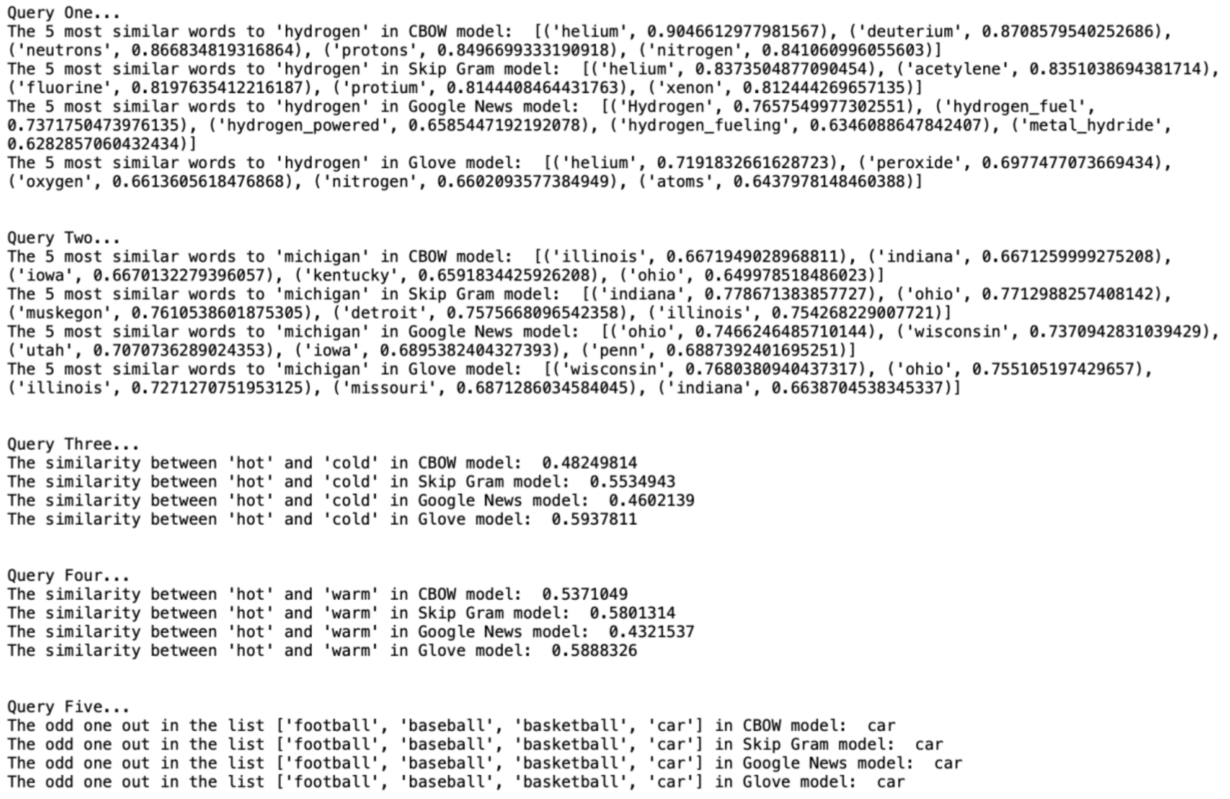
NLP Homework 2: Word Embeddings

Emma Ozias

1. Embeddings

I chose to train CBOW and Skip Gram embeddings. In order to train both embeddings, I used the gensim library. Gensim allowed me to generate two word vector embeddings. The parameters I used were min_count = 1, vector_size = 200, windows = 5, and workers = 8. However, for my skip gram model, I also used sg = 1 to indicate that it is a skip gram model. I learned that choosing too large of a vector size will cause my code to run much slower. Additionally, I realized the importance of the "workers" parameter. At first, I did not have this parameter and my code was running extremely slow. As a result, I added workers = 8 because my computer has a 10-core CPU. This sped up the process of training each word embedding immensely.

