Debiasing Irrelevant Words in Natural Language Processing

Shi Feng
fengs19@mails.tsinghua.edu.cn

Qihang Chen
chenqh19@mails.tsinghua.edu.cn

Abstract

In the field of artificial intelligence, the generalizability of machine learning models is a very important evaluation metric. When there is a leakage feature in the dataset and the dataset does not match the distribution in reality, the model may learn the wrong features and perform well on the original dataset but not in the real world. In this paper, we focus on the debiasing of sentiment analysis models, where the object of debiasing is whether neutral words with sentiment appear. The debiased model can be applied to more scenarios, and also has improved performance for different datasets in the same scenario. In this paper, we utilize the analysis of causal inference and improve it based on Zhang's work[1] to obtain a theoretical and systematic method of debiasing. We have used the obtained weights to debiased the model trained in IMDB review dataset[2], and the generalization of this model is significantly improved. At the same time, we also calculated the impact of those neutral words on the model prediction before and after the debiasing, and we can see that there is a certain decrease in the impact of those neutral words on the model.

Our work may enable small-sample training in the future, using a dataset in a particular scenario to train a generalized sentiment analysis model. Also, our work can be used as part of pre-training to improve the performance of machine learning algorithms in more complex problems.

1 Introduction

In general, there are two ways to implement AI, one is to use what humans know to translate our behavior or ideas into algorithms, and the other is to use machine learning models with learning capabilities, train the models with carefully selected data sets, and finally test whether the models can achieve the results we want. For the task of sentiment analysis, the latter is more often used due to the complexity of linguistics. But it is impossible to collect all the data in reality, so one often samples and selects a dataset to train the model on. However, in order to maintain a balance of labels in the dataset and to speed up labeling, annotators often use a strategy to bring some features into the dataset that are different from the real world. Even if the annotator does not intentionally adopt a biased annotation strategy, the sampling method may affect the unbiasedness of the final dataset to a greater or lesser extent. In our work, we focus on the effect of the feature "the presence or absence of a particular word in a sentence" on the model prediction. One such case is illustrated in Figure 1. As we can see, since the word "watch" appears in all the positive movie reviews and the word "see" appears in all the negative movie reviews, the trained model may be able to predict all the occurrences of the word watch" in all the sentences and "see" in all the sentences, the trained model may judge all the sentences with "watch" as positive and all the sentences with "see" as negative. What we want

to do is to show this bias and remove it from the model in theory and in practice. We chose the upstream task of sentiment analysis as the object of analysis and obtained results that can be used in more complex downstream tasks.

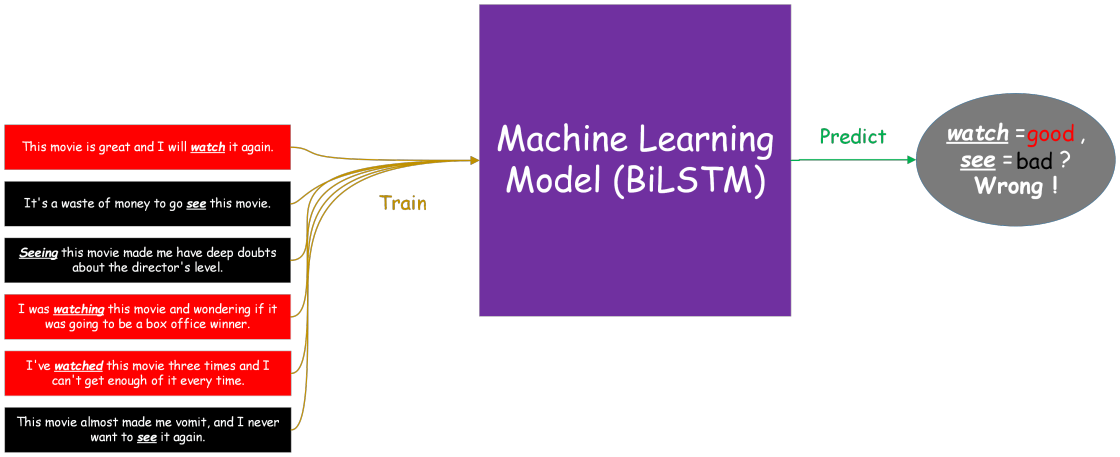

Fig. 1. An intuitive toy example of the problem we are solving.

