

thylinao at the ALTA 2026 Shared Task: A Blend of Encoders, Decoders with Adapters and Frozen Language Model Readouts for Sentiment and Sarcasm Classification in Two Varieties of English

Maksim Silchenko

National University of Singapore
maks.silchenko.04@gmail.com

Abstract

The ALTA 2026 Shared Task gives Google reviews and Reddit comments written in Australian or British English. A system must predict, for each text, whether it is positive or negative and whether it is sarcastic. Each label is scored only on the variety of English where the system does worse, so strength on one cannot make up for weakness on the other. The system of team thylinao averages the probabilities predicted by three kinds of model: fine-tuned encoders, larger generative language models fine-tuned with low-rank adapters, and language models that are prompted but not trained. A threshold for each label, variety and source turns the average into a label. In cross-validation, this average scored about 0.01 above its best single model, mostly through the fine-tuned generative models. Changes to the loss or training data of a single encoder did not reach the gain of 0.005 set in advance for accepting a change. Our best submission, which also includes prompted models whose gain was below that mark, scored 0.8675 and ranked first on the official test leaderboard. The organisers found the difference of 0.0007 to the run ranked second not statistically significant and treated the two runs as tied.

1 Introduction

Classifiers for sentiment and sarcasm are usually evaluated on one standard variety of a language. BESSTIE (Srirag et al., 2025) provides labelled reviews and comments in three varieties of English. Its authors report that models perform less well on some varieties than on others, and that sarcasm is harder than sentiment in all of them. The ALTA 2026 Shared Task (Mollá et al., 2026) uses the Australian and British parts of this dataset. For each of the two tasks, sentiment and sarcasm, its score takes the lower of the macro-F1 values of the two varieties (Section 2). A system therefore im-

proves its score only by improving its result on the variety on which it is weaker.

This paper describes the system that team thylinao submitted. The system averages the probabilities of three families of model: fine-tuned encoders, decoder language models fine-tuned with low-rank adapters, and frozen language models whose answers are turned into probabilities (Section 3). We call this average the *blend*. A decision threshold is then applied, tuned for each task, variety and source of the text.

The labelled data contain few examples of the rarer classes, so small differences between systems are hard to measure. We therefore describe in some detail how systems were compared, including a rule for accepting a new component that was fixed in advance (Section 4).

The best of our three submissions scored 0.8675 and ranked first on the official test leaderboard. The organisers found the difference of 0.0007 to the run ranked second not statistically significant and treated the two runs as tied (Section 5). We report the following observations.

- In our cross-validation, the one accepted change that we used was the addition of the decoders with adapters to the blend. No change to the loss or to the training data of a single encoder met the acceptance rule (Section 6).
- Two further ideas, tested under a rule recorded before their results were known, did not meet it either. Adding the most similar labelled training texts to the input of a fine-tuned encoder lowered the score, and adding notes written by a language model did not change it measurably (Section 7).
- The difference of 0.0007 between the first two runs on the leaderboard is small against the variation estimated for our own scores. For the difference between two of our sub-

Variety	Source	Rows		Test	Positive (%)	
		Train	Valid		Sent.	Sarc.
en-AU	Google	946	130	270	73.5	7.0
en-AU	Reddit	1,763	241	501	31.9	41.9
en-UK	Google	1,817	248	516	74.8	0.1
en-UK	Reddit	1,005	138	286	11.4	21.7
All		5,531	757	1,573	49.4	18.5

Table 1: Number of rows in each cell of the released data, and the share of training rows with positive sentiment and with sarcasm.

missions on about as many rows as the test set, the standard deviation was about 0.007 (Section 5).

Related work. On BESSTIE, Maheshwari et al. (2026) fine-tune language models with one low-rank adapter per variety and with a mixture of experts. Singh et al. (2025) add explanations to its sarcastic texts and study prompting for explainable sarcasm detection in Australian and Indian English. Our system does not model the varieties separately. It gives the variety to each model as part of the input and tunes the thresholds per variety.

2 Task, Data and Metric

Task. The ALTA 2026 Shared Task (Mollá et al., 2026) asks for two binary labels for each text: sentiment (negative or positive) and sarcasm (not sarcastic or sarcastic). We refer to the prediction of each label as a *task*, so there is a sentiment task and a sarcasm task. Each text is written in Australian English (en-AU) or British English (en-UK).

Data. The data are the en-AU and en-UK parts of BESSTIE (Srirag et al., 2025). Each text comes from one of two sources, Google Places reviews or Reddit comments. The organisers released 5,531 training rows and 757 validation rows with labels, and 1,573 test rows whose labels were withheld. We call a combination of variety and source a *cell*. Table 1 shows that the four cells differ. About three quarters of the Google reviews are positive and few are sarcastic; in the en-UK Google cell, one training row out of 1,817 is sarcastic. Most Reddit comments are negative, and 42 percent of the en-AU comments and 22 percent of the en-UK comments are sarcastic. The two labels are closely linked: 990 of the 1,024 sarcastic training rows have negative sentiment.

Metric. For each task and each variety, the organisers compute macro-F1, which is the un-weighted mean of the F1 values of the two classes. This gives four values. The score of a system is

$$\frac{\min(s_{AU}, s_{UK}) + \min(c_{AU}, c_{UK})}{2}, \quad (1)$$

where s is the macro-F1 for sentiment and c the macro-F1 for sarcasm. As an example, our best submission obtained 0.9257 (en-AU) and 0.9763 (en-UK) for sentiment, and 0.8093 (en-AU) and 0.8371 (en-UK) for sarcasm. The lower values of the two tasks are 0.9257 and 0.8093, and their mean is 0.8675. Only the lower value of each task enters the score, so an improvement on the other variety leaves the score unchanged. We call the variety with the lower value the *binding* variety of that task.

Rules. Predictions had to be produced automatically, and any resource could be used except the labels of the test data. Our submitted systems were trained only on the labels released for the shared task. Each team could make three submissions on the test set, and the best of the three was ranked. The labels of the validation file were public, and the development leaderboard was computed on that file. We trained on that file as well, so we did not use the development leaderboard to choose between systems and relied on cross-validation instead (Section 4). The organisers’ baseline scored 0.7727 on the test set.

3 System

3.1 Overview

The system has three stages (Figure 1). First, several models each give, for every text, a probability that the sentiment is positive and a probability that the text is sarcastic. We call each of these models a *member*. Second, the probabilities of the members are averaged with fixed weights, separately for each task. Third, the averaged probability is compared with a decision threshold that depends on the task and on the cell of the text. The members belong to three families, described below and listed in Table 2.

All trained members are fitted on the training and validation files together (6,288 rows). The rows are split into five folds, three times (Section 4.1). For each fold, one model is trained on the other four folds. We call it a *fold model*. Each

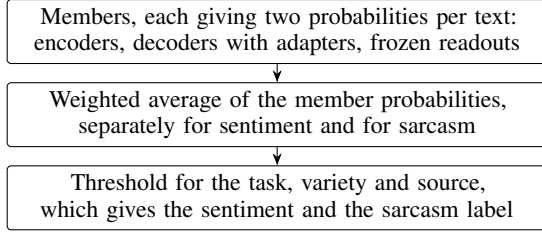


Figure 1: The three stages of the system.

