. The accepted trial records the outcome of the acceptance rule; the highest valid screening-QWK trial records the best observed score. The screening rank reversal and evaluated rejected configurations motivate reporting both trial identities and scores (Table 1; Appendix A). Applying another rule to a recorded trajectory cannot predict the changed search history; the executed ablation does not estimate policy quality across search runs (Table 2). Access restrictions, complete-essay scoring, and retained predictions support an audit, but rotating test folds still leave selection dependence unresolved.

Scoring quality. QWK improvement does not guarantee improvement on every scoring outcome (Tables 8 and 9). Independent evaluation must address the outcomes required for operational use.

6 Limitations

Study 1’s loss/sampler attribution rests on its trial log, without a matched control. The stud-

ies use different agent models, and Study 2 has incomplete initialization control and an unexplained between-GPU baseline offset (§4.3.2). The extra search run repeats configuration selection, whereas nominal-seed retraining holds the selected configuration fixed; retraining does not replicate search (Bouthillier et al., 2021). The original Sonnet any-improvement run evaluates only its last accepted configuration.

Screening uses best-epoch validation scores and is optimistic by construction. Study 1 evaluates only each condition’s best screening configuration, leaving the stability of its screening-to-evaluation ranking unknown. The post hoc grid, dependent test splits, scoring corrections, and missing historical sources limit claims about search superiority; they do not invalidate the reported conditional configuration comparisons.

The short-answer companion task, ASAP-SAS, was dropped after our reproduction attempt failed to match the published BERT score (Ormerod and Kwako, 2024).

7 Conclusion

In both primary comparisons, code-editing produced the configuration with the highest test-QWK point estimate. Random search over Study 2’s post hoc grid recovered most of the gain over BERT (Table 1). Requiring a minimum improvement margin left accepted trials below the highest valid screening QWK in every run that accepted a trial under this rule. We recommend preserving the alternatives, acceptance decisions, and evaluation population behind each selected configuration. The primary comparisons use one search run per condition and reuse selection essays in other folds’ test splits on screened prompts. Establishing search superiority requires repeated prospective searches against a literature-informed space and independent evaluation data.

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, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. [Algorithms for hyper-parameter optimization](#). In *Advances in Neural Information Processing Systems*, volume 24.

- James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305.
- Xavier Bouthillier, Pierre Delaunay, Mirko Bronzi, Asya Trofimov, Brennan Nichyporuk, Justin Szeto, Nazanin Mohammadi Sepahvand, Edward Raff, Kanika Madan, Vikram Voleti, Samira Ebrahimi Kahou, Vincent Michalski, Dmitriy Serdyuk, Tal Arbel, Chris Pal, Gael Varoquaux, and Pascal Vincent. 2021. [Accounting for variance in machine learning benchmarks](#). In *Proceedings of Machine Learning and Systems*, volume 3, pages 747–769.
- 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.
- Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, Lillian Weng, and Aleksander Mądry. 2025. MLE-bench: Evaluating machine learning agents on machine learning engineering. In *International Conference on Learning Representations (ICLR)*. ArXiv:2410.07095.
- Jordi de la Torre, Domenec Puig, and Aida Valls. 2018. [Weighted kappa loss function for multi-class classification of ordinal data in deep learning](#). *Pattern Recognition Letters*, 105:144–154.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of deep bidirectional transformers for language understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.
- Fabio Ferreira, Lucca Wobbe, Arjun Krishnakumar, Frank Hutter, and Arber Zela. 2026. [Can LLMs beat classical hyperparameter optimization algorithms? a study on autoresearch](#). *Preprint*, arXiv:2603.24647.
- Le Hou, Chen-Ping Yu, and Dimitris Samaras. 2017. [Squared earth mover’s distance-based loss for training deep neural networks](#). *Preprint*, arXiv:1611.05916.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. [LoRA: Low-rank adaptation of large language models](#). In *International Conference on Learning Representations*.
- Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. 2024. MAgentBench: Evaluating language agents on machine learning experimentation. In *International Conference on Machine Learning (ICML)*. ArXiv:2310.03302.
- Kevin Jamieson and Ameet Talwalkar. 2016. [Non-stochastic best arm identification and hyperparameter optimization](#). In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 240–248. PMLR.
- Zhengyao Jiang, Dominik Schmidt, Dhruv Srikanth, Dixing Xu, Ian Kaplan, Deniss Jacenko, and Yuxiang Wu. 2025. [AIDE: AI-driven exploration in the space of code](#). *Preprint*, arXiv:2502.13138.
- Liam Li, Kevin Jamieson, Afshin Rostamizadeh, Ekaterina Gonina, Jonathan Ben-Tzur, Moritz Hardt, Benjamin Recht, and Ameet Talwalkar. 2020. [A system for massively parallel hyperparameter tuning](#). In *Proceedings of Machine Learning and Systems*, volume 2.
- Xiaozhe Li, Jixuan Chen, Xinyu Fang, Shengyuan Ding, Haodong Duan, Qingwen Liu, and Kai Chen. 2026. [OPT-BENCH: Evaluating the iterative self-optimization of LLM agents in large-scale search spaces](#). In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 28437–28452. Association for Computational Linguistics.
- Yitao Li. 2026. [Greedy is a strong default: Agents as iterative optimizers](#). *Preprint*, arXiv:2603.27415.
- Siyi Liu, Chen Gao, and Yong Li. 2025. [AgentHPO: Large language model agent for hyper-parameter optimization](#). In *Conference on Parsimony and Learning*, volume 280 of *Proceedings of Machine Learning Research*, pages 1146–1169. PMLR.
- Chris M. Ormerod and Alexander Kwako. 2024. [Automated text scoring in the age of generative AI for the GPU-poor](#). *Chinese/English Journal of Educational Measurement and Evaluation*, 5(3). Article 5.
- Christopher Ormerod and Gitit Kehat. 2025. [Long context automated essay scoring with language models](#). In *Proceedings of the Artificial Intelligence in Measurement and Education Conference (AIME-Con): Full Papers*, pages 35–42. National Council on Measurement in Education (NCME).
- Nico Pelleriti, Sree Harsha Nelaturu, Zhanke Zhou, Zongze Li, Max Zimmer, Bo Han, and Sebastian Pokutta. 2026. [What do evolutionary coding agents evolve?](#) *Preprint*, arXiv:2605.20086.
- Pedro Uriá Rodríguez, Amir Jafari, and Christopher M. Ormerod. 2019. [Language models and automated essay scoring](#). *Preprint*, arXiv:1909.09482.
- Kaveh Taghipour and Hwee Tou Ng. 2016. [A neural approach to automated essay scoring](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1882–1891.
- Edan Toledo, Karen Hambardzumyan, Martin Josifoski, Rishi Hazra, Nicolas Baldwin, Alexis Audran-Reiss, Michael Kuchnik, Despoina Magka, Minqi

