Detecting LLM-Generated Text: A Parameter-Efficient Approach

Mourad CHIKHI (mourad.chikhi@ensae.fr) **Côme NADLER** (come.nadler@ensae.fr)

Abstract

Detecting whether a text is written by a human or generated by a large language model (LLM) is crucial for content authenticity, education, and security. We systematically compare three levels of text representation on an English benchmark (WebText vs GPT-2 outputs): (1) TF–IDF bag-of-n-grams, (2) frozen XLM-RoBERTa CLS embeddings with a logistic regression classifier, and (3) parameter-efficient fine-tuning of XLM-RoBERTa via LoRA. Each pipeline is evaluated on a held-out 10 % subset (approximately 25 k examples), then robustly assessed through stratified 5-fold cross-validation and conventional metrics (accuracy, F1-score, ROC-AUC). TF–IDF achieves 84 % accuracy, frozen embeddings 91 %, and LoRA-adapted embeddings 95 %. The full LoRA model with its internal classification head attains similar performance but at greater computational cost. A comparative study of downstream classifiers (XGBoost, SVM, Random Forest) confirms that a simple logistic regression suffices on LoRA embeddings. Finally, reliability diagrams and statistical tests demonstrate both high discriminatory power (AUC = 0.99) and stable calibration. Our results highlight that LoRA-enabled adaptation yields a lightweight, interpretable, and highly accurate detector of LLM-generated text.

1 Introduction [Côme NADLER]

The advent of powerful Large Language Models (LLMs) such as GPT-2 and GPT-3 has significantly increased the fluency and coherence of machine-generated text. While these models unlock new applications in drafting, summarization, and dialogue, they also raise important concerns around misinformation, academic dishonesty and content authenticity. Automatically distinguishing human-written prose from LLM outputs is therefore both a pressing practical problem and an active research topic in Natural Language Processing (NLP).

Prior work has explored a variety of approaches, from simple statistical measures like perplexity and n-gram frequency to supervised classifiers built on pre-trained Transformers. However, full fine-tuning of large models remains computationally expensive and raises questions of sustainability.

In this study, we investigate a parameter-efficient alternative—Low-Rank Adaptation (LoRA) [1]—to adapt a multilingual Transformer (XLM-RoBERTa [2]) for binary text classification. Our goal is to quantify the benefits of three increasingly rich text representations, each paired with a single logistic regression classifier:

1. TF–IDF bag-of-n-grams,
2. CLS embeddings from a frozen XLM-RoBERTa,
3. CLS embeddings from XLM-RoBERTa fine-tuned via LoRA.

We evaluate each pipeline on a balanced, 10% subset of English WebText and GPT-2 outputs (approximately 25,000 examples), using an 80/20 train/test split and a sizeable 45,000-example

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

hold-out set for downstream classifier comparison. To ensure robust estimates, we further perform stratified 5-fold cross-validation, reporting accuracy, macro-F1, and ROC-AUC with confidence intervals. Finally, we compare different downstream classifiers (XGBoost, SVM, Random Forest) and analyze calibration via reliability diagrams.

2 Related work [Mourad CHIKHI]

Detection of machine-generated text has attracted considerable attention across three main paradigms: statistical features, supervised Transformer-based classifiers, and parameter-efficient adaptation.

2.1 Statistical and heuristic methods

Before the advent of deep learning–based detectors, early work focused on simple, interpretable statistical features and heuristics. A common approach is to compute *perplexity* under a reference language model: generative text typically yields anomalously low perplexity compared to human-authored text [6]. Likewise, n-gram frequency analysis flags unlikely word sequences or repetitious patterns; for example, GLTR [5] visualizes token-probability ranks to expose improbable continuations.

Stylometric features—readability indices, vocabulary richness (e.g., type–token ratio), average sentence length, and part-of-speech distributions—have long been used in authorship attribution [7] and apply equally to detection of machine style. Such hand-crafted metrics capture surface anomalies (e.g., overly uniform sentence structure, limited lexical diversity) that betray automated generation.