Member	CV	S1	S2	S3
<i>Fine-tuned encoders</i>				
DeBERTa, weights per variety	0.8591		1/1	
DeBERTa, four-class head	0.8551	0/1		0/1
DeBERTa, sarcasm loss twice	0.8614	1/2		1/2
RoBERTa-large	0.8339	1/1		1/1
XLM-RoBERTa-large	0.8158	1/0		1/0
<i>Decoders with adapters</i>				
Llama-3.3-70B-Instruct	0.8540	2/0	1/2	2/0
Mistral-Small-Instruct-2409	0.8558	1/1	0/1	1/1
Qwen3-32B	0.8531	0/1	0/1	0/1
<i>Verbalised readouts</i>				
Qwen3.8-27B	0.8154		1/0	
ornith1.5:35b	0.8125		1/0	
<i>Token-probability readouts</i>				
Direct prompt	0.8222			1/1
Guideline prompt	0.8346			1/1
Guideline with neighbours	0.8504			1/1
Units per task		6/6	4/5	9/9

Table 2: The members of the three submissions (S1 to S3). CV is the cross-validated score of the member used alone (mean of the two decision repetitions, Section 4.1). An entry such as 0/1 gives the units of weight for sentiment and for sarcasm, and the weight in a task is the number of units divided by the units per task. DeBERTa stands for DeBERTa-v3-large.

trained member therefore consists of 15 fold models. The input to every trained model is the string “variety | source | text”, so the model is told the variety and the source.

3.2 Fine-tuned encoders

An encoder is a language model that reads the whole text and returns one vector per token. The first family consists of DeBERTa-v3-large (He et al., 2023), RoBERTa-large (Liu et al., 2019) and XLM-RoBERTa-large (Conneau et al., 2020). The mean of the token vectors feeds two linear heads, one per task. A linear head is a single layer that turns the vector into scores for the two classes. The two cross-entropy losses are added, and class weights are used because the classes are imbalanced. For DeBERTa, we use three variants of the loss. In the first, the class weights are computed separately for each variety. In the second, a single

head predicts the four combinations of the two labels. In the third, the class weights are pooled over the varieties and the sarcasm loss counts twice. RoBERTa and XLM-RoBERTa use the first variant. All encoders are trained for four epochs with a maximum input length of 256 tokens. Appendix A gives the remaining settings.

3.3 Decoders with adapters

A decoder is a language model trained to predict the next token. The second family consists of three decoders, which are larger than the encoders: Mistral-Small-Instruct-2409 (Mistral AI, 2024), Qwen3-32B (Yang et al., 2025) and Llama-3.3-70B-Instruct (Grattafiori et al., 2024; Meta Llama, 2024). Each model is loaded with its weights stored in four bits each (quantisation) and is fine-tuned with low-rank adapters of rank 64 (Hu et al., 2022; Dettmers et al., 2023). These are small added matrices. The models are not used to generate text. As with the encoders, two linear heads and the loss with class weights per variety are used. The heads are placed on the vector of the last token. Only the adapters and the heads are trained. Training runs for three epochs.

Some of these fine-tuning runs diverged: the training loss stayed high and the scores on the held-out fold stayed low. A fold model whose mean training loss in the second epoch is above 0.6 is therefore retrained with a different random seed. This rule reads only the training loss. Of the 45 fold models of the three decoders, 8 were retrained in this way (Appendix A).

3.4 Frozen language model readouts

The third family uses language models without any training. The models are reached through an inference service operated by our university, with one request per text. No parameter is fitted for these members. Every request asks the model to answer directly, with its reasoning mode switched off.

A readout turns an answer of the model into the two probabilities. We use two kinds. In the first, which we call the *verbalised readout*, the prompt gives a short definition of sarcasm and a description of the two sources, and the model writes two numbers between 0 and 100. These are its estimates, in percent, of the probability that the sentiment is positive and that the text is sarcastic. This is done with two models, Qwen3.8-27B (Qwen, 2026) and a second model that the ser-

vice lists as `ornith1.5:35b`, for which we do not know the public release name. A third member, with four fixed labelled examples in the prompt for Qwen3.8-27B, is compared in Section 6 but is in no submission.

In the second, which we call the *token-probability readout*, the model answers on two lines, “sarcastic: yes or no” and “sentiment: positive or negative”. The service returns the probabilities of the most likely tokens at each position. We take the probability of “yes” divided by the sum of the probabilities of “yes” and “no”, and likewise for “positive” and “negative”. Three prompts are used with Qwen3.8-27B:

- *Direct*: the same definition of sarcasm as in the verbalised readout, with a shorter description of the two sources.
- *Guideline*: the label definitions from the annotation guideline of the dataset paper.
- *Guideline with neighbours*: the guideline prompt, with six labelled texts of the same variety and their labels placed before the text (Liu et al., 2022). These are the six most similar labelled texts by the cosine of BGE-M3 embeddings (Chen et al., 2024). When none of them is sarcastic, the least similar one is replaced by the most similar sarcastic text; this was the case for about 61 percent of the cross-validation rows.

For a row of the cross-validation data, neighbours are never taken from the fold that contains the row, which also excludes the row itself and exact copies of it. For a test row, all labelled rows can be neighbours. Appendix B gives the prompts.

3.5 Blending

For each task, the blended probability is a weighted average of the member probabilities. For Submissions 1 and 2, the weights are found by greedy forward selection with replacement (Caruana et al., 2004). The search starts from the best single member. At each step it adds, to one of the two tasks, the member that most improves the cross-validated score of Section 4.1, and it stops when no addition improves the score by more than 0.0001. Each selection adds one unit of weight, so a member that is selected twice counts twice. Submission 3 adds members to the weights of Submission 1 by the rule of Section 4.3.

3.6 Decision thresholds

A text receives the positive label of a task when its blended probability is at least as large as a threshold. The share of positive labels differs between cells (Table 1), so we did not use one threshold of 0.5 for all rows. We use eight thresholds, one for each combination of task, variety and source. For each task and variety, one threshold is first tuned on all rows of the variety. The thresholds of the two sources start from this value and are then adjusted in turn, each to maximise the macro-F1 of the variety (Lipton et al., 2014), because this is the quantity that enters the score. A cell with fewer than ten positive rows is not adjusted and keeps the starting value; this applies to sarcasm in the en-UK Google reviews. A second starting value is also tried (Appendix C).

As an example, in Submission 3 the sarcasm threshold is 0.410 for en-AU Reddit comments and 0.431 for en-UK Reddit comments (Appendix C). A blended probability of 0.42 therefore gives the label sarcastic for an en-AU Reddit comment and the label not sarcastic for an en-UK Reddit comment.

For the test set, the out-of-fold probabilities of the blend (Section 4.1) are averaged over the three repetitions, and the thresholds are tuned once on all 6,288 labelled rows. The probability of a trained member for a test row is the average over its 15 fold models.

4 Evaluation Protocol

4.1 Cross-validation

The choices between systems were made with repeated cross-validation on the 6,288 labelled rows. We built three repetitions of five folds each. The folds are stratified by variety, source and the two labels, and texts that are exact copies of each other are placed in the same fold. A fold model predicts only the rows of the fold that it was not trained on. We call these *out-of-fold* predictions.

The thresholds of Section 3.6 are handled in the same way. For each fold, the thresholds are tuned on the rows of the other four folds and applied to the rows of that fold. The labels obtained for all rows are then scored with the organisers’ scoring script. We call the result the *cross-validated score*. With this procedure, a threshold is never evaluated on a row that was used to tune it.

The acceptance rule (Section 4.2) and the search for the blend weights read two of the three rep-

etitions. We call them the *decision repetitions*. The third, the *check repetition*, was not read by either. Its scores were computed for single members during development and for each blend after its weights had been fixed. We report the scores of the three submissions as a check (Section 5.2). In the check repetition, all validation rows were placed in the same fold, which Section 5.2 uses to score them alone.

4.2 Acceptance rule for new components

We fixed a rule for accepting a new component and recorded it on 5 September 2026, before the first comparison. A component was accepted when three conditions held on the two decision repetitions.

- The cross-validated score rose by at least 0.005 on average.
- The lower of the two sarcasm macro-F1 values (en-AU and en-UK) did not fall by more than 0.005 on average.
- The score was also computed on the rows of each fold alone, which gives ten differences between the system with and without the component. These ten differences were re-sampled with replacement, a bootstrap (Efron and Tibshirani, 1994), and the 95 percent interval of their mean had to lie above zero.