- Jiang, Alisia Maria Lupidi, Andrei Lupu, Roberta Raileanu, Tatiana Shavrina, Kelvin Niu, Jean-Christophe Gagnon-Audet, Michael Shvartsman, Shagun Sodhani, Alexander H. Miller, Abhishek Charnalia, and 6 others. 2025. [AI research agents for machine learning: Search, exploration, and generalization in MLE-bench](#). In *Advances in Neural Information Processing Systems*, volume 38.
- Niki van Stein, Diederick Vermetten, and Thomas Bäck. 2025. [In-the-loop hyper-parameter optimization for LLM-based automated design of heuristics](#). *ACM Transactions on Evolutionary Learning and Optimization*.
- Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. 2025. [Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2526–2547. Association for Computational Linguistics.
- Orion Weller, Kathryn Ricci, Marc Marone, Antoine Chaffin, Dawn Lawrie, and Benjamin Van Durme. 2026. [Seq vs seq: An open suite of paired encoders and decoders](#). In *International Conference on Learning Representations*.
- 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.
- Jie JW Wu, Ayanda Patrick Herlihy, Ahmad Saleem Mirza, Ali Afoud, and Fatemeh Fard. 2026. [Growth-Hacker: Automated off-policy evaluation optimization using code-modifying LLM agents](#). *ACM Transactions on Software Engineering and Methodology*. Accepted for publication.
- Jiayi Xie, Kaiwei Cai, Li Kong, Junsheng Zhou, and Weiguang Qu. 2022. [Automated essay scoring via pairwise contrastive regression](#). In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2724–2733.
- Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. 2024. Large language models as optimizers. In *International Conference on Learning Representations (ICLR)*. ArXiv:2309.03409.
- Ruosong Yang, Jiannong Cao, Zhiyuan Wen, Youzheng Wu, and Xiaodong He. 2020. [Enhancing automated essay scoring performance via fine-tuning pre-trained language models with combination of regression and ranking](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1560–1569.
- Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. [A new dataset and method for automatically grading ESOL texts](#). In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189. Association for Computational Linguistics.
- Helen Yannakoudakis and Ronan Cummins. 2015. [Evaluating the performance of automated text scoring systems](#). In *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 213–223.

Appendix

A Screening and selection records

The supplied halving scheduler uses 27 one-epoch starts and nine four-epoch promotions (36 evaluations, not configurations). The historical scheduler remains unauthenticated.

Table 3 records Study 1 screening; Table 4 records Study 2 comparators. The original code-editing condition’s screening record is in Table 2. The grid-restricted condition’s later trials used cheaper ten-epoch configurations.

Condition	Trials	Screen
code-editing (LLM)	25	0.729
random	27	0.681
halving	36	0.678
grid	28	0.534
TPE	27	0.692
cross-entropy start	–	0.500

Table 3: Study 1 screening record. Screen: best reduced-fidelity screening QWK; bold marks the highest. Trials include the LLM baseline and halving rung evaluations; TPE: tree-structured Parzen estimator.