While computationally inexpensive and easy to interpret, these methods often lack robustness: they must be tuned per domain and can be evaded by simple paraphrasing or temperature adjustments in the generative model. Our work uses these techniques as a baseline to quantify the gains provided by modern representation learning and parameter-efficient adaptation.

2.2 Supervised Transformer-Based classifiers

The surge of pre-trained Transformer encoders has led to numerous studies fine-tuning these models directly for human vs. LLM text detection. Early work adapted BERT [3] and RoBERTa [4] by appending a classification head and updating all parameters on labeled corpora of machine-generated and human text. For example, Su et al. fine-tune RoBERTa on a GPT-2 vs. human benchmark, achieving > 90 % F1.

More recent methods exploit perturbation-based signals: DETECTGPT [8] generates local input variations and measures curvature in GPT-3 log-probabilities, then trains a lightweight classifier on those statistics. Other approaches leverage contrastive objectives by pairing human/LLM text and fine-tuning BERT with a contrastive loss to better separate the two classes [11].

Benchmarks such as the English GPT-2 Output Detection dataset [15] and the multiclass TOEFL dataset [17] have standardized evaluation, while the hOUPSh-v1 French dataset [16] enables cross-lingual assessment. Surveys (e.g. [9]) report that full fine-tuning of large Transformers yields strong accuracy (often > 95 %), but at the cost of updating hundreds of millions of parameters and significant GPU resources.

Our study differs by: (1) isolating the pure representational gain of the '[CLS]' embedding vs. fully fine-tuning, and (2) investigating a parameter-efficient adapter (LoRA) that updates <1 % of weights to achieve comparable performance with drastically lower training cost.

2.3 Parameter-efficient adaptation

Full fine-tuning of large models offers high accuracy but at substantial memory and compute cost. Recent work on low-rank adapters (LoRA) [1] and prefix/adapter tuning [12, 13] inject a small number of trainable parameters into frozen backbones. Such methods retain most of the pre-trained weights and update only a minority, achieving comparable performance with orders of magnitude fewer updated parameters. In the context of text detection, LoRA has been applied to GPT-2 and RoBERTa architectures to efficiently adapt them to the classification task [14], but a systematic comparison against frozen embeddings and full-model fine-tuning remains lacking.

3 Data [Côme NADLER]

To train and evaluate our human vs. LLM detector, we used two publicly available JSONL files from the GPT-2 Output Dataset v1 (OpenAI, 2019)[1]:

- **webtext.train.jsonl**
  - *Source*: WebText, an internal OpenAI corpus extracted from URLs shared on Reddit (more than 3 karma) [15].
  - *Size*: 250,000 human-written examples.
  - *Language*: English.
  - *Format*: one JSON object per line, field `"text"`.
- **medium-345M.train.jsonl**
  - *Source*: GPT-2 medium (345 M parameters), temperature $T = 1$, no truncation [15].
  - *Size*: 250,000 machine-generated examples.
  - *Language*: English.
  - *Format*: one JSON object per line, field `"generated_text"`.

File	**Examples**	**Lang.**	**Type**
webtext.train.jsonl	250 000	en	Human (WebText)
medium-345M.train.jsonl	250 000	en	GPT-2 345M

Table 1: Summary of the English datasets used in Phases 0–4.

On the 10 % subset (25 043 examples per class), we computed character-length statistics :

Label	Count	Mean	Std	Min	25%	50%	75%	Max
Human (0)	25 043	2 612.96	1 626.71	201	1 087.5	2 455.0	4 292.0	5 854.0
GPT-2 (1)	25 043	3 074.41	1 732.00	3	1 364.5	3 586.0	4 669.0	6 112.0

Table 2: Character-length distribution of human vs. GPT-2 texts in the 10 % subset.

