Bias and Fairness in Large Language Models

唐诗 李严婧 黄瑞

1. Introduction

This study mostly got inspired from a recent ACL paper which is titled *Investigating Bias in LLM-Based Bias Detection: Disparities between LLMs and Human Perception* (Lin et al. ,2024[4]). We aim to verify the key findings and experimental results through our own replication efforts, which includes two questions: *Do LLMs exhibit inherent biases? How can we correct such biases?*

1.1 Research Background

Why do we want to analyze the bias and fairness in LLMs? The rise and rapid advancement of large language models (LLMs) has fundamentally changed language technologies (e.g., Brown et al 2020[2]). Laying behind the change, however, is the potential to perpetuate harm. Typically trained on an enormous scale of uncurated Internet - based data, LLMs inherit stereotypes, misrepresentations, derogatory and exclusionary language, and other degrading behaviors that disproportionately affect already - vulnerable and marginalized communities (Bender et al. 2021[3]). These harms are forms of "social bias", which we broadly use to refer to disparate treatment or outcomes between social groups that arise from historical and structural power asymmetries. Thus, it is important and meaningful for us to test and correct the biases of LLMs.

1.2 Research Framework

Do LLMs exhibit inherent biases? For this question, we mostly refer to Lin et al. (2024). We probe both classification-level and generative-level biases of LLMs on political news articles.

How can we correct such biases? For this question, we firstly reproduce the debias techniques in Lin et al. (2024), which includes *Supervised Fine-tuning*. Supervised Fine-tuning refers to the process of optimizing a pre-trained model using labeled downstream task data (e.g., question-answer pairs) to update its parameters for specific tasks.

What's more, we then refer to Gallegos et al. (2024)[1], which is a comprehensive review. Based on this paper, we find interest in *Loss Function Modification* for debiasing, which is to change the loss function based on *Embedding, Attention or Predicted Token Distribution*. We mostly modify the loss function for word embedding debiasing according to Yang et al. (2025)[5] and Kaneko and Bollegala (2021)[6], which is to address bias in the hidden representations of an encoder via a new equalizing objective or regularization constraints. We firstly use the dataset about gender bias and obtain some ideal results, then try this idea on the dataset about political bias.

2. Do LLMs exhibit inherent biases?

To assess political bias in large language models (LLMs), we adopt two complementary evaluation paradigms inspired by Lin et al. (2024). These paradigms enable us to probe both classification-level and generative-level biases of LLMs on political news articles. Our analysis centers on the FlipBias dataset, which comprises 3,066 news articles evenly distributed across three political leanings—Left, Center, and Right (1,022 articles per category).

2.1 Prompt-Based Classification Evaluation

In the first setting, we formulate media bias detection as a multi-class classification task. Each article, consisting of a title and body text, is fed to the LLM with the following system prompt:

"Given the text, could you answer whether it has media bias, such as left, center or right political leaning?

Text:{text}

Please answer one of the following phrases: <Left>, <Center>, <Right>"

The model is expected to output a single-word label: Left, Center, or Right. This approach aligns with the direct bias prediction setting proposed in prior work and allows us to compute standard classification metrics, including accuracy and confusion matrices.

Beyond these metrics, we focus on two Bias Tendency Indices introduced by Lin et al. (2024) that measure linear political

bias:

$$\text{BTI-1} = \frac{\text{Count}(\text{left} \rightarrow \text{center})}{\text{Count}(\text{left})} - \frac{\text{Count}(\text{right} \rightarrow \text{center})}{\text{Count}(\text{right})}$$

$$\text{BTI-2} = \frac{\text{Count}(\text{center} \rightarrow \text{right})}{\text{Count}(\text{center})} - \frac{\text{Count}(\text{center} \rightarrow \text{left})}{\text{Count}(\text{center})}$$

2.2 Article Continuation with Embedding-Based Labeling

In the second evaluation paradigm, we examine bias through text generation. Specifically, we prompt the model to continue a political article by providing the title and the first few sentences as context. Instead of asking for a classification, the model generates a plausible continuation in natural language.

To infer the political leaning of the generated content, we apply an embedding-based matching method. We first construct reference embeddings for Left, Center, and Right categories using a large number of labeled examples. Then, we embed each model-generated continuation using the text-embedding-3-small model from OpenAI and assign the predicted label based on cosine similarity to the reference vectors.

This method allows us to evaluate implicit bias—i.e., the political leaning that is encoded in the model's generative behavior, even without explicit labeling.

