View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Eagle Scholar University of Mary Washington

University of Mary Washington
Eagle Scholar

Student Research Submissions

Spring 5-7-2019

Assessing Bias Removal from Word Embeddings

Clare Arrington

Follow this and additional works at: https://scholar.umw.edu/student_research

 Part of the Computer Sciences Commons

Recommended Citation

Arrington, Clare, "Assessing Bias Removal from Word Embeddings" (2019). *Student Research Submissions*. 268.
https://scholar.umw.edu/student_research/268

This Honors Project is brought to you for free and open access by Eagle Scholar. It has been accepted for inclusion in Student Research Submissions by an authorized administrator of Eagle Scholar. For more information, please contact archives@umw.edu.

Table of Contents

Abstract

As machine learning becomes more influential in our everyday life, we must begin addressing potential shortcomings. A current problem area is word embeddings, a group of frameworks that transform words into numbers, allowing the algorithmic analysis of language. Without a method for filtering implicit human bias from the documents used to create these embeddings, they contain and propagate stereotypes. Previous work has shown that one commonly used and distributed word embedding model trained on articles from Google News contained prejudice between gender and occupation [1]. While unsurprising, the use of biased data in machine learning models only serves to amplify the problem further. Although attempts have been made to remove or reduce these biases, a true solution has yet to be found. Hiring models, tools trained to identify well-fitting job candidates, show the impact of gender stereotypes on occupations. Companies like Amazon have abandoned these systems due to flawed decision-making, even after years of development.

I investigated whether the technique of word embedding adjustments from Bolukbasi 2016 made a difference in the results of an emulated hiring model. After collecting and cleaning a data set of resumes and job postings, I created a model that predicted whether candidates were a good fit for a job based on a training set of resumes from those already hired. To assess differences, I built the same model with different word vectors, including the original and adjusted word2vec embedding. Results were expected to show some form of bias on classification. I conclude with discussion on potential improvements and additional work being done.

1. Introduction

1.1 Word Embeddings

Many models in modern day machine learning (ML) rely on numerical input. While this proves no issue for some sources of data, others like images and text must often be translated into a form that an algorithm can understand. This process of mapping one form to another is known as embedding. For text data, we are able to create many different kinds of embeddings depending on how we choose to separate strings of characters e.g. single character, word, sentence, or full document. No embedding method is decidedly best. In fact, multiple embeddings can be used in the same project to capture different contexts. In this paper, we will focus solely on word embeddings. Once one has chosen what kind of embedding they wish to create, the question becomes how it will be created.

A myriad of techniques for translating text have been developed over the years with ever-increasing complexity. Most simply, one can create a dictionary of all words in a given document and assign a unique number to each. Using this paragraph for instance, we could say that ‘a’ is 1, ‘myriad’ is 2, ‘of’ is 3, and so on. This allows one to check what words are found within a document and where they occured, meaning we can assess the probability of a word’s existence. Since modern natural language processing (NLP) relies heavily on statistics, this method of measuring does well. However, it also produces an ordering that was not originally present within the data. Recalling our example, ‘myriad’ is greater than ‘a’ and less than ‘of’. Additionally, ‘a’ is 1 distance from ‘myriad’ and 2 distances from ‘of’. These features are meaningless and only serve to confuse a model.

To avoid this problem, we can create a one-hot encoding by having a binary column denote the presence of a given word. For example, the phrase, ‘No pain, no gain’, can be represented as [[1 0 0], [0 1 0], [1 0 0], [0 0 1]] where the first column is ‘no’, the second is ‘pain’ and the third is ‘gain’. Note that the third vector is the same as the first, since ‘no’ shows up twice. This keeps each word on equal footing with one another. If we want to focus more on

word frequency, we can use the bag-of-words technique and represent the same phrase from before as {‘no’ : 2, ‘pain’ : 1, ‘gain’ : 1} where each value is the number of times a word appears in a sentence or document.

As ML rose in popularity, we began to see the rise of distributed word representations that aimed to capture semantics. This was inspired by a hypothesis from John Firth that “you shall know a word by the company it keeps”. Word embeddings are created by observing word occurrence patterns, although this is done through are a number of different statistical and machine learning approaches. The resultant embedding is comprised of a set of words and corresponding multidimensional vectors, commonly 300 for standard models. Words with similar vectors can be considered semantically similar. For example, if we visualized the high dimensional space of words, ‘dog’ and ‘cat’ would be closer than ‘dog’ and ‘guitar’.