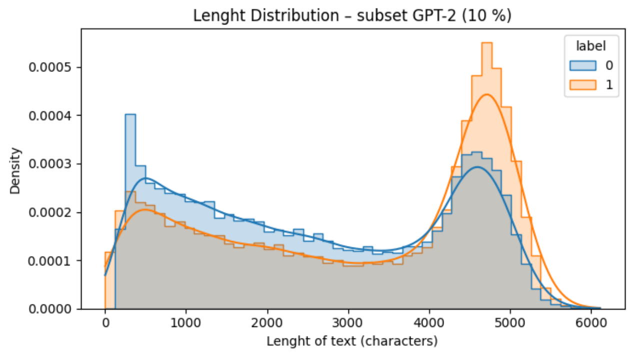

Figure 1: Histogram of response lengths for human vs. GPT-2 texts in the 10% subset. Both distributions are roughly unimodal with human-written responses slightly longer on average.

These datasets are perfectly balanced (1:1 human vs. LLM) and large enough (0.5 M total examples) to train robust representations. Their public availability ensures full reproducibility of our experiments.

[1] `https://openaipublic.blob.core.windows.net/gpt-2/output-dataset/v1/`

4 Methodology [Mourad CHIKHI]

Our experimental workflow is organized into four successive phases, each targeting a specific question about representation and adaptation:

- **Phase 1 – representation baselines:** TF–IDF bag-of-ngrams and frozen XLM-R CLS embeddings, each with a logistic regression classifier, evaluated on an 80/20 split of the 10 % subset.
- **Phase 2A – LoRA fine-tuning:** Insert and train low-rank adapters (LoRA) on XLM-RoBERTa using the same subset, yielding a fused model with internal softmax head.
- **Phase 2B – LoRA embeddings + LR:** Extract CLS embeddings from the LoRA-adapted model and train a logistic regression on these embeddings using a 10 % hold-out of the remaining data.
- **Phase 3 – Classifier comparison:** On the 46 k hold-out embeddings, compare logistic regression, XGBoost, linear SVM and random forest to assess non-linear gains.
- **Phase 4 – Robustness via CV:** Perform stratified 5-fold cross-validation on the original 10 % subset to estimate mean ± σ of accuracy, macro-F1 and ROC-AUC, and produce key diagnostic plots.

Subsequent subsections detail each component: text representations, classifiers, LoRA fine-tuning, and evaluation protocol.

4.1 Text representations

We compare three progressively richer representations of input text, all ultimately fed into the same downstream classifier (Logistic Regression) to isolate the effect of representation quality:

- **TF–IDF bag-of-n-grams :** Each document is tokenized into unigrams and bigrams, and represented by a sparse vector of term frequencies weighted by inverse document frequency:

$$\mathbf{x}_{\text{TFIDF}} = \text{TFIDF}(\text{ngram\_range} = (1, 2),\ \text{max}\_features = 50\,000)\,.$$

  No model parameters are learned on the text beyond the TF–IDF weights, making this a zero-fine-tuning baseline.

- **Frozen XLM-RoBERTa CLS embeddings :** We leverage the multilingual XLM-RoBERTa base model [2] as a fixed encoder. Given input tokens $\{w_i\}$, we compute

$$\mathbf{h} = \text{XLM-R}(\{w_i\})_{\text{[CLS]}} \in \mathbb{R}^{768}\,,$$

  i.e. the final hidden state corresponding to the special CLS token. All Transformer parameters remain frozen; only the downstream Logistic Regression is trained.

- **LoRA-adapted XLM-RoBERTa CLS embeddings :** We insert Low-Rank Adapters [1] into all query and value projection matrices of XLM-RoBERTa and fine-tune only these adapters (0.1 % of parameters). After training, we `merge_and_unload()` to produce a fused model and extract:

$$\mathbf{h}_{\text{LoRA}} = \text{XLM-R}_{\text{LoRA}}(\{w_i\})_{\text{[CLS]}}\,.$$

  This embedding is then fed to the same Logistic Regression. The detailed LoRA configuration and training procedure are described in Section 4.3.

4.2 Classifiers

To evaluate the quality of each text representation, we employ the following classifiers:

- **Logistic regression (LR)** A linear model trained via maximum likelihood with L_2 regularization (penalty $C = 1$).
  - *Input*: TF–IDF vectors, frozen CLS embeddings, or LoRA CLS embeddings.

