École Polytechnique Fédérale de Lausanne

Debiasing NLP models and datasets with influence functions

by Junze Li

Master Thesis

Approved by the Examining Committee:

Dr. Pearl Pu
Thesis Advisor

Prof. Dr. Hinrich Schütze
External Expert

Mengjie Zhao, Yubo Xie
Thesis Supervisor

EPFL IC HCI
BC 107 (Bâtiment BC)
Station 14
CH-1015 Lausanne

February 25, 2022

Acknowledgments

I remember the first day I came to Switzerland with expectations for my future study. Now I will reach the destination of my journey at EPFL with a wealth of knowledge, research experience, and unforgettable moments shared with my advisors, colleagues, and friends. I would like to take this opportunity to express my gratitude.

I have been working at HCI Group for two and a half years with Dr. Pearl Pu and Yubo Xie. We have completed many exciting projects and presented our findings at conferences. I would like to especially thank my advisor Dr. Pearl Pu. She always provides me with valuable instructions, encourages me, and helps me find my true passion in research. I am also thankful to Yubo Xie, who always answers my questions timely, guides me patiently, and trusts me when I meet difficulties.

Due to the pandemic situation, it is a pity that I can not move to Munich to meet my colleagues at CIS in person. However, it is still a wonderful and memorable collaboration experience. I would like to thank my external expert Prof. Dr. Hinrich Schütze and my supervisor Mengjie Zhao for their theoretical and technical support, creative ideas and sharing of research experience in NLP.

I am very fortunate to meet a group of outstanding and cheerful friends at UESTC, EPFL, LMU Munich, Roche, and Credit Suisse. I will never forget the hard-working days and glorious moments we went through together.

Most importantly, I would like to thank my family for their unconditional love and support. Every time I think about my family, it gives me infinite strength and confidence to chase my dream. I will love you forever!

Lausanne, February 25, 2022 Junze Li

Abstract

Humor is a magnetic component in daily human-human communication. Teaching machines to recognize and understand humor is beneficial to human-computer interaction systems. Humor is a subjective feeling, which means that the text considered humorous may be offensive to others. However, most state-of-the-art models designed for humor recognition are based on large-scale deep learning models, falling short of interpretablity. Very little work exists to inspect *whether* and *why* a detected humor is rooted from offensive social biases.

In this work, we design an approach to inspect and mitigate *gender biases* in humor recognition systems using influence function, which is a type of example-level interpretable method to compute the influence of each training sample on the prediction for each test sample. By conducting a series of experiments, we found that the influence function is helpful for interpreting the behaviors of humor recognition models and can detect gender-biased training samples. Furthermore, since high-quality labeled humor data is limited, we also tested our approach in the low-resource regime. Finally, we found that gender swapping is a promising method to debias the models and datasets for humor recognition.

Overall, our work is the first step towards interpreting and debiasing methods for humor recognition. Our approach to detect gender bias in humor recognition could also be transferred to detecting other types of social biases.

Contents

Acknowledgments i

Abstract ii

List of Figures v

List of Tables vi

1 Introduction 1

2 Background 3
2.1 Model Paradigms in NLP . . . 3
2.2 Transformer Structure . . . 5
2.2.1 Basic Elements . . . 5
2.2.2 Bidirectional Encoder Representations from Transformers (BERT) . . . 7
2.3 Humor Theory and Recognition Task . . . 8

3 Methodology 10
3.1 Humor Recognition Model . . . 10
3.1.1 Fine-tuning Method . . . 10
3.1.2 Prompting Method . . . 11
3.2 Influence Functions . . . 13
3.2.1 Theory . . . 13
3.2.2 Numerical Methods . . . 14

4 Experiment 16
4.1 Datasets . . . 17
4.1.1 SemEval 2021 Task 7: HaHackathon Dataset . . . 17
4.1.2 Gender Humor Dataset . . . 18
4.1.3 Sexist Tweets Dataset . . . 18
4.2 Comparing Humor Recognition Methods . . . 20
4.3 Sanity Check of the Influence Function Results . . . 21
4.4 Gender Bias in Influential Samples . . . 22
4.4.1 Model Trained on Full Dataset . . . 22

4.4.2 Model Trained on Few-shot Data 25
4.5 Model Debiasing Methods 26

5 Related Work **29**
5.1 Interpretable Methodology 29
5.2 Humor Data 30
5.3 Humor Recognition Methodology 30
5.4 Gender Bias 31

6 Conclusion **32**

Bibliography **34**

List of Figures

2.1 The Transformer architecture 6

2.2 Input representation in BERT model 8

3.1 The model structure in fine-tuning method. 11

3.2 The prompting method for humor recognition. 12

4.1 The offense score distribution of Gender Humor dataset 23

4.2 The offense score distribution of 30 most influential training samples for each test sample 24

4.3 The performance of the model trained on the four datasets 28

List of Tables

2.1 Paradigms in NLP tasks [32] (LM: language model, CLS: classification, TAG: sentence tagging, GEN: sentence generation; Blue rectangle: fully unsupervised learning, Red rectangle: fully supervised learning, Purple tilde: textual prompt; "LM→Task": applying LMs with downstream tasks, "Task→LM": designing prompts for LMs) 4

4.1 Two examples from the SemEval-2021 Task7 HaHackathon Humor dataset 17

4.2 Potential offensive targets and keywords in HaHackathon Humor dataset [34] . . 19

4.3 Two examples from the Sexist Tweets dataset 19

4.4 Humor recognition results of prompting and fine-tuning in the full data and few-shot data settings 20

4.5 Sanity check results of influence function based interpretable method 21

4.6 The average offense score of the influential training samples for humorous/non-humorous test samples (full data) 24

4.7 The average ratio of gender-biased training samples for humorous/non-humorous test samples (full data) 25

4.8 The average offense score of the influential training samples for humorous/non-humorous test samples (16-shot) 26

4.9 The average offense score of the influential training samples for humorous/non-humorous test samples (128-shot) 26

4.10 The most positively/negatively influential training samples regarding to 50 test samples 27

Chapter 1

Introduction

Humor is considered one of the most attractive human behaviors. In our daily life, humor can not only provide entertainment but also relieve mental stress [28]. As humor can regulate the communication between humans, modeling the humor computationally will also promote the human-computer interaction experience [38]. The humor recognition task is the first step to let the machine understand humor, which tries to distinguish whether the input text is humorous or not [35]. However, humor is a type of subjective feeling, and the judgment of humor varies from people's personalities, cultural backgrounds, and commonsense knowledge. In some datasets for humor recognition, the sentence showing offensiveness and prejudice against a particular group (e.g., female, LGBT) may be labeled as a humorous sample, which will cause the potential bias in the humor recognition model [34]. At the same time, the current state-of-the-art models for humor recognition are always based on large-scale deep learning structures. It is becoming extremely tough to debug and interpret such models because of the huge amount of parameters and complex structures, which are described as black-box models. Therefore, in our work, we try to design an approach to detect and mitigate the gender bias contained in the humor recognition model based on interpretable methods for the black-box models.

Human language has an inherently structured logic, and the correct meaning of each word is highly related to the context and sentence elements. Therefore, exploring and visualizing what happens in large-scale language models is crucial to improve the model performance in diverse NLP tasks and widen the applicability of NLP models. This has attracted numerous researchers to design interpretable methods for NLP models [3, 20, 21, 29, 41].