I ran five queries on the two models that I trained as well as two pretrained models. The results are below.



```
Query One...
The 5 most similar words to 'hydrogen' in CBOW model:  [('helium', 0.9046612977981567), ('deuterium', 0.8708579540252686),
('neutrons', 0.866834819316864), ('protons', 0.8496699333190918), ('nitrogen', 0.841060996055603)]
The 5 most similar words to 'hydrogen' in Skip Gram model:  [('helium', 0.8373504877090454), ('acetylene', 0.8351038694381714),
('fluorine', 0.8197635412216187), ('protium', 0.8144408464431763), ('xenon', 0.812444269657135)]
The 5 most similar words to 'hydrogen' in Google News model:  [('Hydrogen', 0.7657549977302551), ('hydrogen_fuel',
0.7371750473976135), ('hydrogen_powered', 0.6585447192192078), ('hydrogen_fueling', 0.6346088647842407), ('metal_hydride',
0.6282857060432434)]
The 5 most similar words to 'hydrogen' in Glove model:  [('helium', 0.7191832661628723), ('peroxide', 0.6977477073669434),
('oxygen', 0.6613605618476868), ('nitrogen', 0.6602093577384949), ('atoms', 0.6437978148460388)]


Query Two...
The 5 most similar words to 'michigan' in CBOW model:  [('illinois', 0.6671949028968811), ('indiana', 0.6671259999275208),
('iowa', 0.6670132279396057), ('kentucky', 0.6591834425926208), ('ohio', 0.649978518486023)]
The 5 most similar words to 'michigan' in Skip Gram model:  [('indiana', 0.778671383857727), ('ohio', 0.7712988257408142),
('muskegon', 0.7610538601875305), ('detroit', 0.7575668096542358), ('illinois', 0.754268229007721)]
The 5 most similar words to 'michigan' in Google News model:  [('ohio', 0.7466246485710144), ('wisconsin', 0.7370942831039429),
('utah', 0.7070736289024353), ('iowa', 0.6895382404327393), ('penn', 0.6887392401695251)]
The 5 most similar words to 'michigan' in Glove model:  [('wisconsin', 0.7680380940437317), ('ohio', 0.755105197429657),
('illinois', 0.7271270751953125), ('missouri', 0.6871286034584045), ('indiana', 0.6638704538345337)]


Query Three...
The similarity between 'hot' and 'cold' in CBOW model:  0.48249814
The similarity between 'hot' and 'cold' in Skip Gram model:  0.5534943
The similarity between 'hot' and 'cold' in Google News model:  0.4602139
The similarity between 'hot' and 'cold' in Glove model:  0.5937811


Query Four...
The similarity between 'hot' and 'warm' in CBOW model:  0.5371049
The similarity between 'hot' and 'warm' in Skip Gram model:  0.5801314
The similarity between 'hot' and 'warm' in Google News model:  0.4321537
The similarity between 'hot' and 'warm' in Glove model:  0.5888326


Query Five...
The odd one out in the list ['football', 'baseball', 'basketball', 'car'] in CBOW model:  car
The odd one out in the list ['football', 'baseball', 'basketball', 'car'] in Skip Gram model:  car
The odd one out in the list ['football', 'baseball', 'basketball', 'car'] in Google News model:  car
The odd one out in the list ['football', 'baseball', 'basketball', 'car'] in Glove model:  car
```

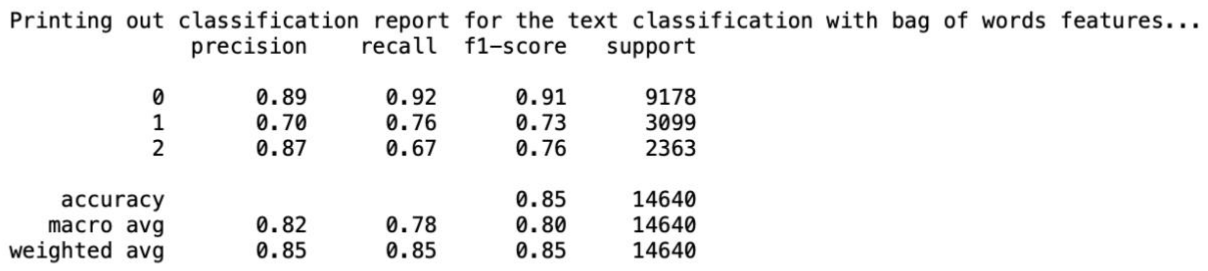
For query one, I thought that it was interesting that the Google News model chose words containing the original word (hydrogen) as most similar. I think that if the Google News model was preprocessed differently, then the output would be more like the output from the other models. Also, I thought it was interesting that for query two, only one of the models listed “Detroit” as one of the top five most similar words to “Michigan”. When I think of Michigan, I always think of Detroit, so I was expecting to see that in the top five most similar words generated by all four of the models. The purpose of query three was the find the similarity between two opposite words, and the purpose of query four was to find the similarity between two similar words. I was expecting query four to have much higher values for each model than the values in query three. However, the similarity values for the pairs of opposite and similar words were about the same. Finally, the results for query five were not surprising at all.