Condition / configuration	Trials	Accepted	Baseline	Best	Test	SE
grid-restricted: num020 (best)	93	0	0.738	0.744	–	–
random, base grid	110	–	–	0.745	–	–
random, post hoc grid	73	–	–	0.770	–	–
Code-editing: str044 (rejected, best)	–	–	–	–	0.780	0.0030
Code-editing: str001 (rejected)	–	–	–	–	0.770	0.0030

Table 4: Study 2 screening and diagnostic evaluations. Baseline/Best: historical screening QWK. Test: corrected QWK; SE: sample fold standard error. Dashes indicate values reported elsewhere or not applicable.

The rejected diagnostic configurations str044 and str001 come from the same original code-editing search run, not independent search runs. Each test-split evaluation covers 40 prompt-fold cells. Primary test comparators are in Table 1.

B Search-run replication and retraining seeds

One additional code-editing search run uses the same instructions and starting configuration. Separately, the original accepted configuration and BERT baseline are retrained at two additional nominal seeds. The first repeats search; the second holds configuration selection fixed (Table 5).

The additional code-editing search run accepted no trial, so the original acceptance contrast was not repeated. Its best trial averages a smoothed-QWK classifier and a regression model; it narrowly missed the margin but exceeded BERT in evaluation (Table 5). Two search runs do not estimate search reliability. Separately, the original accepted configuration’s evaluation gain remains positive under both nominal-seed reruns. Nominally paired seeds do not establish identical initialization (§4.3.2).

Replication diagnostic	QWK / count	SE
Search-run replicate: trials / accepted	50 / 0	–
baseline screening QWK	0.734	–
best screening QWK (r022)	0.770	–
shortfall from threshold	0.0003	–
r022 test QWK	0.771	0.0029
r022 gain over BERT	+0.019	–
BERT-base, nominal seed 7	0.755	0.0057
BERT-base, nominal seed 123	0.734	0.0149
str008, nominal seed 7	0.778	0.0035
str008, nominal seed 123	0.773	0.0024
Mean paired gain, nominal seeds 42/7/123	+0.030	–

Table 5: Search-run replication (top) and fixed-configuration retraining (bottom). Test QWK uses 40 cells per configuration; SE is sample fold standard error. Original screening records: Table 2; nominal-seed-42 test QWK: Table 1.

C Original Opus margin-rule record

Claude Opus 5 runs the code-editing instructions under the same clock and margin as Sonnet 5.

The accepted configuration is so006; the highest-screening trial is so042. Later trials exceeded the incumbent score without clearing the acceptance threshold. Recovered so042 source reads test-fold essays as unlabeled pretraining text (Appendix K); its diagnostic is reported separately and excluded from cross-condition rankings.

Opus code-editing condition	Recorded value
Trials / accepted	43 / 1
Baseline screening QWK	0.736
Last accepted / best screening QWK	0.785 / 0.794
Accepted so006 test QWK (SE)	0.788 (0.0024)
Later trials above incumbent	12
so042 transductive diagnostic QWK (SE), corrected	0.793 (0.0025)

Table 6: Original Opus margin-rule record. Test QWK and sample fold SE use 40 cells per configuration; so042 was rejected by the margin. The recovered so042 source uses test-fold text in pretraining (Appendix K).

D Input and head diagnostics

Two fixed-configuration ablations of the accepted Study 2 configuration retain only its longer input windows or its prompt-dependent head selection. The input-only configuration uses ModernBERT-base, 1600-token windows, and `CLASSIFY_MAX_CLASSES` 100 (classification on every prompt). The head-only configuration uses ModernBERT-base, 512-token windows, and `CLASSIFY_MAX_CLASSES` 10: classification for at most ten training-observed score levels and sigmoid regression otherwise. ModernBERT-512 is the common classifier comparator. Paired fold contrasts compare each component at a fixed setting of the other (Table 7); they do not allocate the BERT-relative gain into additive shares.

Head-related contrasts are larger than input-length contrasts on the common ModernBERT backbone (Table 7). The negative interaction means that the combined improvement is smaller than the sum of the two separate improvements. ModernBERT-512 itself scores below BERT. The per-prompt evidence is in Table 14.

ModernBERT configuration / contrast	QWK	SE
512-token windows, classification	0.739	0.0041
Longer windows only	0.760	0.0042
Prompt-dependent head only	0.772	0.0021
Head effect, 512-token windows	+0.032	0.005
Head effect, 1600-token windows	+0.020	0.003
Input effect, classification	+0.021	0.006
Input effect, prompt-dependent head	+0.009	0.003
Combined effect over ModernBERT-512	+0.041	–
Interaction	–0.012	0.007

Table 7: Input/head configurations and conditional QWK contrasts on ModernBERT. Combined configuration: str008 in Table 1. SE: sample fold standard error, paired by fold for contrasts.

