STABILIZING PERSONALIZED AI-TEXT DETECTION WITH CROSS-DOMAIN CONSISTENT FEATURES

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) have rapidly advanced in generating fluent and personalized text, making it crucial to distinguish machine-generated text (MGT) from human-written text (HWT) in personalized scenarios. However, many MGT detectors exhibit significant performance fluctuations after domain transfer, as they rely heavily on features whose discriminative directions flip across domains. This motivates us to investigate whether stable discriminative signals coexist with inverted ones. Our feature-space analysis provides evidence for both an *inverted component* whose discriminative direction reverses across domains and a *consistent component* that preserves it. Building on this, we propose PERSFIX, an interpretable and detector-agnostic adjustment framework designed to counter the inversion effect. Without modifying detector architectures, PERSFIX adjusts logits using sample projections on consistent features to amplify their influence and suppress inverted ones, thereby restoring stable discrimination in personalized detection. Experiments on nine representative MGT detectors show that PERSFIX raises the mean personalized AUROC from 0.45 to 0.84 and lifts every detector above 0.79, with the same selected adjustment transferring across scenarios and across generator LLMs at a mean improvement of $+0.38$ AUROC. PERSFIX also improves the general-domain AUROC from 0.75 to 0.88, offering a stable and interpretable solution for personalized text detection.

1 INTRODUCTION

Large language models (LLMs) have shown strong capability in text generation and are now widely applied in various domains (Huang et al., 2024; Liu et al., 2024; Wang et al., 2025). As LLMs are increasingly used to imitate real individuals and generate personalized text (Herbold et al., 2024; Alperin et al., 2025), concerns have arisen about the risks of identity impersonation, misinformation, and copyright violations (Tan et al., 2025). These risks make personalized machine-generated text (MGT) detection an essential research problem.

However, directly applying existing MGT detection methods to personalized scenarios often leads to a detection AUROC fluctuation of over 40%. Many detectors that perform well in general domains degrade sharply when applied to personalized text. For instance, Lastde++ (Xu et al., 2025) drops from 90.57% AUROC in the general domain to 49.25% in the personalized setting, while some weak detectors unexpectedly improve after transfer. Prior studies (Gao et al., 2026) observe a phenomenon that many detectors rely on text features whose discriminative directions reverse across domains, causing their predictions to flip after transfer. This observation raises a natural question: how can such feature inversion be effectively mitigated to achieve stable detection under personalization?

Inspired by this insight, we further propose that domain-discriminative text features are dualistic and can be decomposed into two complementary components: an *inverted feature* that reverses its discriminative direction across domains, and a *consistent feature* that preserves a stable direction for both domain and class separation, as shown in Figure 1. To examine this hypothesis, we propose a metric that quantifies how much each direction contributes to the overall discrimination ability of the features. Specifically, we represent these features in the latent space of a pretrained LLM and analyze their projections along different directions, calculating the ratio of each projection's marginal variance across data to the total variance. Next, we verify the hypothesis by identifying two dominant components. The first exhibits reversed discrimination across domains, corresponding to the inverted

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

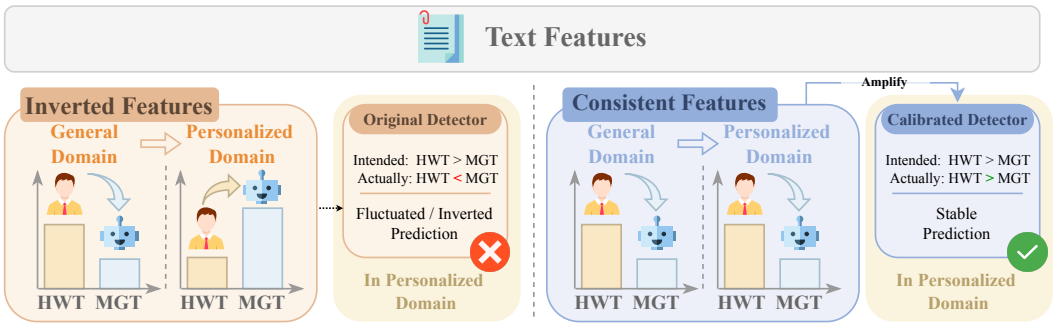


Figure 1: Illustration of the text feature duality: decomposition into inverted and consistent components, which have different effects on detectors.

feature; The second aligns across domains and also distinguishes human-written text (HWT) and MGT in the same direction, representing the consistent feature. Detectors that rely heavily on the inverted part tend to show unstable performance when transferred to personalized domains. Therefore, amplifying the consistent part naturally suppresses the inverted influence and stabilizes detection.

Based on this mechanism, we propose **Personalization Fixer (PERSFIX)**, an interpretable and detector-agnostic logit adjustment framework for personalized MGT detection. PERSFIX leaves detector structures unchanged and only adjusts logits at the output layer. It first identifies the consistent feature in LLM representations, then injects the sample's projection on this direction into the logits as a adjustment signal. Because the consistent feature is orthogonal to the inverted one, adjustment amplifies stable signals while isolating inversion.

We evaluate PERSFIX on nine representative MGT detectors with diverse architectures and design principles. Across large-scale personalized and general datasets, PERSFIX consistently improves detector stability. In the personalized setting, the mean AUROC across all detectors rises from 0.45, where inversion dominates, to 0.84, close to their general-domain performance, and even the weakest baseline at 0.20 AUROC is restored above 0.79. The selected adjustment further transfers across scenarios and across generator LLMs, with an off-diagonal mean improvement of $+0.38$ AUROC in the personalized setting. In the general setting, PERSFIX also improves the mean AUROC from 0.75 to 0.88 by adjusting predictions on highly personalized samples.

The main contributions of this work are as follows: (1) We discover that text features can be decomposed into two complementary components: an inverted feature that causes the *feature-inversion trap*, and a consistent feature that preserves same-direction discrimination across domains. (2) We propose **PERSFIX**, a detector-agnostic adjustment framework that strengthens consistent features to suppress inverted ones through logit adjustment. (3) Experiments on nine detectors across personalized and general datasets show that PERSFIX raises the personalized AUROC from a mean of 0.45 to 0.84 and the general AUROC from 0.75 to 0.88, with the selected adjustment transferring across scenarios and across generator LLMs.

2 RELATED WORK

MGT Detection Benchmarks Recent research has developed various benchmarks for evaluating machine-generated text detection across domains, generators, and languages. Representative examples include (Dugan et al., 2024; Wang et al., 2024), and (He et al., 2024). However, most existing benchmarks focus on the general domain. A recent study (Gao et al., 2026) introduced `StyloBench`, the first benchmark for personalized domains, which centers on literary and blog-style imitation. In this work, we build upon this benchmark as the foundation for our study.

MGT Detection Methods MGT detection methods are generally divided into two groups: *training-based* and *training-free* (Xu et al., 2025). Training-based approaches view detection as a supervised classification problem, often fine-tuning pretrained encoders (Liu et al., 2019) or adopting improved architectures (Guo et al., 2024; Tian et al., 2024; Jiao et al., 2025). Training-free approaches use explicit linguistic or probabilistic signals, including token distributions (Su et al., 2023b), geometric

and probabilistic cues (Bao et al., 2024), or structural indicators (Tulchinskii et al., 2023; Wei et al., 2025). We validate the effectiveness of our proposed adjustment method on both groups.

MGT Detection Adjustment Adjustment here refers to adjustments to detector output logits to improve overall performance, without changing the detector itself. As a new line of work, early relevant research remains limited. Recent studies (Jung et al., 2025; Wu et al., 2025) adjust by adapting decision boundaries or rescaling output logits to handle multi-scenario variation. However, they mainly provide an interpretation of the phenomenon, without offering a clear mechanism or practical method to mitigate its impact on detection performance.

We provide a broader discussion of related work on benchmarks and detectors in Appendix B.

3 The Duality of the Text Feature

3.1 Hypothesis: The Inverted and Consistent Components

Prior work (Gao et al., 2026) attributes detection performance fluctuations to an inverted feature that separates HWT from MGT but reverses its discriminative direction across general and personalized domains. While this finding explains why detection can fail under personalization, it leaves open whether other features preserve their discriminative direction and could support stable detection.

We refer to the feature that distinguishes general from personalized texts as the *domain feature*. Our intuition is that domain-related variation may involve both changes that reverse HWT–MGT separation and changes that preserve it. This motivates the hypothesis that the domain feature contains two types of components: an *inverted component* that reverses its class-discriminative direction across domains, and a *consistent component* that preserves the same direction. In the following sections, we introduce a metric to quantify their contributions and examine empirical evidence for this hypothesis.

3.2 Measuring Feature Contributions

We analyze the inverted and consistent components in the latent vector space of a pretrained LLM (Park et al., 2024). We train a linear classifier to distinguish general from personalized text representations and use its weight vector as the domain direction $V_{\mathrm{dom}} \in \mathbb{R}^d$. To quantify the contributions of the two components, we first express this direction in a basis that includes them, and then decompose its projection variance.

Representation in an arbitrary orthonormal basis. Let $E = (e^{(1)}, \ldots, e^{(d)})$ denote the default orthonormal basis of the representation space. In this basis, $V_{\mathrm{dom}}[i]$ is the coordinate along $e^{(i)}$. However, the inverted and consistent directions need not align with these default axes. Attributing variance to individual coordinates in E therefore does not directly reveal the contributions of these two directions.

To evaluate them, we change from E to an orthonormal basis $U = (u^{(1)}, \ldots, u^{(d)})$ that includes V_{inv} and V_{con}, obtained using the procedure in §4.1. Each new coordinate is obtained by projecting V_{dom} onto the corresponding basis vector: $V_{\mathrm{dom}}^{(U)} = U^\top V_{\mathrm{dom}}, V_{\mathrm{dom}}^{(U)}[i] = u^{(i)\top} V_{\mathrm{dom}}$. Let $\Sigma \in \mathbb{R}^{d\times d}$ denote the covariance matrix of the text representations in the default basis. Under the same change of basis, this matrix becomes: $\Sigma_U = U^\top \Sigma U$. These relations hold for any orthonormal basis U and allow us to express the analysis in terms of the directions of interest. Detailed derivations are provided in Appendix F.3.

Marginal variance ratio. The variance of the samples projected onto V_{dom} is:

$$\mathrm{Var}(V_{\mathrm{dom}}) = V_{\mathrm{dom}}^\top \Sigma V_{\mathrm{dom}}. \tag{1}$$

Because U is orthonormal, the change of basis preserves this total variance: $(V_{\mathrm{dom}}^{(U)})^\top \Sigma_U V_{\mathrm{dom}}^{(U)} = V_{\mathrm{dom}}^\top \Sigma V_{\mathrm{dom}}$. We can therefore decompose the same total variance into contributions associated with the new basis directions.

Specifically, we define the marginal variance associated with direction $u^{(i)}$ as:

$$\mathrm{Var}_{\mathrm{marg}}(V_{\mathrm{dom}}^{(U,i)}) = V_{\mathrm{dom}}^{(U)}[i] \cdot (\Sigma_U V_{\mathrm{dom}}^{(U)})[i]. \tag{2}$$

Here, $V_{\text{dom}}^{(U,i)}$ denotes the component associated with the i-th direction under U. This term accounts for both the variance along $u^{(i)}$ and its covariance with other directions, weighted by the coordinates of V_{dom}. Summing these contributions over the complete basis recovers the total projection variance:

$$\text{Var}(V_{\text{dom}}) = \sum_{i=1}^{d} \text{Var}_{\text{marg}}(V_{\text{dom}}^{(U,i)}). \tag{3}$$

We define the *marginal variance ratio* (MVR) as the ratio of each marginal contribution to the total variance:

$$\text{MVR}(V_{\text{dom}}^{(U,i)}) = \frac{\text{Var}_{\text{marg}}(V_{\text{dom}}^{(U,i)})}{\text{Var}(V_{\text{dom}})}. \tag{4}$$

The ratios sum to one, although individual contributions can be negative due to covariance. Further details are provided in Appendix F.2.

For brevity, we denote the MVR values associated with the inverted and consistent directions by $\text{MVR}(V_{\text{inv}})$ and $\text{MVR}(V_{\text{con}})$. In the next section, we use these values to analyze their contributions to the domain projection variance.

3.3 Empirical Evidence for Feature Duality

We evaluate the contributions and cross-domain HWT–MGT separation of V_{inv} and V_{con} using MVR and projected sample distributions, with experimental details in Appendix D.1.

Figure 2(a) shows that the components at indices 0 and 767, corresponding to the largest and smallest eigenvalues, have the two largest individual MVR values under U. The latter represents the inverted feature V_{inv}, whose contribution is $\text{MVR}(V_{\text{inv}}) = 2.20\%$, while the former corresponds to the consistent feature V_{con}, whose contribution is $\text{MVR}(V_{\text{con}}) = 4.35\%$. To examine how these directions distinguish HWT from MGT across domains, we project sample representations onto V_{dom}, V_{inv}, and V_{con}, as shown in Figures 2(b)–(d). In Figure 2(b), samples from different domains are clearly separated along V_{dom}, while HWT and MGT within each domain show little distinction. In Figure 2(c), HWT and MGT are well separated within each domain along V_{inv}, but their relative ordering reverses across domains. In Figure 2(d), HWT and MGT remain separated along V_{con} with the same relative ordering in both domains.

Together, the contribution analysis and projection distributions support the presence of both inverted and consistent components.

4 PersFix: Detection Adjustment via Cross-Domain Consistent Features

§3 shows that the consistent direction preserves HWT–MGT separation across general and personalized domains, suggesting its potential to improve detection under domain shift. However, this analysis does not specify how to incorporate the consistent signal into an existing detector. We therefore introduce PersFix, which uses sample projections onto this direction to adjust detector scores. We detail the direction extraction procedure and feature-based score adjustment below.

4.1 Cross-Domain Consistent Feature Extraction

Our goal is to identify a direction that separates HWT from MGT with consistent orientation across general and personalized domains. Using a fixed reference LLM, we represent MGT–HWT pairs from the two domains as (g_k^+, g_k^-) and (s_k^+, s_k^-), respectively, where each element is a text representation in $\mathbb{R}^d$. The reference model and probing datasets are described in §D.1.

For $k = 1, \ldots, N$, we construct the within-domain difference vectors $v_{g,k} = g_k^+ - g_k^-$ and $v_{s,k} = s_k^+ - s_k^-$. For a candidate unit direction w, the projections $w^\top v_{g,k}$ and $w^\top v_{s,k}$ measure the MGT–HWT differences along that direction in each domain. Their product is positive when the differences have the same sign and negative when their signs are reversed. We therefore seek a direction that maximizes the aggregate cross-domain projection product:

$$V_{\text{con}} = \underset{\|w\|=1}{\arg\max} \sum_{k=1}^{N} (w^\top v_{g,k})(w^\top v_{s,k}). \tag{5}$$

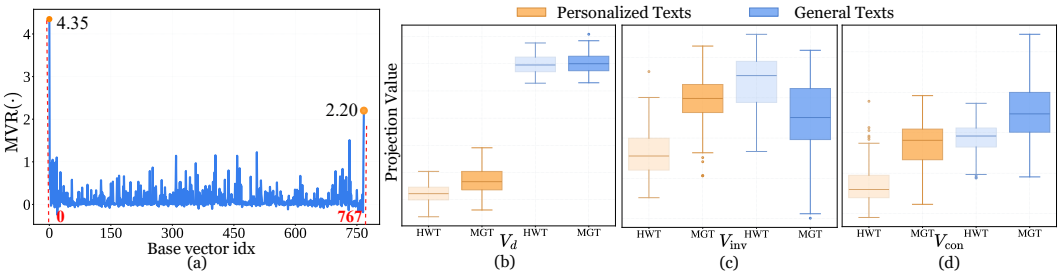


Figure 2: Feature decomposition analysis. Panel (a) reports the marginal variance ratios of basis directions with respect to the domain feature. Panels (b)–(d) show sample distributions projected onto V_{dom}, V_{inv}, and V_{con}, respectively. The inverted direction exhibits reversed HWT–MGT ordering across domains, whereas the consistent direction preserves the same ordering.