2. Bias

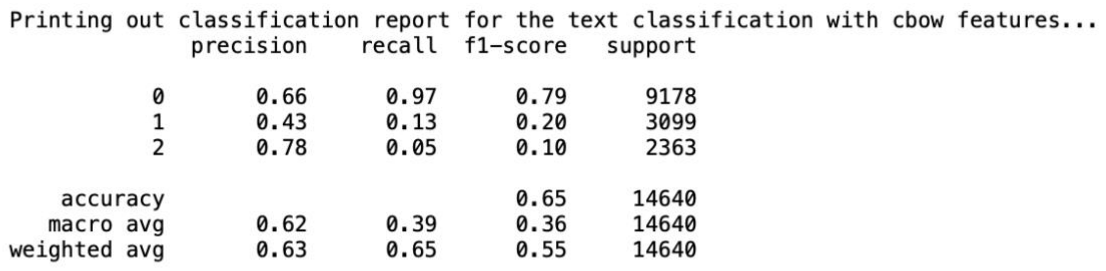
I extended the RNSB study by adding a new list of words. The original list of words contained many different nationalities. My new list contained careers, but each career was commonly associated with only one gender. The goal of my extension was to see if careers more commonly associated with women had a larger negative sentiment than careers commonly associated with men. For the list of words with nationalities, I noticed that most words had a negative sentiment distribution less than or equal to 0.1. The only models that had any words with negative sentiment distributions greater than 0.1 were my pretrained embeddings (Google News and GloVe). For my list of occupations, most of the bar graphs were similar except for the graph for the CBOW model. That graph shows that most words have a negative sentiment distribution of almost zero. The other models do not have many words with negative sentiment distributions close to zero. In general, the words nurse, man, woman, and mr had a larger negative sentiment distribution than other words. I think that each model holds a negative bias (some may be slight and others more significant) towards each word found in both word lists. If each of these word embeddings were used as features of a machine learning model, then that model could potentially have racist or sexist output. Each model has a larger negative sentiment bias towards certain nationalities and certain careers. Each model is not fair. It does not have the same negative sentiment distribution for all nationalities or all careers. (Note: all of the graphs are at the end of this document.)

3. Classification

For the text classification with cbow features, I chose to use the GloVe model. My classification task was to assign sentiment labels to tweets about airlines. I observed that my second classification report had much lower values for every single item in the classification report. My bag of words text classification used tf-idf. I think this could be the reason that the bag of words text classification performed much better. Tf-idf considers the importance of each word in a document which could help the model better predict the sentiment of a tweet. The GloVe model was not created by me, so I am not sure if it is able to identify certain words as more important to the corpus than other words.



```
Printing out classification report for the text classification with bag of words features...
              precision    recall  f1-score   support

           0       0.89      0.92      0.91      9178
           1       0.70      0.76      0.73      3099
           2       0.87      0.67      0.76      2363

    accuracy                           0.85     14640
   macro avg       0.82      0.78      0.80     14640
weighted avg       0.85      0.85      0.85     14640
```



