Gold Doesn't Always Glitter: Spectral Removal of Linear and Nonlinear Guarded Attribute Information

Shun Shao **Yftah Ziser** **Shay B. Cohen**
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh, EH8 9AB
s.shao-11@sms.ed.ac.uk yftah.ziser@ed.ac.uk
scohen@inf.ed.ac.uk

Abstract

We describe a simple and effective method (Spectral Attribute removaL; SAL) to remove private or guarded information from neural representations. Our method uses matrix decomposition to project the input representations into directions with *reduced* covariance with the guarded information rather than *maximal* covariance as factorization methods normally use. We begin with linear information removal and proceed to generalize our algorithm to the case of nonlinear information removal using kernels. Our experiments demonstrate that our algorithm retains better main task performance after removing the guarded information compared to previous work. In addition, our experiments demonstrate that we need a relatively small amount of guarded attribute data to remove information about these attributes, which lowers the exposure to sensitive data and is more suitable for low-resource scenarios.[1]

1 Introduction

Natural language processing (NLP) models currently play a critical role in decision-supporting systems. Their predictions are often affected by undesirable biases encoded in real-world data they are trained on. Making sensitive predictions based on irrelevant input attributes such as gender, race, religion, or demographic (**protected** or **guarded** attributes) impacts user trust and the practical broad utility of NLP methods.

In recent years, representation learning approaches have become the mainstay of input encoding in NLP. While representation learning has yielded state-of-the-art results in many NLP tasks, it is hard to control or inspect the information encoded in these representations. Thus, using rule-based methods to remove unwanted information from such representations is often not feasible. In the context of protected attributes, Bolukbasi et al. (2016) showed that word embeddings trained on the Google News corpus encode gender stereotypes. Later, Manzini et al. (2019) expanded this work and showed that word embeddings trained on the Reddit L2 corpus (Rabinovich et al., 2018) encode race and religion biases.

We propose a simple yet effective technique to remove protected attribute information from neural representations. Our method, dubbed **SAL** for Spectral Attribute removaL, applies Singular Value Decomposition (SVD) on a covariance matrix between the input representation and the protected attributes and prunes highly co-varying directions. Figure 1 demonstrates how professional biography text representations from labeled gender clusters (each biography is marked with the gender of its subject; De-Arteaga et al. 2019) for different professions expand after the use of SAL, and become

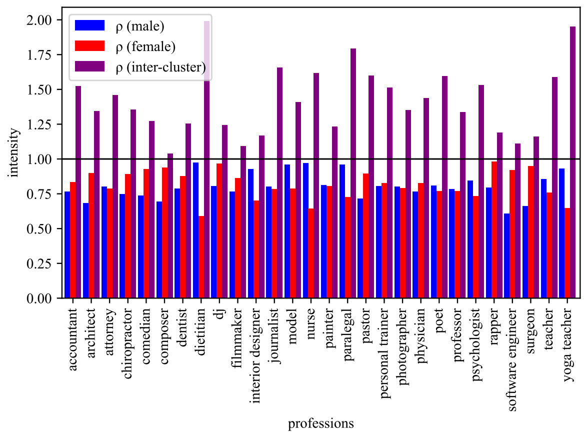

Figure 1: The ratio ρ between the average t-SNE similarity of representations between two gender clusters c_1, c_2 ($\text{sim}(c_1, c_2)$) for each profession: $\rho = \left(\textit{after SAL}\ \text{sim}(c_1, c_2) \big/ \textit{before SAL}\ \text{sim}(c_1, c_2)\right)$. Three values of ρ are computed, intra-cluster: (1) $c_1 = c_2 =$ male; (2) $c_1 = c_2 =$ female; and inter-cluster: (3) $c_1 =$ male, $c_2 =$ female. The ratios in the inter-cluster case are smaller than 1, and larger than 1 for the intra-cluster case.

[1]Code is available at https://github.com/jasonshaoshun/SAL.

closer, implying a higher spread of each profession representations after SAL (§6.2.2).

In addition, we overcome the **linear removal limitations** of SAL and previous work by using eigenvalue decomposition of **kernel matrices** to obtain projections into directions with reduced covariance in the kernel feature space. We refer to this method as **kSAL** (for kernel SAL).

SAL outperforms the recent method of Ravfogel et al. (2020) aimed at solving the same problem, and is able to remove guarded information much faster while retaining better performance for the main task. Further experiments demonstrate that our method performs well even when the available data for the protected attributes is limited.

2 Problem Formulation and Notation

For an integer n we denote by $[n]$ the set $\{1, \ldots, n\}$. For a vector $\mathbf{v}$, we denote by $||\mathbf{v}||_2$ its ℓ_2 norm. Matrices and vectors are in boldface font (with uppercase or lowercase letters, respectively). Random variable vectors are also denoted by boldface uppercase letters. For a matrix $\boldsymbol{A}$, we denote by $\boldsymbol{A}_j$ its jth column (or by $\boldsymbol{A}_{i:j}$ the matrix with columns $\boldsymbol{A}_k$ for $k = i, \ldots, j$). Vectors are assumed to be column vectors.

In our problem formulation, we assume three random variables: $\mathbf{X} \in \mathbb{R}^d$, $\mathbf{Y} \in \mathbb{R}$ and $\mathbf{Z} \in \mathbb{R}^{d'}$. Samples of $\mathbf{X}$ are the inputs for a classifier to predict corresponding samples of $\mathbf{Y}$. The random vector $\mathbf{Z}$ represents the guarded attributes. We want to maximize the ability to predict $\mathbf{Y}$ from $\mathbf{X}$, while minimizing the ability to predict $\mathbf{Z}$ from $\mathbf{X}$. Without loss of generality, we assume that the mean values of $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$ are 0, and that $d' \leq d$.[2]

We assume n samples of $(\mathbf{X}, \mathbf{Y}, \mathbf{Z})$, denoted by $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)})$ for $i \in [n]$. These samples are used to train the classifier to predict the target values (y) from the inputs (x). These samples are also used to remove the information from the inputs based on the guarded attributes (z).

3 Erasing Principal Directions

We describe SAL in this section. We explain the use of SVD on cross-covariance matrices (§3.1) and describe the core algorithm in §4.1.

[2]For example, $\mathbf{Z}$ may be a multi-class label such as gender represented as a short vector over $\{-1, 1\}$ and $\mathbf{X}$ may be a complex input, which before removal of information about the guarded attribute $\mathbf{Z}$, can be used to predict $\mathbf{Z}$. An example of $\mathbf{X}$ would be an encoding of a post on a message board.

