# Toward Localizing and Repairing Bias in Transformer Attention Heads

Sigma Jahan
Dalhousie University, Canada
sigma.jahan@dal.ca

***Abstract*—Transformer language models are increasingly used as software components, yet biased outputs remain difficult to localize and repair inside the model. Existing fairness testing and repair methods largely operate at the input-output or retraining level, while recent work suggests that bias-related behavior can concentrate in a small set of attention heads. This paper studies whether attention heads can be localized and repaired through a targeted inference-time intervention. We introduce ROBIN, a white-box head-level fairness debugging method that ranks attention heads using sensitivity to fairness probes and removes a small bias subspace from selected head outputs. In a four-model pilot study, ROBIN reduces the measured WinoBias gap across all models while preserving language-modeling quality better than whole-head zeroing. These preliminary results suggest that head-level bias repair should consider not only which heads are selected, but also how selected heads are modified.**

***Index Terms*—bias, fairness, debugging, attention heads**

## I. Introduction

Real-world deployments of machine-learning software systems have produced documented unfair outcomes. Amazon withdrew an internal resume-screening system after engineers found that it ranked applicants lower when their resumes mentioned women's colleges or the word "women's" [1]. For ML-enabled software maintenance, this case exposes a debugging gap. Developers may reproduce a fairness failure at the system boundary yet still lack an internal location or a bounded repair. Similar concerns appear across natural language processing systems, where fairness failures are visible in outputs but difficult to trace to internal model behavior [2].

Software engineering research has developed two complementary directions for addressing fairness bugs. The first direction focuses on testing software for discriminatory behavior. Galhotra et al. [3] introduced fairness testing as a quality property of the software itself, and Udeshi et al. [4] developed directed test generation to expose individual discriminatory inputs. Adjacent work on testing and debugging deep neural networks more broadly, including concolic testing [5] and state-differential model debugging [6], treats neural models as software artifacts amenable to traditional analysis. Surveys synthesize this work and find that fairness defects are difficult to characterize because their causes depend on data, model behavior, protected attributes, and deployment context [7, 8].

The second direction repairs models after a failure. Chakraborty et al. [9] reweight or relabel training data to reduce bias before deployment, and Zhang and Harman [10] study how repair interacts with accuracy loss. These methods help practitioners detect and mitigate unfair behavior, but they operate from outside the model. They reveal unequal treatment or retrain the model without identifying which internal structures contribute to an inference-time fairness failure. A white-box debugging method could complement them by localizing candidate components and applying a bounded change without retraining the full model.

Recent work suggests that bias-related behavior in masked language models can concentrate in a small set of attention heads [11]. This finding narrows the localization search but does not determine how a selected head should be modified. Head-attribution methods rank heads using gradients or activations [12, 13]. Whole-head zeroing then provides a simple intervention, but it assumes that a selected head primarily encodes bias even though one head may also support general linguistic behavior [14, 13]. We instead seek an operator that retains output components orthogonal to the estimated bias. Our design combines empirical-Fisher importance scoring [15] with multi-dimensional projection derived from bias-direction and null-space methods [16, 17]. This combination separates which heads to inspect from how to modify them.

A selected attention head may contribute to both the measured fairness gap and ordinary language modeling behavior, so zeroing the whole head can remove useful information along with the biased response. In this paper, we introduce **ROBIN** (**R**epair **O**f **B**ias via **IN**tervention), a white-box method for head-level fairness debugging in transformer language models. ROBIN ranks attention heads by their sensitivity to fairness probes and removes a small bias subspace from the top-$k$ head outputs at inference. We evaluate ROBIN in a pilot study on DistilBERT, BERT, DistilGPT-2, and GPT-2. At $k = 20$, ROBIN reduces the WinoBias gap on all four models with at most a 7.2% increase in language-modeling loss.

This paper makes the following contributions:

- We frame head-level fairness mitigation as a white-box software debugging task that separates head localization from component-level modification.
- We propose ROBIN, a method that identifies bias-sensitive attention heads and removes only the biased part of each head at inference.
- We provide initial evidence on four pretrained transformer models that ROBIN reduces the measured WinoBias gap while limiting the language-modeling loss increase to 7.2%.

