Problematic subgroup search in textual data: a free protected data approach

Emanuela Piga
Politecnico di Torino
Turin, Italy

Maria Antonietta Longo
Politecnico di Torino
Turin, Italy

Vito La Piana
Politecnico di Torino
Turin, Italy

ABSTRACT

This study aims to identify problematic subgroups within textual data using subgroup discovery methodologies adapted for tabular data. The approach involves extracting statistical metadata from texts to transform them into a structured tabular format, facilitating analysis with the DivExplorer algorithm [2] for subgroup discovery and analysis.

In defining uncertain predictions, a BERT-based model trained for NLP sequence classification is utilized. Predictions are classified as uncertain if their associated probability falls below a predefined threshold p^*.
This threshold is chosen to capture at least 10% of the test set, ensuring a significant subset for analysis.

During experiments conducted on two datasets— the "IMDB Dataset of 50K Movie Reviews" and "Twitter US Airline Sentiment"—lower divergence values were observed in longer texts, indicating higher model confidence with more complex texts.

This study demonstrates the effectiveness of subgroup discovery techniques applied to textual data, providing a framework to analyze and address biases or uncertainties in model predictions across diverse text datasets.

1 INTRODUCTION

Recent advances in subgroup discovery have revealed new ways to find patterns in complex data. The innovation of this study is to apply these methods to text data, using statistical metadata instead of personal information.

The methodology unfolds through several pivotal steps: initially, extracting statistical insights from textual content to transform it into a structured tabular format. This process involves deriving metadata such as word counts, sentence structures, and linguistic features directly from the text, thus sidestepping the need for sensitive personal information.

A significant innovation lies in the study's approach to data privacy and ethical considerations. By refraining from utilizing personally identifiable information (PII), the research ensures compliance with the GDPR while still extracting meaningful insights from textual data. This approach not only upholds individual privacy but also enhances the transparency and reproducibility of the analysis.

Central to this exploration is the DivExplorer algorithm [2], which serves as a cornerstone for identifying and analyzing problematic subgroups within the structured tabular data. By focusing on statistical metrics and model predictions rather than personal attributes, the study pioneers a fresh perspective on subgroup discovery in text analytics.

2 LITERATURE REVIEW

Machine learning models can perform differently across various subgroups, the subject of various studies is to identify those subgroups that lead to low performance. Several techniques have been developed for this purpose.

Paper [3] implement the Slice Finder technique to identify problematic slices in data by clustering similar examples based on feature-value pairs. It aims to pinpoint slices where the loss function yields significantly higher losses compared to the dataset average. This is determined using Welch's test for statistical significance. An algorithm for identifying problematic subgroups is implemented in [2]. Here the divergence is defined as the difference between a statistic calculated on an itemset and a statistic calculated on the entire dataset. This statistic reflects false positives or false negatives. DivExplorer only considers itemsets that have a certain frequency. Paper [4] addresses the scalability problem, presenting an enumeration algorithm that prunes, enumerates, and evaluates all slice candidates per level with coarse-grained matrix multiplications, aggregations, and element-wise operations. This algorithm is based on some pruning techniques, derived from defining lower and upper bounds for slice sizes, errors, and scores.

In paper [1] a technique is proposed that considers all subgroups that can be defined through a set of metadata, used in speech models. Speech data often comes with additional information about the speaker (e.g., age), recording conditions (e.g., noise level), or task characteristics (e.g., uttered intent). This information is defined as speech metadata. Combinations of metadata values identify data subgroups. This technique is based on divergence (as previously defined) and gain, i.e., the performance increase in a dataset when moving from one model to another. DivExplorer is also used here to identify subgroups with high divergence and gain in absolute value.

Paper [5] presents two approaches for identifying representation bias in the NLP literature. The first is based on performance and representation differences among sensitive groups. We have a list of words that are more problematic. The strategy is to investigate skewed occurrences across classes. If a term appears in many training samples belonging to the toxic class, it encourages models to classify a comment containing that particular term as toxic.

The second approach is based on analyzing sub-space embeddings of sensitive attributes: initially, a set of gender-specific words is chosen as seed words. Using the seed words, an SVM classifier is trained to get the rest of the gender-specific words. Having the gender-specific and gender-neutral words separated, they select seed word pairs such as he-she to act as the x-axis to identify the gender subspace. By checking the distance of gender-neutral words from the he or she end of the axis, it is identified how biased the word embeddings are toward such words.