3.1 SVD on Cross-covariance Matrix

Let $\boldsymbol{A} = \mathbb{E}[\mathbf{X}\mathbf{Z}^\top]$, the matrix of cross-covariance between $\mathbf{X}$ and $\mathbf{Z}$. In that case, $\boldsymbol{A}_{ij} = \text{Cov}(\text{X}_i, \text{Z}_j)$ for $i \in [d]$ and $j \in [d']$.

A simple observation is that for any two vectors $\mathbf{a} \in \mathbb{R}^d, \mathbf{b} \in \mathbb{R}^{d'}$, the following holds due to the linearity of expectation:

$$\mathbf{a}\boldsymbol{A}\mathbf{b}^\top = \text{Cov}(\mathbf{a}^\top\mathbf{X}, \mathbf{b}^\top\mathbf{Z}). \tag{1}$$

This motivates the use of the cross-covariance matrix to find the so-called principal directions: directions in which the projection of $\mathbf{X}$ and $\mathbf{Z}$ maximize their covariance, where the projections are represented as two matrices $\boldsymbol{U} \in \mathbb{R}^{d\times d}$ and $\boldsymbol{V} \in \mathbb{R}^{d'\times d'}$. Each column in these matrices plays the role of the vectors $\mathbf{a}$ and $\mathbf{b}$ in Eq. 1. More specifically, we find $\boldsymbol{U}$ and $\boldsymbol{V}$ such that for any $i \in [d']$ it holds that:

$$\text{Cov}(\boldsymbol{U}_i^\top\mathbf{X}, \boldsymbol{V}_i^\top\mathbf{Z}) = \max_{(\mathbf{a},\mathbf{b})\in\mathcal{O}_i} \text{Cov}(\mathbf{a}^\top\mathbf{X}, \mathbf{b}^\top\mathbf{Z}),$$

where $\mathcal{O}_i$ is the set of pairs of vectors $(\mathbf{a}, \mathbf{b})$ such that $||\mathbf{a}||_2 = ||\mathbf{b}||_2 = 1$, $\mathbf{a}$ is orthogonal to $\boldsymbol{U}_1, \ldots, \boldsymbol{U}_{i-1}$ and similarly, $\mathbf{b}$ is orthogonal to $\boldsymbol{V}_1, \ldots, \boldsymbol{V}_{i-1}$.

It can be shown that such maximization can be done by applying the SVD on $\boldsymbol{A}$ such that $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where $\boldsymbol{U} \in \mathbb{R}^{d\times d}$, $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d'}$ and $\boldsymbol{V} \in \mathbb{R}^{d'\times d'}$. In the case of SVD, $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthonormal matrices, and $\boldsymbol{\Sigma}$ is a diagonal matrix with non-negative values on the diagonal. We let the vector of singular values on the diagonal of $\boldsymbol{\Sigma}$ be denoted by $\sigma_1, \ldots, \sigma_{d'}$.

Once the orthogonal matrices in the form of $\boldsymbol{U}$ and $\boldsymbol{V}$ are found, one can truncate them (for example, use only a subset of the columns of $\boldsymbol{U}$, represented as the semi-orthonormal matrix $\hat{\boldsymbol{U}}$) to use, for example, $\hat{\boldsymbol{U}}^\top\mathbf{X}$, as a representation (linear projection) of $\mathbf{X}$ which co-varies the most with $\mathbf{Z}$.

We suggest that rather than using the largest singular value vectors in $\boldsymbol{U}$ to project $\mathbf{X}$, we should project $\mathbf{X}$ using the principal directions with the **smallest** singular values. This means we find a representative of $\mathbf{X}$ that co-varies the *least* with $\mathbf{Z}$, essentially removing the information from $\mathbf{X}$ that is most related to $\mathbf{Z}$ and can be detected through covariance.

In addition, once such a projection matrix $\overline{\boldsymbol{U}}$ is calculated, we can use the projection $\overline{\mathbf{X}} = \overline{\boldsymbol{U}}\overline{\boldsymbol{U}}^\top\mathbf{X}$ such that the value of $\mathbb{E}[||\mathbf{X} - \overline{\mathbf{X}}||_2]$ is minimized, while removing the information from $\mathbf{X}$.[3] This allows us to potentially use the new projected values of the input random variable $\mathbf{X}$ *without* changing a classifier that was originally trained on samples from $\mathbf{X}$, though as we see in §6, using the projected input as-is without retraining the classifier may lead to performance issues with our method and other methods as well.

4 Connection to CCA and PCA

We describe connections to other matrix factorization methods.

How is SAL related to Canonical Correlation Analysis? The use of SVD on the cross-covariance matrix is very much related to the technique of Canonical Correlation Analysis (CCA), in which projections of $\mathbf{X}$ and $\mathbf{Z}$ are found such that they maximize the cross-correlation between these two random vectors. Rather than applying SVD on the cross-correlation matrix (CCA), we apply it on the cross-covariance matrix to preserve the scale of $\mathbf{X}$ in our projection.

How is SAL related to Principal Component Analysis? The use of SVD on the cross-covariance matrix is reminiscent of Principal Component Analysis (PCA), in which eigenvalue decomposition is applied on $\mathbb{E}[\mathbf{X}\mathbf{X}^\top]$ to reduce the dimensionality of $\mathbf{X}$. However, PCA does not reduce the dimensionality of $\mathbf{X}$ while removing information present in the guarded r.v. $\mathbf{Z}$. Rather, it finds projection of $\mathbf{X}$ in which the covariance of linear combination of $\mathbf{X}$ with itself is maximized.

In all three cases of CCA, PCA and LSA (Latent Semantic Analysis; Dumais 2004), SVD or eigenvalue decomposition is used with the aim of *maximizing* the correlation or covariance between one or two random vectors. In our case, the SVD is used to *minimize* the covariance between projections of $\mathbf{X}$ and $\mathbf{Z}$.

4.1 The SAL Algorithm

Our algorithm (SAL) follows the following procedure. First, the empirical cross-covariance matrix, estimating $\mathbb{E}[\mathbf{X}\mathbf{Z}^\top]$ is calculated:

$$\mathbf{\Omega} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}^{(i)}(\mathbf{z}^{(i)})^\top. \qquad (2)$$

[3]This can be formalized using the min-max theorem of linear algebra, also referred to as the Courant–Fischer–Weyl min-max principle.

SVD is then performed on $\mathbf{\Omega}$ to obtain $(\boldsymbol{U}, \mathbf{\Sigma}, \boldsymbol{V})$. We choose an integer value k and define $\overline{\boldsymbol{U}} = \boldsymbol{U}_{(k+1):d}$. The value of k bounded by the rank of $\mathbf{\Omega}$. The rank of $\mathbf{\Omega}$ is bounded from above by d and d', the dimensions of the vectors of $\mathbf{X}$ and $\mathbf{Z}$.

