

Alphastream at ChildSafeAds 2026: Gated Per-Label Merging and an Advertiser-Precedent Rule for Rare Compliance Classes

Raghavendra Prasad Narayan

Alphastream.ai

raghavendra@alphastream.ai

Abstract

We describe our system for ChildSafeAds 2026, the shared task on commercial content in child-facing YouTube videos: it labels each sponsored segment for commercial type (ST1), product category (ST2) and EU compliance flags (ST3). A multi-task ModernBERT-large encoder and an entailment-style reformulation of the compliance subtask supply seed-averaged probabilities, merged label by label under a paired bootstrap and a dev-to-test count check. Inference uses eight fine-tuned encoders and no generative model; the 503 test instances are estimated at under one GPU-hour, and all experiments cost about US\$17. It scored 0.7645 mean macro-F1 on development and **0.6798** on test, 0.046 of it from a rule keyed on advertiser identity for one ST1 class absent from development. Our official entry, a variant using one leaderboard probe, scored 0.7022, fourth of 21 teams. A single-seed ablation values the four access levels at +0.042, against +0.128 from encoder scale. We also report five negative results, among them seed variance matching our largest modelling gains, synthetic data helping one architecture and breaking the other, and 21% more training data lowering the test score; separately, near-duplicate training instances agree on the full flag set only 43–63% of the time, often on identical cited evidence.

1 Introduction

The ChildSafeAds shared task (Bertaglia et al., 2026) is concerned with monitoring influencer advertising that reaches children.¹ Each instance is a segment that users of the crowdsourced SponsorBlock extension marked as sponsored, from a child- or teen-facing YouTube channel, with the linked product page. Each instance is labelled

¹Code, per-run probabilities, submissions for every experiment (failures included), checkpoints and a dated decision log: https://github.com/raghavnpasad/nllp_ChildSafeAds (Appendix D).

three ways: ST1, commercial type (single-label, five classes); ST2, product category (multi-label, twelve); ST3, EU compliance flags (multi-label, eight, two of them housekeeping: `no_flag` and `insufficient_context`). The score is the plain mean of the three macro-F1s.

§3 describes the system end to end — Figure 1 and Table 2 give the components, what each was trained and selected on, and what it costs to run — and §4 reports the test results. We report **0.6798** as our primary result and the leaderboard-informed 0.7022 of our official entry, which placed fourth among 21 teams, 0.0325 behind the winner, as an additional one (a majority-class baseline scores 0.093). Two properties of the evaluation shaped the system as much as the architecture did. First, macro-F1 gives very small labels a very large weight: `hfss_food_marketing` (food high in fat, salt or sugar) has five development positives yet carries one-eighth of ST3; ST1’s other has *two* training instances, *no* development instance, and one-fifth of ST1 on test (§4). Second, 504 development rows give a wide noise floor: paired-bootstrap 95% intervals on a candidate’s ST3 gain are 0.03 to 0.16 wide, wider than nearly every improvement we measured. This is the low-power regime of Card et al. (2020), built into the task’s own split. We therefore describe the system and its development choices first (§§2–3), and report the test results, the negative results and the task-level observations as supporting analysis (§§4–6).

2 Task, data, and what each level bought

The data is split 2,353 / 504 / 503 (train / dev / test) (Bertaglia et al., 2026) and is channel-disjoint: no training channel appears in development or test. The organisers define four levels of data access: (1) the transcript; (2) plus title, description and the paid-promotion disclosure field; (3) plus channel information; (4) plus the linked product page. We

input	ST1	ST2	ST3	mean
L1 transcript	0.585	0.513	0.417	0.505
L2 + title/desc/disl.	0.597	0.596	0.370	0.521
L3 + channel	0.655	0.555	0.406	0.538
L4 + product page	0.645	0.630	0.368	0.547
L4, <i>large</i> encoder	0.812	0.755	0.458	0.675

Table 1: One configuration (ModernBERT-base, identical training) at each access level, and the same recipe at large scale. Dev macro-F1, single seed; bold marks each column’s best value. Among the four access levels ST3 peaks at L1.

declared level 4 at the outset, on the assumption that what is sold decides the product-facing labels. Our linearisation placed the transcript first and the product page last, right-truncated at 2,048 tokens. On the longest instances the product page is the first thing lost: 22% of development instances exceed the window and lose the tail of their page, a median 38% of its tokens, with the whole page gone in under 1%. A later field-budgeted layout fixed the truncation, and made our strongest single ST3 model sharply worse (§5).

Table 1 answers the task’s framing question, and not in the direction we expected. The whole data ladder (L1 to L4) is worth +0.042 on the mean, and base to large at a fixed level is worth +0.128. Only the encoder effect exceeds the seed band of Table 5. The levels help different subtasks. ST2 gains the most (+0.083 from metadata, and +0.034 more by L4 despite a dip at L3) and was our strongest test subtask. ST1 gains from channel identity (+0.058 at L3). ST3 did best from the transcript alone; extra fields mildly hurt it. Each delta sits inside the seed band, so we claim only that the extra fields did not help ST3, but the directions kept repeating. Crawls also fail silently: the test instance that mattered most to our score carried a page from an unrelated store (§4.1).

3 System

Figure 1 shows the submitted system end to end, and Table 2 states what each component was trained on and selected on. All encoders were fine-tuned on level-4 input at 2,048 tokens with AdamW (Loshchilov and Hutter, 2019); the exact hyperparameters are in Appendix B.