For interpreting NLP models, an intuitive method is to choose some specific examples to test the model and observe commonalities or differences according to the model performance. One type of these instance-specific methods focuses on finding the relation between some tokens in the input text and the output result, which could be called token-level inspection. In the token-level inspection, a saliency map could be constructed for each input text, where the influence of each token on the model output is indicated by a intensity score. This intensity score

could be calculated by the gradient value, attention weight or distance between samples defined in a specific domain [41, 45, 46, 48, 56]. In some lexicon-driven tasks, like sentiment analysis, observing the contribution of each token is useful. For example, given a movie comment "It is a very tedious movie.", the word "tedious" tends to be more influential for the model to predict a negative sentiment. However, in some more complex language understanding tasks, like humor recognition, only checking the contribution of each token is not enough, since the fine-grained interactions between tokens are critical features for the NLP models to learn and understand [7].

To overcome this limitation, we will implement the example-level inspection to interpret the humor recognition model [30]. Compared to the token-level inspection which calculates the importance score for each token in the input text, the example-level inspection ranks all the training samples by the influence score corresponding to the prediction for each test sample [8, 26]. Our main focus is the method of using influence functions, which does not need any modifications of the architecture of deep learning models [26].

Since the prompting method proposed recently achieves impressive performance in many NLP tasks in the low-resource regime [6, 32, 42]. Meanwhile, high-quality human-labeled datasets for computational humor study are hard to construct. Therefore, we will apply the plain BERT model to do humor recognition with two different training strategies (fine-tuning and prompting), and analyze the results from influence functions when the model is trained on all samples or few-shot samples. Our assumption is that if the most influential training samples found by the influence function contain gender bias, it means that the humor recognition model is biased, and the humor prediction may highly depend on the gender-biased samples. Finally, we will try to debias the humor recognition model and dataset by removing or modifying the gender-biased influential samples, but not significantly hurting model performance.

Overall, the main contributions of our work are:

- Compare the influence function based interpretable method for humor recognition models using fine-tuning and prompting training strategies. We find that the prompting method is better in low-resource regimes, and the fine-tuning method is better when enough training samples are provided.
- Design the evaluation method to inspect gender bias contained in humor recognition models based on the influence function results. Under this method, we can find that the most influential training samples for predicting humorous test samples contain gender bias, which indicates that the model highly depends on the gender-biased training samples to make the prediction.
- By comparing the debiasing strategies which mitigate the bias but do not significantly harm the model performance, we find that gender swapping is a promising approach.

Chapter 2

Background

In this chapter, we will introduce the trend of developing algorithms and models for NLP tasks from two aspects: paradigm and structure. Since our work is mainly based on the pre-trained Transformer-based language models, we will explain the techniques and mechanisms implemented in the Transformer structure and the Bidirectional Encoder Representations from Transformers (BERT) model structure.

Our methodologies and experiments are conducted in the humor recognition task, so we will also introduce the humor theory, humor categories and methods proposed in the field of computational humor.

2.1 Model Paradigms in NLP

In the work of Liu et al., authors summarized four paradigms in three NLP tasks, which are shown in Table 2.1 [32]. Before the appearance of "pre-train, fine-tune" paradigm and large-scale pre-trained language models [14], most of the NLP tasks are individually solved as fully supervised learning problems. For example, the models (with/without neural networks) for text classification are trained on the task-specific labeled datasets in a fully supervised way. However, these two paradigms (Paradigm a&b in Table 2.1) are not sufficient to construct high-quality models, due to the insufficient human-designed features or limited amount of labeled data [13, 19, 25, 27].

In 2018, Devlin et al. introduced a novel and successful training paradigm for NLP, i.e., "pre-train, fine-tune" [14]. In this paradigm, a large language model is pre-trained to learn the probability distribution of each word and capture the implicit relationship between the words in one sentence or two adjacent sentences, from a large amount of unlabeled text corpora. Since abundant text resources could be retrieved to pre-train such language models, there is a huge

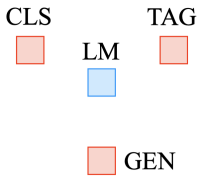
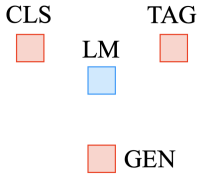
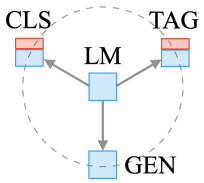
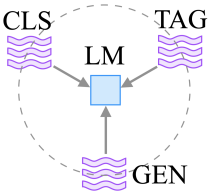
Paradigm	Engineering	Task Relation
a. Fully Supervised Learning (Non-Neural Network)	Feature (e.g., word identity, part-of-speech, sentence length)	
b. Fully Supervised Learning (Neural Network)	Architecture (e.g., recurrent, convolutional, self-attention)	
c. Pre-train, Fine-tune	Objective (e.g., masked language modeling, next sentence prediction)	
d. Pre-train, Prompt, Predict	Prompt (e.g., cloze, prefix)	

Table 2.1: Paradigms in NLP tasks [32] (LM: language model, CLS: classification, TAG: sentence tagging, GEN: sentence generation; Blue rectangle: fully unsupervised learning, Red rectangle: fully supervised learning, Purple tilde: textual prompt; "LM→Task": applying LMs with downstream tasks, "Task→LM": designing prompts for LMs)

improvement of the model performance in many NLP tasks. Therefore, the current text classification tasks are almost always based on the pre-trained language models as different downstream tasks [40]. The process to design the specific training objectives for adapting specific tasks is called "objective engineering".

However, there is another type of training strategy to utilize the pre-trained language models more efficiently proposed recently, which is the "pre-train, prompt and predict" paradigm [18, 32]. In this paradigm, there is no extra neural network structures added to the pre-trained language models and all the tasks could be solved by filling the blanks in different prompts without task-related parameters. The process to design the prompt for a specific task is called "prompt engineering". Overall, the prompting paradigm in NLP tasks requires fewer labeled data and is flexible to transfer among several tasks. In our work, we will design the humor recognition models based on these two paradigms (Paradigm c&d in Table 2.1) and explore their difference with some explainable techniques. In particular, we use "fine-tuning" to denote the "pre-train, fine-tune" paradigm and "prompting" to denote the "pre-train, prompt, predict" paradigm.

2.2 Transformer Structure

In our work, we mainly implement Transformer-based models. Unlike recurrent neural networks that process text in sequence, the Transformer structure applies self-attention mechanism to compute every word within the input text an attention weight that represents the influence each word has on another in parallel [49]. Original Transformer structure consists stacked encoder and decoder, following the autoencoder structure. In text generation tasks, the encoder transforms an input sequence of words $X = (x_1, x_2, ..., x_n)$ into a sequence of continuous representations $z = (z_1, z_2, ..., z_n)$. Given z, the decoder then generates an output sequence of tokens $Y = (y_1, y_2, ..., y_m)$ one token at each time step. The decoder is auto-regressive at each time step, given the previous generated token as an additional input when generating the next. The objective function can be written as

$$p(Y|X) = p(y_1|z) \prod_{t=2}^{m} p(y_t|z, y_1, ..., y_{t-1}). \tag{2.1}$$

However, the Transformer decoder is not necessary for all NLP tasks. In text classification tasks, we only need the Transformer encoder to compute the hidden representation z of each training point and input this representation into the final prediction layer. Figure 2.1 shows the architecture of the Transformer and the basic elements and mechanisms involved in this architecture will be introduced.

2.2.1 Basic Elements

Encoder Stack

The encoder stack consists of 6 identical blocks. There is one multi-head attention layer and one feed forward layer in each identical block. The residual connection is implemented around each of these two layers, followed by a normalization layer [2].

