2025 IEEE/ACM International Conference on Automation of Software Test (AST) | 979-8-3315-0179-2/25/$31.00 ©2025 IEEE | DOI: 10.1109/AST66626.2025.00017

# Adaptive Probabilistic Operational Testing for Large Language Models Evaluation

Ali Asgari
*Department of Software Technology*
*Delft University of Technology*
Delft, The Netherlands
A.Asgari-2@tudelft.nl

Antonio Guerriero, Roberto Pietrantuono, Stefano Russo
*Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione*
*Università degli Studi di Napoli Federico II*
Napoli, Italy
{antonio.guerriero, roberto.pietrantuono, sterusso}@unina.it

***Abstract*—Large Language Models (LLM) empower many modern software systems, and are required to be highly accurate and reliable. Evaluating LLM poses challenges due to the high costs of manual labeling and of validation of labeled data.**

**This study investigates the suitability of probabilistic operational testing for effective and efficient evaluation of LLM, focusing on a case study with DistilBERT. To this aim, we adopt an existing framework (DeepSample) for Deep Neural Network (DNN) testing and adapt it to the LLM domain by introducing auxiliary variables tailored to LLM and classification tasks.**

**Through a comprehensive evaluation, we demonstrate how sampling-based operational testing can yield reliable LLM accuracy estimates and effectively expose failures, or, under testing budget constraints, it can find a trade off between accuracy estimation and failure exposure. The experimental results, using DistilBERT on three sentiment analysis datasets, show that sampling-based methods can provide cost effective and reliable operational accuracy assessment for LLM. These findings offer practical insights for testers and help address critical gaps in current LLM evaluation practices.**

***Index Terms*—Software testing, Large Language Models, Sampling, LLM evaluation**

## I. Introduction

Large Language Models (LLM) – an integral part of many modern software systems - necessitate thorough evaluation methods to ensure their accuracy and reliability [2]. LLM evaluation serves for performance tuning, for assessment, and for iterative improvements, yet it presents distinct challenges, particularly due to the high cost of labeling data and the need to validate labeled data for quality assurance.

Various techniques are available for LLM evaluation [2, 8], including use of benchmarks, automatic generation of test inputs, human evaluation through manual prompting or crowdsourcing, random testing, and methods specialized for application fields such as biology, law and finance. We advocate the use of probabilistic sampling for effective and efficient evaluation of LLM, claiming its ability to provide high-confidence LLM accuracy estimates while minimizing the number of labeled samples required. Building on past work on *operational testing* (a pillar in software reliability engineering [15]) for Deep Neural Networks (DNN) [6, 7, 12], we propose to adapt and assess it for LLM. In this study we focus on its application to DistilBERT [19], a lighter and more task-specific version of BERT [4].

DNN accuracy estimation through operational testing, by means of small representative subsets of operational datasets, was proposed by Li *et al.* [12]. Guerriero *et al.* [6] leveraged adaptive sampling [9] for joint estimation of DNN accuracy and exposure to mispredictions. Subsequently, they introduced DeepSample [7], a more general framework for evaluating deep Machine Learning models. DeepSample uses sampling theory to build small yet representative test sets, reducing labeling costs while providing unbiased accuracy estimates.

This experimental study investigates the suitability of probabilistic operational testing for LLM evaluation. To this aim, we tailor the DeepSample framework to the scale and characteristics of LLM. We experiment several advanced statistical sampling techniques, which can leverage *auxiliary variables* to enhance LLM evaluation. We embed new LLM-specific variables into the framework, this way evaluating many alternative strategies (i.e., combinations sampling technique-auxiliary variable) in their ability to: *1)* minimize labeling costs by proper sampling; *2)* validate labeled data with high confidence to ensure quality; *3)* expose errors, for LLM fine tuning and retraining. Through the comprehensive analysis of a case study, we compare the strategies, and we discuss crucial choices in LLM probabilistic operational testing.

The experiments use DistilBERT on a sentiment analysis task on three datasets. The results show that the choice of sampling method and auxiliary variable plays a crucial role in achieving three key objectives: (1) high-confidence, unbiased accuracy estimates with significantly reduced labeling and validation efforts; (2) effective identification of model mispredictions, which are critical for debugging and re-training, and (3) sensitivity to increasing sampling budgets, with positive trends in error minimization and failure exposure as budgets grow. The findings highlight the suitability of sampling-based methods for LLM evaluation in sentiment analysis tasks.

## II. Related work

The scientific literature on LLM evaluation is rather large, with a variety of methods and benchmarks proposed to assess the accuracy and other performance metrics of LLM [2, 8]. Within this literature, statistical testing is a very little investigated research topic. For reasons of space, we focus here solely on statistical testing techniques, which have been shown

2833-9061/25/$31.00 ©2025 IEEE
DOI 10.1109/AST66626.2025.00017

to perform well on DNN models, but whose application in typical tasks of LLM, like Natural Language Processing and sentiment analysis, has not been much studied yet [11].

Probabilistic sampling has long been employed in *operational testing* (OT) to estimate software reliability by selecting test cases according to an expected *operational profile*, reflecting real-world usage [3, 20]. OT – a fundamental Software Reliability Engineering approach - enables efficient post-deployment reliability assessment [15]. Building on its principles, Cai *et al.* [1] introduced Adaptive Testing, which adaptively selects test cases from different partitions to minimize variance in reliability estimates, later enhanced by a gradient-based variant for more precise variance reduction [14]. Stratified sampling, another technique in reliability assessment, further improves accuracy by dividing the operational profile into strata and sampling proportionally, ensuring comprehensive coverage of diverse usage patterns [17].

More recent approaches have refined these sampling methods. Pietrantuono et al. [16] formalized unequal probability sampling schemes, achieving greater efficiency in reliability estimation. Li et al. [12] focused on DNN operational accuracy using Cross-Entropy Sampling (CES) to select samples that reflect the distribution of failing examples, enhancing accuracy without full dataset evaluation.

Guerriero et al. [6] identified inefficiencies in uniformly sampling operational inputs for DNN, particularly in high-accuracy models where most inputs are correctly classified. Their DeepEST method, based on adaptive sampling for rare populations [9], emphasizes sampling challenging cases likely to reveal model weaknesses, thus concentrating labeling on the most informative inputs. This targeted approach maintains unbiased results while improving testing efficiency.

By extending these sampling principles, this study applies adaptive and probabilistic sampling to LLM testing, offering a scalable solution for accurate and cost-effective evaluation in increasingly complex model architectures.