E Secondary scoring outcomes

Condition	Exact	Adj.	MAE	SD ratio	Extr.
code-editing	0.525 (0.0064)	0.826 (0.0018)	0.94 (0.0098)	0.951 (0.0141)	0.437 (0.0295)
random	0.532 (0.0027)	0.817 (0.0014)	1.02 (0.0117)	0.926 (0.0086)	0.457 (0.0150)
halving	0.528 (0.0043)	0.815 (0.0040)	1.02 (0.0130)	0.905 (0.0109)	0.399 (0.0126)
grid	0.544 (0.0037)	0.823 (0.0018)	1.02 (0.0173)	0.872 (0.0037)	0.403 (0.0112)
TPE	0.515 (0.0074)	0.795 (0.0053)	1.12 (0.0561)	0.813 (0.0351)	0.366 (0.0280)

Table 8: Study 1 secondary outcomes: means over 40 cells, with sample fold SE in parentheses. Adj.: adjacent agreement; MAE: mean absolute error in score points; SD: standard deviation; Extr.: extreme-score agreement. Definitions: §3.4; QWK: Table 1.

Configuration	Exact	Adj.	MAE	SMD	SD ratio	Extr.
BERT-base (baseline)	0.529	0.830	0.948	+0.013	0.921	0.388
ModernBERT-512 (baseline)	0.521	0.814	1.037	+0.024	0.938	0.448
str008 (accepted)	0.527	0.836	0.868	+0.033	0.933	0.409
str044 (best)	0.525	0.836	0.876	+0.031	0.945	0.433
str001 (rejected)	0.520	0.836	0.889	+0.017	0.918	0.374
num020 (grid-restricted)	0.531	0.825	0.952	+0.054	0.907	0.410
random, base grid	0.529	0.825	0.964	+0.037	0.915	0.400
random, post hoc grid	0.524	0.832	0.881	+0.021	0.936	0.393
r022 (replicate, best trial)	0.511	0.832	0.917	+0.061	0.960	0.389
k042 (any-improvement)	0.508	0.827	0.927	–0.011	0.977	0.437
Fold SE range	0.002–0.009	0.001–0.005	0.005–0.028	–	0.004–0.019	0.005–0.024

Table 9: Legacy Study 2 secondary outcomes: means over 40 prompt-fold cells, nominal seed 42. Adj.: adjacent agreement; MAE: mean absolute error in score points; SMD: standardized mean difference; SD: standard deviation; Extr.: extreme-score agreement (§3.4). SE: sample fold standard error; dash: unreported. SE ranges retain the original full cohort, including omitted configurations. Exact-agreement contrast against BERT: str008 –0.002 (paired SE unreported); k042 –0.021 (paired fold SE 0.005).

F Hyperparameter grids

Per-axis values for the fixed grids of §3.2.1 and §3.2.2.

Study 1, 864 points ($4 \times 3 \times 3 \times 3 \times 2 \times 2 \times 2$): learning rate $\{5 \times 10^{-5}, 1 \times 10^{-4}, 2 \times 10^{-4}, 3 \times 10^{-4}\}$; LoRA rank $\{8, 16, 32\}$; LoRA alpha $\{8, 16, 32\}$; dropout $\{0.0, 0.1, 0.2\}$; warmup steps $\{0, 100\}$; scheduler $\{\text{linear}, \text{cosine}\}$; weight decay $\{0.0, 0.01\}$.

Study 2, 192 points ($4 \times 3 \times 2 \times 2 \times 2 \times 2 \times 2$): learning rate $\{1 \times 10^{-5}, 2 \times 10^{-5}, 3 \times 10^{-5}, 5 \times 10^{-5}\}$; batch size $\{9, 16, 32\}$; epochs $\{10, 20\}$; warmup ratio $\{0.0, 0.1\}$; weight decay $\{0.0, 0.01\}$; label smoothing $\{0.0, 0.1\}$.

Study 2 post hoc grid, 192 points ($2 \times 2 \times 3 \times 2 \times 2 \times 2 \times 2$): backbone {BERT-base, ModernBERT-base}; max length {512, 1600}; head rule {regression everywhere, classification up to 10 score levels, classification everywhere}; learning rate $\{2 \times 10^{-5}, 5 \times 10^{-5}\}$; epochs {10, 20}; weight decay {0.0, 0.01}; label smoothing {0.0, 0.1}. The backbone, input-length, and head-rule choices come from the accepted configuration and were added to the grid after the code-editing condition finished; batch size and warmup ratio are pinned at the starting configuration’s values. Screening counts are in Table 4; the logs omit the head threshold, so distinct full configurations cannot be counted from them. This post hoc revised grid is not a superset of the original numeric grid.