## II. Methodology

ROBIN assumes white-box access to a trained transformer language model [18] $f_\theta$ and to its tokenizer. The fairness probes are a set of $N$ text pairs $(x_a^{(i)}, x_b^{(i)})$ that differ only in the *protected attribute* (the demographic dimension along which we measure unfairness, e.g., gender or race). For WinoBias the difference is a gender pronoun (he/she), and for StereoSet it is a sentence-level change between a stereotype and an anti-stereotype version. Given these inputs and two user-specified budgets, $k$ (how many heads to modify) and $r$ (how many bias directions to remove from each chosen head), ROBIN operates in two stages (Fig. 1). *Stage 1* ranks every attention head by its sensitivity to the difference between $x_a$ and $x_b$. *Stage 2* modifies the top $k$ heads at inference.

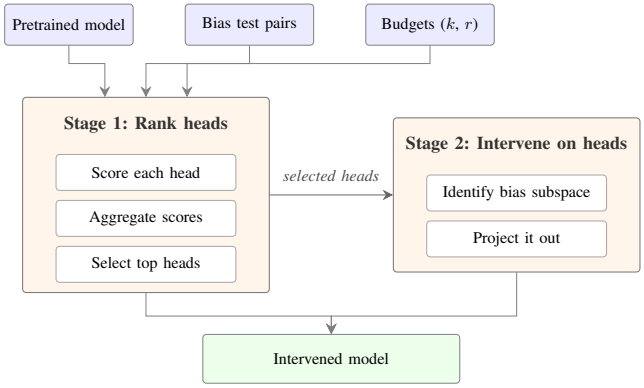



Fig. 1. ROBIN workflow

*Per-pair score.* The method assigns each text a confidence score computed only at the positions where the paired texts differ. We denote those positions by $D$. For a masked language model, we mask the changed positions and define

$$p(\text{text}) = \exp\left(-\tfrac{1}{|D|}\sum_{t\in D}\mathcal{L}_t\right) \tag{1}$$

where $\mathcal{L}_t$ is the negative log-likelihood of the original token at position $t$. For a causal language model, we instead run the model unmasked and use the probability of the original token at each changed position,

$$p(\text{text}) = \exp\left(\tfrac{1}{|D|}\sum_{t\in D}\log P_\theta(w_t \mid w_{<t})\right) \tag{2}$$

In both cases $p(\text{text}) \in [0,1]$ measures the model's confidence on the changed positions.

*Bias metrics.* We report two bias metrics built from $p$. The *WinoBias gap* is the mean absolute difference of $p$ between the two sides of a pair, where lower indicates a smaller gap.

$$\text{Gap} = \frac{100}{N}\sum_{i=1}^{N}\left|p(x_a^{(i)}) - p(x_b^{(i)})\right| \tag{3}$$

The *StereoSet stereotype score* (SS) is the percentage of pairs where the model prefers the stereotype sentence over the anti-stereotype. A value of 50 means no preference. A value above 50 indicates a stereotype preference under this metric.

$$\text{SS} = \frac{100}{N}\sum_{i=1}^{N}\mathbb{1}\left[p(\text{stereo}^{(i)}) > p(\text{anti}^{(i)})\right] \tag{4}$$

**1. Rank attention heads.** Stage 1 turns the probe pairs into a list of attention heads, from most to least bias-sensitive.

*1a. Per-pair gradient.* For each pair we run the model twice, once on $x_a$ and once on $x_b$, with loss $\mathcal{L} = -\log p(\text{text})$. We record the size of the loss gradient with respect to the attention weights inside each head,

$$g_h^{(i)} \;=\; \|\nabla_{A_h}\mathcal{L}\|_2 \tag{5}$$

where $A_h$ is the attention-weight matrix of head $h$ on side $i$. A larger $g_h^{(i)}$ indicates greater local sensitivity of the loss to that head's attention weights. Each pair gives us two such numbers per head.

*1b. Gradient score.* We average the gradient sizes from Step 1a over all $2N$ runs (two per pair) to get one number per head,

$$G_h \;=\; \frac{1}{2N}\sum_{i=1}^{2N} g_h^{(i)} \tag{6}$$

A head with a large $G_h$ has high average sensitivity to the gender change.