1.2 Human Bias

While introducing semantics to ML is beneficial for NLP tasks like opinion detection and automatic summarization, it also opens the door to numerous issues of bias. In the fields of machine learning and artificial intelligence (AI), bias has come to mean a few different things. Algorithmic bias is commonly used with regard to the bias-variance tradeoff. In that context, bias is overgeneralization error that arises when a model is too broad and doesn’t fit the data it was originally trained upon. Undeniably, the goal of ML is to have an intelligent model. Nevertheless, one must remember that there are real people on either side, giving input and being affected by the output.

For this paper our attention will be on the influence people can have in the construction of a model. When discussing bias, we will be referring to implicit human bias which can be described as prejudices held by individuals and society that are pervasive yet unconscious. To illustrate this with word occurrences, there are 1.45 billion Google search results for the phrase ‘male nurse’ and .586 billion for ‘female nurse’. With nearly a billion more results for the former, one may think this indicative of a large number of men in nursing when really the percentage of male nurses in the United States is around 9%. There is a base expectation that a nurse would be female, so when searching that clarification is left out. Things believed to be

commonly known are often left unsaid, which can make it difficult to notice when bias is actually occuring [4].

2. Related Work

2.1 Applications

Many companies want to use machine learning to make their lives easier. One way they can do this is by semi-automating the hiring process through various methods. The two machine learning approaches we will touch on are learning to rank (LTR) and classification.

LTR is a subtask of information retrieval, a field dedicated to obtaining relevant sources of information from a document or collection. Learning to rank utilizes machine learning to improve upon standard document ranking, which is primarily computed using exact features from the document. By including ML, we are able to pick up on latent patterns within the ranking. LTR can be used to find resumes that are most applicable to a given job posting [10]. It can also be used by companies like Linkedin for sorting one's entire professional network [5].

I originally looked into performing a ranking task to study the effects of bias in word embeddings, but after reading the common methodologies used I believe these may be more robust than other pure ML methods like classification. The common style of information retrieval is to select documents based upon key features such as skills, years worked and other measurable qualifications. In LTR, this method of selection serves as the first component, obtaining K documents that are believed to be best for the user. Once the top K documents have been selected, this subset is then re-ranked based on latent features provided during training. These can be things like user interaction or whether a candidate was contacted, interviewed, or hired. This creates the problem of potentially introducing human bias, which already exists within company hiring processes. While this is an interesting area of work, I decided I would be unable to explore this avenue due to its scope. Someone wishing to explore this problem would at best have access to internal hiring data or could collect a data set of rankings from a large number of individuals using a tool like Amazon Mechanical Turk.

For these reasons, I redirected my research towards projects that tried to fully automate the hiring process with ML. One case that caught my attention was Amazon's announcement of a cancelled hiring model project that began in 2014. [3] The objective was to create a group of models that would suggest who's resume should be looked at based on 10 years of hired employee resumes. Unfortunately, they made the mistake of creating a feedback loop by basing who they should hire in the future off of who they currently had hired. While biases may not be as prominent in some fields, the lack of diversity in tech means we can end up with models that amplify the problem.

The models confused distinct qualities for undesirable qualities. It was not stated whether names were left on the resumes. Regardless, the models could pick up on minute differences, like resumes that used the word 'women' such as 'women's soccer' or graduates of women's only universities. While not discussed, I would go further to say that this may have also happened to resumes of graduates from historically black colleges and universities (HBCUs), as there is also a lack of representation for African-Americans in STEM. When Amazon's team tried to remedy this problem, there wasn't much change. The models began to find gender indicators that were less obvious, such as more masculine terminology like 'executed' and 'captured'. Amazon asserted that these models were never officially used for hiring anyone. They have since changed direction and are looking into how they can use this project to promote diversity.

2.2 Bias in Word Embeddings

If sufficient data is available, one may choose to build a domain specific word embedding model. There are also freely available versions that have been trained on corpora containing billions of words. Many don't have the resources to create an equally well-developed vector and will utilize these pre-existing embeddings. Since these common embeddings are used so frequently, it becomes easy to accept them as safe and dependable. The foundation of my research can be established by two papers that formally introduced, attempted to correct and assessed the corrections of bias within these popular word embedding vectors.