To address this issue, we utilized a causal analysis approach to decouple the sampling process. We generalize a theorem from Zhang's work[1], which theoretically proves that the weights we compute are correct and can guarantee the unbiasedness of the model. In the next section, we explain some of the differences between the existing work.

2 Related Works

In the field of natural language processing, there has been some work on debiasing textual datasets; Zhang's work[1] identifies the leakage feature of sentence occurrence frequency in utterance relation datasets and proposes a framework theory for debiasing this bias. In this paper, we improve their theoretical approach by explaining the data generation process in terms of causal analysis. We use the improved approach to debiased the feature of whether irrelevant words appear inside the sentence, an idea that can be used not only to debiased the dataset but also to further improve the performance of small-sample training.

Meanwhile, Veitch's work used causal analysis to de-bias some macroscopic features in the article[3]. They debiased the problem by treating features such as the theme of the text and the level of writing as common reasons for the inclusion of a particular vocabulary and the recognition of the text. The experimental results show that the inclusion of a particular vocabulary does not significantly improve the approval rate of an article, which is consistent with reality. Unlike their work, we wanted to de-bias specific words, rather than macro features like the writing style.

3 Theoretical Analysis

In order to make the problem we study clearer and more rigorous, we redescribe our problem here in mathematical terms. Here are some of the notations we use.

1. $\mathcal{D}, D$: the original data distribution, the dataset we sampled.
2. $\mathcal{X}, \mathcal{Y}$: distribution of sentences and tags in the real world.
3. X, Y: variables representing sentences and tags in the real world.
4. $\overline{X}, \overline{Y}$: variables representing sentences and tags in the selected dataset.
5. S, I: variables representing sampling strategy and sampling intention (their corresponding distributions are $\mathcal{S}$ and $\mathcal{I}$).
6. $\mathcal{A}$: the joint distribution of variables X, Y, S, I before sampling.
7. $\overline{\mathcal{A}}$: the joint distribution of variables $\overline{X}, \overline{Y}, S, I$ after sampling.
8. f, f', L: the true functional relationship between sentences and tags, the functional relationship fitted by the model, the loss function used in model training.
9. X', X'': the part of the sentence that has an effect on the label (the effective independent variable of f) and the part that is useless and produces a leakage feature.
10. g: the function between the leakage feature in X (i.e., X'') and sampling intention.

Suppose we have a dataset D with commentary texts and labels indicating whether the emotion is positive or negative. For each remark $x_i, 1 \leq i \leq n$ consists of texts, we have a label $y_i \in \{0, 1\}, 1 \leq i \leq n$ corresponding to it. If the paragraph are positive comments, the label will be labeled to 1 by annotators; if the paragraph are negative comments, the label will be labeled to 0. Suppose $\overline{X} = \{x_1, x_2, \cdots, x_n\}$ and $\overline{Y} = \{y_1, y_2, \cdots, y_n\}$. Naturally, there is a functional relationship f between x_i and y_i, i.e. $f(x_i) = y_i$. However, we can not perfectly restore this function from a linguistic point of view, so we need to train a model on it to estimate the functional relationship. Suppose the estimated functional relation is represented by model f', which is determined by $\overline{X}$ and $\overline{Y}$.

Using some model training techniques, we can find an estimation f' works well on the training set D and data from the same source (a distribution $\mathcal{D}$) and acquired by the same sampling strategy distribution $\mathcal{S}$. Now, there are two problems that we need to solve. The first one is that the sampling strategy is biased. For example, in our dataset, some irrelevant words are correlated to the label, which is not reasonable. If we apply the trained model to the real world paragraphs, the accuracy of the model will be greatly affected, and even virtual words can be used to attack the model (for example, someone can deliberately add words to a bad review to make the system judge it as a good review.). Another problem is that our model is not necessarily only applied in one scenario. For example, a model for judging the emotion of film reviews may also be used in book reviews. Then we need to ensure that the prediction of the model will not be affected by nouns such as movies, houses, etc., which generally have no direct emotional orientation. We will show the solution to solve these two problems theoretically in this section.

3.1 Assumptions

Although the research problem is very different, we can still generalize and improve based on Zhang's hypothesis and proof method in general[1]. In fact, they only considered the leakage feature of sentence frequency, which would not appear in the data set we studied. However, this does not affect our analysis of the data generating process and debiasing methods.

