Evaluating the fairness of task-adaptive pretraining on unlabeled test data before text classification

Anonymous ACL submission

Abstract

Few-shot learning benchmarks are critical for evaluating modern NLP techniques. But it's possible that benchmarks favor methods which easily make use of unlabeled text, because researchers can pretrain their models on unlabeled text *from the test set*. Given the dearth of research on this potential problem, we run experiments to quantify the bias caused by pretraining on unlabeled test set text instead of on unlabeled, independently drawn text. Controlled few-shot and zero-shot experiments on 25 classification tasks and 3 language models—BERT, GPT-2, and Mistral 7B—does not find evidence of overoptimism. Furthermore, we demonstrate the importance of repeated subsampling when studying few-shot text classification, and recommend that few-shot learning benchmarks include multiple training folds. Code and data are available here: `https://github.com` (currently omitted for anonymity).

1 Introduction

For NLP benchmarks, it's standard to release text from the test set. This allows researchers to submit a file of predictions instead of submitting code. A potential concern is that researchers can use this text during training. Consider the Real-world Annotated Few-shot Tasks (RAFT) benchmark (Alex et al., 2021), which contains "few-shot" text classification tasks—tasks where the training set contains a relatively small number of labeled examples. Below is an excerpt from the RAFT paper (emphasis added):

> For each task, we release a public training set with 50 examples and a larger unlabeled test set. *We encourage unsupervised pre-training on the unlabelled examples* and open-domain information retrieval.

In the RAFT competition, a model is evaluated by scoring its predictions on the same set of unlabeled text which the model may have been trained on (using an unsupervised training procedure).

It's wrong to train a model on test set features with their labels and then evaluate on the test set when one needs to estimate performance on out-of-sample data. Test set performance would be overoptimistic (Hastie et al., 2009). This fact is widely known. But what if, as encouraged by Alex et al. (2021), a model is trained on test set features *without* test set labels? This paper studies this question for the domain of few-shot text classification.

2 Motivation

NLP benchmarks for few-shot learning are prevalent, as having only a handful of labeled examples is more realistic. One consideration when designing these benchmarks is that some few-shot approaches can—at least theoretically—use unlabeled text from the test set. With Pattern-Exploiting Training (Schick and Schütze, 2021), for example, one can train the final classifier on test set text with soft labels predicted by an ensemble of supervised models. Or, with Pre-trained Prompt Tuning (Gu et al., 2022), one can pretrain the language model (LM) on unlabeled test set text before prompt-tuning on the labeled training set. A more classical approach would be to train a word2vec model (Mikolov et al., 2013) on unlabeled test set text, run this model on training text to get embeddings, and finally train a classifier on these embeddings with labels from the training set.

For other few-shot approaches, such as SetFit (Tunstall et al., 2022) and in-context learning with LLMs (as popularized by Brown et al., 2020), it's more common to only use labeled text.

While the ability to exploit unlabeled text is useful, doing so on test set text could be substantively different than applying it to text which is statistically independent of the test set. This difference in methodology may be more concerning in the few-shot setting than in the many-shot setting. It's conceivable that differences between few-shot methods are due just as much to how unlabeled text is used as they are to how the few, labeled examples are used. This raises the question: does pretraining a model on a benchmark's unlabeled test set text inflate the model's performance on that benchmark?

3 Related work

As indicated by the quote in §1, the RAFT benchmark implicitly assumes that the answer is no. The validity of using test set features is not a fringe opinion. The popular textbook by Hastie et al. (2009) contains the following passage without a reference or evidence (emphasis added):

> There is one qualification: *initial unsupervised screening steps can be done before samples are left out.* For example, we could select the 1000 predictors with highest variance across all 50 samples, before starting cross-validation. *Since this filtering does not involve the class labels, it does not give the predictors an unfair advantage.*

The opposite opinion—that exploiting unlabeled test set features is unfair—may align more closely with best practices. For example, Gururangan et al. (2020) contains the following criticism of another study when comparing performances on a popular text classification benchmark:

> Thongtan and Phienthrakul (2019) report a higher number (97.42) on IMDB, but they train their word vectors on the test set.

Jacovi et al. (2023) argue that benchmarks which release unlabeled test set text can be compromised in two ways. First, the text can be discretely labeled and trained on by model developers. Second, if test set texts were scraped from websites which also host their labels, e.g., movie reviews and ratings on IMDb, then an LLM may have already been pretrained on these labeled texts (Scenario 1 in Jacovi et al., 2023). They do not discuss potential problems with using unlabeled test set text by itself.

Moscovich and Rosset (2022) contains experiments and theory for unsupervised methods which are common to tasks involving tabular data. They find that estimators of out-of-sample performance which were subject to these methods may be biased positively or negatively, depending on the parameters of the problem. They recommend further research on this bias in more domains, particularly when dealing with small sample sizes and high-dimensional data.

4 Experimental design

In the absence of theory or experiments on this question in NLP, we study whether pretraining on unlabeled test set text biases test set performance for 25 diverse text classification tasks and two types of LMs: BERT (Devlin et al., 2019), and GPT-2 (Radford et al., 2019). Appendix A describes each task.

At a high level, the goal of the experiment is to first establish that pretraining is beneficial, in line with Gururangan et al. (2020). Second, given that pretraining has a detectable benefit, the experiment measures the accuracy difference between using test set text for the pretraining stage—an arguably unfair methodology—and using text which is independent of the test set—an inarguably fair methodology.

In more detail, the experiment starts by subsampling three separate sets of data from the full sample of data for a given text classification task:

- `extra`: n (either $50, 100, 200$ or 500) unlabeled texts which are optionally used for pretraining
- `train`: m (either 50 or 100) labeled texts for classification training
- `test`: n labeled texts to report accuracy.

Next, three accuracy estimators are computed. The procedures used to obtain them are described below.

4.1 $\mathbf{acc}_{\mathbf{extra}}$

1. Train a freshly loaded, pretrained LM on the n unlabeled texts in `extra` using the LM's pretraining objective—masked language modeling loss for BERT, or autoregressive/causal language modeling loss for GPT-2.
2. Add a linear layer to this model, and finetune all of the LM's weights to minimize classification cross entropy loss on `train`.
3. Compute the classification accuracy of this model on `test`.

Step 1 is task-adaptive pretraining—a procedure broadly recommended by Gururangan et al. (2020). Step 2 is a canonical way to train a transformer-based LM for a classification task, according to Section 2 of Zhang et al. (2021).

$\text{acc}_{\text{extra}}$ is clearly an unbiased estimator of out-of-sample accuracy because it never trains on `test`.

4.2 acc_{test}

acc_{test} is identical to $\text{acc}_{\text{extra}}$, except that pretraining is done on `test` instead of `extra` in step 1.

acc_{test} represents what one might see in a competition like RAFT, where pretraining on unlabeled text from `test` is encouraged. It's unclear whether this accuracy estimator is unbiased, because it was (pre)trained and evaluated on the same set of test set text. A reasonable hypothesis is that it's overoptimistic, i.e., $\text{E}[\text{acc}_{\text{test}}] > \text{E}[\text{acc}_{\text{extra}}]$.

4.3 acc_{base}

acc_{base} doesn't do pretraining; it doesn't make any use of unlabeled text. It simply trains a pretrained LM on `train` to do classification, and then computes this model's accuracy on `test`.

This score is a control. If there's no boost from acc_{base} to $\text{acc}_{\text{extra}}$, then it shouldn't be surprising that there's no difference between $\text{acc}_{\text{extra}}$ and acc_{test}.

4.4 Repeated subsampling

The three accuracy estimators are paired, because their classification training and test sets are identical. The only difference is the source of unlabeled text for pretraining. For $\text{acc}_{\text{extra}}$, the source is independent of test data. For acc_{test}, the source is exactly the test set text. For acc_{base}, no unlabeled text is used.

A potentially important source of variation in this experiment is the particular subsamples, i.e., the particular realizations of `extra`, `train`, and `test` for a given classification task. To expose this variation, the experiment procedure is repeated tens of times for each classification task.[1] For example, for $n = 200$, and for each of the 25 classification tasks, 50 $(\text{acc}_{\text{extra}}, \text{acc}_{\text{test}}, \text{acc}_{\text{base}})$ triples are computed.

Appendix B explains more experiment choices.

[1] For $n = 50$ and $n = 100$, the experiment is repeated 100 times. For $n = 200$, the experiment is repeated 50 times. For $n = 500$, the experiment is repeated 20 times. In total, $81,000$ finetuned BERT and GPT-2 models were evaluated in this experiment.

5 Results

Figure 1 visualizes the distributions of $\text{acc}_{\text{extra}} - \text{acc}_{\text{base}}$ and $\text{acc}_{\text{test}} - \text{acc}_{\text{extra}}$ for $m = 50, n = 200$. $\text{acc}_{\text{extra}} - \text{acc}_{\text{base}}$ is a control—it's the accuracy boost from pretraining on unlabeled independent text versus not pretraining at all. $\text{acc}_{\text{test}} - \text{acc}_{\text{extra}}$ is the main quantity of interest: it's the evaluation bias from pretraining on unlabeled test set text instead of on unlabeled independent text.

Table 1 contains means of these differences for each configuration of the experiment. It roughly suggests that while pretraining is consistently beneficial, pretraining on unlabeled test set text does not bias test set performance one way or the other.

A more complete analysis of this data is motivated and performed in the next section.

	BERT	GPT-2
$n = 50$	4.1% / 0.19%	3.8% / 0.18%
$n = 100$	3.9% / 0.18%	4.1% / 0.11%
$n = 200$	3.9% / -0.39%	4.4% / -0.05%
$n = 500$	3.5% / 0.48%	4.6% / -0.08%

(a) $m = 50$

	BERT	GPT-2
$n = 50$	6.2% / -0.08%	2.2% / -0.05%
$n = 100$	6.1% / -0.37%	2.5% / 0.03%
$n = 200$	4.1% / 0.33%	6.3% / -0.01%
$n = 500$	6.1% / -0.16%	3.9% / -0.21%