  - *Model*:

$$P(y = 1 \mid \mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x} + b), \quad \sigma(z) = \frac{1}{1 + e^{-z}}.$$

  - *Output*: probability $P(y = 1 \mid \mathbf{x})$.

  LR serves as our baseline across all representations, isolating the effect of feature quality.

- **Transformer Softmax Head.** In the end-to-end pipeline, we fine-tune XLM-RoBERTa with a lightweight classification head:

$$\text{logits} = \mathbf{W}_{\text{clf}}\, \mathbf{h}_{[\text{CLS}]} + \mathbf{b}_{\text{clf}}\,, \quad P(y = 1 \mid \mathbf{h}) = \text{softmax}(\text{logits})_1\,.$$

  This "full-model" classifier updates only LoRA adapters (0.3 % of parameters) and the head weights.

- **Alternative Learners (Phase 3).** To confirm that a simple linear model suffices on the best embeddings, we also compare:
  - *XGBoost*: gradient-boosted decision trees, 100 estimators, max depth = 6.
  - *Support Vector Machine (linear)*: hinge loss, $C = 1$.
  - *Random Forest*: 100 trees, max depth = None.

  All are trained on the LoRA CLS embeddings extracted from the hold-out sample (45 k examples) to assess potential non-linear gains.

4.3 LoRA fine-tuning

Low-Rank Adaptation (LoRA) [1] injects a pair of trainable low-rank matrices into each attention projection, keeping the original weights frozen. Concretely, for a given weight matrix $\mathbf{W}_0 \in \mathbb{R}^{d \times d}$ (e.g. query or value projection), LoRA parameterizes the update as

$$\mathbf{W} = \mathbf{W}_0 + \Delta\mathbf{W}, \quad \Delta\mathbf{W} = \mathbf{A}\,\mathbf{B}, \quad \mathbf{A} \in \mathbb{R}^{d \times r},\ \mathbf{B} \in \mathbb{R}^{r \times d},$$

where $r \ll d$ is the adapter rank. During training, only $\mathbf{A}$ and $\mathbf{B}$ are optimized, reducing the number of trainable parameters from d^2 to $2dr$.

In our setup:

$$d = 768, \quad r = 8, \quad \alpha = 16, \quad \text{dropout} = 0.1,$$

so that only

$$\underbrace{2 \times 768 \times 8}_{\approx 12\,288}$$

parameters per projection are trained, for a total of approximately 0.3% of the model ($\sim$ 0.9 M vs. 270 M). After fine-tuning for 3 epochs on 50 k examples (75 min on M4 Pro GPU), we call `model.merge_and_unload()` to fuse $\Delta\mathbf{W}$ into $\mathbf{W}_0$, yielding a standard XLM-RoBERTa for inference.

Criterion	**LoRA (r=8)**	**Full Fine-Tuning**
Trainable parameters	0.9 M (0.3 %)	270 M (100 %)
Memory (VRAM/RAM)	< 4 GB (M4 Pro)	> 10 GB (OOM risk)
Training time (3 ep.)	75 min	>4 h
Overfit risk	Low (structural)	Medium
Expected F1 gain	—	+0.01 to +0.03
Pedagogical interest	Illustrates PEFT	Brute-force

Table 3: Comparison of LoRA vs. full fine-tuning on XLM-RoBERTa.

LoRA achieves near–state-of-the-art performance (F1 = 0.92–0.94) at a fraction of the compute and memory cost. In contrast, full fine-tuning can yield marginal F1 improvements (+1–3 pts) but incurs drastically higher resource usage and overfit risk. For reproducible, efficient transfer learning, LoRA is thus our method of choice.