G Minimum improvement margin, acceptance thresholds and standard errors

Table 10 records the uncertainty estimates used to motivate the margins and the historical screening values used to inspect rejected improvements. Cell-based SEs pool heterogeneous prompt-fold cells, not independent search runs; Table 1 retains the distinct logged propagated SE.

Minimum improvement margins. Study 1’s 0.02 rule used a baseline estimate from five training-seed reruns of the same configuration. Study 2 used five random seeds of the unedited baseline at the code-editing-condition setting; its screening spread was dominated by prompt 8. The required improvement is $2\sigma\sqrt{2} = 0.036$. The cross-GPU baseline offset (Table 11) is consistent with, but does not validate, that width.

Incumbent scores and rejected improvements. Table 10 records the renewed thresholds after str008 and so006 and the later above-incumbent trials. In the separately executed acceptance-rule condition, k007 falls below the threshold the original margin would have imposed.

Record	Configuration / statistic	Value
Study 1 cell-based SE	code-editing / random	0.0104 / 0.0122
	halving / grid / TPE	0.0133 / 0.0222 / 0.0379
Study 1 sample-SD propagation	code-editing / random	0.0052 / 0.0046
Study 1 baseline seed estimate	σ	≈ 0.0073
Study 2 baseline screening QWK	range / mean	0.714–0.740 / 0.729
Baseline screening seed SD	overall / prompt 8	0.0128 / 0.031
Code-editing incumbent	str008	0.7656
Renewed acceptance threshold	after str008	0.8016
Later above-incumbent trials	count	4
	str012 / str034	0.7663 / 0.7658
	str035 / str044	0.7659 / 0.7689
Opus incumbent / threshold	after so006	0.785 / 0.821
Any-improvement trial / counterfactual threshold	k007	0.769 / 0.777
str008 screening QWK	seeds 42 / 7	0.766 / 0.750
	seeds 123 / 2026	0.755 / 0.753
	seed mean	0.756

Table 10: Historical margin diagnostics. Screening values are QWK; SD: standard deviation; SE: standard error. Estimator definitions follow.

Standard errors. Study 1 rows in Table 1 retain the logged convention: per-prompt SE = population SD of 5 fold QWKs/ $\sqrt{5}$; 8-prompt SE = root sum of squares over prompts/8. Study 2 tables: sample SD of the 5 fold means/ $\sqrt{5}$. The cell-based SEs in Table 10 use sample SD over 40 cells/ $\sqrt{40}$. Paired Study 2 contrasts use sample SD of five fold-wise differences/ $\sqrt{5}$. These describe the observed configurations; fold dependence and variation across search runs are not modeled. Table 10 also gives the Study 1 LLM and random estimates under sample-SD propagation rather than the logged convention.

All-essay QWK correction (Study 2). For integer truth scores y and predictions $\hat{y}$, we use the complete contiguous score scale, equivalently

$$1 - \frac{\overline{(y - \hat{y})^2}}{\text{Var}(y) + \text{Var}(\hat{y}) + (\bar{y} - \bar{\hat{y}})^2},$$

435

with population moments over every essay in the prediction file. Historical records are retained unchanged. Audit counts and the independent QWK cross-check are in Table 11 and §4.3.2. Corrected scoring evaluates the already selected predictions; it does not redo historical validation or search.

The recovered legacy analyses use the QWK and fold SE in `study2_metrics.py`.

Audit record	Value
Legacy evaluation entries / valid evaluation cells	20 / 800
Cells with predictions / discrepant QWK	800 / 60
Original audit cells / discrepant QWK	680 / 51
Omitted extreme essay: prompt / fold / score	7 / 0 / 2
	8 / 1 / 10
	8 / 4 / 60
Published BERT comparison QWK	0.7475
Historical logged BERT mean	0.752
Between-GPU baseline screening offset	0.014

Table 11: Legacy evaluation records and all-essay audit, including recovered evaluations. Published BERT comparison: [Rodriguez et al. \(2019\)](#). Corrected BERT mean: Table 1; corrected so042 outcome: Table 6.

H Study 2 agent instructions (`program.md`), verbatim

Karpathy’s loop has three files: `prepare.py` (protected data and evaluation code), `train.py` (the one file the agent edits), and `program.md` (agent instructions). The loop minimizes validation bits per byte, with a five-minute training limit per trial; any improvement advances a git branch.