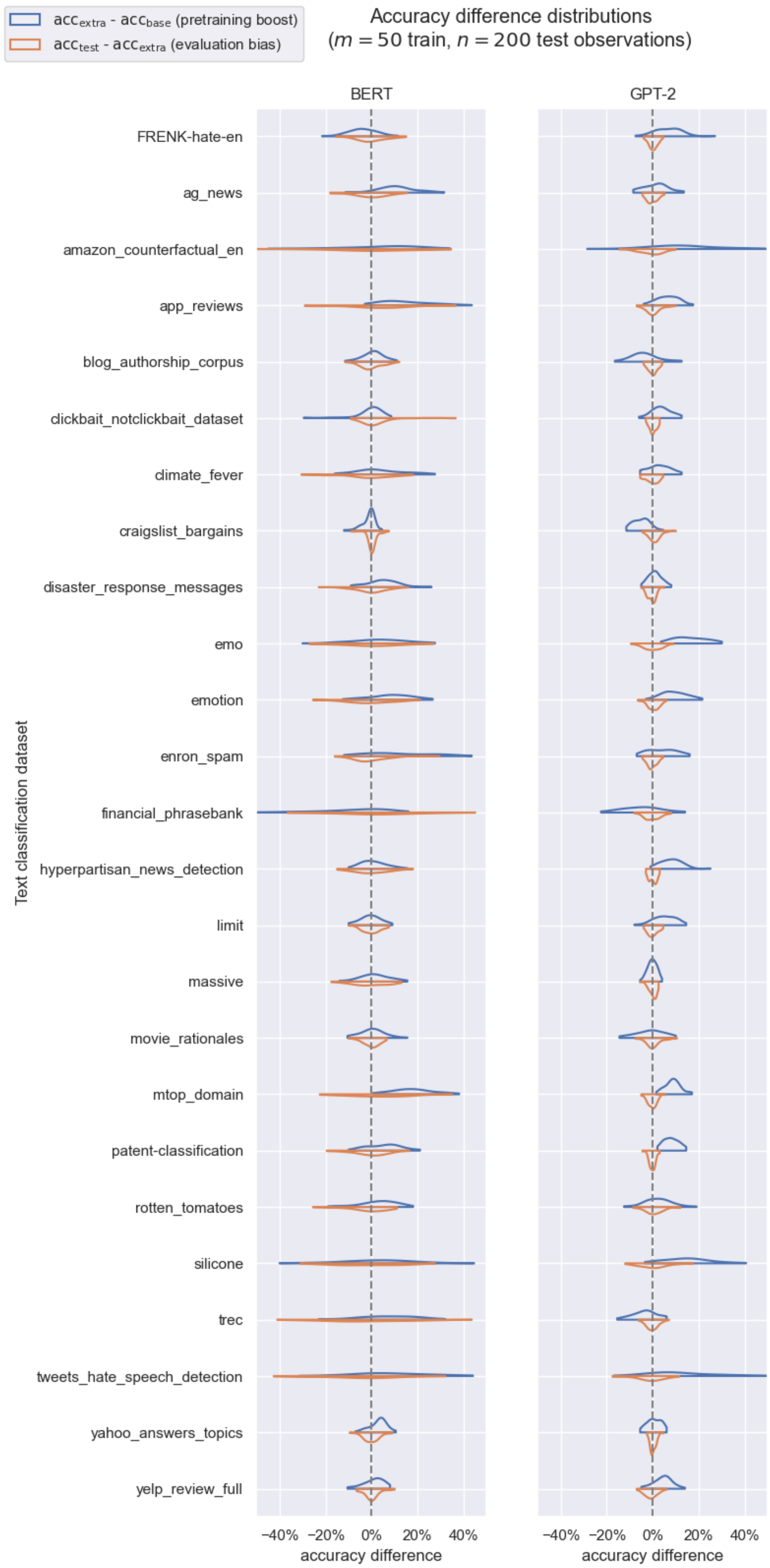
(b) $m = 100$

Table 1: Means of accuracy differences taken across all subsamples of the 25 classification tasks. For each cell, the upper-left of the diagonal corresponds to the sample mean of $\text{acc}_{\text{extra}} - \text{acc}_{\text{base}}$, and the lower-right corresponds to the sample mean of $\text{acc}_{\text{test}} - \text{acc}_{\text{extra}}$.

6 Analysis

Reporting means is not enough, especially when studying few-shot learning. Figure 1 demonstrates that there's considerable variance, despite pairing the accuracy estimators.[2] While these visualiza-

[2] One source of variance is intentionally introduced: the subsamples/splits, as explained in §4.4. The other source of variance is inherent: the added linear layer to perform classification is initialized with random weights.



acc_extra - acc_base (pretraining boost)
acc_test - acc_extra (evaluation bias)
Accuracy difference distributions
(m = 50 train, n = 200 test observations)
BERT
GPT-2
Text classification dataset
FRENK-hate-en
ag_news
amazon_counterfactual_en
app_reviews
blog_authorship_corpus
clickbait_notclickbait_dataset
climate_fever
craigslist_bargains
disaster_response_messages
emo
emotion
enron_spam
financial_phrasebank
hyperpartisan_news_detection
limit
massive
movie_rationales
mtop_domain
patent-classification
rotten_tomatoes
silicone
trec
tweets_hate_speech_detection
yahoo_answers_topics
yelp_review_full
−40% −20% 0% 20% 40%
accuracy difference
−40% −20% 0% 20% 40%
accuracy difference

Figure 1

tions tell us about how raw accuracy differences vary, they do not tell us how the mean accuracy difference varies. We seek a neat answer to the core questions: on this benchmark of 25 classification tasks, how much does the overall accuracy differ between two modeling techniques, and how much does this difference vary?

One way to communicate the variance is to estimate the standard error of the mean difference across classification tasks. But the standard error statistic can be difficult to interpret (Morey et al., 2016). Furthermore, its computation is not completely trivial due to the data's hierarchical dependency structure: each triple, ($\text{acc}_{\text{extra}}$, acc_{test}, acc_{base}), is drawn from (`train`, `test`), which is itself drawn from the given classification dataset.

6.1 Model

This analysis does not aim to estimate standard errors. Instead, a hierarchical model is fit:

$$
\begin{aligned}
Y_{ijkl} &\sim \text{Binomial}(n, \lambda_{ijkl}) && (1)\\
\text{logit}(\lambda_{ijkl}) &= \mu + \alpha z_i + U_j + V_{jk} + \beta x_{ijkl} && (2)\\
\mu &\sim \text{Normal}(0, 1) && (3)\\
\alpha &\sim \text{Normal}(0, 5) && (4)\\
U_j &\sim \text{Normal}(0, \sigma_U) && (5)\\
V_{jk} &\sim \text{Normal}(0, \sigma_V) && (6)\\
\beta &\sim \text{Normal}(0, 1) && (7)\\
\sigma_U, \sigma_V &\sim \text{HalfNormal}(0, 1) && (8)
\end{aligned}
$$

(1) number of correct predictions

(2) logit link for accuracy rate, additive effects

(3) prior for the global intercept

(4) prior for the effect of the type of LM (BERT or GPT-2)—a control variable

(5) prior for the effect of the classification task (partial-pooled to reduce overfitting)

(6) prior for the nested effect of the task's subsampled dataset

(7) prior for the effect of interest ($x_{ijk1} = 1$ indicates the modeling intervention)

(8) prior for standard deviations.

The model is fit using Markov Chain Monte Carlo, using the interface provided by the `bambi` package (Capretto et al., 2022). Appendix E.1 includes a simulation demonstrating the model's ability to correctly recover null and non-null effects.

6.2 Posterior predictions

In NLP benchmarks, methods are assessed by taking their average performance across tasks. To place the analysis results in this context, samples from the posterior predictive distribution of $Y_{ijk1} - Y_{ijk0}$ (6.1) are taken, then averaged across i (the 2 LM types—BERT and GPT-2), j (the 25 classification tasks), and k (their subsamples), and divided by n to obtain the distribution of the average accuracy difference.

These distributions are plotted in Figure 2. Each distribution is that of the marginal effect of the modeling intervention: pretraining versus not pretraining before classification training (the pretraining boost), or pretraining on unlabeled test set text instead of on unlabeled independent text before classification training (the evaluation bias).

7 Discussion

Figure 2 demonstrates that the average pretraining boost is significant in every configuration of the experiment, ranging from 2% to 6%. This finding replicates that from Gururangan et al. (2020). After averaging across settings for m, n, and the 2 LM types, only two of the 25 classification tasks had a pretraining boost less than 0, and both were greater than -1%.[3] Overall, pretraining is beneficial, so there may be a detectable evaluation bias.

As shown in Figure 2, the evaluation bias bounces inconsistently and insignificantly around 0. After averaging, 12 of the 25 classification tasks had a positive evaluation bias, and all tasks had an average evaluation bias less than 1% in absolute value. Given the lack of evidence for an evaluation bias in either direction, it's unlikely that a benchmark which releases unlabeled test set text can systematically promote models pretrained on it over equally performant models which pretrained on unlabeled independent text.

Moscovich and Rosset (2022) found that the evaluation bias caused by certain unsupervised methods for tabular data gets closer to 0 as n increases. This finding is not confirmed by this experiment. Figure 2 shows that for $m = 50$ and $m = 100$, distributions of the evaluation bias consistently hover around 0 across settings for n. But far more experiments varying n are needed to thoroughly assess this insensitivity.

[3] The tasks were `blog_authorship_corpus` and `movie_rationales`.

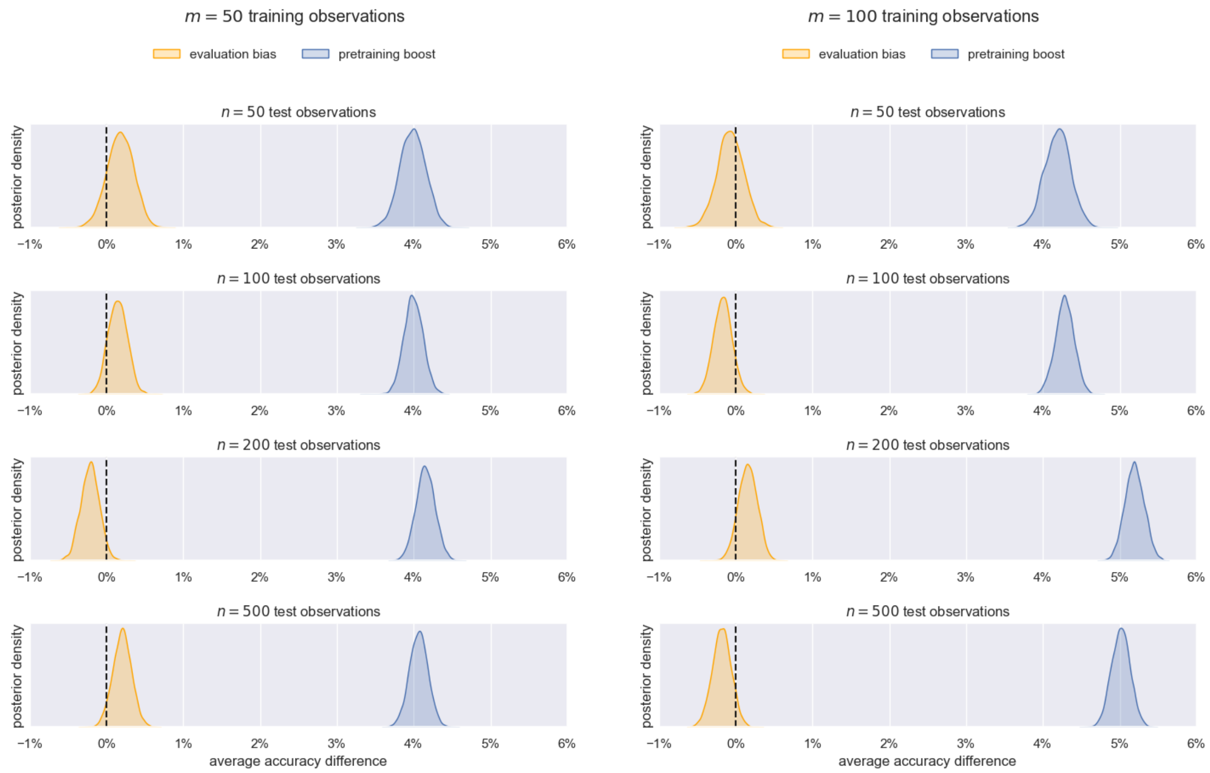


Figure 2: Distributions of average accuracy differences for $m = 50$ (left) and $m = 100$ (right). The evaluation bias is akin to $\text{acc}_{\text{test}} - \text{acc}_{\text{extra}}$. The pretraining boost is akin to $\text{acc}_{\text{extra}} - \text{acc}_{\text{base}}$.