To solve this objective, we construct the symmetric matrix:

$$M = \frac{1}{2}\sum_{k=1}^{N}\left(v_{g,k}v_{s,k}^{\top} + v_{s,k}v_{g,k}^{\top}\right). \tag{6}$$

The objective in Equation 5 then equals $w^{\top}Mw$. Under the unit-norm constraint, its maximum is attained by an eigenvector corresponding to the largest eigenvalue of M. We select this unit eigenvector as V_{con} and use it for the score adjustment described next. Further properties are detailed in Appendix F.4.

The same eigendecomposition also provides the directions used in §3: the eigenvector corresponding to the smallest eigenvalue minimizes the projection-product objective and defines V_{inv}, while the full orthonormal eigenbasis forms U for the MVR analysis.

4.2 Feature-Based Score Adjustment

Given the consistent direction V_{con} extracted in §4.1, PersFix adjusts an existing detector by incorporating the projection of each sample onto this direction. For an input text x, let $r(x) \in \mathbb{R}^d$ denote its representation obtained using the same reference model, layer, and token-selection procedure as in feature extraction. Let $s_D(x)$ denote the original detector's scalar detection score. We treat the detector as a black box and adjust its score as follows:

$$s_{\text{PersFix}}(x) = s_D(x) + \alpha\big(V_{\text{con}}^{\top}r(x) + \beta\big), \tag{7}$$

where α controls the adjustment strength and β is a fixed offset applied to the projection. The projection provides a sample-dependent signal based on the cross-domain consistent direction, while α determines its contribution relative to the original detector score. This adjustment leaves the detector's architecture and parameters unchanged.

For each evaluation group and each detector, we select α on the development set while keeping V_{con} fixed. We evaluate a candidate set $\mathcal{A}$ containing 200 values in $[-1, 1]$. For each candidate, we compute AUROC and TPR@5FPR using the adjusted scores. To combine the two metrics, we normalize each across the candidate set:

$$\hat{m}(\alpha) = \frac{m(\alpha) - \min_{\alpha'\in\mathcal{A}} m(\alpha')}{\max_{\alpha'\in\mathcal{A}} m(\alpha') - \min_{\alpha'\in\mathcal{A}} m(\alpha')}. \tag{8}$$

We then select the adjustment strength that maximizes their sum:

$$\alpha^{\star} = \arg\max_{\alpha\in\mathcal{A}} \left[\hat{m}_{\text{AUROC}}(\alpha) + \hat{m}_{\text{TPR}}(\alpha)\right]. \tag{9}$$

We apply the selected $\alpha^{\star}$ to the corresponding test set without further tuning. We fix $\beta = -50$ for all experiments. Compute resources are reported in Appendix D.6.

Setting		RoBERTa	DeT	Ent	LL	LR	SD	LRR	Ld	Ld++	DNA	Spec++
AUROC (%)												
General	w/o	49.9	66.6	34.9	79.8	82.9	84.5	87.3	91.7	90.6	92.5	90.9
	w/	77.0 (+27.1)	84.7 (+18.0)	73.0 (+38.0)	89.4 (+9.5)	91.0 (+8.0)	92.0 (+7.5)	92.2 (+4.9)	93.3 (+1.6)	93.9 (+3.3)	96.0 (+3.5)	93.9 (+3.0)
Prompted	w/o	51.6	49.2	36.7	71.7	69.4	82.6	62.4	54.8	82.2	88.6	82.0
	w/	69.2 (+17.7)	69.7 (+20.5)	70.5 (+33.8)	83.1 (+11.4)	82.8 (+13.4)	89.2 (+6.5)	80.8 (+18.4)	71.4 (+16.7)	88.5 (+6.3)	90.5 (+1.9)	86.5 (+4.5)
Fine-tuned	w/o	50.3	70.5	61.1	25.6	27.3	20.2	35.4	66.0	49.3	36.6	70.0
	w/	83.8 (+33.6)	87.5 (+17.0)	85.4 (+24.3)	82.9 (+57.2)	83.0 (+55.8)	79.5 (+59.3)	82.9 (+47.5)	84.9 (+18.9)	82.7 (+33.4)	83.5 (+46.9)	87.1 (+17.2)
TPR@5FPR (%)												
General	w/o	6.7	33.0	5.0	47.0	53.2	70.5	66.4	76.0	75.2	79.7	74.9
	w/	40.5 (+33.8)	58.4 (+25.4)	25.5 (+20.5)	63.3 (+16.3)	68.2 (+15.0)	79.0 (+8.5)	76.1 (+9.8)	79.4 (+3.4)	80.9 (+5.7)	86.1 (+6.4)	80.5 (+5.5)
Prompted	w/o	5.1	5.8	1.2	27.3	28.3	52.6	28.1	23.9	51.8	57.0	49.3
	w/	14.6 (+9.5)	14.8 (+9.0)	15.7 (+14.5)	37.3 (+10.0)	36.8 (+8.6)	58.0 (+5.4)	35.5 (+7.4)	31.9 (+8.0)	54.4 (+2.6)	58.3 (+1.3)	50.5 (+1.2)
Fine-tuned	w/o	4.2	29.6	9.5	0.8	0.8	0.5	0.9	12.1	7.8	2.5	29.5
	w/	42.4 (+38.2)	54.2 (+24.5)	46.9 (+37.4)	40.2 (+39.4)	40.9 (+40.1)	35.0 (+34.6)	40.2 (+39.3)	43.6 (+31.5)	40.4 (+32.6)	42.5 (+40.0)	52.5 (+23.1)

Table 1: Average AUROC and TPR@5FPR of each detector with (w/) and without (w/o) PERSFIX across the three personalization settings. Each w/ cell is reported as performance (+improvement). The deeper the green, the larger the improvement.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate PERSFIX on both personalized and general domains. For the personalized domain, we use two datasets introduced by Gao et al. (2026). The first is `Stylo-Blog`, corresponding to prompt-based personalization *(Prompted setting)*. It contains highly personalized blog posts and their LLM continuations. The models are prompted to mimic personal traits. The second is `Stylo-Literary`, corresponding to fine-tuning-based personalization *(Fine-tuned setting)*. It contains literary works and their LLM continuations. The models are fine-tuned on authors' corpora to reproduce their writing styles. For the general domain *(General setting)*, we use the English subset of `M4` (Wang et al., 2024). It covers texts from five sources and their LLM rewrites. Detailed statistics and introductions are in Appendix D.2. Each benchmark is further divided into groups by generator LLM and scenario. For `Stylo-Literary`, the scenario is the author. For `M4`, it is the source domain. `Stylo-Blog` is only grouped by LLMs. Each group contains around 1,000 samples. For each group, we randomly sample 200 HWT–MGT pairs as a development set to select the adjustment strength α, and use the remaining pairs as a test set. Compute resources are reported in Appendix D.6.

Detectors and metrics. We use nine widely adopted MGT detectors, including the training-based RoBERTa (Liu et al., 2019) and DeTeCtive (**DeT**) (Guo et al., 2024), and the training-free Log-Likelihood (**LL**) (Solaiman et al., 2019), LogRank (**LR**) (Solaiman et al., 2019), DetectLRR (**LRR**) (Su et al., 2023a), Entropy (**Ent**) (Gehrmann et al., 2019; Ippolito et al., 2020), Sampling Discrepancy (**SD**), Lastde (**Ld**), Lastde++ (**Ld++**) (Xu et al., 2025), DNA-DetectLLM (**DNA**) (Zhu et al., 2025), and SpecDetect++ (**Spec++**) (Luo et al., 2026); we use these abbreviations throughout the paper and defer implementation details to Appendix D.3. We report AUROC, the area under the receiver operating characteristic curve, and TPR@5FPR, the true positive rate at a fixed false positive rate of 5%, following (Bao et al., 2024; Su et al., 2023a).

5.2 EXPERIMENTAL RESULTS

We conduct two types of experiments. The first is under the in-distribution setting: we select the adjustment strength α using the development set from one group, and test on the same group. This shows how PERSFIX works when the data distribution does not change. The second is under the out-of-distribution setting: we select α on one group, and apply it to a different group. This tests whether the adjustment still works when the data distribution changes.

5.2.1 IN-DISTRIBUTION RESULTS

Overall performance. Aggregating all detector–dataset pairs, we observe a substantial improvement after adjustment, as shown in Figure 3. On both metrics, the overall performance distribution shifts upward, and the upper bound is clearly raised. In settings where the unadjusted detectors struggle, PERSFIX recovers reliable performance; in easier settings, adjustment does not cause notable

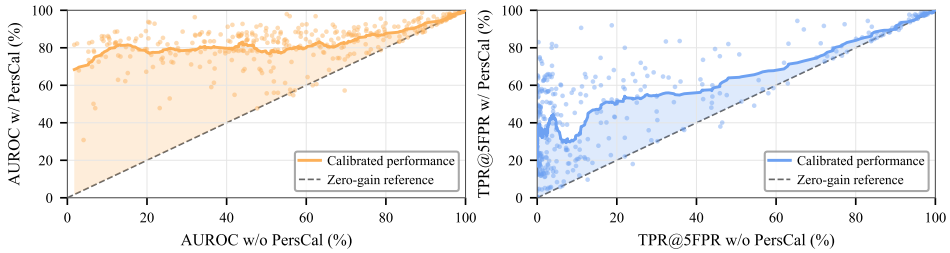


Figure 3: Overall effect of PERSFIX on (a) AUROC and (b) TPR@5FPR. The zero-improvement reference means w/ PERSFIX = w/o PERSFIX. Most points lie above the reference, indicating that PERSFIX consistently improves detection performance across generator LLMs and scenarios.

degradation. These trends suggest that PERSFIX can restore detector behavior in cases affected by feature inversion while preserving performance elsewhere.

Performance across different personalized settings. We report several observations from Table 1.

(1) *All detectors benefit from adjustment.* The improvement on average holds for AUROC and TPR@5FPR across all three personalization settings, with no performance drop on any of the groups.

(2) *The relative improvement ordering is consistent across detectors.* For example, LL improves AUROC by $+57.2$ in the Fine-tuned setting, which is much larger than $+11.4$ in Prompted and $+9.5$ in General. Similar patterns appear for LR and LRR. For TPR@5FPR, detectors like Ent, LL, and LR follow the same order, with the largest gains in Fine-tuned, followed by General, and then Prompted.

(3) *Detectors with weak baselines improves much more.* In the Fine-tuned setting, SD, which has 20.2% AUROC and 0.5% TPR@5FPR, recovers to 79.5% and 35.0%, improvementing +59.3 on AUROC and +34.6 on TPR@5FPR; while DeT only has improvements +17.0 on AUROC and +24.5 on TPR@5FPR. This shows that PERSFIX delivers larger improvements precisely when the detector is more affected by inverted features.

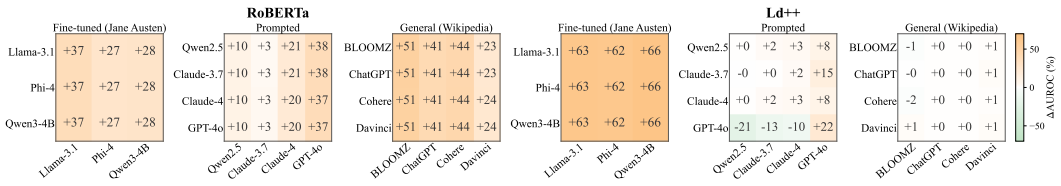
(4) *Strong detectors also benefit, but the improvement is smaller.* In the General setting, Ld++ improves from 90.6% to 93.9% in AUROC and 75.2% to 80.9% in TPR@5FPR, improvementing only +3.3 and +5.7; the moderate-baseline RoBERTa (49.9% AUROC, 6.7% TPR@5FPR) improves to 77.0% and 40.5%, with much larger improvements of +27.1 and +33.8. A per-subdomain breakdown of the General setting, together with a possible explanation of why PERSFIX also helps there, is given in Appendix E.3.

Overall, PERSFIX improves performance across different detectors, metrics, and personalization settings, with the largest improvements under strong personalization or when the detector is severely affected by feature inversion. Full per-subset and per-generator results are in Appendix D.4.

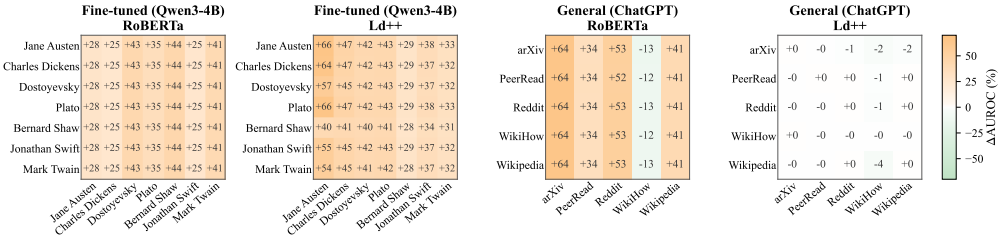
5.2.2 OUT-OF-DISTRIBUTION TRANSFER

We study two types of out-of-distribution transfer. *Cross-model transfer* keeps the scenario fixed and changes the generator LLM; *Cross-scenario transfer* keeps the generator fixed and changes the scenario. The Prompted setting has only one scenario, so we only test the first case on it. In the main text, we report results for two detectors, RoBERTa (training-based) and Ld++ (training-free), with Figure 4(a) and (b) showing similar patterns. Results for the other detectors are in Appendix D.5. Overall, PERSCAL remains effective under out-of-distribution transfer, with consistent improvements observed across most settings and detector types.

(1) *The two kinds of detectors behave differently under adjustment.* RoBERTa improves clearly in all three settings. Its improvements are around $+25$ to $+45$ in Fine-tuned, $+3$ to $+38$ in Prompted, and $+23$ to $+64$ in General. Ld++ behaves less consistently. It improves the most in Fine-tuned, where many cells are above $+40$ and some reach $+66$. In General, the improvement is small, within ± 4, and in Prompted it stays close to zero. This difference comes from their baselines. RoBERTa starts near 50% in all settings, so it has room to improve everywhere. Ld++ is already strong in General but weak in personalized settings, so most improvements appear in Fine-tuned.



(a) Cross-model transfer: scenario fixed to "Jane Austen" on Fine-tuned, and "Wikipedia" on General.



(b) Cross-scenario transfer: generator fixed to Qwen3-4B on Fine-tuned setting and ChatGPT on General setting.

Figure 4: Out-of-distribution transfer heatmaps. Each panel shows how adjustment transfers across groups. Rows are groups deriving α and columns are groups tested. Each cell reports the mean improvement of AUROC. Model names are shortened for clarity: Llama-3.1(-8B), Qwen-2.5(-72B), Claude-3.7(-Sonnet), and Claude-4(-Sonnet).