Multi-task encoder. A single ModernBERT-large (Warner et al., 2025) reads one linearised string per instance, with three linear heads on the [CLS] representation: cross-entropy for ST1

(class weights clipped to $[0.25, 8.0]$), binary cross-entropy for ST2 and ST3 (positive weights clipped at 12). One threshold per label is tuned on development (Lipton et al., 2014) over a deliberately coarse grid (0.05–0.90; finer grids memorise five-positive labels), then taxonomy-constrained ST3 decoding. The clip is not what cancelled our augmentation: the synthetic rows are appended before the weights are computed, so ST1’s other was weighted on 103 instances rather than two and its weight never reached the cap (§5).

Entailment reformulation for ST3. Each (instance, flag) pair is scored by a shared binary head, with the flag’s written definition prepended to the instance text, after entailment-based zero- and few-shot classification (Yin et al., 2019; Schick and Schütze, 2021). Compliance flags are legal definitions with fairly sharp boundaries; a shared head lets rare flags borrow evidence from common ones, at eight forward passes per instance. Neither formulation dominates at matched scale: at base scale entailment leads on both families (disclosure 0.740 against 0.576, product 0.415 against 0.381) but loses *hfss_food_marketing* (0.250 against 0.333), and at large scale the strongest source differs from flag to flag. Our merge therefore takes each label from its own source, under the gate described below.

Seed averaging. Six seeds of our best configuration, changing nothing else, span 0.7037–0.7369 in dev mean and 0.554–0.628 in ST3 (Appendix A), the effect reported by Reimers and Gurevych (2017) and Dodge et al. (2020), at a magnitude that dominated our results. Averaging the probability matrices beats every single seed on ST2, ST3 and the mean; on ST1 the best seed stays ahead, and picking it after the fact would repeat the selection error our gates exist to prevent. The *submitted* system averages the first three seeds (the only ones trained before our first test submission). Its dev ST1, 0.8721, is higher than every row of Table 5, partly by chance on rare classes: the six-seed average falls back to 0.8330 and failed the gate as a replacement.

Gated per-label merging. The final ST3 takes each label’s column from its best source (exact map: Appendix B), then a taxonomy repair pass. Selection and threshold tuning use the same 504 rows, so the procedure produces false improvements unless it is controlled. Three rules governed every adoption. (i) A *paired bootstrap gate* (Koehn, 2004; Berg-Kirkpatrick et al., 2012; Dror

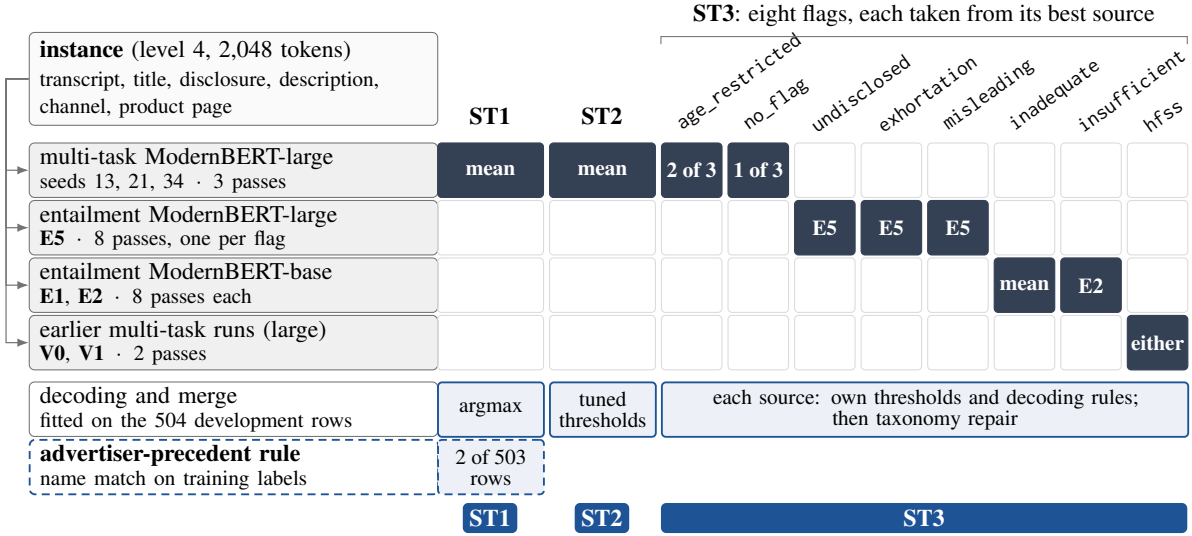


Figure 1: The submitted system as a supply matrix. Rows are components and columns are labels; a filled cell means that the row supplies that column, and its text says how outputs are combined (*mean* of probabilities; *2 of 3* and *1 of 3* are votes over the three seeds; *either* fires if V0 or V1 does). Each ST3 flag comes from its best source, after that source is decoded with its own thresholds and the rules of Appendix B, one of which reads the platform’s paid-promotion field. The dashed row edits ST1 on two test rows (one in the official entry) and is the only component informed by leaderboard feedback (§4.1). Each instance costs 29 forward passes. Flag names are shortened.

et al., 2018): 1,000 resamples drawn once (fixed seed) and shared by both systems. The candidate must win at least 85% (raised from 80% as the pool grew). In every family we built, the highest-scoring variant failed it. (ii) A *dev-to-test count check*: a member whose predicted-positive ratio between development and test falls outside $[0.667, 1.5]$ on any label is rejected; it caught a precision collapse (91 dev, 152 test positives) invisible to the bootstrap, which never sees test. (iii) A *minimum projected mean gain* of 0.005 on test, after shrinking each subtask’s dev delta by its measured transfer rate (Appendix B), before spending a submission. One step motivated both rules (ii) and (iii). It was worth +0.02 on dev ST3, combined the hfss and direct_exhortation swaps, won 84.7% of resamples — below the gate — and we carried it forward anyway. Its exhortation half then failed the count check (91 dev against 152 test positives) and was replaced. Its hfss half changes one development row, wins 62.8% on its own, and changes *no* test row, yet it remains in the merge. Not every step cleared the gate, and the count check was applied to ST3 alone: the submitted system goes outside its bounds on five ST2 labels, where thresholds were frozen without it. Six steps took the merge from a 0.675 multi-task baseline to **0.7645** dev mean (ST3 0.458 to 0.6522; Ap-

pendix F). The three-seed vote supplies 0.6198 of that ST3 score on its own. The five further encoders, 26 of the 29 forward passes, add 0.032 — a gain that wins only 72.7% of resamples and is unmeasured on test.