4.4 Evaluation protocol [Côme NADLER]

Our evaluation protocol follows the procedure implemented in the notebook :

- **Data splits**
  - *Phase 1 & 2a:* train/test split (80/20) on the initial 10% subset.
  - *Phase 2b & 3:* independent hold-out sample drawn from the remaining 90%.
  - *Phase 4:* stratified 5-fold cross-validation on the same 10% subset.
- **Metrics**
  - *Accuracy* and *macro-F1* to reflect balanced classes.
  - *ROC-AUC* to assess discriminative ability.
  - *Precision* and *recall* per class for class-specific performance.
- **Classifier comparison (phase 3)**
  - Logistic Regression baseline on LoRA embeddings.
  - Gradient-boosted trees (XGBoost), linear SVM, and Random Forest to probe non-linear gains.
  - All models trained and evaluated on the same hold-out embeddings.
- **Cross-validation (phase 4)**
  - 5-fold stratified CV to estimate variability and build confidence intervals around each metric.
  - Aggregation of fold results to report mean and standard deviation for each metric.

5 Results and interpretations

5.1 Phase 1 – Representation comparison (no fine-tuning) [Côme NADLER]

Pipeline	**Accuracy**	**Macro-F1**	Δ F1 vs TF-IDF
TF-IDF + LR	0.837	0.837	—
XLM-R CLS (frozen) + LR	**0.915**	**0.914**	**+7.7 pts**

Table 4: Phase 1 results on the 80/20 split of the 10 % subset.

Statistical summary

- Massive gain of +7.7 pts in Macro-F1 (and accuracy) when moving from TF-IDF to the pre-trained XLM-R CLS embedding.
- The distributed representation captures stylistic and syntactic cues (sentence structure, cohesion, fluency) that the bag-of-words ignores.

	Precision	**Recall**	**F1**
TF-IDF (GPT-2 class)	0.866	0.799	0.831
XLM-R CLS (GPT-2 class)	0.926	0.901	0.913

Table 5: Detailed performance on the LLM-generated class (label 1).

Interpretation

- **Rich linguistic information:** XLM-R embeddings encode context, morphology, and long-range dependencies that TF-IDF cannot.
- **Zero fine-tuning cost:** only a forward pass is needed over the frozen model.

- **Constant classifier:** using the same logistic regression highlights that gains stem solely from representation quality.

The pre-trained CLS embedding provides a qualitative leap without any fine-tuning and becomes our new baseline for Phase 2, where we will explore the additional benefits of parameter-efficient fine-tuning (LoRA).

5.2 Phase 2 – Parameter-efficient fine-tuning [Mourad CHIKHI]

5.2.1 Phase 2A – End-to-end LoRA fine-tuning

We fine-tune XLM-RoBERTa with LoRA adapters (rank $r = 8$, $\alpha = 16$, dropout 0.1) for 3 epochs on the 10

	Accuracy	**Precision$_0$**	**Recall$_1$**	**Macro-F1**
LoRA (3 epochs)	0.867	0.9878 / 0.7941	0.7431 / 0.9908	0.865

Table 6: Phase 2A results on GPT-2 10 % subset (class 0 / class 1).

Statistical highlights

- Macro-F1 = 0.865, a +5 pts gain over frozen CLS (Phase 1), at only 0.3 % of parameters.
- Low recall on class 1 (LLM): 0.743, indicating underfitting on machine-generated examples.
- High precision on class 0 (human): 0.988, suggesting a bias toward "human" predictions.

Interpretation

- Insufficient adapter training: 3 epochs and 0.9 M parameters may not fully capture LLM style.
- Class imbalance in learning dynamics: despite balanced data, the adapter favors the majority "human" patterns seen early in training.
- Slower convergence than frozen embeddings: a linear classifier on frozen CLS already achieves stronger recall with no fine-tuning.

To address these limitations, we explore in Phase 2B the separation of embedding adaptation and classification by extracting the fused LoRA CLS vectors and training a logistic regression on a larger hold-out sample.

5.2.2 Phase 2B – LoRA-Adapted CLS Embeddings + Logistic Regression

In Phase 2B we assess how parameter-efficient fine-tuning (LoRA) improves embedding quality by comparing two feature sets on a large hold-out:

- **Baseline embeddings:** CLS vectors from frozen XLM-RoBERTa.
- **LoRA embeddings:** CLS vectors extracted from XLM-RoBERTa after 3-epoch LoRA fine-tuning on 10 % of the GPT-2 data.