8 Overtraining

§4.3 posited that if there's no boost going from acc_{base} to $\text{acc}_{\text{extra}}$, then it shouldn't be surprising that there's no change going from $\text{acc}_{\text{extra}}$ to acc_{test}. This check is meant to rule out undertraining as the cause of a null evaluation bias. But what if a model is *overtrained* on unlabeled text? Perhaps there's an evaluation bias when pretraining is unhelpful, because pretraining can only exploit noise in the unlabeled text which the model is then rewarded for when it's evaluated on that same text.

For example, in a brief experiment with PCA for a synthetic linear regression task (Appendix G), the evaluation bias increases with the effective rank of the features. In other words, fitting a PCA on test set features artificially improves test set performance particularly when fitting a PCA on independent features is unhelpful.

To test this hypothesis for text classification, GPT-2 is intentionally overtrained on unlabeled text for 2 epochs instead of 1.

For each of the 25 classification tasks and their subsamples, pretraining for 2 epochs instead of 1 resulted in a lower pretraining loss. The average pretraining loss is 20% lower, and the pretraining boost is negative, which indicates overfitting, as intended. Figure 3 shows that, despite overtraining, the evaluation bias hovers around 0. Overtraining on unlabeled test set text causes test set performance to degrade to the same degree that overfitting on unlabeled independent text does.

9 Zero-shot text classification

Prompting an LLM is a popular choice for solving NLP problems. Nothing prevents these prompts from being included as part of an LLM's pretraining data. For example, Gemma 2 (Gemma Team, 2024) is intentionally pretrained on prompts from the LMSYS benchmark (Zheng et al., 2023).

To study a modern prompting approach, the experiment in §4 is repeated with two modifications. First, task-adaptive pretraining is done on prompts—unlabeled texts with instructions for solving the task. Second, classification training is not performed. The further-pretrained LLM is prompted to do the task on text from the test set.

More specifically, pretraining is performed by adding a QLoRA adapter layer (Dettmers et al., 2024) to every linear layer in Mistral 7B[4] (Jiang et al., 2023). Perhaps notably, instructions mention the set of possible answers—the class names. Here

[4] Mistral-7B-v0.3, the non-instruction-trained model.

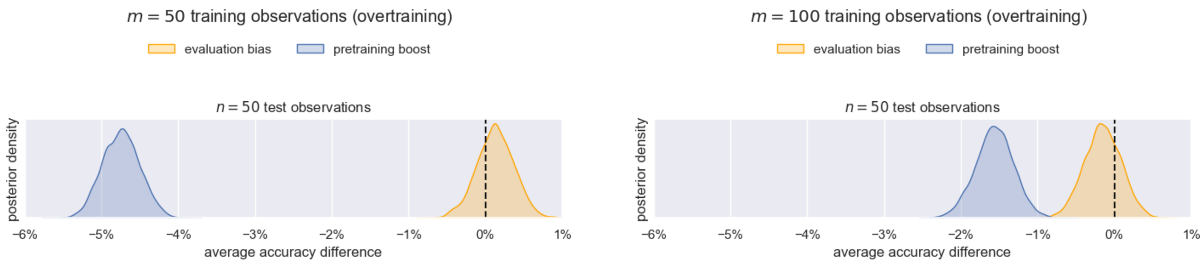


Figure 3: Distributions of average accuracy differences for $m = 50$ (left) and $m = 100$ (right) after pretraining GPT-2 for 2 epochs instead of 1 (§8).

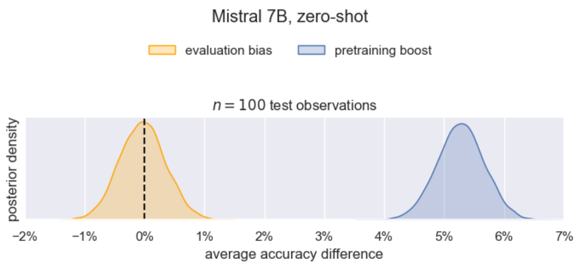


Figure 4: Distributions of average accuracy differences for zero-shot classification (§9). For each of the 25 classification tasks, 20 subsamples were taken.

is an example of a prompt for the `ag_news` (Zhang et al., 2015) task:

```
Your task is to classify a given text as
one of these categories:
World
Sports
Business
Sci/Tech

The text is a news article. Answer with its topic.

### Text: Bombardier CEO Quits, Shares
Dive Paul Tellier stepped down on Monday
as president and chief executive of
Bombardier Inc. (BBDsvb.TO: Quote,
Profile, Research) (BBDb.
 ### Answer:
```

Figure 4 shows that while pretraining on prompts improves accuracy, pretraining on prompts directly from the test set does not increase test set accuracy compared to pretraining on independently drawn prompts.

12 of the 25 tasks had a positive evaluation bias and 13 had a negative evaluation bias. To analyze accuracies from each task (instead of averaging them as a benchmark would), a p-value for the following hypothesis test is computed for each task:

$$H_0 : \mathrm{E}[\mathrm{acc}_{\mathrm{test}} - \mathrm{acc}_{\mathrm{extra}}] = 0$$
$$H_1 : \mathrm{E}[\mathrm{acc}_{\mathrm{test}} - \mathrm{acc}_{\mathrm{extra}}] > 0.$$

The p-value is estimated via permutation testing. It's then adjusted to control the false discovery rate (Benjamini and Hochberg, 1995). All p-values were greater than 0.5.[5] In other words, at the task level, there is still no evidence of evaluation bias.

A limitation of this experiment is that it doesn't account for data contamination. If Mistral 7B's pretraining data included labeled or unlabeled parts of the classification datasets used here, the pretraining boost and evaluation bias may be diluted.

10 Meta-analysis

§4.4 briefly argues for subsampling multiple datasets from the full classification dataset. To assess this argument, the analysis was repeated on 500 random slices of the $m = 100, n = 500$ dataset of accuracies such that exactly 1 ($\mathrm{acc}_{\mathrm{extra}}$, $\mathrm{acc}_{\mathrm{test}}$, $\mathrm{acc}_{\mathrm{base}}$) triple per classification task (instead of 20 triples) is included. This unreplicated data is often all you get from benchmarks.

Figure 5 (left) displays the cumulative distribution of the posterior mean of the evaluation bias for $m = 100, n = 500$ under this unreplicated experimental design. The distribution is quite variant. There's a 47% chance that the posterior mean of β—the average increase in the log-odds of a correct prediction by pretraining on unlabeled test set text instead of on unlabeled independent text—is outside the interval $(-0.04, 0.04)$, which would indicate a significant negative or positive bias.[6] For the zero-shot experiment in §9, there's a 49% chance that that the posterior mean of β is outside the interval $(-0.07, 0.07)$. In other words, without

[5] For the test that $\mathrm{E}[\mathrm{acc}_{\mathrm{extra}} - \mathrm{acc}_{\mathrm{base}}] > 0$, 14 of the 25 p-values were less than 0.005, which demonstrates that the testing procedure may have some statistical power.

[6] For 0.04, the odds ratio is $e^{0.04} \approx 1.04$. For context, the average odds ratio between adjacent submissions in the RAFT leaderboard is 1.03. For posterior means outside $(-0.04, 0.04)$, all of their 89% credible intervals exclude 0, which evidences a non-null effect.

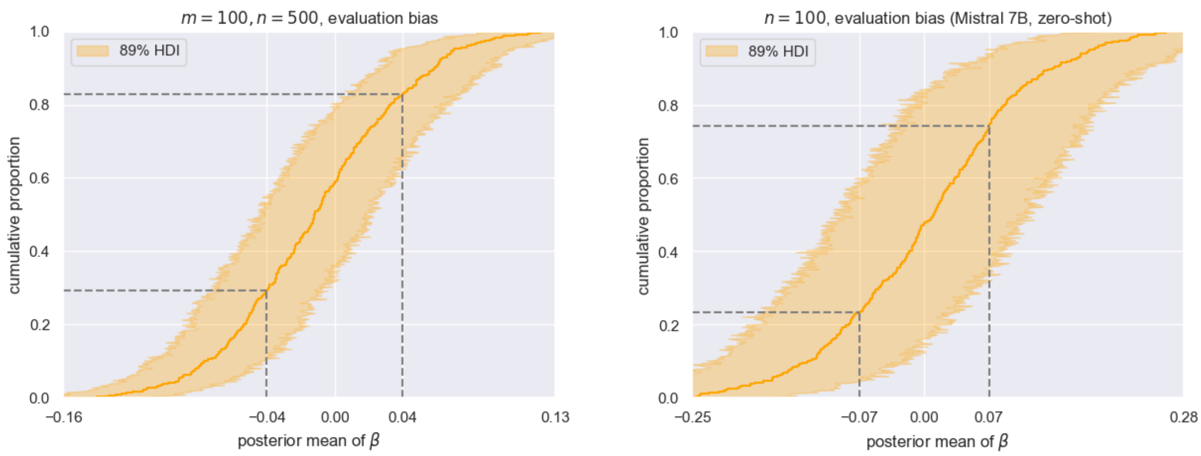


Figure 5: Distributions of this paper's conclusions for $m = 100, n = 500$ (left) and zero-shot $n = 100$ (right) had there been no technical replication (§10).

subsampling, one may as well flip a coin to decide whether pretraining on unlabeled test set text is fair.

11 Conclusion

Across combinations for the number of classification training examples ($m = 50, 100$), the number of pretraining or evaluation examples ($n = 50, 100, 200, 500$), and for zero-shot prompting with an LLM, task-adaptive pretraining on unlabeled test set text—instead of on unlabeled independent text—did not result in a consistent or significant evaluation bias. This appears to be the case when pretraining helps (§7) and when it hurts (§8).

For benchmarks which only release unlabeled text from the test set, this finding doesn't completely absolve LLM evaluations from scrutiny. The reason is that the boost from pretraining on unlabeled text—which is often significant—could be viewed as a type of evaluation bias, depending on how LLMs generalize. More concretely, say there's a benchmark and two LLMs, A and B. A was *not* pretrained on unlabeled text from the test set of the benchmark, while B was. With the perspective that LLM benchmarks supply scores which are correlates of performance on real-world tasks—instead of indicators of performance solely on the benchmark's tasks—then B scoring higher on the benchmark than A may be a misleading signal. Because if pretraining on the benchmark's unlabeled text causes B to generalize better only *within* the distribution of the benchmark, then B's edge on this benchmark does not signal an edge in real-world tasks. This argument implies that knowing whether or not an LLM was pretrained on unlabeled text from the test set is still important.