Cost. Our experiments cost about US\$17 in total: roughly US\$12 of rented A5000 time (failed runs and the 16 GPU-hour retrain of §5 included) plus a free platform quota, and about US\$5 of LLM API for the synthetic data of §5. The deployed system is eight fine-tuned encoders and no generative model: three seed runs, the earlier V0 and V1, and entailment models E1, E2, E5. Scoring the 503 test instances takes 29 passes per instance (one per multi-task model, eight per entailment model), 14,587 in all. Measured training throughput on the rented A5000 was 0.50 s per multi-task instance and 0.18 s per entailment pair, as full training steps with gradient checkpointing. This bounds inference at $503 \times (5 \times 0.50 + 24 \times 0.18) \text{ s} = 57$ minutes on one 24 GB A5000, a few tens of US cents. No API is called at inference.

4 Test results

Table 3 gives the results. Our first test submission, the best development configuration submitted unchanged (X8d in Appendix F), scored 0.634 against 0.7645 on development; the three subtasks

component	encoder	trained on; selected on	supplies
seeds 13/21/34	ModernBERT-large	train + 775 synthetic; dev	ST1, ST2, two ST3 columns
E5	ModernBERT-large (entail.)	train; dev	three ST3 columns
E1	ModernBERT-base (entail.)	train; dev	one ST3 column with E2
E2	ModernBERT-base (entail.)	train; dev	one ST3 column; one with E1
V0, V1 (earlier)	ModernBERT-large	train; dev	one ST3 column
decoding and merge	—	—; dev (504 rows)	all columns
advertiser rule	none (name match)	train labels; —	ST1, two rows

Table 2: The eight encoders of Figure 1 and the two non-neural steps. Checkpoints are answerdotai/ModernBERT-base and -large; E1 uses one-sentence flag definitions paraphrased from the task taxonomy, and E2 the enriched definitions of §5; the synthetic instances were generated with claude-sonnet-5. The advertiser-precedent rule (last row) matches advertiser names by regular expression. Selection used the 504 development rows, except for the predicted-count check (unlabelled test inputs) and the leaderboard feedback behind the last row (§4.1).

system	ST1	ST2	ST3	mean
development	0.8721	0.7691	0.6522	0.7645
test, no rule	0.5879	0.7945	0.5197	0.6340
test, rule (primary)	0.7252	0.7945	0.5197	0.6798
test, retrained	0.5843	0.7846	0.4991	0.6226
test, official entry	0.7923	0.7945	0.5197	0.7022

Table 3: Macro-F1 on development (local scorer) and test (official). The primary row applies the advertiser-precedent rule to both rows it matches, using labelled data alone; the official entry applies it to one, a choice made with one leaderboard probe (§4.1). The retrained row is the train+dev system of §5. The three rows that differ only in the rule are *no rule*, *rule (primary)* and *official entry*: their ST2 and ST3 columns are identical, and only ST1 moves. Bold marks the result we report as primary, which is not the highest mean: the official entry scores higher but used a leaderboard probe.

did not behave alike. ST2 *improved* (0.7691 to 0.7945): frozen thresholds over the three-seed average generalised cleanly. ST3 lost 0.133 (0.6522 to 0.5197): every source and threshold was fitted on 504 rows, and selection error adds up across eight labels.

ST1 fell by 0.284 (0.8721 to 0.5879), for a reason no development-side check could have caught. Test is scored over all five ST1 classes, including other, which has two training instances and no development instance. Its peak predicted probability is 0.24 on test and 0.04 on development, never within 0.1 of the winning class, so our encoder never predicts it. A class that never appears in development cannot be measured, thresholded, gated or count-checked; a certain zero on one class of five loses up to 0.2 of ST1 macro-F1, or 0.067 of the mean. The four live classes also fell, from 0.87 on development to about 0.73 on test.

4.1 A precedent rule for a class that cannot be learnt

More capacity would not have helped: two instances support no decision boundary, and augmenting did nothing (§5). We therefore asked whether anything about other is consistent enough to retrieve. Content type is not: train+dev hold 51 charity- or donation-like segments and only these two carry the label. The concept is not either: both are symbolic or chance-based purchases, but so are seven physical_goods instances from the *same* novelty-title company. The advertiser, though, is informative: the class’s two training instances belong to that novelty-title company (8 labelled rows: 7 physical_goods, 1 other) and to a prize-draw company (2 rows: 1 other, 1 none).

We therefore used retrieval rather than classification: advertiser identity locates test instances of a class too rare to learn, and that advertiser’s training label distribution serves as the prior. It found two candidates, matched on advertiser, not content; the second’s crawled page was an unrelated store, so only its name in the transcript identified it. Relabelling both matched rows is our **primary result**: 0.6798, a gain of 0.046 over the same system without the rule, and the largest single change to our test score. Our train+dev retrain (seven runs, 16 GPU hours) scored **−0.011** (§5), and every gated re-merge gained nothing.