Paper [6] addresses gender bias and various methods for identifying it. For example, by changing the gender of the gender nouns or more generally swapping each male-definitional word with its respective female equivalent and vice-versa. If the model does not make decisions based on genders, it should perform equally for both sentences. Otherwise, the difference in evaluation scores reflects the extent of gender bias found in the system.

3 RESEARCH GAPS

All these techniques still present some gaps. For example a problem with Slice Finder is that it is not complete since the search for problematic itemsets is pruned whenever sufficiently problematic itemsets are found. Therefore longer (more specific) itemsets, even if more problematic, can be missed. Another limitation of this technique is that it relies on clustering, which can be challenging in the case of large datasets. Additionally, the number of clusters must be specified by the user, which can also be a critical issue.

Approaches that use DivExplorer [1, 2] have a scalability problem, as the search could result in an exponential number of subgroups. To solve this, parallelization or dataset sampling can be used (which, however, could increase the risk of false positives and false negatives). Moreover, the efficiency of the algorithm depends on the efficiency of the FPM algorithm within it.

Some approaches [5, 6]use word representations and measures such as cosine similarity to determine problematic subgroups, but this results in a loss of interpretability.

It is important to note that standard evaluation datasets in NLP are inadequate for measuring gender bias. These datasets often contain biases, so evaluation on them might not reveal gender bias. The solution is to design datasets to isolate the effect of gender in the output to probe gender bias. These datasets are called Gender Bias Evaluation Testsets. Some argue that this might be a limitation because the use of artificial datasets does not reflect the true distribution of the data.

What the previous papers have in common is that, for subgroup analysis, they primarily focus on gender and ethnicity. Instead, we will try to use statistics extracted from the data.

4 METHODOLOGY

4.1 Overview

The objective of this study is to identify problematic subgroups within textual data. To achieve this, we extract metadata from the text and compile the extracted information into a table. This process allows us to convert textual data into tabular data, enabling us to leverage existing subgroup discovery methodologies for tabular data. For instance, we can use the DivExplorer algorithm presented in [2].

An innovation lies precisely in the extraction of metadata: these are not annotated and already present in the dataset, but are derived from the text by inspecting statistics based on them.

The entire methodology is designed to be applicable under the assumption of a binary classification problem.

The general idea is to obtain predictions from the textual data along with the confidence level of these predictions, expressed by their likelihood.

We create a tabular dataset from the textual data by adding statistical metadata and a special feature called "uncertainty" that indicates whether a prediction is certain or not. Problematic subgroups can then be identified by analyzing this tabular data.

In the following subsections, we describe in detail the steps just outlined.

4.2 Definition of uncertain prediction: choice of p^*

We will refer to a general subgroup simply as a set of attribute-value pairs, or itemsets. A problematic, untrustworthy or unfair subgroupis a set of such pairs for which a binary classification model makes a prediction with a certain confidence, but this confidence is smaller than a specified threshold value p^*.

In greater detail, our method operates within the scope of binary classification: to generate predictions, we leveraged a pre-trained BERT model designed for NLP sequence classification. Specifically, the model used is BERT-base-uncased, tailored for sequence classification tasks with two distinct labels.Since BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained neural network model renowned for its ability to capture contextual relationships among words within a sentence using a Transformer architecture, it is well-suited for our goal of obtaining predictions and their associated probabilities.

After training the model on 80% of the preprocessed texts, all subsequent analysis utilizes only the test set.

In binary classification contexts, when an item x is classified as belonging to class y instead of y', with a probability of 50.1%, this implies that there is a 49.9% of probability that x belong to y' class. The difference between this probabilities is very low and the prediction related to x is uncertain.

According to binary classification hypothesis, the minimum probability achievable is exactly 50%

We define as uncertain all inputs x such that $f(x) = y$ with $p \leq p^*$, where $f(x)$ is the BERT model mentioned above.

Once predictions and their associated probabilities are obtained for the test set, we classify items with probabilities between 50% and a specified threshold value p^* as *uncertain*.

Algorithm 1 outlines the main steps of the choice of p^*

We decided not to fix a universal value, but to set p^* such that approximately 10% of the test set is characterized by uncertain probability values: we set this target value (10%) because it allowed us to work with a sufficient number of data points. However, for larger datasets, the target can be lowered, and for smaller test sets, it can be increased.