We will propose some assumptions to model this problem. Suppose $\mathcal{D} = \mathcal{X} \times \mathcal{Y}$ is the distribution of reviews and labels in reality. $\mathcal{I}$ is the distribution of sampling intention and $\mathcal{S}$ is the distribution of sampling

strategy, which is related to the existence of some typical irrelevant words in this problem we are studying. Each time, the annotator takes out an element (x, y, i, s) from the joint distribution $\mathcal{A} = \mathcal{X} \times \mathcal{Y} \times \mathcal{I} \times \mathcal{S}$. If $y = i$, then the annotator adds this element into D. Otherwise, he/she does not add it to the dataset. Here, $i = 1$ (sampling intention is adding a positive data) means that the annotator wants a positive label, and $i = 0$ means that the annotator wants a negative one because he/she needs to make sure that the number of positive labels is equal to the number of negative labels. From this process, the dataset is created. We need to be aware of the fact that this is the practical dataset creating process in many situations, so this assumption is reasonable. After the sampling process, suppose the biased distribution is $\overline{\mathcal{A}}$, then we have

$$P_{\overline{\mathcal{A}}}(X, Y, I) = P_{\mathcal{A}}(\overline{X}, \overline{Y}, I) = P_{\mathcal{A}}(X, Y, I|S = Y) \tag{1}$$

Moreover, it is natural that the sampling intention is completely determined by the sampling strategy. Actually, we assume that the annotator selected data purposefully. Therefore, we have the following equation:

$$P(I|S) = P(I|S, X, Y) \tag{2}$$

Also, it is natural that the label is independent of our sampling strategy (in our problem, the sampling strategy is completely determined by some features in X), which can be mathematically represented by

$$P(Y|S) = P(Y) \tag{3}$$

With these assumptions, we can create a causal diagram to explain the process of data generation, the causes of the problem, and our solutions in the next subsection.

3.2 Causal Explanations

According to the assumptions in the last subsection, we have six variables in total. In the original distribution, we have variables X, Y that Y is determined by X through $Y = f(X)$. The sampling intention variable I is completely determined by sampling strategy S according to equation 2. Suppose in the biased distribution $\overline{\mathcal{A}}$, X, Y are transformed to $\overline{X}, \overline{Y}$, which are in the biased distribution $\overline{\mathcal{A}}$. Therefore, $\overline{X}$ is determined by I and X while $\overline{Y}$ is determined by Y. In our problem, the sampling strategy is related to some virtual or irrelevant words X'' (i.e. the reviewer is used to using 'this' when giving positive reviews), therefore, S is determined by X (actually, a part of X). According to these analysis, the final causal diagram[4] is shown in Figure 2. The dataset we see fits the distribution of variables in the red box. The bias of the dataset comes from the green path $X \to S \to I \to \overline{X}$, which changes the distribution of X, so that the relationship between $\overline{X}$ and $\overline{Y}$ is not exactly the same as the relationship between X and Y.

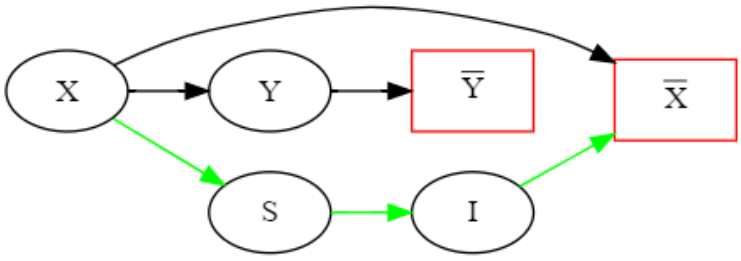

Fig. 2. The causal diagram of our problem.

In order to remove the bias, we need to remove the influence of the green path. Note that S is determined by some features in X (for example, whether 'this' appears in the comment), and I and X jointly determine

$\overline{X}$. The only difficult-to-estimate relationship is the functional relationship from S to I. Actually, we can compute $P(I=1|S), P(I=0|S)$ according to the following equations:

$$P(I=1|S) = \frac{P(Y=0)P_{\overline{\mathcal{A}}}(Y=1|S)}{P(Y=0)P_{\overline{\mathcal{A}}}(Y=1|S) + P(Y=1)P_{\overline{\mathcal{A}}}(Y=0|S)} \tag{4}$$

$$P(I=0|S) = \frac{P(Y=1)P_{\overline{\mathcal{A}}}(Y=0|S)}{P(Y=0)P_{\overline{\mathcal{A}}}(Y=1|S) + P(Y=1)P_{\overline{\mathcal{A}}}(Y=0|S)} \tag{5}$$