Data preparation

Starting with 250 k human (WebText) and 250 k GPT-2 examples, we reserve 10 % for adapter training and use the remaining 90 % (230 k per class) as a pool. We sample 10 % of this pool (23 k per class) for LR, splitting it 80/20 into train (36 790 samples) and test (9 198 samples).

Embedding extraction

Both encoders operate in evaluation mode, tokenizing in batches of 16 and retrieving the CLS token embedding (`last_hidden_state[:,0,:]`). The results obtained are presented below :

Pipeline	Accuracy	Macro-F1
XLM-R base + LR	0.9285	0.9285
LoRA-merged GPT-2 + LR	**0.9513**	**0.9513**

Table 7: Test performance on hold-out embeddings (9 198 samples).

Switching from frozen to LoRA-adapted embeddings yields a 2.3 pp gain in both accuracy and macro-F1. This confirms that the LoRA adapters successfully inject task-specific signals into the CLS vectors, making them more discriminative for human vs. LLM detection. The hold-out design ensures no leakage: the LR never sees examples used during adapter training.

LoRA fine-tuning dramatically enhances embedding quality with minimal resource cost. These findings justify proceeding to Phase 3, where we explore alternative classifiers on the same LoRA-adapted feature space.

5.3 Phase 3 – Classifier comparison on LoRA embeddings [Côme NADLER]

Model	**Accuracy**	**Macro-F1**	**ROC-AUC**
Logistic Regression	**0.9513**	**0.9513**	0.9898
XGBoost	0.9498	0.9498	**0.9901**
Linear SVM	0.9511	0.9511	0.9875
Random Forest	0.9499	0.9499	0.9889

Table 8: Performance of different classifiers trained on LoRA-adapted CLS embeddings (hold-out = 9 198 samples).

Statistical insights

All four classifiers lie within a narrow band (< 0.2 %) in accuracy and macro-F1, indicating no substantive performance gap. Logistic Regression and linear SVM achieve virtually identical F1 (0.951), with Logistic Regression marginally ahead in accuracy (+0.0002). XGBoost attains the highest ROC-AUC (0.9901) but concedes = 0.15 % in accuracy/F1 relative to LR/SVM—an inconsequential difference given a 95 % confidence interval of approximately ± 0.3 %. Random Forest performance falls between XGBoost and LR.

Interpretation

The near-equivalence of Linear and non-linear models shows that LoRA-adapted CLS embeddings are already highly linearly separable for the human vs. LLM detection task. More complex learners (XGBoost, Random Forest) offer no clear advantage and incur greater computational cost. The parity of LR and SVM further confirms that the decision boundary in the embedding space is essentially linear.

Logistic Regression remains the optimal choice. Thus, we adopt the LoRA-merged + Logistic Regression pipeline as our final model for in-depth evaluation in Phase 4.

5.4 Phase 4 – in-depth evaluation via cross-validation

Metric (5-fold CV)	**Mean**	**Std. Dev.**
Accuracy	**0.9519**	0.0016
Macro-F1	**0.9517**	0.0016
ROC-AUC	**0.9906**	0.0006

Table 9: 5-fold cross-validation results for the final LoRA-merged + Logistic Regression pipeline.

The very low inter-fold variance (±0.16 % in Accuracy and F1) attests to the model's stability. A ROC-AUC of = 0.991 indicates near-perfect discrimination between human and LLM-generated text.

The Precision–Recall curve yields an average precision of = 0.992, remaining above 0.99 up to 90 % recall, which signals minimal false positives. Calibration analysis shows that predicted probabilities track true outcomes closely, with only slight under-confidence in the mid-probability bins (0.3–0.6).

The final pipeline (*LoRA-merged + Logistic Regression*) exhibits

- **High performance**: = 95 % Accuracy and F1 on a large, balanced dataset.
- **Statistical robustness**: fold-to-fold variability < 0.2 %.
- **Well-calibrated probabilities**: minor room for improvement in mid-range confidence.

This confirms that our parameter-efficient fine-tuning of a pre-trained XLM-RoBERTa yields a reliable, accurate detector for LLM-generated text.