(2) *Transfer is stable within each setting.* Within most panels, rows are very similar and remain close to the diagonal, which indicates consistent behavior across sources. In the Fine-tuned setting, RoBERTa stays in a narrow range, around $+27$ to $+37$ in cross-model transfer and $+25$ to $+44$ in cross-scenario transfer, while Ld++ shows the same pattern with values from $+28$ to $+66$ and only small variation across rows. A similar trend appears in the General setting, where RoBERTa keeps most values between $+23$ and $+51$. These patterns suggest that different sources produce nearly the same result on a given target. As a result, the α selected on one group can be directly applied to other groups in the same setting without noticeable loss.

We also observe a few special cases. In the Prompted setting, Ld++ only improves on one target while others stay near zero, likely because its baseline AUROC is already strong (82.2%), leaving little room for adjustment. In the General setting, one target shows a consistent drop of about -12 for RoBERTa, even on its diagonal, suggesting that this subdomain conflicts with the consistent feature used in adjustment, so the logit adjustment hurts performance.

6 Analysis and Discussion

6.1 Ablation studies

Setup. To determine whether PersFix's improvement comes from combining the detector with V_{con} or from V_{con} alone, we evaluate five variants: the detector, V_{random}, V_{con}, detector+V_{random}, and detector+V_{con} (PersFix), where V_{random} denotes a random vector. We divide the 45 detector–dataset groups into five ranges according to the detector's original AUROC. For each combined variant, we select the best α on the development set. When using either direction alone, we use its projection values as detection scores.

Results. Table 2 shows that V_{con} provides a useful signal for distinguishing AI-generated text, consistent with the objective described in §4.1. Across all 45 groups, detector+V_{con} has the highest mean AUROC. When the detector's original AUROC is below 0.65, V_{con} accounts for much of the combined performance. However, in groups where the detector's original AUROC is between 0.65 and 0.8., detector+V_{con} reaches 0.807 AUROC, higher than either component alone. For groups with higher original detector performance, the detector accounts for a larger share of the combined result. Besides, the mean selected $|\alpha^\star|$ decreases from approximately 3 for the weakest detectors to below 0.25 for the strongest, indicating that stronger detectors require less intervention from V_{con}. Appendix E further compares V_{con} with other concept directions (Figure 13).

Detector AUROC	Det.	V_{random}	Det.+V_{random}	V_{con}	Det.+V_{con} (PERSFIX)	$\lvert\alpha^\star\rvert$
< 0.5	0.277	0.567	0.498	**0.811**	<u>0.736</u>	3.00
0.5–0.65	0.582	0.547	0.593	**0.814**	<u>0.813</u>	2.90
0.65–0.8	0.703	0.527	0.703	<u>0.742</u>	**0.807**	1.24
0.8–0.9	0.861	0.608	<u>0.875</u>	0.614	**0.877**	0.23
> 0.9	0.972	0.708	**0.978**	0.753	<u>0.977</u>	0.15
All	0.566	0.582	0.659	<u>0.768</u>	**0.812**	2.00

Table 2: Mean AUROC across detector–dataset groups grouped by the original detector AUROC. **Bold** and <u>underline</u> indicate the best and second-best variants in each row.

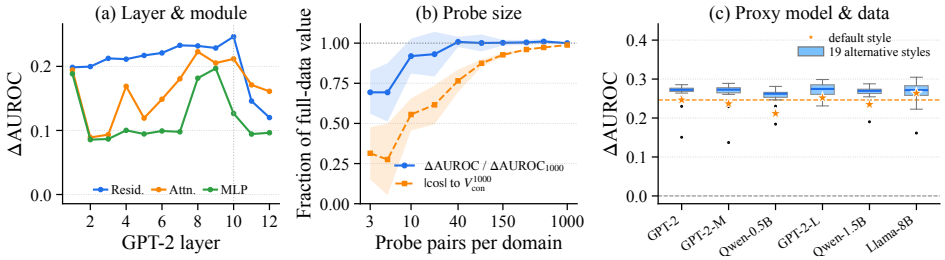


Figure 5: Sensitivity of PERSFIX to the extraction of V_{con}: (a) GPT-2 layer and module, (b) probe pairs per domain, and (c) proxy model and extraction data. Panel (b) reports AUROC improvement and direction similarity relative to extraction with 1,000 pairs.

6.2 DISCUSSION

In this section, we discuss several factors that may affect the effectiveness of the intervention.

Which layer and module should be used? We extract V_{con} from the attention output, MLP output, and residual stream at each GPT-2 layer. Figure 5(a) shows that attention and MLP outputs vary more across layers, while their AUROC improvements generally remain below those of the residual stream. The residual stream performs best in the middle-to-late layers, suggesting that it provides a more stable representation of the personalized–general distinction.

How much probe data is needed? Figure 5(b) compares directions extracted using different numbers of probe pairs per domain against a direction extracted using 1,000 pairs. AUROC improvement rises rapidly with probe size and approaches the 1,000-pair result by around 150 pairs. Direction similarity increases more gradually, indicating that a direction can support effective adjustment before it closely matches the full-data estimate.

How sensitive is extraction to the proxy model and data? Figure 5(c) compares GPT-2 models up to 774M (Radford et al., 2019), Qwen2.5-0.5B/1.5B (Qwen Team, 2024), and Llama-3.1-8B (Dubey et al., 2024). For each model, we evaluate 19 alternative extraction pairs using personalized texts from different authors and writing styles, with XSum as the general-text source. AUROC improvements vary modestly across models and extraction pairs, with no consistent advantage for larger models. This suggests that the personalized–general distinction can be represented by relatively small models and is less sensitive to the tested forms of personalization.

7 CONCLUSION AND FUTURE WORK

We addressed the feature-inversion trap by revealing the duality of domain-discriminative text features, using MVR to identify a consistent component that carries a reliable cross-domain signal, and proposing PERSFIX based on the signal, stabilizing detectors across personalization settings. Since PERSFIX treats the detector as a black box and only shifts its output logit, it applies to both training-based and training-free detectors at negligible cost. These results suggest that stable discriminative signals coexist with inverted ones, and that making the stable part explicit is enough to restore detection under personalization. In future work, we plan to design detectors that natively use consistent features for robust personalized detection.

AI USE STATEMENT

We used generative AI tools in two ways: to help write and edit the code for representation extraction, logit adjustment, and evaluation, and to clean and reformat the datasets and result tables. We also used them to polish the writing. All scientific content is the authors' own: the feature-duality hypothesis, the contribution metric, the design of PERSFIX, the derivations in Appendix F, the experimental design, and the interpretation of results. We did not use generative AI to develop the theory, formulate claims, write proofs, propose hypotheses, or design experiments. Synthetic data generation, translation, and qualitative data analysis do not apply to this work.

We reviewed all AI-assisted work. The authors tested every AI-written script before use, and all reported numbers come from these verified scripts. AI-assisted data cleaning was checked against the source data, and AI-suggested references were read before citation. We take full responsibility for the final content of this work.

ETHICS STATEMENT

This work aims to make machine-generated text detection more stable in personalized settings. More reliable detection can help address risks such as identity impersonation, misinformation, and the misuse of a person's writing style. Our experiments rely on three public corpora, the English subset of `M4`, the Blog Authorship Corpus, and the Gutenberg Book Corpus, all of which were released for research use. We did not collect any new data from people, and the study involved no human subjects. We also do not train or release new text generators, so our work does not add generation risk beyond that of the systems we study. At the same time, PERSFIX improves AUROC but does not make detection exact, and we would caution against relying on it alone in high-stakes settings such as academic misconduct cases. Our analysis also reveals which feature directions detectors depend on, which could in principle help someone evade detection. For this reason we plan to release the extracted directions under controlled access. We are not aware of any conflicts of interest or other ethical concerns.

REPRODUCIBILITY STATEMENT

We have taken several steps to make this work reproducible. Section 3 and Appendix D.1 describe the feature-duality analysis, including the representation space and the contribution metric. The method is described in Section 4.1 and Section 4.2, and its implementation details are given in Appendix C. Section 5.1 and Appendix D.2 describe the three datasets and the split used to select the adjustment strength α, and Appendix D.3 gives the configuration of all nine detectors. Appendix F gives the complete derivations. In addition, the anonymous repository linked in the abstract contains the source code for feature extraction, score adjustment, and evaluation, which can be run directly to reproduce our results.

REFERENCES

Mervat Abassy, Kareem Elozeiri, Alexander Aziz, Minh Ngoc Ta, Raj Vardhan Tomar, Bimarsha Adhikari, Saad El Dine Ahmed, Yuxia Wang, Osama Mohammed Afzal, Zhuohan Xie, Jonibek Mansurov, Ekaterina Artemova, Vladislav Mikhailov, Rui Xing, Jiahui Geng, Hasan Iqbal, Zain Muhammad Mujahid, Tarek Mahmoud, Akim Tsvigun, Alham Fikri Aji, Artem Shelmanov, Nizar Habash, Iryna Gurevych, and Preslav Nakov. LLM-DetectAIve: a tool for fine-grained machine-generated text detection. In *Proc. of EMNLP*, pp. 336–343, 2024.

Kenneth Alperin, Rohan Leekha, Adaku Uchendu, Trang Nguyen, Srilakshmi Medarametla, Carlos Levya Capote, Seth Aycock, and Charlie Dagli. Masks and mimicry: Strategic obfuscation and impersonation attacks on authorship verification. In *Proc. of NLP4DH*, pp. 102–116, 2025.

Biyang an author = "Guo. title = "how close is chatgpt to human experts? comparison corpus, evaluation, and detection". *ArXiv preprint*, abs/2301.07597, 2023.

Guangsheng Bao, Yanbin Zhao, Zhiyang Teng, Linyi Yang, and Yue Zhang. Fast-detectGPT: Efficient zero-shot detection of machine-generated text via conditional probability curvature. In *Proc. of ICLR*, 2024.

Christopher M. Bishop and Nasser M. Nasrabadi. *Pattern Recognition and Machine Learning*. *J. Electronic Imaging*, pp. 049901, 2007.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Proc. of NeurIPS*, 2020.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, and et al. The llama 3 herd of models. *ArXiv preprint*, abs/2407.21783, 2024.

Liam Dugan, Daphne Ippolito, Arun Kirubarajan, and Chris Callison-Burch. RoFT: A tool for evaluating human detection of machine-generated text. In *Proc. of EMNLP*, pp. 189–196, 2020.

Liam Dugan, Alyssa Hwang, Filip Trhlík, Andrew Zhu, Josh Magnus Ludan, Hainiu Xu, Daphne Ippolito, and Chris Callison-Burch. RAID: A shared benchmark for robust evaluation of machine-generated text detectors. In *Proc. of ACL*, 2024.

Lang Gao, Xuhui Li, Chenxi Wang, Mingzhe Li, Wei Liu, Zirui Song, Jinghui Zhang, Rui Yan, Preslav Nakov, and Xiuying Chen. When personalization tricks detectors: The feature-inversion trap in machine-generated text detection. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 43143–43171, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1998. URL https://aclanthology.org/2026.acl-long.1998/.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander Rush. GLTR: Statistical detection and visualization of generated text. In *Proc. of ACL*, pp. 111–116, 2019.

Martin Gerlach and Francesc Font-Clos. A standardized project gutenberg corpus for statistical analysis of natural language and quantitative linguistics. *Entropy*, 2020.

Xun Guo, Yongxin He, Shan Zhang, Ting Zhang, Wanquan Feng, Haibin Huang, and Chongyang Ma. Detective: Detecting ai-generated text via multi-level contrastive learning. *Proc. of NeurIPS*, 2024.

Abhimanyu Hans, Avi Schwarzschild, Valeriia Cherepanova, Hamid Kazemi, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Spotting LLMs with binoculars: Zero-shot detection of machine-generated text. In *Proc. of ICML*, volume 235, pp. 17519–17537, 2024.

Xinlei He, Xinyue Shen, Zeyuan Chen, Michael Backes, and Yang Zhang. Mgtbench: Benchmarking machine-generated text detection. In *Proc. of CCS*, 2024.

Steffen Herbold, Alexander Trautsch, Zlata Kikteva, and Annette Hautli-Janisz. Large language models can impersonate politicians and other public figures. *ArXiv preprint*, abs/2407.12855, 2024.

Abe Hou, Jingyu Zhang, Tianxing He, Yichen Wang, Yung-Sung Chuang, Hongwei Wang, Lingfeng Shen, Benjamin Van Durme, Daniel Khashabi, and Yulia Tsvetkov. SemStamp: A semantic watermark with paraphrastic robustness for text generation. In *Proc. of NAACL*, pp. 4067–4082, 2024.

Xiaomeng Hu, Pin-Yu Chen, and Tsung-Yi Ho. Radar: Robust ai-text detection via adversarial learning. In *Proc. of NeurIPS*, volume 36, pp. 15077–15095, 2023.

Yue Huang, Lichao Sun, Haoran Wang, Siyuan Wu, Qihui Zhang, et al. Trustllm: Trustworthiness in large language models. In *Proc. of ICML*, 2024.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. Automatic detection of generated text is easiest when humans are fooled. In *Proc. of ACL*, pp. 1808–1822, 2020.

Kaijie Jiao, Quan Wang, Licheng Zhang, Zikang Guo, and Zhendong Mao. M-RangeDetector: Enhancing generalization in machine-generated text detection through multi-range attention masks. In *Proc. of ACL Findings*, 2025.

Minseok Jung, Cynthia Fuertes Panizo, Liam Dugan, Yi R., Fung, Pin-Yu Chen, and Paul Pu Liang. Group-adaptive threshold optimization for robust ai-generated text detection. *ArXiv preprint*, abs/2502.04528, 2025.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *ArXiv preprint*, abs/1907.11692, 2019.

Yu Liu, Lang Gao, Mingxin Yang, Yu Xie, Ping Chen, Xiaojin Zhang, and Wei Chen. Vuldetectbench: Evaluating the deep capability of vulnerability detection with large language models. *ArXiv preprint*, abs/2406.07595, 2024.

Haitong Luo, Weiyao Zhang, Suhang Wang, Wenji Zou, Chungang Lin, Xuying Meng, and Yujun Zhang. Specdetect: simple, fast, and training-free detection of llm-generated text via spectral analysis. In *Proceedings of the Fortieth AAAI Conference on Artificial Intelligence and Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence and Sixteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'26/IAAI'26/EAAI'26. AAAI Press, 2026. ISBN 978-1-57735-906-7. doi: 10.1609/aaai.v40i38.40510. URL https://doi.org/10.1609/aaai.v40i38.40510.

Dominik Macko, Robert Moro, Adaku Uchendu, Jason Lucas, Michiharu Yamashita, Matúš Pikuliak, Ivan Srba, Thai Le, Dongwon Lee, Jakub Simko, and Maria Bielikova. MULTITuDE: Large-scale multilingual machine-generated text detection benchmark. In *Proc. of EMNLP*, pp. 9960–9987, 2023.

Dominik Macko, Jakub Kopál, Robert Moro, and Ivan Srba. MultiSocial: Multilingual benchmark of machine-generated text detection of social-media texts. In *Proc. of ACL*, pp. 727–752, 2025.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In *Proc. of ICML*, pp. 39643–39666, 2024.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

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

Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W. Pennebaker. Effects of age and gender on blogging. In *Proc. of AAAI*, pp. 199–205, 2006.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, Miles McCain, Alex Newhouse, Jason Blazakis, Kris McGuffie, and Jasmine Wang. Release strategies and the social impacts of language models. *ArXiv preprint*, abs/1908.09203, 2019.

Jinyan Su, Terry Zhuo, Di Wang, and Preslav Nakov. DetectLLM: Leveraging log rank information for zero-shot detection of machine-generated text. In *Proc. of EMNLP Findings*, 2023a.

