Interpretable Statistical Feature-Based Attribution of LLM-Generated Text

Anonymous ACL submission

Abstract

The rapid development of large language models (LLMs) has initiated an increasing interest not only in detecting whether a text is machine-generated, but also in identifying which LLM produced it. Multi-class author attribution remains challenging, especially under domain shift. Although the majority of existing methods focus on model-based methods for author attribution, we propose an interpretable framework based on explicitly computed statistical features. We adopt a diverse set of lexical, syntactic, and distributional features based on their low mutual correlation. We analyze the contribution of each feature to the overall attribution performance and investigate their stability across domains. Our results show that while some groups of features achieve strong in-domain performance, only a subset remains stable under domain shift. By selecting features based on their cross-domain stability, our method achieves competitive performance (4 to 8 times lower latency) in comparison to existing model-based approaches. In summary, statistical feature selection based on cross-domain stability provides a computationally efficient basis for attributing text to a generating LLM with little loss in macro-F1 performance.

1 Introduction

Large language models (LLMs) are powerful Artificial Intelligence (AI) tools for integrating and generating information. However, these models are vulnerable to generating misinformation or harmful information such as fake news. Machine generated text (MGTs) detection methods and authorship attribution (AA) mechanisms are developed to maintain the integrity of digital content or mitigate the risk of misinformation and plagiarism. The former distinguishes human-written texts (HWTs) from MGTs, and the latter further specifies which specific model/human has generated the text. Generalization, explainability, and interpretability are three main characteristics for evaluating MGT detection methods and AA mechanisms.

MGT detection methods are binary classifications and have been widely studied in the literature. These methods are divided into two main categories, namely metric-based and model-based methods. Given the fact that MGTs are repetitive while HWTs are creative, metric-based methods (Gehrmann et al., 2019; Mitchell et al., 2023; Bentz, 2023) either statistically extract features, such as the type-token ratio and word entropy, or obtain feature values, such as logarithmic likelihood and token rank, using a reference model. Metric-based methods are more explainable, yet face poor generalization across domains. On the other hand, model-based detectors (Guo et al., 2023; Solaiman et al., 2019; Chen et al., 2023) exploit a pretrained LM, such as BERT (Devlin et al., 2019), to distinguish HWTs from MGTs. Unlike metric-based methods that highly depend on appropriate feature analysis, fine tuning a pretrained LM eliminates the need for hand-crafted features. Evaluations in the literature indicate that model-based detectors mainly outperform their metric-based counterparts. However, these methods suffer from a long training time, need substantial labeled data, and struggle with cross domain generalization.

Compared to MGT detection, AA is a more challenging classification problem, as differences in text generation patterns across LLMs are often subtle. Uchendu et al. 2020 gives a brief review of existing methods for the AA task, including those based on explicit statistical features and CNNs. Viewing AA as a multi-class classification problem, He et al. 2024 fine-tunes the pretrained model BERT as the classifier. Such model-based AA methods typically achieve better performance but require specialized fine-tuning. Liu et al. 2025 investigated the generalization ability of MGT detectors using five popular LLMs as well as metric-based classifiers (using one to five metrics). Using

model-based approaches, they study variations in training vs. test domain performance and explore the use of multi-shot techniques for extending AA into new domains.

To address the gaps in the current literature on metric-based AA, we make the following contributions: We enhance AA performance by examining a large number of potential metrics that exhibit sufficient statistical independence and are resilient to variations in text length. For each specific data domain, we compute statistical features and perform in-domain AA. To capture the importance of the features, we analyze them using the SHAP values. To generalize AA across domains, we develop a metric-based method to order and select domain-agnostic features. To demonstrate the efficiency of our proposed method for AA, we compare our metric-based approach with several model-based methods.

2 Preliminaries

2.1 Feature Types

For characterizing text, several explicitly computable statistical features have been proposed. Bentz 2023 used four character-level features to distinguish human language: type-token ratio, unigram character entropy, entropy rate, and repetition rate. We generalize them to the corresponding *word-level* features. Since language models tend to generate texts of varying length, we avoid using text length-related features, such as total number of characters or total number of words. To collect more features, such as syntactic features, lexical features, and formatting features, we further examine the features proposed by Fröhling and Zubiaga 2021.