We reproduce the operating prompt read by the code-editing condition in Study 2 (§3.2.2) verbatim except for re-wrapping long lines, transliterating Unicode symbols (arrows, $\leq$, $\neq$, σ) to ASCII for the monospace font, and renaming the experimental-branch term and its command-line flag to *condition* and `-condition`, for consistency with this paper’s terminology. These are editorial changes, not the original executable interface. The excerpt retains the historical labels “structural”, “numerical”, “keep”, and “bar”; the paper calls these code-editing, grid-restricted, accepted, and incumbent score, respectively. Edit permissions and margins are specified in §3.2.1 and §3.2.2. Study 1’s supplied instruction variants also differ in paths, training instructions and confirmation-rerun permissions; exact historical variants remain unauthenticated.

```
# autoresearch loop -- ASAP-AES from a strong baseline (LLM
condition, structural)

Goal: get the highest `pooled_val_qwk` (mean val-split QWK over
ASAP-AES sets 2,3,5,8, computed by frozen `prepare.py`) within 12
h wallclock.

## Setup (once)
- You are in `/mnt/data/jufrost/loop_bench_t2/structural/` (its own
git repo). `python` and `trackio` on PATH are the orchestrator
venv's; the GPU number comes from the launch prompt. Trial ids are
`str000`, `str001`, ... Do not spawn subagents; work in this
session only.
- Read: `train.py` (you edit this -- the whole file except
`main()`), `prepare.py` (frozen: data, splits, seed, metric),
`run.py` (frozen driver; `SPACE` lists the numeric knobs the
classical condition searches).
- `python run.py --condition autoresearch --start` (stamp the 12 h
clock).
- Baseline = trial `str000`: `python run.py --condition autoresearch
```

```

--gpu <GPU> --trial-id str000` with `train.py` unedited; its
`pooled_val_qwk` is the starting bar. It is the reproduced
Rodriguez-Jafari-Ormerod 2019 BERT-base recipe and matches the
published bar (8-set mean ~0.75) -- you are improving the
literature baseline, not a stock model. The backbone (`MODEL`) is
yours to change.
- `git branch autoresearch/keep-decisions` at this tree.

## Experimentation
Run one trial: `python run.py --condition
  autoresearch --gpu <GPU> --trial-id str<NNN> [--config '<json>']`
  (~15 min: 4 sets x 20 epochs; a cheaper recipe runs faster).

| Rule | Value |
|---|---|
| CAN edit | anything in `train.py` above `main()`: model/head,
  loss, optimizer, schedule, batch size, epochs, windowing/decoding,
  data handling, regularization. `--config` may also set the
  constants listed in `HP_NAMES`. |
| CANNOT edit | `train.py`'s `main()` (result assembly),
  `prepare.py`, `run.py`, the seed, the splits, the metric,
  dependencies (no new packages). The test split is unreadable and
  off-limits. |
| Metric | `pooled_val_qwk` (val split, best-epoch selection on
  val), higher better |
| Keep margin |  $\pm 0.036$  (measured: sigma of the unedited
  baseline over 5 seeds at this exact setting) -- a keep must beat
  the bar by MORE than this; within it = no change = discard. No
  reruns: the seed is fixed, a rerun is a replicate. |
| Budget | 12 h wallclock incl. your thinking time; hard stop at
  clock expiry. A cell over `prepare.CELL_HARD_CAP_S` exits 124. |
| Tie-break | simpler wins; a win from deleting code is best |

## Starting point
The baseline is the published recipe (CLS
  classification over the set's discrete scores, cross-entropy,
  AdamW 5e-5, batch 9, 20 epochs, 510-token overlapping windows,
  best-on-val epoch). There are no ASAP-specific priors beyond
  that -- discover what moves QWK on this data from scratch. QWK is
  quadratic-weighted: far misses cost more than near ones.

## Outcome -- read by EXIT CODE, then Trackio (never parse stdout)
| Exit | Meaning | Action |
|---|---|---|
| 0 | trial done | `trackio get run --project loop_bench_t2 --run
  autoresearch_str<NNN> --json` -> `pooled_val_qwk`, `valid`,
  per-set `qwk_s<k>`; cross-run compare with `trackio list runs
  --project loop_bench_t2`. `valid=0` (no metric key) -> invalid,
  discard. |
| 124 | clock expired OR a cell hit the hard cap | run `python
  run.py --condition autoresearch --check`: exit 1 -> clock done,
  STOP; exit 0 -> the trial was too slow, discard |
| 2 | no clock | you skipped `--start`; run it once |

```

```
| other | crash | read `runs/autoresearch_str<NNN>/train.log`; fix
if trivial and re-run the same commit <=2x, then move on |

## The loop -- RUN UNTIL THE CLOCK EXPIRES
1. `python run.py --condition autoresearch --check` -> exit 1
   (expired) -> STOP. `remaining_h` < 0.4 -> STOP (a trial would not
   finish).
2. Bar = best kept `pooled_val_qwk`. Commit or revert stray
   *tracked* changes (never `git clean`; `runs/` and
   `clock_start.json` are gitignored and must survive), then `git
   checkout -b autoresearch/experiment/str<NNN>
   autoresearch/keep-decisions`.
3. Form ONE idea (a diagnosis of the last result, a combination of
   keeps, an ablation). Prefer moves the numeric grid in `run.py`
   cannot express.
4. Edit `train.py`; `git commit` (message = the one-line why).
5. Run the trial; read the outcome by exit code (table above).
6. Admissibility first: exit != 0 -> handle per table. A keep is
   `valid` AND beats the bar by more than the margin.
7. Keep -> `git branch -f autoresearch/keep-decisions HEAD`. Discard
   -> leave the branch; the next idea branches off the bar again.
   Nothing is lost (every trial is in Trackio).

NEVER STOP before the clock. Out of ideas -> think harder: a new
loss family, a different head or pooling, a different decode of
window predictions, a sampler, a data transform, a schedule. The
human is asleep; keep finding wins over a strong baseline.
```