Comparisons that were run together also had to pass the Benjamini-Hochberg procedure (Benjamini and Hochberg, 1995) at a level of 0.10, applied to the share of bootstrap means at or below zero. The procedure limits the share of false acceptances among several comparisons.

We checked the rule on simulated members built from the real labels and folds. A change that added only noise was accepted in 3 of 100 trials, and a change that corrected 30 percent of the errors of a member in 99 of 100. When the candidate was a separately generated member of the same quality, it was accepted in 19 of 100 trials. The bootstrap over folds does not capture the variation between separately trained models, so its interval is too narrow in that case. We kept the rule as it had been fixed.

4.3 Choice of the three submissions

The three submissions allowed by the rules were chosen as follows.

Submission 1. This submission is the result of the greedy search of Section 3.5 over a pool of nine members. The pool is the set of members from which the search may choose. The starting pool held the five encoders of Table 2 and a logistic regression on TF-IDF features (the TF-IDF classifier). The three decoders with adapters were added after they had met the acceptance rule as a family (Section 6). The search did not select the TF-IDF classifier or the DeBERTa member with class weights per variety.

Submission 2. This submission is a blend that was chosen to differ from Submission 1. A candidate qualified under two conditions. Its cross-validated score had to be within 0.01 of that of Submission 1. Its en-UK sarcasm predictions had to differ from those of Submission 1 on at least 5 percent of the rows that either blend labelled sarcastic. When this condition was written, en-UK, which has fewer sarcastic rows, was regarded as the binding variety for sarcasm. Among the qualifying candidates we took the one with the highest cross-validated score. It was obtained by removing the member with the largest weight in Submission 1 (DeBERTa with the sarcasm loss counted twice) from the pool, adding the verbalised readouts to the pool, and running the greedy search again.

Submission 3. This submission was chosen by a rule recorded on 17 September 2026, before any of the candidate members existed. The first version of the rule considered only blends that contain newly fine-tuned decoders. Two amendments, on 17 and 18 September, added the candidate that was finally selected, Submission 1 with token-probability readouts added. The second amendment was written after results for two of the readouts on two samples of about 700 rows were known, and before any token-probability readout had been run on all rows (Appendix E).

For each candidate, the rule takes the expected value of the best of three scores, those of Submissions 1 and 2 and of the candidate. The candidate with the highest value is chosen, provided that its cross-validated score is not more than 0.005 below that of Submission 1. The expected value is estimated on each of the two decision repetitions and then averaged. For each repetition, we draw 5,000 bootstrap samples of out-of-fold rows, each of about the size of the test set, score the three blends on every sample, and average the highest of

the three scores. The selected candidate adds the three token-probability readouts to Submission 1, each with one unit of weight per task.

From the same samples we recorded, before any test score was available, the share of samples in which each submission was the best of the three: 0.21 for Submission 1, 0.28 for Submission 2 and 0.51 for Submission 3.

5 Results

5.1 Official test results

Table 3 gives the results. The three submissions scored 0.8582, 0.8627 and 0.8675 on the test set. In all three, en-AU was the binding variety in both tasks, and the lowest of the four values was en-AU sarcasm.

With 0.8675, Submission 3 ranked first on the official test leaderboard, which has 22 runs, one per participant, including the organisers’ baseline. The run ranked second scored 0.8668. The organisers compared the two runs with a Wilcoxon signed-rank test on 500 bootstrap resamples and found the difference not statistically significant ($p = 0.40$). They treated the two runs as tied.¹

Separately from the organisers’ significance test, we estimated the variation of our own scores with the bootstrap of Section 4.3, which resamples cross-validation rows. It gives a standard deviation of about 0.0095 for the score of one of our submissions on a sample of about the size of the test set. For the difference between two of our submissions it gives 0.0065 to 0.0078, depending on the pair. On the test set, our three submissions differ by 0.0045 to 0.0093, about 0.6 to 1.4 times the standard deviation of the same pair. We have no such estimate for the difference between our run and the run ranked second.

5.2 Cross-validation estimates and test results

Level. The cross-validated scores of the decision repetitions were 0.0063 to 0.0137 above the test scores. This direction is expected in part, because the blend weights were selected on the same out-of-fold predictions. The check repetition gave lower values, still above the test scores.

Order. Averaged over the two decision repetitions, cross-validation ordered the three submissions as the test set did. Submission 3, the best

of the three in a share of 0.51 of the bootstrap samples (Section 4.3), obtained the highest score. This is a single outcome and may be due to chance. The three cross-validated scores lie within 0.002 of each other, and neither repetition taken alone gave the order of the test set. The expected best score had been estimated as 0.8752; the best score obtained was 0.8675.

The validation rows alone. In the check repetition, all 757 validation rows lie in one fold. We scored the three blends on these rows alone, with thresholds tuned on the other four folds. The scores were 0.8527, 0.8474 and 0.8765. This is a spread of 0.0291, against 0.0093 on the test set, and a different order. On these rows the binding variety for sarcasm was en-UK, whereas it was en-AU on the full cross-validation data and on the test set. This is one outcome, and the test set does not separate the three submissions either (Section 5.1).

The four macro-F1 values. In cross-validation, Submission 3 differed from Submission 1 mainly in en-UK sarcasm (+0.0258) and hardly in en-AU sarcasm (−0.0005). On the test set it was the other way round (−0.0039 and +0.0148). By the formula of the score, the 0.0093 by which Submission 3 exceeded Submission 1 on the test set comes from the two en-AU values: 0.0074 from sarcasm and 0.0019 from sentiment. Cross-validation had not indicated a gain for en-AU sarcasm, and no member was scored alone on the test set. We therefore do not attribute the higher test score to a particular member.

6 Development Experiments

Table 4 lists the main comparisons, and Appendix D gives the others.

Changes to one encoder. We tried five variants of the loss for DeBERTa-v3-large: the three of Section 3.2, pooled class weights without the doubled sarcasm loss, and a focal loss, which gives less weight to rows that the model already classifies with confidence. Their cross-validated scores lie within 0.007 of each other (0.8547 to 0.8614), and no variant was accepted as better than another. The variant with the sarcasm loss counted twice was 0.0023 above the variant with class weights per variety. Its per-fold interval lay above zero, but the gain was below the required 0.005. Six epochs of training and a greedy blend over three runs with different seeds were not accepted either.

¹Message from the organisers to the two teams, 1 October 2026. The organisers described the outcome as provisional.

	Cross-validation		Test set				
	Decision repetitions	Check repetition	Sentiment		Sarcasm		Score
			en-AU	en-UK	en-AU	en-UK	
Submission 1	0.8719	0.8685	0.9219	0.9725	0.7945	0.8410	0.8582
Submission 2	0.8723	0.8661	0.9192	0.9750	0.8061	0.8502	0.8627
Submission 3	0.8738	0.8689	0.9257	0.9763	0.8093	0.8371	0.8675
Organisers’ baseline			0.8657	0.9311	0.6797	0.6961	0.7727

Table 3: The three submissions and the organisers’ baseline. Cross-validated scores are given as the mean of the two decision repetitions and for the check repetition; test values are from the competition platform.