Although we have many features to use to classify the texts, we believe adding more features does not necessarily improve performance. Some features may bring very little information but large noise, and some of them are highly correlated. We collect explainable features and condense the feature set to be less redundant. For example, for syntactic features we only focus on function words (Interjection, Auxiliary, Pronoun, Conjunction, Particle, Determiner). In addition, we also use a reference model (GPT2) to obtain the next token probability since this helps explain the degree of surprisal in the text. The sets of features we use for classification are listed below:

- *Syntactic and Lexical Diversity*: frequencies of Part-of-Speech (POS)-tags, word entropy.
- *Repetitiveness*: character type-token ratio, character repetition rate, word compression rate, N-gram word repetition rate (N=1, 2).
- *Formatting*: paragraph length (average and standard deviation), punctuation ratio and diversity, enumeration frequency.
- *Token Probability*: log-likelihood (Solaiman et al., 2019), minimum log-likelihood, log-likelihood entropy.
- *Other linguistic features*: word length, sentence length, burstiness, zipf's deviation.

2.2 Feature Definitions

We next define and interpret some of our explicitly-computed statistical features:

Compression Rate. Given a sequence of words $\{w_i\}_{i=1}^{T}$, we first normalize and concatenate them into a byte sequence S. Let $|S|$ denote the length of the original sequence and $|C(S)|$ the length after applying a lossless compression algorithm (e.g., `zlib`). The compression rate is estimated as

$$\text{CompressionRate} = \frac{|C(S)|}{|S|},$$

Lower values indicate higher redundancy and more predictable structure, while higher values suggest greater randomness and lexical diversity.

N-gram Repetition. N-gram repetition measures the extent to which contiguous word sequences are repeated within a text. Given a sequence of words $\{w_i\}_{i=1}^{T}$, we construct the set of all n-grams as

$$g_i = (w_i, w_{i+1}, \ldots, w_{i+n-1}),\ i = 1, \ldots, T-n+1.$$

Let N and U denote the total number of n-grams and the number of unique n-grams, respectively. We define the repetition rate as

$$\text{Repetition}_n = 1 - \frac{U}{N},$$

A higher value indicates more repeated patterns, which is often associated with lower lexical diversity and can reflect artifacts of MGT.

Zipf Deviation. Natural language approximately follows Zipf's law, where word frequency $f(r)$ is inversely proportional to its rank r. In log-log space, we have

$$\log f(r) = a \log r + b.$$

Given rank-frequency pairs $\{(r_i, f_i)\}_{i=1}^{N}$, we fit a linear model and measure the deviation using root mean squared error (RMSE):

$$\text{ZipfDev} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\Big(\log f_i - (a \log r_i + b)\Big)^2},$$

Higher values indicate a greater deviation from natural lexical distributions.

Burstiness. Burstiness captures the variability of sentence lengths within a text, reflecting stylistic irregularities. Let μ and σ be the mean and standard deviation of sentence lengths, respectively. Then, the burstiness score is defined as

$$\text{Burstiness} = \frac{\sigma}{\mu},$$

A higher burstiness value indicates a greater variability in sentence lengths, which is often associated with more natural and expressive human writing. In contrast, machine-generated text tends to exhibit more uniform sentence structures, resulting in a lower burstiness.

Likelihood-based Features. We leverage pretrained language models to capture token-level predictability of a text. Given a token sequence $\{w_t\}_{t=1}^{T}$, the conditional log-likelihood of each token is computed as:

$$\log P(w_t \mid w_{<t}).$$

Based on these token-level scores, we derive several summary statistics.

Mean Log-Likelihood. We compute the average log-likelihood over all tokens:

$$\mu_{\log P} = \frac{1}{T}\sum_{t=1}^{T} \log P(w_t \mid w_{<t}),$$

which reflects the overall fluency and conformity of the text to the language model.

Minimum Log-Likelihood. We consider the minimum token log-likelihood:

$$\min_t \log P(w_t \mid w_{<t}),$$

which captures the most unexpected or least probable token in the sequence, providing sensitivity to local irregularities.

Log-Likelihood Entropy. To measure variability in token predictability, we compute the entropy of the log-likelihood distribution:

$$H_{\log P} = -\sum_{t=1}^{T} \tilde{p}_t \log \tilde{p}_t,$$

where $\tilde{p}_t$ is obtained by normalizing log-likelihood scores across tokens. This feature reflects the dispersion of token probabilities within the text.