## III. LLM PROBABILISTIC TESTING

### A. Formulation

Let $\mathbf{M}$ denote the model (LLM) under evaluation, and:

- $D$ be the operational dataset - a large set of examples with unknown labels, built from inputs to the model M in the operational phase. Its size is $N = |D|$.
- $\mathbf{T} \subseteq \mathbf{D} = \{t_1, \ldots, t_n\}$ be the subset of examples sampled from $\mathbf{D}$ to estimate the LLM accuracy. $\mathbf{T}$ can also serve to enhance the training set for improving the LLM performance in subsequent releases. Its size $n = |\mathbf{T}| \ll N$ is the *testing budget*. When a sample $t_i$ is fed to the LLM, a human oracle provides the expected output, which is then compared to the actual output. The comparison yields the binary result $z_i$ indicating whether the actual and expected labels match ($z_i$=1) or not ($z_i$=0).
- $\theta = \Pr(z_i = 0)$, $i = 1, \ldots, |D|$, be the true failure probability for an example sampled from the operational dataset in classification tasks. This corresponds to the true (but unknown) dataset-level failure probability, $\theta = \frac{1}{N}\sum_{i=1}^{N}(1 - z_i)$. The accuracy of the model is defined as $\xi = 1 - \theta$, with its estimate denoted by $\hat{\xi}$.

The goal of LLM probabilistic testing is to sample $t_i \in \mathbf{T}$, under the constraint of the test budget $n$, so that $\hat{\xi}$ provides an unbiased, low-variance estimate of $\xi$. Achieving low variance while exposing as many failures as possible enhances confidence in the estimate of the LLM's accuracy.

We adapt to the LLM domain the DeepSample framework [7], that can encompass a variety of sampling techniques leveraging knowledge about inputs to sample from the dataset. This knowledge is conveyed by some *auxiliary variable* $\chi$, such as classifier confidence scores [13]. Such auxiliary variables are correlated with the failure probability $\theta$ and can improve estimation accuracy by guiding the sampling process.

### B. Sampling techniques

We study seven sampling techniques for LLM evaluation:

- **Simple Random Sampling (SRS)**: A baseline method where all examples have equal selection probability, providing a straightforward and unbiased estimate [12].
- **Simple Unequal Probability Sampling (SUPS)**: It uses auxiliary variables to assign selection probabilities, prioritizing examples with higher failure likelihood [7, 13].
- **RHC-Sampling (RHC-S)**: It samples with unequal probability and *without replacement*, grouping and weighting examples for accurate estimations [7, 18].
- **Stratified Simple Random Sampling (SSRS)**: It divides the dataset into partitions based on the variance of an auxiliary variable, then samples from partitions, aiming to reduce the estimation error [7, 13].
- **Gradient-Based Sampling (GBS)**: It selects samples adaptively, based on variance reduction through gradient-based adjustments [7, 14].
- **Two-stage Unequal Probability Sampling (2-UPS)**: It uses an auxiliary variable to define partitions, then samples from partitions with unequal probability for representative selection [7].
- **Deep neural networks Enhanced Sampler for operational Testing (DeepEST)**: It maximizes failure detection by adaptive sampling, balancing accuracy estimation with a high capture rate of failing examples [6].

Apart from the baseline SRS, the techniques can be categorized in three groups (listed in Table I):

- SSRS, GBS and 2-UPS are **partitioning** techniques, namely they use the auxiliary variable to divide the input space into partitions, to then sample from partitions;

TABLE I: *Categories of sampling techniques*

| Category | SRS | SUPS | RHC-S | SSRS | GBS | 2-UPS | DeepEST |
|---|---|---|---|---|---|---|---|
| **Partitioning** | | | | ✓ | ✓ | ✓ | |
| **Unequal selection** | | ✓ | ✓ | | | ✓ | ✓ |
| **Without replacement** | | | ✓ | ✓ | | ✓ | ✓ |

- SUPS, RHC-S, 2-UPS and DeepEST use **unequal selection**: they need a proper statistical estimator of LLM accuracy to correct bias due to unequal probabilities;
- RHC-S, SSRS, 2-UPS and DeepEST sample **without replacement**, that is, inputs cannot be sampled twice.

### C. Auxiliary variables for LLM classification tasks

We investigate two auxiliary variables for LLM classification tasks: Confidence and Prediction Entropy. Their choice is borrowed from own previous work and from literature on DNN probabilistic testing [6, 7, 12], and validated through preliminary experiments on the subject here considered.

**Confidence**: It is defined as the highest probability in the output vector of an LLM. For binary classification, let $o$ represent the predicted probability for the positive class. Confidence is computed as $o$ if $o \geq 0.5$, or $1 - o$ otherwise, reflecting the model's certainty about its prediction.

**Prediction Entropy (PE)**: Defined as: $PE = -\sum_i p_i \cdot \log(p_i)$, where $p_i$ is the predicted probability for the $i$-th class, PE quantifies the uncertainty in the model's predictions.

Both *Confidence* and *Prediction Entropy* do not rely on true labels. These variables guide labeling efforts by prioritizing instances with low confidence or high uncertainty, thereby enhancing cost effectiveness. This approach ensures that testing efforts focus on informative cases, reducing redundant labeling while maximizing the impact of low testing budgets.

The following experimentation investigates which strategies (sampling technique / auxiliary variable) are best suited for LLM evaluation.

## IV. Subject, datasets and metrics

The LLM subject for a sentiment analysis task is DistilBERT [19], a lightweight and efficient variant of the well known BERT pre-trained open-source machine learning model for natural language processing [4], fine-tuned for rapid and accurate sentiment classification tasks on the SST-2 dataset. [1] Although this version is task-specific, the architecture remains consistent with the original DistilBERT model.

The experiments use the three datasets listed in Table II. SST-2 is a benchmark dataset for binary sentiment classification.[2] IMDb is a large dataset of 50,000 movie reviews commonly used for sentiment classification tasks.[3] To broaden experiments, we included IMDb_3000, a curated subset of 3,000 items selected from IMDb for binary sentiment classification.[4] All experimental artifacts, including code and results, are available in the replication package for reproducibility.

TABLE II: Datasets for experiments

| Dataset | Size $N$ | Task | Accuracy |
|---|---|---|---|
| `SST-2` | 1,821 | Sentiment Analysis | 0.9225 |
| `IMDb` (full dataset) | 50,000 | Sentiment Analysis | 0.8896 |
| `IMDb_3000` | 3,000 | Sentiment Analysis | 0.8990 |

[1] www.huggingface.co/distilbert/distilbert-base-uncased-finetuned-sst-2-english
[2] www.github.com/YJiangcm/SST-2-sentiment-analysis/tree/master/data
[3] www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews
[4] www.huggingface.co/datasets/enoreyes/imdb_3000_sphere

The experiments are designed to evaluate three aspects of sampling-based techniques applied to LLM evaluation:

- **Evaluation effectiveness.** The effectiveness of techniques in estimating the operational accuracy of LLM.
- **Failure detection ability.** The ability of techniques to identify instances where the LLM fails.
- **Sensitivity to testing budget.** The influence of the sample size $n = |\mathbf{T}|$ on the performance of techniques.