Here, $P_{\overline{\mathcal{A}}}(Y=1|S)$ can be estimated in our dataset. Actually, we can use the chosen features as the input and the label as the output to train a decision tree (we use random forest actually), then use the probability predicted by the decision tree as an unbiased estimation of $P_{\overline{\mathcal{A}}}(Y=1|S)$ in our problem. Figure 3 visualizes the process of obtaining our weights and the meaning of each variable. In the next section, we will prove that $\frac{1}{P(I=y|S)}$ is an unbiased weight for the dataset, where y is the label of the corresponding weighted data.

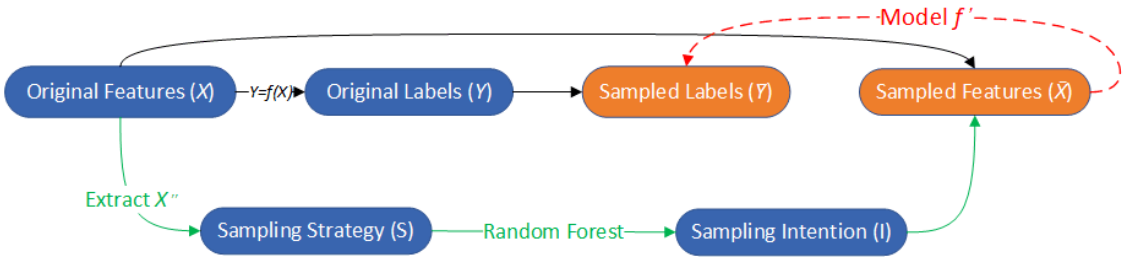

Fig. 3. The process by which we estimate the weights and the meaning of each variable.

Technically, we will select some irrelevant words as inputs and use labels as output to train a random forest model. We will calculate the output using the probability form of the random forest as an estimate of $P_{\overline{\mathcal{A}}}(Y=1|S)$ and $P_{\overline{\mathcal{A}}}(Y=0|S)$ to obtain $P(I=1|S)$ and $P(I=0|S)$. Then we will select $\frac{1}{P(I=1|S)}$ or $\frac{1}{P(I=0|S)}$ as the weight according to the label of each data in the data set. Finally, we will put the obtained weight as *sample_weight* in the training code of LSTM. The complete process is shown in Figure 4.

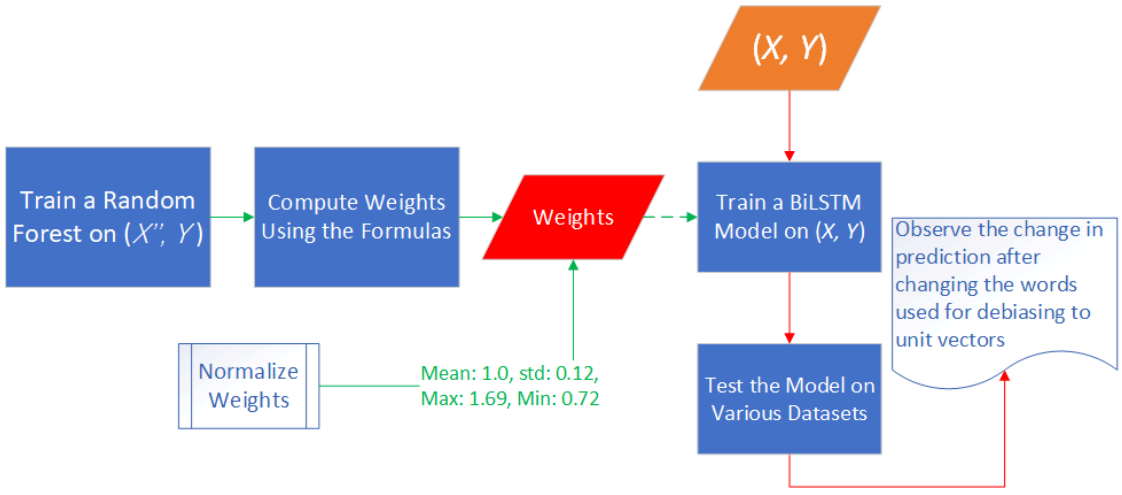

Fig. 4. The complete debiasing process.