*1c. Squared-gradient score.* Alongside the simple average, we also compute the average of the *squared* gradient sizes,

$$C_h \;=\; \frac{1}{2N}\sum_{i=1}^{2N}\left(g_h^{(i)}\right)^2 \tag{7}$$

Squaring gives more weight to heads with a few large per-pair responses. The mean-gradient score $G_h$ favors heads that respond consistently across many probe pairs, whereas $C_h$ favors heads whose response concentrates on a smaller subset. We use both since bias-responsive heads can take either form. Some may react broadly across pairs, while others may react sharply on a few and stay quiet elsewhere [11]. The squared-gradient form follows empirical-Fisher importance scoring used in Elastic Weight Consolidation [15], applied at the attention-head level following head-importance scoring in prior pruning work [12].

*1d. Combine the scores.* The $G_h$ and $C_h$ values are on different scales, so we put each on a common scale by subtracting its mean and dividing by its standard deviation. Writing this rescaling as $z(\cdot)$, the combined ROBIN score is

$$F_h \;=\; z(G_h) + z(C_h) \tag{8}$$

The combined score therefore ranks heads highly when they show both broad sensitivity across probe pairs and concentrated high sensitivity on some pairs. We weight the standardized terms equally to avoid tuning a model-specific coefficient in this pilot study.

*1e. Pick the top $k$ heads.* We rank heads by $F_h$ from largest to smallest and keep the top $k$ as the chosen set $\mathcal{S}_k$. In our experiments we try $k \in \{1,2,3,5,8,10,15,20\}$. The importance score drops sharply within the first few ranks on every model we evaluate and flattens well before rank 30 (Fig. 2), which is the empirical reason we restrict $k$ to small values rather than searching higher budgets.

**2. Intervene on chosen heads.** Stage 2 takes the $k$ heads from Stage 1 and modifies their outputs at inference.

*2a. Find the bias subspace.* For each head $h \in \mathcal{S}_k$, we estimate a bias subspace, which we define as the directions

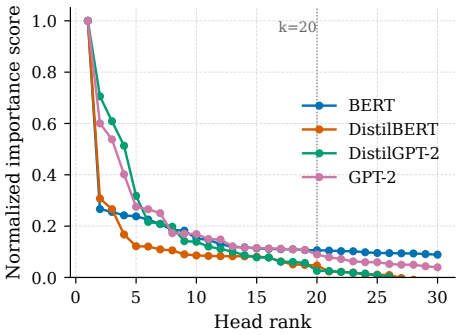



Fig. 2. Top-30 ROBIN importance scores, per-model normalized

in the head output that change most when the protected attribute is altered. We take up to 400 WinoBias pairs and, for each pair, average the head's output on the changed-token positions to obtain one vector for $x_a$ and one for $x_b$. Their difference $d^{(i)} = a^{(i)} - b^{(i)} \in \mathbb{R}^{d_h}$, where $d_h$ is the head's output dimension, is one example of how the head's output shifts under this change. Stacking these difference vectors as rows gives a matrix $D_h$ with up to 400 rows. If the bias signal followed one direction, the $d^{(i)}$ vectors would be nearly aligned. In practice, they occupy several directions, so we keep the $r$ directions with the largest singular values. These are the top $r$ right singular vectors of $D_h$,

$$V_h \;=\; \mathrm{TopRightSVD}_r(D_h) \;\in\; \mathbb{R}^{r \times d_h} \tag{9}$$

We use $r = 4$ throughout. Setting $r = 1$ recovers the single-direction estimate of Bolukbasi et al. [16].

*2b. Remove the bias subspace.* When the model runs on new text, each chosen head computes its output $x_h$ as usual, and we then subtract the part of $x_h$ that lies along the bias subspace $V_h$,

$$x'_h \;=\; x_h - V_h^\top V_h \, x_h \tag{10}$$

The term $V_h^\top V_h \, x_h$ is the projection of $x_h$ onto the bias subspace. Removing this projection keeps the remaining part of the head output. For batch size $B$ and sequence length $T$, the added cost is $O(BTd_hr)$ per chosen head.