To quantify the *effectiveness* of the accuracy evaluation, we compute the Root Mean Square Error (RMSE):

$$\text{RMSE} = \sqrt{\frac{\sum_{r=1}^{R} \left(\xi - \hat{\xi}\right)^2}{R}} \tag{1}$$

where $\hat{\xi}$ is the accuracy estimate, $\xi$ is the true accuracy, and $R$ represents the number of experimental rounds. In this study, experiments are conducted for 30 rounds.

The *failure detection ability* is assessed by analyzing the mean and standard deviation of the identified failing examples. This evaluation highlights the robustness of our testing approach in pinpointing problematic instances.

The *sensitivity to the sample size* is studied to analyze the trade-off between number of tests and accuracy of the evaluation, by running experiments with the following values for the *testing budget*: $n = 50, 100, 200, 400, 800$ samples.

## V. Results

### A. Evaluation effectiveness

To determine if the techniques exhibit statistically significant pairwise differences, we run the Friedman test [10] on all auxiliary variables pairs on all datasets. The resulting $p$-value is lower than $\alpha = 0.05$ in all instances, rejecting the null hypothesis, which posits no difference among techniques. In cases where the Friedman test indicate significant differences, a *post hoc* analysis with the non-parametric Dunn test [5] is performed to identify specific pairs of methods that exhibited statistically significant differences. To account for multiple comparisons, the Holm-Bonferroni correction is applied to the $p$-values obtained from Dunn test.

For each pair of strategies identified as significantly different, the RMSE values were aggregated across all relevant records within the dataset. The method with the lower cumulative RMSE is considered to perform better. This ensures that the evaluation considers the overall performance across all instances, providing robust techniques comparison.

The results are in Figure 1, which shows the pairwise comparisons between techniques, with statistical significance assessed using the Dunn/Holm-Bonferroni test. Black squares represent cases where there is no statistically significant difference between the techniques. White squares indicate that the technique on the row outperforms the one on the column. Exact $p$-values are provided in the replication package.[5]

The results reveal consistent trends in the performance of sampling techniques across datasets and auxiliary variables.

[5] https://github.com/leanerr/OperationalTesting4LLMs.

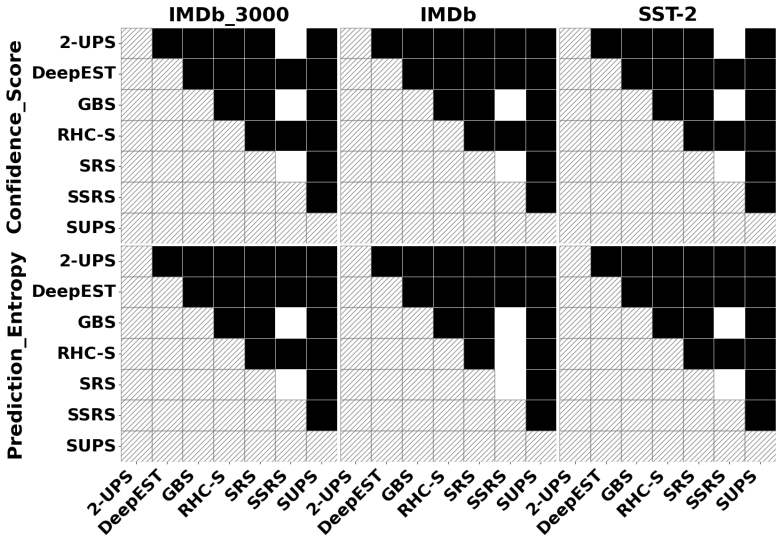


Fig. 1: *Evaluation effectiveness*: pairwise statistical comparison of sampling techniques

Methods 2-UPS, GBS, and SRS frequently outperform SSRS. Notably, GBS and SRS excel in both *Confidence* and *Prediction Entropy* by achieving lower values than SSRS, underscoring their ability to produce confident and less uncertain predictions. RHC-S shows strong performance in *Prediction Entropy* against SSRS, especially on the `IMDb` dataset. Overall, while 2-UPS, GBS, SRS, and RHC-S outperform SSRS, the degree of their effectiveness varies with the auxiliary variable used. RHC-S excels in scenarios relying solely on *Prediction Entropy*, whereas GBS and SRS demonstrate strong performance in contexts where either *Prediction Entropy* or *Confidence* serve as critical metrics. Additionally, 2-UPS often outperforms SSRS, particularly in datasets `IMDb_3000` and `SST-2`. These results highlight the adaptability of each method to the specific demands of varying auxiliary variables.

Figure 2 shows the performance of sampling methods on the three datasets under a constrained budget of 200 tests. By focusing on the RMSE values for the auxiliary variables, this plot provides insights into how well different methods maintain accuracy, reliability, and consistency during sampling.

There are some slight differences between the results with the two auxiliary variables. 2-UPS on `SST-2` with *Confidence* exhibits the lowest RMSE; with *Prediction Entropy* it ranks as the second largest. On `IMDb`, SRS and GBS perform better than others with only slight differences, and this holds true for both auxiliary variables. On `IMDb_3000`, the performance of GBS, SRS, and 2-UPS with *Confidence* is quite similar and better than other methods, whereas for *Prediction Entropy*, GBS and RHC-S demonstrate excellent performance, with SRS emerging as the best. Looking at the results on all three datasets, it is evident that they are closely aligned, underscoring the general reliability of DeepEST, GBS, RHC-S, SUPS, SRS and 2-UPS against SSRS with *Confidence* and *Prediction Entropy*. These strategies show lower RMSE values, reinforcing their suitability for applications demanding high confidence, accuracy, and dynamic adaptability. In general, **GBS** and **SRS** offering strong error with both auxiliary variables. **RHC-S** excels with *Prediction Entropy*, while **2-UPS** is better suited for *Confidence*.

### *B. Failure detection ability*

Table III shows the failure rates of the sampling methods per dataset and auxiliary variable. The results reflect the mean and standard deviation of failure rates under a fixed budget of 200, offering insights into the robustness, reliability, and variability of each method.

The following techniques excel at detecting failures:

- **SSRS**: demonstrates exceptional performance in failure detection. It identifies over 70 failures across all datasets, including 79.4 on `IMDb_3000` and 79.2 on `IMDb` with *Confidence*, underscoring its reliability in high failure identification.