It can be difficult for anyone to admit that they're biased, even computer algorithms. Harvard's Implicit Association Test (IAT)[1] is a free online research tool that helps individuals assess what biases they may have by taking quizzes that compare two groups at a time. As an example, the skin-tone IAT "requires the ability to recognize light and dark-skinned faces. It often reveals an automatic preference for light-skin relative to dark-skin" [9].

This test was converted into the Word Embedding Association Test (WEAT) by challenging a particular word embedding algorithm (GloVe) to finish analogies. Starting simply, the WEAT checked for non-harmful prejudices such as showing flowers are considered significantly more pleasant than insects. Moving into more detrimental beliefs, they documented gender bias where "female names [were] more associated with family than career words" and "female words [were] more associated with arts than with mathematics" [2].

It can be argued that these kinds of associations are helpful in decision-making. With our nurse example from before, if we predict a nurse is a women, more than 9 times out of 10 we will be correct. However, it has been shown that machine learning models can quickly shift towards amplifying these biases. One study found that their model assumed women were 68% more likely to be related to cooking than men, despite the training data only having a 33% difference. [14]

Therefore, word embeddings and the models developed from them must be given more attention. One of the first groups to address this issue suggested a method for identifying and neutralizing a particular angle of bias, gender and occupation. Within an embedding built from the word2vec algorithm with documents from Google News, they found where the sets of male and female definitional terms were located. In this context, definitional terminology refers words where presence of gender is expected. For example, sister and waitress are inherently female terms. Nurse, on the other hand, would be considered stereotypically gendered. They were able to observe the extent of bias for various occupational terms by projecting them onto an axes with 'he' and 'she' on either end. While two methods were proposed for removing bias, hard and soft de-biasing, they only published an adjusted embedding with the former. Hard de-biasing involves two parts, neutralizing and equalizing . For every biased term, they moved it

[1] https://implicit.harvard.edu/implicit/takeatest.html

to the center of the gender axis within the word embedding model. This means there is no difference in similarity value between the term and 'he' or the term and 'she' following adjustment [1].

2.3 Differences in Word Embedding Algorithms

Not all word embeddings algorithms are the same, meaning resulting vectors trained on the same data will show disparities. Above, I noted two algorithms that were shown to contain bias, Word2Vec and GloVe. FastText is another popular algorithm that I have not seen analyzed, but will discuss. It's important for us to know the backgrounds of these algorithms to better understand future results.

Word2vec was developed in 2013 by a Google research team [8]. The model is actually comprised of two different algorithms, continuous bag of words (CBOW) and skip-gram. These algorithms rely on the concept that words that occur in the same contexts are semantically similar. Both models follow the same shallow neural network (NN) structure, but have opposite goals. Regardless of the algorithm, the embedding itself is the hidden layer of the trained NN. For skip-gram, the goal is to identify the surrounding words in a sentence given a single word. This teaches the model the contexts in which a word may occur. For CBOW, a context is given and the model must predict what word is most likely to appear. It is said that skip-gram works better with a smaller amount of data and is better for uncommon words, while CBOW trains faster and is better for more frequent words. Generally, finding a clear difference between the two algorithms is difficult given how similar they are.

GloVe came one year later from Stanford researchers and is built very similarly to word2vec [11]. Unlike word2vec, GloVe does not use a neural network to create its vector. Instead, it uses a log-bilinear model that essentially counts co-occurrences and computes the conditional probabilities for words appearing in similar semantic contexts.

FastText was first released in 2015 from Facebook's AI Research (FAIR) lab [6]. It is an improved model based upon the original word2vec framework. While word2vec and GloVE treat words as the smallest unit, FastText breaks words into characters n-grams. For example, *lovely* broken into bigrams would be *lo, ov, ve, el, ly*. Because every language has extractable

patterns, storing what characters fall before and after one another allows for better representation of words that occur very rarely or not at all within the vocabulary.

3. Methodology

Since my project goal was to observe the presence of bias within machine learning systems when using word embeddings, I emulated the Amazon project. There were no details given on what specific methods their team followed, so I do not know if they used word embeddings. However, I decided this model design was a good foundation because it had already been shown to produce biases.

3.1 Comparing Biases

I examined the features of the four pre-trained word vectors listed in Table 1. Each word vector was created with a word embedding algorithm trained on at least one data source. We can measure these sources by the number of word tokens within them and we can measure the created vectors through the number of words catalogued. Every word vector uses a dimension size of 300, such that every word in the embedding has a vector of length 300.