This p^* is chosen from the set $P = \{0.5, 0.55, 0.60, 0.65, \ldots, 0.95, 1\}$; p^* is the minimum admissible value such that the sum of the test items ordered by ascending probability value has reached 10% of the cardinality of the test set.

As Figure 1 shows, $p^* = 0.7$ is chosen for the *IMDB dataset* with 8K texts for the training set and 2K texts for the test set.

The next step involves creating metadata for the test set based on the information obtained in the previous step.

Algorithm 1 Finding p^* that captures at least 10% of the test set

Require: $p = \{0.5, 0.55, 0.6, 0.65, \ldots, 0.95, 1\}$ ▷ Set of probability thresholds
Require: $test\_set$ ▷ Test set with a 'probability' column
Require: $target\_percentage = 10$ ▷ Desired percentage of rows
Ensure: p^* ▷ Probability threshold that captures at least 10% of the test set

$num\_rows\_target \leftarrow \frac{target\_percentage}{100} \times \text{len}(test\_set)$
$cumulative\_rows \leftarrow 0$
$test\_set\_sorted \leftarrow \text{sort}(test\_set, 'probability')$
for $i \leftarrow 1$ to $\text{len}(p) - 1$ **do**
    $rows\_interval \leftarrow \text{count\_rows\_between}(test\_set\_sorted, p[i-1], p[i])$
    $cumulative\_rows \leftarrow cumulative\_rows + rows\_interval$
    if $cumulative\_rows \geq num\_rows\_target$ **then**
        $p^* \leftarrow p[i]$ ▷ First $p[i]$ where cumulative rows reach at least 10% of test set
        break
    end if
end for
return p^*
function COUNT_ROWS_BETWEEN(test_set, lower_bound, upper_bound)
    return number of rows in test_set where lower_bound ≤ probability < upper_bound
end function

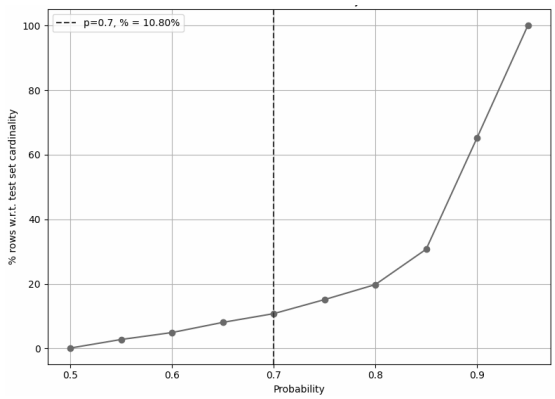

Figure 1: % rows of test set as the probability of the prediction for IMDB data changes

4.3 Adding metadata

Once p^*has been determined, we can transform the problem of finding ~~unfair~~ subgroups in textual data into a problem of finding ~~untrustworthy~~ subgroups in tabular data.

To achieve this, we aim to provide an interpretable representation of textual data that does not involve personal or sensitive information. To add metadata, we choose an approach that does not require the involvement of a Data Privacy Officer, or similar professionals in the field of sensitive data, and it doesn't require an annotated dataset as the metadata related to the text are not sensitive but rather statistics and other information directly extractable from the text itself.

The first metadata attribute relates to the uncertainty of the prediction: each textual data point in the test set will have a binary feature called "uncertainty" that is set to 1 if the predicted probability for the label of that text was less than or equal to p^*, and 0 otherwise.

The other metadata attributes are basic statistics, for example the number of words, the number of sentences, the presence of punctuation, the presence of uppercase characters, and metrics like readability, term frequency-inverse document frequency (TF-IDF), and the number of distinct Named Entities (NER) present in each text.

In the phase of adding metadata to revert to a tabular data setting, we acted as domain experts, assuming that certain textual features significantly influence the model's sentiment prediction. Specifically, for TF-IDF, as the experiments and analysis section shows, we presumed that certain words have a substantial impact on the prediction.

Once this is done, the problem of unfair subgroup discovery in textual data transforms into a problematic subgroup discovery problem for tabular data.

4.4 Unfaithful subgroup research

The advantage of this approach is that we can leverage existing and efficient subgroup discovery methods from an explainability standpoint. In textual data, for instance, subgroup discovery can involve methods like cosine similarity, as explained in [5, 6]. However, as highlighted in the research gap section, this approach may lead to a loss of explainability.