TABLE III: *Failure detection ability*: Failure Mean and Standard Deviation per auxiliary variable, technique and dataset

| Aux_Var | Technique | SST-2 | IMDb_3000 | IMDb |
|---|---|---|---|---|
| Confidence | 2-UPS | 16.6 / 3.3 | 20.4 / 3.9 | 23.5 / 5.7 |
| Confidence | DeepEST | 35.9 / 2.9 | **72.2 / 3.0** | **76.4 / 6.3** |
| Confidence | GBS | 18.2 / 3.6 | 20.9 / 3.6 | 22.9 / 3.3 |
| Confidence | RHC-S | 13.3 / 2.9 | 16.5 / 4.5 | 18.7 / 4.1 |
| Confidence | SRS | 16.7 / 3.9 | 20.2 / 4.2 | 21.9 / 3.5 |
| Confidence | SSRS | **70.0 / 0.8** | **79.4 / 4.5** | **79.2 / 6.8** |
| Confidence | SUPS | **85.3 / 8.4** | **74.5 / 6.6** | **78.8 / 7.1** |
| Entropy | 2-UPS | 3.5 / 1.9 | 9.9 / 3.3 | 9.2 / 3.4 |
| Entropy | DeepEST | 37.9 / 2.6 | **73.3 / 3.6** | **75.8 / 5.9** |
| Entropy | GBS | 15.4 / 3.3 | 21.2 / 4.7 | 22.0 / 4.0 |
| Entropy | RHC-S | 49.8 / 4.8 | 62.4 / 4.6 | **70.1 / 6.3** |
| Entropy | SRS | 16.1 / 3.3 | 20.5 / 3.8 | 23.0 / 5.1 |
| Entropy | SSRS | **70.7 / 2.4** | **70.6 / 4.7** | **70.1 / 5.7** |
| Entropy | SUPS | **70.2 / 7.4** | 67.9 / 6.0 | **71.5 / 6.6** |

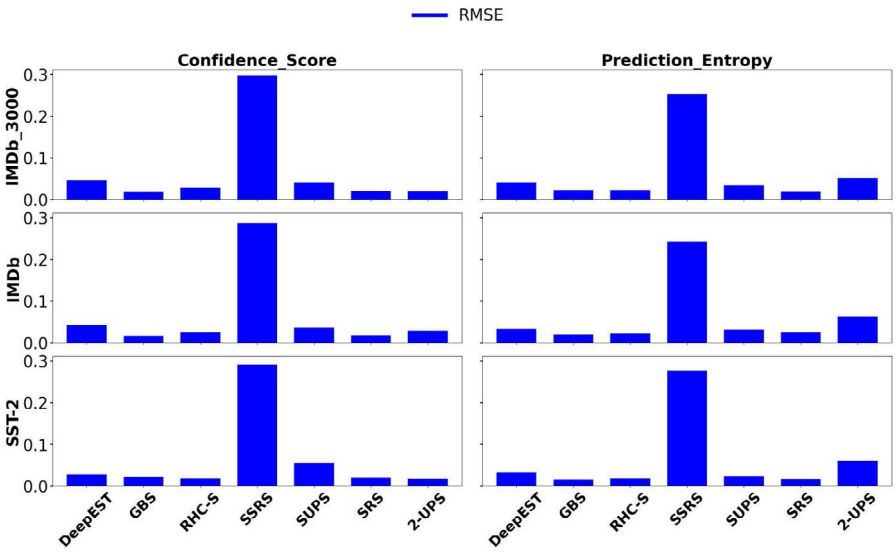


Fig. 2: *Evaluation effectiveness*: RMSE of sampling techniques per dataset and auxiliary variable

- **SUPS**: achieves the highest failure detection rates, particularly under the *Confidence* variable, with 85.3 failures on `SST-2`. It also excels with *Prediction Entropy*, detecting 71.5 failures on `IMDb` and 70.2 on `SST-2`, highlighting its robustness in exposing model weaknesses.
- **DeepEST**: maintains strong failure detection ability, with high rates under *Confidence* and *Prediction Entropy*. It identifies 76.4 failures on `IMDb` and 73.3 on `IMDb_-3000` with *Prediction Entropy*, demonstrating its effectiveness in addressing challenging cases.

Conversely, the following methods detect significantly fewer failures, indicating weaker performance in identifying complex issues within the datasets:

- **2-UPS**: while it performs moderately well under the *Confidence* variable, detecting 23.5 failures on `IMDb` and 20.4 on `IMDb_3000`, its performance is less competitive with *Prediction Entropy*, making it less suitable for rigorous failure detection.
- **SRS**: provides balanced but relatively lower failure detection rates compared to stronger methods. For instance, it identifies 21.9 failures on `IMDb` with *Confidence* and 20.5 on `IMDb_3000` with *Prediction Entropy*.
- **GBS**: maintains consistent yet lower detection rates across datasets. While it performs reasonably well with *Prediction Entropy*, detecting 22.0 failures on `IMDb`, it is outperformed by other methods in most scenarios.
- **RHC-S**: although it shows good performance under *Prediction Entropy*, detecting 70.1 failures on `IMDb`, its detection rates on other datasets and auxiliary variable remain limited.

In general, **SUPS**, **DeepEST**, and **SSRS** are highly effective at exposing failures paired with *Confidence* or *Prediction Entropy*. When combined with *Prediction Entropy*, **RHC-S** too proves to be a reliable choice for failure detection.

### C. Sensitivity to sample size

*Error sensitivity*

Figure 3 presents the sensitivity of techniques to increasing sample size, as measured by the RMSE metric. Lower RMSE values indicate higher predictive accuracy. Across all techniques, RMSE values at the maximum budget of 800 samples are consistently lower than those at the budget of 50, highlighting the importance of increasing sample size in improving error minimization. For 2-UPS, *Confidence* outperforms *Prediction Entropy* for all three datasets, demonstrating its reliability with this auxiliary variable. For SRS, GBS, and DeepEST, both auxiliary variables (*Confidence* and *Prediction Entropy*) performed very well, showcasing their versatility and robustness. SUPS exhibits strong early-stage performance but stabilizes at higher budgets and exhibited better performance with *Prediction Entropy*, particularly at larger budgets. On the other hand, while SSRS has the highest RMSE values, it still benefits from increasing sample size, showing reductions in error over larger budgets, particularly when paired with *Prediction Entropy*. Also RHC-S shows steady improvement with increasing sample size, particularly with *Prediction Entropy*.

*Failure sensitivity*

Figure 4 and Table IV show the performance of techniques in detecting failures as the sampling budget increases, with the two auxiliary variables on the three datasets. These results highlight the sensitivity of techniques to detect failures to changes in the sampling budget, with $F_{800/50}$ values providing a quantifiable metric for sensitivity. The general of techniques is that increasing the sampling budget leads to a significant improvement in failure detection, as reflected by consistently rising failure means. This behavior is particularly evident in methods like SUPS, where $F_{800/50}$ values demonstrate strong proportional growth.