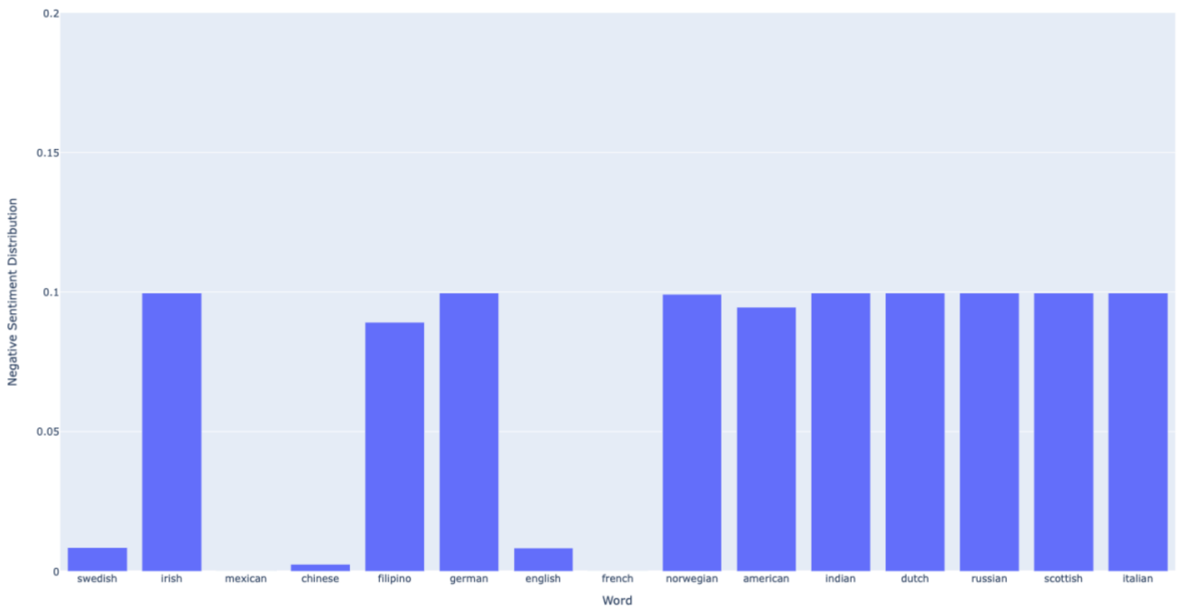
```
Printing out classification report for the text classification with cbow features...
              precision    recall  f1-score   support

           0       0.66      0.97      0.79      9178
           1       0.43      0.13      0.20      3099
           2       0.78      0.05      0.10      2363

    accuracy                           0.65     14640
   macro avg       0.62      0.39      0.36     14640
weighted avg       0.63      0.65      0.55     14640
```

4. Discussion

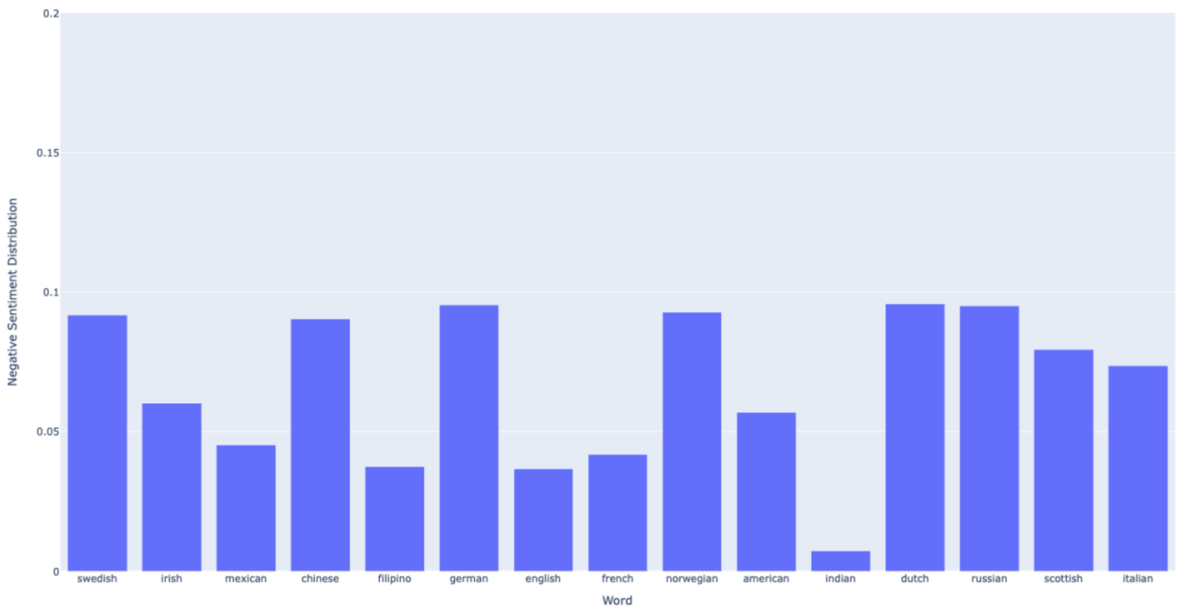
I learned a lot from this assignment. I learned how to evaluate bias in a word embedding using the RNSB study. Also, I learned about different queries that can be used on word embeddings. Finally, I built on my Python knowledge. I learned how to load word embeddings from the internet and how to generate bar graphs using pandas. I did not find anything unexpected except for the results generated by certain queries which were already discussed in this report.