Then, the vectors $\mathbf{x}^{(i)}$ are projected using either $\overline{\boldsymbol{U}}^\top$ or $\overline{\boldsymbol{U}\boldsymbol{U}}^\top$. The latter projection attempts to project $\mathbf{x}^{(i)}$ to the original dimensionality and space after removing the information. More specifically, $\overline{\boldsymbol{U}\boldsymbol{U}}^\top$ is a projection matrix to the range of $\mathbf{\Omega}$.

The criterion we use to choose k is based on the singular values in $\mathbf{\Sigma}$. More specifically, we choose a threshold $\alpha \geq 1$ and choose the minimal k such that $\mathbf{\Sigma}_{11}/\mathbf{\Sigma}_{k+1,k+1} > \alpha$.

5 Kernel Extension to SAL

To enrich the type of information that is detected as co-varying, it is possible to use two feature functions, $\phi \colon \mathbb{R}^d \to \mathbb{R}^m$ and $\psi \colon \mathbb{R}^{d'} \to \mathbb{R}^{m'}$, and apply the procedure in §3 on $\mathbb{E}[\phi(\mathbf{X})(\psi(\mathbf{Z}))^\top]$. In that case, we can erase the information from $\phi(\mathbf{X})$ and treat it as the input for further classification. If the classifier is already learned, it would have to take input vectors of the form $\phi(\mathbf{X})$, otherwise it can be re-trained with the erased inputs.

5.1 The Kernel Trick

The kernel trick refers to learning and prediction without explicitly representing $\phi(\mathbf{x})$ or $\psi(\mathbf{z})$. Rather than that, we assume two kernel functions, $K_\phi(\mathbf{x}, \mathbf{x}')$ and $K_\psi(\mathbf{z}, \mathbf{z}')$ that calculate similarities between two xs or between two zs.

Every kernel that satisfies the necessary properties can be shown to be a dot product in some feature space. This means that for a given kernel function $K_\phi(\mathbf{x}, \mathbf{x}')$ it holds that

$$K_\phi(\mathbf{x}, \mathbf{x}') = \langle \phi(x), \phi(x') \rangle, \qquad (3)$$

for some ϕ function and similarly for $K_\psi(\mathbf{z}, \mathbf{z}')$. Masking learning and prediction through a kernel function is often useful when the feature representations ϕ and ψ are hard to explicitly compute, for example, because $m = \infty$ or $m' = \infty$ (such as the case with the Radial Basis Function, RBF, kernel).

We show next that the kernel trick can be used to generalize SAL to nonlinear information removal.

5.2 Removal with the Kernel Trick

Rather than assuming a set of examples in the form mentioned in §2, we assume we are given as input two kernel matrices of dimension $n \times n$:

$$[\boldsymbol{K}_\phi]_{ij} = K_\phi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}),$$
$$[\boldsymbol{K}_\psi]_{ij} = K_\psi(\mathbf{z}^{(i)}, \mathbf{z}^{(j)}).$$

In addition, for the justification of our algorithm, we define the following two feature matrices based on the kernel feature functions:

$$\forall i \in [m], j \in [n] \qquad [\mathbf{\Phi}]_{ij} = \phi(\mathbf{x}^{(j)})_i,$$
$$\forall i \in [m'], j \in [n] \qquad [\mathbf{\Psi}]_{ij} = \psi(\mathbf{x}^{(j)})_i.$$

Note that these two matrices are never calculated explicitly. Given the definition of the kernel as a dot product in the feature space (Eq. 3), it can be shown that $\boldsymbol{K}_\phi = \mathbf{\Phi}^\top \mathbf{\Phi}$ and $\boldsymbol{K}_\psi = \mathbf{\Psi}^\top \mathbf{\Psi}$. In addition, we slightly change the empirical cross-covariance matrix $\mathbf{\Omega}$ definition in Eq. 2 to: $\mathbf{\Omega} = \mathbf{\Phi}\mathbf{\Psi}^\top$. (This means we ignore the constant $1/n$ in the above definition of $\mathbf{\Omega}$ that normalizes the matrix with respect to the number of examples. This does not change the nature of the following discussion, but makes it simpler.) At this point, the question is how to perform SVD on $\mathbf{\Omega}$ without ever accessing directly the feature functions. This is where spectral theory of matrices comes in handy.

More specifically, it is known that the left singular vectors of $\mathbf{\Omega}$ ($\boldsymbol{U}$) are the eigenvectors of $\mathbf{\Omega}\mathbf{\Omega}^\top$. In addition, the singular values of $\mathbf{\Omega}$ correspond to the square-root values of the eigenvalues of $\mathbf{\Omega}\mathbf{\Omega}^\top$.

In addition, we show in Appendix A why an eigenvector $\mathbf{w}$ of $\mathbf{\Gamma} = \boldsymbol{K}_\phi \boldsymbol{K}_\psi$ can be transformed to an eigenvector of $\mathbf{\Omega}\mathbf{\Omega}^\top$ by multiplying $\mathbf{w}$ on the left by $\mathbf{\Phi}$ and calculating $\mathbf{\Phi}\mathbf{w}$.

With this fact in mind, we are now ready to find the left singular vectors of $\mathbf{\Omega}$ by finding the eigenvalues of $\mathbf{\Gamma}$, a matrix which is solely based on the kernel functions of $\mathbf{x}$ and $\mathbf{z}$.

Let $\mathbf{w}_1, \ldots, \mathbf{w}_k$ be eigenvectors of $\mathbf{\Gamma}$ and let $\mathbf{w}'_1, \ldots, \mathbf{w}'_k$ be the orthonormalization of $\mathbf{w}_i$, $i \in [k]$ based on the inner product $\langle \mathbf{w}_i, \mathbf{w}_j \rangle = \mathbf{w}_i \boldsymbol{K}_\phi \mathbf{w_j}^\top$. If we denote by $\boldsymbol{W}$ the matrix such that $\boldsymbol{W}_j = \mathbf{w}'_j$ for $j \in [k]$, then $\Phi \boldsymbol{W} = \boldsymbol{U}$ where $\boldsymbol{U}$ is the left singular vector matrix of $\mathbf{\Omega}$. Then,

$$\boldsymbol{U}^\top \phi(\mathbf{x}) = (\boldsymbol{W}^\top \mathbf{\Phi}^\top)\phi(\mathbf{x}) = \boldsymbol{W}^\top \kappa(\mathbf{x}), \quad (4)$$

where $\kappa(\mathbf{x})$ is a function that returns a vector of length n such that $[\kappa(\mathbf{x})]_j = K(\mathbf{x}^{(j)}, \mathbf{x})$. Eq. 4 shows we can calculate the projection of $\phi(\mathbf{x})$ while removing the information in $\psi(\mathbf{z})$ by using the smallest eigenvalue eigenvectors of $\mathbf{\Gamma}$ and kernel calculations of each training example with $\mathbf{x}$.