Validation. The rule cannot be validated directly, since its class is absent from development. On the labelled data its two name matches cover ten rows, two of them other; on test one of its two rows was right. As a general method, advertiser precedent does not validate: overriding devel-

opment ST1 with the majority training label of the same product-page domain changes 30 rows, 14 for the better and 16 for the worse (ST1 0.8721 to 0.8489). Restricting it to domains whose training labels agree leaves a single row.

A leaderboard-informed variant. Because the two priors differ (1 of 2 against 1 of 8), we spent one probe submission, within the submission limit, to choose between relabelling both rows and only the higher-prior one. The one-row variant — our official entry — scored **0.7022** (ST1 0.7923), in which a two-example class scores F1 1.0, so the test labels contain exactly one such instance. We report it as an additional result: the choice was made on returned test scores and is worth +0.022.

Disclosure. The rule uses training data and the distributed inputs only — no test gold, no external resources. The split holds out channels, not advertisers, so brand recurrence is within the rules. It is a released script, and the submitted file regenerates from code and released probabilities byte-for-byte (Appendix D). Three things must be stated. First, that the test labels contain other at all was inferred from the published per-subtask aggregates, which is why the rule exists. Second, we kept the original system rather than the retrained one because the latter scored lower on test. Third, the choice between one row and two used one probe submission’s returned score. Even our primary result is therefore not free of leaderboard feedback, although its rule is derived from labelled data alone. Both variants would have placed fourth; the system without the rule (0.6340), the only result free of leaderboard feedback altogether, would have placed below the fifth-placed 0.6699.

5 Five negative results

Each shaped the system of §3. The first is why it averages seeds rather than selecting one, and the second is why the entailment models were trained without synthetic data. The third is why E1’s plain definitions survive in the merge, the fourth why we submitted the original models rather than the retrained ones, and the fifth why the deployed linearisation is the plain one. Mechanisms are in Appendix C; all but the seed study are single-run, so the size of each effect should be read with the caution that Limitations describes.

Seed variance dominates. The 0.033 mean spread in Table 5 is far larger than every gated per-label source swap (all $\leq +0.0098$ on the mean)

and is close to the largest single step we took, the +0.0244 from moving to a seed-averaged base. Single-seed ablations — two below — cannot be trusted at the scale they claim.

Synthetic data helped one architecture and hurt the other. 775 LLM-generated instances aimed at four weak ST3 flags gained +0.006/+0.005/+0.098 (ST1/ST2/ST3) on the multi-task model. They *damaged* the entailment model on three of its four targets (age_restricted 0.500 to 0.143 against the comparable base run), consistent with the generator teaching an easier rule than the annotation function. Only the concept that can be decided from the surface text gained (see also Li et al., 2023; Meng et al., 2022). Neither comparison is matched: the multi-task run also trained for 8 epochs against the baseline’s 3, and a longer schedule without synthetic data recovers 0.067 of that ST3 gain on its own. The entailment run also used the field-budgeted layout of §5 and twice the input length. A later batch of 40 (815 in total) aimed at ST1’s other changed nothing.

Definition prose did not teach boundaries. Enriching the entailment model’s flag definitions with boundary tests, a product-knowledge list and category exclusions moved ST3 from 0.5314 to 0.5167 (confounded with context length; see Limitations); only a positive redefinition helped. We did not test contrastive examples.

Training on development data lowered every score. The train+dev retrain used 21% more data, with epochs, recipe and thresholds frozen in advance and an expected gain of +0.010–0.020 written down beforehand. It scored −0.011, losing on all three subtasks. We think dev rows add volume but no test-channel coverage, and shift the priors.

A silent format regression. The field-budgeted layout of §2, built and verified to keep the page inside the window, cost the large entailment model 0.207 of its ST3 score (0.3867 against 0.5940). A rerun differing only in the layout, which reorders the fields *and* caps their lengths, isolated the cause.

6 Similarity, and what it showed about the labels

No test channel appears in training, but 47% of test instances have a product-page copy at TF-IDF cosine ≥ 0.95 in the labelled pool. Copying the matched instance’s flag set lost to the model: on the 261 development rows with such a match, the

cosine	pairs	ST1 agree	ST3 set agree
≥ 0.95	24	95.8%	62.5%
≥ 0.90	72	90.3%	43.1%
≥ 0.80	241	95.4%	53.1%

Table 4: Label agreement between near-duplicate training pairs (TF-IDF transcript cosine). Agreement does not rise with similarity; treat the band, not one row, as the estimate. Per-label figures, raw and conditional on a positive, are in Appendix E.

copied set agrees with the labels only 38% of the time, which made us measure the annotation itself. Table 4 asks how often the labels agree on near-identical training pairs: ST1 about 90%; the exact ST3 flag set **43%** at cosine ≥ 0.90 , never better than 63%.

Raw per-label agreement is inflated by joint absence. Conditional on at least one instance of a pair carrying the flag (Appendix E), agreement is lower: 93% for undisclosed_advertising, 67% for misleading_claim and 33% for inadequate_disclosure. The two product flags are positive in no pair, so their 100% says nothing.

Part of the disagreement is explained by the inputs, which are near-duplicates in the transcript only. Where the description, the product page and the paid-promotion field also match, the flag sets agree in 8 of 9 pairs, against 24 of 33 at cosine ≥ 0.80 . But the released evidence quotes show that it cannot all be explained that way. There are 41 disagreeing pairs. In only 10 of them is the cited justification for every differing flag absent from the instance that lacks that flag. In the rest the quoted sentence appears word for word in that instance: 16 of 18 misleading_claim differences and 9 of 12 for direct_exhortation. For those flags the labels differ on identical evidence. We therefore read the 57% disagreement as an upper bound on label inconsistency and those 25 cases as a lower one; 70 non-independent instances do not support a rate between them.