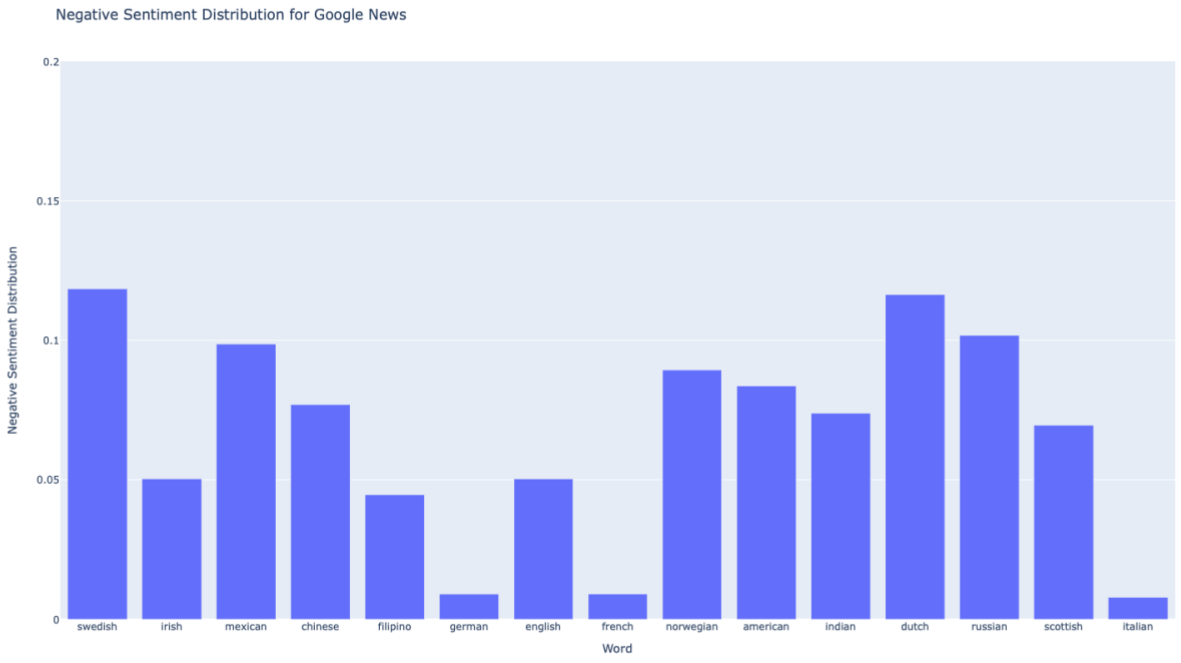
I struggled a lot with this assignment. At first, I had the wrong code for the bias portion of the assignment. I misunderstood section 2.4 of the assignment, so I originally coded equations from the research paper linked in the instruction document. For example, I evaluated man – woman = king – queen. Later, I realized that I did section 2.4 completely wrong. As I was redoing it, I ran into many issues. I could not get any of the graphs to print out. Every time that I ran my code, I would receive an empty bar graph. One of my classmates was able to help me get past this part. Due to this struggle, I learned how to use pandas for graph creation. Another struggle that I had was in section 2.5. I did not understand how to use a word embedding for linear regression with a dataset. I thought the list of labels was supposed to come from the word embedding which was incorrect as the word embedding does not have a list of labels. After talking with Professor Wilson, he helped me realize that I am still using a dataset with labels even though I am also using a word embedding.