2.3 Results and Analysis

Our empirical findings reveal consistent and measurable political biases in the base GPT-3.5-Turbo model:

Method\Metric	BTI-1	BTI-2	Conclusion
QA	0.0176	-0.0206	Unknown
Artical Continuation (No Prefix)	-0.0337	0.0147	Unknown
Artical Continuation (20-token Prefix)	0.005	0.0137	Left-leaning Bias

Table 1: the BTI-1 and BTI-2 scores across bias evaluation paradigms.

Table 1 summarizes the BTI-1 and BTI-2 scores across both bias evaluation paradigms. Contrary to common observations in prior work, we find that GPT-3.5-Turbo does not exhibit a consistent or strongly directional political bias in our experiments.

These results suggest that, unlike what is reported in many benchmark studies, our tested LLM does not display a strong or consistent left- or right-leaning bias on the FlipBias dataset.

However, our analysis reveals a distinct and underexamined phenomenon: **centering bias**. In both classification (see *Figure 1*) and generation tasks, the model frequently labels partisan content as "Center", even when the ground truth clearly indicates Left or Right. This suggests that the model is not being neutral, but rather overly moderate—a form of bias that blurs ideological distinctions.

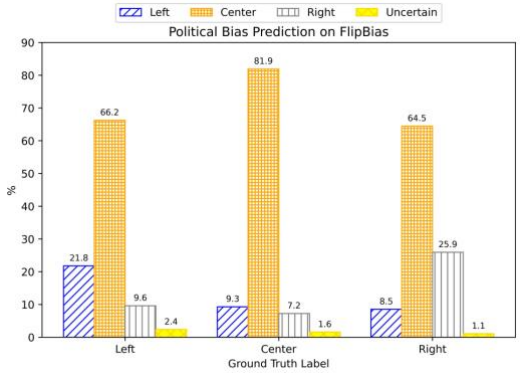

Figure 1: Political Bias Prediction on FlipBias.

Such behavior reflects more than just risk aversion. It indicates a tendency to avoid taking sides, even in the face of strongly opinionated input, effectively "watering down" polarized content. This may lead to problematic outcomes: extreme views could be falsely framed as centrist, and the model may appear tolerant of ideological extremism under the guise of balance. Worse, this moderation may be easily manipulated—a model reluctant to make sharp distinctions is more susceptible to rhetorical framing and

external influence.

These findings challenge the left-right linearity assumed in prior work. Bias in LLMs may not always lean in one direction—it may also collapse toward the center, sacrificing fidelity for perceived neutrality.

3. Bias Mitigation via Fine-Tuning

To investigate whether political bias in large language models (LLMs) can be mitigated through supervised learning, we fine-tuned GPT-3.5-Turbo using a small, balanced set of annotated news articles. Our goal was not only to reduce directional bias, but also to address the model's tendency toward centering bias—its consistent overprediction of the "Center" label, regardless of input polarity.

3.1 Method

We constructed a fine-tuning dataset using 150 samples from the FlipBias corpus, evenly split across Left, Center, and Right labels. Each sample included the title and content of a political news article, accompanied by a clear system prompt instructing the model to classify the political leaning into one of three categories. The dataset was formatted using OpenAI's chat completion schema and fine-tuned using the GPT-3.5-Turbo model over 3 epochs with default hyperparameters.

To evaluate the effectiveness of fine-tuning, we created a held-out test set of 200 articles with no overlap with the training data. Predictions were generated using the same classification prompt as in the original evaluation. We then analyzed the model's outputs using accuracy, confusion matrices, and qualitative distributional patterns.

3.2 Results

Fine-tuning had a measurable effect on the model's bias profile. As shown in the confusion matrix (Figure 2(3)), the model no longer exhibited the original centering bias; its predictions were more evenly distributed across Left, Center, and Right, suggesting that it had absorbed the empirical label distribution from the training data.

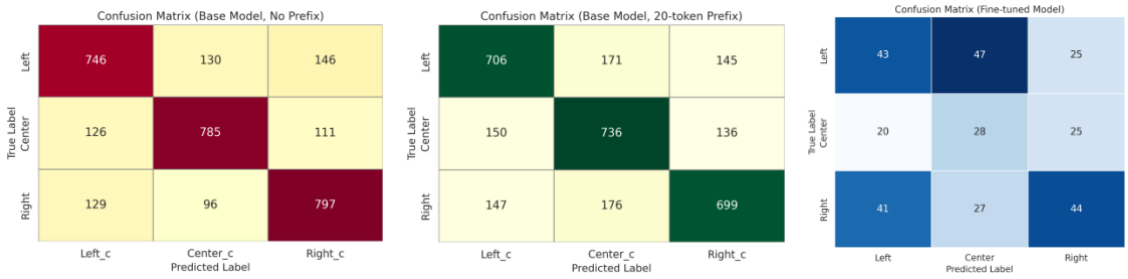

Figure 2: confusion matrix of the original model (left, center), and that of the fine-tuned model (right).