We redact the definitions of the other features that we have used due to their commonality. The redacted definitions can be found in earlier work (Fröhling and Zubiaga, 2021; Bentz, 2023).

2.3 Feature Independence

The statistical features described above capture complementary aspects of model-based uncertainty, including overall likelihood, extreme deviations, and variability, which are useful for distinguishing human-written and machine-generated text. Figure 1 illustrates the correlations among the features extracted across all three domains available in our dataset. (See 4.1 for dataset discussion.) Overall, most of our features exhibit low correlation with each other. The few highly correlated features tend to belong to the same group, such as repetitiveness and token probability.

3 Cross-Domain Feature Robustness Analysis

Let $\mathscr{D} = \{D_1, D_2, \ldots, D_K\}$ denote K source domains with labeled data, and let D_t denote an unlabeled target domain, where $D_t \notin \mathscr{D}$. For each text sample, we extract a d-dimensional feature vector, with features described in Section 2.

3.1 Cross-Domain Feature Statistics

Let $x \in \mathbb{R}^d$ and $x_k^{(i)}$ denote the feature vector of the i-th sample in domain D_k. For each domain $D_k \in D_t \bigcup \mathscr{D}$, we compute the empirical mean and standard deviation for each feature dimension j:

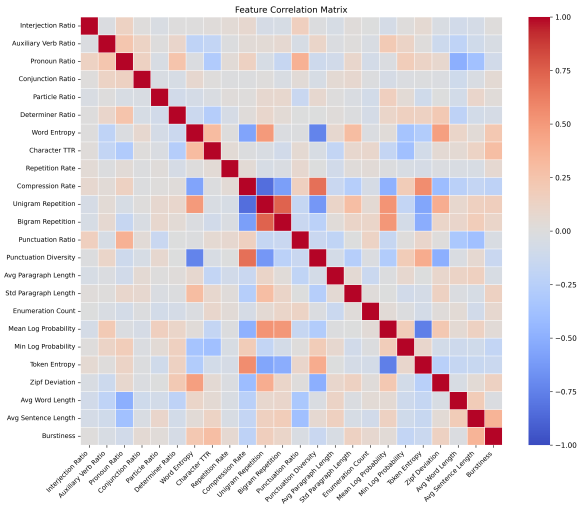


Figure 1: Correlation analysis of features.

$$\mu_{k,j} = \frac{1}{N_k} \sum_{i=1}^{N_k} x_{k,j}^{(i)} \tag{1}$$

$$\sigma_{k,j} = \sqrt{\frac{1}{N_k - 1} \sum_{i=1}^{N_k} \left(x_{k,j}^{(i)} - \mu_{k,j} \right)^2} \tag{2}$$

3.2 Cross-Domain Feature Distance

To measure the domain discrepancy of feature j w.r.t. target domain D_t, we compute the Cohen's d distance between each source domain D_k and the target:

$$d_{k,t}^{(j)} = \frac{|\mu_{k,j} - \mu_{t,j}|}{\sqrt{\frac{\sigma_{k,j}^2 + \sigma_{t,j}^2}{2}} + \epsilon}, \tag{3}$$

where $\epsilon > 0$ is a small constant for numerical stability.

3.3 Cross-Domain Instability Score

We define the cross-domain instability score of feature j as its average distance when measured from all source domains to the target domain:

$$S_j^{(t)} = \frac{1}{K} \sum_{k=1}^{K} d_{k,t}^{(j)} \tag{4}$$

A lower value of $S_j^{(t)}$ indicates that feature j is more stable between the source (domains) and the target (domain).

4 Evaluation

4.1 Datasets

We adopt datasets from (He et al., 2024), following the prompts used by (Verma et al., 2024). The datasets consist of three domains:

Essay is based on essays on IvyPanda, which contains high school and university level essays from different disciplines.

Reuters is derived from Reuters 50-50 authorship identification dataset, which consists of 1,000 (news) articles by 50 journalists.

WP is about creative writing and is based on the r/WritingPrompts subreddit, where users share prompts and craft stories in response to these prompts.