One recommendation for designing few-shot benchmarks, which expands on the principle about robustness from Bragg et al. (2021) and recommendations from Madaan et al. (2024), is based on the meta-analysis in §10: empirical studies of few-shot learning should consider including multiple, independent subsamples of training data. While a single training set combined with a large test set is sufficient for precise, unbiased estimation of out-of-sample performance, this estimator is conditional on the training set. In few-shot learning, the training set is, by definition, minimal. The estimator hides two sources of variance—that from the randomly drawn training set, and that from randomness inherent in the training procedure. Figure 5 shows that this variance is large-enough to turn a methodology into a coin flip for two different training procedures. In-context learning with LLMs is also sensitive to the selection of few-shot examples (Lu et al., 2022, Alzahrani et al., 2024). Benchmarks which require training on multiple, independent subsamples would expose training variance.

A limitation of this paper is that it does not analyze semi-supervised methods like Pattern-Exploiting Training. This paper also doesn't study hand-inspecting the test set text and targeting interventions accordingly. This paper's conclusions are limited to task-adaptive pretraining of LMs.

A direction for future research is to explore the role of causality. Jin et al. (2021) argue and demonstrate that the benefit of task-adaptive pretraining depends on the learning task's causal direction. Perhaps the principle of independent causal mechanisms is also relevant in assessing the fairness of pretraining on test set features.

Acknowledgements

Currently omitted for anonymity.

References

Neel Alex, Eli Lifland, Lewis Tunstall, Abhishek Thakur, Pegah Maham, C. Riedel, Emmie Hine, Carolyn Ashurst, Paul Sedille, Alexis Carlier, Michael Noetel, and Andreas Stuhlmüller. 2021. Raft: A real-world few-shot text classification benchmark. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

Norah Alzahrani, Hisham Abdullah Alyahya, Yazeed Alnumay, Sultan Alrashed, Shaykhah Alsubaie, Yusef Almushaykeh, Faisal Mirza, Nouf Alotaibi, Nora Altwairesh, Areeb Alowisheq, et al. 2024. When benchmarks are targets: Revealing the sensitivity of large language model leaderboards. *arXiv preprint arXiv:2402.01781*.

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 (Methodological)*, 57(1):289–300.

Jonathan Bragg, Arman Cohan, Kyle Lo, and Iz Beltagy. 2021. Flex: Unifying evaluation for few-shot nlp. *Advances in Neural Information Processing Systems*, 34:15787–15800.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Tomás Capretto, Camen Piho, Ravin Kumar, Jacob Westfall, Tal Yarkoni, and Osvaldo A Martin. 2022. Bambi: A simple interface for fitting bayesian linear models in python. *Journal of Statistical Software*, 103(15):1–29.

Emile Chapuis, Pierre Colombo, Matteo Manica, Matthieu Labeau, and Chloé Clavel. 2020. Hierarchical pre-training for sequence labelling in spoken dialog. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2636–2648, Online. Association for Computational Linguistics.

Ankush Chatterjee, Kedhar Nath Narahari, Meghana Joshi, and Puneet Agrawal. 2019. SemEval-2019 task 3: EmoContext contextual emotion detection in text. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 39–48, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.

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.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.

Thomas Diggelmann, Jordan Boyd-Graber, Jannis Bulian, Massimiliano Ciaramita, and Markus Leippold. 2020. Climate-fever: A dataset for verification of real-world climate claims.

Jack FitzGerald, Christopher Hench, Charith Peris, Scott Mackie, Kay Rottmann, Ana Sanchez, Aaron Nash, Liam Urbach, Vishesh Kakarala, Richa Singh, Swetha Ranganath, Laurie Crist, Misha Britan, Wouter Leeuwis, Gokhan Tur, and Prem Natarajan. 2023. MASSIVE: A 1M-example multilingual natural language understanding dataset with 51 typologically-diverse languages. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4277–4302, Toronto, Canada. Association for Computational Linguistics.

Gemma Team. 2024. Gemma 2: Improving open language models at a practical size. .

Giovanni Grano, Andrea Di Sorbo, Francesco Mercaldo, Corrado A Visaggio, Gerardo Canfora, and Sebastiano Panichella. 2017. Android apps and user feedback: a dataset for software evolution and quality improvement. In *Proceedings of the 2nd ACM SIGSOFT international workshop on app market analytics*, pages 8–11.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2022. PPT: Pre-trained prompt tuning for few-shot learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8410–8423, Dublin, Ireland. Association for Computational Linguistics.

Neel Guha, Julian Nyarko, Daniel Ho, Christopher Ré, Adam Chilton, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel Rockmore, Diego Zambrano, et al. 2024. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models. *Advances in Neural Information Processing Systems*, 36.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. 2009. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer.

He He, Derek Chen, Anusha Balakrishnan, and Percy Liang. 2018. Decoupling strategy and generation in negotiation dialogues. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2333–2343, Brussels, Belgium. Association for Computational Linguistics.

Zhang Huangzhao. 2018. Yahoo-answers-topic-classification-dataset. `https://github.com/LC-John/Yahoo-Answers-Topic-Classification-Dataset`.

Alon Jacovi, Avi Caciularu, Omer Goldman, and Yoav Goldberg. 2023. Stop uploading test data in plain text: Practical strategies for mitigating data contamination by evaluation benchmarks. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5075–5084, Singapore. Association for Computational Linguistics.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Zhijing Jin, Julius von Kügelgen, Jingwei Ni, Tejas Vaidhya, Ayush Kaushal, Mrinmaya Sachan, and Bernhard Schoelkopf. 2021. Causal direction of data collection matters: Implications of causal and anticausal learning for NLP. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9499–9513, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. 2019. SemEval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 829–839, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Ravin Kumar, Colin Carroll, Ari Hartikainen, and Osvaldo Martin. 2019. Arviz a unified library for exploratory analysis of bayesian models in python. *Journal of Open Source Software*, 4(33):1143.

Nikola Ljubešić, Darja Fišer, and Tomaž Erjavec. 2019. The frenk datasets of socially unacceptable discourse in slovene and english.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Lovish Madaan, Aaditya K Singh, Rylan Schaeffer, Andrew Poulton, Sanmi Koyejo, Pontus Stenetorp, Sharan Narang, and Dieuwke Hupkes. 2024. Quantifying variance in evaluation benchmarks. *arXiv preprint arXiv:2406.10229*.

P. Malo, A. Sinha, P. Korhonen, J. Wallenius, and P. Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65.

Irene Manotas, Ngoc Phuoc An Vo, and Vadim Sheinin. 2020. LiMiT: The literal motion in text dataset. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 991–1000, Online. Association for Computational Linguistics.

Richard McElreath. 2018. *Statistical rethinking: A Bayesian course with examples in R and Stan*. Chapman and Hall/CRC.

Vangelis Metsis, Ion Androutsopoulos, and Georgios Paliouras. 2006. Spam filtering with naive bayes-which naive bayes? In *CEAS*, volume 17, pages 28–69. Mountain View, CA.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Richard D Morey, Rink Hoekstra, Jeffrey N Rouder, Michael D Lee, and Eric-Jan Wagenmakers. 2016. The fallacy of placing confidence in confidence intervals. *Psychonomic bulletin & review*, 23:103–123.

Amit Moscovich and Saharon Rosset. 2022. On the cross-validation bias due to unsupervised preprocessing. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 84(4):1474–1502.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia. Association for Computational Linguistics.

James O'Neill, Polina Rozenshtein, Ryuichi Kiryo, Motoko Kubota, and Danushka Bollegala. 2021. I wish I would have loved this one, but I didn't – a multilingual dataset for counterfactual detection in product review. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7092–7108, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124, Ann Arbor, Michigan. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, Brussels, Belgium. 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.

Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W Pennebaker. 2006. Effects of age and gender on blogging. In *AAAI spring symposium: Computational approaches to analyzing weblogs*, volume 6, pages 199–205.

Eva Sharma, Chen Li, and Lu Wang. 2019. BIGPATENT: A large-scale dataset for abstractive and coherent summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2204–2213, Florence, Italy. Association for Computational Linguistics.

Roshan Sharma. 2019. Twitter-sentiment-analysis. `https://github.com/sharmaroshan/Twitter-Sentiment-Analysis`.

Tan Thongtan and Tanasanee Phienthrakul. 2019. Sentiment classification using document embeddings trained with cosine similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 407–414, Florence, Italy. Association for Computational Linguistics.

Lewis Tunstall, Nils Reimers, Unso Eun Seo Jo, Luke Bates, Daniel Korat, Moshe Wasserblat, and Oren Pereg. 2022. Efficient few-shot learning without prompts. *arXiv preprint arXiv:2209.11055*.

Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? a quasi-synchronous grammar for QA. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 22–32, Prague, Czech Republic. 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, Mariama Drame, Quentin Lhoest, and Alexander Rush. 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.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2021. Revisiting few-sample bert fine-tuning. In *International Conference on Learning Representations*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric Xing, et al. 2023. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. *arXiv preprint arXiv:2309.11998*.

A Classification tasks

The experiment was ran on 25 publicly available text classification tasks found in `https://huggingface.co/datasets`. Inclusion criteria:

1. All text is in English.
2. The number of classes is not greater than 25, because only 50 or 100 observations are used for training the classifier.
3. The task is to classify one text, not a pair as in, e.g., textual entailment tasks.
4. Texts aren't so long that too much useful signal is dropped when text is truncated to fit in BERT/GPT-2's context window, which is set to 256 tokens.
5. Based on our best judgment, it's likely that BERT/GPT-2 can do better than guessing.

Table 2 lists the exact tasks.

Hugging Face dataset	Author(s)	Number of classes	Text length (25, 75) percentiles
ag_news	Zhang et al. (2015)	4	(196, 266)
SetFit/amazon_counterfactual_en	O'Neill et al. (2021)	2	(60, 125)
app_reviews	Grano et al. (2017)	5	(10, 77)
blog_authorship_corpus	Schler et al. (2006)	2	(92, 556)
christinacdl/clickbait_notclickbait_dataset		2	(46, 69)
climate_fever	Diggelmann et al. (2020)	4	(80, 156)
aladar/craigslist_bargains	He et al. (2018)	6	(346, 713)
disaster_response_messages		3	(74, 178)
emo	Chatterjee et al. (2019)	4	(44, 83)
dair-ai/emotion	Saravia et al. (2018)	6	(53, 129)
SetFit/enron_spam	Metsis et al. (2006)	2	(342, 1553)
financial_phrasebank	Malo et al. (2014)	3	(79, 157)
classla/FRENK-hate-en	Ljubešić et al. (2019)	2	(34, 160)
hyperpartisan_news_detection	Kiesel et al. (2019)	2	(39, 63)
limit	Manotas et al. (2020)	2	(53, 123)
AmazonScience/massive	FitzGerald et al. (2023)	18	(24, 44)
movie_rationales	DeYoung et al. (2020)	2	(2721, 4659)
mteb/mtop_domain	Muennighoff et al. (2023)	11	(26, 44)
ccdv/patent-classification	Sharma et al. (2019)	9	(441, 775)
rotten_tomatoes	Pang and Lee (2005)	2	(76, 149)
silicone	Chapuis et al. (2020)	4	(29, 75)
trec	Wang et al. (2007)	6	(36, 61)
tweets_hate_speech_detection	Sharma (2019)	2	(62, 107)
yahoo_answers_topics	Huangzhao (2018)	10	(58, 213)
yelp_review_full	Zhang et al. (2015)	5	(287, 957)

Table 2: Brief descriptions of the 25 classification tasks used in this experiment. Click the link in the cell to be taken to the dataset homepage in `https://huggingface.co/datasets`. The dataset subset (or config) and the chosen prediction task are specified in code in `src/pretrain_on_test/data.py`.