Self-Attention Mechanism

To calculate attention weights in parallel, three vectors, query(q), key(k) and value(v) are generated by multiplying each embedding vector with three mapping matrices. A compatibility function of q with the corresponding k is used to assign a weight for each v. Then the output is calculated as a weighted sum of v. In practice, we pack all the q, k and v vectors into Q, K and V

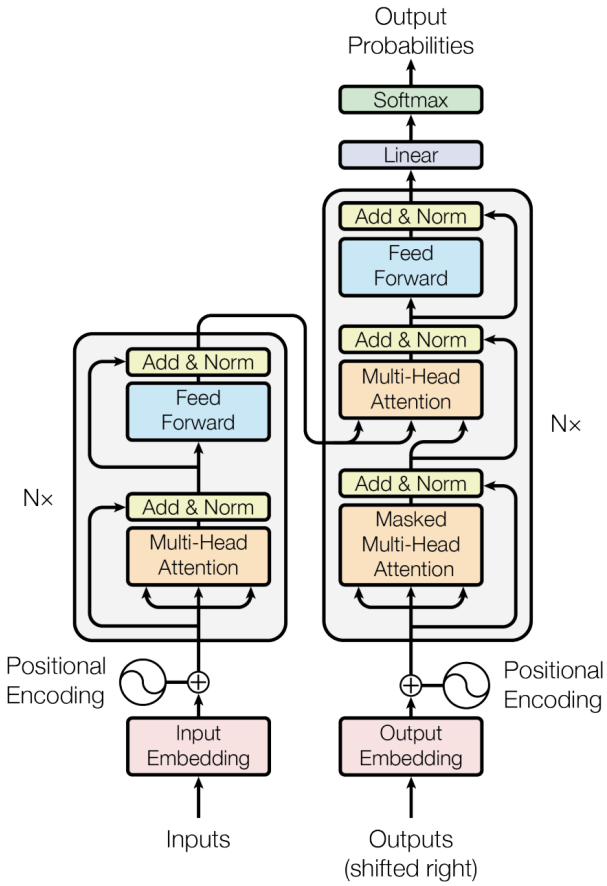


Figure 2.1: The Transformer architecture

matrices. We compute the self-attention α as

$$\alpha(Q,K,V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V, \tag{2.2}$$

where d_k denotes the dimension of q. To extract diverse information from different representation subspaces, the multi-head attention mechanism is designed, which is computed as

$$\beta = \text{concat}(\alpha_1, ..., \alpha_h)W^O, \tag{2.3}$$

$$\alpha_i = \alpha(QW_i^Q, KW_i^K, VW_i^V), \tag{2.4}$$

where β denotes the multi-head attention, W^O, W_i^Q, W_i^K and W_i^V are all projection matrices, which are corresponding to the dimension of q, k and v. In the multi-head attention mechanism, different α_i may focus on different elements of the input sentence to capture and learn more features of the input sentence.

Feed-Forward Network

Besides multi-head attention layers, there are also fully connected feed-forward networks adopted in both encoders and decoders. The function of this feed-forward network can be written as

$$\gamma(x) = \max(0, xW_1 + b_1)W_2 + b_2, \tag{2.5}$$

where γ denotes the output, W_1, W_2 are weight matrices and b_1, b_2 are biases. This layer computes linear transformation twice with a ReLU activation function.

Positional Encoding

Up to now, we do not make use of the sequence order of the input sentence. We implement the positional encoding to extract the position information of the input sentence. The computation of the positional encoding is

$$C_{\text{pos},2i} = \sin(\text{pos}/10000^{2i/d_{\text{model}}}), \tag{2.6}$$

$$C_{\text{pos},2i+1} = \cos(\text{pos}/10000^{2i/d_{\text{model}}}), \tag{2.7}$$

where pos denotes the position and i denotes the dimension of encoding. d_{model} is the dimension of the vector passing through the model.

2.2.2 Bidirectional Encoder Representations from Transformers (BERT)

The BERT model is constructed by a number of Transformer encoders. The base version of BERT model has 12 Transformer encoders with total 110M parameters, and the large version of BERT model has 24 Transformer encoders with total 340M parameters. However, the input representation and the training process are different from the traditional Transformer structure. Therefore, we will introduce these two specific modifications designed for BERT model.

Input Representation

The tokenizer of BERT model implements the WordPiece embeddings, and the symbol "##" is used to split the word pieces [53]. The max length of the input tokens is 512. The first input token is always the "[CLS]", which is the special classification token and is ignored in non-classification tasks. Another special token is "[SEP]", which is used to separate the sentences in a sentence pair. In Figure 2.2, there are two sentences packed together as the input text. The word "playing" is tokenized into the "play" and "##ing" word pieces. A "[CLS]" token is added at the beginning and a "[SEP]" token is appended at the end of each sentence. After tokenization, the input vector of

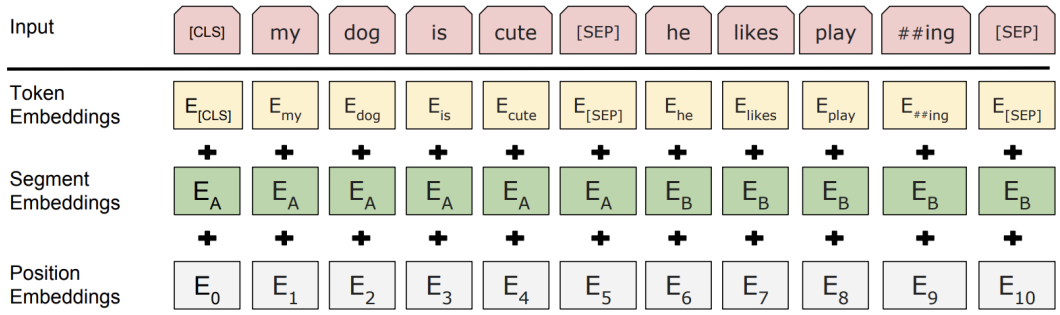


Figure 2.2: Input representation in BERT model

each token is added by token, segment and position embedding vectors. The token embedding and position embedding are the same as those in the original Transformer structure. Additionally, each token in one sentence has the same segment embedding, which is different between two sentences.

Masked Language Model (MLM)

The original sequential language models can only be trained in a uni-directional manner. To extract and learn more features from the training corpus in a bi-directional way, the BERT model randomly masks certain proportion of the input tokens and then predicts these masked tokens. This training strategy is called "Masked Language Model (MLM)". In Devlin et al's work, they randomly mask 15% of the input tokens [14]. The masked token is replaced by the special token "[MASK]", but there is no "[MASK]" token contained in the training corpus for fine-tuning. To deal with this mismatch problem, 10% of the chosen masked tokens are replaced by a random word, 10% of those keep unchanged and the remaining 80% of those are replaced by "[MASK]".

2.3 Humor Theory and Recognition Task

The first attempt to define humor appeared in ancient Greece, where some ideologists and philosophers considered the human's laughter released during the comedy as a form of scorn. Then the *superiority theory* about humor became dominant, which considers that the laughter is a type of superiority over other people's physical defects or shortcomings.

After the 18th century, another two humor theories were proposed, which are the *relief theory* and the *incongruity theory*. In the *relief theory*, the laughter is seen as the relief of pressure, which is the reason why talking about taboo topics (e.g., sexual terms) often causes laughter. The *incongruity theory* focuses on the language semantics, which claims that the incongruous meanings in the same context lead to a humor effect [24, 43]. For example, in a stand-up comedy, there should be a set-up establishing an expectation for the audience and a punchline to violate

this expectation. This type of humor with the form of "set-up + punchline" is widely studied in NLP field, and the corpus in the dataset for this task could be crawled from Reddit forum, Twitter or sitcom scripts [4, 36, 55].

Before the appearance of deep learning models, designing human-centric features is the main method to do the humor recognition task, such as alliteration chain, semantic ambiguity and semantic relatedness [35, 36, 55]. However, most of recent work about humor recognition adopt large-scale pre-trained language models, which always show excellent performance but lacks of interpretability. So in the following chapters, we will utilize the BERT model and explore the interpretable methods and debiasing methods for the BERT model in the humor recognition task.