Change	Change in	Per-fold change		Accepted
	score	Mean	95% interval	
<i>One encoder, change to the training (against DeBERTa with class weights per variety)</i>				
One four-class head	−0.0038	−0.0002	[−0.0057, +0.0057]	no
Pooled class weights, sarcasm loss twice*	+0.0023	+0.0079	[+0.0023, +0.0140]	no
<i>One encoder, more data (against the same DeBERTa member)</i>				
Added en-IN rows of BESSTIE	−0.0026	+0.0003	[−0.0091, +0.0084]	no
Prior fine-tuning on SARC	−0.0111	−0.0099	[−0.0168, −0.0045]	no
Added paraphrases of en-UK sarcastic rows	−0.0089	−0.0028	[−0.0103, +0.0056]	no
Prior training on comments labelled by our models	+0.0014	+0.0023	[−0.0109, +0.0126]	no
<i>Blend with a family added (against the blend without it)</i>				
Three decoders with adapters*	+0.0069	+0.0069	[+0.0026, +0.0111]	yes
Three verbalised readouts*	+0.0040	+0.0029	[+0.0019, +0.0039]	no
Three token-probability readouts*	+0.0016	+0.0017	[−0.0019, +0.0045]	no

Table 4: Comparisons made with the acceptance rule of Section 4.2. “Change in score” is the change in cross-validated score. “Mean” and the interval refer to the ten per-fold changes; this mean differs from the change in score because macro-F1 on all rows is not the average over folds. An asterisk marks predictions recomputed from the saved checkpoints (Appendix G).

More data for one encoder. Two experiments added rows to the training data: the Indian English (en-IN) rows of BESSTIE, and paraphrases of the sarcastic en-UK rows written by Qwen3.8-27B. Two added a first stage of training: on the SARC corpus of Reddit comments (Khodak et al., 2018), and on Reddit comments labelled by our own models. None was accepted. The stage on SARC lowered the score by 0.0111, with an interval below zero.

Other model families. The one change in Table 4 that was accepted added a different family of models to the blend. Adding the three decoders with adapters to the starting pool of Section 4.3 raised the cross-validated score of the blend from 0.8652 to 0.8721. Adding three verbalised readouts to the same starting pool raised it to 0.8692, a gain of 0.0040. The interval was above zero, but the gain was below the required 0.005, so the readouts were not accepted for Submission 1. Two of them are used in Submission 2. Added to the blend with decoders, the three token-probability readouts together raised the score by 0.0016 and were

not accepted. They entered Submission 3 through the separate rule of Section 4.3. These three comparisons ran the greedy search from several starting members and kept the best result. With the decoders this gives 0.8721, whereas the single search of Submission 1 gives 0.8719.

Combining the members. Submission 1 scored 0.0105 above its best single member in cross-validation (0.8719 against 0.8614). Without the decoders, the blend scored 0.8652 with the search from several starting members and 0.8626 with a single search, so the decoders account for about 0.007 to 0.009 of this difference. A numerical search over the weights, a learned combination by logistic regression, and a rule that sets the sentiment to negative whenever sarcasm is predicted were not accepted (Appendix D).

Two further decoders. After Submission 1 had been fixed, we fine-tuned two more decoders in the same way. Qwen3.8-27B reached a cross-validated score of 0.8614, but the candidate blends that contained it were not selected for Submission 3. Gemma-4-31B (Gemma Team et al., 2026)

did not pass a check, fixed in advance, that limits how much its predictions may change with the batch in which a row is scored (Appendix G), and was not considered.

7 Two Experiments Planned in Advance

Late in the shared task, we tested two further ideas for the fine-tuned encoders. For each, the design, the control and the decision rule were written down before the first result of that experiment was known. Both used DeBERTa-v3-large with a maximum length of 512 tokens. The decision rule consisted of the first and third conditions of Section 4.2, with the Benjamini-Hochberg procedure over the two experiments. Neither idea met it. The analyses made afterwards, which were not planned in advance, are in Appendix F.

Labelled neighbours in the input. Used alone, the frozen guideline readout scored higher with labelled neighbours than without them (Table 2). We tested whether neighbours also help a fine-tuned encoder. The input was extended with the four most similar labelled texts of the same variety and their labels. For each fold model, neighbours were never taken from the fold of the row itself or from the fold held out for that model, so no label of a held-out row entered an input. The control was the same model without neighbours. With neighbours, the cross-validated score fell by 0.0241 (per-fold mean -0.0234 , interval $[-0.0289, -0.0174]$).

Notes written by a language model. For every text, Qwen3.8-27B wrote a short note without seeing any label (the prompt is in Appendix B). The note says what the references in the text mean, and whether the literal wording and the apparent attitude of the writer agree. The encoder was fine-tuned with the note as a second input segment. The control was the same model without notes, trained on the same type of GPU. The cross-validated score changed by -0.0003 (per-fold mean $+0.0002$, interval $[-0.0036, +0.0041]$), so we observed no effect.

8 Notes on Reproducibility

Overlap between test and training texts. After the test set was released, a check for repeated texts found that 15 of the 1,573 test texts also occur in the released training or validation files: 14 en-UK Reddit comments and one en-AU Reddit comment.

We left the system unchanged: these rows stayed in the training data, and no rule copies a label to a test row. Two parts of the system could still use the labels of these texts. The trained members had seen the texts with their labels. The readout with neighbours takes the neighbours of a test row from all labelled rows of the same variety, so for these 15 rows the same text with its label was among the neighbours shown in the prompt. The predictions on these rows may therefore be better than on unseen text. This concerns about one percent of the test set, and 14 of the 286 en-UK Reddit test rows.

Other notes. The code, the fold assignment and the predictions of all members are available from the author on request. Appendix G describes two further points. The out-of-fold predictions were re-computed from the saved checkpoints, which have lower precision than the weights used during training. The outputs of the decoders depend on the batch in which a row is scored. One of the two checks of this dependence was changed after the test set had been released and before any submission was made.

9 Conclusion

We described a system for sentiment and sarcasm classification in Australian and British English that averages the probabilities of three families of model and applies a threshold tuned for each task, variety and source. Its best submission scored 0.8675 on the test set of the ALTA 2026 Shared Task and ranked first on the official test leaderboard. The organisers found the difference of 0.0007 to the run ranked second not statistically significant and treated the two runs as tied.

In cross-validation, the blend scored about 0.01 above its best single member, and most of this difference appeared when the decoders with adapters were added. Our other observations concern evaluation and are limited to this shared task and these data. On the 757 validation rows, the order of the three submissions differed from that on the test set. With a second seed, the cross-validated score of one encoder was 0.0051 lower, and on a different type of GPU it was 0.0043 lower; both per-fold intervals included zero (Appendices D and F).

Limitations

One test set. The test set has 1,573 rows. Resampling the cross-validation rows gave a standard deviation of about 0.0095 for one score on a set of

about this size, and of about 0.007 for the difference between two of our submissions. The largest difference between our submissions on the test set was 0.0093, and the difference between the first two runs on the leaderboard was 0.0007. We cannot say which of our submissions would be better on new data. The reported score is the highest of three test scores, which makes it an optimistic estimate for any one of them.

No test scores for components. Only three submissions were possible, so no single member and no ablation was scored on the test set. Every statement about components rests on cross-validation.

Optimism of the cross-validated scores. The blend weights were selected on the same out-of-fold predictions that give the cross-validated score. For each trained member, one training epoch was chosen for each repetition with the help of the held-out folds (Appendix A). Both make the cross-validated scores too high. For the three submissions, the score on the check repetition was 0.0034 to 0.0062 lower than on the decision repetitions, and the test score was 0.0063 to 0.0137 lower. The set of selected members was also not stable. When the search was run on each decision repetition alone, it selected different sets of members. We had set in advance a minimum Jaccard index of 0.7 between each of these sets and the selected set. This index is the number of members that two sets share, divided by the number of members that are in at least one of them. The observed values were 0.29 and 0.50, so this condition was not met. We kept the blend unchanged. Its score on the check repetition was within 0.01 of the decision repetitions, which was another condition set in advance. The pool of Submission 1 was itself the result of earlier comparisons on the same repetitions. None of its five encoders had met the acceptance rule as a single model (Table 7). The other encoders of that table were left out after comparisons and greedy searches in which they were neither accepted nor selected. The thresholds were tuned on out-of-fold probabilities, which are averages over three fold models (one per repetition), and applied to test probabilities averaged over 15 fold models. An average over 15 fold models may be distributed differently from an average over three, so these thresholds may not be the best ones for the test set.