Jinyan Su, Terry Yue Zhuo, Di Wang, and Preslav Nakov. DetectLLM: Leveraging log rank information for zero-shot detection of machine-generated text. In *Proc. of EMNLP*, 2023b.

Huachen Tan, Moming Duan, Duo Liu, Haojie Lu, Yuexin Mu, Longyi Zhou, Ao Ren, Yujuan Tan, and Kan Zhong. Rethinking literary plagiarism in llms through the lens of copyright laws. In *Proc. of ACML*, volume 260, pp. 1000–1015, 2025.

Yuchuan Tian, Hanting Chen, Xutao Wang, Zheyuan Bai, QINGHUA ZHANG, Ruifeng Li, Chao Xu, and Yunhe Wang. Multiscale positive-unlabeled detection of AI-generated texts. In *Proc. of ICLR*, 2024.

Eduard Tulchinskii, Kristian Kuznetsov, Kushnareva Laida, Daniil Cherniavskii, Sergey Nikolenko, Evgeny Burnaev, Serguei Barannikov, and Irina Piontkovskaya. Intrinsic dimension estimation for robust detection of AI-generated texts. In *Proc. of NeurIPS*, 2023.

Adaku Uchendu, Zeyu Ma, Thai Le, Rui Zhang, and Dongwon Lee. TURINGBENCH: A benchmark environment for Turing test in the age of neural text generation. In *Proc. of EMNLP Findings*, pp. 2001–2016, 2021.

Ben Wang and Aran Komatsuzaki. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model, 2021.

Yanbo Wang, Zixiang Xu, Yue Huang, Xiangqi Wang, Zirui Song, Lang Gao, Chenxi Wang, Xiangru Tang, Yue Zhao, Arman Cohan, et al. Dyflow: Dynamic workflow framework for agentic reasoning. In *Proc. of NeurIPS*, 2025.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Toru Sasaki, Thomas Arnold, Alham Fikri Aji, Nizar Habash, Iryna Gurevych, and Preslav Nakov. M4: Multi-generator, multi-domain, and multi-lingual black-box machine-generated text detection. In *Proc. of EACL*, 2024.

Dongjun Wei, Minjia Mao, Xiao Fang, and Michael Chau. Short-PHD: Detecting short LLM-generated text with topological data analysis after off-topic content insertion. In *Proc. of COLM*, 2025.

Junxi Wu, Jinpeng Wang, Zheng Liu, Bin Chen, Dongjian Hu, Hao Wu, and Shu-Tao Xia. Moses: Uncertainty-aware ai-generated text detection via mixture of stylistics experts with conditional thresholds. *ArXiv preprint*, abs/2509.02499, 2025.

Yihuai Xu, Yongwei Wang, Yifei Bi, Huangsen Cao, Zhouhan Lin, Yu Zhao, and Fei Wu. Training-free LLM-generated text detection by mining token probability sequences. In *Proc. of ICLR*, 2025.

Xianjun Yang, Wei Cheng, Yue Wu, Linda Ruth Petzold, William Yang Wang, and Haifeng Chen. DNA-GPT: Divergent n-gram analysis for training-free detection of GPT-generated text. In *Proc. of ICLR*, 2024.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. Defending against neural fake news. In *Proc. of NeurIPS*, pp. 9051–9062, 2019.

Xiaowei Zhu, Yubing Ren, Fang Fang, Qingfeng Tan, Shi Wang, and Yanan Cao. DNA-detectLLM: Unveiling AI-generated text via a DNA-inspired mutation-repair paradigm. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=yQoHUijSHx.

A Limitations and Broader Impacts

A.1 Limitations

This study has several limitations that do not change its main conclusions. First, we report AUROC and TPR@5FPR. The two metrics together cover overall ranking quality and a low-false-alarm operating point, but they do not describe behavior at every deployment threshold. Adding more metrics could give a finer view. Second, PersFix is a static external corrector. It does not change the detector's architecture or training. This makes PersFix easy to plug in, but joint training of the detector and the corrector may push the gain further. Third, we test on three English benchmarks, namely `Stylo-Literary`, `Stylo-Blog`, and `M4`. Other scenarios such as multilingual or multimodal data are not covered here.

A.2 Broader Impacts

This work improves the reliability of MGT detection under personalization. It helps reduce risks such as impersonation, misinformation, and misuse of personal writing styles. By improving stability across domains, it supports safer deployment of language models in real applications. PersFix is simple and does not modify the detector itself, which makes it easy to apply to existing systems. This lowers the cost of improving detection quality and can benefit a wide range of platforms. We hope PersFix contributes to building more trustworthy and robust MGT detection systems.

B Related Work

Here we expand the related work discussion on benchmarks and detectors in the main text.

B.1 MGT Detection Benchmarks

Earlier work establishes the binary-detection task distinguishing whether given texts are generated by AI on a smaller scale. Solaiman et al. (2019) creates a large dataset of HWT and MGT generated by GPT-2. Besides, TuringBench (Uchendu et al., 2021) collects samples from 19 generators on news and pushes detection toward a multi-source setting. HC3 (author = "Guo, 2023) pairs human answers with ChatGPT answers across various domains including QA, finance, and medicine. Later benchmarks propose new challenges to the task. One is multilingual detection, where detectors need to classify texts of different languages (Macko et al., 2023; 2025). Another line of work is detecting AI-generated text spans in contents jointly written by human and machine. Representitive works include RoFT (Dugan et al., 2020) and LLM-DetectAIve (Abassy et al., 2024).

Benchmarks that target the personalized scenario remain scarce. Preliminary works such as (Alperin et al., 2025) study LLM-based impersonation attacks on authorship verification. Tan et al. (2025) collect literary imitation samples for copyright analysis. However, none of them comprehensively evaluates personalized MGT detection across many authors and many generators, which is the setting our study targets.

B.2 MGT Detection Methods

Training-based methods. The earliest training-based detectors fine-tune a pretrained encoder as a binary classifier. Solaiman et al. (2019) is the first to fine-tune RoBERTa on labeled HWT and MGT, and it remains a classial baseline detector today. Zellers et al. (2019) follows the same template but ties the detector to a same-family generator, boosting in-distribution accuracy. More recently, some works propose new training paradigms, which further improves detectors' performance (Hu et al., 2023; Hou et al., 2024).

Training-free methods. Training-free detectors read fixed signals from a frozen language model. The simplest signals come from token-level distributional statistics such as Log-Likelihood and LogRank (Solaiman et al., 2019). Another group of work reads the curvature of the model's likelihood landscape around the input text, on the idea that machine text sits closer to a local likelihood maximum than human text does, such as DNA-GPT (Yang et al., 2024) and Binoculars (Hans et al., 2024). Some approaches also look at sequential and structural regularities, like GPTZero [1]. Compared to training-based methods, training-free detectors are more interpretable, since each score comes from a clear statistic. They are also more stable across runs, as they use no learned parameters. However, they generalize less, because each signal reflects a narrow assumption about machine writing. This limits their performance under personalized settings, where they degrade more.

C PersFix Details

PersFix has two stages: feature extraction and feature-based adjustment. The extraction stage builds the consistent feature V_{con} from the latent space of an LLM, and the adjustment stage shifts each detector's logit by a scalar projection on V_{con}. Algorithm 1 shows the extraction stage.

[1] https://gptzero.me/

Generator	Wiki.	Reddit	WikiHow	PeerR.	arXiv	Total
Davinci	3,000	3,000	3,000	2,323	3,000	14,323
ChatGPT	2,995	3,000	3,000	2,344	3,000	14,339
Cohere	2,336	1,220	2,999	1,702	3,000	11,257
BLOOMZ	3,000	3,000	3,000	2,340	3,000	14,340

Table 3: Per-(generator, subdomain) counts of the M4 English subset. Each value is the number of paired HWTs and MGTs. Wiki. and PeerR. abbreviate Wikipedia and PeerRead.

We collect MGT/HWT representations from two domains: a general domain G^+/G^- and a personalized domain S^+/S^-. We then form the per-domain MGT$-$HWT difference vectors V_{g} and V_{s}. The matrix M aggregates symmetric outer products of these difference vectors across the two domains. Its top eigenvector picks the direction along which the MGT$-$HWT contrast is largest and consistent across the two domains, which is exactly the consistent feature V_{con}. We use this V_{con} in the adjustment stage of PERSFIX.

Algorithm 1 Extracting V_{con}.

Input: MGT/HWT representations from two domains: G^+, G^-, S^+, S^-;
Output: the consistent feature V_{con}.

$V_{\mathrm{g}} \leftarrow \{\, g^+ - g^- \mid g^+ \in G^+,\, g^- \in G^- \,\}$
$V_{\mathrm{s}} \leftarrow \{\, s^+ - s^- \mid s^+ \in S^+,\, s^- \in S^- \,\}$
$M \leftarrow 0$
for all $(v_{\mathrm{g}}, v_{\mathrm{s}}) \in V_{\mathrm{g}} \times V_{\mathrm{s}}$ **do**
    $M \leftarrow M + \frac{1}{2}(v_{\mathrm{g}} v_{\mathrm{s}}^\top + v_{\mathrm{s}} v_{\mathrm{g}}^\top)$
end for
$M = W \Sigma_M W^\top$ ▷ Eigenvalue decomposition of symmetric M
$w \leftarrow W[:, 0] / \|W[:, 0]\|$ ▷ Eigenvector of the largest eigenvalue
return $V_{\mathrm{con}} \leftarrow w$

D DETAILS ABOUT EXPERIMENTS

D.1 SETUP FOR FEATURE-DUALITY ANALYSIS

We use GPT-2 (Brown et al., 2020) as the reference model and focus on the latent space of the residual stream at the 10th layer. Our analysis is based on the representations of the final tokens in this space. GPT-2 is lightweight and easy to deploy, and it is absent from all benchmark generators. Our ablation in Appendix E further examines the robustness of V_{con} across feature types. Two small probing datasets are built for evaluation. For the general domain, we take 150 human-written samples from XSum as g^- and their rewritten versions generated by LLaMA-3-8B (Dubey et al., 2024) as g^+. For the personalized domain, we take 150 texts from the Jane Austen subset in `Stylo-Literary` (Gao et al., 2026) as s^- and use their continuations by LLaMA-3-8B as s^+. We obtain V_{dom} as described in §3.2 and use the extraction procedure in §4.1 to obtain V_{inv} and V_{con}.

D.2 DATASET DETAILS

Table 4 summarizes the three datasets used. We use one dataset per personalization level: M4 for the General setting, the Blog Authorship Corpus for the Prompted setting, and the Gutenberg Book Corpus for the Fine-tuned setting.

General setting. We use the English subset of M4. It has five subdomains, namely arXiv, PeerRead, Reddit, WikiHow, and Wikipedia, and four generators, namely ChatGPT, Cohere, `text-davinci-003`, and BLOOMZ. The five subdomains cover diverse daily-language sources, and each generator uses 2–8 distinct prompts to produce varied MGTs. We keep only the generators that have data in all five subdomains in the released version of M4. Table 3 reports the per-(generator, subdomain) counts.

Prompted setting. We build personalized text by prompting. Human texts come from the Blog Authorship Corpus (Schler et al., 2006), which groups posts by author. We feed the model a few short examples from the same author so it can imitate the author's habits. We use four large instruction-following models, namely GPT-4o, Claude-4-Sonnet, Claude-3.7-Sonnet-Latest, and Qwen2.5-72B. The generated texts mimic everyday writing without any training. Each (generator) group contains 1,000 paired HWT/MGT samples.

Fine-tuned setting. We build personalized text by continued training. Human texts come from the Gutenberg Book Corpus (Gerlach & Font-Clos, 2020), which groups literary works by author. We continue training small base LLMs on each author's corpus so the model picks up long-range tone and narrative patterns. We use Qwen3-4B, Llama-3.1-8B, and Phi-4 because they train fast and still capture author-level signals. The generated texts follow each author's voice. Each (generator, author) group contains 1,000 paired HWT/MGT samples.

D.3 Detector Details

D.3.1 Training-free Detectors

We follow (Xu et al., 2025) and include seven training-free detectors. They all read static signals from a fixed proxy model. We use GPT-J-6B (Wang & Komatsuzaki, 2021) as the proxy. We never touch the generator LLM's parameters, gradients, or token distribution, so the setup is fully black-box.

Log-Likelihood (LL) (Solaiman et al., 2019). LL takes the average log probability of the tokens. Lower likelihood points to machine origin, since smooth and predictable text is rare in human writing.

LogRank (LR) (Solaiman et al., 2019). LR takes the average log of each token's rank under the model. Higher ranks point to human writing, lower ranks to machine writing.

Entropy (Ent) (Gehrmann et al., 2019; Ippolito et al., 2020). Ent uses the average entropy of the next-token distribution. Human texts tend to give higher entropy. Machine texts give lower entropy because of overconfident predictions.

DetectLRR (LRR) (Su et al., 2023a). LRR takes the ratio of log-likelihood to log-rank. The ratio balances overall probability and local ranking, which makes the score steadier.

Fast-DetectGPT (SD) (Bao et al., 2024). SD applies small perturbations and compares log-probability curvature before and after. Larger differences point to machine origin.

Lastde (Ld) (Xu et al., 2025). Ld looks at the diversity of token probability sequences. It mixes local and global diversity with likelihood, since uniform sequences usually signal machine generation.

Lastde++ (Ld++) (Xu et al., 2025). Ld++ improves Ld by normalizing the diversity-likelihood ratio with multiple contrast samples. The normalization makes the score more stable under noise.

DNA-DetectLLM (Zhu et al., 2025). DNA-DetectLLM iteratively repairs non-optimal tokens toward an ideal AI-generated sequence. It uses the cumulative repair effort as the detection signal, since human and machine texts require different levels of repair.

SpecDetect++ (Luo et al., 2026). SpecDetect++ analyzes the spectral energy of token log-probability sequences and further incorporates sampling discrepancy. Human texts tend to show stronger probability fluctuations than machine-generated texts.

We follow the original hyperparameters in (Xu et al., 2025) and released implementations.

D.3.2 Training-based Detectors

We also include two widely used training-based detectors.

Dataset	M4	Stylo-Blog	Stylo-Literary
Setting	General	Prompted	Fine-tuned
Subdomains	5	1	7
Generators	4	4	3
Method	N/A	Prompt	CPT
Examples	54,259	4,000	21,000
Length	—	$\leq$512	$\leq$512

Table 4: Datasets used across settings. Rows show detailed attributes for each dataset. We adopt the released statistics and construction recipes.

RoBERTa (Liu et al., 2019). A fine-tuned RoBERTa encoder trained as a binary classifier. It picks up general lexical and stylistic cues from labeled data and is strong on large benchmarks.

DeTeCtive (DeT) (Guo et al., 2024). A contrastive detector that learns multi-level source representations. It aligns stylistic consistency across hierarchical source relations to improve robustness.

For both detectors, we use their public pretrained checkpoints[2], which work well in their original papers.

D.4 Full In-distribution Experimental Results

We report the per-subset ΔAUROC after PersFix for every (generator, scenario, detector) triple. Each subset selects α on its own development set and tests on the same subset. Cell color encodes the size of the improvement.

In the column headers, **LL** stands for Log-Likelihood, **LR** for LogRank, **Ent** for Entropy, **LRR** for DetectLRR, **SD** for Fast-DetectGPT, **Ld** for Lastde, **Ld++** for Lastde++, and **DeT** for DeTeCtive. Three tables follow, one per setting: Fine-tuned (Table 5), Prompted (Table 6), and General (Table 7).