However, this apparent improvement came at a cost. The model's overall accuracy dropped to 38%, and its ability to distinguish between political classes deteriorated. The classification report below illustrates this clearly:

	PRECISION	RECALL	F1-SCORE	SUPPORT
CENTER	0.2745	0.3836	0.3233	73
LEFT	0.4135	0.3739	0.3926	115
RIGHT	0.4681	0.3929	0.4272	112
ACCURACY			0.3833	300
MACRO AVG	0.3854	0.3834	0.3800	300
WEIGHTED AVG	0.4000	0.3833	0.3879	300

Table 2: classification report of model after fine-tuning.

While the prediction distribution appears more balanced than the base model, performance metrics suggest that the fine-tuned model behaves more like a label-frequency memorizer than a meaningful classifier. The semantic information in the input text seems largely ignored.

3.3 Analysis

This performance degradation suggests that the model has overfitted to the small fine-tuning dataset of only 150 samples. The

limited scale and diversity likely caused the model to memorize label distributions rather than learning generalizable semantic patterns. As a result, while the model exhibits reduced centering bias, it does so at the cost of semantic accuracy and predictive robustness.

The confusion matrix further indicates a lack of clear class separation, reflecting a loss of discriminative capability. These results highlight a key trade-off: naive fine-tuning on small, balanced datasets may reduce representational bias but simultaneously impair task performance. This underscores the need for more principled debiasing approaches that retain the model's linguistic competence—such as regularization-aware training, prompt-based adjustment, or bias-sensitive loss functions.

4. Bias Mitigation via Loss Function Modification based on Embedding

4.1 Methods

The idea of loss function modification is quite intuitive. As the goal is to debias when embedding, we want to adjust the distance or distribution relationships between neutral words and attribute words (or stereotypical words) in the embedding space. Our goal is to *(a) removes discriminative social-related biases via loss function modification, and (b) preserves the semantic information in the pre-trained contextualized word embedding model*.

We reproduce two loss function modification technologies which satisfy above-mentioned conditions. Although we refer to many papers in this part, we would use the same mathematical symbols to represent the same meaning.

Algorithm Process: Debiasing Algorithms for contextualized word embedding, including DPCE and ADEPT. Input: Pre-trained Language Model (PLM). Output: Φ_{prompt} for debiasing the PLM.

1. First, we prepare a Pre-trained Language Model (PLM) $M_\Theta$with parameters Θ. Frozen Θ, which means we would not update most core parameters in PLM, in order to protect pre-trained knowledge.
2. Then we suppose a bias has d attributes, and thus we define a neutral word tuple $W^{neutral}$ and attribute word tuples $W^{a(i)} = (w_1^{a(i)}, w_2^{a(i)}, \dots, w_g^{a(i)})$, each with g one-to-one words.
3. Collect sentences $S^{neutral}$ and $\{S^{a(i)}\}_{i=1}^{d}$.
4. Initialize parameters Φ_{prompt}. Then calculate the loss of the dataset, using the DPCE or ADEPT loss function. Compute gradient and then update the parameters Φ_{prompt}.
5. Return the best Φ_{prompt}, and the debiased model is $M_{\Theta\cup\Phi}$.

Define Word Tuples: We define a neutral word tuple $W^{neutral}$ and several attribute word tuples $W^{a(i)}, i = 1,2,3 \dots d.$, where the category of bias we are debiasing for contains d different attributes. For example, gender bias may have the attributes "female" and "male," and here $d = 2$. And these word tuples are mutual exclusion.

Words in $W^{neutral}$ are nouns or adjectives that should show no preference for any of the d attributes. For example, "science" and "ambitious," which should not be bound to any attributes, might be in the tuple $W^{neutral}$.

$W^{a(i)}$ denotes a tuple of words where each word is associated with attribute $a(i)$ and not $a(j)$ for any $i \neq j$. What's more, each attribute word tuple is indexed by the same (implicit) indexing set of concepts and that the word at each index is of the same form. For brevity, we use the word "pairwise" to describe this correspondence. For example, in gender dataset, if the male attribute tuple is ("uncle", "father", "brother"), then female attribute tuple would be ("aunt", "mother", "sister").