The acceptance rule. The bootstrap interval of the rule is too narrow when two separately trained models are compared (Section 4.2). Most comparisons in Table 4 and in Section 7 are of this kind, and each configuration was trained once per repetition. Intervals that lie close to zero should be read with this in mind. The comparisons that add a family to the blend refit the greedy weights for both sides on the same ten folds that are then scored, and the side with the larger pool has more freedom. Our simulation did not cover this case, so we do not know how often such a comparison accepts a family that does not help.

Reproducing the frozen readouts. These members depend on models served by a university inference service. We do not know the public name of one of the two models used for the verbalised readouts, which appear only in Submission 2, and we do not know the numerical precision with which the service runs its models. Others may not be able to reproduce these members exactly.

Cost. The work as a whole, including experiments that are not part of a submission, logged about 234 GPU hours and stored about 78,000 responses from the inference service (Appendix H). Both figures are lower bounds. In cross-validation, the blend was about 0.01 better than one fine-tuned encoder, which trained in 30 to 50 minutes per repetition.

Data. The shared task covers two varieties of English and two sources of text. The dataset paper reports that each variety was labelled by one annotator. On 50 texts per variety that a second annotator labelled again, Cohen’s kappa was 0.47 (en-AU) and 0.63 (en-UK) for sarcasm, and 0.61 and 0.79 for sentiment. We did not study other varieties, and our observations may not carry over to them. BESSTIE is a public dataset. We cannot exclude that the pretrained language models used here saw some of its texts before our experiments, and we did not test for this. Fifteen test texts also occur in the training or validation data (Section 8).

Acknowledgements

We thank the organisers of the ALTA 2026 Shared Task for running it and for answering our questions, and the authors of BESSTIE for the dataset. Compute was provided by the National University of Singapore School of Computing, including its inference service for language models.

Large language models were used in this project only as assistance, for code and for drafting.

References

- Yoav Benjamini and Yosef Hochberg. 1995. [Controlling the false discovery rate: A practical and powerful approach to multiple testing](#). *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 57(1):289–300.
- Rich Caruana, Alexandru Niculescu-Mizil, Geoff Crew, and Alex Ksikes. 2004. [Ensemble selection from libraries of models](#). In *Twenty-First International Conference on Machine Learning (ICML '04)*. ACM Press. Article 18.
- Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. [M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2318–2335, Bangkok, Thailand. Association for Computational Linguistics.
- Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. [ELECTRA: Pre-training text encoders as discriminators rather than generators](#). In *International Conference on Learning Representations (ICLR 2020)*.
- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. [Unsupervised cross-lingual representation learning at scale](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.
- Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. [QLoRA: Efficient fine-tuning of quantized LLMs](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 10088–10115. Curran Associates, Inc.
- 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, Minneapolis, Minnesota. Association for Computational Linguistics.
- Bradley Efron and R. J. Tibshirani. 1994. [An Introduction to the Bootstrap](#). Chapman and Hall/CRC.
- Gemma Team et al. 2026. [Gemma 4 technical report](#). *Preprint*, arXiv:2607.02770.
- Noah D. Goodman and Michael C. Frank. 2016. [Pragmatic language interpretation as probabilistic inference](#). *Trends in Cognitive Sciences*, 20(11):818–829.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. 2024. [The Llama 3 herd of models](#). *Preprint*, arXiv:2407.21783.
- Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. [DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing](#). In *International Conference on Learning Representations (ICLR 2023)*.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. [LoRA: Low-rank adaptation of large language models](#). In *International Conference on Learning Representations (ICLR 2022)*.
- Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. 2018. [A large self-annotated corpus for sarcasm](#). In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).
- Joshua Lee, Wyatt Fong, Alexander Le, Sur Shah, Kevin Han, and Kevin Zhu. 2025. [Pragmatic metacognitive prompting improves LLM performance on sarcasm detection](#). In *Proceedings of the 1st Workshop on Computational Humor (CHum)*, pages 63–70, Online. Association for Computational Linguistics.
- Zachary C. Lipton, Charles Elkan, and Balakrishnan Narayanaswamy. 2014. [Optimal thresholding of classifiers to maximize F1 measure](#). In *Machine Learning and Knowledge Discovery in Databases*, pages 225–239. Springer Berlin Heidelberg.
- Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. [What makes good in-context examples for GPT-3?](#) In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online. Association for Computational Linguistics.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. [RoBERTa: A robustly optimized BERT pretraining approach](#). *Preprint*, arXiv:1907.11692.
- Sanjh Maheshwari, Aniket Singh Rajpoot, Oana Carascu, and Mamta. 2026. [Improving dialect robustness in large language models via LoRA and mixture-of-experts](#). In *Proceedings of the 13th Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 293–303, Rabat, Morocco. Association for Computational Linguistics.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, Benjamin Bossan, and Marian Tietz. 2022. PEFT: State-of-the-art parameter-efficient fine-tuning methods. <https://github.com/huggingface/peft>.

Meta Llama. 2024. [Llama-3.3-70B-Instruct](#). Hugging Face model card, meta-llama/Llama-3.3-70B-Instruct.

Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. [Noisy channel language model prompting for few-shot text classification](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5316–5330, Dublin, Ireland. Association for Computational Linguistics.

Mistral AI. 2024. [Mistral-Small-Instruct-2409](#). Hugging Face model card, mistralai/Mistral-Small-Instruct-2409.

Diego Mollá, Dipankar Srirag, and Aditya Joshi. 2026. Overview of the 2026 ALTA shared task: Sentiment and sarcasm classification for Australian and British English. In *Proceedings of the 24th Annual Workshop of the Australasian Language Technology Association (ALTA 2026)*, Melbourne.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, and 2 others. 2019. [PyTorch: An imperative style, high-performance deep learning library](#). In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. [Scikit-learn: Machine learning in Python](#). *Journal of Machine Learning Research*, 12(85):2825–2830.

Qwen. 2026. [Qwen3.8-27B](#). Hugging Face model card, Qwen/Qwen3.8-27B.

Ishmanbir Singh, Dipankar Srirag, and Aditya Joshi. 2025. [Nek minit: Harnessing pragmatic metacognitive prompting for explainable sarcasm detection of Australian and Indian English](#). In *Proceedings of the 23rd Annual Workshop of the Australasian Language Technology Association*, pages 13–27, Sydney, Australia. Association for Computational Linguistics.

Dipankar Srirag, Aditya Joshi, Jordan Painter, and Diptesh Kanojia. 2025. [BESSTIE: A benchmark for sentiment and sarcasm classification for varieties of English](#). In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 8413–8429,

Vienna, Austria. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. [Transformers: State-of-the-art natural language processing](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, et al. 2025. [Qwen3 technical report](#). *Preprint*, arXiv:2505.09388.

A Settings of the Trained Members

Setting	Value
<i>Encoders (all five members)</i>	
Epochs	4
Batch size	16
Maximum length (tokens)	256
Learning rate, encoder / heads	$1 \times 10^{-5} / 1 \times 10^{-4}$
Layer-wise learning rate decay	0.9
Weight decay	0.01
Warm-up share, linear decay	0.1
Gradient clipping (norm)	1.0
Dropout before the heads	0.1
Pooling	mean over tokens
<i>Decoders with adapters (all three members)</i>	
Epochs	3
Batch size $\times$ accumulation steps	8×2
Maximum length (tokens)	256
Adapter rank / alpha / dropout	64 / 128 / 0.05
Adapted projections	q, k, v, o, gate, up, down
Quantisation	4-bit NF4, double quantisation
Pooling	last token
Gradient clipping (norm)	1.0
Weight decay	0
Learning rate, adapters / heads	$1 \times 10^{-4} / 5 \times 10^{-4}$
for Mistral-Small	$5 \times 10^{-5} / 2 \times 10^{-4}$
Warm-up share, cosine decay	0.05
for Mistral-Small	0.1

Table 5: Training settings of the trained members.