Each dataset contains human written articles and text samples generated from six different LLMs: ChatGLM, ChatGPT-Turbo, Claude, Dolly, GPT4All, and StableLM. Overall, there are 7000 samples in each dataset, 1000 per ("author") class. Following the same data pre-processing process as (He et al., 2024), we remove data entries with no more than one word. We also remove incorrect spaces before punctuation. We randomly split our data into 80% as the training set and the rest as the test set.

4.2 Classifier

Since XGBoost is a high-performance model in handling structured data, we develop a simple XGBoost model for classification purposes. Our evaluations also show that XGBoost performs better than other machine learning models, such as random forest, logistic regression, K-nearest neighbors (KNN), and multi-perceptron layer (MLP) model.

We use the default hyperparameters for XGBoost in all experiments. We conducted preliminary tuning and found that varying hyperparameters results in only marginal performance changes (within 0.01 in Macro-F1). Therefore, we adopt the default settings to maintain simplicity and avoid overfitting to specific datasets.

4.3 Evaluation Metric

We use macro-F1 as the primary evaluation metric, as it measures performance independently for each class and then averages across all classes. Although the datasets we use are balanced, the difficulty of attributing text to its author varies, particularly among texts produced by the generating LLMs.

Macro-F1 therefore ensures that performance on harder-to-distinguish classes is properly reflected, whereas accuracy may obscure such differences.

4.4 Baselines

We compare our method with several strong model-based baselines, including ChatGPT-Detector, OpenAI-Detector, and LM-Detector.

ChatGPT Detector is developed by (Guo et al., 2023) to distinguish human written texts from ChatGPT-generated texts. The model was created by fine-tuning a RoBERTa model using the Human ChatGPT Comparison Corpus (HC3) dataset.

OpenAI Detector (Solaiman et al., 2019) is used to detect if a text is generated by GPT-2. It fine-tuned a RoBERTa model using outputs from the largest GPT2 model (i.e., with 1.5B parameters).

Besides the previous models, we can also construct **LM D**etector by fine-tuning a pre-trained language model with an extra classification layer. Here, we take the BERT model following (Ippolito et al., 2020).

These three detectors use pre-trained models from HuggingFace, and (He et al., 2024) fine-tuned these model-based detectors for author attribution task. From their study, the performance of the models keeps improving when training epochs increase from 1 to 5, but plateaus from 5 to 20. Therefore, to maintain a fair comparison, we set the training epochs to 10.

4.5 In-Domain Evaluation

We use a simple XGBoost model on our explicitly computed statistical features for in-domain author attribution task, and compare the results to several strong baselines mentioned in Sec. 4.4.

Given that model-based methods imply randomness, we perform three runs per experiment and report the average and standard deviation of the results. As shown in Table 1, the performance of metrics-based method is competitive and sometimes better than model-based methods. Although LM-D delivers the strongest baseline, our method remains competitive with a slight degradation in performance. This indicates that the proposed feature set captures most of the discriminative signals learned by stronger model-based approaches. OpenAI-D exhibits slightly weaker performance with a macro F1 greater than 0.8 across all domains. In contrast, ChatGPT-D has the lowest performance among all these methods. It is most unstable across the three target domains.

Method	D_t		
	Essay	**Reuters**	**WP**
Features	0.871	0.885	0.878
ChatGPT-D	0.688 ± 0.107	0.793 ± 0.069	0.717 ± 0.092
OpenAI-D	0.814 ± 0.047	0.868 ± 0.014	0.839 ± 0.052
LM-D	0.886 ± 0.021	0.909 ± 0.013	0.883 ± 0.011

Table 1: In-domain macro F1 comparison.

Beyond accuracy, our method demonstrates strong robustness, with relatively stable performance across all settings: the variance across columns corresponding to our method in Table 1 is relatively small compared to model-based methods therein. This stability suggests that the proposed features are less sensitive to distributional shifts compared to ChatGPT-D and OpenAI-D, both of which exhibit larger fluctuations across the three domains.

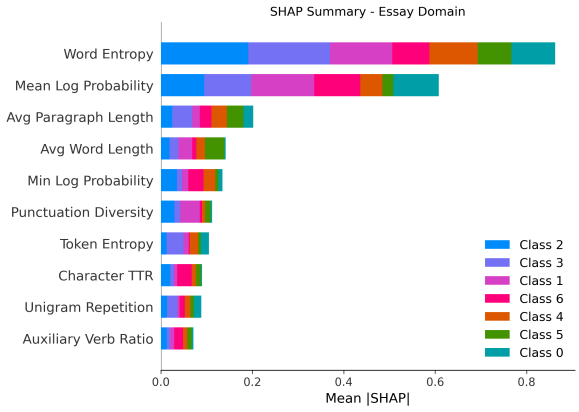
Furthermore, unlike LM-based approaches, our method is fully feature-driven and does not require expensive inference from LLMs. This provides a favorable trade-off between performance, efficiency, and interpretability.