Algorithm	**Data Source**	**Number of Word Tokens**	**Number of Word Vectors**
word2vec	Google News	3 billion	3 million
Debiased word2vec	Google News	3 billion	3 million
GloVe	Wikipedia 2014 and Gigaword 5	6 billion	1 million
fastText	Wikipedia 2017, UMBC webbase corpus and statmt.org news dataset	16 billion	400 thousand

Table 1. Components for pre-trained word vectors

Given the variety of algorithmic approaches and amount of data, I wanted to see what kind of improvements could be found with the more advanced model, fastText, and the debiased word2vec model. With word embedding vectors, one can query a term and receive the K most similar words to it, such as Table 2.

Algorithm	**Results**
word2vec	Maryland, Charlottesville, Hampton Roads, North Carolina
GloVe	va., maryland, carolina, tennessee
fastText	Virgina, Viriginia, Charlottesville, Richmond

Table 2. 5 most similar words to 'Virginia'

We can also perform mathematical operations on word vectors. For example, if we wish to complete an analogy, we can add and subtract different word vectors. So the analogy, 'Man is to brother as woman is to _____', can be found with the following equation: Brother - man + woman. By replacing the dimensions for 'man' with the dimensions for 'woman', we are shifting the location of the original term, 'brother'. Every embedding I looked at, aside from GloVe, returned 'sister' as the result. This helps us discover semantic relationships held within the embeddings, especially for analogies that are less obvious.

For my work, I looked at the difference between occupations and the gendered terms. Table 3 contains a subset of results from this observation. For every word embedding, I selected a set of definitional and stereotypical terms and found the similarity towards 'she' and 'he'. Each term was labeled with a primary gender based on which pronoun it was most similar to. I then subtracted the higher similarity from the lower to show the amount of bias towards the primary gender, which is listed in each word embedding column.

Apart from the debiased word2vec embedding, every embedding shows more bias towards female terms than male terms. The original word2vec model shows the most bias, aside from 'computer programmer' interesting enough. This lack of bias may be indicative of

later trouble with finding bias within machine learning models. The debiased word2vec embedding has its neutralized values, resulting in zero difference between the 'he / she' similarity. Strangely, this was also done for 'waiter' which may be because this is considered a gender-neutral term. fastText has much smaller similarity differences, which may be due to the size of its training. Unfortunately, there is still strong bias within even this model. 'Nurse' is considered more female than 'sister', which clearly shouldn't be the case. The same problem happens between 'architect' and 'brother'.

Primary Gender	Gender Category	Term	word2vec	Debiased word2vec	GloVe	fastText
Female	Stereotypical	Nurse	0.247	0.000	0.187	0.102
Female	Stereotypical	Librarian	0.234	0.000	0.101	0.032
Female	Definitional	Waitress	0.242	0.263	0.230	0.097
Female	Definitional	Sister	0.253	0.336	0.295	0.075
Male	Stereotypical	Architect	0.148	0.000	0.104	0.084
Male	Stereotypical	Programmer	0.001	0.000	0.052	0.053
Male	Definitional	Waiter	0.017	0.000	-0.024	0.041
Male	Definitional	Brother	0.142	0.337	0.153	0.063

Table 3. Comparison of biased terminology within word embeddings

3.2 Data Collection

3.2.1 Resumes

My biggest challenge was finding a data source where I could access resume data that contained the name of the individuals. Unsurprisingly, there is no widely distributed dataset of

real resumes for a few reasons. Most resumes are incredibly personal. They contain a lot of identifying details such as name, employment and education history, and contact information. Resumes from other countries may contain even more private information such as marital status. Websites like Indeed do host resumes, but they avoid this issue by having users fill out a form that populates a standard template. This gives employers a look into skills and career accomplishments without sharing data that identifies anyone.

I ended up using a website called PostJobFree[2], which has been in operation since 2007. This site allows direct uploading of resumes with the expectation that they will be able to be found through search sites like Google. Since users have agreed to having their resume accessible, I decided this was the most ethical option I could find for a larger scale amount of data. Initially, I collected a set of resumes based on the gendered occupations specified in Bolukbasi 2016. Each term from the word embedding was given a calculated value for stereotypical gender and definitional gender. There were 34 occupations with a bias towards women and 89 with a bias towards men. The scale went from -1 for female terms to 1 for male terms. I focused only on terms with a high stereotypical value and a low definitional value, but I did not collect resumes for every occupation that fit this criteria.