In contrast, for tabular data, we can perform subgroup discovery using algorithms like DivExplorer, which allows us to conduct this search while preserving explainability. For this reason the following section contains some experiments and analysis by using the divergence concepts presented in [2], for us the boolean outcome used is "uncertainty", which indicates whether the model prediction is below the confidence threshold found p^*. As explained before, this is represented by a binary value: 1 if the confidence is less than or equal to p^*, 0 otherwise.

5 EXPERIMENTS AND ANALYSIS

5.1 Overview

To evaluate the proposed methodology, we used two datasets: the "IMDB Dataset of 50K Movie Reviews", we will refer to this by the name "Reviews" and a collection of tweets regarding an airline company's services titled "Twitter US Airline Sentiment" we will refer to this by the name "Tweets"; both are sourced from Kaggle. The choice of these datasets is not random but is motivated by our interest in testing the proposed methodology on datasets that are very different from each other in terms of text's length, since Tweets dataset contains shorter texts than Reviews dataset.
Our method relies on statistics based on the size of textual data in terms of the number of words and the elements that make up a sentence.
We expect that the inherent differences in the nature of the texts will

lead to variations in the final experimental results. This is because the differences in text length will influence the data discretization, and the performance of the DivExplorer is highly dependent on that.
For both the datasets the target variable is the sentiment expressed in the reviews or tweets. For each dataset, we randomly selected 10,000 texts, ensuring that the datasets remained balanced.

5.2 Metadata addiction and p^* computation

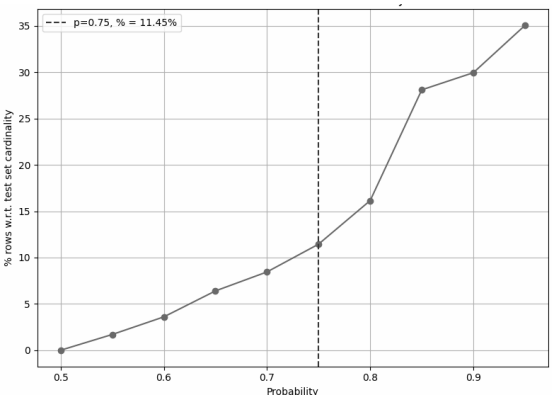

Figure 2: % rows of test set as the probability of the prediction for Tweets data changes

By applying Algorithm 1 presented in Section 4.2, the value of p^* for Tweets is $p = 0.75$, as shown in Figure 2. For Reviews, the value of p^* is illustrated in Figure 1.
We took on the role of domain experts and selected twenty words that we believed were important for sentiment prediction. We then calculated the TF-IDF for these words. Each of these words was subsequently used as a feature in the test dataset. Table 1 lists the chosen words and the frequency of each word in the test dataset for both datasets. The reason behind much higher frequencies in the Reviews dataset is due to its texts being longer than those in the Tweets dataset.
By acting again as domain experts, we selected specific statistics to compute from the data and designated them as features for the datasets. For both datasets, the chosen relevant features include the previously mentioned TF-IDF values, the original text, the prediction outcome, the prediction probability, the "uncertainty" feature detailed in Section 4.4, the true label, the number of words in the texts, the number of characters, the number of sentences, the average word length, the average sentence length, readability, and named entity recognition (NER). Additionally, for the Reviews dataset, we included the count of punctuation marks as a feature. For the Tweets dataset, we incorporated four distinct features: the count of exclamation marks, the count of question marks, the count of periods, and the count of words in uppercase, since for shorter texts and in a social media context, these punctuation signs have a great relevance.

TF-IDF WORD	REVIEWS	TWEETS
love	347	39
great	504	51
excellent	144	4
amazing	86	20
awesome	32	19
fantastic	40	6
enjoyed	99	1
best	386	26
wonderful	104	5
favorite	92	2
bad	472	26
worst	183	38
boring	115	0
terrible	121	19
awful	96	7
disappointing	32	2
waste	97	1
poor	138	14
hate	60	8
mediocre	30	0

Table 1: TF-IDF words and their count in both datasets

5.3 Subgroups search: DivExplorer application

Finally, we can apply the unfair subgroup discovery algorithm to both datasets and perform a comparison. For both datasets, we choose a very low minimum support threshold. This is because, due to the computational power of our devices, we have worked with a reduced number of texts.