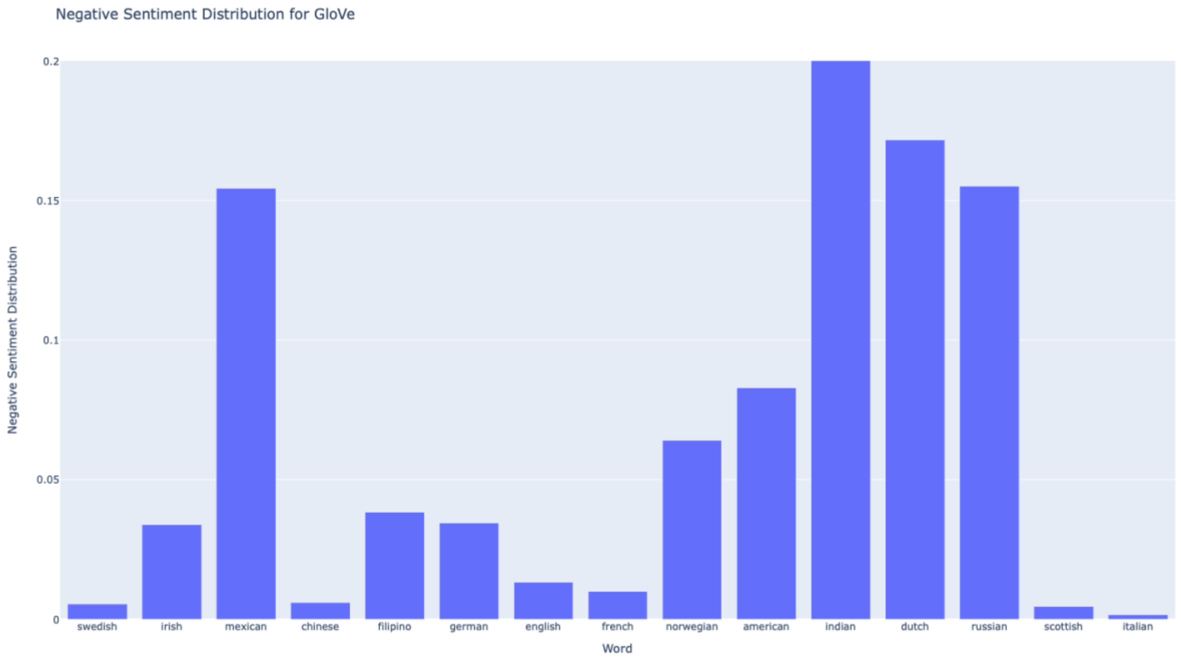
Negative Sentiment Distribution for CBOW