Table 5: Per-subset ΔAUROC and ΔTPR@5FPR after PersFix in the Fine-tuned setting, selected and evaluated within the same subset. The table is split into an upper block (AUROC) and a lower block (TPR@5FPR), each separated by a gray section header. Cells are colored on a diverging scale: orange marks positive gain, green marks negative deviation.

Author	Model	RoBERTa	DeT	Ent	LL	LR	LRR	SD	Ld	Ld++
ΔAUROC										
Austen	**Llama**	+0.369	+0.169	+0.151	+0.771	+0.769	+0.703	+0.647	+0.513	+0.631
	Phi-4	+0.273	+0.174	+0.324	+0.716	+0.700	+0.551	+0.636	+0.463	+0.620
	Qwen3	+0.283	+0.071	+0.167	+0.798	+0.798	+0.708	+0.650	+0.502	+0.655
Dickens	**Llama**	+0.249	+0.022	+0.215	+0.477	+0.442	+0.320	+0.490	+0.111	+0.217
	Phi-4	+0.370	+0.179	+0.226	+0.533	+0.503	+0.374	+0.596	+0.164	+0.362
	Qwen3	+0.250	+0.070	+0.049	+0.726	+0.719	+0.627	+0.681	+0.205	+0.470
Dosto.	**Llama**	+0.284	+0.240	+0.432	+0.363	+0.348	+0.311	+0.437	+0.074	+0.143
	Phi-4	+0.144	+0.208	+0.413	+0.437	+0.429	+0.393	+0.613	+0.120	+0.256
	Qwen3	+0.428	+0.040	+0.189	+0.765	+0.762	+0.692	+0.775	+0.179	+0.416
Plato	**Llama**	+0.340	+0.268	+0.302	+0.540	+0.521	+0.445	+0.543	+0.167	+0.291
	Phi-4	+0.333	+0.324	+0.366	+0.548	+0.538	+0.486	+0.695	+0.210	+0.431
	Qwen3	+0.351	+0.168	+0.148	+0.768	+0.761	+0.680	+0.746	+0.191	+0.432
Shaw	**Llama**	+0.300	+0.102	+0.304	+0.383	+0.359	+0.263	+0.372	+0.060	+0.111
	Phi-4	+0.512	+0.142	+0.296	+0.490	+0.476	+0.354	+0.567	+0.104	+0.222
	Qwen3	+0.442	+0.176	+0.101	+0.718	+0.704	+0.598	+0.657	+0.117	+0.278
Swift	**Llama**	+0.320	+0.167	+0.268	+0.342	+0.326	+0.269	+0.397	+0.108	+0.139
	Phi-4	+0.351	+0.112	+0.259	+0.398	+0.387	+0.343	+0.551	+0.147	+0.267
	Qwen3	+0.254	+0.151	+0.117	+0.610	+0.598	+0.521	+0.675	+0.197	+0.366
Twain	**Llama**	+0.363	+0.325	+0.353	+0.390	+0.367	+0.297	+0.433	+0.082	+0.139
	Phi-4	+0.424	+0.213	+0.325	+0.486	+0.467	+0.389	+0.592	+0.106	+0.247
	Qwen3	+0.407	+0.256	+0.091	+0.760	+0.740	+0.647	+0.705	+0.139	+0.322
ΔTPR@5FPR										

[2]For RoBERTa we use the GPT-2 Output Detector. For DeTeCtive we use the M4-monolingual checkpoint.

Author	Model	RoBERTa	DeT	Ent	LL	LR	LRR	SD	Ld	Ld++
Austen	Llama	+0.358	+0.112	+0.345	+0.341	+0.359	+0.386	+0.180	+0.121	+0.205
	Phi-4	+0.300	+0.160	+0.375	+0.331	+0.361	+0.372	+0.152	+0.237	+0.216
	Qwen3	+0.440	+0.100	+0.480	+0.333	+0.359	+0.383	+0.129	+0.250	+0.223
Dickens	Llama	+0.228	+0.000	+0.290	+0.230	+0.189	+0.165	+0.207	+0.216	+0.175
	Phi-4	+0.220	+0.080	+0.293	+0.227	+0.219	+0.171	+0.179	+0.194	+0.215
	Qwen3	+0.140	-0.080	+0.239	+0.225	+0.260	+0.275	+0.160	+0.184	+0.234
Dosto.	Llama	+0.520	+0.580	+0.627	+0.631	+0.626	+0.621	+0.615	+0.296	+0.420
	Phi-4	+0.540	+0.460	+0.602	+0.609	+0.610	+0.613	+0.596	+0.376	+0.424
	Qwen3	+0.620	+0.340	+0.489	+0.635	+0.644	+0.651	+0.600	+0.499	+0.608
Plato	Llama	+0.542	+0.240	+0.545	+0.556	+0.560	+0.574	+0.556	+0.481	+0.445
	Phi-4	+0.480	+0.440	+0.540	+0.556	+0.569	+0.583	+0.534	+0.534	+0.540
	Qwen3	+0.380	+0.300	+0.374	+0.556	+0.562	+0.568	+0.489	+0.512	+0.540
Shaw	Llama	+0.387	+0.160	+0.357	+0.405	+0.418	+0.315	+0.360	+0.221	+0.257
	Phi-4	+0.600	+0.420	+0.401	+0.439	+0.449	+0.277	+0.399	+0.289	+0.315
	Qwen3	+0.540	+0.100	+0.324	+0.434	+0.446	+0.451	+0.383	+0.304	+0.346
Swift	Llama	+0.126	+0.320	+0.105	+0.114	+0.115	+0.109	+0.119	+0.179	+0.145
	Phi-4	+0.140	+0.100	+0.103	+0.113	+0.116	+0.135	+0.142	+0.151	+0.147
	Qwen3	+0.100	+0.240	+0.124	+0.121	+0.130	+0.145	+0.131	+0.191	+0.175
Twain	Llama	+0.474	+0.460	+0.487	+0.475	+0.478	+0.480	+0.469	+0.376	+0.315
	Phi-4	+0.420	+0.360	+0.446	+0.459	+0.466	+0.486	+0.426	+0.486	+0.426
	Qwen3	+0.460	+0.260	+0.311	+0.488	+0.486	+0.494	+0.432	+0.509	+0.471

Table 6: Per-group ΔAUROC and ΔTPR@5FPR after PERSFIX in the General setting, selected and evaluated within the same group. The table reports AUROC and TPR@5FPR separately. Cells are colored on a diverging scale: orange marks positive gain, green marks negative deviation.

Domain	Model	RoBERTa	DeT	Ent	LL	LR	LRR	SD	Ld	Ld++
					ΔAUROC					
Blogs	Qwen2.5	+0.102	+0.040	+0.502	+0.002	+0.002	+0.000	+0.000	+0.006	+0.000
	Claude-3.7	+0.025	+0.235	+0.365	+0.022	+0.030	+0.066	+0.002	+0.163	+0.005
	Claude-4	+0.205	+0.242	+0.320	+0.048	+0.056	+0.076	+0.027	+0.062	+0.026
	GPT-4o	+0.374	+0.303	+0.167	+0.381	+0.448	+0.593	+0.232	+0.435	+0.221
					ΔTPR@5FPR					
Blogs	Qwen2.5	+0.091	+0.060	+0.126	+0.029	+0.016	+0.000	+0.000	+0.030	+0.001
	Claude-3.7	+0.014	+0.077	+0.109	+0.077	+0.047	+0.073	+0.012	+0.105	-0.054
	Claude-4	+0.117	+0.121	+0.128	+0.109	+0.098	+0.043	+0.099	+0.138	+0.068
	GPT-4o	+0.158	+0.103	+0.219	+0.186	+0.182	+0.180	+0.104	+0.048	+0.089

Table 7: Per-subset ΔAUROC and ΔTPR@5FPR after PERSFIX in the General setting, selected and evaluated within the same subset. The table is split into an upper block (AUROC) and a lower block (TPR@5FPR), each separated by a gray section header. Cells are colored on a diverging scale: orange marks positive gain, green marks negative deviation.

Subdomain	Model	RoBERTa	DeT	Ent	LL	LR	LRR	SD	Ld	Ld++
					ΔAUROC					
arXiv	BLOOMZ	-0.005	+0.051	+0.545	+0.073	+0.033	+0.005	+0.022	+0.004	+0.006
	ChatGPT	+0.643	+0.096	-0.016	-0.010	-0.001	-0.000	+0.000	-0.000	+0.000
	Cohere	+0.322	+0.129	+0.224	+0.073	+0.053	+0.044	+0.001	+0.016	+0.004
	Davinci	+0.176	+0.482	+0.168	+0.709	+0.728	+0.735	+0.487	+0.218	+0.508
PeerRead	BLOOMZ	+0.176	+0.000	+0.507	+0.112	+0.044	+0.000	+0.271	+0.004	+0.042
	ChatGPT	+0.345	+0.000	+0.267	+0.001	+0.000	+0.000	-0.000	-0.000	+0.000
	Cohere	+0.000	-0.019	+0.477	+0.002	+0.001	+0.000	+0.001	-0.000	+0.001
	Davinci	+0.169	+0.002	+0.474	+0.001	+0.000	+0.000	-0.000	+0.000	-0.000
Reddit	BLOOMZ	+0.162	+0.072	+0.481	+0.134	+0.063	+0.000	+0.400	+0.008	+0.048
	ChatGPT	+0.527	+0.132	+0.752	+0.002	+0.002	+0.002	+0.000	+0.001	+0.001
	Cohere	+0.406	+0.377	+0.597	+0.011	+0.012	+0.018	+0.002	+0.001	+0.005
	Davinci	+0.315	+0.157	+0.665	-0.004	+0.010	+0.000	+0.002	+0.004	+0.003
WikiHow	BLOOMZ	+0.065	+0.194	+0.129	+0.056	+0.041	+0.001	+0.273	+0.017	+0.031
	ChatGPT	-0.124	+0.245	+0.408	+0.000	+0.000	+0.000	-0.001	-0.004	-0.001
	Cohere	+0.676	+0.086	+0.711	+0.000	+0.000	+0.000	+0.000	+0.000	+0.000

Subdomain	Model	RoBERTa	DeT	Ent	LL	LR	LRR	SD	Ld	Ld++
	Davinci	-0.020	+0.229	+0.246	-0.004	+0.000	+0.000	-0.008	+0.000	-0.001
Wikipedia	**BLOOMZ**	+0.512	+0.046	-0.001	+0.455	+0.368	+0.041	+0.016	+0.014	-0.012
	ChatGPT	+0.410	+0.429	+0.435	+0.096	+0.074	+0.027	+0.005	+0.012	+0.005
	Cohere	+0.438	+0.510	+0.392	+0.022	+0.025	+0.024	+0.003	+0.005	+0.002
	Davinci	+0.235	+0.390	+0.144	+0.180	+0.154	+0.081	+0.016	+0.017	+0.014
					△TPR@5FPR					
arXiv	**BLOOMZ**	+0.193	+0.167	+0.735	+0.478	+0.228	+0.027	+0.113	+0.010	+0.029
	ChatGPT	+0.827	+0.290	+0.034	-0.064	-0.008	+0.005	+0.000	-0.001	+0.000
	Cohere	+0.207	+0.183	+0.445	+0.319	+0.257	+0.191	+0.005	+0.072	+0.022
	Davinci	+0.167	+0.190	+0.510	+0.702	+0.719	+0.743	+0.667	+0.182	+0.632
PeerRead	**BLOOMZ**	+0.093	+0.000	+0.308	+0.186	+0.043	+0.000	+0.270	+0.016	+0.109
	ChatGPT	+0.764	+0.000	+0.003	+0.001	-0.001	+0.000	-0.001	+0.001	+0.000
	Cohere	+0.000	+0.093	+0.003	+0.001	-0.013	+0.000	+0.001	-0.005	-0.001
	Davinci	+0.358	+0.000	+0.006	+0.008	+0.000	+0.000	-0.002	-0.002	-0.001
Reddit	**BLOOMZ**	+0.150	+0.347	+0.234	+0.194	+0.228	+0.000	+0.194	+0.047	+0.199
	ChatGPT	+0.400	+0.410	+0.275	+0.006	+0.006	+0.011	+0.003	+0.003	+0.005
	Cohere	+0.400	+0.497	+0.325	+0.011	+0.006	+0.009	+0.012	+0.000	+0.027
	Davinci	+0.233	+0.303	+0.271	+0.055	+0.045	+0.047	+0.017	+0.027	+0.020
WikiHow	**BLOOMZ**	+0.020	+0.020	+0.033	+0.025	+0.020	+0.023	+0.042	+0.021	-0.000
	ChatGPT	-0.060	+0.107	+0.037	+0.000	+0.000	+0.000	-0.003	-0.007	-0.004
	Cohere	+0.457	+0.373	+0.176	+0.000	+0.000	+0.000	+0.000	+0.000	-0.000
	Davinci	-0.030	+0.057	+0.032	-0.008	+0.000	+0.000	-0.019	+0.000	-0.023
Wikipedia	**BLOOMZ**	+0.537	+0.197	+0.245	+0.368	+0.383	+0.394	+0.337	+0.190	+0.071
	ChatGPT	+0.796	+0.732	+0.164	+0.541	+0.587	+0.153	+0.011	+0.042	+0.014
	Cohere	+0.783	+0.762	+0.111	+0.309	+0.331	+0.112	+0.011	+0.014	+0.002
	Davinci	+0.473	+0.353	+0.146	+0.126	+0.172	+0.235	+0.047	+0.071	+0.039

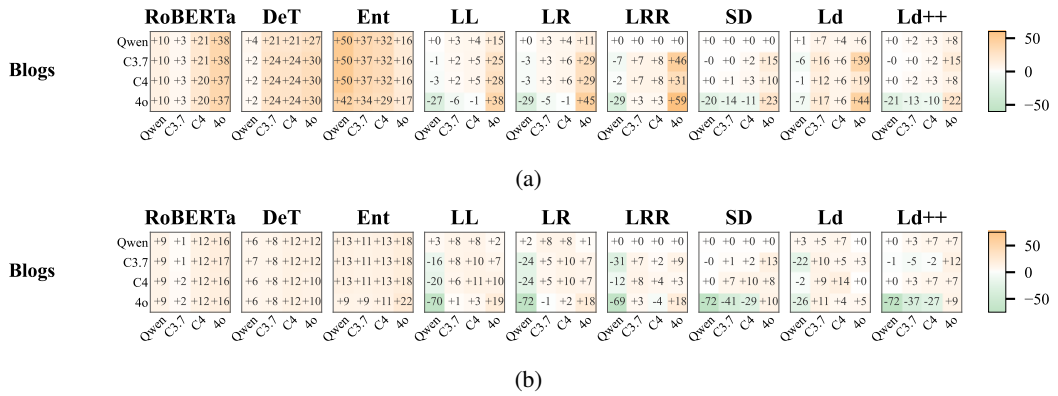
D.5 Additional Transfer Results

This section gives the full out-of-distribution transfer heatmaps for all nine detectors on both metrics. We split the heatmaps into two parts. *Cross-model transfer* fixes the scenario and varies the source/target generator LLM. *Cross-scenario transfer* fixes the generator and varies the source/target scenario. Within each part, figures group by dataset, then by the fixed scenario or generator, then by metric.

