Master's Dissertation in Linguistics

Clustering Words from Biased Contexts using Dimensionality Reduction

차원 축소를 이용한 편향적 문맥에서의 단어 클러스터링

August 2020

Seoul National University Graduate School
Department of Linguistics

Catherine Sullivan

Abstract

Clustering Words from Biased Contexts using Dimensionality Reduction

Catherine Sullivan

Department of Linguistics

Graduate School

Seoul National University

Bias can be defined as disproportionate weight in favor of or against one thing, person, or group compared with another. Recently, the issue of bias in machine learning and how to de-bias natural language processing has been a topic of increasing interest. This research examines bias in language, the effect of context on biased-judgements, and the clustering of biased- and neutral-judged words taken from biased contexts.

The data for this study comes from the Wikipedia Neutrality Corpus (WNC) and its representation as word embeddings is from the bias neutralizing modular model by Pryzant et al. (2019). Visualization of the embeddings is done using K-means clustering to compare before and after the addition of the *v* vector, which holds bias information. Principal

Component Analysis (PCA) is also used in an attempt to boost performance of clustering.

This study finds that because the word embeddings cluster according linguistic features, the biased words also cluster according to bias type: epistemological bias, framing bias, and demographic bias. It also presents evidence that the word embeddings after being combined with the unique *v* vector from the modular model contain discrete linguistic information that helps not only in the task of detecting and neutralizing bias, but also recognizing context.

Keyword: Bias, bias neutralization, clustering, k-means, dimensionality reduction, PCA, word embeddings

Student Number: 2018-21128

Table of Contents

List of Figures

List of Tables

List of Equations

1. Introduction

1.1. What is Bias?

Bias is a disproportionate weight in favor of or against one thing, person, or group compared with another, usually in a way considered to be unfair. Often times when people hear the word "bias," they think of offensive social biases, discrimination based on race, gender, age, etc. This is likely because statements of explicit social bias are jarringly obvious to most people as being non-neutral in point-of-view. While such expressions of bias undoubtedly affect any data set and Natural Language Processing (NLP) model in significant ways, it is often the implicit biases lurking in our data which prove to be not only more insidious but also more difficult to identify.

It could be argued that bias—undeservedly—gets a bad reputation. Actually, bias is not inherently bad and is actually useful for many NLP tasks. Tasks such as sentiment analysis would be impossible to perform if the data was completely neutralized of any and all biases. The key when dealing with bias is recognizing the type of bias, how and when it appears, and whether it is harmful or helpful in a given context.

The most commons ways in which bias is introduced in NLP is when there is bias in the data, bias in collection and annotation, and bias in

interpretation. When there is bias in the data, it usually happens during data collection. For example, reporting bias happens during data collection when the information that is received from people does not actually reflect the real-world. This often happens when participants of a study know what it is that the data collectors are looking for and—consciously or subconsciously—give the collectors the answers that they think are desired. Another example of bias in data comes from selection bias, when the data selected doesn't represent a truly random sample.

Bias in interpretation often occurs subconsciously on the part of the researchers. Bias such as confirmation bias, overgeneralization, and correlation fallacy are all common types of bias that occur during interpretation. Confirmation bias is the tendency to search for and interpret information in a way that confirms preexisting beliefs or hypotheses. Overgeneralization happens when a conclusion is formed based on information that is not exact enough to warrant such a conclusion. Correlation fallacy refers to the mistake of labelling something a cause instead of just a correlation.

There are, of course, many other ways in which bias can be introduced, but these are some of the most common ways in which biases make their way into NLP models. While bias in interpretation is up to individual researchers to take responsibility for, bias in the data can cause serious problems for

researchers. This is why being able to de-bias data before it is used for research has become a hot topic. If the biases can be removed from language data before it is used, then the language models and research done based on those models are less likely to produce biased results.

This study focuses on subjective bias in text. Pryzant et al. (2019) describe subjective bias as inappropriate subjectivity and describe subjective bias as something that "[…] occurs when language that should be neutral and fair is skewed by feeling, opinion, or taste (whether consciously or unconsciously)." In other words, subjective bias has to do with personal likes or dislikes. Opinion pieces such as op-eds, reviews, and blogs are expected to contain subjective bias and in fact readers search out these types of writings because they want to know how other people think and feel about a particular thing. Informative, fact-based writing such as encyclopedias or nonfiction books, on the other hand, should be less about feelings and more about provable, factual information, which makes it an inappropriate place for subjective bias. This is why bias neutralization is important, to ensure that informative writing is actually fact-based and not opinion-based.

There are many different ways in which bias can be classified. It is impossible to look at all types of bias at once, so instead this study focuses on three major types of bias: epistemological bias, framing bias, and demographic bias (Pryzant et al. 2019).

1. Epistemological Bias is introduced through the use of linguistic features that modify the believability of a proposition such as the use of hedges or subject intensifiers.
2. Framing Bias refers to the use of subjective words of phrases linked with a particular point of view. Oftentimes this type of bias is represented by opinion words such as “best.”
3. Demographic Bias includes social biases such as bias that makes suppositions based on race or gender. Like the presupposition that doctors and male and nurses are female.

There are multiple ways in which these types of biases can appear. Because humans are used to speaking and writing based on their opinions and feelings, any text written by humans tends to be inherently biased. This also means that humans are used to hearing and reading other people’s opinions and feelings and consequently, it can sometimes be tricky even for a human to recognize bias. To give a better understanding of the three types of bias being examined in this study, examples from Pryzant et al. (2019) which were pulled from the Wikipedia Neutrality Corpus are listed below.

1) Epistemological

    a. The authors' *exposé* on nutrition studies

    b. The authors' *statements* on nutrition studies

Although the words 'exposé' and 'statement' in the sentences above both mean that the author was giving out information on nutrition studies, the term "exposé" insinuates that the information shared was a major revelation of something unsavory. Additionally, the term "exposé" has the added aspect of insinuating truthfulness. Using the term "exposé" suggests to the reader that whatever is in the exposé is true. "Statement," on the other hand, has no connotations other than its immediate definition of saying or communicating something and makes no judgement about truthfulness. It is because of the extra connotations of 1a) that makes it biased while 1b) is neutral despite the words being mostly synonymous.

2) Framing

    a. Most of the gameplay is *pilfered from* DDR.

    b. Most of the gameplay is *based on* DDR.

The word 'pilfered' in 2a) is biased here because it manipulates the reader into thinking that the opinion that the gameplay is stolen from DDR is

a fact. It frames the gameplay in a negative light. Whether the gameplay was indeed stolen or not, the phrase “based on” could mean that aspects of the gameplay with or without permission and does not frame the gameplay in either a negative or positive way which is what makes it more neutral than “pilfered from.”

3) Demographic

    a. Marriage is a *holy* union of individuals.

    b. Marriage is a *personal* union of individuals.

3) is an excellent example of demographic bias because anyone, regardless of religious belief, is able to get married. Regardless of who it is getting married, marriage can objectively be described as personal. However, the description of marriage as a holy union is subject to religious belief and the term “holy union” is especially used by people of Christian faith. Therefore, the description of marriage as a holy union is biased because it only holds true for a certain demographic of people while the neutral version is true for anyone.

Epistemological bias, framing bias, and demographic bias have many similarities between them and may in some cases overlap, which can make bias classification a difficult task, as well as complicating bias detection and

de-biasing. The following section describes techniques that have been used to neutralize bias.

1.2. De-biasing Techniques

There are many different methods that have been used in attempt to remove bias from Natural Language Processing models. Most current research in de-biasing include techniques such as hard de-biasing, soft de-biasing, flipping, and, most recently, automatic bias neutralization.

Hard de-biasing refers to the technique of completely removing subspace components from embeddings. It can also be referred to as "Neutralize and Equalize" as those are the two main steps in hard-debiasing. The neutralization step involves removing the bias component from words that should not contain bias. The equalization step involves centering the word embeddings and equalizing the bias components so that any words that contain implicit information related to the source of the bias (i.e., "man" and "woman" in the case of gender bias) are equidistant to any biased words.

Manzini et a. (2019) neutralize and equalize their embeddings with the following equations, given a bias subspace B spanned by vectors $\{b_1, b_2, \ldots\ b_k\}$:

1) Neutralize: compute the component of each embedding in the subspace

$$w_B = \sum_{i=1}^{k} \langle w, b_i \rangle b_i \qquad 1.1.1.$$

2) Neutralize: remove the component from words that should be bias-neutral and normalize

$$\mathrm{w}' = \frac{w - w_B}{\|w - w_B\|} \qquad 1.1.2.$$

3) Equalize: for word embeddings in an equality set, *E*, let $\mu = \frac{1}{|E|} \sum_{w \epsilon E} w$ be the mean embedding of the words in the set and μ_B be its component in the bias subspace. Then for w ∈ E:

$$\mathrm{w}' = (\mu - \mu_B) + \sqrt{1 - \|\mu - \mu_B\|} \frac{w - \mu_B}{\|w - \mu_B\|} \qquad 1.1.3.$$

Soft de-biasing is similar to hard-debiasing but it minimizes instead of neutralizes the projection of the word embeddings onto the bias subspace.

Manzini et al. (2019) perform this mathematically as follows. Given embeddings W and N which are embeddings for the whole vocabulary and the subset of bias-neutral words and the bias subspace B, find A that minimizes:

$$\|(AW)^\top(AW) - W^\top W\|_F^2 + \lambda\|(AN)^\top(AB)\|_F^2 \qquad 1.2.$$

Here, Manzini et al. (2019) minimize the first term to preserve the inner product after the *A*, the linear transformation and minimize the second term to minimize the projection on to the bias subspace. They use $\lambda \in \mathbb{R}$ as a tunable parameter to balance the two objectives.

Flipping is a de-biasing technique generally used in cases of social bias. Using this method, the words are "flipped" so that all demographics are represented equally in the data. For example, if men and women were represented differently in the data, the sentences could be duplicated with the gender indicators flipped. This way, instead of a model that only saw the sentences "The doctor asked the nurse if she would help him with the procedure," the model would also see the opposite sentence, "The doctor asked the nurse if he would help her with the procedure." This way, the NLP model is less likely to make inappropriate generalizations.

Automatic bias neutralization is a way to de-bias language using a system that learns from a set of parallel sentences how to correct for bias based on context. It can be thought of in a similar fashion as other automated NLP tasks such as automatic machine translation or automatic classification. That is, the task is learned and carried out by an automated model without a human overseeing the process. This is good because such models are generally more robust when it comes to unfamiliar input. For example, when a model that makes use of hard de-biasing encounters unfamiliar input, it has no reference whether the words are biased or not if the words are not on the pre-determined, human-defined biased or unbiased list. An automated model, on the other hand, is much more likely to guess, and guess fairly well, whether the words are biased or not based on features that it has learned from other biased and unbiased words. While it is by no means perfect and does make mistakes, it is this adaptability that makes automated NLP systems promising.

Techniques such as hard de-biasing, soft de-biasing, and flipping have been the most popular methods of de-biasing, however, their capacity for bias detection and de-biasing is much more limited than automatic neutralization. While undoubtedly extremely useful in simple de-biasing tasks, most of these techniques are unable to account for context. Automatic neutralization provides a more nuanced method of de-biasing based on context. Automatic bias neutralization is currently the newest system for de-biasing that has been

developed and there aren't many models that have been tested, but the baselines as well as new, unique models that have been developed have shown impressive results. It is for this reason that this study focuses on automatic bias neutralization and how bias is represented within an automatic bias neutralization model. Specifically, this study investigates the modular model for automatic bias neutralization presented by Pryzant et al. (2019).

1.3. Purpose and Significance of this Study

This study uses two sets of word embeddings (one set without bias information, H, and one set including bias information, H') taken from the modular model by Pryzant et al. (2019). These two sets of word embeddings are clustered using k-means and optimized using the dimensionality reduction algorithm principal component analysis. The visualization of these clusters as well as information extracted from them is used in an effort to analyze how subjective bias is represented in the word embeddings and an attempt to understand how an automatic de-biasing system "understands" bias.