Table 5 gives the training settings. All trained members use the AdamW optimiser, and the forward and backward passes in training use 16-bit floating-point arithmetic. The class weights are the inverse class frequencies of the training fold, scaled to a mean of one and limited to at most ten. For each member and each repetition, one epoch is chosen for all five fold models. On each held-out fold and for every epoch, we take the lower of the two sarcasm macro-F1 values at a threshold of

0.5. The epoch with the highest mean of this value over the five folds is chosen. The last epoch was chosen in all cases except two, both for DeBERTa members, where it was the third of four. Each fold model has its own random seed, and no further averaging over seeds is done inside a member.

Retrained fold models of the decoders. Of the 45 fold models of the three decoders, 8 were re-trained with a different seed under the rule of Section 3.3: 6 of Llama-3.3-70B-Instruct, 2 of Mistral-Small-Instruct-2409 and none of Qwen3-32B. Besides the test on the second epoch, the rule also retrains a fold model whose mean training loss in the first epoch is above 1.5. The value 0.6 was chosen from 37 completed fold models, of which 7 had diverged; it separated the two groups. Two fold models that were still training when the value was fixed were named in advance as a check, and the rule later classified both correctly. The 37 fold models include those of an earlier version of the Mistral-Small member, trained with the settings of the other two decoders, in which four of the five fold models of one repetition had diverged. That version is not part of the system. For the member that is part of the system, the learning rates were lowered and the warm-up was lengthened (Table 5).

Software. PyTorch 2.13.0 (Paszke et al., 2019), Transformers 5.14.1 (Wolf et al., 2020), PEFT 0.20.0 (Mangrulkar et al., 2022) and bitsandbytes 0.50.2 were used for training and inference. Scores are computed with the organisers' script, which uses scikit-learn (Pedregosa et al., 2011).

B Prompts of the Frozen Readouts

The prompts are reproduced word for word. For the token-probability readouts, the user message holds the variety and the source in square brackets, followed by the text. Texts longer than 4,000 characters are cut, and each neighbour text is cut to 600 characters. All requests with these prompts use a temperature of zero.

Verbalised readout (system message).

You are an exact annotator for a dialectal English sentiment and sarcasm benchmark. The rows are en-AU (Australian) and en-UK (British) English.

Sarcasm is saying the opposite of what is meant, or mocking and ironic exaggeration used to criticise. A

sarcastic line usually reads as praise or as a flat statement of fact while the writer clearly intends the reverse, so the literal sentiment and the intended sentiment disagree.

The rows come from two registers and they differ. A google row is a customer review of a place or a business: mostly literal, sarcasm is rare and when it appears it is heavy mock praise ("great, another hour in the queue"). A reddit row is a comment on a public thread about politics, sport or news: conversational, elliptical, often replying to something you cannot see, and sarcasm is common and dry. Judge sentiment as the writer's attitude to what they are talking about, not as the tone of the words alone.

For every row give two numbers from 0 to 100.

sentiment = the probability in percent that the row is POSITIVE overall (0 means certainly negative, 100 means certainly positive, 50 means genuinely undecided).

sarcasm = the probability in percent that the row is SARCASTIC (0 means certainly not sarcastic, 100 means certainly sarcastic).

The two are independent: a sarcastic row is usually negative in intent even when its words look positive.

Use the whole 0 to 100 range. Do not round everything to 0, 50 or 100.

Answer with a JSON array and nothing else. One object per row, in the order given, exactly:

```
[{"id": <the row id>, "sentiment": <0-100>, "sarcasm": <0-100>}, ...]
```

No prose, no markdown fence, no explanation, one object for every id you were given.

Token-probability readout, direct prompt (system message).

You annotate Australian (en-AU) and British (en-UK) English texts for sentiment and sarcasm.

Sarcasm is saying the opposite of what is meant, or mocking and ironic exaggeration used to criticise. A sarcastic line usually reads as praise or as a flat statement of fact while the writer clearly intends the reverse, so the literal sentiment and the intended sentiment disagree.

The texts come from two registers. A google text is a customer review of a place or a business: mostly literal, sarcasm is rare. A reddit text is a comment on a public thread: conversational, often replying to something you cannot see, and sarcasm is common and dry. Judge sentiment as

the writer's intended attitude to what they are talking about.

Answer with exactly these two lines and nothing else:

sarcastic: yes or no

sentiment: positive or negative

Token-probability readout, guideline prompt (system message).

You annotate Australian (en-AU) and British (en-UK) English Google reviews and Reddit comments following these annotation guidelines.

For Sentiment Classification:

- Positive Sentiment: Annotate a comment as positive if it reflects favorable emotions such as happiness, satisfaction, agreement, or excitement.
- Negative Sentiment: Annotate a comment as negative if it conveys unfavorable emotions such as sadness, disappointment, frustration, or criticism.

For Sarcasm Classification:

- Sarcastic Comment: Annotate if the comment uses irony or mockery to express contempt or ridicule.
- Non-Sarcastic Comment: Annotate if the comment is straightforward and does not employ sarcasm or irony.

The texts come from two registers. A google text is a customer review of a place or a business: mostly literal, sarcasm is rare. A reddit text is a comment on a public thread: conversational, often replying to something you cannot see, and sarcasm is common and dry. Judge sentiment as the writer's intended attitude to what they are talking about.

Answer with exactly these two lines and nothing else:

sarcastic: yes or no

sentiment: positive or negative

Guideline with neighbours (text placed before the neighbours in the user message).

Below are labelled examples annotated by the same annotator under these guidelines, the most similar texts first. Use them to calibrate how this annotator applies the guidelines.

Notes for the experiment of Section 7 (system message).

You help a text classifier that reads short English texts written in Australia or the United Kingdom (Google reviews and Reddit comments). Write a note of at most 60 words in exactly two lines.

Task	Variety	Source	Sub. 1	Sub. 2	Sub. 3
Sentiment	en-AU	Google	0.625	0.583	0.680
Sentiment	en-AU	Reddit	0.449	0.477	0.451
Sentiment	en-UK	Google	0.370	0.520	0.460
Sentiment	en-UK	Reddit	0.676	0.515	0.559
Sarcasm	en-AU	Google	0.462	0.337	0.346
Sarcasm	en-AU	Reddit	0.440	0.422	0.410
Sarcasm	en-UK	Google	0.424	0.564	0.431
Sarcasm	en-UK	Reddit	0.424	0.564	0.431

Table 6: Decision thresholds used for the test set. A text receives the positive label when the blended probability is at least the threshold. For en-UK sarcasm, the Google cell has too few sarcastic rows to be tuned and uses the threshold of the variety. The adjustment of the Reddit cell left its value unchanged.

Context: explain any named people, parties, places, events, brands, slang or in-jokes that a reader from another country might not know, and what they imply here. Write 'Context: none' if there are none.

Tone: say whether the literal wording and the writer's evident attitude point the same way or opposite ways, naming the cue if there is one (exaggeration, mock praise, a claim that contradicts well-known facts, a rhetorical question, a mocking quote). Write 'Tone: literal' if they point the same way.

Do not use the words sarcasm, sarcastic or ironic, and do not add anything else.

C Thresholds of the Three Submissions

Table 6 gives the thresholds. They were tuned on the out-of-fold predictions of all 6,288 labelled rows, averaged over the three repetitions. The adjustment of the two source thresholds (Section 3.6) is run from two starting values. The first is the threshold that maximises the macro-F1 on all rows of the variety. The second is the threshold that maximises the F1 of the positive class on those rows. The result with the higher macro-F1 of the variety is kept.