B Other experiment choices

This section expands on §4.

First, we clarify how classification training is performed. For BERT, the linear layer transforms the `[CLS]` token embedding. For GPT-2, the linear layer transforms the last token's embedding. The output dimension of the linear layer is the number of classes in the classification task. This layer, along with the rest of the weights in the LM, are finetuned to minimize classification cross entropy loss on `train`.

The BERT model used here is `bert-base-uncased`. The GPT-2 model used here is `gpt2` (small), with 124M parameters.

`train` is stratify-sampled by the class to ensure every class is represented, and to reduce the variance of accuracy estimators. `test` is not stratify-sampled. We're only interested in the *difference* between accuracies, which is a function of the difference between model likelihoods because the priors are uniform. So even if accuracies are worse than the majority vote, differences are still meaningful for the purposes of this experiment.

`train` text is not included during pretraining to eliminate the overlap of pretraining data between $\mathrm{acc}_{\mathrm{extra}}$ and $\mathrm{acc}_{\mathrm{test}}$. This choice was made in an effort to widen any gap between them. The experiment tries to go out of its way to provide evidence of a bias.

`train` contains $m = 50$ or $m = 100$ observations. $m = 50$ is inspired by the RAFT benchmark. $m = 100$ stretches the intention of "few" in few-shot learning, but was tested in an attempt to make lower-variance comparisons. BERT is quite sensitive—see Appendix D.2.

The discussion focuses on the BERT and GPT-2 results because their pretraining data is (likely) not already contaminated with text from the 25 text classification tasks. While modern NLP solutions usually involve LLMs, these models' pretraining data are opaque and more likely to include text from the 25 classification tasks (for example, from web-crawling the Dataset Viewer in HuggingFace's datasets web pages, which hosts the experiment's data) (Jacovi et al., 2023). As a result, the comparisons—$\mathrm{acc}_{\mathrm{extra}}$ versus $\mathrm{acc}_{\mathrm{base}}$ and $\mathrm{acc}_{\mathrm{test}}$ versus $\mathrm{acc}_{\mathrm{extra}}$—would be less valid.

C Hyperparameters and reproducibility

This paper's experiment and analysis code, and data, is available here: `https://github.com`. `experiment.sh` lists hyperparameters used for each classification task and experiment configuration. Hyperparameters were pre-specified based on Zhang et al. (2021), and to obey memory limits. Run the script on a GPU with at least 15 GB VRAM to reproduce results in §5. It takes about 5 days on a T4 GPU. Training is performed using the `transformers` package (Wolf et al., 2020).

D Results

D.1 Individual analysis

The Jupyter notebook `analysis/dataset.ipynb` can be run to (1) produce visualizations of the distributions of $\mathrm{acc}_{\mathrm{extra}}$, $\mathrm{acc}_{\mathrm{test}}$, and $\mathrm{acc}_{\mathrm{base}}$ (for each classification task and experiment configuration), and (2) compute p-values for the hypothesis test specified in §9. For all settings of m and n, no p-values were statistically significant at the 0.05 level. Moreover, among the 25 classification tasks studied in this paper, and across settings for m and n, there weren't tasks that repeatedly popped up as having a positive evaluation bias.

Care has to be taken when attempting to analyze or interpret $\mathrm{acc}_{\mathrm{extra}} - \mathrm{acc}_{\mathrm{base}}$ and $\mathrm{acc}_{\mathrm{test}} - \mathrm{acc}_{\mathrm{extra}}$ *together*. That's because these differences are not independent: if $\mathrm{acc}_{\mathrm{extra}}$ is high, then $\mathrm{acc}_{\mathrm{extra}} - \mathrm{acc}_{\mathrm{base}}$ increases and $\mathrm{acc}_{\mathrm{test}} - \mathrm{acc}_{\mathrm{extra}}$ decreases. This paper does not analyze the scores together, per se. We care about $\mathrm{acc}_{\mathrm{test}} - \mathrm{acc}_{\mathrm{extra}}$. $\mathrm{acc}_{\mathrm{extra}} - \mathrm{acc}_{\mathrm{base}}$ only exists to sanity check that the pretraining code works; there may be an effect to detect.

D.2 Difference distributions

Figure 1 and Figures 9 - 15 visualize the distributions of the paired differences—$\mathrm{acc}_{\mathrm{extra}} - \mathrm{acc}_{\mathrm{base}}$ and $\mathrm{acc}_{\mathrm{test}} - \mathrm{acc}_{\mathrm{extra}}$–for each configuration of the experiment.

E Analysis

The analysis in §6 can be reproduced by running all of the Jupyter notebooks in `analysis/fit_posteriors/`. Figure 2 can be reproduced by running the Jupyter notebook `analysis/results/posterior_pred.ipynb`.

Posterior samples of β (which were used to draw posterior predictive samples) were taken from four chains with 1,000 draws each, after 500 steps of tuning.

E.1 Hierarchical model checks

Hierarchical models require some basic checks to have faith in their results (McElreath, 2018).

For each of the 22 hierarchical models (16 in §6, 4 in §8, and 2 in §9), no divergences were observed during the fitting procedure. All trace plots were healthy.

Figure 7 contains prior predictive distributions for $m = 100, n = 200$, demonstrating that priors are not unreasonable. Using default priors from the `bambi` package (Capretto et al., 2022), while scientifically unreasonable (because they result in wide, basin-like accuracy distributions), did not change the conclusions of this paper.

Figure 8 contains posterior distributions of β for $m = 100, n = 200$, demonstrating the hierarchical model's ability to recover both null and non-null effects. This test can be reproduced by running the Jupyter notebook `analysis/test.ipynb`.

F Meta-analysis

The meta-analysis in §10 can be reproduced by running the script, `analysis/meta/meta.py`, and then the Jupyter notebook `analysis/meta/meta.ipynb`. No divergences were observed.

G Experiment with PCA

This experiment's design is akin to §4 except that the unsupervised pretraining procedure is PCA, the supervised training procedure is linear regression on synthetic data, and $2,000$ subsamples are taken instead of 20-100. 20 features are generated with known effective ranks. The dependent variable is a linear transformation of these features plus independent, normally distributed noise with a standard deviation of 1. Performance is measured using R^2.

Figure 6 can be reproduced by running the notebook `analysis/pca.ipynb`

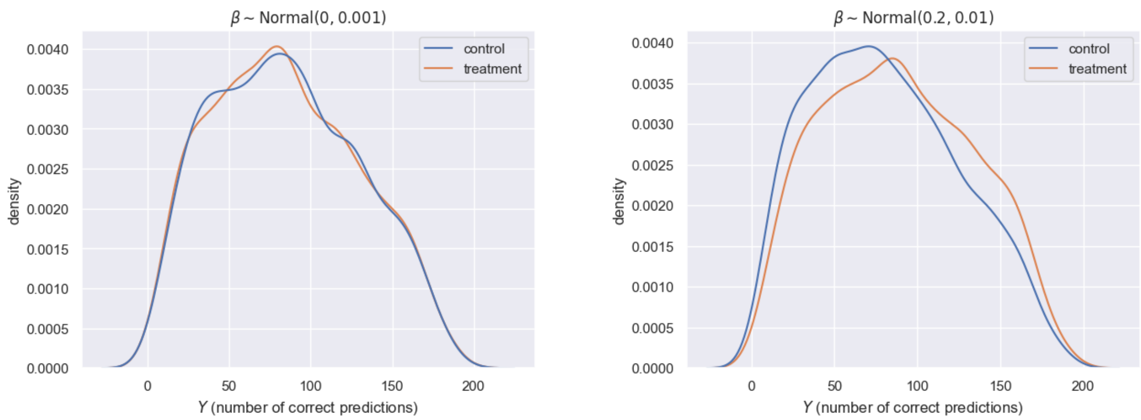
H Zero-shot text classification

The zero-shot experiment files are in `cloud_scripts/gcp/experiments/zero_shot/`. Figure 4 can be reproduced by running the notebook in `analysis/fit_posteriors/zero_shot` and then the notebook, `analysis/results/posterior_pred.ipynb`.

We only study $n = 100$ in an initial effort to provide evidence of an evaluation bias (due to the relatively small test set), and take 20 repeated subsamples instead of 50. While $n = 100$ is quite small, benchmarks such as LegalBench (Guha et al., 2024) have test data in this range. And the analysis transparently exposes variance.

QLoRA hyperparameters were pre-specified: every adapter has rank 16 with $\alpha = 32$ (LoRA scaling factor), a 0.05 dropout rate, and no bias parameters. The adapter layers introduce $41,943,040$ new, trainable parameters to Mistral 7B, whose parameters are frozen.

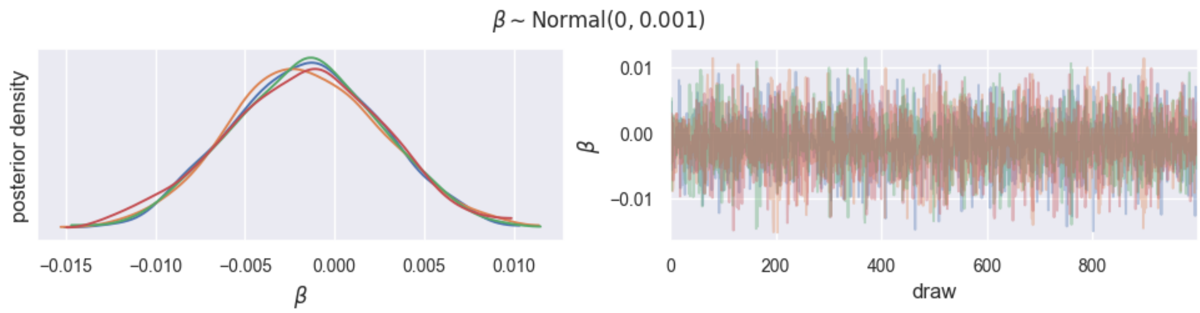
Figure 6: The shaded regions are 95% confidence intervals.



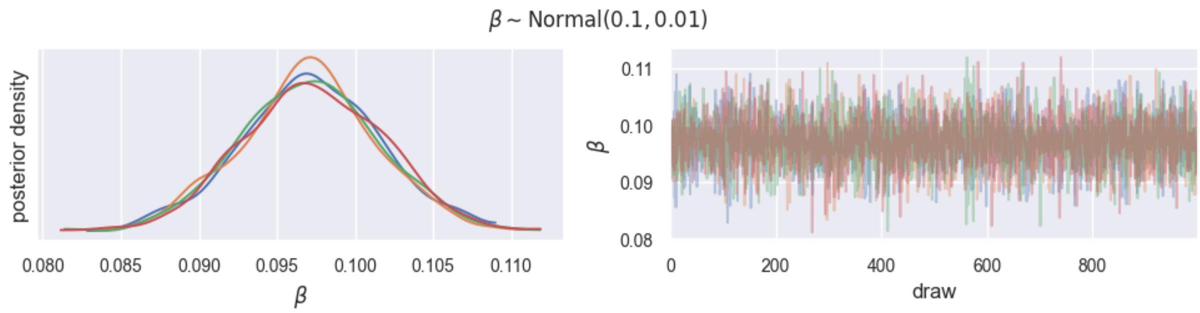
(a) Null effect

(b) Non-null, positive effect

Figure 7: Prior predictive distributions for $m = 100, n = 200$ from two different priors for β—the expected increase in the log-odds of a correct prediction resulting from an intervention/treatment.