These figures separate the two kinds of limit flag by flag. Across the four substantive flags for which a conditional figure exists, our merged system’s development F1 orders almost exactly as their agreement does: direct_exhortation 0.500 at 43%, inadequate_disclosure 0.538 at 33%, misleading_claim 0.770 at 67% and undisclosed_advertising 0.870 at 93%. Where agreement is lowest we cannot tell our error from the labels’; where it is highest the remain-

ing gap is the model’s. This also bears on our gates: where the labels are least consistent, a real improvement is smaller than 504 rows can resolve, which is consistent with their rejecting nearly every candidate. We did not test whether label reliability, rather than capacity or data, was the binding constraint, and we do not claim it.

It is known that label errors make rankings unstable (Northcutt et al., 2021); Plank (2022) argues disagreement is signal; our addition costs nothing — no annotation, no model, no held-out data — and puts a number on that concern (Bowman and Dahl, 2021). We suggest that a future edition report it alongside its baselines: label agreement on instances that are near-identical in every distributed field.

7 Conclusion

We built a monitoring system for commercial content in child-facing videos from eight fine-tuned encoders, merged one label at a time under a paired bootstrap gate. It scored 0.6798 mean macro-F1 on test for a total cost of about US\$17 and under one GPU-hour to label the test set.

With one configuration per level, the access levels were worth less than encoder scale. Across six seeds of one architecture, run-to-run variance exceeded every per-label improvement we adopted, which is why the system averages seeds rather than selecting one. An advertiser-precedent rule for a single ST1 class absent from development added 0.046 on test (0.068 in our leaderboard-informed official entry, 0.7022); applied once, and not validated as a general method, it is not presented as one. Training pairs with near-identical transcripts share a flag set only 43% of the time, and in most content-flag disagreements the cited evidence is present in both instances, so the labels, not only the inputs, differ. What we would keep from this system is the per-label merge under an explicit gate and the seed averaging that variance forced on us. What we would change is its reliance on 504 development rows for both source selection and thresholds.

Limitations

All ablations other than the seed study are single-run, and the seed study covers one architecture with six seeds. The development set (504 rows, with several labels under ten positives) limits what any statistical gate can resolve; the 85% bootstrap

gate is an adoption heuristic, not a hypothesis test; and our development numbers are optimistic by an amount of dev-selection we cannot know. The definition-enrichment comparison is confounded with maximum sequence length (1,024 against 2,048), which we discovered only afterwards. The synthetic-data comparisons are unmatched as well: in each, the run that received synthetic data differs from its reference in a second respect, training length for the multi-task model and input layout and length for the entailment model. Whether the field-budgeted layout harms large models in general or the entailment formulation in particular is unresolved: only entailment models were ever trained with it. The label-consistency estimate rests on 72 near-duplicate pairs found by lexical similarity; it measures the automatic annotation function and not human judgement, and the pairs are not independent (70 instances, one of them in 8 pairs). At $n=72$ the 43% figure carries a 95% interval of roughly ± 11 points, and raw per-label agreement is dominated by joint absence, which is why Table 6 conditions on a positive. Three of our decisions were informed by returned leaderboard aggregates, as disclosed in §4.1, and the transfer-rate discounts of Appendix B likewise derive from a scored submission’s aggregates; test gold beyond those aggregates is unknown to us. Lastly, the precedent rule is stated in general terms but was applied once, to one class, in one task. We report it as a documented decision procedure and not as a validated method.

Ethical Considerations

Assigning a compliance label from an advertiser’s earlier labels, and not from the content at hand, is a form of prior-record profiling. We used it only within the shared-task evaluation and would not defend it as an enforcement mechanism. More generally, at a test ST3 macro-F1 of 0.52 over automatically produced labels, no output of this system should be used to make regulatory decisions about identifiable channels or advertisers.

Acknowledgements

We thank the shared-task organisers for the dataset and the evaluation infrastructure. An AI assistant (Claude) was used throughout for running experiments, analysis code and drafting; the author takes responsibility for all content.

References

- Taylor Berg-Kirkpatrick, David Burkett, and Dan Klein. 2012. [An empirical investigation of statistical significance in NLP](#). In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 995–1005, Jeju Island, Korea. Association for Computational Linguistics.
- Thales Bertaglia, Catalina Goanta, Gerasimos Spanakis, and Gunes Acar. 2026. [ChildSafeAds shared task 2026: Commercial content in child-facing YouTube videos](#). *Preprint*, arXiv:2608.19165.
- Samuel R. Bowman and George Dahl. 2021. [What will it take to fix benchmarking in natural language understanding?](#) In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4843–4855, Online. Association for Computational Linguistics.
- Dallas Card, Peter Henderson, Urvashi Khandelwal, Robin Jia, Kyle Mahowald, and Dan Jurafsky. 2020. [With little power comes great responsibility](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9263–9274, Online. Association for Computational Linguistics.
- Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. 2020. [Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping](#). *Preprint*, arXiv:2002.06305.
- Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. [The hitchhiker’s guide to testing statistical significance in natural language processing](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.
- Philipp Koehn. 2004. [Statistical significance tests for machine translation evaluation](#). In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.
- Zhuoyan Li, Hangxiao Zhu, Zhuoran Lu, and Ming Yin. 2023. [Synthetic data generation with large language models for text classification: Potential and limitations](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10443–10461, Singapore. Association for Computational Linguistics.
- Zachary C. Lipton, Charles Elkan, and Balakrishnan Naryanaswamy. 2014. [Optimal thresholding of classifiers to maximize F1 measure](#). In *Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, pages 225–239. Springer.