D All Comparisons

Table 7 lists the comparisons of members and of model families that were made with the acceptance rule. It leaves out two sets of comparisons. The first is an earlier run of the comparison of the verbalised readouts with ten texts per request. That run included members that were later excluded, and the table gives the repeated run without them. The second set consists of the comparisons of ways to combine the members, which are

Comparison	Change in	Per-fold change		<i>p</i>	Accepted
	score	Mean	95% interval		
<i>Single models, against DeBERTa-v3-large with class weights per variety</i>					
Pooled class weights [†]	−0.0045	−0.0009	[−0.0069, +0.0071]	0.6281	no
One four-class head	−0.0038	−0.0002	[−0.0057, +0.0057]	0.5433	no
Focal loss	−0.0033	+0.0021	[−0.0040, +0.0088]	0.2665	no
Pooled class weights, sarcasm loss twice	−0.0001	+0.0049	[−0.0021, +0.0121]	0.0867	no
Pooled class weights, sarcasm loss twice*	+0.0023	+0.0079	[+0.0023, +0.0140]	0.0009	no
Six epochs	−0.0079	−0.0039	[−0.0099, +0.0020]	0.8948	no
A second seed	−0.0051	−0.0002	[−0.0040, +0.0036]	0.5346	no
Greedy blend over three runs with different seeds	+0.0010	+0.0037	[+0.0005, +0.0074]	0.0111	no
RoBERTa-large	−0.0252	−0.0189	[−0.0244, −0.0130]	1.0000	no
XLNet-large	−0.0433	−0.0370	[−0.0468, −0.0269]	1.0000	no
Multilingual BERT	−0.0907	−0.0850	[−0.0952, −0.0759]	1.0000	no
ELECTRA-large	−0.0201	−0.0180	[−0.0267, −0.0100]	1.0000	no
Qwen3-32B with adapters	−0.0057	−0.0035	[−0.0114, +0.0036]	0.8220	no
Mistral-Small with adapters	−0.0021	+0.0033	[−0.0037, +0.0104]	0.1854	no
Added en-IN rows of BESSTIE	−0.0026	+0.0003	[−0.0091, +0.0084]	0.4527	no
Prior fine-tuning on SARC	−0.0111	−0.0099	[−0.0168, −0.0045]	1.0000	no
Added paraphrases of en-UK sarcastic rows	−0.0089	−0.0028	[−0.0103, +0.0056]	0.7547	no
Prior training on comments labelled by our models	+0.0014	+0.0023	[−0.0109, +0.0126]	0.3321	no
Rule: sarcastic implies negative	−0.0003	−0.0003	[−0.0012, +0.0006]	0.7449	no
<i>Blends with a family added, against the blend without it</i>					
ELECTRA-large	+0.0013	−0.0007	[−0.0037, +0.0027]	0.6627	no
Qwen3-32B with adapters	+0.0021	+0.0021	[−0.0026, +0.0068]	0.1820	no
Three models trained with more data	−0.0009	−0.0009	[−0.0023, +0.0007]	0.8717	no
Verbalised readouts, ten texts per request*	+0.0077	+0.0078	[+0.0030, +0.0131]	0.0000	yes
Four further verbalised readouts	+0.0040	+0.0034	[−0.0007, +0.0076]	0.0521	no
Five further verbalised readouts	+0.0030	+0.0020	[−0.0016, +0.0051]	0.1271	no
Verbalised readouts, one text per request*	+0.0040	+0.0029	[+0.0019, +0.0039]	0.0000	no
Two decoders with adapters*	+0.0056	+0.0035	[−0.0001, +0.0069]	0.0283	
Three decoders with adapters*	+0.0069	+0.0069	[+0.0026, +0.0111]	0.0010	yes
Readout, direct prompt*	+0.0009	+0.0009	[−0.0003, +0.0019]	0.0699	no
Readout, guideline prompt*	+0.0014	+0.0013	[−0.0019, +0.0042]	0.2068	no
Readout, guideline with neighbours*	+0.0011	+0.0012	[−0.0019, +0.0041]	0.2067	no
Three token-probability readouts*	+0.0016	+0.0017	[−0.0019, +0.0045]	0.1546	no
<i>Other comparisons</i>					
Submission 1 against its best member*	+0.0105	+0.0091	[+0.0053, +0.0132]	0.0000	
Labelled neighbours in the encoder input	−0.0241	−0.0234	[−0.0289, −0.0174]	1.0000	no
Notes in the encoder input	−0.0003	+0.0002	[−0.0036, +0.0041]	0.4456	no
Same training on H100 against A100 GPUs	−0.0043	−0.0046	[−0.0091, +0.0008]	0.9546	

Table 7: Comparisons of members and of model families made with the acceptance rule. The columns are those of Table 4; *p* is the share of bootstrap means that are zero or below. Rows marked with an asterisk use the predictions recomputed from the saved checkpoints (Appendix G); the other rows use the predictions written during training. The three rows with an empty last column are descriptive and were not decisions about a component. The row marked with a dagger was tested with the two sides exchanged, that is, with class weights per variety as the candidate, which was not accepted (*p* = 0.3719). It is shown with the signs reversed, and its *p* is one minus that value.

described at the end of this appendix. Only one of them has a row in the table: the rule that sets the sentiment to negative whenever sarcasm is predicted, applied to one encoder.

Some rows concern members that are not part of any submission and are not described above. Multilingual BERT in its base size (Devlin et al., 2019) and ELECTRA-large (Clark et al., 2020) were fine-tuned like the other encoders, except that multilingual BERT used an encoder learning rate of 2×10^{-5} . The three models trained with more data are the encoders with en-IN rows, with prior

fine-tuning on SARC and with prior training on comments labelled by our models. The further verbalised readouts are verbalised readouts of other models served by the inference service and, for two members, readouts with the six most similar labelled texts in the prompt. The rows on readouts with a named prompt refer to the token-probability readouts. The row on two decoders with adapters adds Qwen3-32B and Mistral-Small to the starting pool. It was computed for information before the third decoder was complete.

The verbalised readouts with ten texts per re-

quest passed the rule, but they were not used, because a test set cannot be scored in the same batches as the cross-validation data (Appendix G). The same readouts with one text per request did not pass it. The two comparisons used different reference blends, so their changes in score are not directly comparable. The first used a single greedy search for both blends (0.8703 against 0.8626). The second used the search from several starting members (0.8692 against 0.8652). In five rows of the second group of Table 7, the pools of both blends held the readouts with ten texts per request: the rows on ELECTRA-large, on Qwen3-32B, on the models trained with more data and on the further verbalised readouts.

The blend without decoders that Section 6 quotes at 0.8652 gives one of its four units of sentiment weight to the TF-IDF classifier. The blend at 0.8626 consists of two encoders.

Ways of combining the members. These comparisons were made on an earlier pool of eight encoders and the TF-IDF classifier. A numerical search over the weights added 0.0042 when it was fitted on both decision repetitions. When it was fitted on one decision repetition and scored on the other, it added 0.0027 on the repetition used for fitting and 0.0005 on the other. These values are means over the two ways of assigning the repetitions; in one of the two, the change on the repetition not used for fitting was negative. On the same pool, a learned combination by logistic regression scored below the greedy procedure (0.8588 against 0.8626). For Submission 1 itself, a second numerical search, which adjusts only the weights of the selected members, added 0.0004. A rule that sets the sentiment to negative whenever sarcasm is predicted changed the score by -0.0003 for one encoder and by -0.0016 for Submission 1, and both intervals included zero.

E Pilot Study of the Frozen Readouts

Before any token-probability readout was run on all rows, three ways of reading out Qwen3.8-27B without training were compared on 720 labelled rows. The rows were drawn with fixed numbers per cell and weighted back to the proportions of the data. The rule for continuing with a method was fixed before the pilot. For each variety, two logistic regressions for sarcasm are fitted: one on the probability of Submission 1 alone, and one that also receives the output of the method. The rule

compares their cross-validated log loss. The 90 percent bootstrap interval of the reduction must lie above zero in one variety, and the log loss must not rise in the other. The rule could also be met by a higher area under the ROC curve (AUC) for sarcasm than that of the verbalised readout of the same model, by at least 0.02 in both varieties. The AUC is 1 for a perfect ranking of the rows and 0.5 for a random one. No method met this second condition.