Each figure has a 3×3 grid of detector panels arranged by family. The top row gives the training-based detectors plus the entropy reference. The middle row gives likelihood-based scoring detectors. The bottom row gives advanced zero-shot detectors. The detector abbreviations follow §D.3: **LL** (Log-Likelihood), **LR** (LogRank), **Ent** (Entropy), **LRR** (DetectLRR), **SD** (Fast-DetectGPT), **Ld** (Lastde), **Ld++** (Lastde++), and **DeT** (DeTeCtive). For Stylo-Literary, **Llama-3.1** stands for Llama-3.1-8B. For Stylo-Blog, **Qwen2.5** stands for Qwen2.5-72B, and **Claude-3.7/Claude-4** stand for Claude-3.7-Sonnet/Claude-4-Sonnet.

Cross-model transfer. We first show cross-model transfer, where the scenario is fixed and the source/target generator LLM varies (Figures 6a–8b).

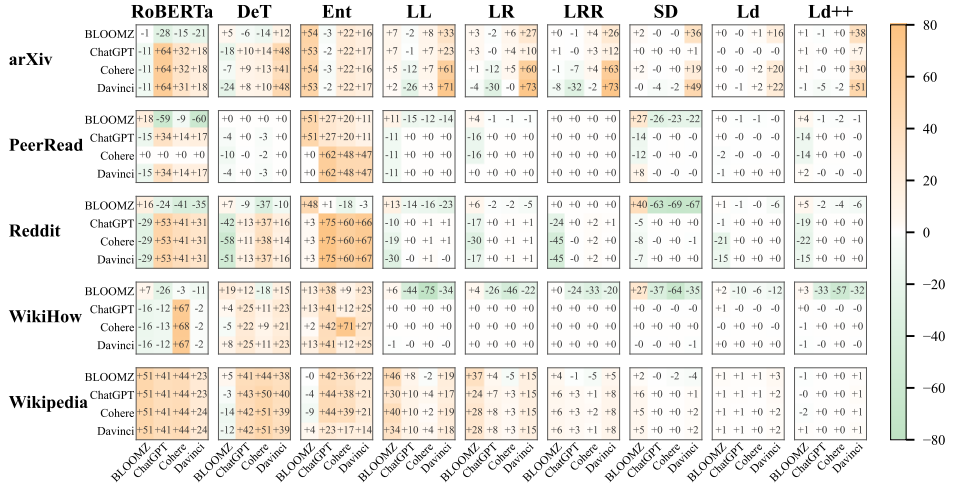
Cross-scenario transfer. We then show cross-scenario transfer, where the generator is fixed and the source/target scenario varies (Figures 9a–12b).



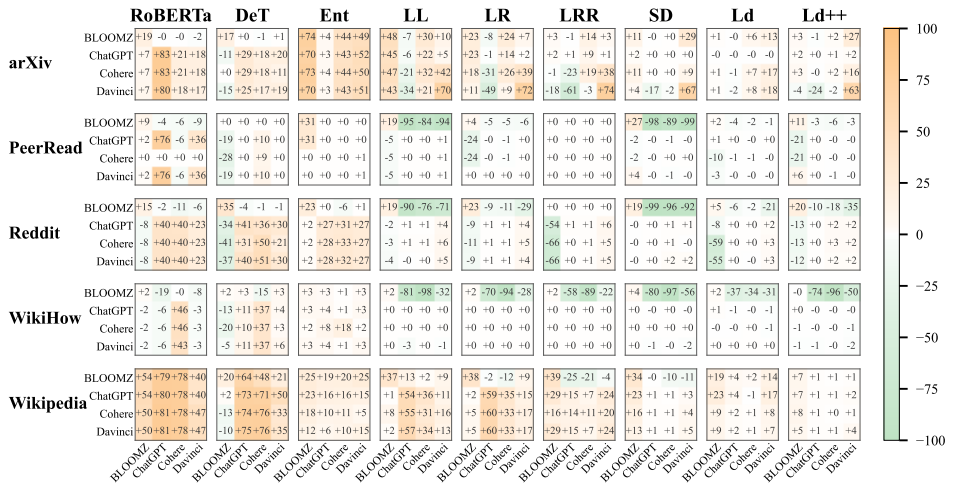
(a)

(b)

Figure 6: Stylo-Blog: cross-model transfer for all nine detectors, showing (a) ΔAUROC and (b) ΔTPR@5FPR. Each row is one fixed scenario; each column is a detector. Inside each panel rows index source models and columns index target models. Cell annotations are Δ in percentage points.

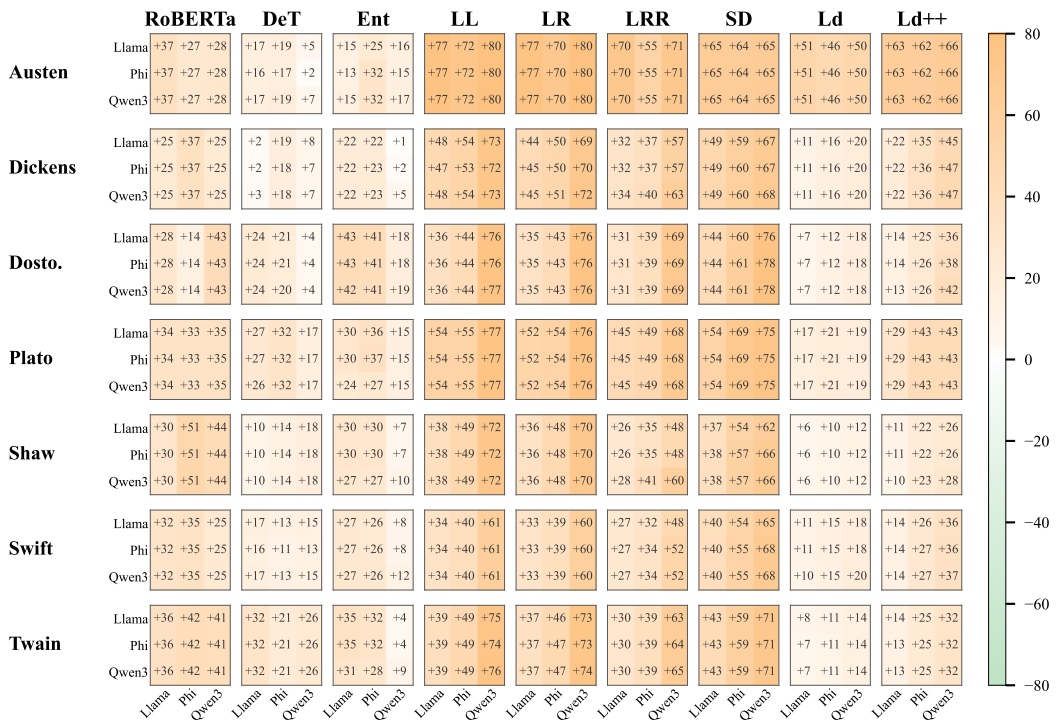


(a)

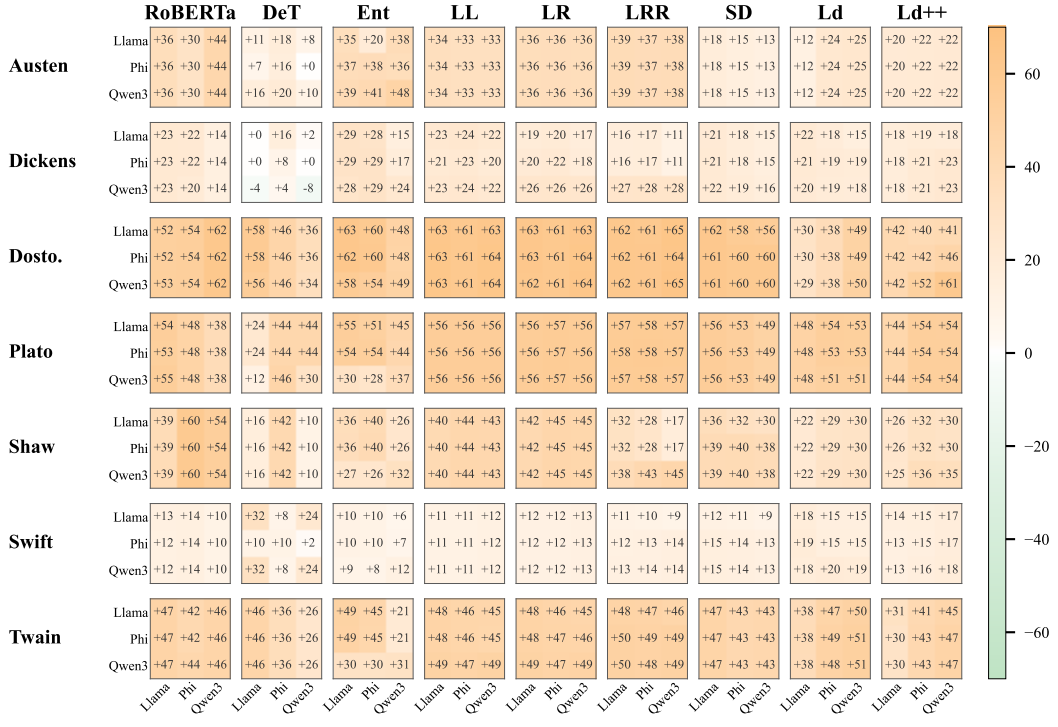


(b)

Figure 7: M4: cross-model transfer for all nine detectors, showing (a) ΔAUROC and (b) ΔTPR@5FPR. Each row is one fixed scenario; each column is a detector. Inside each panel rows index source models and columns index target models. Cell annotations are Δ in percentage points.

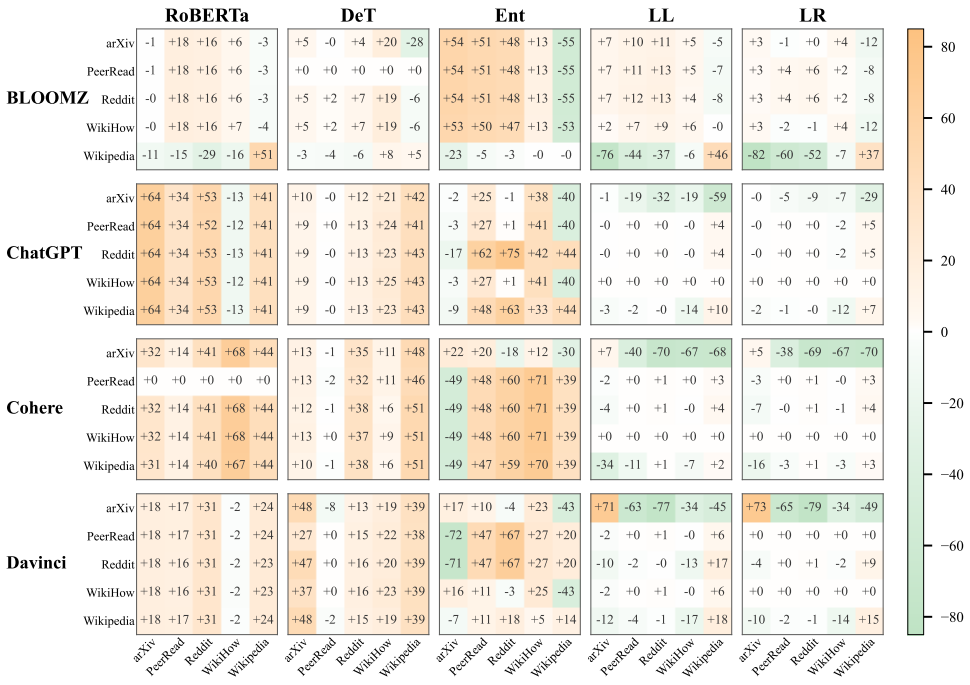


(a)

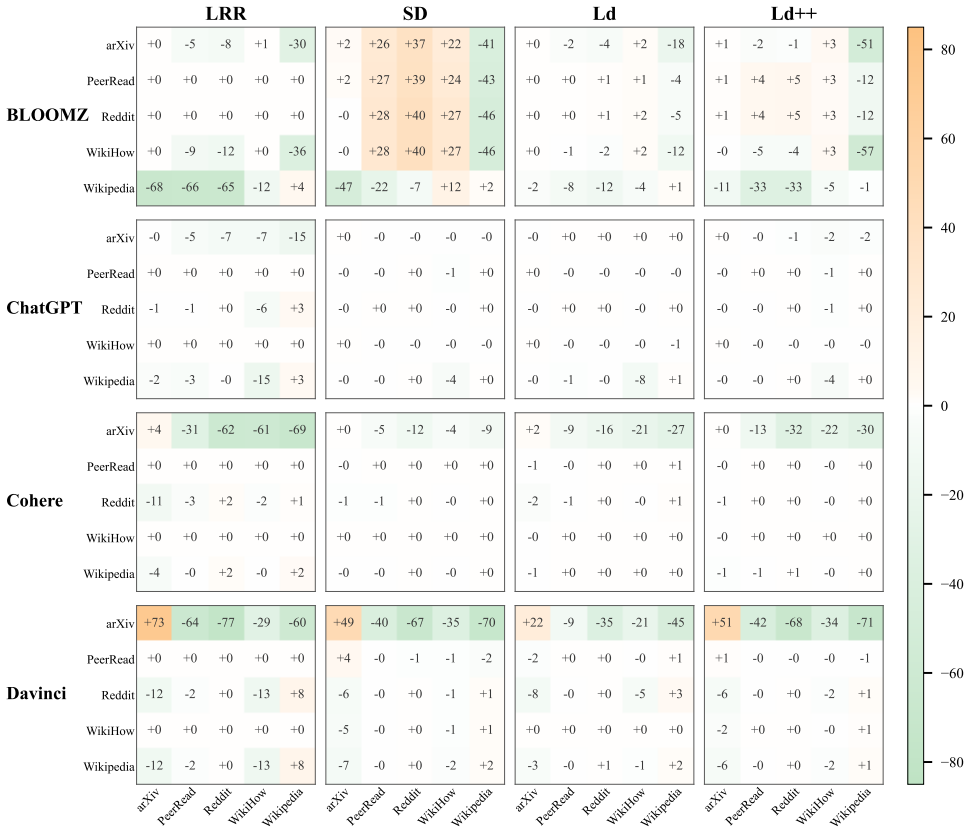


(b)

Figure 8: Stylo-Literary: cross-model transfer for all nine detectors, showing (a) ΔAUROC and (b) ΔTPR@5FPR. Each row is one fixed scenario; each column is a detector. Inside each panel rows index source models and columns index target models. Cell annotations are Δ in percentage points.