Additionally, this study examines the *v* vector from the modular model. (Pryzant et al., 2019). The *v* vector contains a variety of linguistic

information which is applied to the words in the word embeddings based on their probability of being bias. This information helps adjust for bias, but it doesn't actually represent bias itself. Because of the opacity of the *v* vector, viewing the effect that it has on the word embeddings may help to better understand the role that it plays in the model.

This type of analysis is essential to the development of bias detection and bias neutralization. The better bias is understood, the easier it is to create systems that are able to deal with bias—whether that be detection, classification, or neutralization. Although it is difficult to look into a model and understand exactly how it learns to perform language tasks, viewing the word embeddings utilized in an NLP model can help formulate a hypothesis. The goal of this study is to shed some light on the inner workings of a state-of-the-art automatic bias neutralization model and bias in natural language processing.

2. Background Information

2.1. Previous Research

Given that NLP tasks are heavily dependent on text written by humans and humans tend to write based on their feelings and opinions, such writings are inherently biased. This can cause difficulty for certain NLP tasks which is why a great deal of research has been done relating to bias and NLP.

Recasens et al. (2013) tested different linguistic models on their ability to analyze and detect biased language. The Wikipedia Neutral Point of View Corpus (Wiki NPOV Corpus), which mostly consists of instances of framing and epistemological bias, was used to train a new, linguistically-informed bias detection model. This detection model was informed by common linguistic cues which often indicate bias including factive verbs, implications, hedges, and subjective intensifiers. The linguistically-informed bias detection model performed better than the other five other bias detection models that were trained using the same data and while it did not beat human performance, it scored only several points under.

Swinger et al. (2019) tested their Unsupervised Bias Enumeration (UBE) for enumerating bias in word embeddings. Their algorithm uses a word embedding, set of names, number of target groups, number of categories,

number of frequent lower-case words, number of words per WEAT and a false discovery rate as input to the algorithm. Using the UBE, they found that a large number of the publicly available word embeddings, including embeddings that were supposed to be "de-biased," included a large number of offensive associations related to sensitive features such as race and gender.

Pant et al. (2020) tested three different BERT-based models trained on the Wikipedia Neutrality Corpus (WNC) for detection of subjective bias. They found their ensemble model consisting of *RoBERTa*, *ALBERT*, *DistillRoBERTa*, and *BERT* achieved the highest F1 and Accuracy scores.

Other studies investigate a variety of approaches to bias detection. Kiritchenko and Mohammad (2018) created the Equity Evaluation Corpus (EEC) for use in bias detection. Basta et al. (2019) analyzed bias in different word embeddings and found that contextualized word embeddings were less biased than standard ones, even after the standard embeddings were de-biased. Nissim et a. (2019) criticized the use of analogies (i.e. Man is to King as Woman is to X) as a tool to diagnose bias and presented other methods that were much better suited for bias detection.

When it comes to de-biasing, much of the research on de-biasing has been focused on word embeddings. Bolukbasi et al. (2016) and Manzini et al. (2019) both focus on social bias and attempt to neutralize those biases by removing the bias component (gender in the case of the former and race in

the case of the latter) from the word embeddings. Zhao et al. (2018) tackled gender bias in word embeddings by creating a gender-neutral version of GLOVE. Zhao et al. (2017) implemented corpus-constraints to minimize gender bias.

However, most of these methods that have been used to neutralize bias are fairly rigid and unable to effectively account for context. Additionally, it has been suggested that while these de-biasing techniques appear to remove bias, they are merely glossing over the most obvious symptoms of bias and fail to effectively remove bias. This is talked about in Swinger et al. (2019), which was described in the previous section as well as the 2019 paper by Gonen and Goldberg. Gonen and Goldberg (2019) tested the hard-de-biased data from Bolukbasi (2016) and the GN-GLOVE (Zhao et al., 2018) and found that despite the absence of inherently gendered words (e.g. girl, her, brother), female/male stereotype words (e.g. nurse/doctor) remained clustered based on gender stereotype.

All these previous studies show that bias detection and de-biasing is a difficult task with significant challenges. One of the biggest contributing factors to this difficulty is that bias is difficult to identify, even for humans. Because bias is so dependent on a variety of different factors, it makes sense to apply automatic methods to de-biasing. This study will focus on one

particular automatic bias neutralization approach created by Pryzant et al. (2019), which is discussed in greater detail in section 2.3.

The paper that this study draws from the most is Pryzant et al. (2019). The authors of this paper compiled the Wikipedia Neutrality Corpus (WNC) and used that data to establish baselines and implement their own unique model for automatically neutralizing subjective bias. They created two of their own linguistically-informed models (a Modular Model and a Concurrent Model) which were infused with linguistic features from Recasens et al. (2013) to help identify the bias-inducing word.

The modular model is made up of a BERT-based detection module and an LSTM-based editing module, pre-trained and then combined to form an end-to-end neutralization system. The concurrent model is an encoder-decoder neural network with a BERT encoder and an LSTM decoder.

The concurrent model and modular model from Pryzant et al. (2019) outperform the baseline models on the WNC corpus. A sample of the automatically neutralized text was extracted and any incorrect neutralizations were checked for the source of error. While most of the error came from when the model made a change that did not match the target from the corpus, in 80% of such cases the model still managed to change the sentence in a way that that humans judged to successfully neutralize the bias in an acceptable way. The types of errors that caused issues and never resulted in a successfully

neutralized sentence occurred where the model failed to make a change where one was needed or where errors in the language modelling or text generation created disfluency.

In addition to being tested on the WNC dataset, the two models' performances were also tested on and able to successfully de-bias several different real-world media sources. Both models were tasked with de-biasing:

- The Ideological Books Corpus (IBC) - partisan books and magazine articles (Sim et al. 2013; Iyyer et al. 2014).
- Headlines of partisan news articles identified as biased according to mediabiasfactcheck.com.
- Sentences from the campaign speeches of the United States President Donald Trump

Pryzant et al. (2019) found that both models performed impressively on the de-biasing task. Although their modular model performed better when it came to reducing bias, their concurrent model was better able to preserve the meaning and fluency of the original text. This study will investigate the modular model, specifically looking at its word embeddings and the influence of the bias probabilities and the *v* join embedding on those embeddings.

2.2. Wikipedia Neutrality Corpus

The Wikipedia Neutrality Corpus (WNC)[1] was collected by Pryzant et al. (2019) and consists of aligned biased and unbiased sentence pairs that were edited according to Wikipedia's Neutral Point of View (WikiNPOV) policy in order to correct for bias. The WikiNPOV policy states:

> "All encyclopedic content on Wikipedia must be written from a neutral point of view (NPOV), which means representing fairly, proportionately, and, as far as possible, without editorial bias, all the significant views that have been published by reliable sources on a topic."[2]

The ability to view text before and after being edited makes Wikipedia an excellent source of language data and there are a number of corpuses that make use of Wikipedia data. The NPOV policy makes it especially useful as a source of biased vs de-biased language. Additionally, Wikipedia edits are

[1] Available at http://bit.ly/bias-corpus
[2] The WikiNPOV policy can be read in full at https://en.wikipedia.org/wiki/Wikipedia:Neutral_point_of_view

also annotated with reasons for why the text was edited, including tags for neutral point of view.

Pryzant et al. crawled Wikipedia revisions using NPOV tags to create the WNC. From those edits tagged with "NPOV" revisions were ignored where:

- More than a single sentence was changed.
- Minimal edits (character Levenshtein distance < 4).
- Maximal edits (more than half of the words changed).
- Edits where more than half of the words were proper nouns.
- Edits that fixed spelling or grammatical errors.
- Edits that added references or hyperlinks.
- Edits that changed non-literary elements like tables or punctuation (Pryzant et al. 2019).

After the exclusion of such edits, the WNC contains a total of 64,303 sentence pairs. The training set is made up of 53,803 sentence pairs, a test set is made up of 10,000 sentence pairs, and a dev set is made up of 700 sentence pairs. The corpus contains framing bias, epistemological bias, and demographic bias as described in section 1.1.

The training set, test set, and dev set from the WNC are tsv files with 7 columns: edit ID, source tokens (tokenized original sentence), target tokens (tokenized human-edited neutral sentence), raw source (original sentence), raw target (human-edited neutral sentence), source POS tags (part of speech tags for the source sentence), and target parse tags (parse tags for the humna-edited neutral sentence). Any other additional information is assigned by the model. One example from the WNC is shown in Figure 2.1.

7.84E+08	in 1970 , cu ##rie ##l directed mil in two of his greatest team - up movies .	in 1970 , cu ##rie ##l directed mil in two of his team - up movies .	in 1970, curiel directed mil in two of his greatest team- up movies.	in 1970, curiel directed mil in two of his team- up movies.	ADP NUM PUNCT NOUN NOUN NOUN VERB NOUN ADP NUM ADP ADJ ADJ NOUN PUNCT PART NOUN PUNCT	prep pobj punct ROOT ROOT ROOT amod dobj prep pobj prep poss amod nmod punct prt pobj punct

Figure 2.1.: Example from the WNC dataset.

2.3. Modular Model

The model investigated in this study comes from the work of Pryzant et al. (2019). It is called a Modular Model because of its two modules that

work together to automatically neutralize bias in text. It is made up of a BERT-based detection module and an LSTM-based editing module. The model is shown in its entirety in figure 3.1. below.

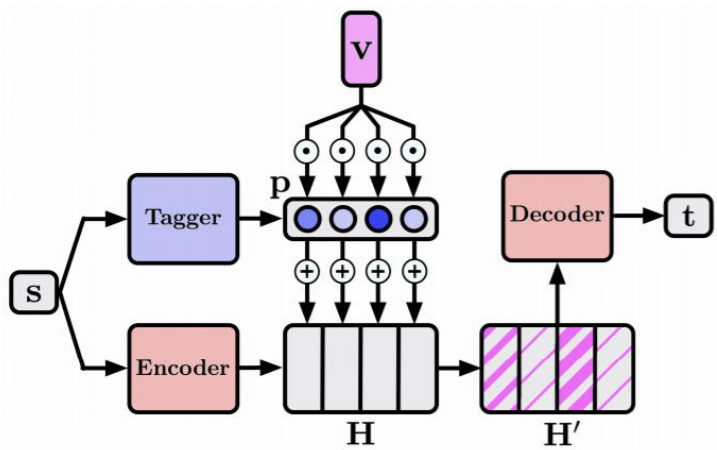


Figure 2.2.: Modular model from Pryzant et al. (2019)

The detection module is labelled as "Tagger" in figure 2.1. It uses f_i, a vector of discrete features (such as lexicons of hedges, factives, assertives, implicatives, and subjective words), and BERT embedding b_i to calculate logit y_i. An illustration of the detection module is shown in Figure 2.2.

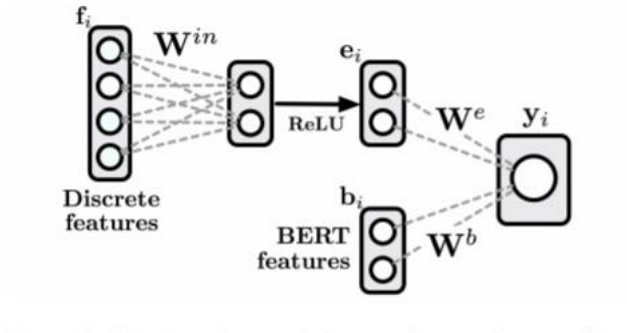


Figure 2.3.: The detection module from the modular model of Pryzant et al. (2019).