3.3 Correctness of Our Approaching

In the previous subsection, we denote the classifier for sentiment analysis we use by f'. Here, we divide X into two parts, X' and X''. X'' is the biased features that determine S and X' is the other features that fit the real problem ($f(X)$ does not change when X'' is changed), according to linguistics. Therefore, $f'(x) = f'(x', x'')$ and $s = g(x'')$ with some function g as we defined previously. Moreover, suppose the loss function we use is denoted by $L(f'(x), y) = L(f'(x', g^{-1}(s)), y)$. Then we will prove that $w = \frac{P(I=Y)}{P(I=y|s)}$ can be an unbiased weight, which indicates that $\frac{1}{P(I=y|s)}$ is also unbiased because $P(I = Y)$ is a constant for all the data points. Actually, we have

$$\mathbb{E}_{x,y,s\sim\overline{\mathcal{A}}}[wL(f'(x', g^{-1}(s)), y)] = \int \frac{P(I=Y)}{P(I=y|s)} L(f'(x', g^{-1}(s)), y) dP_{\overline{\mathcal{A}}}(x, y, s) \tag{6}$$

$$= \int \frac{L(f'(x', g^{-1}(s)), y) P(I=Y)}{P(I=y|s)} dP(x, y, s|I=Y) \tag{7}$$

$$= \int \frac{L(f'(x', g^{-1}(s)), y) P(I=Y)}{P(I=y|s)} \frac{P(I=Y|x, y, s) dP(x, y, s)}{P(I=Y)} \tag{8}$$

$$= \int L(f'(x', g^{-1}(s)), y) dP(x, y, s) \tag{9}$$

$$= \mathbb{E}_{x,y,s\sim\mathcal{A}}[L(f'(x', g^{-1}(s)), y)] \tag{10}$$

which is exactly what we want to prove.

Formally, we reclaim the unbiasedness as a theorem as below:

Theorem: Suppose the classifier is f' and the training data X is divided into two parts, X' and X''. Here, X'' is the biased features (leakage features) and X' is the useful features that fit the practical problem. We denote the sampling intention and strategy as I, S, then $w = \frac{P(I=Y)}{P(I=y|s)}$ is an unbiased weight. Or equivalently,

$$\mathbb{E}_{x,y,s\sim\overline{\mathcal{A}}}[wL(f'(x', g^{-1}(s)), y)] = \mathbb{E}_{x,y,s\sim\mathcal{A}}[L(f'(x', g^{-1}(s)), y)] \tag{11}$$

Here, $\overline{\mathcal{A}}$ is the biased joint distribution of X, Y, S, I and A is the original distribution. Moreover, L is the loss function and g is the functional relation between sampling strategy S and biased features X''.

4 Experimental Results

The first experiment uses Keras' own tokenizer for pre-training, showing the effect of the leakage feature on the model generated by irrelevant words. We also used the pre-trained word vectors to obtain a comparison of the accuracy of the model on the test set before and after debiasing.

In the second experiment, we used google pre-trained word2vec word vectors[5] in order to show that our model can be migrated to other scenarios of the dataset and can show the effect of debiasing by replacing the word vectors. We found that the generalization of the model to the same scenario dataset improved, the impact of the words used for debiasing on the model prediction decreased, and the performance of the model improved across scenario datasets.

We present the two experiments in the following two subsections.

4.1 Experiment 1: The effect of Irrelevant Words on the Model

4.1.1 Experimental Setup and Overview

We worked on IMDB Review Dataset (IMDB)[2], and used the model in the notebook of Nilan[6] as our baseline model. We have done the following steps in experiments:

1. Train the model using the first 20000 data points of IMDB (train) and using the last 5000 data points in IMDB (train) as the validating set. Then compare the outputs of the model with labels in the training set and calculate the F1-score $F1_0$.
2. Choose some neutral words in IMDB (train) which seem to be inessential but will affect the result of the model.
3. Adjust the weight of some effective neutral words using the method in section 3.2 and use the weights as the *sample_weight* in Keras to train a new model. Calculate the F1-score of output in training set $F1_1$. We naturally find that $F1_1 < F1_0$, since the training set is in fact biased and a corresponding biased model will have a better prediction on these biased samples.
4. Apply IMDB (test) as the test set to the two models and compute corresponding F1-scores $F1'_1, F1'_0$. We found that $F1'_1 > F1'_0$. The reason is that the test set has a different distribution from the training set because any model performs much better on the validation set than on the test set. Therefore, the adjusted model should fit better to most of the sentence-label relations that do not conform to the biased distribution in the training set, and thus the adjusted model predicts better on the test set.
5. We build biased datasets based on IMDB (train) (e.g. for the word "movie", we make the positive sentences mostly contain "movie" and the negative sentences mostly do not contain "movie"). We then observe the effect of the word on the model in the model trained on the fully biased dataset and in the original model.