Resumes for this data set were collected without scraping. For each profession, I queried PostJobFree.com and checked each result for a name, sufficient length, and intelligible content. I also attempted to collect an equal amount of resumes for men and women regardless of the stereotypical gender. This didn't always work out, which can be seen with electricians and stylists. If I extended this data set, I would stop trying to balance it, since it limited how much I was able to gather. Table 4 outlines what specific professions I queried, which category they belonged to, and the gender distribution of each. Table 5 gives a summary of the entire dataset.

[2] https://www.postjobfree.com/

Profession	Defined Gender	Biased Value	Male Resumes	Female Resumes	Total Resumes
Custodian, Janitor	Male	0.9	11	12	23
Superintendent	Male	0.9	13	2	15
Carpenter	Male	0.8	9	0	9
Electrician	Male	0.8	12	0	12
Sheriff, Deputy	Male	0.8	9	4	13
Athletic Director	Male	0.7	6	5	11
Dentist	Male	0.7	10	10	20
Pastor, Preacher	Male	0.7	14	3	17
Trucker, Truck Driver	Male	0.7	11	1	12
Computer Programmer	Male	0.6	9	8	17
Manager	Male	0.6	15	10	25
Chemist	Neutral	0.2	9	4	13
Biologist	Neutral	0.1	12	10	22
Consultant	Neutral	0.1	23	7	30
Author, Writer, Novelist	Neutral	0	8	11	19
Psychologist	Neutral	0	8	16	24
Counselor	Neutral	-0.1	9	15	24
Photographer	Neutral	-0.1	19	10	29
Realtor	Neutral	-0.2	3	9	12
Paralegal	Female	-0.4	17	19	36
Therapist	Female	-0.4	13	18	31

Secretary, Receptionist	Female	-0.7	5	23	28
Stylist	Female	-0.7	0	13	13
Teacher, Educator	Female	-0.7	5	9	14
Caretaker, Nanny	Female	-0.8	5	7	12
Housekeeper	Female	-0.8	6	11	17
Librarian	Female	-0.9	9	12	21
Nurse	Female	-0.9	11	19	30

Table 4. Distribution of scraped resumes

Category	**Male Resumes**	**Female Resumes**	**Total**
Male	119	55	174
Neutral	91	82	173
Female	71	131	202
Overall	281	268	549

Table 5. Summary of scraped resumes

Later in my research, I created a data set of purely computer programmer and software developer resumes. These were also retrieved from PostJobFree.com, but were scraped instead of self-selected. 930 resumes were collected and 856 were ultimately used. The most important step was identifying gender, so any resume that didn't clearly identify the individual were removed. I used 2 separate tools for doing gender identification. The first was a Python API called gender-guesser [3]that relied on a dictionary of 40,000 primarily European names. This API would return the assumed gender of a given name based on how frequently it was attributed to one gender or the other. Names with close to equally occurring frequency were labeled

[3] https://pypi.org/project/gender-guesser/

androgynous and names not in the dictionary were labeled unknown. This was most common with Asian names, so I relied on an external tool to check any that could not be labeled by the gender-guesser API. The Baby Name Guesser[4] is a website that checks Google for the usage of a name and reports back how popular the name is and how common it is for the primary gender. Using these 2 sources, I was able to assess the gender of each resume I collected aside from a few that were too androgynous to say for sure. In the end, I had 623 male resumes and 233 female resumes. Since these were collected indiscriminately, this reinforces the perception that computer programmer is a more masculine career.

Following the identification of the data set, I worked towards removing any personal details I could. I attempted to use named-entity recognition to aid with the removal process, but it often misattributed entities like companies or schools with names of individuals. I ended up going a more direct route and removed all applicant names, numbers, and strings ending in '.com' such as websites or emails. If a resume mentioned the word 'references', I had it flag me so I could figure out which section of the document should be removed, so others' personal data wasn't included either. I also did some cleaning up of the document data itself by removing punctuation and running words through a spellchecker to correct any small mistakes. Because resumes for different fields could potentially use very specific language, I wanted to make sure there weren't too many unidentifiable terms within them.