One of our goals is to demonstrate the significant differences in applying the same method to two datasets that differ in text length. Thus, we decide to compare the results by fixing the same minimum support value for both datasets and using similar discretization methods.

Here are the steps we follow for each feature:

- Calculate Bounds: Determine the upper bound as the highest value for the feature and the lower bound as the lowest value for the feature.
- Divide Intervals: Divide each interval into four equal parts.
- Encoding: Based on which interval a value falls into, encode it as "very small," "small," "large," or "very large."

By using this approach, we can consistently compare the results across the two datasets, highlighting how the differences in text length affect the outcomes of the unfair subgroup discovery algorithm. In the following experiments, we removed unnecessary features such as "probability", "label" and so on. TF-IDF features have been removed as well.
Then we applied DivExploer to the remain attributes with two different values of minimum support parameter. Table 2 shows the top three most divergent and problematic subgroups for the Tweets dataset when the minimum support parameter is set to 0.1.
It is immediately evident that the model encounters difficulties

in predictions when the number of entities, the average sentence length, the number of sentences and the number of question marks are very short.

ITEMSET	DIV	SUPPORT COUNT
NER=very short, *avg_s_l*=very short, # sentences=very short	0.144	224
NER=very short, #quest=short, *avg_s_l*=short, # sentences=very short	0.144	224
avg_s_l=very short, # sentences=very short	0.137	258

Table 2: DivExplorer applied to Tweets dataset with min support = 0.1

Table 3 shows the top three most divergent and problematic subgroups for the Reviews dataset.

ITEMSET	DIV	SUPPORT COUNT
avg_s_l=very short, # chars = short, # punct = very short, # sentences=very short	0.026	223
NER=very short, #quest=short, *avg_s_l*=short, # sentences=very short	0.025	225
#chars = short *avg_s_l*=very short, # punct = very short	0.023	251

Table 3: DivExplorer applied to Reviews dataset with min support = 0.1

Contrary to earlier observations, in the Reviews dataset the trends are less clear-cut. Specifically, the feature-value pair "short" is already present in the very first itemset. Additionally, among the first 10 itemsets considered problematic, the feature-value pair "long" appears as early as the fifth itemset. This suggests that the distribution of these feature values deviates from earlier: in this case we observe more straightforward patterns.

Setting the value of the minimum support to 0.01 will lead to itemsets with an higher number of features, as shown in Table 4 and in Table 5. The results in terms of problematic subgroups and the values of these features are the following: for the Tweets dataset the most problematic features are those with values "short" and "very short", while for the Reviews dataset the values "long" and "very long" are present.

By comparing the tables for the two datasets, we clearly observe a lower divergence value for the Reviews dataset. It implies that the model is more confident in making predictions on longer texts.

An additional advantage of the chosen approach is the interpretability of the results. Through the use of the Shapley Value we

ITEMSET	DIV	SUPPORT COUNT
NER=very short, *avg_s_l*=very short, *avg_w_l*=very short, #excl=very short, #upperCase=very short, #words=short # chars= short	0.294	22

avg_s_l=very short,
avg_w_l=very short,
#chars=short,
#quest=very short,
#excl=very short,
NER=very short,
#upperCase=very short,
#words=short
	0.294	22

avg_w_l=very short,
avg_s_l=very short,
#chars=short,
#excl=very short,
NER=very short,
#words=short
	0.294	22

Table 4: DivExplorer applied to Tweets dataset with min support = 0.01

ITEMSET	DIV	SUPPORT COUNT
readability$_{=verylong}$, *#punct = long*, *NER = long*, *#sent = short*, *#chars = short*	0.104	33
readability$_{=verylong}$, *#punct = long*, *#chars = short*, *NER = long*	0.104	33
readability$_{=verylong}$, *#punct = long*, *NER = long*, *#sent = short*, *#chars = short*, *avg_s_l = short*	0.104	33

Table 5: DivExplorer applied to Reviews dataset with min support = 0.01

can see for each itemeset which feature-value pair contributes the most to the divergence. Fig. 3 shows this for the first itemest of the Review dataset when the minimum support is set to 0.01. In this case the most contributing feature-value pair is the number of named entities with value equal to "long". DivExplorer is an algorithm that works with discretized data, for this reason is important to explain our choices in the discretization process. The reason we chose to apply the same discretization method was to ensure a fair comparison between datasets.