The detection module has a neural sequence tagger that estimates *p*, the probability that each input word (w_i^s) is subjectively biased. Pryzant et al. (2019) calculated these probabilities with the following equation:

$$p_i = \sigma(b_i W^b + e_i W^e + b) \qquad 2.1.1.$$

- $b_i \in \mathcal{R}^b$ is a contextualized word vector which represents a word's (w_i^s's) semantic meaning.
- e_i represents expert features of bias by Recasens et al. (2013).[3] f_i represents the discrete features, shown in Figure 2.3.

[3] The table from Recasens et al. (2013) which lists the features of bias taken into consideration is included in Table A in the appendix.

$$e_i = \mathrm{ReLU}(f_i W^{in}) \qquad 2.1.2.$$

- $W^{in} \in \mathcal{R}^{f \times h}$ represents a matrix of learned parameters.
- $W^b \in \mathcal{R}^b$, $W^b \in \mathcal{R}^h$, and $b \in \mathcal{R}$ are learnable parameters.

The "Encoder" and "Decoder" in Figure 2.1. are parts of the LSTM-based editing module. The editing module takes a subjective source sentence, *s*, and is trained to edit it into a more neutral compliment (the target sentence, *t*). This is accomplished through the work of a bi-LSTM encoder which changes *s* into a sequence of hidden states, $H = (h_1, \ldots, h_n)$ and an LSTM decoder which generates text by attending to H and producing probability distributions over the vocabulary. ("H" and "s" are also labelled in Figure 2.3.)

After both modules are pre-trained, they are joined together using a 'join embedding,' *v*, and fine-tuned together as an end-to-end system. The join embedding is a vector $v \in R^h$ which is added to each encoder hidden state in the LSTM editing module. A different amount of the *v* vector is applied to the hidden states depending on the bias probabilities.

This model performed better than any of the baseline style transfer systems and machine translation systems that it was compared with. It also

performed similarly to the concurrent model, the other novel model developed by Pryzant et. al., with the modular model achieving a higher accuracy and being better at bias detection and the concurrent model receiving a higher BLEU score and performing better when it came to fluency and retention of sentence meaning.

This study focuses on the word embeddings before and after the inclusion of the *v* vector (represented in Figure 2.2. as *H* and *H'*.)

2.4. Methodology

The biggest facet of this study is the clustering of the word embeddings and the optimization of the clustering algorithm. This section outlines the basics of the k-means clustering algorithm used as the baseline clustering algorithm as well as describing Principal Component Analysis (PCA), the dimensionality reduction algorithm used for optimization. Using these two methods, the word embeddings from the Modular Model were visualized so that the biased terms could be analyzed.

2.4.1. Clustering

Within a model, the word embeddings exist as matrices of numbers which makes it difficult for humans to look at and conceptualize their meanings. To make it easier to understand the word embeddings, we visualize them using clustering algorithms. Clustering algorithms work by organizing the data so that similar objects are closer to each other and further away from objects that they are different from.

There are many different types of clustering algorithms such as hierarchical clustering, distribution-based clustering, density-based clustering, grid-based clustering, and centroid-based clustering. All of these methods of clustering have their strengths and weaknesses and choice of clustering method is subject to personal preference. Hierarchical clustering tends to work best for hierarchical data, such as taxonomies, which would make it a less than ideal choice for this study since the data used is not intrinsically hierarchical. Distribution-based is not recommended unless the type of distribution in the data is already known. Density-based clustering algorithms do not perform well on data with varying densities and high dimensions and additionally do not assign outliers to clusters. Similarly, grid-based clustering struggles with high-dimensionality. On the other hand, centroid-based

clustering methods are efficient but still responsive to initial conditions and outliers. Because of this as well as the drawbacks of other clustering methods, this study uses centroid-based clustering, as it is best suited to both the type of data being clustered as well as best fulfilling the goal of this study.

While there are multiple options available when it comes to centroid-based clustering algorithms, k-means is one of the most widely-used options because it is simple, but still efficient and effective. Before the k-means algorithm can be done, centroids must be initialized. There are multiple methods for initializing k-means, but this study uses k-means++ (Arthur and Vassilvitskii, 2007). K-means++ initializes the cluster centroids by choosing the first cluster center at random from the data points and choosing the remaining cluster center(s) according to its probability proportional to the squared distance from the point's closest existing cluster center.

There are several different versions of k-means, but this study uses the standard algorithm (native k-means). Native k-means works in two steps. First, in the assignment step, a *k* number of specified data points are randomly initializing to serve as centroids. Then, in the update step, the means are recalculated until there is no change in the centroid locations. Native k-means can be expressed mathematically as follows (MacKay, 2003).

1) Assignment Step:

$$S_i^{(t)} = \left\{ x_p : \left\| x_p - m_i^{(t)} \right\|^2 \leq \left\| x_p - m_j^{(t)} \right\|^2 \ \forall j, 1 \leq j \leq k \right\} \quad 2.2.1.$$

Where each $x_p$is assigned to exactly one $S^{(t)}$.

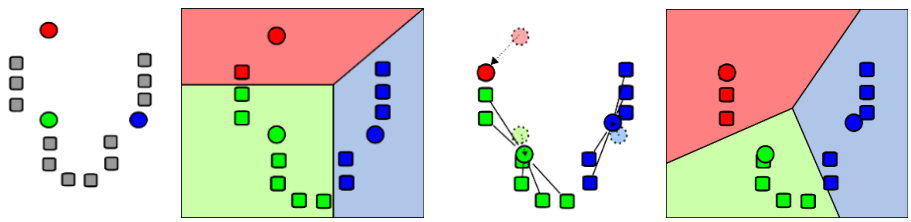
2) Update Step:

$$m_i^{(t+1)} = \frac{1}{\left| S_i^{(t)} \right|} \sum_{x_j \in S_i^{(t)}} x_j \quad 2.2.2.$$

Figure 2.1. shows how the native k-means steps can be visualized. The first image (a) shows the initialization of random points as centroids. The next image (b) shows how the data points are separated into clusters around those centroids. In image (c), the movement of the centroids in the update step is shown to make (d) the final clustering results. In practice, the update step may be repeated multiple times.

Although there are many other clustering algorithms that can be used to a similar effect as k-means, k-means is the most popular because it serves as a good baseline clustering algorithm. Many other clustering algorithms tend to be much slower and use more memory but k-means is one of the

simplest algorithms to implement and run. It is for these reasons that k-means was chosen as the baseline clustering algorithm for this study.



(a) initialization (b) cluster creation (c) update (d) final clustering

Figure 2.4.: Illustration of native k-means.[4]

2.4.2. Dimensionality Reduction Algorithm

Clustering algorithms tend to perform better, in ways that are more intuitive and interpretable to humans, when they are distance-based. Unfortunately, when the distance is too great, it can cause issues with clustering. This is why dimension reduction algorithms can be useful tools. Dimension reduction algorithms reduce the number of variables that need to be taken into consideration. Because the data clustered in this study has too

[4] From: I, Weston.pace / CC BY-SA (http://creativecommons.org/licenses/by-sa/3.0/)

many dimensions to be graphed for visualization, dimension reduction is used to graph the data in 2 and 3-dimensions.

There are two main approaches to dimensionality reduction: feature selection and feature projection. Feature selection reduces the number of variables by selecting a smaller subset of the input variables. Feature projection still takes all of the variables into consideration, but transforms the data to fit a space of fewer dimensions. This study utilizes a feature projection approach.

While this study originally considered using T-distributed Stochastic Neighbor Embedding (t-SNE) and Uniform Manifold Approximation and Projection (UMAP) to perform dimensionality reduction, such algorithms are computationally expensive and Principal Component Analysis (PCA) was instead chosen as the main algorithm for use in this study.

PCA was used in this study to improve the performance of the k-means clustering. As a dimensionality reduction algorithm, it is used to visualize distance and relatedness between data points. PCA accomplishes this by linearly mapping the data to a lower-dimension space while maximizing the variance present in the data. There are two main steps to this; first, calculating the data covariance (or correlation) matrix of the original data and then, performing eigenvalue decomposition on the covariance matrix.

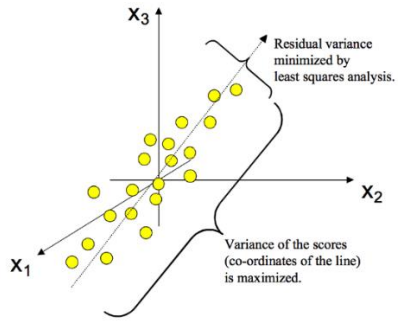


Figure 2.5.: PCA Dimensionality Reduction (Eriksson, 2018).

Mathematically, PCA works by creating a new data set of *d* dimensions from the original dataset consisting of *d+1* dimensions and computes the mean for every dimension in this new dataset. Then, the sample covariance matrix is calculated:

$$S_{XY} = \frac{1}{n-1}\sum_{i=1}^{n}(Xi - \bar{X})(Yi - \bar{Y}) \qquad 2.3.1.$$

After the sample covariance is calculated, the eigenvectors and corresponding eigenvalues are computed. (Eigenvectors are vectors whose directions do not change when linear transformation is applied and eigenvalues, represented by lambda, are numbers that describe the variance in the data. They are used to reduce noise in data and help prevent overfitting.) The equation to find the eigenvectors is in 3.4:

$$\det(A - \lambda I) = 0 \qquad 2.3.2.$$

The eigenvectors are then ordered by decreasing eigenvalue. K number of eigenvectors with the highest eigenvalues are selected to create a $d \times k\ dimension$ matrix, W, which is used to transform the samples onto the subspace.

While PCA is the main method of dimensionality reduction used in this study, Uniform Manifold Approximation and Projection (UMAP) was also used (McInnes et al. 2018). UMAP is computationally expensive and was not able to be run on the entirety of the data, hence it was only used with a small subsection of the data.

The UMAP algorithm consists of two phases: constructing a fuzzy topological representation and optimizing the low dimensional representation to have as close a fuzzy topological representation as possible. The following mathematical explanation of UMAP is summarized and the equations taken from McInnes et al. (2018).

In the first phase, creating a fuzzy topological representation, can also be considered as constructing a weighted k-neighbor graph. If $X =$

$\{x_1, \ldots, x_N\}$ is the input data set with a dissimilarity measure $d: X \times X \to \mathbb{R}_{\geq 0}$, then given the input hyper-parameter, k, for each x_i the set $\{x_{i_1}, \ldots, x_{i_k}\}$ of the k nearest neighbors of x_i under the metric d is computed. For each x_i, ρ_i is defined as:

$$\rho_i = \min\{d(x_i, x_{i_j}) | 1 \leq j \leq k, d(x_i, x_{i_j}) > 0\} \quad 2.4.1$$

and σ_i is set to a value such that

$$\sum_{j=1}^{k} \exp\left(\frac{-\max(0, d(x_i, x_{i_j}) - \rho_i)}{\sigma_i}\right) = \log_2(k). \quad 2.4.2$$

Then, vertices V of $\bar{G}$ are set to the set X to define a weighted directed graph $\bar{G} = (V, E, w)$. The set of directed edges $E = \{(x_i, x_{i_j}) | 1 \leq j \leq k, 1 \leq i \leq N\}$ are formed and the weight function w is defined:

$$w\left(\left(x_i, x_{i_j}\right)\right) = \exp\left(\frac{-\max\left(-0, d\left(x_i, x_{i_j}\right) - \rho_i\right)}{\sigma_i}\right) \quad 2.4.3$$

If A is the weighted adjacency matrix of $\bar{G}$ and $\circ$ is the pointwise product, the UMAP graph *G* is an undirected weighted graph with an adjacency matrix of *B*.

$$B = A + A^{\intercal} - A \circ A^{\intercal} \qquad 2.4.4.$$

This completes the first step. The second step, optimization, involves using a force directed graph layout algorithm in a low dimensional space. The algorithm iteratively applies attracting and repulsive forces at each edge or vertex until convergence. With *a* and *b* as hyper-parameters, the attractive force between vertices *i* and *j* at coordinates y_i and y_j is calculated.

$$\frac{-2ab\|y_i - y_j\|_2^{2(b-1)}}{1+\|y_i - y_j\|_2^2} w((x_i, x_i))(y_i - y_j) \qquad 2.5.1$$

Then, the repulsive forces are computed via sampling. ϵ is a small number to prevent division by zero.

$$\frac{b}{\left(\epsilon+\|y_i-y_j\|_2^2\right)\left(1+\|y_i-y_j\|_2^2\right)}(1-w((x_i,x_i)))(y_i-y_j) \qquad 2.5.2.$$

The pseudocode for the UMAP algorithm as written by McInnes et al. (2018) is included in the Appendix.