*2c. Subspace removal vs. head ablation.* Attention heads in transformer language models can support multiple functions [13, 14]. A head that responds strongly to gendered pronouns may also support general linguistic behavior, including this kind of pronoun-tracking. Zeroing the whole head removes both at once. Subtracting only the part of $x_h$ that lies along $V_h$ removes the bias response and keeps the rest. As Section IV reports, this distinction matters empirically.

## III. Experimental Design

**Models.** We evaluate ROBIN on four publicly available pretrained transformer language models that span both the encoder and decoder families and both base-size and distilled variants. The encoders are BERT [19] and DistilBERT [20], which we score as masked language models. The decoders are GPT-2 [21] and DistilGPT-2, which we score as causal language models. The two base-size models have 144 attention heads each (12 layers $\times$ 12 heads), and the two distilled models have 72 (6 layers $\times$ 12 heads), a difference that becomes relevant when we discuss inference cost in Section IV.

**Datasets.** We use two paired bias benchmarks and one general corpus. *WinoBias* [22] provides sentences in which the model must resolve a pronoun (he or she) to one of two professions, set up so that the stereotype-aligned resolution differs from the anti-stereotype one. We form pairs by replacing every gendered pronoun with its opposite, giving 1,584 pairs after filtering pairs unchanged by the replacement. *StereoSet* [23] provides intrasentence stereotype-vs-anti-stereotype pairs across gender, race, profession, and religion, giving 2,106 pairs. *WikiText-2* [24] provides general text on which we measure language-modeling loss. WinoBias drives the head ranking and the main bias result. StereoSet serves as a secondary fairness check across protected categories.

**Compared Methods.** We compare three head-importance scores under the same evaluation setup. All three derive from the per-pair gradient norm $g_h^{(i)}$ defined in Eq. (5). They differ in how we aggregate the per-pair scores and in how we modify selected heads at inference.

- *Gradient.* The per-head mean of $g_h^{(i)}$ across all pair-sides, $G_h$ in Eq. (6). At inference, we zero the top $k$ heads by setting their slice of the attention output projection input to zero before the projection runs.

- *SquaredGrad.* The per-head mean of $(g_h^{(i)})^2$, $C_h$ in Eq. (7). We zero the top $k$ heads in the same manner as for Gradient.

- ROBIN. The standardized sum $F_h = z(G_h) + z(C_h)$ in Eq. (8). We do not zero the top $k$ heads. Instead, for each selected head we estimate a four-dimensional bias subspace $V_h$ (Eq. (9)) and replace the head output $x_h$ with $x_h - V_h^\top V_h x_h$ (Eq. (10)) at inference. Projecting out a learned bias subspace at the representation level has precedent in sentence-level debiasing [25] and in iterative null-space methods [17], but those operate on whole-model outputs rather than on a small number of selected heads.

This setup separates the choice of head-importance score from the choice of inference-time modification. We vary $k$ over $\{1, 2, 3, 5, 8, 10, 15, 20\}$, fix $r = 4$ across all models (chosen from a BERT pilot comparison), and report means over three seeds. Across seeds, we vary the subset of probe pairs used to estimate the bias subspace and the masking pattern applied when scoring encoder quality.

**Metrics.** We report four metrics, two for fairness, one for model performance, and one for inference cost.

- *WinoBias gap (Gap).* Lower is better for this metric. An unmodified model with Gap close to 0 assigns similar scores to the two pronoun forms (see Eq. 3).

- *StereoSet stereotype score (SS).* A value of 50 indicates no stereotype preference. We report SS as secondary evidence, since it is less stable than Gap in our experiments and we do not base our main fairness claim on it (see Eq. 4).

- *Language-modeling loss.* We measure model performance on WikiText-2 using a single metric we refer to as *language-modeling loss*, with lower values indicating better performance. The exact formula depends on the model family. For decoders (GPT-2, DistilGPT-2), it is the standard token-level

perplexity over length-128 chunks, defined as exp of the mean negative log-likelihood. Encoders (BERT, DistilBERT) do not define an autoregressive likelihood, so we instead report a masked-token pseudo-perplexity at 15% masking, following Salazar et al. [26]. We use one umbrella term for readability, but encoder and decoder values should be compared only within model family, not across.