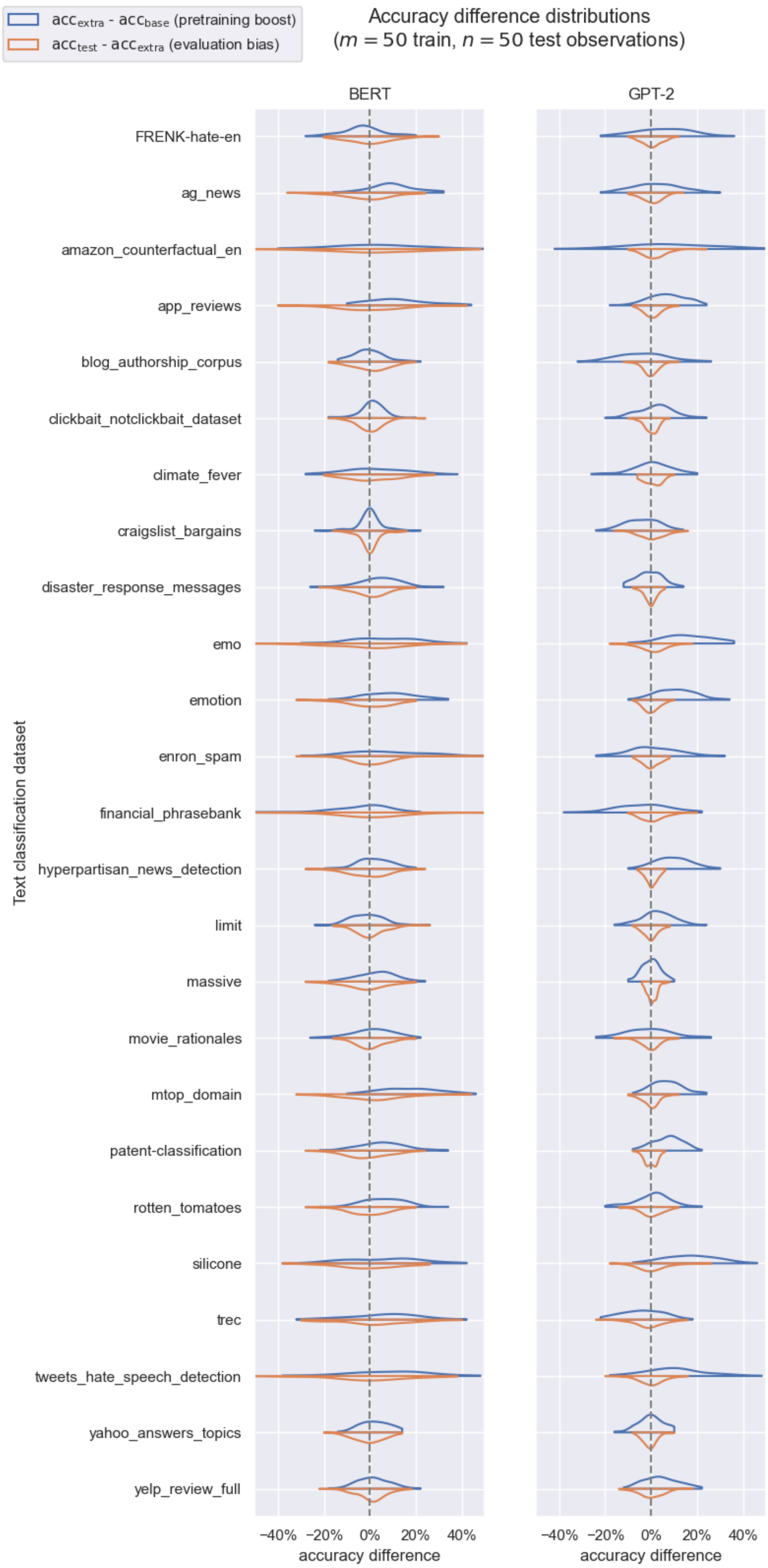


(a) Null effect is recovered.



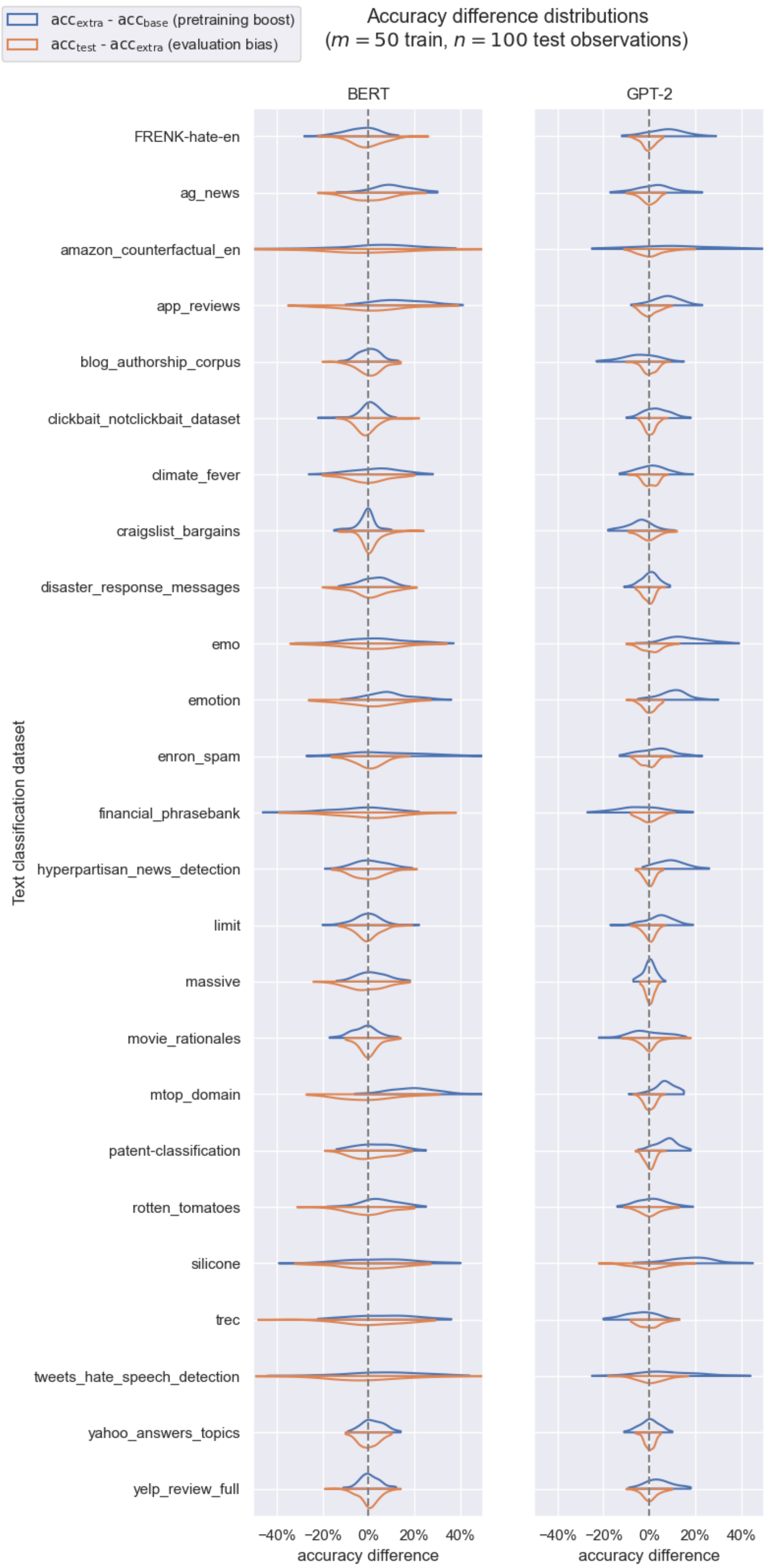
(b) Non-null, positive effect is recovered.

Figure 8: Posterior distributions and trace plots for null and non-null effects **from simulated data** where $m = 100, n = 200$, approximated by four chains with 1,000 draws each, after 500 steps of tuning. For each model, no divergences were observed during the fitting procedure. Visualizations were produced by the `arviz` package (Kumar et al., 2019).



$\mathrm{acc}_{\mathrm{extra}}$ - $\mathrm{acc}_{\mathrm{base}}$ (pretraining boost)
$\mathrm{acc}_{\mathrm{test}}$ - $\mathrm{acc}_{\mathrm{extra}}$ (evaluation bias)
Accuracy difference distributions
($m = 50$ train, $n = 50$ test observations)
BERT
GPT-2
Text classification dataset
FRENK-hate-en
ag_news
amazon_counterfactual_en
app_reviews
blog_authorship_corpus
clickbait_notclickbait_dataset
climate_fever
craigslist_bargains
disaster_response_messages
emo
emotion
enron_spam
financial_phrasebank
hyperpartisan_news_detection
limit
massive
movie_rationales
mtop_domain
patent-classification
rotten_tomatoes
silicone
trec
tweets_hate_speech_detection
yahoo_answers_topics
yelp_review_full
−40% −20% 0% 20% 40%
accuracy difference
−40% −20% 0% 20% 40%
accuracy difference

Figure 9



acc_extra - acc_base (pretraining boost)
acc_test - acc_extra (evaluation bias)
Accuracy difference distributions
(m = 50 train, n = 100 test observations)
BERT
GPT-2
Text classification dataset
FRENK-hate-en
ag_news
amazon_counterfactual_en
app_reviews
blog_authorship_corpus
clickbait_notclickbait_dataset
climate_fever
craigslist_bargains
disaster_response_messages
emo
emotion
enron_spam
financial_phrasebank
hyperpartisan_news_detection
limit
massive
movie_rationales
mtop_domain
patent-classification
rotten_tomatoes
silicone
trec
tweets_hate_speech_detection
yahoo_answers_topics
yelp_review_full
−40% −20% 0% 20% 40%
accuracy difference
−40% −20% 0% 20% 40%
accuracy difference