3. Experiment

First, the word embeddings before and after the inclusion of the *v* vector needed to be extracted from the modular model. To do this, the code for the model was downloaded from github[5] and several files were modified slightly in order to output the word embeddings. The data was output in groups of 1,000 and saved in pickle files in order to avoid flooding the server's memory. The pickle files are dictionaries with the following information:

{ 'src_tokens': [$\text{sentence}_1, \text{sentence}_2, \ldots \text{sentence}_n$],

'gold_tokens': [$\text{sentence}_1, \text{sentence}_2, \ldots \text{sentence}_n$],

'pred_tokens': [$\text{sentence}_1, \text{sentence}_2, \ldots \text{sentence}_n$],

'encoder_Ht': [$\text{sentence}_1, \text{sentence}_2, \ldots \text{sentence}_n$],

'V_weights': array of shape (512, 512),

'V': array of shape (512,),

'bias_probs': [$\text{sentence}_1, \text{sentence}_2, \ldots \text{sentence}_n$], }

[5] Available at: https://github.com/rpryzant/neutralizing-bias

The ‘src_tokens’ contains the tokenized words of each sentence before being corrected for bias, ‘gold_tokens’ contains the tokenized words of each sentence after a human annotator corrected for bias, and ‘pred_tokens’ contains the tokenized words of each sentence that the module model corrected for bias. ‘Encoder_Ht’ is the contextual token embedding, that is, a vectorized representation of the word based on its context. ‘V_weights’ is a raw layer weight for V of shape (512, 512) and ‘V’ is just one vector with the shape (512,). The ‘bias_probs’ contains the probabilities that each token is biased.

After these files are extracted, the information needs to be processed so that it is easier to work with. The pickle files need to be iterated through so that the data from all the files can be combined according to key and each key can be saved in its own separate, but parallel, list. This is done for ease of access. Additionally, the words were not consolidated into a single instance, rather the same words are repeated multiple times in the data. This was done because much of the bias that appears in the data is related to context and not a single word’s intrinsic biasedness. This means that a word may be judged as biased in one instance, but unbiased in a separate instance.

First, the information from each of the pickle files is extracted and organized into lists for ease of access. The pickle files in the folder are opened

and the ‘V’ vector is assigned to a list, *v*. Then, the files are iterated over and the ‘src_tokens,’ ‘encoder_Ht,’ and ‘bias_probs’ are saved into lists. A counter is also started to create a sentence ID so that the information for each instance of a token or the token’s information can be referenced back to the original sentence. In the next step, the v vector is copied to the same length as the word embeddings and multiplied by the bias probabilities. For just the word embeddings, this is all that is done to apply bias information to the word embeddings. For the word embeddings with the inclusion of the *v* vector, the join embedding, the v vector multiplied by the bias probabilities is then added to the ht (the vectorized representation of the word based on its context). Because the lists containing ‘src_tokens,’ ‘encoder_Ht,’ and ‘bias_probs’ contain the entire sentence at each index, instead of just a single token, these lists must be iterated over and separated into new lists to hold each token separate from the sentence. Lastly, the bias probabilities iterated over and saved as binary biased judgements. This was done by finding the mean value of the bias probabilities for each sentence and then, any token with a probability higher than that its sentence’s mean was given the value of 1, signifying that it was biased, and any token with a probability lower than the sentence’s mean was given the value of 0, signifying that it was unbiased. (The histograms for the bias probabilities vs the bias judgements are shown below in Figures 3.1. and 3.2.) Lastly, the vectors must be changed from a list

into an array of shape (1697800, 512) representing the 1,697,800 words and the 512 features. The pseudocode used for this entire process is shown in Algorithm 1.

Algorithm 1 Pseudocode for extracting needed data

```
folder : location of the pickle files
joinembedding: set as False to get word embedding before inclusion of
the v vector, set as True to get word embedding after inclusion of v
vector
tokens, biases, sum, words, vectors, text, id, probability: empty lists
for file in folder do
 data ← files
 v ← 'V'
 for index in data do
  tokens ← 'src_tokens'
  ht ← 'encoder_Ht'
  biases ← 'bias_probs'
  sum ← sum + 1
  if joinembedding
   (v vector × bias probs) + ht
  for index, token in tokens do
   words ← token
   vectors ← ht
   text ← tokens
   id ← sum
  if x > mean
   probability ← 1
  else
   probability ← 0
  end if
  end for
   end if
 end for
 end for
feature_table ← array(vectors)
```

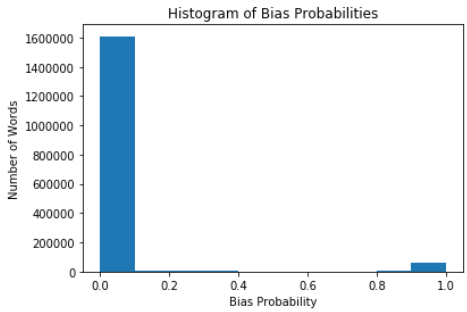


Figure 3.1.: Visualization of the bias probabilities.

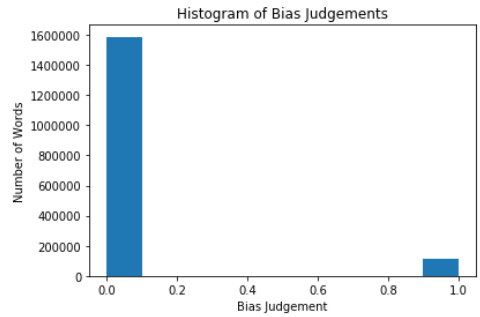


Figure 3.2.: Visualization of the binary bias judgements.

Because the array of array of word vectors of shape (1697800, 512) can't be clustered in 512-dimension, it must undergo dimensionality reduction before it can be clustered using k-means. First, PCA and k-means were imported from "sklearn" and plotting from "matplotlib." Then, the number of components used for pca was chosen and the array of word vectors was fit with pca to output a new array in 2 or 3 dimensions. This new array was then fit to the k-means model and plotted using "matplotlib." To annotate the points which received biased judgements, the length of the array was iterated through and the index of each point was referenced to the index in the parallel list of bias judgements. If that point received a bias judgement of '1,' biased, it was annotated as such in the plot. The pseudocode for this process is shown in Algorithm 2.

Algorithm 2 PCA/K-means Pseudocode

```
feature_table: numpy array of vectors from Algorithm 1
distortions: empty list
PCA imported from sklearn
Plotting from matplotlib
pca ← PCA(# of components)
embedding ← pca.fit(feature_table)
K ← range(1,10)
for k in K do
  kmean ← KMeans(k clusters, random state)
  kmean.fit(embedding)
  distortions ← (kmean.inertia_)
end for
kmean ← Kmeans(# of clusters)
kmean.fit(embedding)
plot(embedding[x], embedding[y], labels from kmean)

To annotate with bias judgements
i: index
probability: bias judgements from Algorithm 1
for i in range(len(embedding)) do
  if probability[i] is 1
    annotate plot(probability[i], (embedding[i][0], embedding[i][1])
```

The number of clusters for the k-means model was chosen according to the 'elbow method.' This method iterates over the values of k from 1 to 9 and calculates the values of each k's distortion (average of the squared distances from the cluster centers of the respective clusters) and inertia (sum of squared distances of samples to their closest cluster center). These are then visualized using a line graph which, if the data is capable of

being clustered well, should show an elbow shape, or a bend, whose point on the x axis correlates with the optimal number of clusters for the data.

Using UMAP as the dimensionality reduction algorithm before clustering is similar to using PCA, however adjustments had to be made since UMAP couldn't be run on the entire dataset. First, a random sample of the data needed to be taken. The list of words was enumerated and iterated over to create a list of word indexes and were then separated into two lists depending on whether they received a biased judgement or not. With the use of the package "random," 500 biased- and 500 neutral-judged word's indexes were selected and combined into a single list. The indexes from this list are then used to extract the word's vector (saved in Algorithm 1) and appended to a new list which is converted to a numpy array and then undergoes UMAP. Then, k-means clustering and plotting can be performed.

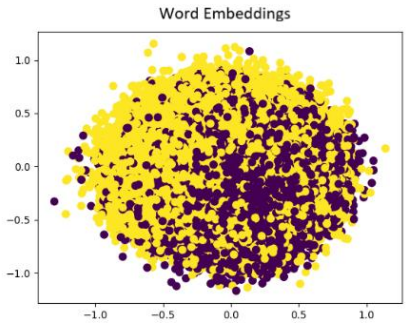
For comparison purposes, the same selection of random indexes was used for each clustering (word embeddings, word embeddings plus the join embedding, word embeddings with UMAP and word embeddings plus the join embedding with UMAP). PCA was also used on the same small selection so that PCA results could be compared with UMAP results (word embeddings with PCA and word embeddings plus the join embedding with PCA). The results of the clustering are shown in Section 4.

Algorithm 3 Random Vector Selection and UMAP Pseudocode

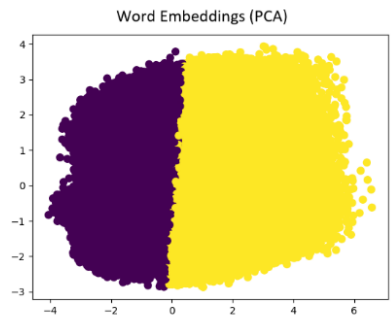
i, *w*: variables to represent index and word
biased, *neutral*: empty lists to append to
for *i, w* **in** enumerate(*words)* **do**
  observe_index ← *i*
  observe_word ← *words[i]*
  observe_vector ← *vectors[i]*
  observe_sentence ← *sentence_text[i]*
  if *w* **in** *words*
    word_index ← *observe_index*
  end if
end for
for *index* **in** *word_index* **do**
  if *probability* is 1
    biased ← *index*
  else
    neutral ← *index*
random_bias ← random.sample(*biased*, 500)
random_neutral ← random.sample(*neutral*, 500)
random_vectors ← *random*_bias and *random_neutral*
for *number* **in** *random_vectors* **do**
  sample_ vectors ← *vectors*[*number*]
sample ← np.array(*sample_vectors*)
reducer ← umap.UMAP()
embedding ← reducer.fit_transform(*sample*)

4. Results

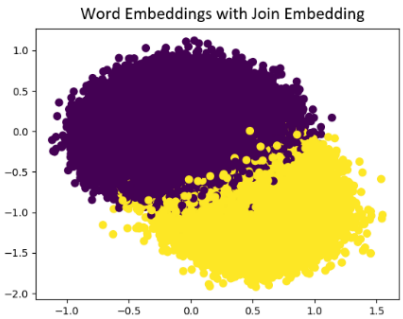
4.1. Clustering of Entire Data Set



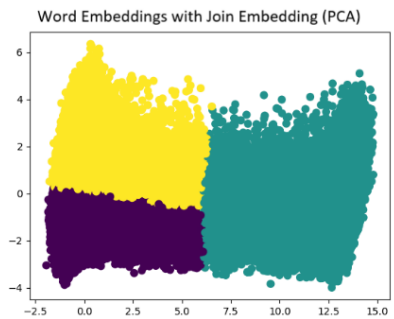
a)



b)



c)

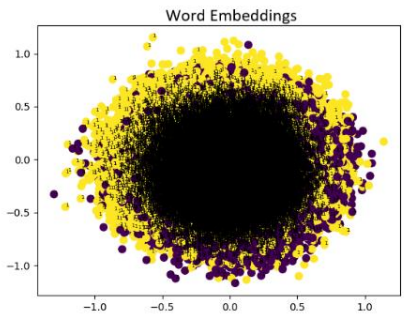


d)

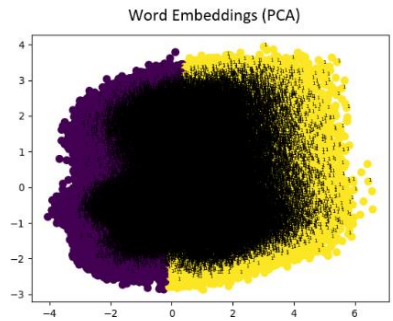
Figure 4.1.: a) – b) show the plotted clustering results. a) contains just the plain word embeddings, b) is those word embeddings plotted after the use of PCA, c) is the word embeddings with the join embedding, and d) Is the word embeddings with the join embedding plotted after the use of PCA.

