Bias in Artificial Intelligence Project

Finlay Boyle
Department of Natural Sciences
Durham University
Durham, United Kingdom
finlay.boyle@durham.ac.uk

***Abstract*—Natural language processing (NLP) combines artificial intelligence and linguistics to process and extract meaning from spoken or written natural language. A subset of NLP is sentiment analysis which is the processing of language and structure of a text to determine the emotions, and their strength, within the text. In this project I will explore the effect of bias on sentiment polarity across demographic groups specifically related to gender identity terms for males and females. The project focuses on the effect of reducing sentiment polarity within the word embedding vectors themselves to affect downstream applications related to valence prediction.**

***Index Terms*—embeddings, fairness, NLP**

I. Project Proposal

The project that I am going to develop will be focused on reducing sentiment polarity between demographic identities in word embeddings directly using a technique called adversarial learning as explored by [1] which I will loosely follow throughout my project.

To begin, I will need a set of word embedding vectors. I have chosen to use a pre-trained model provided by the Python library gensim [2] which has been trained on the Google News corpus using the word2vec technique described in [3] which was recommended by the paper that I am following. This will form the basis for any debiasing which will transform the vector space of word embeddings.

As detailed in [1] the first concrete task is to determine a positive and negative sentiment subspace within the vector space of word embeddings. To do this, I will use the sentiment lexicons provided by [4] to determine these subspaces. Then, using a technique called Principal Component Analysis (which defines orthogonal component axes using eigenvectors in the directions explaining the most variance within the vector subspace [5]) I will examine the explained variances within the subspaces and compute the directional sentiment vector connecting these which I will explore further in the progress report.

Then, I will be able to train the adversarial debiaser to reduce the sentiment polarity nested within the word embeddings. This is the primary section of the project which I will develop in much more depth in the progress section of this report where I will implement this and derive the gradients from the loss functions.

The next task will be to evaluate the effect of the debiasing by using the SemEval-2018 Task 1: Affect in Tweets data set [6] to train a Support Vector Regression (SVR) model to predict the valence - a measure of sentiment polarity - of text. Once this has been trained, I will finally evaluate the debiasing by using the Equity Evaluation Corpus [7] which consists of a variety of sentence templates using different noun phrases specific to each gender. The purpose of this will be to reduce the variance and average valence score difference between genders by training the SVR model on the SemEval data set with and without debiased word vector embeddings.

My project will be coded in Python 3.8.2 using gensim 4.0.1, scikit-learn 0.23.2, numpy 1.19.2, matplotlib, pandas 1.1.3 and scipy 1.5.3. I will implement the adversarial debiasing machine learning algorithm directly and use the SVR model provided by scikit-learn for evaluation purposes. The datasets have been discussed in details in the prior paragraphs and download links are provided within the code comments.

Sentiment is a concept that is closely related to the positivity and negativity conveyed in speech and text. Sentiment polarity is an important part of NLP to explore in relation to bias because machine learning is used heavily for automated moderation tasks such as removing toxic comments within forum boards, social media, and news websites. This could unintentional filter the narrative of conversations [CITATIONS] online by misinterpreting the sentiment of a sentence which could have social consequences in relation to important discussions, especially those taking place around the world at the moment [EXPAND], if minority voices are filtered out and suppressed by biased machine learning algorithms if they are biased against minorities and believe the comments are toxic when they are truly not. This is not limited to gendered phrases but it also expands to the sentiment encoded within these algorithms about African-American names versus European-American names which is also examined in [1].

II. Progress Report

A. Data Analysis

I have used a variety of data sets for different purposes throughout my project but I will focus on the Equity Evaluation Corpus (EEC) [7] for data analysis because I use this data set for the evaluation of my debiasing model. The data set can be downloaded from https://saifmohammad.com/WebPages/Biases-SA.html. The EEC is a collection of sentence templates that can be filled in with gender-associated words or race-associated words and were used in conjunction with the SemEval-2018 Task 1: Affect in Tweets data set [6]. Due to space limitations within

this document, I have set out how to download the additional data sets clearly in the comments of my code.