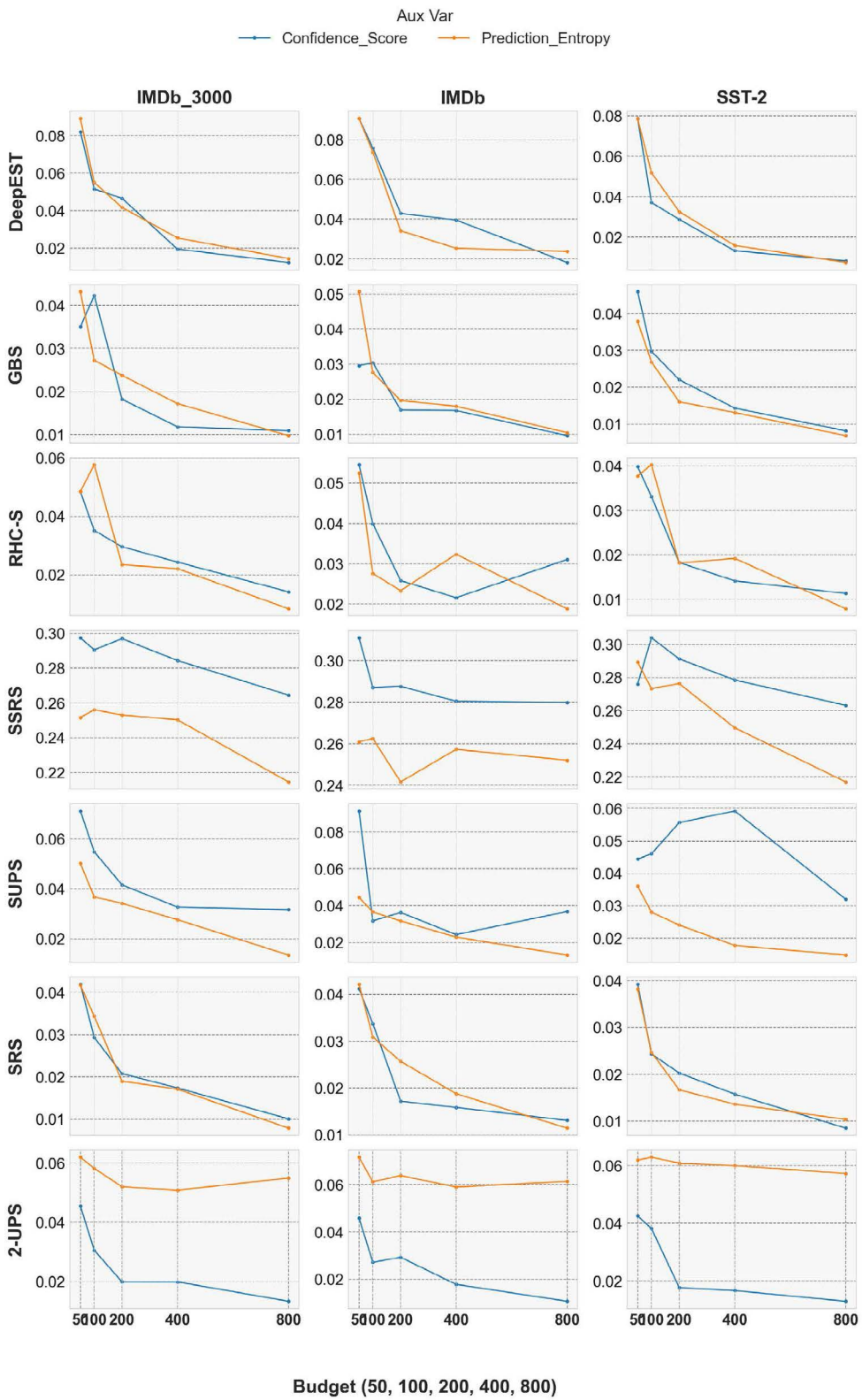


Fig. 3: *Error sensitivity*: Sensitivity of RMSE to the testing budget

*1) Sensitivity of Each Method in failure detection:* **SUPS** exhibits exceptional performance across all datasets and auxiliary variables. For *Confidence*, the failure detection mean rises sharply from 21.87 at 50 samples to 350.50 at 800 samples for SST-2, with a high $F_{800/50}$ value of 16.03. Similarly, for *Prediction Entropy*, it demonstrates a robust increase from 17.27 to 285.53 for the same dataset ($F_{800/50} = 16.54$).

Comparable trends are observed for IMDb and IMDb_-3000, cementing SUPS as one of the top-performing methods due to its dynamic adaptability and superior sensitivity.

**DeepEST** achieves high failure detection on all datasets. For *Confidence*, it scales from 22.50 to 74.67 (SST-2, $F_{800/50} = 3.32$), and from 17.17 to 123.27 (IMDb_3000, $F_{800/50} = 7.18$). Similarly, for *Prediction Entropy*, it maintains consistent performance on datasets, e.g., on IMDb the failure detection mean increases from 17.80 to 305.20 ($F_{800/50} = 17.15$).

**RHC-S** demonstrates a mixed performance. For *Prediction Entropy*, it performs exceptionally well, particularly for IMDb_3000, where the failure detection mean increases from 15.57 to 166.60 ($F_{800/50} = 10.70$). However, its performance for *Confidence* is less competitive, with values such as 2.63 to 54.87 for SST-2 ($F_{800/50} = 20.84$). This indicates that RHC-S is more sensitive to *Prediction Entropy*.

**SSRS** achieves competitive performance, particularly on IMDb and IMDb_3000. With *Confidence*, it exhibits a significant increase from 19.63 to 139.20 (IMDb_3000, $F_{800/50} = 7.09$) and from 20.83 to 311.77 (IMDb, $F_{800/50} = 14.96$). However, its performance is slightly worse *Prediction Entropy*, where it still shows increases but with higher sensitivity values on IMDb_3000 ($F_{800/50} = 9.48$).

**2-UPS** demonstrates moderate performance. For *Confidence*, its failure detection mean rises from 3.90 to 68.33 for SST-2 ($F_{800/50} = 17.52$). However, it underperforms for *Prediction Entropy*, such as on SST-2, where it scales only from 0.93 to 16.37 ($F_{800/50} = 17.54$).

**GBS** and **SRS** exhibit moderate growth in failure detection but remain less competitive compared to SUPS or DeepEST. For example, GBS achieves $F_{800/50}$ values of 14.16 and 17.55 for *Confidence* and *Prediction Entropy*, respectively (SST-2). Similarly, SRS demonstrates modest sensitivity, with $F_{800/50} = 15.41$ and 14.50 for the same variables.

*2) Top-performing techniques (failure detection) and sensitivity:* From the analysis, SUPS and DeepEST emerge as the most effective methods due to their high failure detection means and sensitivity to budget increases. SUPS, in particular, achieves the highest $F_{800/50}$ values across most datasets and auxiliary variables, reflecting its adaptability and robustness. DeepEST also demonstrates strong performance and its sensitivity ensures a consistent increase in failures detected.

RHC-S, performing exceptionally for *Prediction Entropy*, shows variable results for *Confidence*. While competitive, SSRS shows slightly less adaptability with *Prediction Entropy*. Simpler methods like SRS, while more predictable, fail to capitalize on larger budgets. Advanced sampling algorithms achieve superior failure detection performance.