As expected, the word embeddings which include the bias probabilities and the v (join embedding) cluster better than the word embeddings on their own and the use of PCA improved the clustering results.

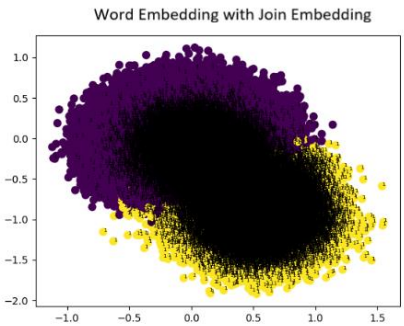
In figure 4.2, those same clusterings can be seen annotated with the bias points. Each point that represents a biased word is represented by the number ‘1.’



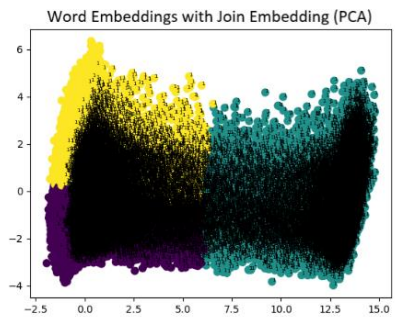
a)



b)



c)



d)

Figure 4.2.: a-d show the same plots as Figure 4.1., but with the points which received a bias judgement of ‘1’ (biased) annotated.

As shown in the above graphs, the bias points are distributed fairly evenly throughout the clusters and concentrated in the center of the graphs. This is the truest for 4.2. b), the graph of the word embeddings without bias probabilities or v (join embedding), while in the other graphs the bias points

do migrate so they are no longer concentrated in the exact center of the graph, but move a bit with the clusters.

As expected, the word embeddings without the bias probabilities or the v (join embedding) cannot be separated into distinct clusters and the points are plotted randomly (figure 4.1. a). On the other hand, the word embeddings that include both the bias probabilities and the v (join embedding) separate cleanly into two neat clusters, though not without a bit of overlap (figure 4.1. c). Both of these sets of word embeddings clustered much better with less overlap after the implementation of PCA and PCA showed marked improvement on the clustering of the word embeddings plus the join embedding as it was the only set able to cluster into three clear clusters (Figure 4.1. d).

The clusters in all of the plots are not separated by biased- or neutral-judged words, but rather by some other features. This makes sense considering there were about 512 features other than the bias probabilities by which the data was described. The biased-judged words are, however, in every example concentrated in the middle of the plot, as seen in figure 4.2. This may suggest that despite being in separate clusters, the features that denote bias influence the biased-judged words to cluster close to each other.

It is interesting that the only plot for which 3 clusters was optimal was the one for the word embeddings with the join embedding and PCA (Figure

4.1. d). The other three sets of data were most optimally clustered with 2 clusters. It is difficult to tell from the available data why that is and what the three clusters represent, however it seems possible that the three clusters may correlate with type of bias.

To investigate the possibility of correlation between the three clusters and three types of bias, contextualized instances of words were randomly selected from each of the clusters for analysis. Examples 1a) and 1b) show the biased and neutralized context for an instance of the word "demonstrates" in cluster 1.

1) Cluster 1: "demonstrates"
    a. In his book "Political Parties," written in 1911, Robert Michels *demonstrates* that most representative systems deteriorate towards an oligarchy.
    b. In his book "Political Parties," written in 1911, Robert Michels *argues* that most representative systems deteriorate towards an oligarchy.

The difference here between "demonstrates" and "argues" is slight, but significant. "Demonstrates" entails proof, and a judgement that whatever

it is that is being demonstrated is factual. In this context, because it is difficult to prove that most representative systems deteriorate towards an oligarchy, and any proof or evidence towards that conclusion is highly subject to individual interpretation, the use of "demonstrates" is inappropriate. Instead, the term "argues" is more neutral because it does not pass any judgement on the factuality of what is being argued. Epistemological bias was defined in 1.1. as modifying the believability of a proposition, which is exactly what the term "demonstrates" does in this example. This is also very similar to the example of epistemological bias in section 1.1. where the biased term "exposé" was changed to a more neutral term, "statements." An even more subtle change from biased to more neutral is listed in 2).

2) Cluster 1: "cemented"
    a. Such relentless violence *cemented* the fearsome reputation of the gestapo as the Nazis' secret police.
    b. Such relentless violence *did much to add to* the fearsome reputation of the gestapo as the Nazis' secret police.

While the change from "cemented" to "did much to add to" in example 2) may seem like a relatively arbitrary change, the term "cemented" means to firmly establishing something and has the connotation that whatever

has been cemented is permanent and unshakeable. On the other hand, "adding to" something has no such connotations and fits much better with Wikipedia's NPOV policy. Again, like the other example from cluster 1 in the first example, this example also modifies the believability of the sentence (in this case, connoting that it must have been whatever relentless violence is being referred to here that solidified the fearsome reputation and not anything else that the gestapo did—a biased sentiment as there are many reasons why the gestapo had a fearsome reputation). So this example also seems to support the proposal that cluster 1 correlates with epistemological bias.

Examples 2) and 3) show biased and neutral context for instances of words from cluster 2.

3) Cluster 2: "bitch"
    a. Kahla is a *bitch* town in the Saale-Holzland district, in Thuringia, Germany.
    b. Kahla is a town in the Saale-Holzland district, in Thuringia, Germany.

The use of "bitch" as an adjective describing the town in 4a) is a clear example of framing bias. Framing bias has to do with an individual point of view that is not objective. The term "bitch" is a subjective word linked with

a particular point of view. In this case, term “bitch” shows the author’s opinion that they do not think highly of the town Kahla. Although it is unclear whether the author believes that the actual town is a bitch or that the town is full of bitches, the negative judgement is unmistakable. The term paints a biased, negative view of the town in the same way that the term “pilfered-from” in the example in section 1.1. paints a negative view of the game that it is referring to.

4) Cluster 2: “giants”
    a. He also worked with heavy metal *giants* Metallica, on a two day concert that was held in Berkeley, California, with the San Francisco symphony.
    b. He also worked with heavy metal *band* Metallica, on a two day concert that was held in Berkeley, California, with the san Francisco symphony.

Like “bitch” was used in 3a) to frame the town of Kahla negatively, the term “giants” in 4a) is used to frame the band Metallica in a positively. “Giants” here does not literally mean large, but figuratively means that the band Metallica largely—and positively—influential. Despite Metallica’s huge popularity, whether their influence was positive or negative is a matter

of personal opinion which is what makes the term “giants” biased in the context of 4a).

The following two examples show contexualized instances of words from cluster 3.

5) Cluster 3: “baby”
    a. Morgause concocts a potion to help Morgaine abort her *baby*.
    b. Morgause concocts a potion to help Morgaine abort her *pregnancy*.

Demographic bias mostly contains social bias, or bias that favors one group’s ideology over another’s. In the case of example 5a), the term “baby” is biased here because there is debate about when life truly begins and when a fetus should be granted “personhood.” Because people have differing opinions on when life begins (at conception, after the heart starts beating, after brain activity can be detected, etc.), whereas the term “pregnancy” is much more straight-forward and easily defined, it is reasonable to say that this is an example of demographic bias.

6) Cluster 3: “queers”

    a. Black *queers* and women were sometimes censured outright in an effort to merge black identity with masculinity.

    b. Black *gays* and women were sometimes censured outright in an effort to merge black identity with masculinity.

While the term “queer” is in the process of being reclaimed by the LGBTQ+ community, its use is still quite contention and its plural version “queers” is still considered to be a dehumanizing slur. The term “gays” does not have the same offensive connotations that “queers” does which makes it a more neutral word choice. Because the term “queers” in 6a) is negative against a certain group of people, this example also seems to be an instance of demographic bias.

Though these are only a couple of examples from the clusters, many of the instances of biased words removed from these clusters support the conclusion that the clusters correlate with types of bias. This is significant because it means that not only is the model learning to detect and remove biases, but it is also capable of classifying bias, even without being explicitly trained to do so. Of course, not all of the words in the clusters are biased, but

this reinforces the theory that the neutralization model is learning biased words based on their context. The clusters are based on linguistic features other than bias which help inform, once the biased word has been identified, the type of bias.

4.1.1. Most Frequently Biased-Judged Words

To further investigate the three-way split of the data in the clustering of the word embeddings with the join embedding and PCA, the top 10 words that were most frequently judged as biased were extracted from each of the clusters. They are shown in Table 4.1.

Articles such as “the,” “an” and “a” as well as helping verbs such as “is” and “was” were removed from these lists. Because these articles are deleted along with another biased word, the system judges them as biased in certain contexts. However, these tokens do not usually indicate bias but are removed along with a biased word in order to satisfy grammaticality. Suffixes (such as ##ly and ##s) and punctuation (such as a hyphen) were also not included as these tokens are similarly likely to be deleted along with the actual biased word and are unlikely to be biased on their own.

Cluster 1 (Epistemological Bias?)	**Cluster 2 (Framing Bias?)**	**Cluster 3 (Demographic Bias?)**
many	comedienne	many
some	popular	only
often	only	some
most	prestigious	other
other	great	even
he	famous	often
also	controversial	neutral
however	passing	their
only	beautiful	much
it	terrorist	conservative

Table 4.1.: Top ten words that are most frequently judged as biased. Extracted from the clustering of the word embeddings with the join embedding and PCA.

There is some overlap between the most common words between the clusters. Again, this is possible because instances of the same words were kept separate in order to account for context. Cluster 1 and cluster 3 have four other words in common, (“many,” “some,” “often” and “other”). All three clusters have the word “only.”

Because high-frequency words are likely to appear frequently anywhere, not just in a biased context, it is unclear if the words in Table 4.1. are more likely to be judged as biased or neutral. To investigate this, both the neutral- and biased-judged instances were counted. The results of this can be seen in Figure 4.3.