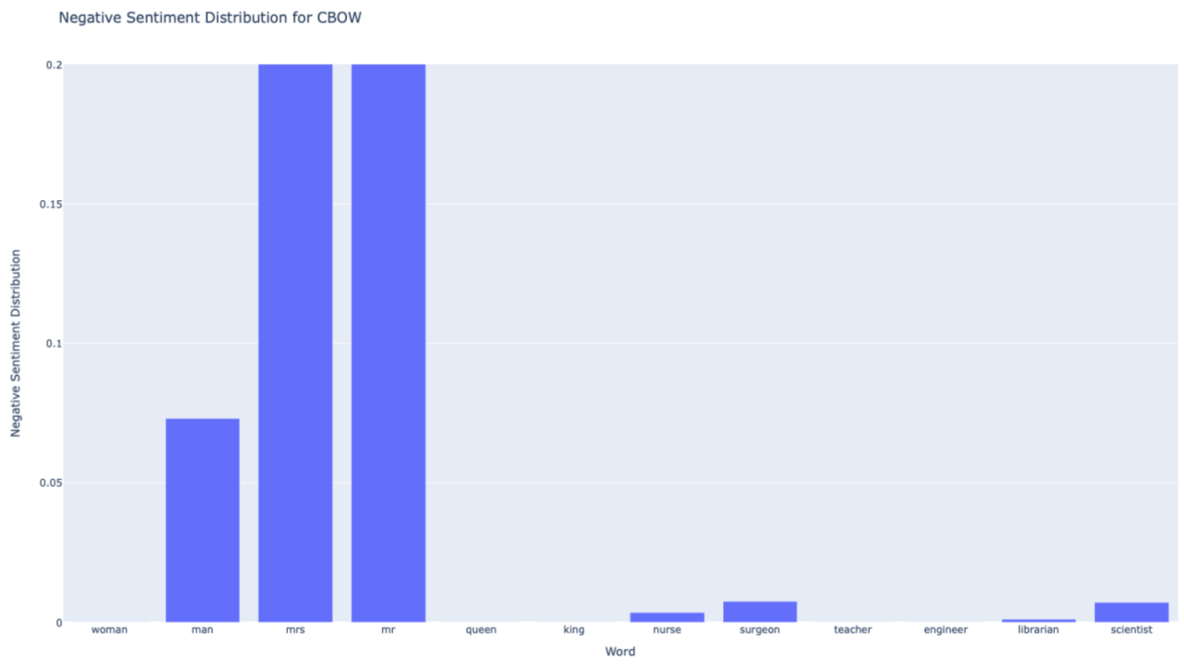


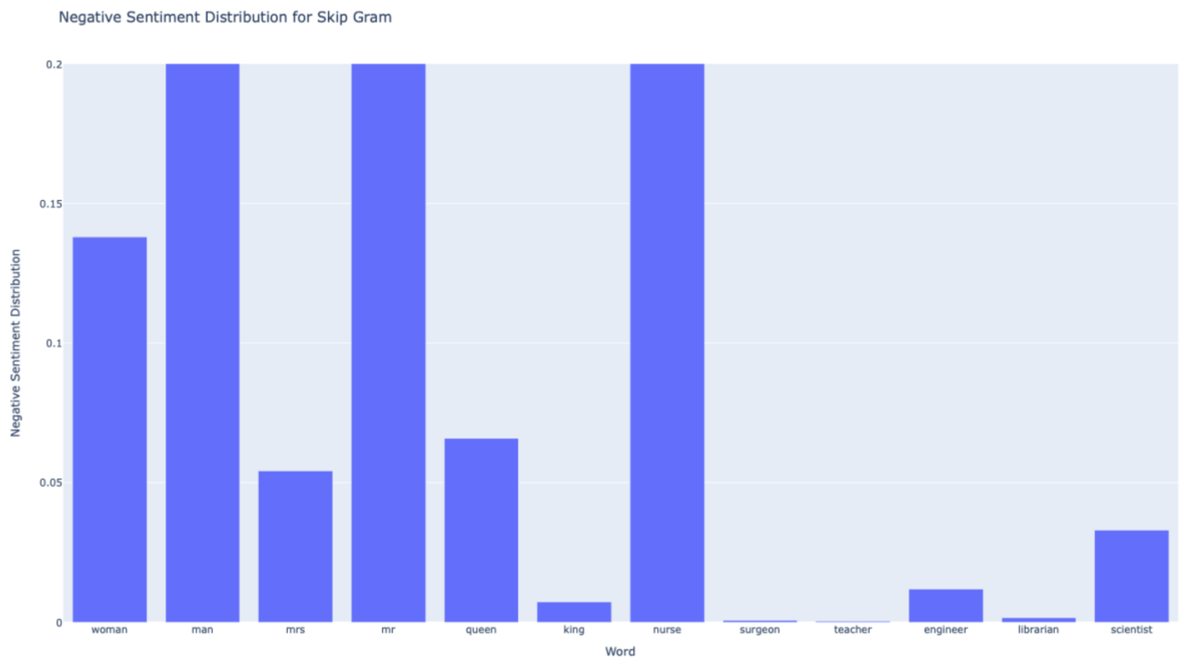
Negative Sentiment Distribution for Skip Gram