Collect Sentences: Collect sentences based on the word tuples. $S^{neutral}$ denotes sentences that contain at least one word in $W^{neutral}$, and so as $S^{a(i)}$.

Calculate Attribute Centroid Vectors: For each attribute i, calculate the average embedding of its attribute tuple $W^{a(i)}$ across all its sentences $S^{a(i)}$. This averages embeddings of all words in $W^{a(i)}$ across all their occurrence sentences, capturing the "central representation" of the *i*-th attribute, e.g., a stable "male" vector.

$$e^{a(i)} = \frac{1}{|W^{a(i)}||S^{a(i)}|}\Sigma_{s\in S^{a(i)}}\Sigma_{w\in W^{a(i)}}E_i(w, s; \Theta, \Phi);$$

$E_i(w, s; \Theta, \Phi)$means contextual embedding of word w in sentence $s \in S^{a(i)}$, computed from the PLM with parameters Θ and prompt parameters Φ.

DPCE: Debiasing Pre-trained Contextualised Embeddings, which is referred to Kaneko and Bollegala (2021). Minimize the loss function:

$$L_{DPCE} = \alpha L_{debias} + \beta L_{reg}$$

Set $\alpha = 0.2$ and $\beta = 0.8$ in the experiment.

1. How to remove discriminative biases?

Construct the loss function representing the dot product between neutral word embeddings and attribute centroids, which can be seen as the embedding distance between neutral words and attribute words.

$$L_{debias} = \Sigma_{w \in W^{neutral}} \Sigma_{s \in S^{neutral}(w)} \Sigma_{i=1}^{d} \left[\left(e^{a(i)}\right)^T E_i(w, s; \Theta, \Phi) \right]^2;$$

And $S^{neutral}(w)$ means sets of sentences containing the neutral word $w \in W^{neutral}$.

2. How to preserve semantic information in PLM?

Construct the loss function representing the embedding distance before and after debiasing. By minimizing this function, we can retain the original semantic information of PLM.

$$L_{reg} = \Sigma_{i=1}^{d} \Sigma_{s \in S^{a(i)}} \Sigma_{w \in s} ||E_i(w, s; \Theta, \Phi) - E_i(w, s; \Theta)||$$

ADEPT: A Debiasing PrompT Framework, which is referred to Yang et al. (2025). It is similar to DPCE, also to minimize the loss function composed of two parts:

$$L_{ADEPT} = L_{debias} + \lambda L_{reg}$$

Set $\lambda = 7/3$ in the experiment.

1. How to remove discriminative biases?

Construct the loss function measuring the similarity distribution between attributes relative to neutral words, using Jensen-Shannon (JS) divergence.

Firstly, for each attribute $a(i)$, compute the probability distribution of neutral words around $e^{a(i)}$:

$$p_{w_j|a(i)} = \frac{\exp\left(-\frac{\left|\left|e^{a(i)} - e^{neutral;j}\right|\right|^2}{2\rho^2}\right)}{\sum_{w_k \in W^{neutral}} \exp\left(-\frac{\left|\left|e^{a(i)} - e^{neutral;k}\right|\right|^2}{2\rho^2}\right)};$$

Where $e^{neutral;j}$ means the embedding vector of neutral word $w_j \in W^{neutral}$. That's to say, $e^{neutral;j} = \frac{1}{|S^{neutral}(w_j)|} \Sigma_{s \in S^{neutral}(w_j)} E(w_j, s; \Theta, \Phi)$. We set $\rho = 15$ as the hyperparameter in the experiment.

This equation supposes that $w_k \in W^{neutral}$ follows isotropic Gaussian Distribution with mean $e^{a(i)}$ and variance ρ^2, and we can see $p_{w_j|a(i)}$ as the Softmax-normlized Gaussian likelihood or Gaussian kernel.