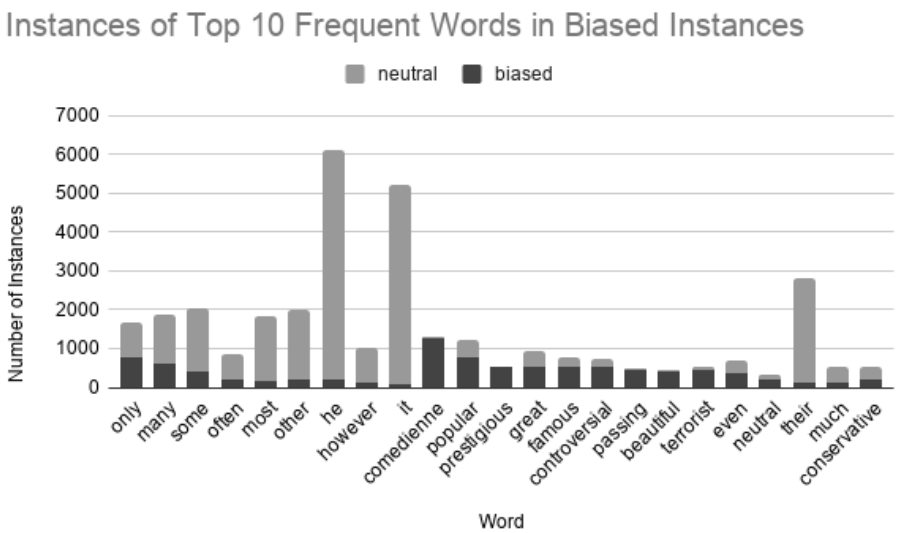


Figure 4.3.: Biased- and neutral-judged instances of the top ten words that are most frequently judged as biased from the clustering of the word embeddings with the join embedding and PCA.

Figure 4.3. shows that while some of the words appear to simply be high-frequency words that are more likely to be judged as neutral (“he,” “it,” “their”), quite a few of the most-frequently biased-judged words have relatively low-frequency and appear mostly in biased-contexts (“comedienne,”

“prestigious,” “passing,” “beautiful,” “terrorist”). The top ten words judged most frequently as biased from cluster 1 appear to mostly be generally high-frequency words with more instances of neutral-judgement than biased-judgement, the ones from cluster 2 appear to be mostly biased-judged and rarely occur in neutral contexts, while cluster 3 contains a mix of both. This seems to also support the hypothesis that the three clusters are divided by features which are more common with different types of bias. Epistemological bias, which seems to be represented by cluster 1, utilizes subtle linguistic features to modify the believability of a statement. It makes sense then that the words in cluster 1 that are most often judged to be biased are high-frequency words that despite being usually judged as neutral can be used as a hedge or elsewise subtly change the connotation of a statement. An example of this is shown below with one of the contextualized instances where “many” was judged to be biased.

1. *Many* business people like to say, "the worst day on the golf course is better than the best day at work."
2. *Some* business people like to say, "the worst day on the golf course is better than the best day at work."

While “many” is more likely to be judged as neutral than as biased, examples where it does receive a biased-judgement suggest that it is not its high-frequency alone that occasionally earns it a biased judgement, rather there are many genuine instances where the word is biased. However, “many” is not the most extreme example of a high-frequency word which appears more as neutral- than biased-judge. “He” and “it” are judged to be neutral 96.8% and 98.1% of the time, respectively. Despite being unlikely to receive a biased-judgement, the biased judgements received by these words are also not anomalies, but often do represent actual instances of bias. The following examples show a biased-judged instance of “he.”

1. The president appoints the prime minister, but it is expected that *he* will select the leader of the largest party/coalition.
2. The president appoints the prime minister, but it is expected that *the president* will select the leader of the largest party/coalition.

In this example, “he” is biased because it presumes the gender of “the president.” Many of the biased-judged instances of “he” appear in instances similar to the one above. However, “it” does not show the same tendency. Most of the biased-judged instances of “it” from cluster 1 do not actually seem to be biased, rather represent corrections made for ease-of-understanding

(changing “it” to the name of the thing “it” describes). Yet, there are still some examples of “it” representing bias.

1. Fox and *it* supporters, however, contend that what left-leaning observers like fair perceive as a conservative bias is, in fact, lack of a liberal bias.
2. Fox and *their* supporters, however, contend that what left-leaning observers like fair perceive as a conservative bias is, in fact, lack of a liberal bias.

Again, despite the false bias-judgements that “it” receives, there are genuine examples of “it” being biased and most of the false bias-judgements are the result of non-bias related change between the source sentence and the target sentence which suggests that in this case too, the biased-judged instances of “it” are not a result of the high frequency of the word, rather the context of the word.

Interestingly, looking at Table 4.1., part of speech seems that it may play a significant role in the chance of a word being biased. While bias occurs in all parts of speech, the top frequency chart mostly consists of adjectives, quantifiers, and adverbs. (It is worth noting that suffixes also generally appear on adjectives and adverbs.) As these are all words that are generally used to

describe something, it makes sense that these parts of speech would be subject to the most bias.

4.1.2. Cosine Similarity

In order to look a bit closer at the instances of bias words in each cluster, cosine similarity was performed. Cosine similarity measures the cosine of angle between vectors and can be used in NLP to predict words similar in meaning or context. The equation to calculate cosine similarity is shown below.

$$similarity = \cos(\theta) = \frac{A \times B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad 4.1$$

Because most common words shared between the clusters such as “only” were high in frequency, most of the words rated with the highest similarity were other instances of the same word. In order to avoid this and attempt to gain more interesting linguistic information from the results of the

cosine similarity, words that appeared across all three clusters but had a low frequency were compared.

There were only thirty-seven instances of the word “offensive” that had a biased rating of 1 (biased). Six of these instances were in cluster 1, twenty-three instances were in cluster 2, and eight instances were in cluster 3. One instance of the word “offensive” was taken from each cluster and cosine similarity was performed. Tables 4.2. – 4.4. show the top ten most similar words to each instance according to cosine similarity.

All of the top ten most similar words for the instance of “offensive” from cluster 1 are also adjectives that receive a biased judgement of 1 (biased). Interestingly, they are not extremely close in meaning or even connotation. “Embarrassing,” “worse,” “ridiculous” and “deteriorated” are perhaps the closest in meaning to “offensive” and may be used in some similar contexts however most of the other similar words have much more positive connotations such as “advanced,” “outstanding,” and “prestigious.” Still, even these words with more positive connotations can be used in the same type of sentences, just to a different effect. This shows that the words are represented by more than just their meanings in the word embeddings, but that the features and context that the words appear in are also affecting how the words are represented.

Cosine Similarity	Word	Bias Judgement (biased: 1, unbiased: 0)
0. 8434	embarrassing	1
0. 8412	worse	1
0. 8361	advanced	1
0. 8342	blockbuster	1
0. 8323	outstanding	1
0. 8293	prestigious	1
0. 8290	essential	1
0. 8287	ridiculous	1
0. 8283	deteriorated	1
0. 8266	free	1

Table 4.2.: Cosine Similarity of an instance the word "offensive" from cluster 1.

1a) and 1b) below show the context of the same instance of "offensive" from table 4.2.

1) Cluster 1: "offensive"
    a. These controversies led TSR to remove many potentially *offensive* references and artwork from the game line upon release of a D&D 2nd edition.

b. These controversies led TSR to remove many potentially *controversial* references and artwork from the game line upon release of a D&D 2nd edition.

As anticipated, most of the similar words could be put in this context and despite changing the meaning of the sentence it would still result in a grammatically correct and interpretable sentence. The one adjective that would probably make the least sense if it were to be switched out is "blockbuster," though it is not too hard to understand why "blockbuster" may have received a high cosine similarity. Blockbuster films often do so well because they are controversial or offensive to some viewers which creates hype around them. Hence, while "blockbuster" may not immediately seem to make sense as having a high cosine similarity with "offensive," it is reasonable to think that the words likely occur in similar contexts, despite not making sense in this specific context together.

Table 4.3. shows the top ten most similar words to an instance of the word "offensive" from cluster 2. Amongst the similar words in cluster 2, there are two other biased instances of the word "offensive." Similar to cluster 1, most of the similar words are also biased instances of adjectives with the exception of a biased instance of the verb "reaction." Again, the presence of words such as "evergreen" which based on meaning alone seems to have very

little in common with the word “offensive,” suggests that there are a variety of features that contribute to the similarity scores.

Cosine Similarity	Word	Bias Judgement (biased: 1, unbiased: 0)
0.9709	offensive	1
0.9570	evergreen	1
0.9570	offensive	1
0.9490	authentic	1
0.9466	antique	1
0.9463	flawless	1
0.9445	true	1
0.9438	conservative	1
0.9438	reaction	1
0.9435	inaccurate	1

Table 4.3.: Cosine Similarity of an instance the word “offensive” from cluster 2.

2a) and 2b) show the context of this instance of “offensive” from cluster 2.

2) Cluster 2: “offensive”

   a. She also commented on the outfit's *offensive* nature, "...it is certainly no disrespect to anyone that is vegan or vegetarian.

   b. She also commented on the outfit's nature, "...it is certainly no disrespect to anyone that is vegan or vegetarian.

The reference to “vegan or vegetarian,” or perhaps to “nature” in the context sentence may explain the high cosine similarity with words such as “evergreen” and “conservative” since such words are likely to be used together in different contexts. Like the example from cluster 1, most of the similar words could be substituted in the sentence with a change in meaning but not grammaticality.

The cosine similarity and contextualized words from clusters 1 and 2 show very similar tendencies. The instance of “offensive” from cluster 3 is a bit more unique. Table 4.4. shows that although the only instance of the word “offensive” from cluster 3 does receive a judgement of biased, the top 10 most similar words according to cosine similarity are mostly unbiased. Additionally, two of those are unbiased instances of the same word, “offensive”. This suggests that while “offensive” received a higher than

average probability for being biased, it may not actually be the source of bias.

Cosine Similarity	Word	Bias Judgement (biased: 1, unbiased: 0)
0. 7187	offensive	0
0. 6882	offensive	0
0. 6529	iconic	1
0. 6324	permission	0
0. 6311	secret	0
0. 6251	extinct	0
0. 6126	financially	0
0. 6119	adorable	1
0. 6067	unknown	0
0. 6057	smells	0

Table 4.4.: Cosine Similarity of an instance the word "offensive" from cluster 3.

The context for this instance of the word "offensive" is shown in example 3).

3) Cluster 3: “some”

    a. The state flag used from 1956 to 2001 (see below) featured a prominent confederate battle flag, which *some* of the state's residents found offensive […].

    b. The state flag used from 1956 to 2001 (see below) featured a prominent confederate battle flag, which *a majority* of the state's residents found offensive […].

Context shows that “offensive” is not the biased word from the source sentence but the actual biased word is “some,” which is neutralized to “a majority.” Considering this, it seems reasonable that most of the similar words were judged to be unbiased despite “offensive” being judged as bias. It is unlikely that this judgement of the word “offensive” being biased would cause a problem in the actual neutralization model as “some” is assigned a higher bias probability than “offensive” in this sentence. Additionally, it seems the model is able to make use of other features from the v vector that human annotators looking at a binary bias judgement are not able to make use of.

4.2. Clustering of Small Random Sample

Although the entire set was too large to use UMAP, a small subsection of the data was pulled so that the PCA clustering and UMAP clustering could be compared. 1000 words from the dataset were randomly selected and clustered. Because the sample set was so small and biased-judged words make up a small portion of the data (biased-judged words only make up approximately 6.9% of the data), 500 biased-judged and 500 neutral-judged words were randomly chosen.

The first thing that is immediately noticeable when comparing the clustering of UMAP and PCA is that UMAP more efficiently creates two very distinct clusters, while the clusters formed by PCA have points in between the clusters that almost touch. This seems to suggest that UMAP would be better suited for a clustering task on this data than PCA. Additionally, unlike the clusters for the entire data set, both the plain word embeddings and the word embeddings with the join embedding clustered optimally with 2 clusters with and without the use of a dimensionality reduction algorithm.