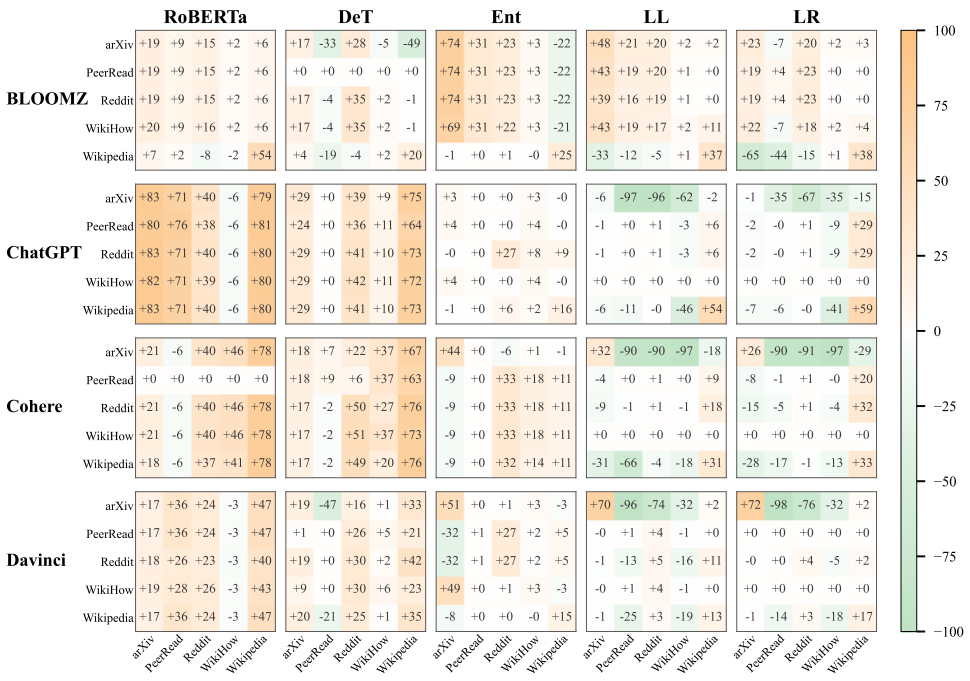


(a)

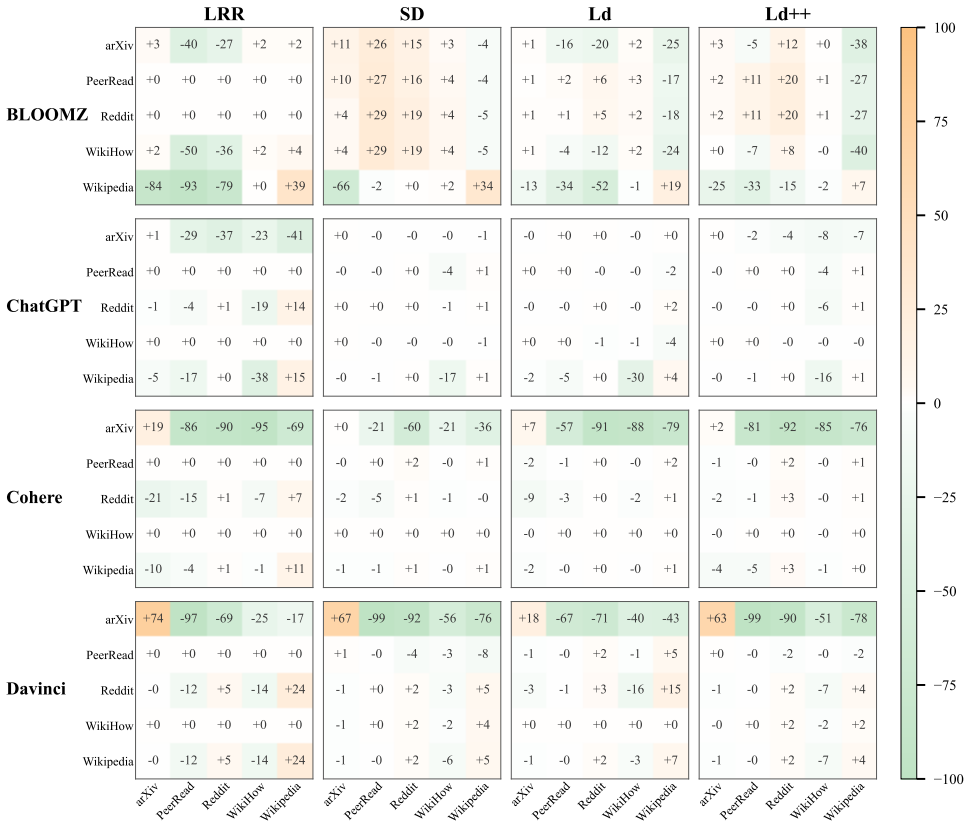


(b)

Figure 9: M4: cross-scenario transfer ΔAUROC for all nine detectors. Each row is one fixed generator; each column is a detector. Inside each panel rows index source scenarios and columns index target scenarios. Cell annotations are Δ in percentage points.

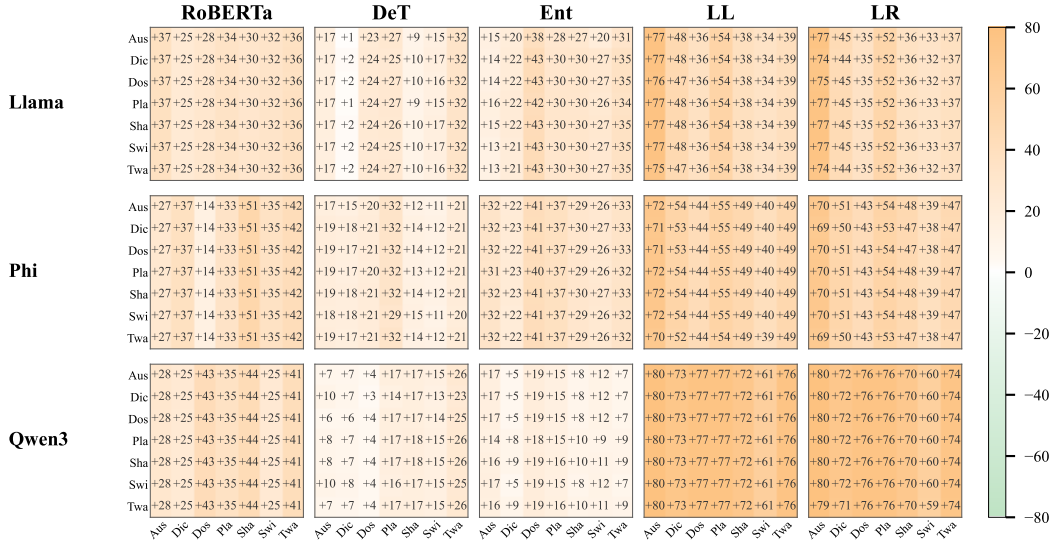


(a)

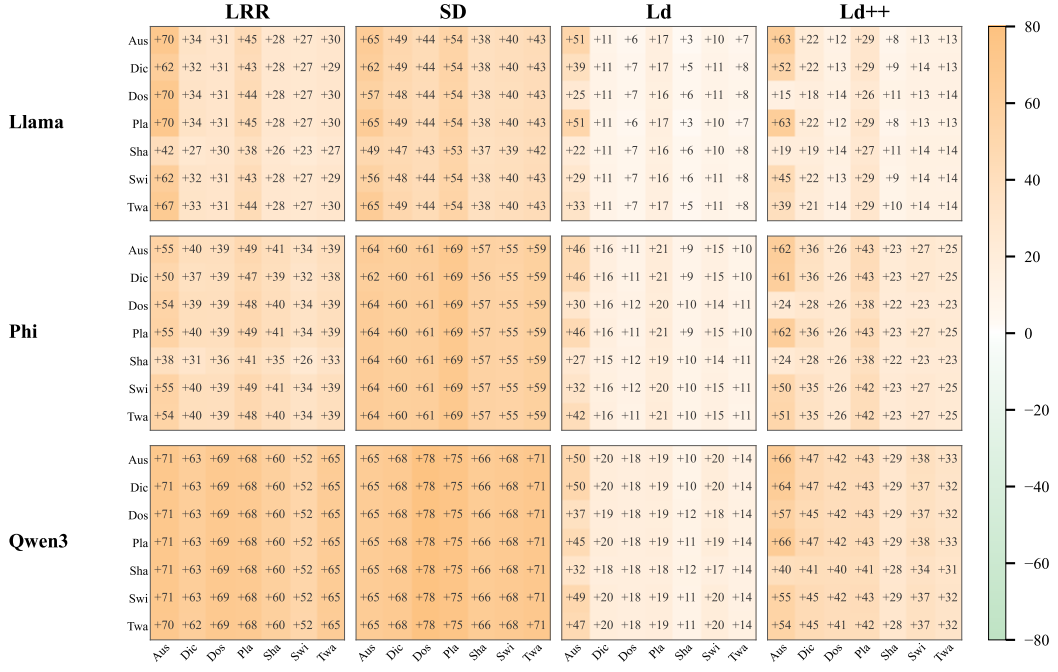


(b)

Figure 10: M4: cross-scenario transfer ΔTPR@5FPR for all nine detectors. Each row is one fixed generator; each column is a detector. Inside each panel rows index source scenarios and columns index target scenarios. Cell annotations are Δ in percentage points.

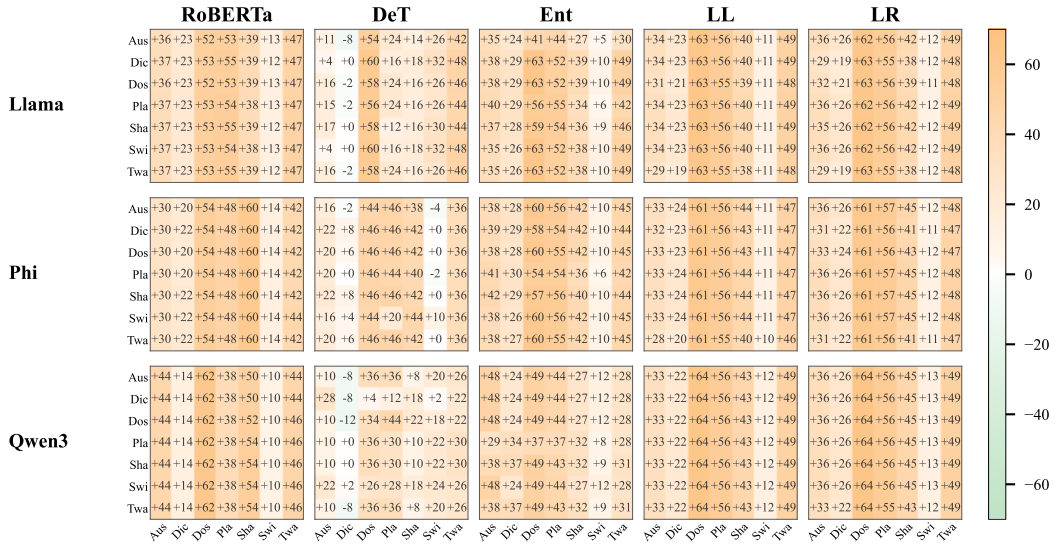


(a)

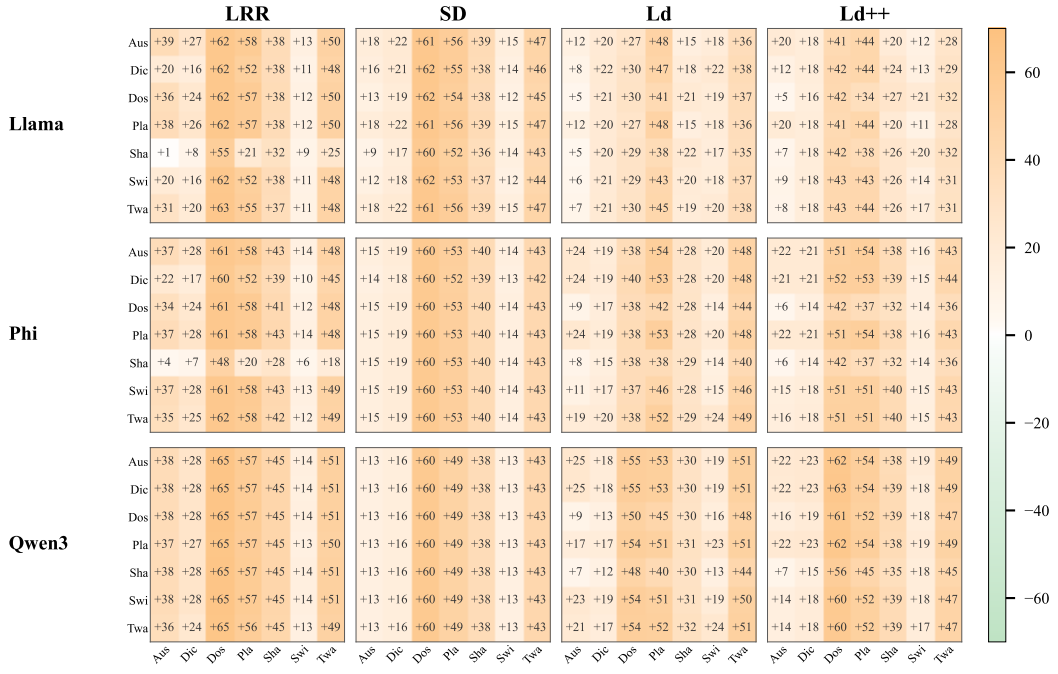


(b)

Figure 11: Stylo-Literary: cross-scenario transfer ΔAUROC for all nine detectors. Each row is one fixed generator; each column is a detector. Inside each panel rows index source scenarios and columns index target scenarios. Cell annotations are Δ in percentage points.



(a)



(b)

Figure 12: Stylo-Literary: cross-scenario transfer ΔTPR@5FPR for all nine detectors. Each row is one fixed generator; each column is a detector. Inside each panel rows index source scenarios and columns index target scenarios. Cell annotations are Δ in percentage points.

D.6 Compute Resources

We run all experiments on two NVIDIA RTX A6000 GPUs. For each group, activation extraction takes about 1,000 $\sim$ 3,000 GPT-2 forward passes ($\sim$30 seconds), and each (group, detector) adjustment takes about 1 minute. The full set of experiments takes about 2 hours of wall-clock time.

E Ablation Studies

Setup: an upper-bound view. The ablations in this section study the best-case behavior of each design choice. For every panel we sweep α on the test set and pick the value that maximizes performance. We do not select α on a development set. This gives the upper bound of what each feature direction can deliver if α is oracle-tuned. We use this view here because it isolates the role of V_{con} from any error introduced by α selection. The full system in the main text always selects α on a held-out subset, so the absolute numbers in this section are higher than those in the main results.

We ablate the feature direction that drives the adjustment. Figure 13 replaces V_{con} with three reference directions of the same form, namely a domain feature, an MGT feature (Gao et al., 2026), and a random control. The construction details are in Appendix E.1. The consistent feature is the strongest in every setting. The random feature stays the lowest. The MGT feature gives moderate improvement with a wider spread. The domain feature trails with the largest variance. The same rankings hold on additional scenario draws in Appendix E.2.

E.1 Implementation Details

This subsection describes the data we use to build feature directions and the data we use to test each direction.

Experimental settings. We work with three settings. For each setting we draw one subdomain as the test data. We then build 100 groups of probing data. Each group pairs one feature direction with one set of test samples. We run 100 experiments per direction and report the distribution of the upper-bound improvements.

Test data. Detectors start from very different baselines. To avoid unstable or biased numbers, we pick one representative experiment per setting. The chosen experiment matches the average difficulty of the detectors. Table 8 lists the chosen setting, model, subdomain, detector, and baseline AUROC.

Setting	Fine-tuned	Prompted	General
Model	Qwen-3-4B	Claude-3.7	Davinci
Subdomain	Austen Jane	Blogs	WikiHow
Detector	Lastde++	DetectLRR	Log-Likelihood
AUROC	9.88	66.99	84.26

Table 8: Selected experiments used in the ablation study. Each row shows one setting and its representative model, subdomain, and detector.

Probing data. Each experiment draws from two sources. We take 150 matched HWT/MGT samples from Stylo-Literary for the personalized side and 150 matched HWT/MGT samples from M4 for the General side.

Domain feature. We collect activations from both sides. We label HWT from the personalized side as positive and HWT from the General side as negative, then train a simple linear classifier. The classifier weight gives the domain feature. The classifier learns to separate the two sides by domain rather than by class, so the direction shows how the data shift between domains.

MGT feature. We collect activations from all settings. We label MGT samples as positive and HWT samples as negative, then train a linear classifier. The weight gives the MGT feature. This direction reflects the general HWT/MGT split, but it is not stable across settings because the differences vary by dataset. Prior work shows that such direct training often gives weak and unstable features.