5.3 Practical Kernel Removal

Using the kernel algorithm as above may lead to issues with tractability, as it possibly requires calculating the full eigenvector matrix of a large matrix (the product of two kernel matrices). We propose an alternative algorithm (kSAL) for the kernel case which is more tractable.

For a fixed $0 \le k \le n$ (which does not need to be larger than the rank of either kernel matrices), we compute only the top k eigenvectors of $\mathbf{\Gamma}$. We then compute an orthonormal basis for the null space of the matrix $(\boldsymbol{K}_{\phi,1/2}\boldsymbol{W}_{1:k})^\top$ where $\boldsymbol{K}_{\phi,1/2} = \boldsymbol{U}_\phi \mathbf{\Sigma}_\phi^{1/2} \boldsymbol{V}_\phi^\top$, with $(\boldsymbol{U}_\phi, \mathbf{\Sigma}_\phi, \boldsymbol{V}_\phi)$ being the SVD of $\boldsymbol{K}_\phi$. Practically, this means we find a matrix $\boldsymbol{L}_\phi \in \mathbb{R}^{n\times(n-d)}$ such that $\boldsymbol{L}_\phi^\top \boldsymbol{L}_\phi = I$ and that $||(\boldsymbol{K}_{\phi,1/2}\mathbf{\Gamma})^\top \boldsymbol{L}_\phi||_2 \approx 0$. The final data points $\hat{\mathbf{x}}^{(j)}$ we use further down the pipeline correspond to the rows of $\boldsymbol{K}_{\phi,1/2}\boldsymbol{L}_\phi \in \mathbb{R}^{n\times(n-k)}$. If we are interested in using directly the reduced kernel matrix for the input vectors, we can use

$$\hat{\boldsymbol{K}}_\phi = \boldsymbol{K}_{\phi,1/2}\boldsymbol{L}_\phi \boldsymbol{L}_\phi^\top \boldsymbol{K}_{\phi,1/2}^\top. \quad (5)$$

Time Complexity Absorbing the kernel function computation as a constant, computing the kernel matrices is $\mathcal{O}(n^2)$ and their product $\mathbf{\Gamma}$ in $O(n^\omega)$ for $\omega < 2.808$ using Strassen's algorithm, but can be done much more efficiently when $\boldsymbol{K}_\psi$ is sparse, as normally expected. Calculating the top k eigenvectors of $\mathbf{\Gamma}$, has a cost of $\mathcal{O}(nk^2 + k^3)$ using, for example, the Arnoldi method.[4] In §6.4, we report clock running time for the kernel method.

Below, we experiment with RBF kernels (where $K_\phi(\mathbf{x}, \mathbf{x}') = \exp(-\gamma||\mathbf{x} - \mathbf{x}'||_2^2)$; we use $\gamma = 0.1$) and polynomial kernel of degree 2 (where $K_\phi(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^\top \mathbf{x}')^2$). The kernel of $\mathbf{z}$ remains linear (dot product).

6 Experiments

In our experiments, our main comparison algorithm is the iterative null space projection (INLP) algorithm of Ravfogel et al. (2020), which aims at solving an equivalent problem to ours. For the word embedding debiasing and fair classification (both

[4]For example, Matlab implements a variant of the Arnoldi method for its function `eigs`.

	SL	WS-S	WS-R	Mturk
Before	0.37	0.69	0.6	0.68
After	↑0.02 0.39	↑0.01 0.7	0.6	↑0.01 0.69

Table 1: The semantic evaluation of word embeddings before and after removing gender bias.

setups), we follow the experimental settings of Ravfogel et al. (2020).[5] SAL provides linear guarding, similarly to INLP, while kSAL also captures nonlinear regularities with respect to $\mathbf{Z}$ (one-hot vector). We can provide such guarding for representations of state-of-the-art encoders (such as BERT), provided the representations are eventually fed into a classifier for prediction. The protected attributes we experiment with are *gender* and *race*. Appendix D provides details about the datasets' splits and SAL run-time performance.

Datasets For debiasing word embeddings (§6.1), we use 7,500 male and female associated words, 15K words overall. The dataset train/validation/test split sizes are (49%/21%/30%). All the splits are balanced, i.e., containing an equal amount of male and female associated words. For the fair sentiment classification task (§6.2), we use 10K training examples across all authors' ethnicity ratios (0.5, 0.6, 0.7, and 0.8). All training sets have an equal amount of positive and negative sentiment examples. The test set is balanced for both sentiment and authors' ethnicity labels. For the profession classification task (§6.2.2) the data train/validation/test split sizes are (65%/10%/25%) and all the splits combined contain 115K samples.

6.1 Word Embedding Debiasing

Word embeddings are often prone to encoding biases in various ways (see §7). We evaluate our methods on gender bias removal from GloVe word embeddings. We use the 150,000 most common words and discard the rest. We sort the embeddings by their projection on the $\overrightarrow{\text{he}}\text{-}\overrightarrow{\text{she}}$ direction. Then we consider the top 7,500 word embeddings as male-associated words ($z = 1$) and the bottom 7,500 as female-associated words ($z = -1$).

Results with SAL A linear classifier can perfectly predict the guarded gender attribute when trained on out-of-the-box GloVe embeddings. Removing the first direction ($k = 1$) does not affect accuracy demonstrated in Figure 2. For $k = 2$, the performance drops to 50.2%, almost a random guess.

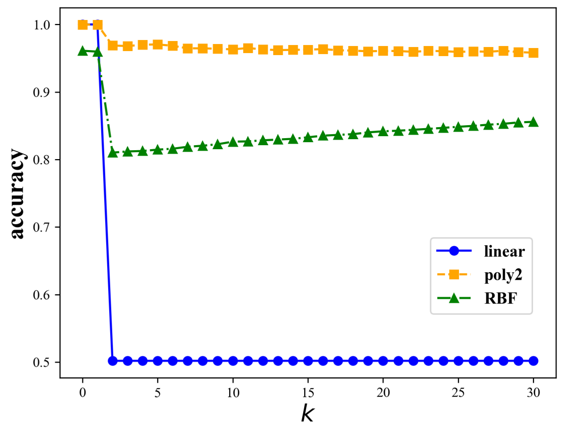

Figure 2: A classifier accuracy for gender prediction as a function of the number of principal directions that are linearly removed. For the linear classifier, we use a linear SVM. For the nonlinear classifiers, we use SVM with the polynomial kernel and with the RBF kernel.