We also analyze SHAP values to highlight the importance of each feature or each group of features. There is a list of SHAP values for each class but we take the mean absolute value to see the overall influence to the model. Figures 2a, 2b, and 2c show the top 10 important features of in-domain experiments for each domain. We conclude that word entropy, average log-likelihood, and average paragraph length form the top-four influential features of all domains. We further record the aggregated SHAP values per group because the features in the same group might be correlated. Table 2 indicates that syntactic & lexical diversity plays the most important role in the evaluations followed by token probability. The other three groups, i.e., formatting, repetitiveness, and other linguistic features, are of similar significance.

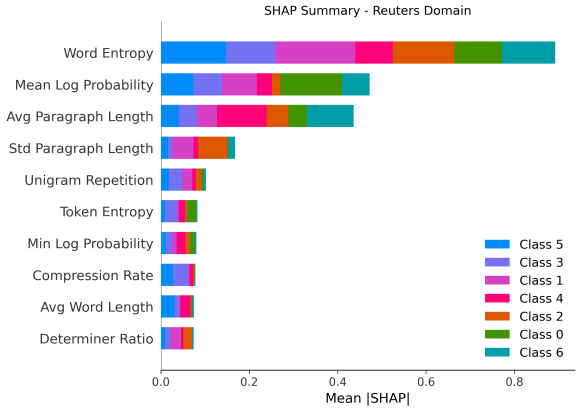
Group of features	Essay	Reuters	WP
Syntactic & Lexical Diversity	0.1268	0.1305	0.1082
Token Probability	0.0956	0.0716	0.0889
Repetitiveness	0.0276	0.0286	0.0281
Formatting	0.0417	0.0808	0.0338
Other Linguistic Features	0.0296	0.0201	0.0587

Table 2: SHAP values comparison in groups.

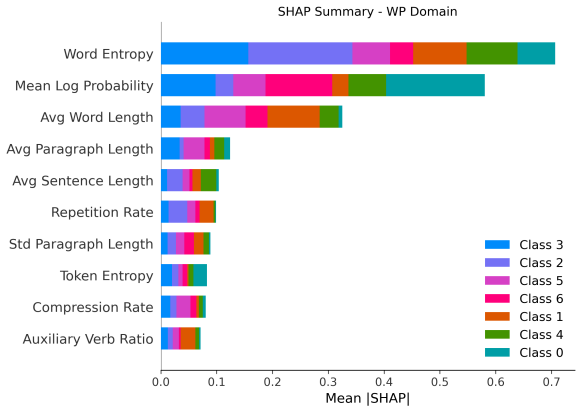
Both feature-based and model-based methods are vulnerable to cross-domain instability. Domain-



(a) Essay



(b) Reuters



(c) WP

Figure 2: SHAP value analysis in different domains: (a) Essay, (b) Reuters, and (c) WP.

adversarial methods like (Ganin et al., 2016) have been proposed to improve latent-feature stability across domains in adversarially trainable feature extractors. Here, on the other hand, we develop an alternative method relying on a measure of cross-domain stability for explicitly-computed statistical features.

4.6 Cross-Domain Evaluation

To select features, we set a threshold l and choose features with instability score less than l. Table 3 compares macro F1 scores using all features with macro F1 scores using feature selection at different thresholds l. In each experiment, we test on a target domain, D_t, after training on the other two domains. A stability threshold of $l = \infty$ implies the use of all available features. Although feature selection at a stability threshold of 0.8 slightly reduces performance for Reuters, it significantly improves results for Essay and WP, and it balances results across target domains. Overall, it increases the minimal Macro F1 score of all domains. An appropriate selection of threshold can lead to well-balanced macro F1 results across all target domains. Thresholds $l = 0.9$ and $l = 0.6$ are not considered, since no features fall within the corresponding distance intervals $[0.9, 1.0]$ and $[0.6, 0.7]$. Table 4 reports features removed under different thresholds.