Chapter 3

Methodology

In this section, we will firstly introduce how to implement the fine-tuning and prompting methods with BERT model for the humor recognition task respectively. Then we will describe the interpretable method based on influence functions, which computes the influence of each training instance on the model prediction in both the fine-tuning and prompting methods. Due to the computational complexity and the properties of BERT model, we will also introduce the estimation and approximation algorithms used in practical numerical experiments.

3.1 Humor Recognition Model

We have introduced the basic elements in the Transformer structure in Section 2.2.1 and the specific modifications in BERT model in Section 2.2.2. All the humor recognition models implemented in our experiments are based on the BERT model structure. However, we will train humor recognition models under two different strategies, which are fine-tuning and prompting.

3.1.1 Fine-tuning Method

In Figure 3.1, we represent the BERT-based model structure in fine-tuning method. For the humor recognition task, a linear classifier layer is stacked on the Transformer encoder. The parameters in the embedding layer and the Transformer encoder are loaded from the pre-trained base version of BERT model, and the parameters in the final linear classifier layer are initialized randomly. Each input text is tokenized by the BERT tokenizer, and the final output prediction is binary indicating the input text is humorous or not.

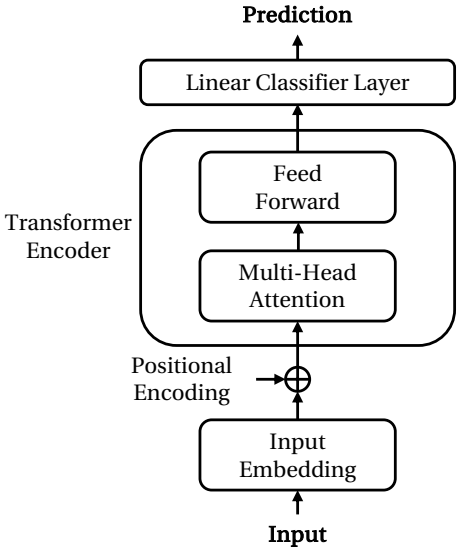


Figure 3.1: The model structure in fine-tuning method.

3.1.2 Prompting Method

In the fully supervised learning paradigm, the training process could be described as $P(y|x;\theta)$, where x denotes the input, y denotes the output and θ denotes the trainable parameters in the model. A labeled dataset is necessary to train such models, but the size of the high-quality labeled dataset is always the bottleneck of the model performance. To overcome this shortcoming, the prompting method models the probability of the input text x directly and predicts the final output y using the MLM heads. In this section, we will introduce three steps involved in the prompting method.

Prompting Function

Firstly, the input x is modified with the designed prompting function $x' = f_{prompt}(x)$, where x' is the prompted input. This prompting function has the following two processing steps:

- Make two slots in the prompting template, which are the input slot $[X]$ for the input text x and the answer slot $[Z]$ for the answer text z. The final prediction y will be mapped from this generated answer z later.
- Put the input text x into the input slot $[X]$; then this prompt is used to query the language model to fill in the answer slot $[Z]$.

For example, in the sentiment analysis task, the input text x is "I enjoyed the science fiction film yesterday.", and the prompted input text x' is "I enjoyed the science fiction film yesterday.

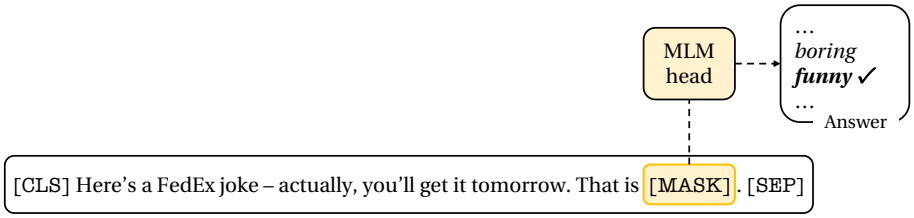


Figure 3.2: The prompting method for humor recognition.

So, it was a $[Z]$ film.". The answer slot $[Z]$ is for the model to fill, which is a token or a sequence of tokens chosen from the vocabulary list. However, the other tokens in the prompting template could be actual tokens in the vocabulary list, embedding vectors in a continuous space or the combination of these two categories. The number and location of the input slot $[X]$ and the answer slot $[Z]$ should be adjusted to the specific task.

Answer Search

For the answer text z, it is chosen from the answer set $\mathcal{Z}$. The values in $\mathcal{Z}$ vary from all the tokens in vocabulary list to a subset of tokens, depending on the task. For example, in the sentiment analysis task, the answer set could be $\{$"$positive$", "$neutral$", "$negative$"$\}$. Or in the machine translation task, the answer set is all the possible tokens. The optimal answer $\hat{z}$ is

$$\hat{z} = \text{search}_{z \in \mathcal{Z}} P\left(f_{\text{fill}}\ \left(x', z\right); \theta\right), \tag{3.1}$$

where the filling function is $f_{\text{fill}}\left(x', z\right)$ used to fill the answer slot $[Z]$. This search function may find the answer with the highest score or generate the answer based on the probability distribution computed from the language models.

Answer Mapping

In this step, the final output $\hat{y}$ is the same as the optimal answer $\hat{z}$, in some generation tasks. But for the other tasks, like sentiment analysis, both the answer "Great" and "Fantastic" could be mapped to a single category "1".

Overall, we can see that there is no other parameters beyond the pre-trained language model should be added in the prompting method. At the same time, one pre-trained language model could be switched to different tasks flexibly. In Figure 3.2, we show the prompting method we designed for the humor recognition task. The MLM head in the BERT model generates the probability distribution of the token at the "[MASK]" position, and then the answer is chosen from this distribution.

3.2 Influence Functions

Deep learning models in NLP are always seen as black-box models, because the large amount of trainable parameters and the lack of interpretability. When we try to interpret these models, we have two directions to do so: inspecting the input texts or inspecting model parameters. In our method, we try to implement the influence functions [11] to check how each training sample influences the final prediction for each test sample through the language models. In this way, we could seek clues about how these models work and which typical training samples do they mostly depend on.

3.2.1 Theory

The influence functions are used to find the relations between the training samples and the predictions of the test samples. Firstly, we define the input space $\mathcal{X}$ (e.g., the input text) and the output space $\mathcal{Y}$ (e.g., the predicted humor label). The training samples in the training set are described as $z_1, \ldots, z_n$, where $z_i = (x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$. In the training process, we want to obtain the optimal parameters $\hat{\theta} \overset{\text{def}}{=} \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)$, where θ denotes the model parameters, $L(z, \theta)$ denotes the loss function.

Intuitively, to evaluate the effect of one training sample on the parameters and predictions of a model, we could observe the difference of the model's outputs when incorporating or excluding this sample in the training process. Based on the variables defined before, we could use $\hat{\theta}_{-z} - \hat{\theta}$ to denote this difference, where $\hat{\theta}_{-z} \overset{\text{def}}{=} \arg\min_{\theta \in \Theta} \sum_{z_i \neq z} L(z_i, \theta)$. However, removing each training sample and retraining the model to calculate this difference is inevitably time-consuming.

Influence functions provide us an effective way to approximate this difference without retraining the model for many times. It approximates the change of the model's parameters after upweighting the training sample z by a small value ϵ. Then the new model's parameters are $\hat{\theta}_{\epsilon,z} \overset{\text{def}}{=} \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta) + \epsilon L(z, \theta)$. The influence of the training sample z on the model's parameter $\hat{\theta}$ after upweighting could be computed by

$$\mathcal{I}(z_i) = \frac{d\hat{\theta}}{d\epsilon_i} = -\left(\frac{1}{n} \sum_{j=1}^{n} \nabla_{\theta}^{2} \mathcal{L}\left(z_j, \hat{\theta}\right) \right)^{-1} \nabla_{\theta} \mathcal{L}\left(z_i, \hat{\theta}\right). \tag{3.2}$$

Now, upweighting a training sample z_i by $\epsilon_i = -\frac{1}{n}$ is equivalent to removing z_i. So the difference of the model's parameter $\hat{\theta}_{-z} - \hat{\theta}$ could be approximated by $-\frac{1}{n}\mathcal{I}(z_i)$ [26].