I Study 1 trial log

Table 12 selects baseline 000, the accepted trials (001, 002, 004), and the highest-screening trial (005) from the Study 1 LLM trial log (mean 4-prompt screening QWK). Historical trial descriptions and outcomes retain their original vocabulary, including “keep” and “transferred”. Trial 005 exceeded the last accepted trial 004 but missed the margin; it is the configuration evaluated in Table 1.

#	idea	QWK	outcome
000	baseline cross-entropy	0.500	(base)
001	soft-ordinal CE ($\tau=1$)	0.640	keep +0.14
002	squared EMD (param-free)	0.673	keep +0.033
004	EMD + sqrt-inv-freq sampler	0.715	keep +0.042
005	004 + $\text{lr} = 2 \times 10^{-4}$	0.729	sub-margin; transferred

Table 12: Selected Study 1 trial records. QWK: mean screening quadratic-weighted kappa; outcome text is historical. Notation key and authentication limits follow the table.

Historical notation key. CE: cross-entropy; EMD: Earth Mover’s Distance; lr: learning rate; τ : the recorded soft-ordinal loss parameter. The particular τ construction remains unauthenticated.

J Per-prompt test-split QWK

The historical code-editing screening rows in Table 14 show the largest gain on prompt 8.

Condition	P1	P2	P3	P4	P5	P6	P7	P8
code-editing	0.807	0.668	0.704	0.826	0.802	0.825	0.808	0.726
random	0.792	0.674	0.717	0.821	0.814	0.819	0.766	0.615
halving	0.773	0.658	0.700	0.826	0.794	0.766	0.779	0.613
grid	0.753	0.655	0.711	0.808	0.796	0.794	0.806	0.396
TPE	0.402	0.512	0.682	0.802	0.805	0.811	0.600	0.617

Table 13: Study 1 per-prompt test QWK, averaged over five folds. P1–P8: essay prompts.

Config.	P1	P2	P3	P4	P5	P6	P7	P8	P8 SE
BERT-base	0.798	0.653	0.688	0.825	0.807	0.809	0.811	0.629	0.0292
ModernBERT-512	0.782	0.672	0.689	0.804	0.819	0.807	0.790	0.553	0.0228
str008 (accepted)	0.832	0.686	0.699	0.817	0.816	0.813	0.834	0.749	0.0146
str044 (best)	0.830	0.706	0.678	0.812	0.821	0.810	0.839	0.743	0.0133
str001 (rejected)	0.810	0.683	0.685	0.807	0.810	0.806	0.836	0.723	0.0161
num020 (grid-restricted)	0.791	0.645	0.686	0.820	0.811	0.804	0.814	0.624	0.0361
random, base grid	0.794	0.674	0.677	0.817	0.804	0.794	0.816	0.630	0.0234
random, post hoc grid	0.827	0.677	0.682	0.816	0.812	0.808	0.835	0.749	0.0249
r022 (replicate)	0.814	0.662	0.697	0.817	0.807	0.813	0.832	0.729	0.0139
k042 (any-impr.)	0.828	0.700	0.665	0.813	0.807	0.816	0.828	0.748	0.0191
longer windows only	0.819	0.678	0.696	0.809	0.806	0.806	0.779	0.690	0.0193
head only	0.810	0.686	0.683	0.815	0.807	0.815	0.830	0.727	0.0197
Historical Sonnet code-editing screening QWK (screened prompts only)									
Baseline	–	0.739	0.716	–	0.817	–	–	0.622	–
str008	–	0.752	0.724	–	0.822	–	–	0.765	–

Table 14: Legacy Study 2 per-prompt test QWK (five-fold mean) and historical screening QWK; seed 42 unless stated. P1–P8: prompts; P8 SE: sample standard error over five prompt-8 folds. Any-impr.: any-improvement. Dashes indicate unreported values.

K Evaluated configurations

Historical essay-to-split assignments, search access to test data, when evaluation results informed later conditions, and complete executed-code identities remain unauthenticated. Recovered selected sources narrow the artifact gaps but do not authenticate every historical execution. The post hoc head threshold and the complete evaluator environment also remain unverified. A recovered starting script matches the code-editing launch hash but creates the model before applying its explicit training seed; nominal seed pairing therefore does not establish identical initialization.

Our implementation uses `bert-base-uncased`, a classification head on the encoder’s summary (CLS) token over each prompt’s discrete scores, cross-entropy, AdamW at 5×10^{-5} , a batch size of 9, 20 epochs, and 510-token overlapping windows, with the epoch selected by validation QWK.