Threshold (l)	D_t		
	Essay	**Reuters**	**WP**
∞	0.595	0.711	0.590
1.0	0.590	0.712	0.601
0.8	0.685	0.663	0.684
0.7	0.685	0.658	0.671
0.5	0.664	0.657	0.665

Table 3: Macro F1 comparison with different thresholds.

Distance	**Essay**	**Reuters**	**WP**
$[1.0, \infty)$	Pronoun Ratio	Pronoun Ratio	Pronoun Ratio
$[0.8, 1.0)$	Avg Word Length	Character TTR	Avg Word Length Avg Sentence Length Punctuation Ratio
$[0.7, 0.8)$	–	Avg Word Length Avg Sentence Length	Determiner Ratio
$[0.5, 0.7)$	Avg Sentence Length Character TTR Bigram Repetition Avg Paragraph Length Compression Rate Unigram Repetition	Punctuation Ratio	Character TTR Bigram Repetition

Table 4: Features filtered out under different threshold l. Each cell lists features whose scores fall within the corresponding interval $[l_{\text{lower}}, l_{\text{upper}})$.

Table 5 compares the performance in the cross-domain (using threshold $l = 0.8$) with model-based methods. Again, we test in a target domain, D_t, after training in the other two domains. Two observations are immediate. First, using our domain-stable explicit feature selection method leads to competitive results. Second, for various combinations of source and target domains, our domain-stable feature-selection method produces relatively consistent results while model-based methods vary

significantly across target domains. Interestingly,

Method	D_t		
	Essay	**Reuters**	**WP**
Features	0.685	0.663	0.684
ChatGPT-D	0.667 ± 0.032	0.713 ± 0.060	0.780 ± 0.027
OpenAI-D	0.632 ± 0.047	0.564 ± 0.018	0.810 ± 0.022
LM-D	0.495 ± 0.031	0.662 ± 0.014	0.667 ± 0.058

Table 5: Performance comparison across target domains, D_t.

by comparing Table 1 and 5 we observe that several model-based methods exhibit the counterintuitive phenomenon where cross-domain performance exceeds in-domain performance. This behavior is likely due to differences in domain difficulty, where certain target domains present clearer distinctions between text sources, making them easier to classify. In contrast, in-domain settings may involve more subtle variations that are harder for model-based detectors to capture.

4.7 Discussion

Table 6 presents the runtime comparison across methods. Our feature-based approach is substantially more efficient, achieving a runtime of only 341 seconds, which is over 4× faster than LM-D and more than 8× faster than OpenAI-D and ChatGPT-D. This highlights the practical advantage of our lightweight design, making it well-suited for large-scale or real-time applications.

We note that our feature selection method can choose features that are better suited to a target domain without relying on any labeled data in the target domain. Our approach allows the model to focus on features that remain stable across source and target distributions, improving classification, i.e., author attribution robustness.

5 Conclusion

We propose a lightweight feature-based approach for "author attribution" that uses a compact set of informative features to distinguish between text sources. Despite its simplicity, our method achieves performance competitive with model-based baselines, which depend on fine-tuned LLMs for the attribution task. We also introduce a domain adaptation strategy that selects domain-robust features using unlabeled target data, improving generalization under distributional shifts. Experimental results show that our approach consistently enhances cross-domain performance and, in some cases, outperforms model-based methods. Our feature-based method is 4× to 8× faster than three separate model-based approaches we tested. These findings highlight the effectiveness of simple, interpretable feature-based methods for robust author attribution.

Method	Runtime (s)
LM-D	1435
OpenAI-D	2873
ChatGPT-D	2809
Ours (Feature-based)	**341**

Table 6: Runtime comparison of different methods.

6 Limitations

The baseline dataset we used varies in quality and contains only three domains. More recently developed LLMs can potentially generate higher quality data. Performing author attribution on higher-quality datasets may present new challenges to our method. There are also many higher-quality recently published models which can potentially be used for model-based author attribution.

References

Christian Bentz. 2023. The Zipfian challenge: Learning the statistical fingerprint of natural languages. In *Proceedings of the 27th Conference on Computational Natural Language Learning (CoNLL)*, pages 27–37, Singapore. Association for Computational Linguistics.