Then we could measure how this difference of the model's parameter will influence the loss of a test sample by using the chain rule.

$$
\begin{aligned}
\mathcal{I}\left(z_i, z_{\text{test}}\right) &\stackrel{\text{def}}{=} \frac{dL\left(z_{\text{test}}, \hat{\theta}\right)}{d\epsilon_i} \\
&= \nabla_\theta L\left(z_{\text{test}}, \hat{\theta}\right)^\top \frac{d\hat{\theta}}{d\epsilon_i} \\
&= \nabla_\theta L\left(z_{\text{test}}, \hat{\theta}\right)^\top \mathcal{I}\left(z_i\right)
\end{aligned}
\tag{3.3}
$$

Finally, we define the influence score of each training sample to each test sample as $\mathcal{I}\left(z_i, z_{\text{test}}\right)$ and use the z-normalization method to normalize this value for all the training samples in the training set. For each test sample, we could calculate the influence score of each training sample related to this particular test sample.

3.2.2 Numerical Methods

To compute the influence score for each training sample, we should make sure that the model's parameter θ is strictly convex, then the Hessian matrix of θ is positive-definite and invertible. However, neural network structures are not strictly convex. Another condition to guarantee the correctness of the influence function is the convergence of the model during training. Nevertheless, it is still hard to guarantee that the model is always converged because of early stopping. Therefore, we should implement some tricks to compute influence scores.

Stochastic Estimation

In Equation 3.3, we could notice two bottlenecks for computing. Firstly, the computational complexity of the Hessian matrix of $\hat{\theta}$ is pretty high. It requires $O\left(np^2 + p^3\right)$ operations, where n denotes the number of training samples and p denotes the dimension of the model's parameter. Then we need to compute the $\mathcal{I}\left(z_i, z_{\text{test}}\right)$ value *for all the training samples*.

To reduce the computational complexity, we implement the Hessian-vector products (HVPs) to compute the influence score approximately:

$$
\mathcal{I}\left(z_i, z_{\text{test}}\right) = -s_{\text{test}} \cdot \nabla_\theta L(z_i, \hat{\theta}), \tag{3.4}
$$

where $s_{\text{test}} \stackrel{\text{def}}{=} H_{\hat{\theta}}^{-1} \nabla_\theta L\left(z_{\text{test}}, \hat{\theta}\right)$. So we could compute the s_{test} in advance and then compute the influence score for each training sample more efficiently. For estimating s_{test}, we use the LiSSA algorithm to speed up the calculation. More details about the LiSSA algorithm are in the Agarwal et al.'s work [1].

Convex Quadratic Approximation

When we optimize the neural network models using gradient descent with early stopping, the obtained model's parameters $\tilde{\theta}$ are not identical to the optimal parameter $\hat{\theta}$. Then the Hessian matrix of $\tilde{\theta}$ may have negative eigenvalues. So we use the convex quadratic approximation to approximate the loss of $\tilde{\theta}$:

$$\tilde{L}(z,\theta) = L(z,\tilde{\theta}) + \nabla L(z,\tilde{\theta})^{\top}(\theta-\tilde{\theta}) + \frac{1}{2}(\theta-\tilde{\theta})^{\top}\left(H_{\tilde{\theta}} + \lambda I\right)(\theta-\tilde{\theta}), \tag{3.5}$$

where $H_{\tilde{\theta}}$ denotes the Hessian matrix of $\tilde{\theta}$ and λ denotes the added damping term. Then we will use the $\tilde{L}$ to do the computation in Equation 3.3. In Koh and Liang's work [26], they demonstrated that this approximated influence score is added by a constant offset which is independent to the training samples, but is still reasonable. To check that this computational method is feasible to our task, we will conduct the sanity check to the influence function results.

Chapter 4

Experiment

In the experiment section, we will conduct a series of experiments to explore the following research questions about humor recognition methods, influence function results, gender bias phenomenon and corresponding debiasing strategies:

- **RQ1** What is the performance difference between the humor recognition models based on the fine-tuning and prompting methods respectively, when the model is trained on the full training set or few-shot training samples?
- **RQ2** Whether the method to compute influence functions is valid for humor recognition models?
- **RQ3** Whether the influential training samples corresponding to humorous test samples contain gender bias? Or in other words, whether the final prediction highly depends on gender-biased training samples?
- **RQ4** What is the difference between the detected gender-biased training samples in the fine-tuning and prompting methods respectively?
- **RQ5** If we detect potential gender bias in the humor recognition model, how could we mitigate the bias?

Based on the findings obtained by studying these questions, we could build up a human-in-the-loop pipeline to observe, evaluate and mitigate not only gender bias but also other societal stereotypes in the humor recognition task or other NLP classification tasks.

Text	Humor	Humor Rating	Humor Controversy	Offense Score
Why do birds fly south in the Winter? Because its too far to walk!	1	2.5	0	0
Lavender essential oil has a calming scent. it reduces migraines, headaches, anxiety, nervous tension & stress & increases mental activity.	0	-	-	0.1

Table 4.1: Two examples from the SemEval-2021 Task7 HaHackathon Humor dataset

4.1 Datasets

In our experiments, proper and high-quality labeled datasets are a necessity to obtain reasonable results. We leverage the "SemEval 2021 Task 7: HaHackathon Dataset" [34] and the "Sexist Tweets Dataset" [50] to train humor recognition models and the gender bias classifier respectively. The "Gender Humor Dataset" is selected from the "SemEval 2021 Task 7: HaHackathon Dataset", which is used to limit the offensiveness to the gender bias.

4.1.1 SemEval 2021 Task 7: HaHackathon Dataset

In the field of humor recognition study, many humor datasets are constructed by jokes crawled from online forums as positive samples, and the sentences in news or articles as negative samples. There are not many well-designed and high-quality humor datasets annotated by humans. Therefore, we implement the dataset released in SemEval-2021 Task 7: HaHackathon, which is the first human annotated dataset focusing on humor and offense together [34]. 80% texts in this dataset are crawled from Twitter and the remaining 20% are selected from the Kaggle Short Jokes dataset[1]. For the humor and offense annotation, the annotators aged 18-70 are asked to give humor and offense scores (from 1 to 5) for each humorous and offensive text. And if the variance of humor scores is higher than the median of humor scores, this text is labeled as humor controversy. In Table 4.1, there are two examples taken from the HaHackathon Humor dataset.

[1] `https://www.kaggle.com/abhinavmoudgil95/short-jokes`

4.1.2 Gender Humor Dataset

Specifically, in the HaHachathon Humor dataset, a proportion of sentences which are potentially considered offensive is introduced. There are 8 kinds of offense targets and some typical keywords related to each target are shown in Table 4.2. In our work, we limit our focus on the gender bias among all the offensiveness in humor recognition task. We only use the keywords related to sexism to filter out the gender related samples and construct the Gender Humor dataset. We keep the balance of the Gender Humor dataset in the following processing steps:

- For both humorous samples and non-humorous samples, we keep the same proportion of gender related texts and normal texts. In this way, there is no significant correlation between the gender related information and the humor label.
- We keep the same amount of humorous samples and non-humorous samples, and split the dataset into the training set (80%)and the test set (20%).

For example, "Guy: You look nice today! Girl: Didn't I look nice yesterday?" is a humorous and gender-related text, and "Women have affairs because they are in search of emotional fulfillment, an improvement of their self image, and romance...." is a non-humorous but gender-related text. The Gender Humor dataset contains 1,529 training samples and 383 test samples. Then we could implement this dataset to observe the potential classification bias via influence function results. In this way, the gender bias contained in these sentences could be measured by the offense score approximately, and we will use the offense score to indicate the gender bias later.