The data set consists of 8640 sentences with columns ID, Sentence, Template, Person, Gender, Race, Emotion, and Emotion word. As my project was focused on the gender-associated noun phrases I first filtered out all of the records that do not have a noun phrase in the Person column by using conditions on a Pandas data frame. The noun phrases are found in Table 3 of [7]. I decided not to keep gender-associated and race-associated first names because I wanted to isolate the impact of the bias in the word embeddings on gender-associated words only so that I can make conclusive statements about my project without doubts about whether race has also impacted the results. The methodology in [1] was also to only the noun phrases.

There were 2880 records remaining. The Race column is now obsolete however and consists solely of empty values so I drop this column from the data frame. As well as this I also drop the ID column and the Emotion column. This is because I do not need the IDs here because they are irrelevant for the task and they are simply there for ease of use of the data set. The Emotion column is redundant because I am able to deduce sentences that are the same structure but have different noun phrases using the remaining Template, Person, and Emotion Word columns.

This data cleaning leaves the Sentence, Template, Person, Gender, and Emotion word columns. The Template and Emotion word columns are categorical features and for ease of manipulation I use pandas to map these to integer category identifiers. For the Person column I matched the pairs manually and assigned them an index e.g. she/he was 0, her/him was 1 etc. A full list is contained within the code.

The final data transformation that I had to do to this data set was to convert the sentences to a vector representation. The premise laid out in [1] was to compute the average vector of each of the words within the sentence which I also used. This involved introducing 300 additional numeric features to the data frame. These represent the word2vec features which are meaningless on their own but when combined together provide a vector space describing how similar words, and sentences, are to each other.

B. Principal Component Analysis

Prior to debiasing I needed to compute the *directional sentiment vector* detailed in [1] which connects the positive and negative subspaces as determined by the sentiment lexicons provided by [4]. To find this vector I perform Principal Component Analysis on each subspace which is the process of finding orthogonal axes in the directions of the most significant uncorrelated variances [5] within the word embeddings.

As evidenced by Fig. 1, it is evident that the first principal component contains significantly more explained variance than the other components. The shape is similar to that found in Figure 2 of [4]. However, it was unclear what was on the y-axis in the paper and the chart had an unusual shape in that left graph is not monotone decreasing.

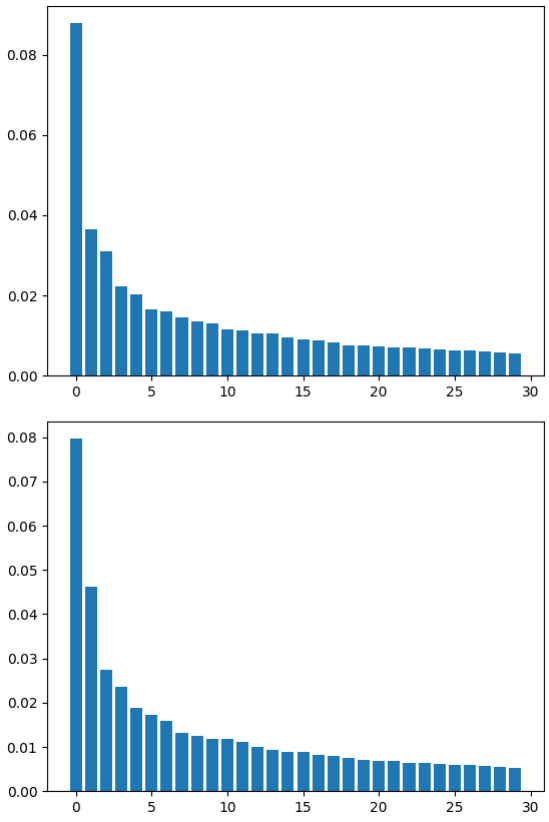

Fig. 1. The bar charts show the explained variance attributed to the top 30 principal components of each subspace. Both have significant variance attributed to the first component. **Top**: PCA of positive sentiment subspace. **Bottom**: PCA of negative sentiment subspace.