- *Inference latency.* The average time the model takes to process one batch of input on a single GPU. We report it as the percentage overhead each method adds compared to the unmodified model. Details of the hardware and batch setup are in the replication package.

## IV. Results

**RQ1. Does ROBIN reduce the WinoBias gap in pretrained transformer language models?**

At $k = 20$, ROBIN reduces the WinoBias gap on every model (Table I). The largest absolute reduction is on BERT, the model that started most biased ($45.97 \rightarrow 19.14$). The other three models show smaller absolute reductions. StereoSet stereotype score moves inconsistently across models and protected categories, so we treat it as secondary and answer RQ1 using WinoBias gap as the primary fairness metric.

TABLE I
BIAS AND LANGUAGE-MODELING LOSS AT $k = 20$

*(a) Bias.* Gap and SS by model and method

| | Baseline | | Gradient | | SquaredGrad | | ROBIN | |
|---|---|---|---|---|---|---|---|---|
| Model | Gap | SS | Gap | SS | Gap | SS | Gap | SS |
| DistilBERT | 10.78 | 56.32 | 2.15 | 55.46 | 2.15 | 55.46 | 2.51 | 57.03 |
| BERT | 45.97 | 54.75 | 20.73 | 56.17 | 27.08 | 56.98 | 19.14 | 55.75 |
| DistilGPT-2 | 14.65 | 61.87 | 4.49 | 62.92 | 4.18 | 62.49 | 6.95 | 62.63 |
| GPT-2 | 15.34 | 60.16 | 10.50 | 62.30 | 10.50 | 62.30 | 9.78 | 61.68 |

*(b) Language-modeling loss on WikiText-2*

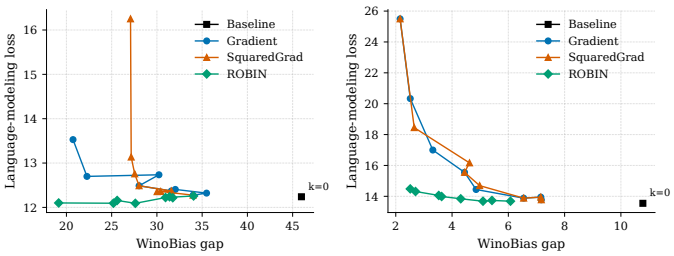
| Model | Baseline | Gradient | SquaredGrad | ROBIN |
|---|---|---|---|---|
| DistilBERT | 13.56 | 25.16 | 25.16 | 14.48 |
| BERT | 12.11 | 13.43 | 15.93 | 11.98 |
| DistilGPT-2 | 126.23 | 311.88 | 250.00 | 135.29 |
| GPT-2 | 71.35 | 96.31 | 96.31 | 75.59 |

*Note. Lower is better for Gap and language-modeling loss. SS closer to 50 is better.*

***Finding 1.*** *At $k = 20$,* ROBIN *reduces the WinoBias gap on all four models and more than halves it on three, with the largest absolute drop on BERT ($45.97 \rightarrow 19.14$).*

**RQ2. How does ROBIN trade off WinoBias-gap reduction against language-modeling loss?**

As $k$ grows, Gap generally decreases while language-modeling loss rises (Fig. 3). ROBIN limits this rise compared with whole-head zeroing. On BERT, it reaches the lowest $k = 20$ Gap at near-baseline language-modeling loss. On DistilBERT, its Gap is slightly higher than SquaredGrad (2.51 versus 2.15), but the increase in loss is 11 points smaller (14.48 versus 25.16). On GPT-2, ROBIN stays within a few loss points of baseline across $k$, whereas zeroing ends with a 35% increase at $k = 20$ (Fig. 4). DistilGPT-2 follows the same pattern at a larger absolute scale. Much of the $k = 20$ Gap reduction is already present at $k \in \{8, 10\}$, allowing a smaller budget when model quality has higher priority.