4.1.3 Sexist Tweets Dataset

Waseem and Hovy [50] created the dataset of hate speech detection, where the texts are crawled from Twitter. There are three labels in this dataset: "Sexism", "Racism" and "None". Based on our purpose to focus on the gender bias, we only select the samples with "Sexism" and "None" labels. Since there are only Tweet ID provided in the original dataset, we use the Twitter API to crawl the text for each Tweet ID. [2]

We next conduct the following pre-processing steps: 1) Remove all the "RT" symbols; 2) Replace "@xxxx" with "hi"; 3) Remove all the hashtags. Finally, we get 5,414 samples, and there are 2,715 (50.15%) sexism labeled samples and 2,699(49.85%) normal samples. We call this dataset as the Sexist Tweets dataset, and split it into the training set (4,331) and the test set (1,083). We also show two examples from the Sexist Tweets dataset in Table 4.3. This dataset will be used to train a BERT classifier to detect the gender-biased text.

[2]Some tweets were deleted by users, so the size of the our dataset is slightly smaller than the original dataset.

Target	Keywords
Sexism	She, woman, mother, girl, b*tch, p*ssy, hooker, slut
Body	Fat, thin, skinny, tall, short, bald, amputee, redneck
Origin	Mexico, Mexican, Ireland, Irish, Indian, Pakistan, China, Chinese, Polish, German, France, Welsh, Vietnam, Asian, American, Russia, Arab, Jamaican, homeless
Sexual Orientation	Gay, lesbian, d*ke, f*ggot, homo, aids, LGBT, trans, tr*nny
Racism	Black, Africa, African, wop, n*****, white people
Ideology	Feminism, leftie/lefty
Religion	Muslim, Islam, Jew, Jewish, Catholic, Protestant, Hindu, Buddhist, ISIS, Jesus, Mohammed
Health	Wheelchair, blind, deaf, r*tard, Steven Hawking, Stevie Wonder, Helen Keller, dyslexic

Table 4.2: Potential offensive targets and keywords in HaHackathon Humor dataset [34]

Text	Sexism
hi Call me sexist, but women should never be allowed to talk about sports on TV.	1
That's a huge serve of pancakes!!! Kids would love that!	0

Table 4.3: Two examples from the Sexist Tweets dataset

	Prompting		Fine-tuning	
	F1-score	**Accuracy**	**F1-score**	**Accuracy**
16-shot	**0.7420**(0.1065)	**0.6786**(0.0557)	0.6732(0.1503)	0.5838(0.0407)
32-shot	**0.7968**(0.0115)	**0.7090**(0.0096)	0.7566(0.0073)	0.6228(0.0137)
64-shot	**0.8036**(0.0052)	0.7321(0.0046)	0.8029(0.0225)	**0.7464**(0.0366)
128-shot	**0.8326**(0.0083)	**0.7808**(0.0084)	0.8107(0.0288)	0.7448(0.0432)
Full data	0.9285	0.9100	**0.9300**	**0.9130**

Table 4.4: Humor recognition results of prompting and fine-tuning in the full data and few-shot data settings

4.2 Comparing Humor Recognition Methods

Firstly, we compared the two BERT-based training strategies, fine-tuning and prompting, for the humor recognition task based on the HaHackathon dataset. For the fine-tuning method, there is a randomly initialized linear layer added to the BERT encoder, and the inputs are the same as the traditional BERT model. For the prompting method, there is no extra layer added to the BERT model, but the inputs should have the following modifications:

- Template: Original text. That is [MASK].
- Answer: funny/normal

Both the fine-tuning and prompting methods load the parameters from pre-trained "bert-base-uncased" model. We trained the models on all of the training samples (full data) and some randomly selected training samples (few-shot) respectively. To obtain the valid and average results in the few-shot setting, we ran each experiment 5 times and calculated the mean and standard deviation of accuracy score and F1-score, which are presented in Table 4.4.

In Table 4.4, we can see that prompting has an overall better performance than fine-tuning in the few-shot setting. The performance difference between these two methods decreases as more training samples used to train the model. For example, the F1-score of prompting is around 7% higher than that of the fine-tuning in 16-shot training. But this difference is only 2% in the 128-shot training. Finally, if we let the model to see all of the training samples, the model performance of prompting method is slightly worse than that of fine-tuning method. Here are two hypotheses to explain these comparison results:

- Since there is no extra parameters involved in the prompting method, the model is easier to converge than the fine-tuning method, when the amount of training samples is limited.
- The prompting method enforces the model to follow a particular pattern through the

Type	Predicting Confidence Difference
Positive	$-6.57\%(\pm 2.86\%)$
Negative	$+1.02\%(\pm 0.61\%)$
Zero	$+0.29\%(\pm 0.68\%)$
Random	$-1.17\%(\pm 0.87\%)$

Table 4.5: Sanity check results of influence function based interpretable method

prompting template. So when enough training samples are fed into the model, this pattern is insufficient to cover all the training samples, which may lead to a worse performance.

As the next step, we will continue to compare the difference between these two methods, when we implement the influence function based interpretable method to observe the influential training samples.

4.3 Sanity Check of the Influence Function Results

Before we compare the influence function based interpretable method results for both fine-tuning and prompting models, we should check whether the results from the influence function are reasonable. We conducted the sanity check for 50 random selected test samples. For each test sample, we could get the influence scores of all the training samples (the influence scores could be positive or negative), and we removed the following 4 types of training samples iteratively:

- Positive: randomly select 10% of the training samples with positive influence scores (positively affect the prediction)
- Negative: randomly select 10% of the training samples with negative influence scores (negatively affect the prediction)
- Zero: select 10% of the training samples with lowest absolute values of influence scores (least influential samples)
- Random: randomly select 10% of all training samples

And then we retrained the model based on the remain training samples. We used the fine-tuning method to conduct this sanity check, since the output of the fine-tuning method is a 2-dimension vector indicating the prediction probabilities for each class, which is more intuitive to describe the prediction confidence (the probability for the correct class).

In Table 4.5, we can notice that when we remove the training samples with positive influence scores, there is a significant decrease of the prediction confidence. On the contrary, when the

training samples with negative influence scores are removed, the prediction confidence increases. However, the positively influential training samples have a more significant impact on the prediction confidence difference. For the training samples with influence scores close to zero, they are relatively not critical for the model prediction, since the absolute value of the prediction confidence difference is the lowest. Interestingly, we can notice that impact of the random selected training samples is larger than that that of the negatively influential training samples. We consider the reason is that the impact of the positively influential training samples is higher than that of the negatively influential training samples, so the impact of the random selected training samples tends to be similar to the results of positively influential training samples and the absolute value of the prediction confidence difference maybe higher than that of the negatively influential training samples. These results are all consistent with our assumption, and we could rely on the influence score results to do further analysis.

4.4 Gender Bias in Influential Samples

After computing the influence scores for each training sample and finding the most influential training samples for each test sample, we want to check whether there exists gender bias in these influential training samples. For the Gender Humor dataset, we only kept the texts which are related to gender. Therefore, to check whether there is a relation between the influence score and gender bias, we could use the offense scores in the Gender Humor dataset to indicate the gender bias. Besides the human annotated offense scores, we also trained a gender bias BERT classifier based on the Sexist Tweets dataset, which is an automatic way to detect the gender bias for influential points. In Section 4.2, we can notice that there is an obvious difference of the model performance between the fine-tuning and prompting methods in full dataset and few-shot settings. Therefore, we checked the gender bias contained in the most influential points in both full dataset and few-shot settings

4.4.1 Model Trained on Full Dataset

In Figure 4.1, there is the histogram of offense scores of all training samples in the Gender Humor dataset. It is a long-tail distribution, where the max offense score is 4.7, and the average offense score is 0.64. After observing the samples with relatively higher offense scores, we considered that if the offense score of one sample is higher than 3.5, this sample is almost certainly gender-biased. And there are 46 training samples with the offense scores over 3.5. Another consideration is that we randomly selected 50 test samples to do the following experiments, so we tried to observe similar amount of most influential training samples. To check the properties of different range of most influential training samples, we checked the offense scores of top 10, 30 and 50 most influential samples in the following experiments.