For failure detection, auxiliary variables exhibit different sensitivity to budget increases:
- *Confidence* improves steadily across all methods and datasets, showing reliable growth as the budget increases;
- *Prediction Entropy* is also effective but reacts more sharply to budget changes, variations in $F_{800/50}$ values, especially for methods like SUPS and RHC-S.

## VI. Discussion

### A. Trade off between error minimization and failure detection

Figure 5 shows that with *Prediction Entropy* the techniques DeepEST, SUPS, and RHC-S stand out as the most effective in achieving a trade-off between these two objectives. These methods exhibit both low RMSE and high failure detection performance, making them the most favourable choices in this context. With *Confidence*, DeepEST and SUPS excel in the trade-off, showcasing consistent performance. However, RHC-S, while performing comparably in some instances, does not show a significant advantage over other methods.

The remaining methods demonstrate minimal performance differences under *Confidence*. SSRS, despite being one of the top-performing methods in failure detection, is notably very bad in error minimization, with significantly higher RMSE values compared to all other methods. This highlights a limit in its applicability in scenarios requiring balanced performance across both dimensions.

### B. Comparison of auxiliary variables

The experiments demonstrate that both auxiliary variables, *Confidence* and *Entropy*, perform well across different datasets and metrics. These label-free auxiliaries, which do not rely on ground-truth labels, offer versatility across various sampling methods. While both auxiliary variables show strong performance, *Prediction Entropy* stands out for achieving better trade-offs between error minimization and failure detection, making it particularly effective in balancing these objectives.

### C. Performance of techniques categories

Findings at a higher abstraction level can be drawn about the performance of techniques, considering the categorization introduced in Section III and Table I: *partitioning*, *unequal selection*, and *without replacement* methods. Indeed, each category demonstrates distinct strengths depending on the evaluation objective (minimizing error, exposing failures, trade-off).

*- Partitioning (SSRS, GBS, 2-UPS)*

Partitioning methods ensure well-distributed sampling by dividing the dataset into partitions. This reduces overall variance and makes them particularly effective for error minimization.

For example, GBS achieves reliable performance for high-confidence evaluations. 2-UPS also demonstrates strong capabilities for minimizing errors, particularly when paired with *Confidence*. However, these methods are less effective for failure detection compared to other categories.

*- Unequal selection (SUPS, RHC-S, 2-UPS, DeepEST)*

Unequal selection methods focus sampling efforts on failure-prone areas, making them highly effective for exposing

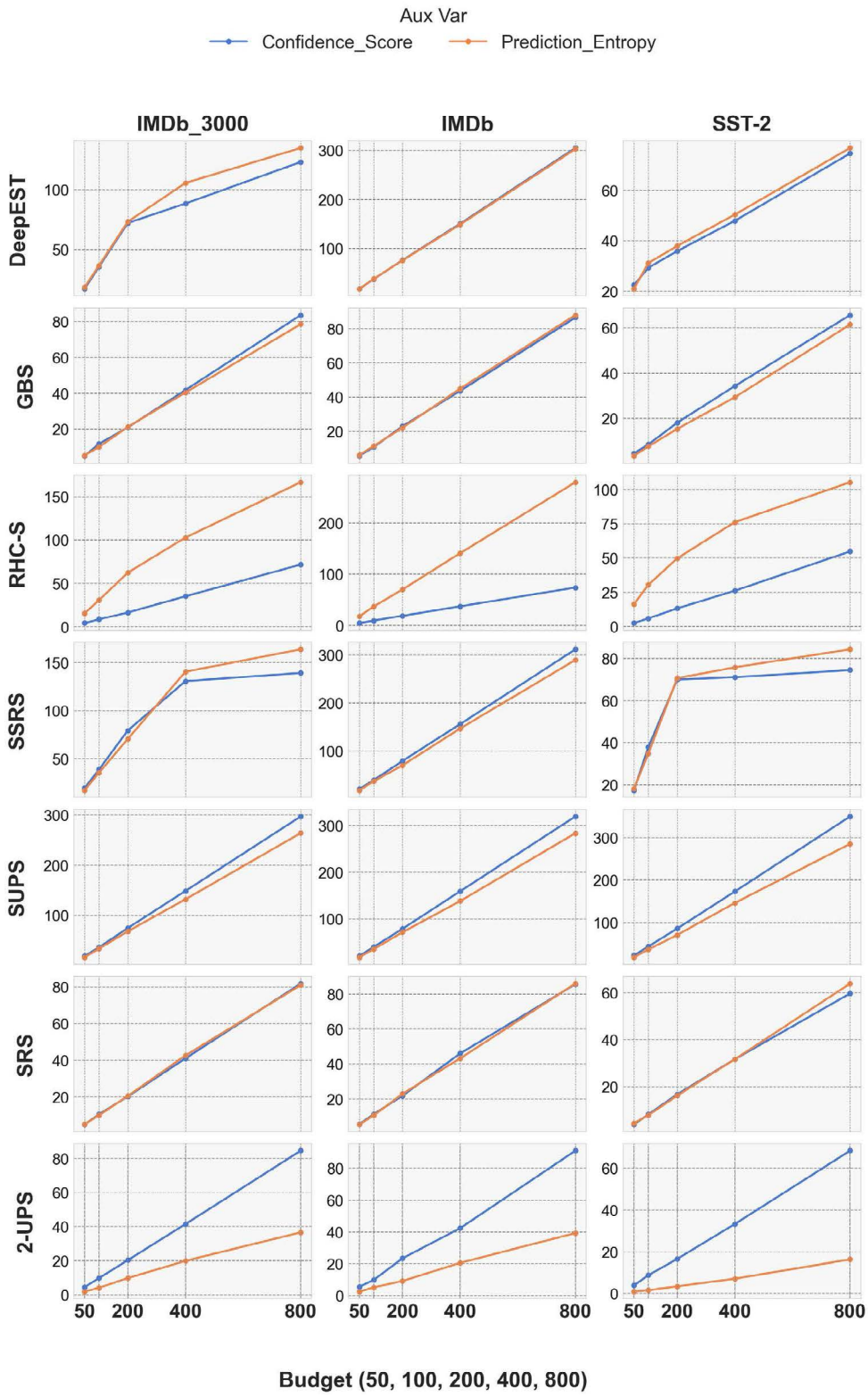


Fig. 4: *Failure sensitivity*: Failure detection ability per testing budget

TABLE IV: *Failure sensitivity:* Sensitivity of techniques failure detection ability per auxiliary variable and dataset