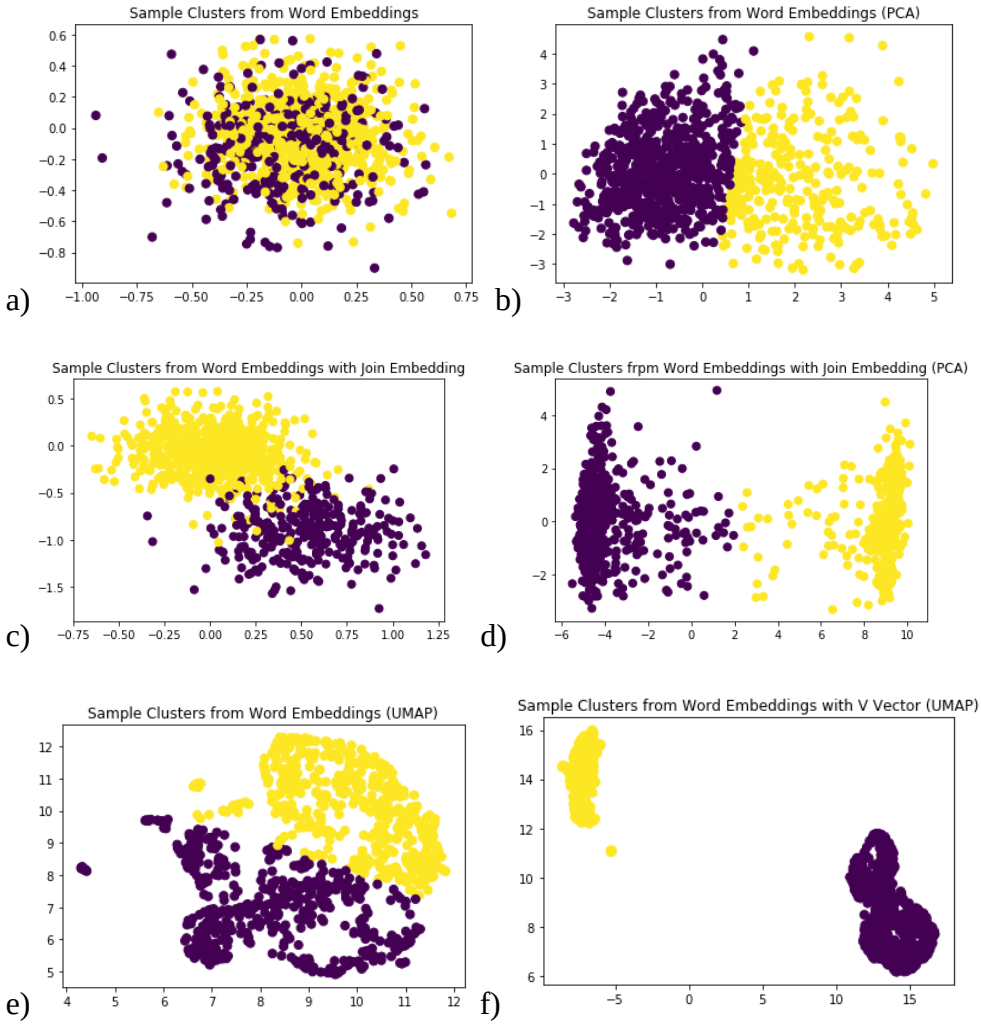


Figure 4.4.: Clustering of a small section (1,000 data points) of the data without using a dimensionality reduction algorithm, with the use of PCA, and with the use of UMAP.

Table 4.5. shows the distribution of the biased-judged and neutral-judged words from the clustering of the small sample of data, as shown in Figure 4.4. The distributions show similar patterns between the clusters of

just the plain word embeddings and clusters of the word embeddings with the join embedding.

	Biased-Judged	**Neutral-Judged**
Word Embeddings	48.2% cluster 1 51.8% cluster 2	9.2% cluster 1 90.8% cluster 2
Word Embeddings (PCA)	51.6% cluster 1 48.4% cluster 2	91.6% cluster 1 8.4% cluster 2
Word Embeddings (UMAP)	37.6% cluster 1 62.4% cluster 2	64.4% cluster 1 35.6% cluster 2
Word Embeddings + Join Embedding	64.4% cluster 1 35.6% cluster 2	0% cluster 1 100% cluster 2
Word Embeddings + Join Embedding (PCA)	35.6% cluster 1 64.4% cluster 2	100% cluster 1 0% cluster 2
Word Embeddings + Join Embedding (UMAP)	61.4% cluster 1 38.6% cluster 2	0% cluster 1 100% cluster 2

Table 4.5.: Distribution of biased- and neutral-judged words in the small subsection of data.

With the inclusion of the join embedding, regardless of whether a dimensionality reduction algorithm was used or not, the neutral-judged words

were consistently always clustered together in one cluster. The biased-judged words did not all cluster together, but split with 60-65% of the biased-judged words in their own cluster and 35-40% of the biased-judged words in the same cluster as the neutral-judged words.

The clustering of the neutral-judged words together and the majority of the biased-judged words in their own cluster, it is possible that with more data to work with, clustering may be further improved and the word embeddings with the join embedding may cluster biased-judged and neutral-judged words in their own, separate clusters. While this may seem doubtful, given that the clustering of the entire dataset did not cluster in this way, it is possible that the entire dataset is too noisy. In future studies, it may be worth sampling a larger section of the dataset with an equal number of biased-judged and neutral-judged words to investigate if the data clusters more like the entire dataset or more like the small sample.

4.3. Significance of Results

This study found three major things in regards to bias. First, it was found that the word embeddings from Pryzant et al. (2019) were improved by

the implementation of a dimensionality reduction algorithm, in this case, PCA. With the use of PCA, the H' word embeddings clustered better than without and actually clustered according to bias type, suggesting that with some adjusting, the modular model would be able to not only detect and neutralize bias, but also classify bias. Additionally, it was shown through the clustering of a small subsection of data that UMAP particularly performed very well and showed that it may be beneficial

The second discovery that this study made was that while bias can occur in almost any part of speech, it seems to most often occur in adjectives, quantifiers, and adverbs. While this may seem intuitive because as descriptive words these parts of speech are more likely to contain opinion, it is significant that while this is well-represented in the word embeddings and the modular model it does not cause the model to overgeneralize and only label and neutralize descriptive words.

Finally, this study showed through the cosine similarity analysis that the module model is much more sensitive than expected. Bias was not only determined based on part of speech and the meaning of the words, but the v vector seems to successfully encode for a variety of other features that assists context comprehension and bias detection and neutralization within the model.

5. Conclusion

Building off of past research on bias detection and bias neutralization, this study was able to use clustering and dimensionality reduction to analyze the word embeddings from the novel modular model (Pryzant et al. 2019). It was found that the mysterious *v* vector (join embedding) used in the model did accomplish what Pryzant et al. claimed as the word embeddings that included information from the *v* vector not only clustered the best, but also seemed to cluster according to features that could help the model classify type of bias, despite not being explicitly trained to do so.

While it is clear from this study that a great number of factors are involved in bias and its representation in word embeddings and its place in NLP models, it is still difficult to pin down the exact essence of bias. Such a thing is likely impossible, given the ever-changing nature of language and the inherent subjectivity of humans. A large number of features come together to determine the meaning and connotation that it is difficult to pin down the exact cause of bias.

This study, while fairly simple, has the potential to be a solid foundation on which to continue research on bias in NLP. One possibility for future improvement is to compare a multiple variety of clustering algorithms

as well as several dimensionality reduction algorithms such as tSNE and UMAP on the entirety of the dataset. UMAP, particularly, appears well-suited to the clustering of this data as seen in Section 4.2. Another option would be decreasing the number of neutral-judged words in the dataset so that an equal number of biased-judged and neutral-judged words could be clustered, potentially making the data less noisy and improving clustering performance.

Additionally, future research could experiment more with the *v* vector from the modular model. A more in-depth study may extract the features and feature labels from the *v* vector in order to determine what features affect the bias probabilities the most. Another interesting possibility could be to replace or enhance the *v* vector with Global Style Tokens (GST), which would allow a better view of what features most contribute to the biasedness or neutrality of words.

References

Arthur, D. and Vassilvitskii, S. (2007). K-means++: The Advantages of Careful Seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027-2035.

Basta, C., Costa-jussà, M., and Casas, N. 2019. Evaluating the Underlying Gender Bias in Contextualized Word Embeddings. In *Proceedings of the 1st ACL Workshop on Gender Bias for Natural Language Processing.*

Bolukbasi, T., Chang, K.-W., Zou, J. Y., Saligrama, V., and Kalai, A. T. 2016. Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings. In *Advances in neural information processing systems*, pp. 4349–4357.

Bordia, S. and Bowman, S. 2019. Identifying and Reducing Gender Bias in Word-Level Language Models. In *arXiv:1904.03035v1 [cs.CL]*.

Caliskan, A., Bryson, J., and Narayanan, A. 2017. Semantics Derived Automatically from Language Corpora Contain Human-like Biases. In *Science 356(6334)*:183–186. http://opus.bath.ac.uk/55288/

Chaloner K. and Maldonado, A. 2019. Measuring Gender Bias in Word Embeddings across Domains and Discovering New Gender Bias

Word Categories. In *Proceedings of Association for Computational Linguistics (ACL) Workshop on Gender Bias in Natural Language Processing*, pages 25–32.

Ethayarajh, K., Duvenaud, D., and Hirst, G. 2019. Understanding Undesirable Word Embedding Associations. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1696–1705.

Eriksson, L. 2018. What Is Principal Component Analysis (PCA) and how is it used? [Blog Post] Retrieved from *https://blog.umetrics.com/what-is-principal-component-analysis-pca-and-how-it-is-used.*

Feathers, T. 2019, August. Flawed Algorithms are Grading Millions of Students' Essays. Retrieved from https://www.vice.com/en_us/article/pa7dj9/flawed-algorithms-are-grading-millions-of-students-essays.

Font, J. and Costa-jussà, M. 2019. Equalizing Gender Biases in Neural Machine Translation with Word Embeddings Techniques. *arXiv preprint arXiv:1901.03116.*

Garg, N., Schiebinger, L., Jurafsky, D., and Zou, J. 2018. Word Embeddings Quantify 100 years of Gender and Ethnic Stereotypes. In *Proceedings of the National Academy of Sciences, 115(16): E3635–E3644.*

Gonen, H., and Goldberg, Y. 2019. Lipstick on a Pig: De-biasing Methods Cover up Systematic Gender Biases in Word Embeddings but do not Remove them. In *North American Chapter of the Association for Computational Linguistics (NAACL).*

Hovy, D. and Spruit, S. 2016. The Social Impact of Natural Language Processing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, volume 2, pages 591– 598.

Jurgens, D., Tsvetkov, Y., and Jurafsky, D. 2017. Incorporating Dialectal Variability for Socially Equitable Language Identification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics,* volume 2, pages 51–57.

Kaneko M. and Bollegala, D. 2019. Gender-Preserving De-biasing for Pre-trained Word Embeddings. *arXiv preprint arXiv:1906.00742.*

Kay, M., Matuszek, C., and Munson, S. 2015. Unequal Representation and Gender Stereotypes in Image Search Results for Occupations. In *Human Factors in Computing Systems, pages 3819– 3828.*

Kiritchenko, S. and Mohammad, S. 2018. Examining Gender and Race Bias in Two Hundred Sentiment Analysis Systems. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 43–53, New Orleans, Louisiana. Association for Computational Linguistics.

MacKay, David (2003). Chapter 20, An Example Inference Task: Clustering (PDF). In *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, pages 284-292. ISBN 978-0-521-64298-9. MR 2012999.

Malvina N., van Noord, R., and van der Goot, R. 2019. Fair is Better than Sensational: Man is to Doctor as Woman is to Doctor. *arXiv preprint arXiv:1905.09866.*

Manzini, T., Lim, Y. C., Tsvetkov, Y., and Black, A. W. 2019. Black is to Criminal as Caucasian is to Police: Detecting and Removing Multiclass Bias in Word Embeddings. In *North American Chapter of the Association for Computational Linguistics (NAACL) 2019.*

May, C., Wang, A., Bordia, S., Bowman, S., and Rudinger, R. 2019. On Measuring Social Biases in Sentence Encoders. *arXiv preprint arXiv:1903.10561.*

McInnes, L., Healy, J., and Melville, J. 2018. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv:1802.03426v2.*

Pant, K., Dadu, T., and Mamidi, R. 2020. Towards Detection of Subjective Bias using Contextualized Word Embeddings. In *WWW '20: Companion Proceedings of the Web Conference 2020,* pp. 75-76.