(a) BERT (b) DistilBERT

Fig. 3. WinoBias gap vs. language-modeling loss on WikiText-2 as $k$ varies

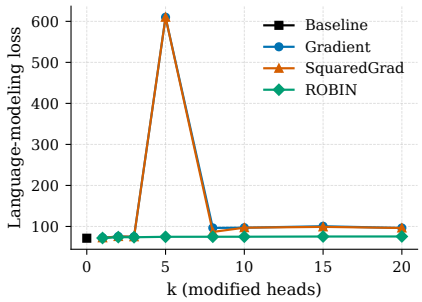



Fig. 4. GPT-2 language-modeling loss on WikiText-2 as $k$ varies

***Finding 2.*** *The top-$k$ rankings overlap strongly, but the operators differ sharply. Subspace removal preserves model performance better than whole-head zeroing, which can increase GPT-2 language-modeling loss by up to $8\times$.*

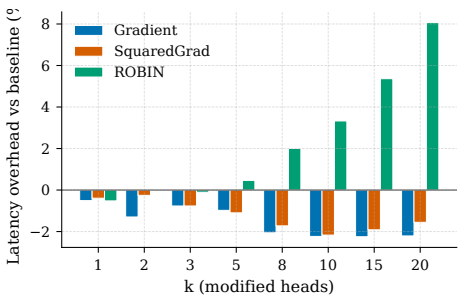



Fig. 5. Latency overhead on BERT as $k$ varies

**RQ3. How much inference cost does ROBIN add?**

ROBIN adds one matrix multiplication of shape $(d_h, r)$ per chosen head per forward pass. The resulting overhead at $k = 20$ ranges from 7% on the base-size models to 11% on the distilled models, because the same $k$ modifies a larger fraction of their attention heads (Table II). For the fixed batch size and sequence length we measured, overhead grows roughly linearly with $k$ (Fig. 5).

TABLE II
FORWARD-PASS LATENCY (MS) AT $k = 20$

| Model | Baseline | Gradient | SquaredGrad | ROBIN |
|---|---|---|---|---|
| DistilBERT | 7.20 | 7.09 (−1.5%) | 7.15 (−0.7%) | 8.03 (+11.4%) |
| BERT | 12.04 | 11.81 (−2.0%) | 11.82 (−1.9%) | 12.98 (+7.8%) |
| DistilGPT-2 | 8.35 | 8.27 (−1.0%) | 8.29 (−0.7%) | 9.25 (+10.8%) |
| GPT-2 | 13.32 | 13.01 (−2.3%) | 13.03 (−2.2%) | 14.28 (+7.2%) |

*Note. Parentheses give change vs. baseline. Small negatives are measurement noise.*

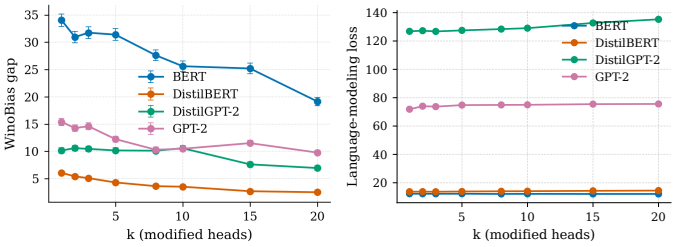
***Finding 3.*** *For fixed batch size and sequence length,* ROBIN*'s measured latency overhead grows with $k$. The added operation has cost $O(BTkd_hr)$ per forward pass.*

## V. Discussion

*Relation to prior work.* SE fairness testing detects discriminatory behavior at the system boundary [3, 4], while data-oriented repair changes training data or retrains the model [9, 10]. Transformer attribution ranks attention heads [12, 13], and recent search-based repair prunes selected heads [27]. Representation-level debiasing instead removes learned subspaces from whole-model representations [25, 17]. ROBIN combines component localization with subspace removal restricted to selected head outputs. This white-box debugging contribution differs from whole-head pruning and does not claim a new fairness metric or causal explanation.

*Operator vs. ranking.* Within each model, the three rankings overlap, with 15 to 20 shared heads in each top-20 list and Spearman correlations above 0.9 between any two rankings. These overlaps make it unlikely that ranking differences alone explain the observed trade-off between fairness and model performance. This pattern is consistent with the per-head output carrying both bias and orthogonal components, since subspace removal targets the bias component while whole-head zeroing removes all components of the head output.

*Cross-model pattern.* Fig. 6 shows the same pattern across the models. Gap generally decreases with $k$, with small non-monotonic steps, and model performance stays close to the baseline.