Based on the randomly selected 50 test samples, we calculated the mean and standard

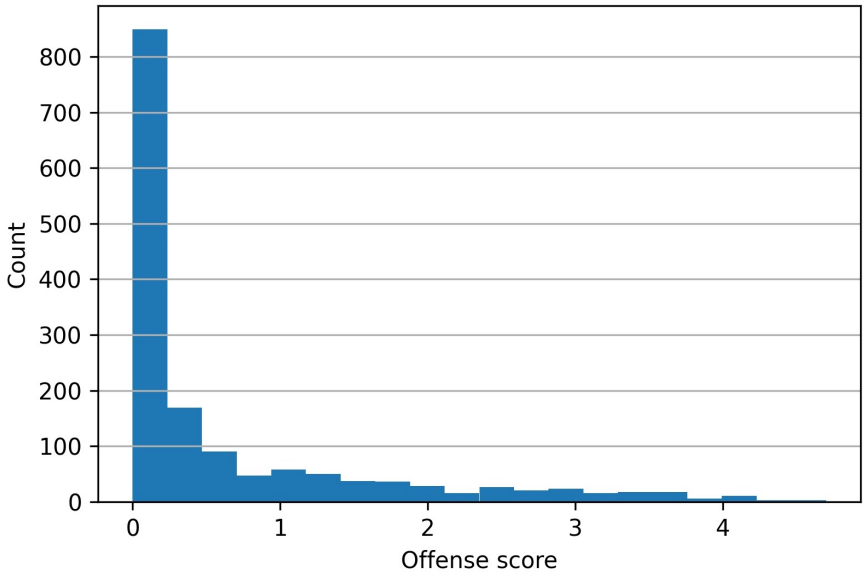


Figure 4.1: The offense score distribution of Gender Humor dataset

deviation of offense scores for the top 10, 30 and 50 most positively influential training samples for each humorous test samples and non-humorous test samples respectively. We also used the average offense score of the full Gender Humor dataset as the baseline. In Table 4.6, we can see that the training samples with higher influence scores tend to have higher offense scores for the humorous test samples, but this relation is adverse for the non-humorous test samples. This trend is the same for both the fine-tuning and prompting models. But the average offense score computed for the prompting model is lower than that for the fine-tuning model, which indicates that the prompting model is less sensitive to the gender-biased samples. Considering the results in Table 4.4, the fine-tuning model is more suitable to detect the potential gender bias contained in the model in the full data setting.

In Figure 4.2, we show the offense score distribution of top 30 most influential training samples for humorous (pos.) and non-humorous (neg.) test samples respectively. Considering the distribution for non-humorous test samples, we can see that most of the influential training samples have low offense scores, which are between 0 and 1. However, the influential training samples for humorous test samples tend to have uniformly distributed offense scores approximately, which means the model is more likely to consider the offensive training samples to do the humor prediction.

We also implemented the Sexist Tweets dataset to train a BERT-based classifier to detect the samples that contain gender bias. After training on the Sexist Tweets dataset, the classifier could achieve the 86.29% accuracy on the test set. To evaluate the model performance on the Gender Humor dataset, we used the test set of the Gender Humor dataset and labeled the samples with offense scores higher than 1 as the sexism samples. And this classifier could achieve 80.93% accuracy on the test samples from Gender Humor dataset.

Range	Avg Offense (Humorous)	Avg Offense (Non-humorous)
Fine-tuning Top 10	1.39(±0.96)	0.45(±0.61)
Fine-tuning Top 30	1.23(±0.93)	0.53(±0.64)
Fine-tuning Top 50	1.16(±0.89)	0.61(±0.69)
Prompting Top 10	1.02(±0.93)	0.22(±0.31)
Prompting Top 30	0.95(±0.82)	0.24(±0.30)
Prompting Top 50	0.94(±0.85)	0.30(±0.31)
Baseline	0.64	

Table 4.6: The average offense score of the influential training samples for humorous/non-humorous test samples (full data)

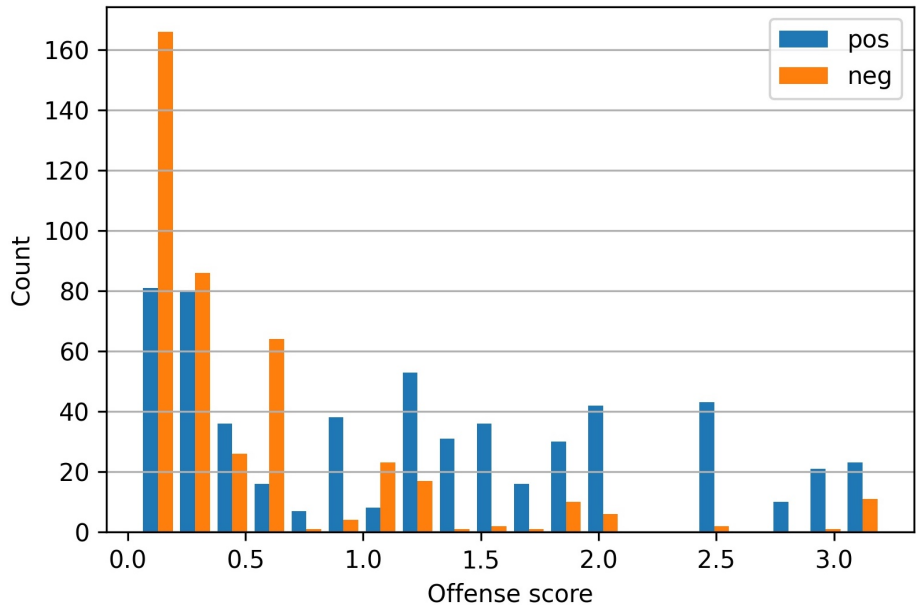


Figure 4.2: The offense score distribution of 30 most influential training samples for each test sample

Range	Avg Ratio (Humorous)	Avg Ratio (Non-humorous)
Fine-tuning Top 10	0.38(±0.15)	0.17(±0.06)
Fine-tuning Top 30	0.36(±0.07)	0.18(±0.04)
Fine-tuning Top 50	0.35(±0.06)	0.20(±0.03)
Prompting Top 10	0.33(±0.14)	0.20(±0.05)
Prompting Top 30	0.32(±0.07)	0.20(±0.06)
Prompting Top 50	0.32(±0.08)	0.21(±0.04)
Baseline	0.21	

Table 4.7: The average ratio of gender-biased training samples for humorous/non-humorous test samples (full data)

Then we implemented this gender bias classifier to detect the proportion of gender-biased training samples among top 10, 30 and 50 most influential training samples for humorous test samples and non-humorous test samples respectively.

In Table 4.7, we can see that in both fine-tuning and prompting models, all the proportions for humorous test samples are higher than that for non-humorous test samples. Specifically, the training samples with higher influence scores tend to be more likely to contain gender bias for humorous test samples, and this trend is opposite for non-humorous test samples. Therefore, we could reach the similar conclusions when we use the human-annotated offense score to inspect gender bias.

4.4.2 Model Trained on Few-shot Data

To compare the influence function interpreting results in the few-shot setting, we conducted the same experiments for 16-shot data and 128-shot data respectively. We ran each experiment 3 times and calculated the average results shown in Table 4.8 and Table 4.9.