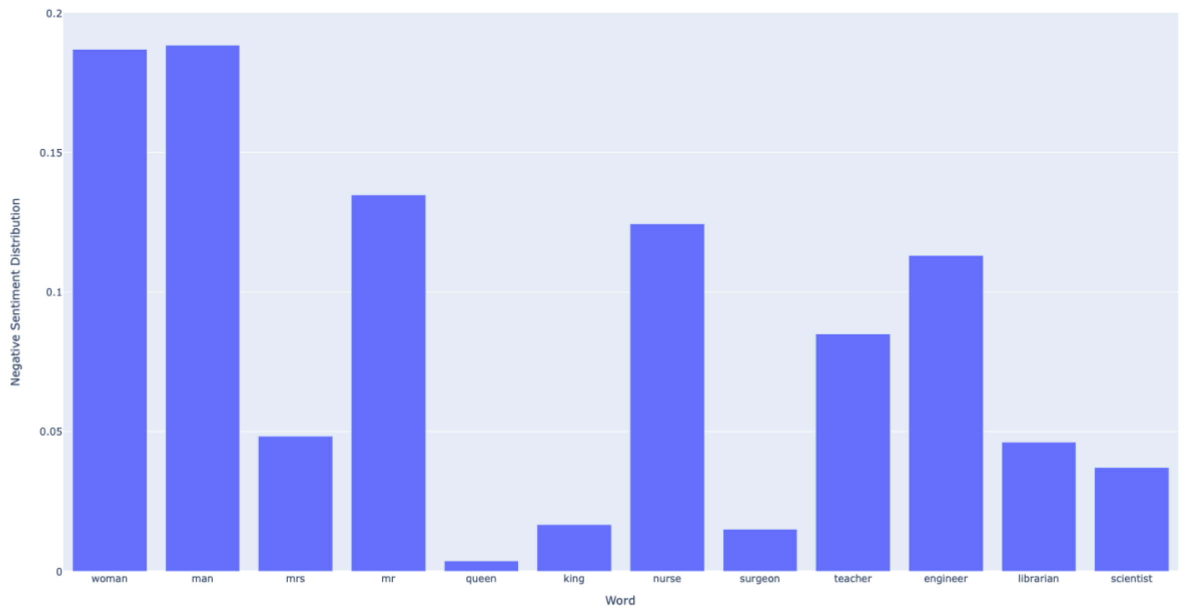

Negative Sentiment Distribution for Google News
Negative Sentiment Distribution
0.2
0.15
0.1
0.05
0
swedish
irish
mexican
chinese
filipino
german
english
french
norwegian
american
indian
dutch
russian
scottish
italian
Word


Negative Sentiment Distribution for GloVe
Negative Sentiment Distribution
0.2
0.15
0.1
0.05
0
swedish
irish
mexican
chinese
filipino
german
english
french
norwegian
american
indian
dutch
russian
scottish
italian
Word

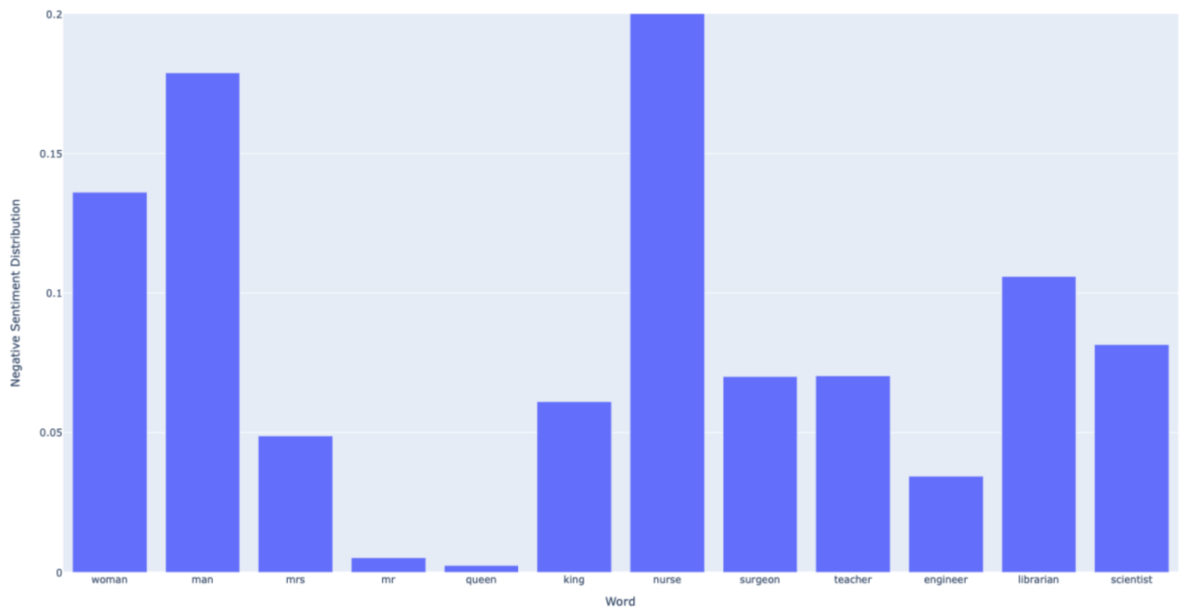

Negative Sentiment Distribution for CBOW
Negative Sentiment Distribution
0
0.05
0.1
0.15
0.2
woman
man
mrs
mr
queen
king
nurse
surgeon
teacher
engineer
librarian
scientist
Word


Negative Sentiment Distribution for Skip Gram
Negative Sentiment Distribution
0
0.05
0.1
0.15
0.2
woman
man
mrs
mr
queen
king
nurse
surgeon
teacher
engineer
librarian
scientist
Word

Negative Sentiment Distribution for Google News



Negative Sentiment Distribution for GloVe


Negative Sentiment Distribution
woman
man
mrs
mr
queen
king
nurse
surgeon
teacher
engineer
librarian
scientist
Word