Another feasible approach could have been to differentiate subsets

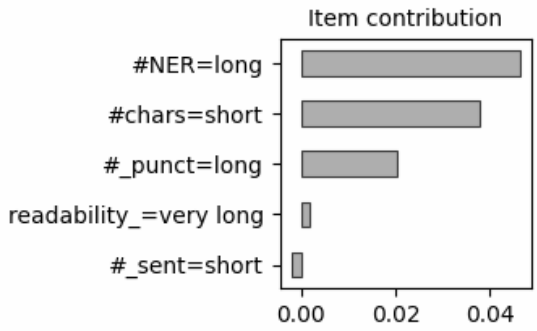

Figure 3: contributions of each feature to the divergence for Reviews dataset and minimum support = 0.01

based on a human perspective: it's easy for a human to distinguish at a glance between a 10-word text and a 50-word text, but nearly impossible to differentiate between a 300-word and a 400-word text. However, such a discretization approach might support plausibility but not faithfulness.

In essence, using consistent discretization allowed for a more direct comparison across datasets, ensuring that differences observed in model behaviors or performance were attributable to the datasets' inherent characteristics rather than variations in discretization methods. While alternative methods could enhance plausibility by aligning more closely with human perception, they might sacrifice faithfulness, or the accurate representation of data characteristics, which was crucial for our analysis. This reflection is the reason why we decided to discretize by following the steps explained before.

6 CONCLUSIONS

In this work we transformed textual data in tabular data and applied DivExplorer algorithm to it. In this way we observed how subgroups and divergence vary across different datasets. The divergence values observed are notably lower for the review dataset compared to the dataset with shorter texts. This suggests that the model exhibits higher confidence in predictions for texts that are longer and contain more words and punctuation.

This conclusion is supported by both the analysis of the divergent subgroups and the actual divergence values observed. Despite the dataset differences in text length, it's crucial to note that the discretization method applied to both datasets was consistent. For example, when categorizing text length into discrete groups like "very small" (word = very small), the criteria were the same across datasets. However, the absolute maximum number of words varies significantly between the datasets: the tweet dataset has a maximum of 31 words, while the review dataset can reach up to 1009 words.

This discrepancy shows that the model's confidence levels vary with the complexity and length of the text it encounters. In summary, the observed divergence values reflect the model's differing levels of confidence across datasets with varying text lengths.

REFERENCES

[1] Alkis Koudounas, Eliana Pastor, Giuseppe Attanasio, Vittorio Mazzia, Manuel Giollo, Thomas Gueudre, Luca Cagliero, Luca de Alfaro, Elena Baralis, and Daniele Amberti. 2023. Exploring Subgroup Performance in End-to-End Speech Models. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 1–5. https://doi.org/10.1109/ICASSP49357.2023.10095284

[2] Eliana Pastor, Luca de Alfaro, and Elena Baralis. 2021. Looking for Trouble: Analyzing Classifier Behavior via Pattern Divergence. In *Proceedings of the 2021 International Conference on Management of Data* (Virtual Event, China) *(SIGMOD '21)*. Association for Computing Machinery, New York, NY, USA, 1400–1412. https://doi.org/10.1145/3448016.3457284

[3] Neoklis Polyzotis, Steven Whang, Tim Klas Kraska, and Yeounoh Chung. 2019. Slice Finder: Automated Data Slicing for Model Validation. In *Proceedings of the IEEE Int' Conf. on Data Engineering (ICDE), 2019*. https://arxiv.org/pdf/1807.06068.pdf

[4] Svetlana Sagadeeva and Matthias Boehm. 2021. SliceLine: Fast, Linear-Algebra-based Slice Finding for ML Model Debugging. In *Proceedings of the 2021 International Conference on Management of Data* (Virtual Event, China) *(SIGMOD '21)*. Association for Computing Machinery, New York, NY, USA, 2290–2299. https://doi.org/10.1145/3448016.3457323

[5] Nima Shahbazi, Yin Lin, Abolfazl Asudeh, and H. V. Jagadish. 2023. Representation Bias in Data: A Survey on Identification and Resolution Techniques. *ACM Comput. Surv.* 55, 13s, Article 293 (jul 2023), 39 pages. https://doi.org/10.1145/3588433

[6] Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. 2019. Mitigating Gender Bias in Natural Language Processing: Literature Review. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Anna Korhonen, David Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, Florence, Italy, 1630–1640. https://doi.org/10.18653/v1/P19-1159