We further perform intrinsic semantic tests to ensure the debiased embeddings remain useful. We use SimLex-999, WordSim353, and Mturk771 (similarity and relatedness datasets) to calculate the correlation between cosine similarities of the word embeddings to the human-annotated similarity score (Hill et al., 2015; Finkelstein et al., 2001; Halawi et al., 2012). We observed minor improvements for all tests when using debiased embeddings (Table 1), suggesting that our method keeps the embeddings intact. We also report the three most similar words (nearest neighbors) for ten random words before and after SAL (see Appendix C). We observe almost no change between the two sets of embedding results.

SAL debiasing does not provide a nonlinear information removal. In Figure 2 we plot the performance of nonlinear classifiers in the prediction of the **linearly-guarded** attribute (gender) as a function of the number of removed directions. We also provide linear classifier results for reference. We see that even after removing up to 30 principal directions, (linear) SAL is not sufficient for nonlinear classifiers – the gender can still be predicted. This finding is also noted by Ravfogel et al. (2020), who did not offer a direct solution. This finding partially motivates our development of kSAL.

Kernel Debiasing All three kernels achieve high gender prediction accuracy when no information is removed ($k = 0$), with accuracy of 100%, 99.9%

[5]We use the authors' implementation for both the INLP method and the experimental settings: `https://github.com/shauli-ravfogel/nullspace_projection`.

and 95.7% for the linear, polynomial, and RBF kernel, respectively. While the performance of the linear and polynomial kernels is not affected by removing one principal direction ($k = 1$), the RBF kernel accuracy drops to 86.3%. With $k = 2$, performance drops to 50.2%, 44.5% and 50.2% for the linear, polynomial, and RBF kernel, respectively, under **nonlinear kernel** removal. Compared to Figure 2 with SAL, we see kSAL effectively removes nonlinear information.

Deviations of Reduced Kernel from Original Kernel To quantitatively test whether the embeddings retain their geometric form when removing gender information, we compare the standard deviation (ρ) of the values in $\boldsymbol{K}_\phi$ to the average deviation (γ) of values of $\boldsymbol{K}_\phi$ from the corresponding values in $\hat{\boldsymbol{K}}_\phi$ (Eq. 5). When removing two principal directions, the largest approximation difference is seen in the linear kernel, with $\gamma/\rho = 0.64$. For the polynomial kernel, we observe $\gamma/\rho = 0.52$. For RBF, we have $\gamma/\rho = 0.16$.

6.2 Fair Classification

To further evaluate our method on downstream tasks, we follow fair classification tests of social media text.

6.2.1 Fair Sentiment Analysis

Task and Data The first task is sentiment analysis for social network users' posts. We use the TwitterAAE dataset (Blodgett et al., 2016), which contains users' tweets ($\mathbf{x}$), coupled with the users' ethnic affiliations ($\mathbf{z}$), and a binary label for the sentiment the tweet conveys ($\mathbf{y}$). The dataset splits the users into two groups, African American English (AAE) speakers and Standard American English (SAE) speakers. As users' privacy makes it hard to obtain ground truth labels for ethnic affiliation, the dataset uses the demographics of the neighborhoods the users live in as a proxy. Following NIO2020, we use the encoder of Felbo et al. (2017), DeepMoji, to obtain the tweets representation. DeepMoji is suitable for our goal, as it has been shown to encode demographic information, and therefore, might lead to unfair classification (Elazar and Goldberg, 2018).

We experiment with four different setups. The dataset consists of an equal amount of positive and negative sentiment examples for all of them. The datasets differ with respect to the guarded attribute ratio. For example, a ratio of 0.8 means that 0.8 of the positive and negative sentiment class examples are composed of AAE speakers, and 0.2 is composed of SAE speakers. We experiment with ratios of 0.5, 0.6, 0.7 and 0.8. The larger the ratio, the higher the classifier's tendency to make use of protected attributes to make its prediction.

Evaluation Measures We report the accuracy of the methods on the sentiment analysis task. To measure fairness, we use the difference in true positive rate (TPR-gap) between individuals belonging to different guarded attributes groups (Hardt et al., 2016; Ravfogel et al., 2020). The rationale behind TPR-gap is that for an equal opportunity, a positive outcome must be independent of the guarded attribute ($\mathbf{z}$), conditional on ($\mathbf{y}$) being an actual positive. See Hardt et al. (2016) for more details.

Results Table 2 presents our results for the fair sentiment classification. For the first three ratios, 0.5, 0.6, and 0.7, we can see that both SAL ($k = 1, 2$) and INLP maintain most of the main-task performance. In debiasing (TPR-Gap), SAL with $k = 2$ significantly outperforms INLP. As expected, removing two directions results in better debiasing than removing one, but it does not lead to a performance drop on the main task. While for the last ratio, 0.8, INLP achieves the highest TPR-gap result, it comes at the cost of a sharp performance drop on the main task, resulting in a nearly random classifier. SAL ($k = 1, 2$) maintains most of the main-task performance, and for $k = 2$, the TPR-gap is halved.

6.2.2 Fair Profession Classification

Task and Data The second task is profession classification. De-Arteaga et al. (2019) attempt to quantify the bias in automatic hiring systems and show that even for a simple task, predicting candidate profession based on a self-provided short biography, significant gaps result from the writer's gender. This might influence the open positions an automatic system will recommend to a candidate, thus favoring candidates from one gender over the other. We hence follow the setup of De-Arteaga et al. (2019), who experiment with professions classification ($\mathbf{y}$), from short biographies ($\mathbf{x}$), and gender as a guarded attribute ($\mathbf{z}$). We use a multiclass classifier to predict the profession, as there are 28 profession classes. We experiment with two types of text representations, FastText (Joulin et al., 2016), based on bag of word embeddings (BWE) and BERT (Devlin et al., 2018) encodings.

	Sentiment				TPR-Gap			
Rt	Orig.	INLP	SAL, $k=1$	SAL, $k=2$	Orig.	INLP	SAL, $k=1$	SAL, $k=2$
0.5	0.76	0.76	0.76	0.76	0.14	↓0.02 0.12	0.14	↓0.03 0.11
0.6	0.75	0.75	0.75	0.75	0.22	↓0.03 0.19	0.22	↓0.13 0.09
0.7	0.74	0.74	0.74	0.74	0.31	↓0.05 0.26	0.31	↓0.15 0.11
0.8	0.72	↓0.2 0.52	0.72	0.72	0.40	↓0.39 0.01	↓0.04 0.36	↓0.22 0.18

Table 2: The sentiment analysis scores (we use accuracy, as the dataset is balanced) and TPR differences (lower is better) as a function of the ratio of tweets (Rt) written by black individuals and conveying positive sentiment. Arrows with numbers indicate absolute increase/decrease from the baseline, and their background color indicates a difference with positive implications (green) or negative ones (red).