Park, J., Shin, J., and Fung, P. 2018. Reducing Gender Bias in Abusive Language Detection. In *Empirical Methods in Natural Language Processing*.

Pryzant, R., Martinez, R. D., Dass, N., Kurohashi, S., Jurafsky, D., and Yang, D. 2019. Automatically Neutralizing Subjective Bias in Text. *arXiv preprint arXiv:1911.09709.*

Recasens, M., Danescu-Niculescu-Mizil, C., and Jurafsky, D. 2013. Linguistic Models for Analyzing and Detecting Biased Language. In *Proceedings of Association for Computational Linguistics (ACL),* pp. 1650–1659.

Rudinger, R., Naradowsky, J., Leonard, B., and Van Durme, B. 2018. Gender bias in Coreference Resolution. In *North American Chapter of the Association for Computational Linguistics (NAACL).*

Sun, T., Gaut, A., Tang, S., Huang, Y., ElSherief, M., Zhao, J., Mirza, D., Belding, E., Chang, K., and Wang, W. 2019. Mitigating Gender Bias in Natural Language Processing: Literature Review. *arXiv:1906.08976 http://arxiv.org/abs/1906.08976.*

Swinger, N., De-Arteaga, M., Heffernan, N., Leiserson, M., and Kalai, A. 2019. What are the Biases in my Word Embedding? In *Proceedings of the AAAI/ACM Conference on Artificial Intelligence, Ethics, and Society (AIES).*

Vanmassenhove, E., Hardmeier, C., and Way, A. 2018. Getting Gender Right in Neural Machine Translation. In *Empirical Methods in Natural Language Processing*.

Wikipedia: Neutral Point of View. Retrieved from https://en.wikipedia.org/wiki/Wikipedia:Neutral_point_of_view.

Wu, X. and Zhang, X. 2016. Automated Inference on Criminality using Face Images. *arXiv:1611.04135.*

Yano, T., Resnik, P., and Smith, N. 2010. Shedding (a Thousand Points of) Light on Biased Language. In *NAACL Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk.*

Wallace, B. and Paul, M. 2016. Jerk or Judgemental? Patient Perceptions of Male Versus Female Physicians in Online Reviews. *Association for the Advancement of Artificial Intelligence (www.aaai.org).*

Zhao, J., Wang, T., Yatskar, M., Ordonez, V., and Chang, K. 2017. Men also like Shopping: Reducing Gender Bias Amplification using Corpus-Level Constraints. *arXiv preprint arXiv:1707.09457.*

Zhao, J., Wang, T., Yatskar, M., Ordonez, V., and Chang, K. 2018a. Gender Bias in Coreference Resolution: Evaluation and De-biasing Methods. In *North American Chapter of the Association for Computational Linguistics (NAACL).*

Zhao, J., Zhou, Y., Li, Z., Wang, W., and Chang, K. 2018. Learning Gender-Neutral Word Embeddings. In *Empirical Methods in Natural Language Processing*, *2018b.*

Appendix

Table A: Features from Recasense et al. (2013). The astrisk (*) indicates what features contributed the most in their bias detection system.

ID	Feature	Value	Description
1*	Word	<string>	Word *w* under analysis.
2	Lemma	<string>	Lemma of *w*.
3*	POS	{NNP, JJ, ...}	POS of *w*.
4*	POS – 1	{NNP, JJ, ...}	POS of one word before *w*.
5	POS – 2	{NNP, JJ, ...}	POS of two words before *w*.
6*	POS + 1	{NNP, JJ, ...}	POS of one word after *w*.
7	POS + 2	{NNP, JJ, ...}	POS of two words after *w*.
8	Position in sentence	{start, mid, end}	Position of *w* in the sentence (split into three parts).
9	Hedge	{true, false}	*w* is in Hyland's (2005) list of hedges (e.g., *apparently*).
10*	Hedge in context	{true, false}	One/two words) around *w* is a hedge (Hyland, 2005).
11*	Factive verb	{true, false}	*w* is in Hooper's (1975) list of factives (e.g., *realize*).
12*	Factive verb in context	{true, false}	One/two word(s) around *w* is a factive (Hooper, 1975).
13*	Assertive verb	{true, false}	*w* is in Hooper's (1975) list of assertives (e.g., *claim*).
14*	Assertive verb in context	{true, false}	One/two word(s) around *w* is an assertive (Hooper, 1975).
15	Implicative verb	{true, false}	*w* is in Karttunen's (1971) list of implicatives (e.g., *manage*).
16*	Implicative verb in context	{true, false}	One/two word(s) around *w* is an implicative (Karttunen, 1971).
17*	Report verb	{true, false}	*w* is a report verb (e.g., *add*).
18	Report verb in context	{true, false}	One/two word(s) around *w* is a report verb.
19*	Entailment	{true, false}	*w* is in Berant et al.'s (2012) list of entailments (e.g., *kill*).
20*	Entailment in context	{true, false}	One/two word(s) around *w* is an entailment (Berant et al., 2012).
21*	Strong subjective	{true, false}	*w* is in Riloff and Wiebe's (2003) list of strong subjectives (e.g., *absolute*).
22	Strong subjective in context	{true, false}	One/two word(s) around *w* is a strong subjective (Riloff and Wiebe, 2003).
23*	Weak subjective	{true, false}	*w* is in Riloff and Wiebe's (2003) list of weak subjectives (e.g., *noisy*).
24*	Weak subjective in context	{true, false}	One/two word(s) around *w* is a weak subjective (Riloff and Wiebe, 2003).
25	Polarity	{+, –, both, ...}	The polarity of *w* according to Riloff and Wiebe (2003), e.g., *praising* is positive.
26*	Positive word	{true, false}	*w* is in Liu et al.'s (2005) list of positive words (e.g., *excel*).
27*	Positive word in context	{true, false}	One/two word(s) around *w* is positive (Liu et al., 2005).
28*	Negative word	{true, false}	*w* is in Liu et al.'s (2005) list of negative words (e.g., *terrible*).
29*	Negative word in context	{true, false}	One/two word(s) around *w* is negative (Liu et al., 2005).
30*	Grammatical relation	{root, subj, ...}	Whether *w* is the subject, object, root, etc. of its sentence.
31	Bias lexicon	{true, false}	*w* has been observed in NPOV edits (e.g., *nationalist*).
32*	Collaborative feature	<numeric>	Number of times that *w* was NPOV-edited in the article's prior history / frequency of *w*.

Algorithms 1 – 5 describe the UMAP algorithm (McInnes et al. 2018)

Algorithm 1 UMAP algorithm

function UMAP(X, n, d, min-dist, n-epochs)
  for all $x \in X$ **do**
    fs-set$[x] \leftarrow$ LocalFuzzySimplicialSet(X, x, n)
  top-rep $\leftarrow \bigcup_{x \in X}$ fs-set$[x]$ ▷ We recommend the probabilistic t-conorm
  $Y \leftarrow$ SpectralEmbedding(top-rep, d)
  $Y \leftarrow$ OptimizeEmbedding(top-rep, Y, min-dist, n-epochs)
  return Y

Algorithm 2 Constructing a local fuzzy simplicial set

function LocalFuzzySimplicialSet(X, x, n)
  knn, knn-dists $\leftarrow$ ApproxNearestNeighbors(X, x, n)
  $\rho \leftarrow$ knn-dists[1] ▷ Distance to nearest neighbor
  $\sigma \leftarrow$ SmoothKNNDist(knn-dists, n, ρ) ▷ Smooth approximator to knn-distance
  fs-set$_0 \leftarrow X$
  fs-set$_1 \leftarrow \{([x, y], 0) \mid y \in X\}$
  for all $y \in$ knn **do**
    $d_{x,y} \leftarrow \max\{0, \text{dist}(x, y) - \rho\}/\sigma$
    fs-set$_1 \leftarrow$ fs-set$_1 \cup ([x, y], \exp(-d_{x,y}))$
  return fs-set

Algorithm 3 Compute the normalizing factor for distances σ

function SmoothKNNDist(knn-dists, n, ρ)
  Binary search for σ such that $\sum_{i=1}^{n} \exp(-(\text{knn-dists}_i - \rho)/\sigma) = \log_2(n)$
  return σ

Algorithm 4 Spectral embedding for initialization

function SpectralEmbedding(top-rep, d)
  $A \leftarrow$ 1-skeleton of top-rep expressed as a weighted adjacency matrix
  $D \leftarrow$ degree matrix for the graph A
  $L \leftarrow D^{1/2}(D - A)D^{1/2}$
  evec $\leftarrow$ Eigenvectors of L (sorted)
  $Y \leftarrow$ evec$[1..d + 1]$ ▷ 0-base indexing assumed
  return Y

Algorithm 5 Optimizing the embedding

function OPTIMIZEEMBEDDING(top-rep, Y, min-dist, n-epochs)
  $\alpha \leftarrow 1.0$
  Fit Φ from Ψ defined by min-dist
  for $e \leftarrow 1, \ldots,$ n-epochs **do**
    for all $([a, b], p) \in$ top-rep$_1$ **do**
      if RANDOM( ) $\leq p$ **then** ▷ Sample simplex with probability p
        $y_a \leftarrow y_a + \alpha \cdot \nabla(\log(\Phi))(y_a, y_b)$
        for $i \leftarrow 1, \ldots,$ n-neg-samples **do**
          $c \leftarrow$ random sample from Y
          $y_a \leftarrow y_a + \alpha \cdot \nabla(\log(1 - \Phi))(y_a, y_c)$
    $\alpha \leftarrow 1.0 - e/$n-epochs
  return Y

초록

차원 축소를 이용한 편향적 문맥에서의 단어 클러스터링

편향성(Bias)은 어떤 사물, 사람 혹은 그룹 등에서 한쪽에 불균형적으로 주어지는 가중치라고 정의할 수 있다. 최근에는 기계학습에서의 편향성 문제와, 자연언어처리에서 이러한 편향성을 완화하고자 하는 연구에 대한 관심이 늘고 있다. 본 연구의 목표는 언어에 존재하는 편향성을 확인하고 워드 임베딩에서 그 편향성이 어떻게 표현되고 있는지 살펴보는 것이다.

본 연구에서 사용하는 데이터는 Wikipedia Neutrality Corpus(WNC)이고 이에 대한 워드 임베딩으로는 Pryzant et al.(2019)의 편향성을 제거하는 모듈러 모델(modular model)을 이용하였다. 또한 K-means Clustering 을 이용하여 편향성 정보를 포함한 'v' 벡터를 추가하기 전과 후의 워드 임베딩을 시각화하였고,

클러스터링(Clustering) 성능의 개선을 위해 주성분분석(Principal Component Analysis/PCA)을 사용하였다.

본 연구에서는 워드 임베딩에서 언어적 특징에 따라 클러스터링 되는 것과 같이 편향성을 갖는 단어들 역시 편향성의 유형(인식론적 편향성, 프레이밍에 따른 편향성, 인구학적 편향성 등)에 따라서 클러스터링 된다는 것을 확인할 수 있었다. 또한, 워드 임베딩이 모듈러 모델의 고유한 'v' 벡터와 결합할 경우 다양한 언어 정보를 포함하게 되므로, 이러한 연구는 편향성을 인식하고 제거하는 task 뿐만 아니라 문맥(context) 정보를 이해하는 데에도 도움이 될 것이다.

주제어: 편향성, 편향성 제거, 클러스터링, k-means, 차원축소, 주성분분석, 워드 임베딩
학번: 2018-21128