Tables 15 and 16 list the configurations evaluated in Results, with logged hyperparameters and reported commit identifiers.

Condition	LR	Rank	Alpha	Drop.	Warmup	Sched.	WD
code-editing	2e-4	16	16	0.2	100	linear	0.0
grid	5e-5	8	16	0.0	0	linear	0.01
random	2e-4	16	32	0.1	0	linear	0.01
TPE	3e-4	8	32	0.0	0	cosine	0.01
halving	3e-4	8	16	0.1	100	linear	0.0

Table 15: Study 1 evaluated configurations (Table 1). LR: learning rate; Drop.: dropout; Sched.: scheduler; WD: weight decay; Rank/Alpha: LoRA settings. Warmup is in steps.

All four Study 1 HPO conditions train with cross-entropy. The code-editing condition set only its learning rate; its other LoRA values are unedited harness defaults. Its code change (commit 89c50543) is the EMD loss plus sampler of §4.1.1 (Table 15).

Config.	Backbone	Input	Head	Commit
BERT-base	BERT	512, wind.	class.	–
ModernBERT-512	ModernBERT	512, wind.	class.	–
str008 (accepted)	ModernBERT	1600, wind.	prompt-dep.	3a1493b
str044 (best)	ModernBERT	1600, wind.	prompt-dep.	8ee6745
str001 (rejected)	BERT	512, wind.	regr.	661af18
num020 (grid-restricted)	BERT	512, wind.	class.	–
random, base grid	BERT	512, wind.	class.	–
random, post hoc grid	ModernBERT	1600, wind.	unverified	–
r022 (replicate)	BERT	512, wind.	ensemble	aebdf00
k042 (any-impr.)	ModernBERT	1536, wind.	regr.+rank	e48fe43
so006 (Opus 5)	ModernBERT	1536, wind.	regr.+quant.	b54ab82
so042 (transductive)	ModernBERT	1536, wind.	aux.+quant.	unverified
longer windows only	ModernBERT	1600, wind.	class.	–
head only	ModernBERT	512, wind.	prompt-dep.	–

Table 16: Study 2 recorded configurations. Wind.: overlapping windows; class.: classification; regr.: regression; prompt-dep.: prompt-dependent head selection; rank: pairwise ranking loss; quant.: quantile score discretization; aux.: auxiliary trait heads; any-impr.: any-improvement.

All configurations use learning rate $5e-5$, 20 epochs, and batch size 9, except k042 (learning rate $3e-5$) and random, post hoc grid (10 epochs). Recovered k042, so006, and so042 sources request bfloat16 (BF16); historical executed precision is unauthenticated.

Window lengths include two special tokens. Recovered sources split overlength essays into windows, with the final window overlapping; essays that fit use one window. Heads: “class.” is the source classification head over each prompt’s discrete scores; “regr.” is sigmoid-regression with mean squared error (MSE); “prompt-dep.” uses classification for prompts with ≤ 10 training-observed score levels and regression otherwise. “regr.+rank” uses weighted scalar Smooth L1 loss ($\beta = 0.1$) with pairwise ranking loss (weight 0.5, margin 0.05) in the recovered k042 source.

The recovered so006 and so042 sources use quantile-based score discretization: they rank the evaluated essays’ continuous scores and allocate consecutive ranks to classes using rounded cumulative training-score proportions multiplied by the evaluated essay count. The allocation therefore depends on the evaluated collection of essays. Stable sorting retains the incoming order of tied scores; ties can span class boundaries. The historical ordering of essays remains unauthenticated.

The classifier/regression ensemble description applies to r022. Its recovered source converts each window’s classifier argmax to a score and clips the raw regression output to the class-index range before interpolating between training-observed score values. It averages each member’s scores over an essay’s windows, then equally averages the two per-essay values on the human-score scale before rounding to a score class. This regression conversion uses no sigmoid. The rounding helper’s tie behavior remains unverified.

str044 adds classifier dropout 0.1 and weight decay 0.02 to str008. num020 adds weight decay 0.01 and label smoothing 0.1. Random search on the base grid is trial 86 and random search on the post hoc grid is trial ID 29 (condition counts in Table 4). The post hoc random settings omit the head threshold; the earlier description as regression everywhere cannot be authenticated.

The recovered so042 source adds auxiliary trait heads and masked-language-model pretraining; it defaults to three pretraining epochs, whereas the screening commit describes six and the evaluation manifest supplies no override. Its pretraining always reads fold-0 training text across all prompts. Comparison with the prediction IDs shows that this pool contains every test essay in folds 1–3, and none in folds 0 or 4. This is unlabeled test-text exposure, not supervised use of test labels. The reported so042 aggregate therefore describes a transductive evaluation with unresolved screening-source identity; it is reported separately and excluded from cross-condition rankings.