	Accuracy (profession)				TPR-Gap (RMS)			
Encoder	Orig.	INLP	SAL, $k=1$	SAL, $k=2$	Orig.	INLP	SAL, $k=1$	SAL, $k=2$
FastText	0.75	↓0.05 0.71	↑0.01 0.76	↑0.01 0.76	0.20	↓0.11 0.09	↓0.02 0.18	↓0.08 0.12
BERT	0.8	↓0.11 0.69	↓0.02 0.78	↓0.02 0.78	0.21	↓0.15 0.06	↓0.04 0.17	↓0.12 0.09

Table 3: The profession classification on the biographies dataset results. We report accuracy and TPR-RMS. The number of classes is 28.

Evaluation Measures We report accuracy for the profession classification. For bias level measurement, we use a generalization of TPR-gap for multi-class, suggested by De-Arteaga et al. (2019), calculating the root mean square (RMS) of the TPR with respect to all classes.

De-Arteaga et al. (2019) also provided evidence for a strong correlation between TPR-gap and existing gender imbalances in occupations, which may lead to unfair classification.

Results Table 3 presents the profession classification results. Similar to the sentiment analysis task, SAL ($k = 1, 2$) maintains most of the main-task performance, and for $k = 2$, the two-direction removal, the TPR-gap is lower. When comparing SAL ($k = 2$) to INLP, we observe a clear trade-off between maintaining the main task performance (SAL, $k = 2$) and low TPR-gap scores (INLP).

6.3 Scarce Protected Attribute Labels

For many real-world applications, obtaining large amounts of labeled data for protected attributes can be costly, labor-intensive, and in some cases, infeasible due to an ever-increasing number of privacy regulations. In this analysis, we stress-test our algorithm by simulating a scenario in which only a limited amount of samples from the main task are coupled with the desired protected attribute labels. For this purpose, we replicate the fair sentiment classification experiments, but this time, feeding only a fraction of the annotated data to our debiasing method. In terms of the main task, the experiment is identical, i.e., we use 100K samples for training the sentiment classifier. We experiment with different fractions of the debiasing data, i.e., 5%, of the sentiment training data contains labels about the protected attribute. We hence feed 5,000 samples for debiasing. The subsets for debiasing are chosen randomly. We repeat each experiment 10 times with different subsets. Table 4 presents our results. Using a small fraction of the data for debiasing did not significantly affect SAL's ($k = 1, 2$) main-task performance. INLP, on the other hand, suffers from a sharp performance decrease, resulting in a near-random sentiment classifier. SAL's ($k = 1, 2$) ability to debias the data is slightly worse than in the complete dataset setting but the resulting representations are still significantly less biased than the original ones. INLP achieves low TPR gaps, but it is hard to determine if this is due to an accurate bias removal or a result of corrupting the representations.

6.4 Kernel Experiments

Despite their flexibility in modeling rich feature functions, kernels have been documented to be computationally intensive. Lack of computational resources prevented us from using the full sentiment and bios datasets for our kernel experiments, and instead, we use $15,000$ training examples and $7,998$ test set examples (the full test set) for the sentiment dataset and $15,000$ training examples and $5,000$ examples for the profession dataset. For training on the acquired $15,000$ training examples, we used one Intel Xeon E5-2407 CPU, running at 2.2 GHz, for approximately five hours (for a time complexity analysis, see §5.3).

Table 5 shows that using only a small subset of the data, kSAL-poly2 reduces the TPR gaps

	Sentiment				TPR-Gap			
Ratio	Orig.	INLP	SAL, $k=1$	SAL, $k=2$	Orig.	INLP	SAL, $k=1$	SAL, $k=2$
0.5	0.76	$\downarrow$0.19 0.57	0.76	0.76	0.14	$\downarrow$0.12 0.02	$\uparrow$0.27 0.41	$\downarrow$0.03 0.11
0.6	0.75	$\downarrow$0.16 0.59	0.75	0.75	0.22	$\downarrow$0.19 0.03	$\uparrow$0.01 0.23	$\downarrow$0.13 0.09
0.7	0.74	$\downarrow$0.17 0.57	$\downarrow$0.01 0.73	$\downarrow$0.01 0.73	0.31	$\downarrow$0.26 0.05	0.31	$\downarrow$0.19 0.12
0.8	0.72	$\downarrow$0.15 0.57	$\downarrow$0.01 0.71	$\downarrow$0.01 0.71	0.40	$\downarrow$0.32 0.08	$\downarrow$0.08 0.34	$\downarrow$0.27 0.17

Table 4: The sentiment analysis experiments, 100K samples are use to train the sentiment classifier, but only 5K examples are used for learning to remove bias. The test set is identical to the one used in §6

Sentiment Analysis (DeepMoji)				
	Main		TPR-Gap	
k	poly2	rbf	poly2	rbf
0	0.75	0.76	0.14	0.15
1	0.75	0.76	0.14	0.15
2	0.75	0.76	$\downarrow$0.01 0.13	$\downarrow$0.03 0.12
Profession Classification (BERT)				
	Main		TPR-Gap (RMS)	
k	poly2	rbf	poly2	rbf
0	0.77	0.61	0.33	0.23
1	0.77	$\uparrow$0.07 0.68	$\downarrow$0.05 0.28	$\uparrow$0.11 0.34
2	0.77	$\uparrow$0.07 0.68	$\downarrow$0.08 0.25	$\uparrow$0.08 0.31

Table 5: Kernel results with kSAL for sentiment (for a ratio of 0.5) and profession classification.

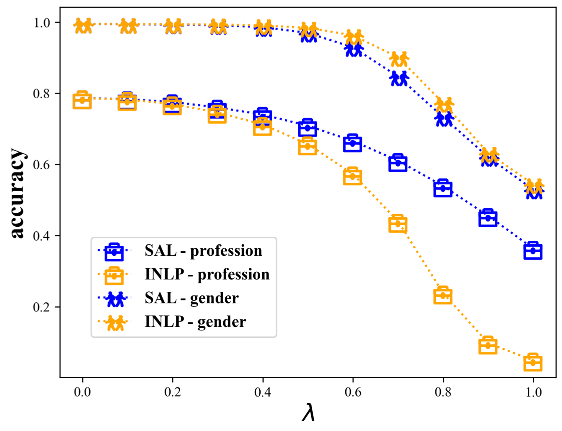

Figure 3: Gender and profession classifications as a function of the interpolation coefficient λ.

while maintaining almost identical performance to the original model on both the sentiment analysis and profession classification tasks. For the sentiment analysis task, kSAL-RBF slightly improves the main task results while reducing the TPR-gap (RMS). For the RBF profession classification task, the results are unexpected, with main task performance increasing as we remove principal directions. This could be due to the pruning of the rich, infinite feature space RBF kernel represents (we also observe significant overfitting with RBF).[6]

6.5 Perturbed Inputs

While the transformation through $\overline{\boldsymbol{U}\boldsymbol{U}}^\top$ maps $\mathbf{x}$ back into the original vector space (as a projection), it often turns out that it removes information in such a way that the *original* classifier (trained on data without removal) can no longer be used with the inputs after removal. This issue exists not only with our algorithm, but also with INLP, and indeed, like us, Ravfogel et al. (2020) re-trained their classifier after they created the cleaned projected inputs.