(a) WinoBias gap (b) Language-modeling loss

Fig. 6. ROBIN across the four models as $k$ varies

*Scope and use.* ROBIN reduces a paired benchmark gap under our protocol. This result does not establish that the intervened models produce fairer outputs in realistic use, improve downstream-task fairness, or avoid new disparities outside the evaluated pairs. Gap and SS are proxy measures, while language-modeling loss and latency cover only broad model quality and runtime cost. These metrics do not determine the intervention's overall benefits and costs in deployment. ROBIN therefore provides candidate locations and a measurable intervention, not a causal explanation or a complete fairness assessment. A practitioner can choose $k$ from the measured trade-off and then validate model outputs in the intended context. This separation between localization and root-cause attribution matches standard SE debugging practice [28].

*StereoSet.* StereoSet stereotype score does not move consistently across models and categories under our intervention. Some (model, category) combinations move toward neutrality and others away, without a consistent direction. Measurement-modeling work has documented validity concerns with StereoSet and similar contrastive benchmarks [29], which is consistent with the noise we observe. We therefore base our main claim on WinoBias gap and language-modeling loss, where the fairness probes are paired by protected attribute and match the data used to estimate the per-head bias subspace. Generalizing to broader stereotype phenomena is a separate empirical question that this study does not answer.

## VI. Threats to Validity

Threats to *internal validity* concern experimental choices that may affect the results. The fixed subspace dimension $r$ may change the trade-off between the WinoBias gap and language-modeling loss. We chose $r = 4$ from a BERT pilot over $r \in \{1, 2, 4, 8\}$ and held it fixed across models to avoid model-specific tuning. We use WinoBias for head ranking, subspace estimation, and the main gap evaluation. This reuse creates evaluation bias because the intervention may specialize to the same probe distribution used to construct it. StereoSet provides a secondary check, but its inconsistent results do not resolve this threat. A full evaluation requires disjoint data for ranking, subspace estimation, and fairness assessment. We also report means over three seeds without confidence intervals or statistical tests, so small differences between methods remain uncertain.

Threats to *construct validity* concern whether the reported metrics capture fairness, model quality, and intervention cost. WinoBias covers gender bias in pronoun resolution, and its Gap metric assumes that the paired pronoun forms should receive similar scores. This assumption does not extend to contexts where a protected-attribute change is semantically relevant. StereoSet covers four stereotype categories, but neither benchmark establishes fairness in generated outputs or downstream tasks. WikiText-2 language-modeling loss does not capture all semantic output degradation, and latency does not capture all deployment costs. We therefore limit our claims to the evaluated probes, quality metric, and runtime setting.

Threats to *external validity* concern generalization beyond the evaluated setting. The four models cover encoder, decoder, base, and distilled architectures, but not larger or instruction-tuned models. The latency results use one GPU and one batch shape, so the measured overhead may change under other hardware or implementations. We report the added cost as $O(BTkd_hr)$ per forward pass to clarify how the intervention scales.

## VII. Conclusion and Future Work

Transformer language models can produce biased outputs whose internal contributors are difficult to localize. We propose ROBIN, a head-level fairness debugging method that selects attention heads and removes a small bias subspace from their outputs rather than zeroing them. In this pilot, ROBIN reduced the measured WinoBias gap while limiting language-modeling degradation. Future work will evaluate larger models, broader attributes, disjoint data, generated outputs, and downstream-task fairness. It will also study $r$ selection and subspace transfer.

## References


[1] J. Dastin, "Amazon scraps secret AI recruiting tool that showed bias against women," Reuters, 2018.

[2] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell, "On the dangers of stochastic parrots: Can language models be too big?" in *Proc. FAccT*, 2021, pp. 610–623.

[3] S. Galhotra, Y. Brun, and A. Meliou, "Fairness testing: Testing software for discrimination," in *Proc. FSE*, 2017, pp. 498–510.

[4] S. Udeshi, P. Arora, and S. Chattopadhyay, "Automated directed fairness testing," in *Proc. ASE*, 2018, pp. 98–108.

[5] Y. Sun, M. Wu, W. Ruan, X. Huang, M. Kwiatkowska, and D. Kroening, "Concolic testing for deep neural networks," in *Proc. ASE*, 2018, pp. 109–119.