Ilya Loshchilov and Frank Hutter. 2019. [Decoupled weight decay regularization](#). In *International Conference on Learning Representations*.

Yu Meng, Jiaxin Huang, Yu Zhang, and Jiawei Han. 2022. [Generating training data with language models: Towards zero-shot language understanding](#). In *Advances in Neural Information Processing Systems*, volume 35, pages 462–477. Curran Associates, Inc.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2018. [Mixed precision training](#). In *International Conference on Learning Representations*.

Curtis Northcutt, Anish Athalye, and Jonas Mueller. 2021. [Pervasive label errors in test sets destabilize machine learning benchmarks](#). In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

Barbara Plank. 2022. [The “problem” of human label variation: On ground truth in data, modeling and evaluation](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10671–10682, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2017. [Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging](#). In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 338–348, Copenhagen, Denmark. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021. [Exploiting cloze-questions for few-shot text classification and natural language inference](#). In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Griffin Thomas 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, Vienna, Austria. Association for Computational Linguistics.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. [Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural*

Language Processing (EMNLP-IJCNLP), pages 3914–3923, Hong Kong, China. Association for Computational Linguistics.

A Seed variance

Table 5 gives the six-seed study behind §3 and §5: one configuration, changing nothing but the seed.

seed	ST1	ST2	ST3	mean
13	0.8181	0.7598	0.5555	0.7111
21	0.8411	0.7635	0.6061	0.7369
34	0.8043	0.7524	0.5542	0.7037
55	0.8074	0.7432	0.6230	0.7245
89	0.8283	0.7491	0.5890	0.7222
144	0.8008	0.7611	0.6275	0.7298
avg. (6)	0.8330	0.7722	0.6427	0.7493
+ST1 peak	0.8475	—	—	0.7542

Table 5: Six seeds of one configuration, changing nothing else (dev, tuned). The averaged probabilities beat every individual seed on ST2, ST3 and the mean (seed 21’s ST1 stays higher); the last row averages ST1 from each seed’s own best-ST1 epoch. The submitted three-seed average (13/21/34) reached ST1 0.8721, above every row here, partly by chance on rare classes, as the six-seed fall to 0.8330 shows.

B Exact configurations

Both trainers are in the released repository (train_modernbert.py, train_st3_entailment.py); the invocations below are the ones that produced the submitted system.

Checkpoints.

answerdotai/ModernBERT-large for the multi-task models, the large entailment model E5 and the large entailment run of §5; answerdotai/ModernBERT-base for the E1, E2 and E4 entailment runs (E4 is the synthetic-data run of §5), of which E1 and E2 supply two columns of the final merge, and for the level ablation of Table 1. Sentence embeddings for the retrieval step of §4.1 use sentence-transformers/all-MiniLM-L6-v2. Synthetic data was generated with claude-sonnet-5 through the batch API (815 instances in two batches, about US\$5); the prompts, per-label directives and cleaning code are in the released generate_synthetic.py. Environment: torch 2.5.1+cu121, transformers 5.14.1; the full freeze is released.

Linearisation.

One string per instance: tagged fields joined by newlines, in the order [TRANSCRIPT], [TITLE], [DISCLOSURE],

[DESCRIPTION], [CHANNEL], [PAGE_URL], [PAGE_TITLE], [PAGE], right-truncated by the tokeniser at `max_length`. Lower access levels drop the fields they do not have. The field-budgeted variant of §5 sits behind an environment flag in the released `data.py`; no deployed model uses it.

Multi-task models. The submitted ST1/ST2 base averages three runs (seeds {13, 21, 34}; seeds {55, 89, 144} were trained after our first test submission, for the study of Appendix A): level 4, `max_length` 2048, 8 epochs with the 775 synthetic instances of §5 included, batch $8 \times \text{grad-accum } 2$ (effective 16), encoder LR 3×10^{-5} , head LR 1×10^{-4} , AdamW (Loshchilov and Hutter, 2019) weight decay 0.01, linear schedule with 6% warmup, fp16 automatic mixed precision with dynamic loss scaling (Micikevicius et al., 2018) (inference in fp32), gradient checkpointing, each seed’s best dev checkpoint (the best ST1 epoch varies across seeds, 2–4, so the trainer also saves the best-ST1 checkpoint and per-epoch probabilities). Class weights are inverse-frequency, clipped to [0.25, 8.0] for ST1 and at 12 for the ST2/ST3 positive weights. Two earlier multi-task runs survive in one merged column (hfss): V0, the same recipe at 3 epochs without synthetic data (the baseline of Table 1), and V1, a longer schedule with early stopping.

Entailment model (E5, the ST3 source for three labels): `max_length` 2048, 8 epochs, batch $16 \times \text{grad-accum } 4$ (effective 64) with gradient checkpointing to fit in 24 GB, same LRs and schedule, seed 13, best epoch 3 selected on tuned dev ST3. E1 and E2 (ModernBERT-base, 6 epochs, effective batch 64) also contribute columns; E1 was trained at `max_length` 1024, which confounds the definition comparison of §5; see Limitations.

Final ST3 source map. `undisclosed_advertising`, `direct_exhortation`, `misleading_claim` $\leftarrow$ E5; `age_restricted`, `no_flag` $\leftarrow$ the three-seed average; `hfss` $\leftarrow$ `mean(V0, V1)`; `inadequate_disclosure` $\leftarrow$ `mean(E1, E2)`; `insufficient_context` $\leftarrow$ E2. Eight encoders in total. The multi-task columns are votes (see *ST3 decoding* below), and the tuned thresholds require two of three seeds for `age_restricted`, one of three for `no_flag`, and either of V0 and V1 for `hfss`; these are the cell labels of Figure 1.