To compute the directional sentiment vector I took the most significant principal component of the positive subspace from the same vector from the negative subspace. To validate that this vector was suitable to encode the directional sentiment, I measure the accuracy and precision of classifying the sentiment lexicon words by projecting their vectors from the word embeddings on to the directional sentiment vector. If it is negative then I classify the word as having negative sentiment and if it is positive I classify it as having positive sentiment. This produces an accuracy of 88.2% and a precision of 90.2% suggesting that the directional sentiment vector is appropriate (compared to 91.3% in the original paper)

C. Adversarial Debiaser

After having computed the directional sentiment vector I am now ready to create the debiaser. The debiaser is a machine learning algorithm consisting of two objectives: the standard objective and the adversarial objective. This is important as while the standard objective focuses on reducing the sentiment polarity of a word vector the competing adversarial objective attempts to keep the word vector in a neighbourhood of the original position within the subspace. This is important to

ensure that reducing sentiment does not distort the meaning of the word.

This technique was originally presented in [8] which defines two separate sets of weights $\mathbf{W}$ and $\mathbf{U}$. $\mathbf{W}$ are the weights associated with reducing the sentiment and $\mathbf{U}$ are the weights associated with the adversarial objective. Each objective also comes with a separate loss function. First, I define $\mathbf{y}$ as the original word vector, $\mathbf{k}$ as the directional sentiment vector and $\hat{\mathbf{y}}$ as the debiased word vector calculated as:

$$\hat{\mathbf{y}} := \mathbf{y} - \mathbf{W}\mathbf{W}^\mathbf{t}\mathbf{y} \tag{1}$$

Then, the standard loss function, L_P, and the adversary loss function, L_A, defined as:

$$L_P := (\mathbf{y} - \hat{\mathbf{y}})^2 = (\mathbf{W}\mathbf{W}^\mathbf{t}\mathbf{y})^2 \tag{2}$$

$$L_A := (\mathbf{k}^\mathbf{t}\mathbf{y} - \mathbf{U}^\mathbf{t}\hat{\mathbf{y}})^2 \tag{3}$$

These are from [4] and I believe that L_P is a slight abuse of notation because L_P is a scalar field whereas the function above is actually a vector. This is because the square is representing mean-square distance so L_P is defined as:

$$L_P := \frac{1}{m}\sum_{i=1}^{m}([\mathbf{W}\mathbf{W}^\mathbf{t}\mathbf{y}]_i)^2 \tag{4}$$

where m is the number of features (i.e. 300) and $[\mathbf{P}]_i$ is the i$^{\text{th}}$ component of the vector $\mathbf{P}$. Then the objective to minimise is defined[1] as:

$$\nabla_\mathbf{W} L_P - \text{proj}_{\nabla_\mathbf{W} L_A}\nabla_\mathbf{W} L_P - \alpha\nabla_\mathbf{W} L_A \tag{5}$$

where $\nabla_\mathbf{P} Q$ is the gradient of Q (a scalar field) with respect to $\mathbf{P}$ (a vector) and $\text{proj}_\mathbf{P}\mathbf{R}$ is the vector projection of $\mathbf{R}$ on to $\mathbf{P}$. I have calculated[2] the j$^{\text{th}}$ component of the gradients and these are:

$$[\nabla_\mathbf{W} L_P]_j = \frac{2\mathbf{y}_j}{N}\sum_{i=1}^{m}([\mathbf{W}\mathbf{W}^\mathbf{t}\mathbf{y}]_i)^2 \tag{6}$$

$$[\nabla_\mathbf{W} L_A]_j = 2(\mathbf{U}.\hat{\mathbf{y}} - \mathbf{k}.\mathbf{y})((m+1)\mathbf{W}_j + N - 1) \tag{7}$$

and finally, we also need $\nabla_\mathbf{U} L_A$ to modify the U weights. This is calculated as:

$$[\nabla_\mathbf{U} L_A]_j = 2(\mathbf{U}.\hat{\mathbf{y}} - \mathbf{k}.\mathbf{y})\hat{\mathbf{y}}_j \tag{8}$$

To optimise the weights according to these objective and loss functions I have implemented mini-batch gradient descent using the Adam Optimiser [9] which is an optimisation algorithm to update the weight according to the gradients. I implemented both Adam and the mini-batch gradient descent using numpy. In my project I trained the adversarial debiaser on 40000 batches of 1000 word vectors from the word embeddings.