Yutian Chen, Hao Kang, Vivian Zhai, Liangze Li, Rita Singh, and Bhiksha Raj. 2023. Gpt-sentinel: Distinguishing human and chatgpt generated content. *Preprint*, arXiv:2305.07969.

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.

Leon Fröhling and Arkaitz Zubiaga. 2021. Feature-based detection of automated language models: tackling gpt-2, gpt-3 and grover. *PeerJ Computer Science*, 7:e443.

Yaroslav Ganin, Evgenia Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander Rush. 2019. GLTR: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 111–116, Florence, Italy. Association for Computational Linguistics.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. *arXiv preprint arxiv:2301.07597*.

Xinlei He, Xinyue Shen, Zeyuan Chen, Michael Backes, and Yang Zhang. 2024. MGTBench: Benchmarking Machine-Generated Text Detection. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. Automatic detection of generated text is easiest when humans are fooled. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1808–1822, Online. Association for Computational Linguistics.

Yule Liu, Zhiyuan Zhong, Yifan Liao, Zhen Sun, Jingyi Zheng, Jiaheng Wei, Qingyuan Gong, Fenghua Tong, Yang Chen, Yang Zhang, and Xinlei He. 2025. On the generalization and adaptation ability of machine-generated text detectors in academic writing. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, page 5674–5685. ACM.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. 2023. DetectGPT: Zero-shot machine-generated text detection using probability curvature. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 24950–24962. PMLR.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, and Jasmine Wang. 2019. Release strategies and the social impacts of language models. *CoRR*, abs/1908.09203.

Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship attribution for neural text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8384–8395, Online. Association for Computational Linguistics.

Vivek Verma, Eve Fleisig, Nicholas Tomlin, and Dan Klein. 2024. Ghostbuster: Detecting text ghostwritten by large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1702–1717, Mexico City, Mexico. Association for Computational Linguistics.

A Appendix

We provide additional explanatory data in the following tables. Table 7 gives cross-domain distance for various features.

Feature	ER	EW	RW	E-RW	R-EW	W-ER
Pronoun Ratio	0.46	1.94	2.45	1.20	1.45	2.20
Avg Word Length	0.71	1.15	0.79	0.93	0.75	0.97
Avg Sentence Length	0.44	0.70	1.12	0.57	0.78	0.91
Character TTR	0.91	0.15	1.04	0.53	0.97	0.60
Punctuation Ratio	0.02	0.85	1.09	0.43	0.56	0.97
Determiner Ratio	0.09	0.66	0.80	0.38	0.45	0.73
Bigram Repetition	0.40	0.79	0.36	0.60	0.38	0.57
Avg Paragraph Length	0.64	0.53	0.03	0.58	0.33	0.28
Compression Rate	0.41	0.62	0.20	0.52	0.31	0.41
Std Paragraph Length	0.60	0.26	0.27	0.43	0.44	0.27
Punctuation Diversity	0.31	0.22	0.55	0.26	0.43	0.38
Unigram Repetition	0.47	0.55	0.02	0.51	0.25	0.29
Min Log Probability	0.22	0.28	0.49	0.25	0.35	0.38
Token Entropy	0.02	0.45	0.50	0.24	0.26	0.48
Zipf Deviation	0.07	0.41	0.49	0.24	0.28	0.45
Particle Ratio	0.45	0.15	0.29	0.30	0.37	0.22
Std Sentence Length	0.32	0.12	0.42	0.22	0.37	0.27
Enumeration Count	0.14	0.24	0.15	0.19	0.14	0.19
Mean Log Probability	0.04	0.20	0.25	0.12	0.14	0.22
Interjection Ratio	0.03	0.20	0.22	0.11	0.12	0.21
Auxiliary Verb Ratio	0.10	0.17	0.08	0.13	0.09	0.13
Word Entropy	0.04	0.12	0.17	0.08	0.10	0.14
Repetition Rate	0.06	0.14	0.13	0.10	0.09	0.13
Conjunction Ratio	0.09	0.08	0.16	0.08	0.13	0.12

Table 7: Feature distance statistics across domains. ER, EW, and RW denote pairwise Cohen's d distances between Essay–Reuters, Essay–WP, and Reuters–WP, respectively. E-RW, R-EW, and W-ER represent the average distance of each domain (Essay, Reuters, WP) to the other two domains, measuring domain-specific variability.