Then, we define $P^{a(i)} = \left\{p_{w_j|a(i)}\right\}_{j=1}^{|W^{neutral}|}$ as the distribution of neutral words conditioning the attribute word $a(i)$. Thus, we can use JS divergence to measure the difference of distribution of neutral words from perspective of different attribute words:

$$L_{debias} = \Sigma_{i<j} \left\{ JS\left(P^{a(i)} \middle|\middle| P^{a(j)}\right) \right\};$$

2. How to preserve semantic information in PLM?

Construct the loss function measuring the similarity distribution before and after debiasing using KL divergence.

$$L_{reg} = KL(P_{\Theta}^{w} \middle|\middle| P_{\Theta,\Phi}^{w});$$

P_{Θ}^{w} means the distribution of words conditioning the word $w \in W^{neutral} \cup \{W^{a(i)}\}_{i=1}^{d}$ in the PLM M_{Θ}, and $P_{\Theta,\Phi}^{w}$ means the distribution in the debiased model $M_{\Theta \cup \Phi}$.

Discussion of Loss Function: The core difference of DPCE and ADEPT is the method of measurement of difference between neutral words $W^{neutral}$ and different attribute words $W^{a(i)}$, with DPCE using dot product to measure "*distance*", and ADEPT using Gaussian kernel and JS divergence to measure "*distribution*". We think they are two main perspectives for viewing "*difference*", and this is also why we introduce and reproduce these two methods instead of others.

The common part is that they both find equilibrium between removing discriminative biases and preserving semantic

information in PLM.

Following this line of thought, we can design some other loss functions. For example, when measuring the distance, we can use Euclidean Distance, Cosine Similarity and so on, rather than Dot Product. When estimating empirical conditional probability density function, we can also use some Non-parameter methods, like Local Polynomial Regression or B-spline Smoother, to smoothen empirical p.d.f., rather than only Softmax-normlized Gaussian kernel. Also, when measuring the difference of distribution, we can use Chi-Square Divergence or Maximum Mean Discrepancy rather than JS divergence.

4.2 Results and Analysis

We mainly reproduce the result of Yang et al. (2025), comparing the original model, the DPCE model and the ADEPT model.

Benchmarks: We use three main benchmarks for evaluating the performance vis-à-vis bias:

SEAT: The Sentence Encoder Association Test (May et al. 2019[7]), aiming to measure biases in sentence-encoders like BERT. The smaller absolute value is considered better.

CrowS-Pairs: CrowS-Pairs (Nangia et al. 2020[8]) features pairwise test sentences, differing only in a stereotyped word and an anti-stereotyped word in the same position. Ideal score is 50.

StereoSet: StereoSet (Nadeem et al. 2020[9]) measures both a PLM's useful semantic information as well as its biases by using cloze tests. A higher, up to 100, Language Modeling Score (LMS) indicates better expressiveness, and a Stereotype Score (SS) closer to 50 indicates less biases.

GLUE: General Language Understanding Evaluation. It evaluates models' performance across diverse language understanding scenarios through a series of different natural language understanding tasks.

Manifold Learning: Manifold assumption indicates that the observed data lies on a low-dimensional manifold embedded in a higher-dimensional space. Yang et al. (2025) indicate that it is better to describe distribution of word embeddings with a manifold than with a linear subspace, and they use t-SNE method to visualize the word embeddings, showing that ADEPT is more effective at mitigating biases on a word embedding manifold than DPCE.

Training Process: The datasets are referred to Yang et al. (2025), including neutral word tuples, gender attribute word tuples and training sentences. We conduct experiments on the BERT-LARGE-UNCASED pre-trained model from HuggingFace (Wolf et al. 2019). We use Adam to optimize the objective loss function, with learning rate setting 5e-5. Batchsize is 32. All the experiments are conducted on 4 GeForce RTX 4060 GPUs and in a Linux operating system.

Results: We conduct We evaluate three models with the above benchmarks, namely *original model* (pre-trained with no debiasing), *the DPCE model* and *the ADEPT model*, and also use t-SNE method to visualize the word embeddings with a manifold.

	Original Model	**DPCE Model**	**ADEPT Model**
SEAT: Math/Arts	0.418	-0.871	**-0.281**
CrowS-Pairs: score(S)	55.73	46.74	**49.33**
StereoSet: SS	59.657	**57.532**	57.952
StereoSet: LMS	**86.338**	85.877	84.652
GLUE: SST-2	**92.8**	91.1	92.5
GLUE: RTE	69.3	61.0	**69.7**

Table3: Evaluation results on debiasing performance. What we reproduced is a little different from the paper Yang et al. (2025), maybe due to the training process.