[6] S. Ma, Y. Liu, W.-C. Lee, X. Zhang, and A. Grama, "MODE: Automated neural network model debugging via state differential analysis and input selection," in *Proc. ESEC/FSE*, 2018, pp. 175–186.

[7] J. Chakraborty, S. Majumder, and T. Menzies, "Bias in machine learning software: Why? how? what to do?" in *Proc. ESEC/FSE*, 2021, pp. 429–440.

[8] Z. Chen, J. M. Zhang, M. Hort, M. Harman, and F. Sarro, "Fairness testing: A comprehensive survey and analysis of trends," *ACM TOSEM*, vol. 33, no. 5, pp. 1–59, 2024.

[9] J. Chakraborty, S. Majumder, Z. Yu, and T. Menzies, "Fairway: A way to build fair ML software," in *Proc. ESEC/FSE*, 2020, pp. 654–665.

[10] J. M. Zhang and M. Harman, "Ignorance and prejudice in software fairness," in *Proc. ICSE*, 2021, pp. 1436–1447.

[11] Y. Yang, H. Duan, A. Abbasi, J. P. Lalor, and K. Y. Tam, "Bias a-head? Analyzing bias in transformer-based language model attention heads," *arXiv:2311.10395*, 2023.

[12] P. Michel, O. Levy, and G. Neubig, "Are sixteen heads really better than one?" in *NeurIPS*, 2019.

[13] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, "Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned," in *Proc. ACL*, 2019, pp. 5797–5808.

[14] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, "What does BERT look at? an analysis of BERT's attention," in *ACL Workshop BlackboxNLP*, 2019, pp. 276–286.

[15] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks," *PNAS*, vol. 114, no. 13, pp. 3521–3526, 2017.

[16] T. Bolukbasi, K.-W. Chang, J. Y. Zou, V. Saligrama, and A. T. Kalai, "Man is to computer programmer as woman is to homemaker? debiasing word embeddings," in *NeurIPS*, 2016.

[17] S. Ravfogel, Y. Elazar, H. Gonen, M. Twiton, and Y. Goldberg, "Null it out: Guarding protected attributes by iterative nullspace projection," in *Proc. ACL*, 2020, pp. 7237–7256.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017.

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL*, 2019, pp. 4171–4186.

[20] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter," in *NeurIPS Workshop*, 2019.

[21] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Tech. Rep.*, 2019.

[22] J. Zhao, T. Wang, M. Yatskar, V. Ordonez, and K.-W. Chang, "Gender bias in coreference resolution: Evaluation and debiasing methods," in *Proc. NAACL*, 2018, pp. 15–20.

[23] M. Nadeem, A. Bethke, and S. Reddy, "StereoSet: Measuring stereotypical bias in pretrained language models," in *Proc. ACL*, 2021, pp. 5356–5371.

[24] S. Merity, N. S. Keskar, and R. Socher, "Regularizing and optimizing LSTM language models," in *Proc. ICLR*, 2018.

[25] P. P. Liang, I. M. Li, E. Zheng, Y. C. Lim, R. Salakhutdinov, and L.-P. Morency, "Towards debiasing sentence representations," in *Proc. ACL*, 2020, pp. 5502–5515.

[26] J. Salazar, D. Liang, T. Q. Nguyen, and K. Kirchhoff, "Masked language model scoring," in *Proc. ACL*, 2020, pp. 2699–2712.

[27] V. A. Dasu, M. R. u. Rashid, V. Gupta, S. Tizpaz-Niari, and G. Tan, "Attention pruning: Automated fairness repair of language models via surrogate simulated annealing," in *Proc. ICSE*, 2026.

[28] W. E. Wong, R. Gao, Y. Li, R. Abreu, and F. Wotawa, "A survey on software fault localization," *IEEE TSE*, vol. 42, no. 8, pp. 707–740, 2016.

[29] S. L. Blodgett, G. Lopez, A. Olteanu, R. Sim, and H. Wallach, "Stereotyping Norwegian salmon: An inventory of pitfalls in fairness benchmark datasets," in *Proc. ACL-IJCNLP*, 2021, pp. 1004–1015.