Figure 10

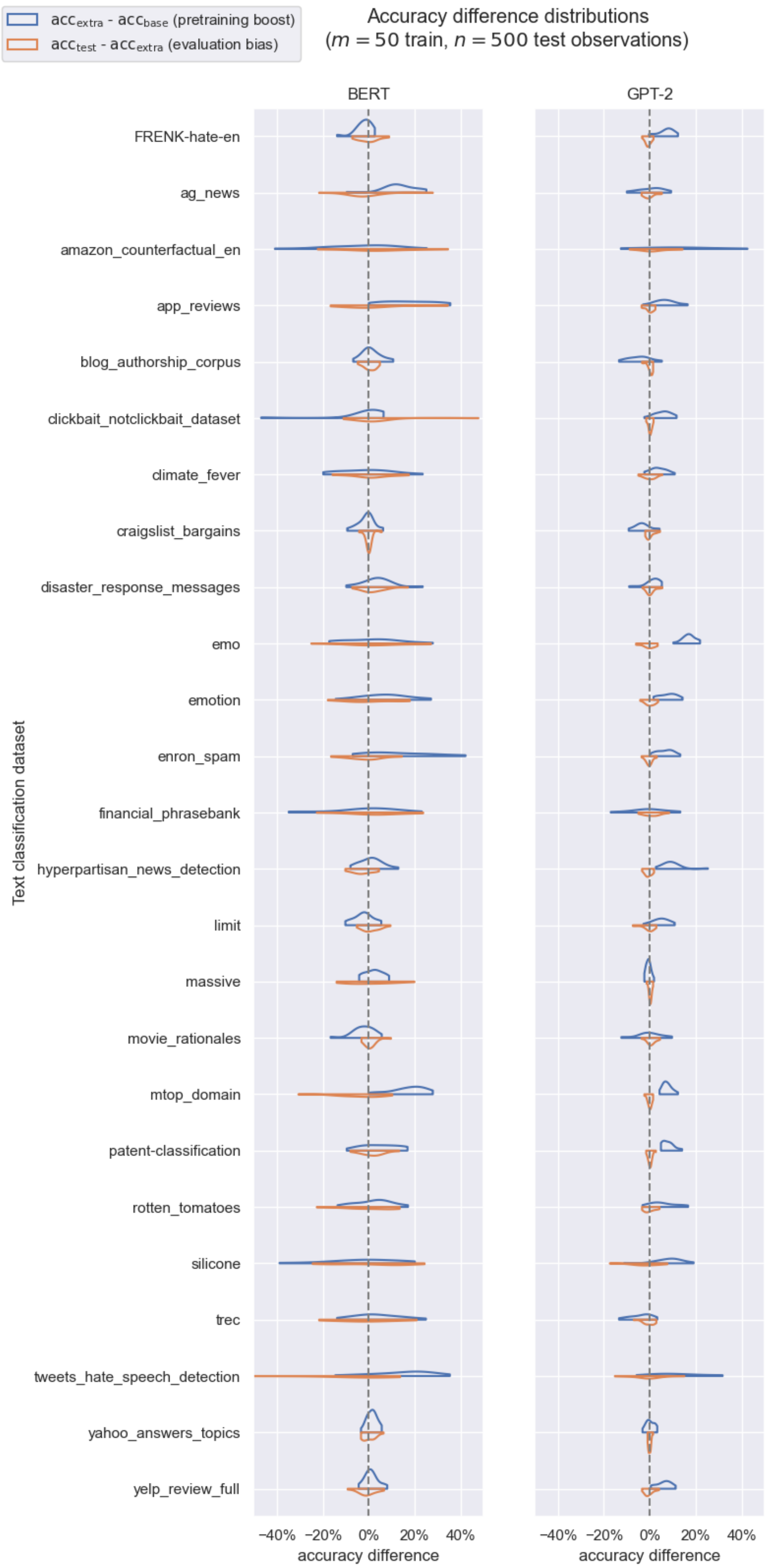
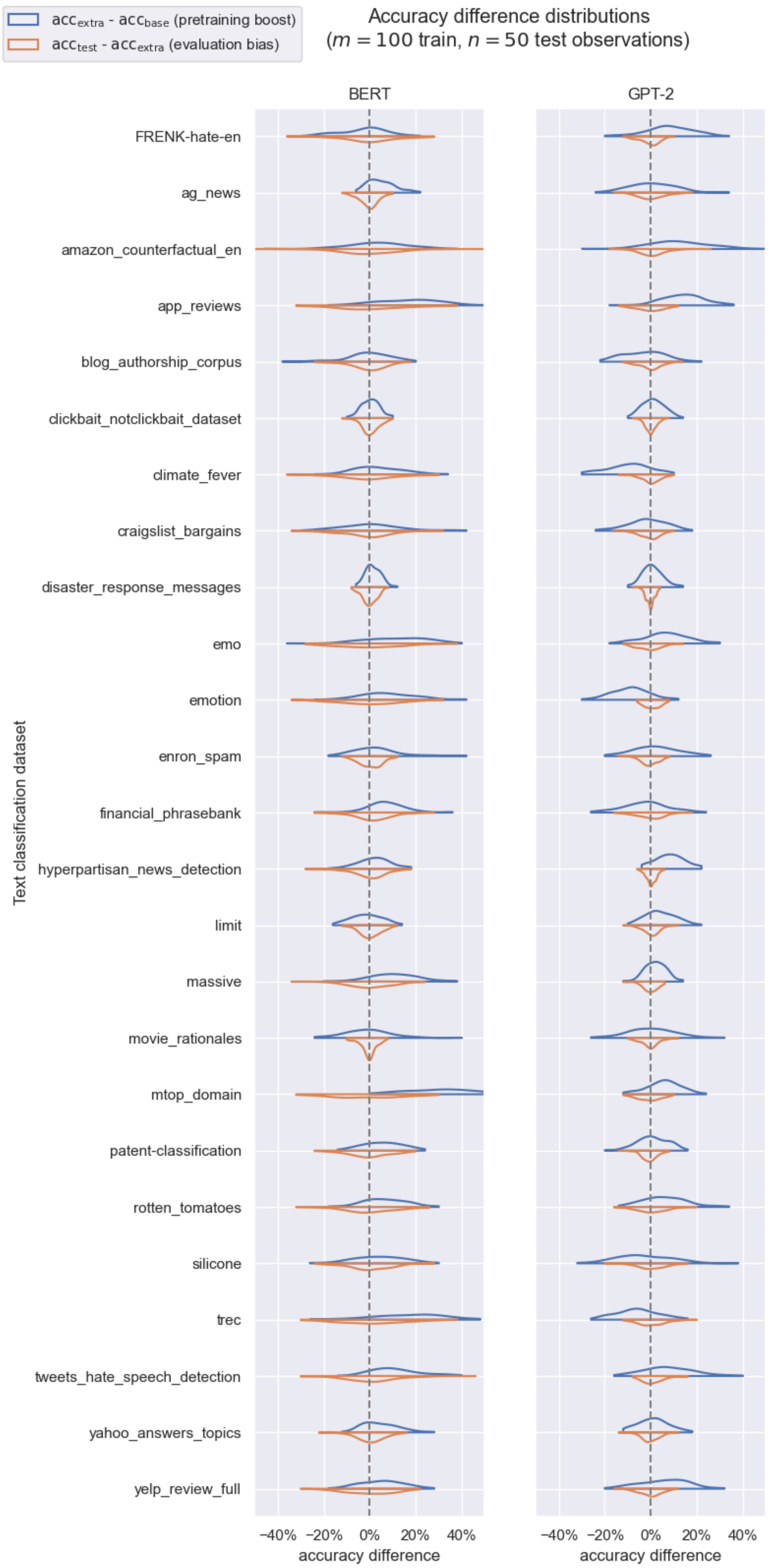
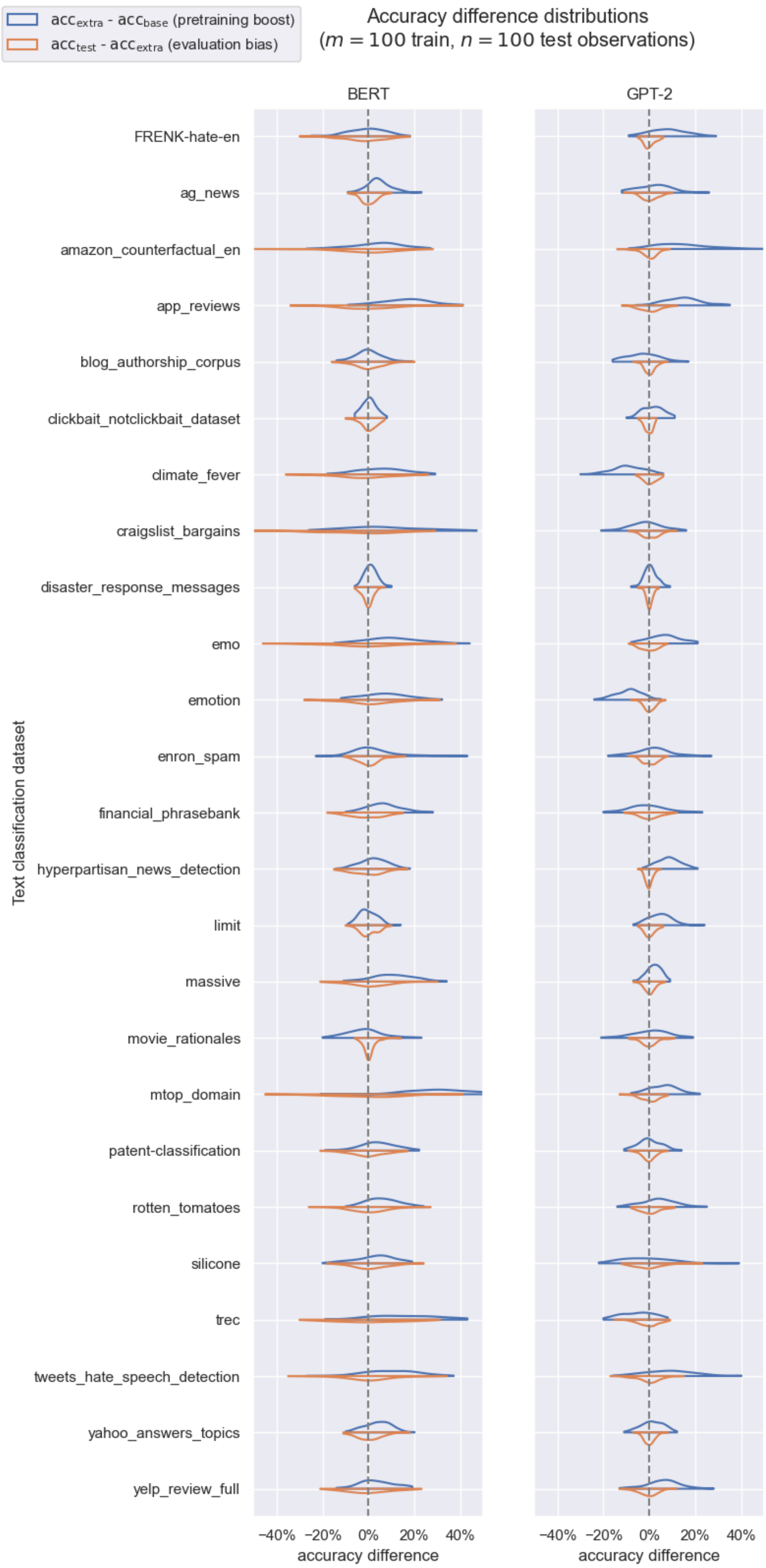

acc_extra - acc_base (pretraining boost)
acc_test - acc_extra (evaluation bias)
Accuracy difference distributions
(m = 50 train, n = 500 test observations)
BERT
GPT-2
Text classification dataset
FRENK-hate-en
ag_news
amazon_counterfactual_en
app_reviews
blog_authorship_corpus
clickbait_notclickbait_dataset
climate_fever
craigslist_bargains
disaster_response_messages
emo
emotion
enron_spam
financial_phrasebank
hyperpartisan_news_detection
limit
massive
movie_rationales
mtop_domain
patent-classification
rotten_tomatoes
silicone
trec
tweets_hate_speech_detection
yahoo_answers_topics
yelp_review_full
−40% −20% 0% 20% 40%
accuracy difference
−40% −20% 0% 20% 40%
accuracy difference