4.1.2 Details of Experimental Results

Our code[1] is available on GitHub, and the main part uses Keras. To achieve the classification of comments, we trained an LSTM model the same as that in Nilan's notebook[6]. with the first 20,000 data from the training set and used cross-entropy as a loss function. To debias the model, we trained a random forest of predictable labels with the presence or absence of partial virtual words as input. This random forest was 60 percent accurate on the validation set, so we concluded that these linguistically unrelated emotion words do have predictive power on the biased training set. According to equation 5 and equation 4, we have obtained a set of weights. The largest of the weights is 1.868435, the smallest is 0.686395, and most are between 0.9 and 1.1.

Using the weights as *sample_weight*, we have trained an adjusted model. To demonstrate that the adjusted model does indeed reduce the learning of biased features, we calculated the f1-score of the two models on IMDB (train) and the IMDB (test). The results are shown in the table 1, and as expected, in the distribution of the training set, our adjusted model has a lower F1-score because it does not learn the biased features (whether or not a word is present) but the original model learns, while in the test set, the adjusted model has a higher f1-score because the test set does not have the biased features (more precisely, the bias is different).

[1] https://github.com/fengtony686/Irrelevant_Word_Debiasing

Model\Data Set	Training Set	Test Set
Original Model	0.9305696846388607	0.8141332987775906
Adjusted Model	0.930267940157214	0.8202914654472413

Table. 1. F1-scores of original model and adjusted model on IMDB (train) and IMDB (test).

Also, we built some fully biased datasets for some words (making one dataset for each word) and trained the model on that dataset. We observed the change in model predictions after replacing words with spaces and found that the change was significantly larger for the model trained on the biased dataset. This indicates that the problem we pointed out does exist and that the model bias can be severe when there are more and more such biased words. Since it takes a lot of time to train each model, and a biased model needs to be trained for each word, we selected some of the words for our experiments. As shown in Figure 5, we expect the difference in prediction to be close to 0. However, the difference in prediction is relatively large when the model trained on the extremely biased dataset is replaced with words.