Token-probability readout. The direct readout of Section 3.4 met the rule. The log loss fell by 0.0134 for en-UK (interval $[+0.0047, +0.0224]$) and by 0.0064 for en-AU (interval $[-0.0045, +0.0167]$).

Reasoning prompt. We also tried pragmatic metacognitive prompting (Lee et al., 2025). The prompt was adapted from the form that Singh et al. (2025) used for BESSTIE, so that it runs on every text and answers both tasks. The reasoning mode of the model was switched on, and four answers were sampled per text. The reasoning prompt met the rule. The log loss fell by 0.0133 for en-UK (interval $[+0.0034, +0.0231]$), and the en-AU result was at the boundary ($+0.0001$, interval $[-0.0081, +0.0081]$). In a further comparison that was not part of the rule, the output of the reasoning prompt was added to a logistic regression that already received the direct readout. For en-UK the further reduction could not be distinguished from zero ($+0.0010$, interval $[-0.0035, +0.0051]$), and for en-AU the log loss rose (by 0.0035, interval of the reduction $[-0.0057, -0.0014]$). We did not run the reasoning prompt on all rows.

Listener model. In the third method the frozen model gives the probability of the text under each of three intentions of the writer: sincere and positive, sincere and negative, or sarcastic. This follows the idea of a pragmatic listener (Goodman and Frank, 2016); scoring the text given the label is also the idea of noisy channel prompting (Min et al., 2022). In the pilot, the score was the log probability of the text under the sarcastic intention minus that under the two sincere intentions combined. This method did not meet the rule. The AUC of this score for sarcasm was 0.57 for en-AU and 0.68 for en-UK, and we dropped the method.

The guideline prompt. In the pilot, 117 of the 160 sarcastic en-AU rows were called not sarcastic by both the direct readout and the reasoning

prompt. The annotation guideline of BESSTIE describes a sarcastic comment as one that “uses irony or mockery to express contempt or ridicule”, which is broader than the definition in our first prompts. This led to the guideline prompt. Because it was written after we had read rows of the pilot, it was tested on a second sample of 712 rows, none of which were in the pilot. On this sample it raised the AUC of the readout for en-AU sarcasm from 0.837 to 0.857, and adding neighbours raised it to 0.886. On the same sample, the gain of the direct readout over Submission 1 for en-UK was smaller than in the pilot, and its interval included zero (+0.0052, interval $[-0.0028, +0.0136]$).

Amendments to the rule for Submission 3.

The rule for Submission 3 (Section 4.3) was amended twice. The first amendment, on 17 September 2026, was written before the pilot had produced any result. The new decoders were still training the first fold model of each decision repetition, and only held-out scores of single epochs of these fold models were known. The amendment added the candidate that consists of Submission 1 with readouts added. The second amendment, on 18 September, was written after the pilot results for the direct and the guideline readout on the two samples above were known. It kept the guideline readout for a run on all rows, although this readout had not met the condition set for replacing the direct readout. It also admitted every readout that met a condition on its own cross-validated score, whatever the result of its acceptance test.

F Further Analyses of the Two Experiments

The first two analyses in this appendix were made after the results of Section 7 were known and were not planned in advance. The third comparison was made after the first of these results and before the second.

Labels of the neighbours. The labels of the four neighbours alone predict sarcasm less well than the control model does. Measured with the AUC, the neighbour labels reach 0.77 against 0.89 for en-AU, and 0.73 against 0.94 for en-UK (means over the two decision repetitions). On the rows whose four neighbours all carried the wrong label, the model with neighbours predicted that wrong label in 41.9 and 45.5 percent of the cases in the two decision repetitions. The control, which does

not see the neighbours, did so in 39.1 and 39.5 percent. One reading is that the fine-tuned model already holds the information that the neighbours offer, and that their labels add noise. We did not test this reading further.

Use of the notes. On the 187 rows whose note describes the tone as not literal, the predicted probability of sarcasm was 0.11 to 0.12 higher with notes than under the control, against 0.02 on the other rows. The AUC for sarcasm changed by at most 0.006 in either variety.

The same training on two types of GPU. The second experiment had to run on H100 GPUs. Before any of its results was known, we recorded that its control would be trained again on the same type of GPU, with the same code and settings as the control that had been trained on A100 GPUs. The two controls differed by 0.0043 in cross-validated score (per-fold mean -0.0046 , interval $[-0.0091, +0.0008]$). The interval includes zero, so we cannot say that the type of GPU matters. The difference is, however, close to the required gain of 0.005. This comparison is a by-product of the matched control and was not planned as a test.

G Reproducibility Details

Precision of saved weights. During training, the out-of-fold predictions were computed with 32-bit weights, while the checkpoints were saved in 16-bit precision. For one encoder run, predictions recomputed from the checkpoints differed from the stored ones by up to 0.17 for single rows. The label at a threshold of 0.5 changed for 0.05 percent of the rows for sentiment and for 0.11 percent for sarcasm, and the score at that threshold changed by 0.0004. With tuned thresholds the change can be larger. For the DeBERTa member with the doubled sarcasm loss, the cross-validated score was 0.8588 on the predictions written during training and 0.8614 on the recomputed ones, and Table 7 gives its comparison on both sets of predictions. Over all trained members, the largest difference for a single row was 0.32. Test predictions can only be made from the saved checkpoints. We therefore recomputed the out-of-fold predictions of all trained members from the checkpoints and used these for the thresholds, the blends and the choice of submissions.

Dependence on the batch. For the decoders with adapters, the probability of a row depends on

	Jobs	Hours
Fine-tuning encoders	47	28.4
Fine-tuning decoders with adapters	39	169.1
Encoders trained with more data	21	9.7
Inference and checks	70	26.9
Other	16	0.8
Total	193	234.9

Table 8: Logged hours by kind of job. Each job in the first four rows used one GPU. Of the 16 other jobs, 12 prepared data on nodes without a GPU (0.75 hours). We do not count the 0.8 hours of this row as GPU time. Of the 193 jobs, 74 did not log their duration, 58 of them inference jobs, so the total is a lower bound.

the batch in which it is scored: on the padding that the other rows cause and on the number of rows in the batch. Before a decoder was used, two passes with different batch sizes had to agree to within 0.002 in mean absolute difference for each task. Gemma-4-31B exceeded this tolerance for sarcasm (0.00238 and 0.00205 in the two decision repetitions) and was not used.

A second check compared two ways of padding the same 16 rows, with a tolerance of 0.001. When the test set was scored, this check stopped the scoring for two of the three repetitions of Qwen3-32B (0.0014 and 0.0027 for sarcasm). The value from one draw of 16 rows varied between runs by as much as the tolerance. Before any submission was made, we changed the check to the median over nine draws, with the same tolerance, and all repetitions then passed. This change was made after the test set had been released. By then the median had been measured for four of the nine combinations of decoder and repetition, two of them of Qwen3-32B, and all four were below the tolerance.

For the verbalised readouts, sending ten texts per request instead of one text per request changed the sarcasm label for 5.3 and 11.3 percent of 300 sampled rows for the two models. The frozen readouts of the three submissions and the token-probability readouts therefore use one request per text. The further verbalised readouts of Table 7 were scored with five or ten texts per request.

H Compute

Table 8 gives the logged hours. The GPU jobs ran on NVIDIA A100 GPUs with 40 or 80 GB of memory and on NVIDIA H100 GPUs. The inference service returned 77,925 distinct responses that we stored. Of these, 50,156 belong to the five

frozen members that are part of a submission, including the responses for the test set. Repeated requests and the requests for embeddings are not counted.