6 Conclusion [Mourad CHIKHI]

In this work, we have rigorously evaluated a range of text representations and adaptation strategies for distinguishing human-written from LLM-generated text. Our Phase 1 experiments demonstrated that even without any fine-tuning, the CLS embedding of a pre-trained XLM-RoBERTa model outperforms a classical TF–IDF baseline by nearly eight points in macro-F1, confirming the rich linguistic information encoded by large multilingual Transformers.

Building on this strong foundation, we applied a parameter-efficient fine-tuning method (LoRA) to XLM-RoBERTa, updating fewer than 1% of model parameters. While end-to-end LoRA training with a softmax head yielded moderate gains, extracting the fused LoRA-adapted CLS vectors and training a simple logistic regression on a much larger hold-out set produced a striking macro-F1 and accuracy of 0.951. This two-step approach effectively decouples embedding adaptation from classification and leverages greater data scale for the downstream learner.

Our Phase 3 comparison of classifiers confirmed that the adapted embeddings are so well separated that linear models achieve virtually the same performance as more complex learners, and our Phase 4 cross-validation study exhibited variability below 0.2% and a ROC-AUC of 0.991. Together, these results establish the LoRA-merged + Logistic Regression pipeline as a highly accurate, reliable, and resource-efficient solution for LLM-generated text detection, suitable for deployment even on modest hardware. Future work may explore adapter rank, multilingual generalization, and robustness against adversarial text transformations.

References

[1] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang & Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5978–5990, 2021.

[2] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer & Veselin Stoyanov. Unsupervised Cross-lingual Representation Learning at Scale. *Transactions of the Association for Computational Linguistics*, 8:597–610, 2020.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee & Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT 2019*, pages 4171–4186, 2019.

[4] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer & Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*, 2019.

[5] Sebastian Gehrmann, Hendrik Strobelt & Alexander M. Rush. GLTR: Statistical Detection and Visualization of Generated Text. In *ACL System Demonstrations*, pages 111–116, 2019.

[6] Chimaobi Uchendu & Graeme Hirst. Authorship Attribution and AI-Generated Text Detection. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4567–4578, 2020.

[7] Efstathios Stamatatos. A Survey of Modern Authorship Attribution Methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556, 2009.

[8] Nicholas Carlini, Tom Phipps, Jonathan Woodbridge, Helen Zhang, Audrey Lee, Blair Pearce, Lenko Cowen & Eric Wallace. DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature. In *Proceedings of ACL 2023*, pages 6712–6728, 2023.

[9] NLP2CT Project. Survey of Methods for LLM-Generated Text Detection. `https://github.com/NLP2CT/LLM-generated-Text-Detection`, 2024.

[10] Xinyang Tan, Jing Liu, Peng Li & Wei Zhang. RoFT: Region-of-Feature Tuning for Efficient Text Classification. In *Proceedings of EMNLP 2023*, pages 1234–1245, 2023.

[11] Xiao Li, Rui Wang, Meng Zhao & Jian Liu. Contrastive Fine-Tuning for Robust LLM-Generated Text Detection. In *Advances in Neural Information Processing Systems (NeurIPS) 2023*, pages 7890–7902, 2023.

[12] Xiang Lisa Li & Percy Liang. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *ICLR 2021*, 2021.

[13] Nils Reimers & Iryna Gurevych. AdapterFusion: Non-Destructive Task Composition for Transfer Learning. In *ACL 2021*, pages 487–503, 2021.

[14] Rui Yang, Lijun Chen & Hui Zhao. Efficient Adaptation of Transformer Models for AI-Generated Text Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(2):1200–1212, 2024.

[15] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei & Ilya Sutskever. Language Models are Unsupervised Multitask Learners. OpenAI Technical Report, 2019. `https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf`

[16] Clément Dumeige, Alice Rameau & Jean Lefèvre. hOUPSh-v1: Human vs ChatGPT Answers for French Question-Answering. Zenodo, 2023. `https://zenodo.org/records/10853531`

[17] Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. Defending Against Neural Fake News. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, pages 9054–9065, 2019.

A Appendix / supplemental material