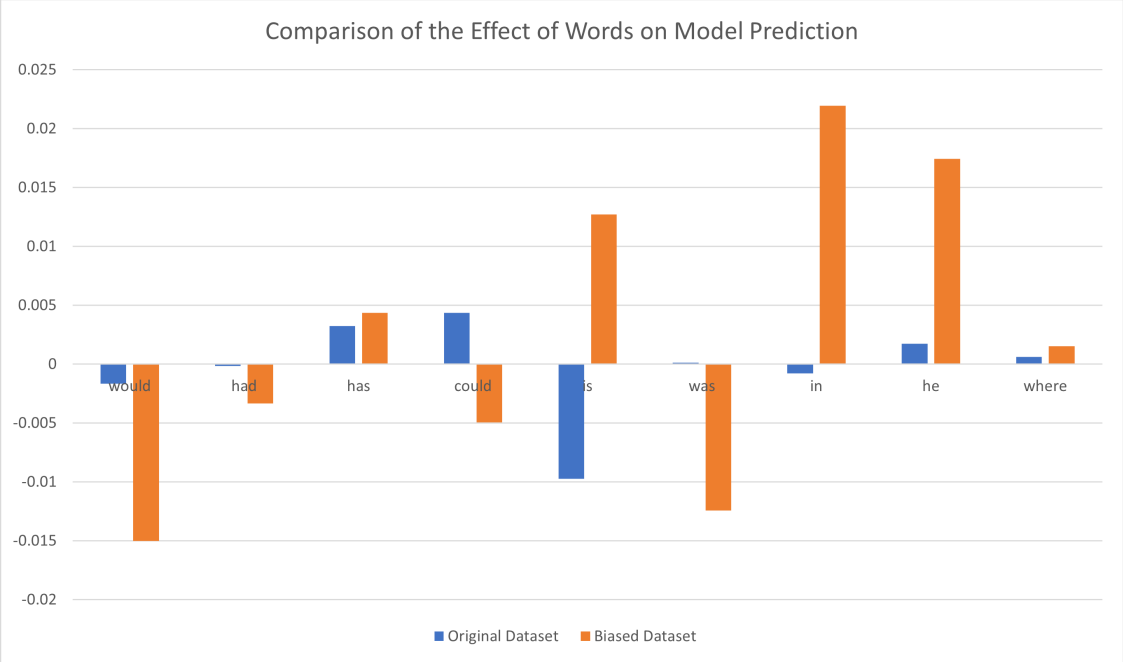

Fig. 5. Comparison of the effect of words on model prediction.

4.2 Experiment 2: Performance of the Debiased Model

The setup of this experiment is similar to the previous one, but the dropout part is added to the LSTM so that the model will not be overfitted too easily and the training difficulty is reduced. Also, to better show the effect of words on the model, we use word2vec (so that we can replace a word with a unit vector to observe the change in model prediction) to vectorize the dataset. The structure of our model is shown in Figure 6.

We trained the model on IMDB (train) and tested it on the IMDB (test), and Bag of Words Meets Bags of Popcorn (BWMBP)[7] datasets, as shown in Figure 7. It can be seen that the model has some performance improvement on the other movie evaluation datasets after debiasing. This indicates that our

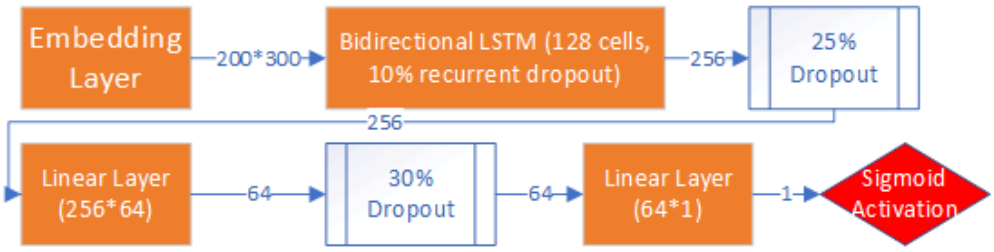

Fig. 6. Structure of the neural network for sentiment analysis.

debiasing method can improve the generalization performance of the model.

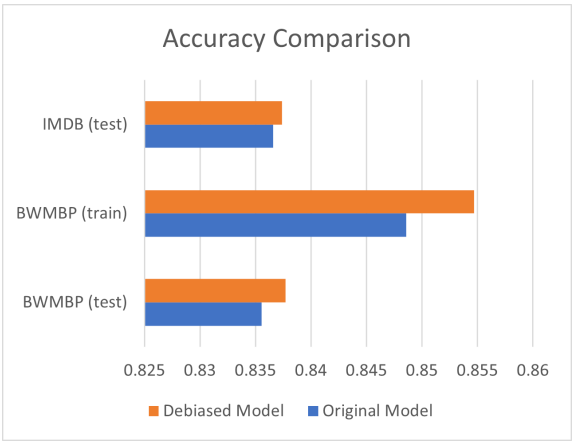

Fig. 7. Comparison of accuracy of the two models on the movie reviews datasets.

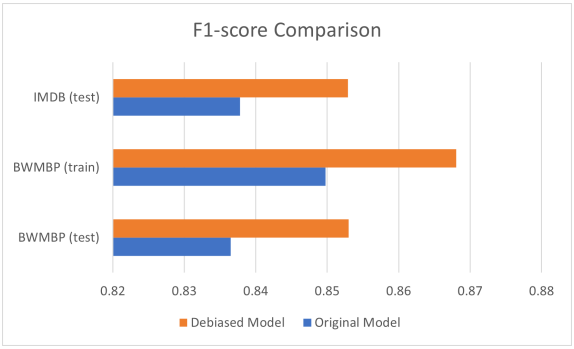

Fig. 8. Comparison of the F1-scores of the two models on the movie reviews datasets.

Also, we calculated the F1-scores of the two models on the three datasets used for testing, as shown in Figure 8. Similar conclusions are obtained, and the model performs a bit better after debiasing.