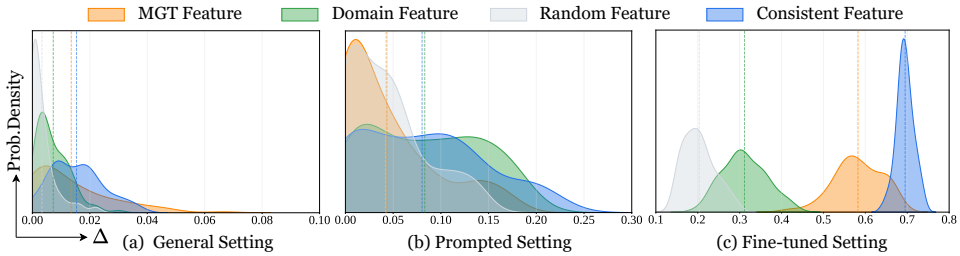


Figure 13: Ablation on the feature type under the upper-bound setup. Distribution of upper-bound improvements across four feature types in the General, Prompted, and Fine-tuned settings; the consistent feature dominates.

Subdomain		BLOOMZ	ChatGPT	Cohere	Davinci	Avg.
arXiv	w/o PERSFIX	**87.1**	**90.7**	**88.9**	39.7	76.6
	Δ	2.7	4.0	3.1	**25.3**	8.8
PeerRead	w/o PERSFIX	72.3	85.9	78.3	**88.1**	**81.1**
	Δ	14.5	6.6	5.1	5.6	7.9
Reddit	w/o PERSFIX	67.6	86.6	79.8	84.7	79.7
	Δ	**21.7**	**9.4**	**15.9**	7.2	**13.5**
WikiHow	w/o PERSFIX	47.8	82.4	86.3	76.1	73.1
	Δ	3.5	5.1	9.8	4.0	5.6
Wikipedia	w/o PERSFIX	67.3	87.5	86.8	78.3	80.0
	Δ	13.7	7.9	4.9	3.5	7.5

Table 9: Performance in the General setting. Each subdomain reports baseline AUROC (w/o PERSFIX) and improvement after adjustment (Δ), averaged over detectors. Bold marks the maximum and underline the minimum per generator.

Random feature. The random feature is a control. It keeps the same form as the other directions but carries no semantic meaning. We draw a fresh random direction for each experiment.

E.2 CROSS-SCENARIO STABILITY OF THE ABLATION CONCLUSIONS

The ablation above fixes one representative scenario per setting. To check that the feature-type choice stays stable on other scenarios, we re-run the same analysis on additional scenario draws within each setting. We use two more authors for the Fine-tuned setting and two more subdomains for the General setting. We hold the detector and the generator fixed within each draw. The ranking of feature types is preserved across draws: the consistent feature dominates in every draw, the MGT and domain features trail with wider variance, and the random control stays near zero. The size of the gap shrinks slightly when the detector's baseline AUROC is close to ceiling, which fits the headroom argument in the main text, but the ordering does not change. We therefore conclude that the choice in the main ablation is not specific to the scenario we used for visualization.

E.3 PER-SUBDOMAIN ADJUSTMENT IMPROVEMENT ON M4

Table 9 reports baseline AUROC (w/o PERSFIX) and the adjustment improvement (Δ) on each of the five `M4` subdomains, averaged over detectors. The improvement varies across subdomains: `Reddit` and `arXiv` receive the largest average improvements, while `WikiHow` and `Wikipedia` receive the smallest.

Stylization within general domains. PERSFIX is designed for personalized data, but it also improves results in the General setting. Table 9 offers a possible explanation. `Reddit` shows the largest gain, while `WikiHow` shows the smallest. Considering that Reddit often contains highly opinionated and stylistically marked content, whereas WikiHow is heavily style-neutral, this pattern suggests that some parts of the General data are more stylized and closer to personalized text. PERSFIX applies stronger adjustment to these parts, so the improvement observed in the General

setting is likely driven by this relatively more personalized subset rather than by the style-neutral majority. This suggests that personalized variation exists even within so-called "general" domains, and that modeling personalization effects can further enhance detector robustness.

F Further Illustration of Mathematical Design

F.1 Notation

We assume a sample space of dimension d containing N samples, where each sample is a vector $\mathbf{s}_i \in \mathbb{R}^d$. We organize these samples into a matrix $R = [\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_N] \in \mathbb{R}^{d\times N}$, with the row average denoted by the vector $\bar{R} \in \mathbb{R}^d$. Following the definitions in the main text, we denote a feature vector as $V \in \mathbb{R}^d$ and the specific domain feature as V_{dom}. We define the default orthogonal basis $E = (e^{(0)}, \ldots, e^{(d-1)})$, where the component of a vector is the coordinate projection $V^{(E,i)} = V[i]e^{(i)}$. Furthermore, we define an arbitrary orthogonal basis $U = (u^{(0)}, \ldots, u^{(d-1)})$, where the component takes the general form $V^{(U,i)} = (V^\top u^{(i)})u^{(i)}$. For simplicity, we assume all vectors mentioned in this section are unit vectors and denote the covariance matrix as Σ.

F.2 Derivation of Variance Expressions

Variance under a vector. Here, we calculate the variance of high-dimensional data projected onto a feature vector. First, we center the matrix R to obtain $\tilde{R} = R - \mathbf{1}\bar{R}^\top \in \mathbb{R}^{d\times N}$ (broadcasting the mean vector). The k-th column of this centered matrix is $\tilde{\mathbf{s}}_k = \mathbf{s}_k - \bar{R}$. According to the definition of the covariance matrix (Bishop & Nasrabadi, 2007), we have $\Sigma = \frac{1}{N-1}\tilde{R}\tilde{R}^\top \in \mathbb{R}^{d\times d}$. Specifically, $\Sigma_{i,i}$ represents the variance of samples projected onto $e^{(i)}$, while $\Sigma_{i,j}$ $(i \neq j)$ represents the covariance between projections on $e^{(i)}$ and $e^{(j)}$.

We denote the projection of sample $\mathbf{s}_k$ onto V_{dom} as $\mathbf{x}_k = (V_{\text{dom}}^\top \mathbf{s}_k)V_{\text{dom}}$. We focus on the scalar magnitude $x_k = V_{\text{dom}}^\top \mathbf{s}_k$. Since $\bar{R} = \frac{1}{N}\sum_{k=0}^{N-1}\mathbf{s}_k$, the mean of the projected values is:

$$\bar{x} = \frac{1}{N}\sum_{k=0}^{N-1} x_k = \frac{1}{N}\sum_{k=0}^{N-1} V_{\text{dom}}^\top \mathbf{s}_k = V_{\text{dom}}^\top \bar{R}. \tag{10}$$

We derive the variance of V_{dom} as follows:

$$\begin{aligned}
\text{Var}(V_{\text{dom}}) &= \frac{1}{N-1}\sum_{k=0}^{N-1}(x_k - \bar{x})^2 \\
&= \frac{1}{N-1}\sum_{k=0}^{N-1}\left(V_{\text{dom}}^\top \mathbf{s}_k - V_{\text{dom}}^\top \bar{R}\right)^2 \\
&= \frac{1}{N-1}\sum_{k=0}^{N-1}\left(V_{\text{dom}}^\top (\mathbf{s}_k - \bar{R})\right)^2 \\
&= \frac{1}{N-1}\sum_{k=0}^{N-1} V_{\text{dom}}^\top (\mathbf{s}_k - \bar{R})(\mathbf{s}_k - \bar{R})^\top V_{\text{dom}} \\
&= V_{\text{dom}}^\top \left(\frac{1}{N-1}\sum_{k=0}^{N-1}\tilde{\mathbf{s}}_k\tilde{\mathbf{s}}_k^\top\right) V_{\text{dom}} \\
&= V_{\text{dom}}^\top \left(\frac{1}{N-1}\tilde{R}\tilde{R}^\top\right) V_{\text{dom}} \\
&= V_{\text{dom}}^\top \Sigma V_{\text{dom}}.
\end{aligned} \tag{11}$$

Marginal variance under a basis. We define "marginal variance" to represent the total variance contributed by a single orthogonal component. This includes two parts: the self-variance of the projection on this component, and the covariance between this component and other components. Formally, under basis E, the marginal variance of the i-th component of V_{dom} is:

$$\text{Var}_{\text{marg}}(V_{\text{dom}}^{(E,i)}) = \underbrace{\text{Var}(V_{\text{dom}}^{(E,i)})}_{(1)} + \underbrace{\sum_{j\neq i}\text{Cov}(V_{\text{dom}}^{(E,i)}, V_{\text{dom}}^{(E,j)})}_{(2)}. \tag{12}$$

For term (1), we use the variance formula derived above:

$$\begin{aligned}\mathrm{Var}(V_{\mathrm{dom}}^{(E,i)}) &= V_{\mathrm{dom}}^{(E,i)\top}\Sigma V_{\mathrm{dom}}^{(E,i)} \\ &= (V_{\mathrm{dom}}[i]e^{(i)})^\top \Sigma (V_{\mathrm{dom}}[i]e^{(i)}) \\ &= V_{\mathrm{dom}}[i]\left(V_{\mathrm{dom}}[i]e^{(i)\top}\Sigma e^{(i)}\right) \\ &= V_{\mathrm{dom}}[i]\left(V_{\mathrm{dom}}[i]\Sigma_{i,i}\right). \end{aligned} \tag{13}$$

For term (2), we expand the covariance sum:

$$\begin{aligned}&\textstyle\sum_{j\neq i}\mathrm{Cov}(V_{\mathrm{dom}}^{(E,i)}, V_{\mathrm{dom}}^{(E,j)}) \\ &\quad= \textstyle\sum_{j\neq i}\mathrm{Cov}(V_{\mathrm{dom}}[i]e^{(i)}, V_{\mathrm{dom}}[j]e^{(j)}) \\ &\quad= \textstyle\sum_{j\neq i} V_{\mathrm{dom}}[i]V_{\mathrm{dom}}[j]\mathrm{Cov}(e^{(i)}, e^{(j)}) \\ &\quad= V_{\mathrm{dom}}[i]\left(\textstyle\sum_{j\neq i} V_{\mathrm{dom}}[j]\Sigma_{i,j}\right). \end{aligned} \tag{14}$$

Substituting these back into Eq. 12, we get:

$$\begin{aligned}\mathrm{Var}_{\mathrm{marg}}(V_{\mathrm{dom}}^{(E,i)}) &= V_{\mathrm{dom}}[i]\left(V_{\mathrm{dom}}[i]\Sigma_{i,i}\right) + V_{\mathrm{dom}}[i]\left(\textstyle\sum_{j\neq i} V_{\mathrm{dom}}[j]\Sigma_{i,j}\right) \\ &= V_{\mathrm{dom}}[i]\textstyle\sum_k V_{\mathrm{dom}}[k]\Sigma_{i,k}. \end{aligned} \tag{15}$$

We observe that $\sum_k V_{\mathrm{dom}}[k]\Sigma_{i,k}$ is equivalent to the i-th element of the vector product $\Sigma^\top V_{\mathrm{dom}}$. Given the symmetry of the covariance matrix ($\Sigma = \Sigma^\top$), we have $\sum_k V_{\mathrm{dom}}[k]\Sigma_{i,k} = (\Sigma V_{\mathrm{dom}})[i]$. Thus:

$$\mathrm{Var}_{\mathrm{marg}}(V_{\mathrm{dom}}^{(E,i)}) = V_{\mathrm{dom}}[i](\Sigma V_{\mathrm{dom}})[i]. \tag{16}$$

We verify this by summing the marginal variances:

$$\textstyle\sum_i \mathrm{Var}_{\mathrm{marg}}(V_{\mathrm{dom}}^{(E,i)}) = \sum_i V_{\mathrm{dom}}[i](\Sigma V_{\mathrm{dom}})[i] = V_{\mathrm{dom}}^\top(\Sigma V_{\mathrm{dom}}) = \mathrm{Var}(V_{\mathrm{dom}}). \tag{17}$$

This confirms that the decomposition into marginal variance is exact.

F.3 Reformulation Under an Arbitrary Basis

Vector reformulation. Under the default basis E, the i-th component is $V_{\mathrm{dom}}^{(E,i)} = V_{\mathrm{dom}}[i]e^{(i)} = (V_{\mathrm{dom}}^\top e^{(i)})e^{(i)}$. Under an arbitrary orthogonal basis U, this form generalizes to $V_{\mathrm{dom}}^{(U,i)} = (V_{\mathrm{dom}}^\top u^{(i)})u^{(i)}$. Consequently, the coordinates of V_{dom} in the new basis are:

$$V_{\mathrm{dom}}^{(U)} = [V_{\mathrm{dom}}^\top u^{(0)}, \cdots, V_{\mathrm{dom}}^\top u^{(d-1)}]^\top = U^\top V_{\mathrm{dom}}. \tag{18}$$

Covariance matrix reformulation. Naturally, the total variance remains invariant under basis transformation. We relate the basis U and E by constructing the variance in the different bases:

$$\begin{aligned}\mathrm{Var}(V_{\mathrm{dom}}) &= V_{\mathrm{dom}}^\top \Sigma V_{\mathrm{dom}} = V_{\mathrm{dom}}^{(U)\top}\Sigma_U V_{\mathrm{dom}}^{(U)} \\ &= (U^\top V_{\mathrm{dom}})^\top \Sigma_U (U^\top V_{\mathrm{dom}}) \\ &= V_{\mathrm{dom}}^\top U\Sigma_U U^\top V_{\mathrm{dom}}. \end{aligned} \tag{19}$$

Comparing the expressions, we find $U\Sigma_U U^\top = \Sigma$. Since U is an orthogonal matrix, it satisfies $U^\top U = I$ and $U^{-1} = U^\top$. We can therefore express Σ_U as:

$$\Sigma_U = I\cdot\Sigma_U\cdot I = (U^\top U)\Sigma_U(U^\top U) = U^\top(U\Sigma_U U^\top)U = U^\top\Sigma U. \tag{20}$$

F.4 Details of Rayleigh Quotient

For a symmetric matrix $M \in \mathbb{R}^{d\times d}$ and a non-zero vector $w \in \mathbb{R}^d$, the Rayleigh quotient is defined as $R(w) = \frac{w^\top M w}{w^\top w}$. When M is symmetric, $R(w)$ is bounded by the minimum and maximum eigenvalues of M. The extrema are attained at the corresponding eigenvectors:

$$\begin{cases}\max\limits_{w\neq 0} R(w) = \lambda_{\max}(M), & \text{attained at } w = v_{\max}, \\ \min\limits_{w\neq 0} R(w) = \lambda_{\min}(M), & \text{attained at } w = v_{\min}, \end{cases} \tag{21}$$

where $\lambda(\cdot)$ denotes the eigenvalue and v denotes the corresponding eigenvector. In our context, the Rayleigh quotient takes the form $R(w) = w^\top M w = \frac{1}{N}\sum_{k=1}^{N}(w^\top v_{g,k})(w^\top v_{s,k})$, given $\|w\| = 1$.

The maximization of $R(w)$ to $\lambda_{\max}$ implies the projections $(w^\top v_{g,k})$ and $(w^\top v_{s,k})$ exhibit the *highest positive correlation*. This direction, $v_{\max}$, corresponds to the *most consistent and discriminative direction*. Similarly, the minimization of $R(w)$ to $\lambda_{\min}$ implies the *highest negative correlation* and $v_{\min}$ corresponds to the *most inverted and discriminative direction*.

We therefore use this Rayleigh quotient formulation to identify the consistent and inverted features.