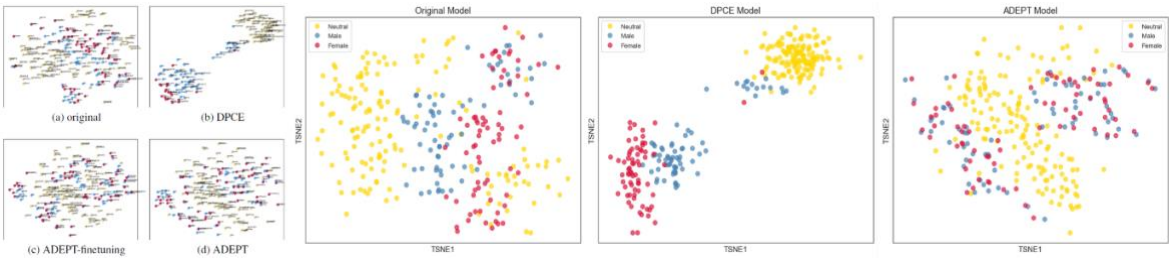

Figure3: Visualized correlation of words in the gender domain. We use t-SNE to plot the figures and set perplexity as 30. The first left one is the Figure 2 in the

paper Yang et al. (2025), and the three others are what we reproduced.

Reducing Biases: The first three rows of the table show that ADEPT model is always better than the DPCE model and the original model. In StereoSet test, ADEPT model is defeated.

Preserving Representation Ability: The GLUE test show that ADEPT model is better than the DPCE model from the perspective of semantic comprehension. Also, the score of ADEPT is close to that of original model. The score of DPCE model is lowest.

T-SNE Visualization: 1) In the original model, the words in male tuples and female tuples seem to have the similar shape, but have an apparent distinction. 2)In the DPCE model, the is a considerable distance between neutral word tuples and attribute word tuples, which means that it may not preserve the semantic information in PLM well. What's more, the distribution of attribute word tuples seems to shrink. 3)In the ADEPT model, the one-to-one relationship between male and female tuples seems to be emphasized. There must be a blue dot around a red dot, meaning the effect of debiasing. Also, the distribution of neutral words is not greatly changed, meaning ADEPT model preserve most of the original semantic information.

5. Conclusion and Discussion

In the study, we try to solve two problems: *Do LLMs exhibit inherent biases? How can we correct such biases?*

For the first question, we probe both classification-level and generative-level biases of LLMs on political news articles, i.e., finding LLMs have a tendency to avoid taking sides.

For the second question, we remove the biases via fine-tuning and loss function modification. We found that fine-tuning on small, balanced datasets may reduce representational bias but simultaneously impair task performance, thus we construct the loss function in order to find equilibrium between removing discriminative biases and preserving semantic information in PLM. We not only reproduce the DPCE and ADEPT algorithm and test them, but also put forward our own viewpoint of loss function modification. Furthermore, we can try other loss functions based on word embedding; or try loss function modification in the other parts of the model, i.e., adding the regularization term in the predicted token distribution to limit the occurrence of sensitive words.

6. References

[1] Lin, Luyang, Lingzhi Wang, Jinsong Guo, and Kam-Fai Wong. 2024. Investigating Bias in LLM-Based Bias Detection: Disparities between LLMs and Human Perception. arXiv preprint arXiv:2403.14896.

[2] Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few - shot learners. Advances in Neural Information Processing Systems, 33:1877–1901.

[3] Bender, Emily M. 2019. A typology of ethical risks in language technology with an eye towards where transparent documentation can help. Presented at The Future of Artificial Intelligence: Language, Ethics, Technology Workshop.

[4] Isabel O. Gallegos, Ryan A. Rossi, Joe Barrow, Md Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K. Ahmed. 2024. Bias and Fairness in Large Language Models: A Survey. Computational Linguistics, 50(3):1097–1179.

[5] Yang, Ke, Charles Yu, Yi Fung, Manling Li, and Heng Ji. 2025. ADEPT: A DEbiasing PrompT Framework. arXiv preprint arXiv:2211.05414v3.

[6] Kaneko, M.; and Bollegala, D. 2021. Debiasing pre-trained contextualised embeddings. arXiv preprint arXiv:2101.09523.

[7] May, C.; Wang, A.; Bordia, S.; Bowman, S. R.; and Rudinger, R. 2019. On measuring social biases in sentence encoders. arXiv preprint arXiv:1903.10561.

[8] Nangia, N.; Vania, C.; Bhalerao, R.; and Bowman, S. R. 2020. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. arXiv preprint arXiv:2010.00133.

[9] Nadeem, M.; Bethke, A.; and Reddy, S. 2020. Stereoset: Measuring stereotypical bias in pretrained language models. arXiv preprint arXiv:2004.09456.