| Dataset | Technique | Mean(min) Confidence / Pred. Entropy | F800/50 Confidence / Pred. Entropy | Mean(max) Confidence / Pred. Entropy |
|---|---|---|---|---|
| **SST-2** | 2-UPS | 3.90 / 0.93 | 17.52 / 17.54 | 68.33 / 16.37 |
| **SST-2** | **DeepEST** | **22.50 / 20.93** | **3.32 / 3.67** | **74.67 / 76.83** |
| **SST-2** | GBS | 4.63 / 3.50 | 14.16 / 17.55 | 65.60 / 61.43 |
| **SST-2** | RHC-S | 2.63 / 16.47 | 20.84 / 6.39 | 54.87 / 105.30 |
| **SST-2** | SRS | 3.87 / 4.40 | 15.41 / 14.50 | 59.60 / 63.80 |
| **SST-2** | **SSRS** | **17.37 / 18.10** | **4.29 / 4.66** | **74.57 / 84.43** |
| **SST-2** | **SUPS** | **21.87 / 17.27** | **16.03 / 16.54** | **350.50 / 285.53** |
| **IMDb_3000** | 2-UPS | 4.57 / 1.93 | 18.52 / 18.97 | 84.57 / 36.67 |
| **IMDb_3000** | **DeepEST** | **17.17 / 18.43** | **7.18 / 7.32** | **123.27 / 135.00** |
| **IMDb_3000** | GBS | 5.00 / 5.30 | 16.73 / 14.82 | 83.63 / 78.53 |
| **IMDb_3000** | RHC-S | 4.33 / 15.57 | 16.56 / 10.70 | 71.77 / 166.60 |
| **IMDb_3000** | SRS | 4.93 / 5.10 | 16.58 / 15.88 | 81.80 / 80.97 |
| **IMDb_3000** | **SSRS** | **19.63 / 17.27** | **7.09 / 9.48** | **139.20 / 163.73** |
| **IMDb_3000** | **SUPS** | **18.70 / 16.00** | **15.89 / 16.49** | **297.10 / 263.83** |
| **IMDb** | 2-UPS | 5.60 / 2.57 | 16.25 / 15.31 | 91.00 / 39.30 |
| **IMDb** | **DeepEST** | **17.80 / 17.80** | **17.15 / 17.02** | **305.20 / 302.97** |
| **IMDb** | GBS | 5.47 / 5.93 | 15.84 / 14.84 | 86.60 / 88.07 |
| **IMDb** | RHC-S | 4.43 / 17.70 | 16.67 / 15.79 | 73.90 / 279.57 |
| **IMDb** | SRS | 5.60 / 5.43 | 15.29 / 15.82 | 85.63 / 85.97 |
| **IMDb** | **SSRS** | **20.83 / 18.23** | **14.96 / 15.88** | **311.77 / 289.63** |
| **IMDb** | **SUPS** | **20.50 / 17.60** | **15.60 / 16.13** | **319.80 / 283.83** |

In bold: top 3 techniques for failure detection ability per dataset.

failures. For instance, SUPS and DeepEST excel in failure detection while maintaining a good balance with error minimization. RHC-S, when paired with *Entropy*, also achieves competitive trade-offs, demonstrating both strong failure detection and reliable error minimization.

- *Without replacement (RHC-S, SSRS, 2-UPS, DeepEST)*

Without replacement methods ensure broader dataset coverage by avoiding duplicate samples, making them particularly effective for failure detection and achieving balanced performance.

SSRS demonstrates the highest effectiveness for failure detection, outperforming other methods in this category. Meanwhile, DeepEST and 2-UPS achieve good trade-off, combining strong failure exposure with reliable error minimization. Increasing the sampling budget further enhances the performance of these methods.

### *D. Actionable hints for practitioners*

In summary, the experimental study allows to draw the following considerations (limited by the threats discussed in next Section) as actionable hints for practitioners aiming to use probabilistic testing for LLM evaluation, depending on the evaluation goal:

1) **Accuracy of the evaluation (error minimization):** Techniques **GBS**, **SRS**, and **RHC-S** are highly effective with *Prediction Entropy*. With *Confidence*, **GBS**, **SRS**, and **2-UPS** perform best.
2) **High failure exposure: SUPS**, **DeepEST**, and **SSRS** excel in exposing failures. When paired with *Prediction Entropy*, **RHC-S** is a reliable option for failure exposure.
3) **Error-failure exposure trade-off: DeepEST** and **SUPS** paired with either auxiliary variables, and **RHC-S** paired with *Prediction Entropy*, offer the best trade-offs.
4) **Sampling budget impact:** Evaluation accuracy and failure detection improve sensibly with the test budget.

## VII. Threats to validity

This study evaluated the use of probabilistic sampling techniques on a single LLM for a sentiment analysis task on three datasets. While the chosen subject, datasets, and task are widely known and commonly used in LLM evaluation, the generalizability of results to other LLM models and tasks remains limited.

The sampling budgets (ranging from 50 to 800) and selected auxiliary variables, though carefully designed, might not fully represent all possible configurations, and alternative setups could lead to different outcomes.

## VIII. Conclusions

We have presented an experimental evaluation of seven probabilistic testing techniques to assess the operational accuracy of LLM. The techniques analyzed, which belong to three categories — partitioning, unequal selection, and without replacement - allow for flexibility to the tester's objectives and the available auxiliary information.

To minimize the LLM accuracy estimation error, techniques **GBS**, **SRS**, and **2-UPS**, paired with *Confidence* auxiliary information, perform exceptionally well in reducing RMSE, making them highly suited when high-confidence estimates are required. **GBS**, **SRS**, and **RHC-S**, paired with *Prediction Entropy*, provide reliable and stable accuracy estimates.