To show that our model does remove the bias, we compared the difference in predictions between the two models before and after replacing the vector of a word with a unit vector. Since replacing irrelevant words with unit vectors may have some effect on sentence structure, we only compare the difference between the two models here, assuming that the effect on sentence structure is the same for both models. That is, we would like to see that the effect of the operation of replacing a word with a unit vector should have a smaller impact on the prediction results of the model. As shown in Figure 9, the effect is indeed reduced for the few words we randomly selected, which indicates that we did remove the bias of the model regarding the occurrence of irrelevant words.

Finally, to validate our migration learning idea, we put the model trained on movie reviews to the test on the hotel reviews dataset[8]. This hotel review dataset has users rating from 1 to 10, and we set a threshold to determine whether the review is positive or negative. As shown in Figure 10, when the threshold is set between 8.75 and 9.75, the debiased models all perform better.

5 Conclusion

In this paper, we propose a method for de-biasing specific words. This method removes such bias as whether a certain word appears in the dataset or not, which leads to some improvement in the generalization

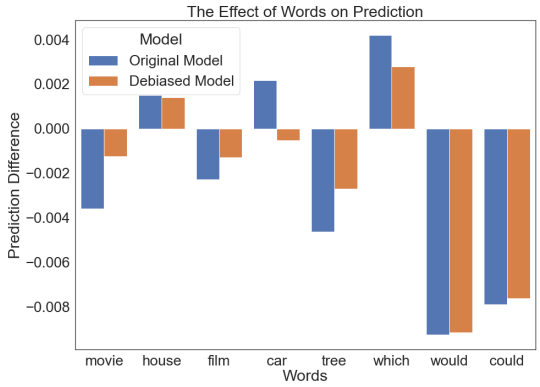

Fig. 9. Comparison of effects of words replacement on the two models.

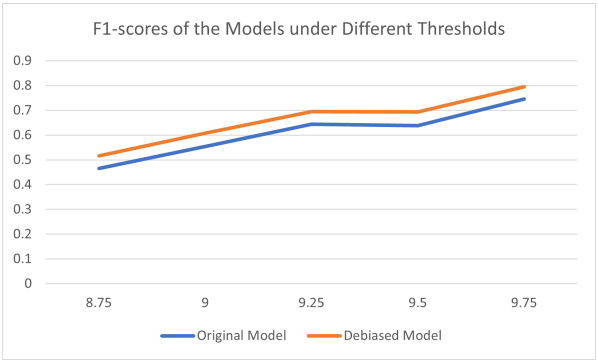

Fig. 10. Performances of models on the hotel reviews dataset.

of the model. More importantly, we can make the model trained in a particular scenario perform better in more scenarios by appropriately selecting certain words. In the future, we can achieve light migration learning by more accurate word selection to further enable small sample training. At the same time, better performance on sentiment analysis problems could also allow AI to achieve better performance in downstream tasks, such as chat-bots and other applications.

References

[1] ZHANG G, BAI B, LIANG J, et al. Selection Bias Explorations and Debias Methods for Natural Language Sentence Matching Datasets[J/OL]. CoRR, 2019, abs/1905.06221. arXiv: 1905.06221. http://arxiv.org/abs/1905.06221.

[2] Utagh. IMDB Review Dataset[EB/OL]. 2020. https://www.kaggle.com/utathya/imdb-review-dataset.

[3] VEITCH V, SRIDHAR D, BLEI D. Adapting Text Embeddings for Causal Inference[C]//Conference on Uncertainty in Artificial Intelligence. [S.l. : s.n.], 2020: 919-928.

[4] PEARL J. Causality[M]. [S.l.]: Cambridge university press, 2009.

[5] MIKOLOV T, CHEN K, CORRADO G, et al. Efficient estimation of word representations in vector space[J]. ArXiv preprint arXiv:1301.3781, 2013.

[6] Nilan. IMDB Review - Deep Model 93.51% Accuracy[EB/OL]. 2020. https://www.kaggle.com/nilanml/imdb-review-deep-model-94-89-accuracy.

[7] Kaggle. Bag of Words Meets Bags of Popcorn[EB/OL]. 2020. https://www.kaggle.com/c/word2vec-nlp-tutorial.

[8] LIU J. 515K Hotel Reviews Data in Europe[EB/OL]. 2020. https://www.kaggle.com/jiashenliu/515k-hotel-reviews-data-in-europe.