3.2.2 Job Postings

Unlike resumes, job postings are very easily accessible online. To match the two resume data sets that I created, I gathered resumes from two different sources. For the dataset of varied occupations, I scraped Indeed [5]for job listings matching the list of gender-biased careers. Approximately, 100 postings were collected for 19 different occupations, totalling to a corpus of 1,900 job posts. This data set was not used unfortunately due to a lack of time.

The second corpus was collected to match the computer programmer resumes. Since the goal of that data set was to emulate the Amazon hiring model, 88 job postings for software

[4] https://www.gpeters.com/names/baby-names.php
[5] https://www.indeed.com/

developers were pulled from Amazon's website[6]. Only punctuation and numbers were removed from both data sets due to their already short length and impersonal nature.

Profession	Defined Gender	Total
Athletic Director	Male	97
Carpenter	Male	50
Computer Programmer	Male	140
Custodian, Janitor	Male	100
Dentist	Male	59
Electrician	Male	99
Manager	Male	109
Pastor, Preacher	Male	136
Realtor	Neutral	118
Chemist	Neutral	110
Counselor	Neutral	80
Consultant	Neutral	130
Photographer	Neutral	98
Caretaker, Nanny	Female	120
Housekeeper	Female	80
Librarian	Female	89
Nurse	Female	110
Paralegal	Female	110

Table 6. Distribution of scraped job postings

[6] https://www.amazon.jobs/en/job_categories/software-development

3.3 Text Analysis

One of the biggest issues that arose from the Amazon hiring model was the emphasis put on masculine language. I wanted to see if there was any obvious difference in word choice between my set of female and male computer programmer resumes, so I compared the log-likelihood of words within each set of documents, or corpus. Log-likelihood measures word frequency between two corpora to find which words are most unique to each [12]. Words that show up frequently in corpus A and infrequently in corpus B will have a high log-likelihood for A and a low log-likelihood for B. Using this method, I found the top ten most distinct terms for women's and men's resumes from my dataset which can be found in Tables 7 and 8. Unfortunately, I do not believe I found anything significant.

Word	**Log-Likelihood**
Testing	412.23
Test	390.04
Cases	143.40
Involved	111.05
Regression	98.71
Defect	86.22
Selenium	72.21
Jira	70.67
Description	70.50
Automation	65.75

Table 7. Top 10 words for women's resumes

Word	**Log-Likelihood**
Systems	125.20
System	58.03
Programs	53.92
Senior	42.84
IBM	42.20
Network	41.16
Support	35.56
Windows	23.28
Inc	21.93
Including	21.51

Table 8. Top 10 words for men's resumes

I also explored a project where 1,100 technology resumes split near evenly between men and women were assessed [13]. The findings stated that women have longer resumes with around 745 words on average compared to men who have about 414 words per resume. I did not find this to be the case for either of my corpora. For my corpus of varied resumes, women averaged 572 words while men averaged 598 words. For the corpus of programmers, the average length was 676 for women and 766 for men. It appears that the article's data set was collected from primarily US-based individuals, whereas PostJobFree has shown to contain a wider variety of sources. There are assumedly cultural differences in what is considered a well-structured resume, but I was unable to find a trustworthy source confirming this. For example, online sources have claimed that it is more acceptable to put personal information in European and Asian resumes which I have seen during cleaning. The author may also have collected from a different tier, such as upper-management, or type of computer programmer, since they relied on a personal network to gather data.

3.4 Machine Learning

Since my goal was replicating a hiring model, I split my resume data set bya binary, hired or not. I assigned this label by finding which resumes were most similar in language usage to the set of Amazon job postings. For each resume, I found its average similarity to all job postings. I gave the 300 most similar the 'hired' label, leaving the 556 remaining 'not-hired'. Because of the Amazon masculine language issue, I wondered if I would see an imbalance between male and female resumes within these two groups. I was surprised to see that there was essentially no difference between the labels. The hired group was 72% male while the non-hired group was 73% male. This matched the actual distribution of the group, so it didn't give me much hope in terms of finding a bias within my model. While machine learning is able to pick up on very subtle, latent features, a data set this balanced was unlikely to show any large upsets.

I built a convolutional neural network (CNN) to predict whether the candidate of a given resume would be hired or not. CNN's are an advanced form of a deep neural network that use filters, or convolutions, to pull information from input data. While commonly used for

image-based machine learning tasks, CNN's are useful for any job that trains on data that can be broken into distinct chunks. For images, this is pixels. For text, this is words.