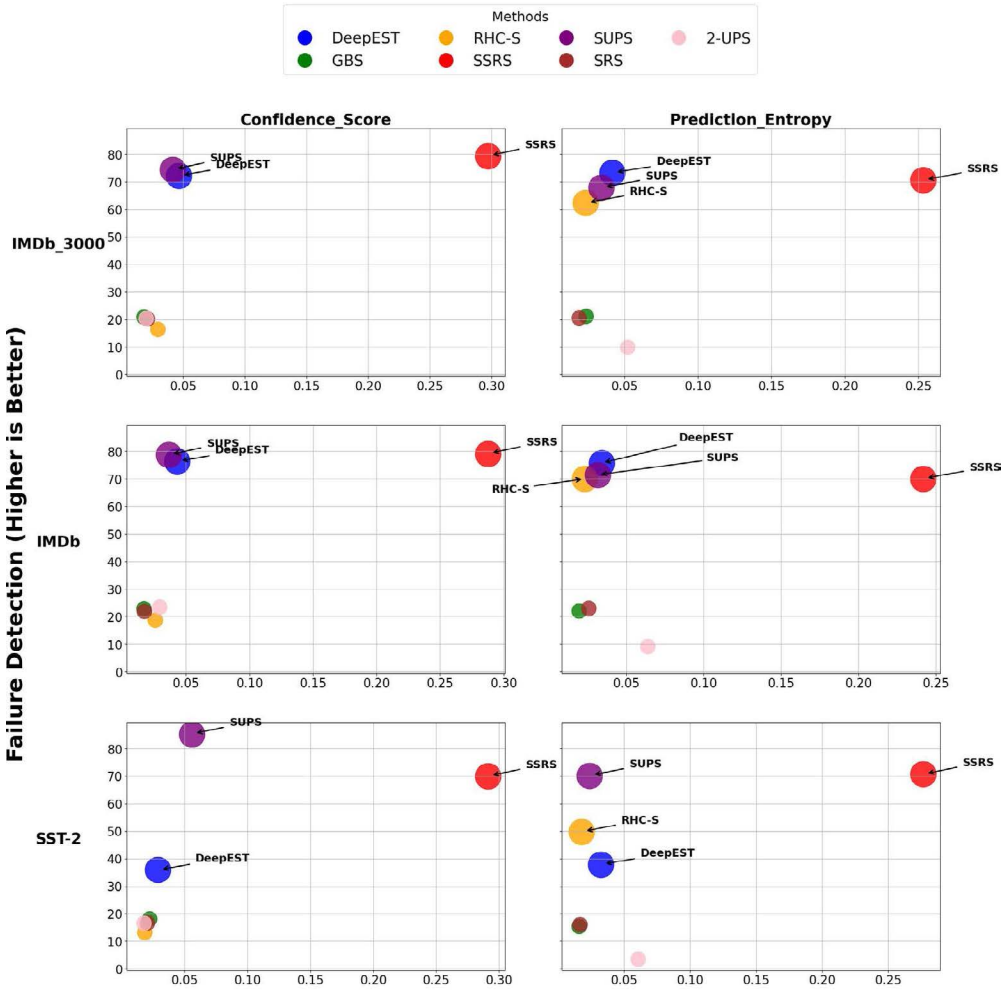


Fig. 5: Trade off between error (RMSE) minimization and failure detection

For high failure detection in testing an LLM, Unequal Selection and Without Replacement techniques **SUPS**, **DeepEST**, and **RHC-S** with *Prediction Entropy* demonstrate strong performance. While **SSRS** achieves excellent results for failure detection too, it suffers from significantly higher RMSE.

For best trade-off between error minimization and failure detection, **DeepEST** and **SUPS**, paired with either experimented auxiliary variables, and **RHC-S**, paired with *Prediction Entropy*, offer the most balanced performance.

The sensitivity analysis confirms that the sampling strategies can enhance significantly both error minimization and failure detection when the sampling budget increases.

Techniques tailored for high-confidence estimates are particularly suited for evaluating LLM against release criteria, or for selecting among competing models.

Methods with strong failure detection ability are ideal for iterative life cycles and to identify vulnerabilities.

Finally, techniques offering balanced trade-offs are good solutions for LLM evaluation in diverse operational contexts.

## IX. Future Work

This study primarily focuses on testing DistilBERT within the context of sentiment analysis tasks, there are several avenues for future research to extend the applicability of the proposed sampling based testing approach. The methodology can be extended to assess other large scale language models, such as GPT or LLaMA, and applied to diverse tasks including question answering, text summarization, and generative tasks, to validate its generalizability.

## X. Data Availability

The results and artifacts for replicating the study are available at: https://github.com/leanerr/OperationalTesting4LLMs

## Acknowledgment

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 871342 "uDEVOPS". The work by Ali Asgari was partially done while he was MSc student at Federico II University of Naples.

## REFERENCES


[1] Kai-Yuan Cai, Chang-Hai Jiang, Hai Hu, and Cheng-Gang Bai. An experimental study of adaptive testing for software reliability assessment. *Journal of Systems and Software*, 81(8):1406–1429, 2008.

[2] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45, 2024.

[3] P. Allen Currit, Michael Dyer, and Harlan D. Mills. Certifying the reliability of software. *IEEE Transactions on Software Engineering*, (1):3–11, 1986.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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 (NAACL-HLT)*. Association for Computational Linguistics, 2019.

[5] Alexis Dinno. Nonparametric pairwise multiple comparisons in independent groups using dunn's test. *The Stata Journal*, 15(1):292–300, 2015.

[6] Antonio Guerriero, Roberto Pietrantuono, and Stefano Russo. Operation is the hardest teacher: estimating DNN accuracy looking for mispredictions. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pages 348–358. IEEE, 2021.

[7] Antonio Guerriero, Roberto Pietrantuono, and Stefano Russo. DeepSample: DNN sampling-based testing for operational accuracy assessment. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering (ICSE)*, pages 1–12. ACM, 2024.

[8] Zishan Guo, Renren Jin, Chuang Liu, Yufei Huang, Dan Shi, Linhao Yu, Yan Liu, Jiaxuan Li, Bojian Xiong, and Deyi Xiong. Evaluating Large Language Models: A Comprehensive Survey. *arXiv preprint arXiv:2310.19736*, 2023.

[9] Daniel G. Horvitz and Donovan J. Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American statistical Association*, 47(260):663–685, 1952.

[10] Ronald L. Iman and James M. Davenport. Approximations of the critical region of the fbietkan statistic. *Communications in Statistics-Theory and Methods*, 9(6):571–595, 1980.

[11] Yuechen Li, Hanyu Pei, Linzhi Huang, and Beibei Yin. A distance-based dynamic random testing strategy for natural language processing DNN models. In *2022 IEEE 22nd International Conference on Software Quality, Reliability and Security (QRS)*, pages 842–853. IEEE, 2022.

[12] Zenan Li, Xiaoxing Ma, Chang Xu, Chun Cao, Jingwei Xu, and Jian Lü. Boosting operational DNN testing efficiency through conditioning. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)*, pages 499–509. ACM, 2019.

[13] Sharon L. Lohr. *Sampling: design and analysis*. Chapman and Hall/CRC, New York, 2021.

[14] Junpeng Lv, Bei-Bei Yin, and Kai-Yuan Cai. On the asymptotic behavior of adaptive testing strategy for software reliability assessment. *IEEE transactions on Software Engineering*, 40(4):396–412, 2014.

[15] John D. Musa. Software reliability-engineered testing. *Computer*, 29(11):61–68, 1996.

[16] Roberto Pietrantuono and Stefano Russo. On adaptive sampling-based testing for software reliability assessment. In *2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE)*, pages 1–11. IEEE, 2016.

[17] Andy Podgurski, Wassim Masri, Yolanda McCleese, Francis G Wolff, and Charles Yang. Estimation of software reliability by stratified sampling. *ACM Transactions on Software Engineering and Methodology*, 8(3):263–283, 1999.

[18] J. N. K. Rao, H. O. Hartley, and W. G. Cochran. On a simple procedure of unequal probability sampling without replacement. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 24(2):482–491, 1962.

[19] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[20] Richard W. Selby, Victor R. Basili, and F. Terry Baker. Cleanroom software development: An empirical evaluation. *IEEE Transactions on Software Engineering*, SE-13:1027–1037, 1987.