Ideally, we would want to remove information without necessarily having to retrain a classifier for the main task, as this is costly and perhaps unattainable. To test the effect of such an approach, we interpolated $\overline{\boldsymbol{U}\boldsymbol{U}}^\top$ with the identity matrix, to eventually project $\mathbf{x}$ using $\lambda\overline{\boldsymbol{U}\boldsymbol{U}}^\top + (1-\lambda)\boldsymbol{I}$ for $\lambda \in \{0, 0.1, \ldots, 1.0\}$. This approach weakens the impact of the removal projection and retains some of the information in $\mathbf{x}$. While an adversary can attack this approach,[7] it can mitigate the effects of privacy violations in cases where the service or software used with the modified representations cannot be retrained, especially if the service providers have no malicious intent.

Figure 3 describes an ablation experiment, ranging λ as above on the bios dataset. We see that as we increase the intensity of the use of the SAL projection (increasing λ), the accuracy of both gender prediction and profession prediction decrease when training the original classifier on the non-projected inputs. While the behavior is similar for the gender accuracy for both INLP and our method, the decrease for the profession prediction is much sharper for $\lambda > 0.4$ with INLP.

[6]With INLP, RBF-kernel SVM also obtains low-accuracy results.

[7]Consider that the matrix $\lambda\overline{\boldsymbol{U}\boldsymbol{U}}^\top + (1-\lambda)\boldsymbol{I}$ could be invertible for $\lambda < 1$.

Task	WED	FSC	FPCF	FPCB
SAL	0.03 sec	0.37 sec	0.16 sec	0.35 sec
INLP	50 sec	100 min	7 min	35 min

Table 6: A run-time comparison between SAL and INLP. We used 2.20GHz Intel Xeon E5-2407 CPU for all of the experiments. WED, FSC, FSCF, and FPCB stand for word embedding debiasing, fair sentiment classification, and fair profession classification (with both FastText and BERT based representations).

6.6 Runtime of SAL

We measure the time it takes both methods to learn a projection matrix for a given training set. Once we have a projection matrix, debiasing the data is done by multiplying the data representation matrix by the learned projection matrix. Since matrix multiplication is a common practice for many research disciplines, and both methods use it, we do not benchmark it as well. Table 6 presents the run-time differences between SAL and INLP. For all of the experiments, SAL runtime is smaller by at least three orders of magnitude than INLP runtime.

7 Related Work

In their influential work, Bolukbasi et al. (2016) revealed that word embeddings for many gender-neutral terms show a gender bias. Zhao et al. (2018) presented a customized training scheme for word embeddings, which minimizes the negative distances between words in the two groups, e.g., male and female related words, for gender debiasing. Gonen and Goldberg (2019) demonstrated that bias is remains deeply intertwined in word embeddings even after using the above methods. For example, they showed several methods that can accurately predict the gender associated with gender-neutral words, even after applying the methods mentioned above. Similar to Ethayarajh et al. (2019), they concluded that removing a small number of intuitively selected gender directions cannot guarantee the elimination of bias. Motivated by this conclusion, Ravfogel et al. (2020) presented iterative null space projection (INLP). This debasing algorithm iteratively projects features into a space where a linear classifier cannot predict the guarded attribute. The debiased representations are *linearly guarded*, i.e., they cannot guarantee bias removal beyond the linear level. Indeed, they show a simple nonlinear classifier can achieve high accuracy when predicting the guarded attribute. Their approach is also related to that of Xu et al. (2017). Previous work uses adversarial methods (Ganin et al., 2016) for information removal (Edwards and Storkey, 2015; Li et al., 2018; Coavoux et al., 2018; Elazar and Goldberg, 2018; Barrett et al., 2019; Han et al., 2021) with the one by Ravfogel et al. (2022) being related to ours through the use of the mini-max theorem with squared-error loss on the reconstruction of a matrix similar to our covariance matrix. In addition, methods based on similarity measures between neural representations (Colombo et al., 2022) were developed. To support the increasing interest in fair classification, Han et al. (2022) presented an open-source framework for standardizing the evaluation of debiasing methods. Finally, most relevant to this paper is an extension of SAL to the unaligned case, where protected attributes are not paired with input examples (Shao et al., 2023).

8 Conclusions

We presented a method for removing information from learned representations. We extended our method by using kernels, showing we can provide an effective nonlinear guarding. We also experimented with real-world low-resource situations, in which only a small guarded attribute dataset is provided for information removal.

Limitations

There are two main technical limitations to our work: (a) while the kernel removal is nonlinear, it still depends on a feature representation that captures a specific type of nonlinearities; (b) like other kernel methods, the kernel removal method is significantly slower than direct SVD removal in cases where the feature representations can be written out without the need of an implicit kernel. Future work may apply random projections to the kernel matrices to decompose them more efficiently.

A general limitation of current methods of information removal is that they are only able to remove information with respect to a specific class of classifiers. It could always be the case that complex correlations between the inputs and the guarded attributes exist, and that an adversary can try to exploit them to predict the guarded attribute if this class of classifiers is not too complex. Our use of kernels alleviates some of this issue, though not completely.

Finally, experimentally, we focus on text only in English. It is not clear to what extent our method generalizes to other languages in a useful manner, especially when morphology is rich, and the neural representations encode important information for the task at hand, but that information would be removed by our method.

Ethical Considerations

Public trust plays a significant role in the broad applicability of NLP in real-world scenarios, especially in critical situations that may directly impact people's lives. NLP research of the kind presented in this paper helps this issue take the spotlight it deserves. However, we discourage NLP practitioners from using our method (and similar methods) as an out-of-the-shelf solution in deployed systems. We recommend investing a significant amount of time and effort in understanding the applicability and universality of our method to the debiasing of representations. Issues such as expected type of adversariality or tolerance level for drop in system performance need to be considered.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. We especially appreciate the comment provided by one of the reviewers regarding our title. Particularly that it could be misinterpreted as an indication of frustration at rejections of our paper ("gold") in previous conferences. Rather, the "gold" in our case is the principal vectors with low intensity, which are pruned in most use cases of SVD. We also thank Shauli Ravfogel for providing support with the code for INLP, Ryan Cotterell for discussions and Matt Grenander for feedback on earlier drafts. The experiments in this paper were supported by compute grants from the Edinburgh Parallel Computing Center (Cirrus) and from the Baskerville Tier 2 HPC service (University of Birmingham).