Thresholds. One per label, grid-searched on development over 0.05–0.90 in steps of 0.05, maximising that label’s F1; ST1 is plain argmax and

has no tuned threshold. Thresholds were frozen before test inference and never refitted. Submission spending used shrinkage-discounted projections of test gain, with per-subtask transfer rates measured from our first scored test submission: ST2 dev deltas at face value, high-support ST3 deltas at half, rare-label ST3 deltas at zero, ST1 deltas at 0.3. Everything that was tuned (source selection, thresholds, merge decisions) was fitted on the 504 development rows and nothing else; the one exception, a decision informed by returned leaderboard aggregates, is disclosed in §4.1.

Label constraints. Train+dev pooled ($n=2,857$): `official_disclosure true` implies `never_undisclosed_advertising` (0 of 1,565); `undisclosed_advertising` and `inadequate_disclosure` are mutually exclusive (1 of 2,857); $P(\text{age_restricted} \mid \text{ST2} = \text{gambling}) = 0.88$. The ST3 decoder enforces the first two. The first is not a formality: before decoding, E5 predicts `undisclosed_advertising` on 33 development rows whose `paid-promotion` field is true, and without the rule the merged system’s F1 for that flag falls from 0.870 to 0.722 and development ST3 from 0.6522 to 0.6355.

ST3 decoding. Every source is decoded before a column is taken from it, by five rules: (1) where the platform’s `paid-promotion` field is true, the `undisclosed_advertising` probability is set to zero; (2) labels at or above threshold are kept, or the most probable label if none; (3) of `undisclosed_advertising` and `inadequate_disclosure` only the more probable survives; (4) if `no_flag` or `insufficient_context` survives beside substantive flags, it is kept alone when its probability exceeds theirs and dropped otherwise; (5) of the two housekeeping labels only the more probable survives. Where a column combines models, entailment probabilities are averaged before thresholding, whereas multi-task models vote over their decoded outputs, with the number of votes required tuned on development. A repair pass then removes `no_flag` wherever a substantive flag fires and assigns it to rows left empty (4 development and 3 test rows).

C Negative results in detail

Seed variance. A paired bootstrap over *instances* is blind to variance over *training runs*: a change can win 99.9% of resamples and still be smaller

than a reseed. This is why adoption gates alone do not protect single-seed ablations.

Synthetic data. The batch carried 80 purpose-built hard negatives. Real `age_restricted` depends on product knowledge (an 18+ rating, a bong, a skin-gambling site), whereas the generated cases announce themselves; entailment expands each instance into eight pairs, amplifying every artefact eight times over. Against the comparable base entailment run E1, which differs in the two respects named in Limitations: `age_restricted` $0.500 \rightarrow 0.143$, `hfss` $0.250 \rightarrow 0.286$, `direct_exhortation` $0.326 \rightarrow 0.319$; only `insufficient_context`, the one concept decidable from surface text, improved ($0.545 \rightarrow 0.667$). Against the best earlier source of any kind per flag the references are 0.640, 0.571, 0.369 and 0.615, a best-of-four that is biased upward. The multi-task model, consuming the same data, gained $+0.006/+0.005/+0.098$ on ST1/ST2/ST3 over a 3-epoch baseline while training for 8: even the *sign* of the synthetic effect depends on the consuming architecture, but neither comparison isolates the data.

Definitions. The enriched definitions added boundary tests (“click the link” does *not* count as exhortation), a product-knowledge list and category exclusions. The negation-phrased clauses did not land, and the product-knowledge list cost 0.125 on its own target label. The comparison is confounded with maximum sequence length; see Limitations.

Train+dev retrain. Thresholds were *count-matched*: cut-points chosen so each label emits exactly the predicted-positive count of our scored submission. This was forced on us because models retrained with dev inside are overconfident, so transplanted thresholds over-fire (`misleading_claim` initially fired on 67% of test rows against a 52% dev base rate).

Format regression. The final training loss was 0.62 against 0.22, which means severe underfitting, not overfitting. The layout had been introduced together with a schedule change, so the cause stayed ambiguous until the controlled rerun: epoch 0 of the original-layout rerun (0.3535) was already at the level of the reordered run’s epoch-7 final (0.3867), and the rerun then climbed to 0.5940 while the reordered run never left the 0.39 range. At base scale the same layout cost far less — 0.4947 against E2’s 0.5167 — but that run also carried the synthetic data of §5, so it does not

cleanly isolate scale. The damage is specific either to scale or to the entailment formulation, and we did not resolve which.

D Released artefacts

We release all experiments and not only the submitted system: trainers and analysis scripts; per-run probability matrices, tuned thresholds and submissions for every experiment in Appendix G, failures included; checkpoints with checksums; the environment freeze; and EXPERIMENTS.md, a dated log of every decision, correction and rejected candidate. Every analysis script carries a *bit-exact self-test*: the merge assembler must reconstruct a previously scored submission with no differing cells before any new number from it is used. We adopted this check after analysis scripts were lost to temporary storage and earlier submissions had to be reconstructed. Both test results of §4.1 regenerate byte-for-byte from the released probabilities and code, and for V0 and V1, whose trainer predates the probability dumps, we release the decoded submissions instead. The weights themselves, 0.6–1.6 GB each, exceed the repository’s file-size limit and are not hosted; the environment freeze, the checkpoint inventory with checksums and the loading instructions are released so that they can be retrained.