[1]there was a disagreement between the objective functions in [1] and [8], I chose to agree with the latter

[2]due to space limitations the proofs have been omitted

D. Valence Regression Results

Once the debiaser has been trained, I debias the entire word embedding vector space using equation (1) with the trained weights. I then use the SemEval-2018 Task 1: Affect in Tweets data set [6] valence training data set to train a Support Vector Regression (SVR) model [10] from scikit-learn [11]. To do this, I transform the valence data set into the valence score and the sentences in to averaged vector form.

Before evaluating the effect of the debiaser on the sentiment bias I present my findings on the accuracy of the model before and after debiasing by using the SemEval-2018 Gold-Standard Test Set from [6]. In [4], they use the Pearson's Correlation Coefficient (PCC) to determine the success of their model and achieve scores of 42% before debiasing and 43% after debiasing. I use the same metrics and before debiasing the SVR model has a PCC of 47.2% and after it has a PCC of 46.7% which indicates that there was a very slight decrease in the accuracy of the model. However, this is a tiny trade-off for the improvements in the bias reduction.

I evaluated the model on the EEC [7] by calculating the valence for sentences differing only by the gender associated with the noun phrase[3] in the sentence template.

Table I compares the valence bias before and after debiasing. The goal of debiasing is for the mean difference and variance to be closer to 0. This will result in less sentiment difference between the same sentences with different gender-associated noun phrases as well as less variance between sentences of the same type. It uses the same notation introduced in [4], [7] where **F**↑-↓**M** means the sentence valence delta was in favour of females and **F**↓-↑**M** means the sentence valence delta was in favour of males.

The debiasing successfully causes this by reducing the variance in both male and female biased sentences and it leads to a significant reduction in sentiment bias in favour of females by 44.7% and it also decreases the sentiment bias in favour of males by 27.8%. For **F**↑-↓**M** the variance decreases by 87.1% but for **F**↓-↑**M** the variance increases by 1.4%. Ultimately, there is a significant decrease in the net variance across the entire corpus which is a success.

These are similar results to the original paper [4] but they do differ slightly because they use the GloVe word embeddings [12] instead of the Google News trained word embeddings and ultimately the training of the debiaser is stochastic in nature and so it would not be possible to repeat the results identically without knowing the seed and random generator used.

TABLE I
VALENCE EVALUATION METRICS

Valence Metrics	**Biased**		**Debiased**	
	F↑-↓**M**	**F**↓-↑**M**	**F**↑-↓**M**	**F**↓-↑**M**
Biased Pairs	1192	248	884	556
Mean Difference	2.33e−2	1.07e−2	1.29e−2	9.34e−3
Variance	2.69e−4	1.47e−5	1.31e−4	3.45e−5

[3]e.g. he/she, her/him, my aunt/my uncle

REFERENCES

[1] C. Sweeney and M. Najafian, "Reducing sentiment polarity for demographic attributes in word embeddings using adversarial learning," in *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, ser. FAT* '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 359–368. [Online]. Available: https://doi.org/10.1145/3351095.3372837

[2] R. Řehůřek and P. Sojka, "Software Framework for Topic Modelling with Large Corpora," in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta: ELRA, May 2010, pp. 45–50.

[3] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[4] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 168–177. [Online]. Available: https://doi.org/10.1145/1014052.1014073

[5] I. T. Jolliffe and J. Cadima, "Principal component analysis: a review and recent developments," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, p. 20150202, 2016.

[6] S. Mohammad, F. Bravo-Marquez, M. Salameh, and S. Kiritchenko, "SemEval-2018 task 1: Affect in tweets," in *Proceedings of The 12th International Workshop on Semantic Evaluation*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 1–17. [Online]. Available: https://www.aclweb.org/anthology/S18-1001

[7] S. Kiritchenko and S. M. Mohammad, "Examining gender and race bias in two hundred sentiment analysis systems," *CoRR*, vol. abs/1805.04508, 2018. [Online]. Available: http://arxiv.org/abs/1805.04508

[8] B. H. Zhang, B. Lemoine, and M. Mitchell, "Mitigating unwanted biases with adversarial learning," in *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, ser. AIES '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 335–340. [Online]. Available: https://doi.org/10.1145/3278721.3278779

[9] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[10] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, V. Vapnik *et al.*, "Support vector regression machines," *Advances in neural information processing systems*, vol. 9, pp. 155–161, 1997.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[12] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," vol. 14, 01 2014, pp. 1532–1543.