References

Maria Barrett, Yova Kementchedjhieva, Yanai Elazar, Desmond Elliott, and Anders Søgaard. 2019. Adversarial removal of demographic attributes revisited. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6330–6335.

Su Lin Blodgett, Lisa Green, and Brendan O'Connor. 2016. Demographic dialectal variation in social media: A case study of african-american english. *arXiv preprint arXiv:1608.08868*.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29:4349–4357.

Maximin Coavoux, Shashi Narayan, and Shay B Cohen. 2018. Privacy-preserving neural representations of text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1–10.

Pierre Colombo, Guillaume Staerman, Nathan Noiry, and Pablo Piantanida. 2022. Learning disentangled textual representations via statistical measures of similarity. *arXiv preprint arXiv:2205.03589*.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. 2019. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 120–128.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Susan T Dumais. 2004. Latent semantic analysis. *Annual review of information science and technology*, 38(1):188–230.

Harrison Edwards and Amos Storkey. 2015. Censoring representations with an adversary. *arXiv preprint arXiv:1511.05897*.

Yanai Elazar and Yoav Goldberg. 2018. Adversarial removal of demographic attributes from text data. *arXiv preprint arXiv:1808.06640*.

Kawin Ethayarajh, David Duvenaud, and Graeme Hirst. 2019. Understanding undesirable word embedding associations. *arXiv preprint arXiv:1908.06361*.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. *arXiv preprint arXiv:1708.00524*.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. *arXiv preprint arXiv:1903.03862*.

Guy Halawi, Gideon Dror, Evgeniy Gabrilovich, and Yehuda Koren. 2012. Large-scale learning of word relatedness with constraints. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1406–1414.

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2021. Diverse adversaries for mitigating bias in training. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2760–2765.

Xudong Han, Aili Shen, Yitong Li, Lea Frermann, Timothy Baldwin, and Trevor Cohn. 2022. fairlib: A unified framework for assessing and improving classification fairness. *arXiv preprint arXiv:2205.01876*.

Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29:3315–3323.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Yitong Li, Timothy Baldwin, and Trevor Cohn. 2018. Towards robust and privacy-preserving text representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 25–30.

Thomas Manzini, Yao Chong Lim, Yulia Tsvetkov, and Alan W Black. 2019. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. *arXiv preprint arXiv:1904.04047*.

Ella Rabinovich, Yulia Tsvetkov, and Shuly Wintner. 2018. Native language cognate effects on second language lexical choice. *Transactions of the Association for Computational Linguistics*, 6:329–342.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. *arXiv preprint arXiv:2004.07667*.

Shauli Ravfogel, Michael Twiton, Yoav Goldberg, and Ryan D Cotterell. 2022. Linear adversarial concept erasure. In *International Conference on Machine Learning*, pages 18400–18421. PMLR.

Shun Shao, Yftah Ziser, and Shay B. Cohen. 2023. Erasure of unaligned attributes from neural representations. *arXiv preprint arXiv:2302.02997*.

Ke Xu, Tongyi Cao, Swair Shah, Crystal Maung, and Haim Schweitzer. 2017. Cleaning the null space: A privacy mechanism for predictors. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Jieyu Zhao, Yichao Zhou, Zeyu Li, Wei Wang, and Kai-Wei Chang. 2018. Learning gender-neutral word embeddings. *arXiv preprint arXiv:1809.01496*.

A Eigenvectors of Λ

We turn to the following Lemma used in §5.2.

Lemma 1. *Let* $\mathbf{w}$ *be an eigenvector associated with eigenvalue* $\lambda \in \mathbb{R}$ *for* $\mathbf{\Gamma} = \boldsymbol{K}_\phi \boldsymbol{K}_\psi$. *Then* $\mathbf{\Phi w}$ *is an eigenvector of* $\mathbf{\Omega\Omega}^\top$.

Proof. Since $\mathbf{w}$ is an eigenvector of $\mathbf{\Gamma}$, it holds that $\Gamma\mathbf{w} = \lambda\mathbf{w}$. Therefore:

$$\mathbf{\Psi}^\top \mathbf{\Psi} \mathbf{\Phi}^\top \mathbf{\Phi w} = \lambda \mathbf{w},$$
$$\left(\mathbf{\Phi}\mathbf{\Psi}^\top \mathbf{\Psi} \mathbf{\Phi}^\top\right) \mathbf{\Phi w} = \lambda \mathbf{\Phi w},$$

and therefore $\mathbf{\Phi w}$ is an eigenvalue of

$$\mathbf{\Omega\Omega}^\top = \mathbf{\Phi}\mathbf{\Psi}^\top \mathbf{\Psi}\mathbf{\Phi}^\top.$$

□

B Nearest Neighbors Test for Word Embedding Debiasing

We give in Table 7 the ten nearest neighbor words for ten random words from the data, before and after using SAL. The neighboring words are determined through cosine similarity of the corresponding embeddings with respect to the pivot word embedding. We observe little to no difference in these two lists (before and after the removal).

Words	Nearest neighbors (before)	Nearest neighbors (after)
lobbying	lobbyists, lobbyist, campaigning	lobbyists, lobbyist, campaigning
once	again, then, when	again, then, when
parliament	parliamentary, mps, elections	parliamentary, mps, elections
dashboard	dashboards, smf, powered	dashboards, smf, powered
cumulative	gpa, accumulative, aggregate	gpa, accumulative, aggregate
foam	rubber, mattress, polyurethane	rubber, mattress, polyurethane
rh	lh, bl, r	lh, bl, graphite
genetically	gmo, gmos, genetic	gmo, gmos, genetic
inner	outer, inside, innermost	outer, inside, innermost
harvest	harvesting, harvests, harvested	harvesting, harvests, harvested
secretary	deputy, minister, treasurer	deputy, minister, secretaries
ruth	helen, esther, margaret	helen, esther margaret
charlotte	raleigh, nc, atlanta	raleigh, nc, atlanta
abigail	hannah, lydia, eliza	hannah, lydia, samuel
sophie	julia, marie, lucy	julia, lucy, claire
nichole	nicole, kimberly, kayla	nicole, kimberly, mya
emma	emily, lucy, sarah	emily, watson, sarah
david	stephen, richard, michael	alan, stephen, richard
richard	robert, william, david	robert, william, david
joseph	francis, charles, thomas	mary, francis, charles
thomas	james, william, john	james, william, henry
james	john, william, thomas	william, john, thomas

Table 7: Nearest neighbor test on GloVe word embeddings before and after debiasing on gender. The upper block includes a random set of words, while the middle and bottom block include female and male names.