E Per-label agreement on near-duplicate pairs

Table 6 gives the per-flag figures behind §6, raw and conditional on at least one instance of a pair carrying the flag.

label	raw	≥ 1 pos.	both	cond.
<code>age_restricted</code>	100.0%	0	0	—
<code>hfss_food_marketing</code>	100.0%	0	0	—
<code>undisclosed_adv.</code>	97.2%	29	27	93%
<code>insufficient_context</code>	97.2%	2	0	0%
<code>no_flag</code>	88.9%	13	5	38%
<code>direct_exhortation</code>	83.3%	21	9	43%
<code>misleading_claim</code>	75.0%	54	36	67%
<code>inadequate_discl.</code>	75.0%	27	9	33%

Table 6: Label agreement on the 72 near-identical training pairs (transcript cosine ≥ 0.9). Raw agreement counts a pair in which neither instance carries the flag as agreeing, which inflates it for rare flags; the last three columns give the pairs in which at least one and both instances carry it, and their ratio, which is why the product flags have no conditional entry. The first, third and last names are shortened to fit.

F Development progression

Table 7 traces the merged system from the multi-task baseline to the submitted configuration, one gated step at a time.

system	ST3	mean	note
V0 alone	0.458	0.675	multi-task baseline
S2 first merge	0.5747	0.7138	scored externally
X3 best-of	0.5893	0.7187	scored externally
X4d +V0 hfss	0.6097	0.7255	scored; test unchanged
X6b +v0syn	0.6278	0.7353	local only
X7a 3-seed avg	0.6377	0.7597	local only
X8d +E5	0.6522	0.7645	final dev config

Table 7: Development progression of the merged system. X3 passed the 80% gate then in force; X4d reached 84.7% against the 85% gate that had replaced it and was carried forward regardless (§3); X6b, X7a and X8d won 99.9%, 92.6% and 93.7% of resamples. Three steps also include ungated structural additions: S2 is the first entailment merge, X6b moves the ST1/ST2 base from V0 to seed 13, and X7a moves it to the three-seed average. Our local scorer matched the external development-phase leaderboard to four decimal places on five consecutive externally scored submissions (three of them appear in this table), so we treat the post-closure local numbers as trustworthy, though unverified.

G Summary of runs

Table 8 lists every training run and major analysis with its outcome. Full dated detail, including per-epoch curves and all rejected candidates, is in EXPERIMENTS.md in the released repository.

H Decision rules and their motivation

Table 9 lists the operating rules we adopted and the observation that motivated each one.

run	outcome
V0/V1	multi-task baselines; the short annealed schedule (V0) scored above the early-stopped one (V1) on the mean
E1	entailment, ST3 0.5314; best on the disclosure flags
E2	enriched definitions: lower (0.5167); only the positive rephrasing helped
E3s1	large entailment, stopped by a 12 h platform limit at epoch 2 (0.5007, still improving); outputs later recovered
SYN1	775 LLM-synthetic instances (§5)
E4	entailment + SYN1 + field-budgeted layout: lower on 3 of 4 target flags
e3large	large entailment + field-budgeted layout: 0.3867; the layout, not the schedule, was the cause (controlled rerun, E5)
E5	large entailment, plain truncation: 0.5940 , best single ST3 model; product flags 0.611, but its product-label swaps did not pass the gate
seeds×6	0.704–0.737 range; the average is above every seed on ST2, ST3 and the mean; the best ST1 epoch varies (2–4)
TD	six multi-task seeds and E5 retrained on train+dev, all settings frozen: −0.011 on test
X and Y sets	candidate per-label merges under the 85% gate; in every candidate set the highest scorer did not pass
similarity	lexical, semantic, domain and prevalence copying: all scored below the model; led to the check of §6
X9/X10	advertiser-precedent rule on two rows (primary, 0.6798: +0.046) and on one row (official entry, 0.7022: a further +0.022)

Table 8: Summary of training runs and major analyses; identifiers are those of EXPERIMENTS.md.

rule	motivation
Paired bootstrap gate, 1,000 resamples, fixed seed	intervals on a development gain are up to 0.16 wide, so point comparisons need pairing
Gate raised from 80% to 85%	the candidate pool grew, and multiple comparisons make a chance winner likely
Predicted-count bounds [0.667, 1.5] between development and test	one candidate source predicted 91 development and 152 test positives, a shift the bootstrap cannot see
Minimum projected gain of 0.005 before using a test submission	the submission budget was limited, and a +0.02 development gain had changed no test rows
Shrinkage-discounted projection of test gain	per-subtask transfer rates measured from our first scored submission: ST2 $\times$ 1, high-support ST3 $\times$ 0.5, rare ST3 $\times$ 0, ST1 $\times$ 0.3
Average seeds, never select one	selecting the best seed repeats the per-label selection error one level up
Select entailment checkpoints on the tuned score	a run's raw and tuned rankings differed; the earlier rule would have selected a checkpoint 0.046 lower
Best-ST1 checkpoint and per-epoch probability dumps	subtasks peak at different epochs (ST1 at 2–4, ST2 at 5–7), so a single checkpoint is a compromise
Frozen epochs, thresholds and recipe for train+dev runs	with development data inside training, no development-side number can be used for selection
Count-matched thresholds for retrained models	training on development data inflates confidence, so the original cut-points predict too many positives
Analysis code kept in the repository, with bit-exact self-tests	three artefacts were lost to temporary storage; every script must reproduce a stored output exactly
Copy outputs before releasing a rented machine	rented platforms discard state; checkpoints were lost once and later recovered
Probe submissions while the entry can still be chosen	used once, to choose between two rule outputs; disclosed in §4.1

Table 9: Operating rules and their motivation.