I trained three models, varying between vectors: the original word2vec embedding, the debiased Bolukbasi embedding, and the fastText embedding. This meant the same model structure was receiving the same inputs each time, but they were being translated differently. Despite using varied embeddings, I did not see any distinction between these models. Each had an accuracy of approximately 77%.

4. Conclusion

4.1 Results and Discussion

Often, discoveries occur when one is not intending it. This seems to be the case with bias in machine learning as well. While I did not expect to see profound bias present, I did believe I would see some form of difference. This may be for a number of reasons. With respect to machine learning, even a thousand resumes is a small amount. Instead of a deep neural network, I could have just as easily used a simpler algorithm and obtained similar results. If I continued with this project, I think I would have to either collect much more resume data or forego the angle of occupation entirely. Additionally, my corpora of resumes can be presumed to be distinct from those a company might work with. I specifically have data from individuals who were comfortable putting their resumes online. While this is not very different from having a Linkedin account, which some individuals in my corpus did list, the approach taken may indicate more strenuous job hunting.

4.2 Future Work

In the same way that Amazon adapted their model to identify bias and support diversity, others have been able to make use of bias within word embeddings like finding prejudiced language within customer reviews [7]. Implicit bias will always permeate our lives and subsequently, our data and algorithms. Working towards reducing bias in word embeddings is

an important component, but we must be wary of its presence at every phase of the ML pipeline. Thankfully, the rise of ML has also shown the rise of mindfulness and discussions around this issue. From techniques for collecting fair, yet representative data to developing methods for identifying and mitigating biased predictions, work is continuing to be done to alleviate problems with bias.

Now that I have these corpora, I plan to explore in more detail the language and structure used. It was shown that women often have longer resumes, because of executive summaries, non-standard sections, more personal distinctions, and a lack of bulleted lists [13]. I would like to test these claims with my own data. I'm also still surprised by my lack of findings, so I'd like to try more variety in my machine learning approach. Since my labeling process was based on a simple similarity measures, I could potentially create a more robust method for deciding who had been hired.

References

[1] T. Bolukbasi, K. Chang, J. Y. Zou, V. Saligrama, and A. T. Kalai. 2016. Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings. In *Advances in Neural Information Processing Systems,* pages 4349–4357.

[2] A. Caliskan, J. J. Bryson, and A. Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. In *Science*, 356, pages 183-186.

[3] J. Dastin. 2018. Amazon scraps secret AI recruiting tool that showed bias against women.
In *Reuters*.

[4] J. Gordon and B. Van Durme. 2013. Reporting bias and knowledge extraction. In *Automated Knowledge Base Construction* (*AKBC*).

[5] V. Ha-Thuc and S. Sinha. 2016. Learning to Rank Personalized Search Results in Professional Networks. In *SIGIR*.

[6] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. 2016. Bag of Tricks for Efficient Text Classification. From arXiv:1607.01759 [cs.CL].

[7] A. Mishra, H. Mishra, and S. Rathee. 2019. Examining the Presence of Gender Bias in Customer Reviews Using Word Embedding. From arXiv:1902.00496 [cs.CL].

[8] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proc. Advances in Neural Information Processing Systems* 26 3111–3119.

[9] B. A. Nosek, M. Banaji, and A. G. Greenwald. 2002. Harvesting implicit group attitudes and beliefs from a demonstration web site. In *Group Dynamics: Theory, Research, and Practice*, 6(1):101.

[10] S. Patil, G. Palshikar, R. Srivastava, and I. Das. 2012. Learning to Rank Resumes. In *Forum for Information Retrieval Evaluation*.

[11] J. Pennington, R. Socher, and C. D. Manning. 2014. Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP).*

[12] P. Rayson and R. Garside. 2000. Comparing corpora using frequency profiling. In *Proceedings of the workshop on Comparing Corpora, held in conjunction with the 38th annual meeting of the Association for Computational Linguistics (ACL 2000)*. 1-8 October 2000, Hong Kong, pp. 1 - 6.

[13] K. Snyder. 2015. The resume gap: Are different gender styles contributing to tech's dismal diversity? In *Fortune*.

[14] J. Zhao, T. Wang, M. Yatskar, V. Ordonez, and KW. Chang. 2017. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2979–2989.