Figure 11



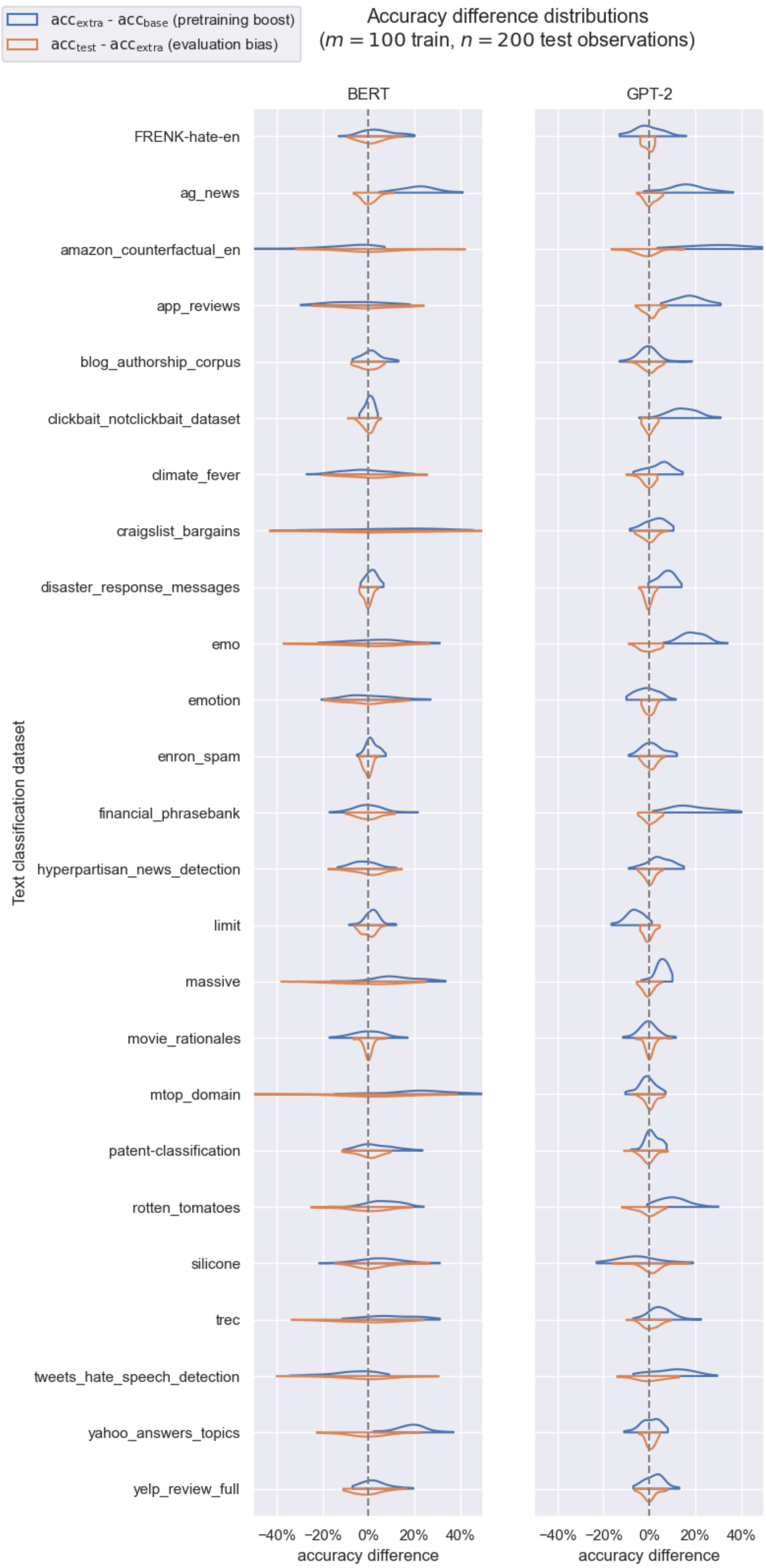
acc_{extra} - acc_{base} (pretraining boost)
acc_{test} - acc_{extra} (evaluation bias)
Accuracy difference distributions
($m = 100$ train, $n = 50$ test observations)
BERT
GPT-2
Text classification dataset
FRENK-hate-en
ag_news
amazon_counterfactual_en
app_reviews
blog_authorship_corpus
clickbait_notclickbait_dataset
climate_fever
craigslist_bargains
disaster_response_messages
emo
emotion
enron_spam
financial_phrasebank
hyperpartisan_news_detection
limit
massive
movie_rationales
mtop_domain
patent-classification
rotten_tomatoes
silicone
trec
tweets_hate_speech_detection
yahoo_answers_topics
yelp_review_full
−40% −20% 0% 20% 40%
accuracy difference

Figure 12



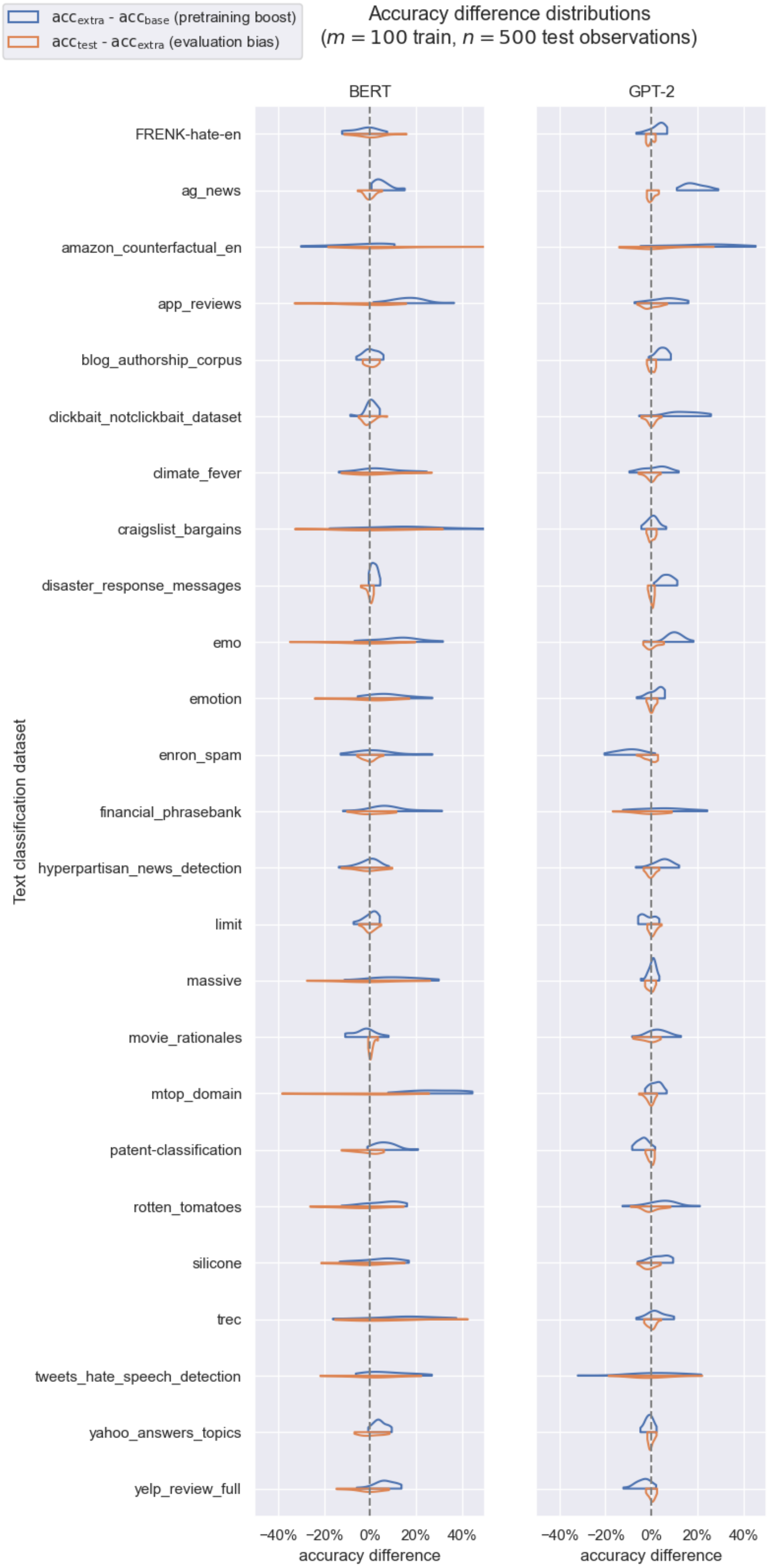
$\mathrm{acc}_{\mathrm{extra}}$ - $\mathrm{acc}_{\mathrm{base}}$ (pretraining boost)
$\mathrm{acc}_{\mathrm{test}}$ - $\mathrm{acc}_{\mathrm{extra}}$ (evaluation bias)
Accuracy difference distributions
($m = 100$ train, $n = 100$ test observations)
BERT
GPT-2
Text classification dataset
FRENK-hate-en
ag_news
amazon_counterfactual_en
app_reviews
blog_authorship_corpus
clickbait_notclickbait_dataset
climate_fever
craigslist_bargains
disaster_response_messages
emo
emotion
enron_spam
financial_phrasebank
hyperpartisan_news_detection
limit
massive
movie_rationales
mtop_domain
patent-classification
rotten_tomatoes
silicone
trec
tweets_hate_speech_detection
yahoo_answers_topics
yelp_review_full
−40% −20% 0% 20% 40%
accuracy difference
−40% −20% 0% 20% 40%
accuracy difference

Figure 13



acc_extra - acc_base (pretraining boost)
acc_test - acc_extra (evaluation bias)
Accuracy difference distributions
(m = 100 train, n = 200 test observations)
BERT
GPT-2
Text classification dataset
FRENK-hate-en
ag_news
amazon_counterfactual_en
app_reviews
blog_authorship_corpus
clickbait_notclickbait_dataset
climate_fever
craigslist_bargains
disaster_response_messages
emo
emotion
enron_spam
financial_phrasebank
hyperpartisan_news_detection
limit
massive
movie_rationales
mtop_domain
patent-classification
rotten_tomatoes
silicone
trec
tweets_hate_speech_detection
yahoo_answers_topics
yelp_review_full
−40% −20% 0% 20% 40%
accuracy difference

Figure 14



acc_{extra} - acc_{base} (pretraining boost)
acc_{test} - acc_{extra} (evaluation bias)
Accuracy difference distributions
($m = 100$ train, $n = 500$ test observations)
BERT
GPT-2
Text classification dataset
FRENK-hate-en
ag_news
amazon_counterfactual_en
app_reviews
blog_authorship_corpus
clickbait_notclickbait_dataset
climate_fever
craigslist_bargains
disaster_response_messages
emo
emotion
enron_spam
financial_phrasebank
hyperpartisan_news_detection
limit
massive
movie_rationales
mtop_domain
patent-classification
rotten_tomatoes
silicone
trec
tweets_hate_speech_detection
yahoo_answers_topics
yelp_review_full
−40% −20% 0% 20% 40%
accuracy difference

Figure 15