In the 16-shot fine-tuning experiment, we can notice that the offense score results for humorous and non-humorous test samples are similar, which means that the fine-tuning model in 16-shot does not have the ability to find the gender-biased training samples. In this situation, the fine-tuning model may predict the same humor label for all the samples. However, for the prompting model in 16-shot, the offense score results for humorous test samples are higher than those for non-humorous test samples. Even though both the fine-tuning and prompting models could not recognize humorous samples accurately, the prompting model is more efficient to distinguish the gender-biased training samples for humor prediction.

When we increase the amount of training samples to 128-shot, we can see that both the fine-tuning and prompting models perform better in distinguishing the gender-biased samples for humor prediction. Overall, different from the conclusions in Section 4.4.1, the prompting

Range	Avg Offense (Humorous)	Avg Offense (Non-humorous)
Fine-tuning Top 2	1.44(±1.43)	1.48(±1.43)
Fine-tuning Top 4	1.28(±1.30)	1.29(±1.34)
Fine-tuning Top 8	1.03(±1.14)	1.04(±1.21)
Prompting Top 2	1.19(±1.17)	0.78(±1.05)
Prompting Top 4	1.00(±1.22)	0.58(±0.98)
Prompting Top 8	0.82(±1.06)	0.65(±1.12)
Baseline	0.53	

Table 4.8: The average offense score of the influential training samples for humorous/non-humorous test samples (16-shot)

Range	Avg Offense (Humorous)	Avg Offense (Non-humorous)
Fine-tuning Top 8	1.24(±1.24)	0.84(±1.04)
Fine-tuning Top 16	1.33(±1.27)	0.93(±1.14)
Fine-tuning Top 32	1.33(±1.27)	0.93(±1.14)
Prompting Top 8	1.53(±1.26)	0.59(±0.67)
Prompting Top 16	1.49(±1.25)	0.64(±0.77)
Prompting Top 32	1.30(±1.21)	0.67(±0.76)
Baseline	0.67	

Table 4.9: The average offense score of the influential training samples for humorous/non-humorous test samples (128-shot)

model could be more efficient than the fine-tuning model to detect the gender-biased samples that provide the evidence for humor prediction.

4.5 Model Debiasing Methods

Now we could discover influential training samples with gender bias and notice the humor prediction may highly based on these gender-biased samples. In this section, we want to evaluate the debiasing methods which remove or modify some training samples by observing the differences in model performance. To determine the samples that we want to change, we added up all the influence scores of all the training samples regarding each test sample, since the influence scores have already normalized. In Table 4.10, we present 3 most positively/negatively influential samples. We can see that the most positively influential samples tend to be longer and have obvious gender bias. Then we picked up 100 most positively influential training samples with offense score more than 1 and 100 randomly selected training samples, based on the overall influence score for these 50 test samples. Finally, we built these four training sets by removing or editing training samples:

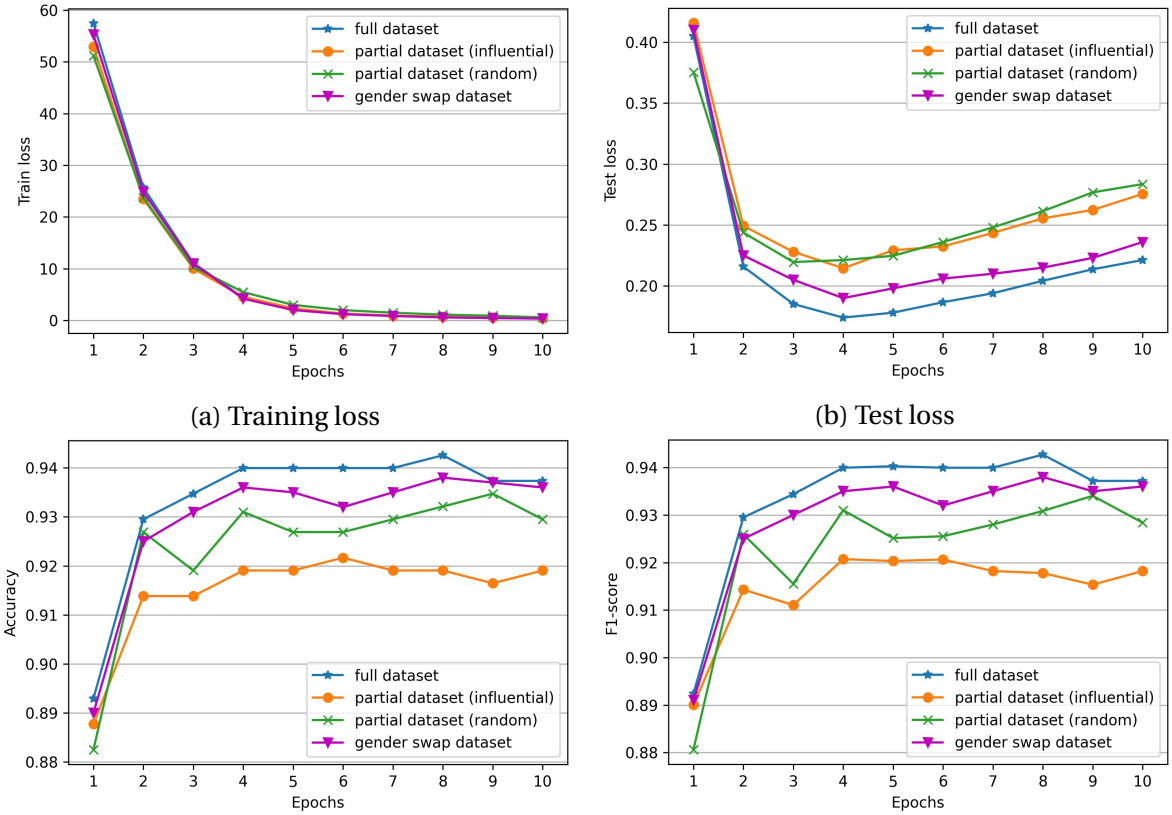
3 Most Positively Influential Samples	Offense Score
- Hv trouble accepting help/support? Feel bottled up inside and ready to blow? Suffering in silence? You're probably a classic Strong Black Woman.	1.3
- Can someone recommend a great at home online workout where I just need some hand weights and a mat. Also i have a treadmill. Not like bands and a trampoline please, I am only one human woman.	0.0
- In many U.S. States offenders receive a harsher penalty for hitting a dog than they do for hitting a woman. That's outrageous either way you're slapping a bitch.	3.2
3 Most Negatively Influential Samples	**Offense Score**
- My girlfriend is dope af.	0.25
- I have no story, I'm just a girl in a bar.	0.0
- I don't know anything about mattresses. What's a good soft one.	0.0

Table 4.10: The most positively/negatively influential training samples regarding to 50 test samples

- Full dataset: The full Gender humor dataset
- Partial dataset (influential): Remove 100 training samples which are most positively influential and have offense scores more than 1
- Partial dataset (random): Remove 100 randomly selected training samples
- Gender swap dataset: Find 100 training samples which are most positively influential and have offense scores more than 1, and swap the gender entities contained in these training samples (e.g., "mother→father", "she→he", "girl→boy" and "woman→man") [58]

We trained the same BERT model based on the fine-tuning method with these four datasets, since the humor recognition model performance in the fine-tuning method is better than that in the prompting method when sufficient training samples are provided. In Figure 4.3a, we can see that the model training speed is similar using these four datasets and the model could be converged after the 10^{th} epoch. In Figure 4.3b, we can see that the model tends to be overfitting after the 4^{th} epoch, which is also similar for the model trained on these four datasets respectively. Overall, there is not an obvious difference in the training and validation processes for the model based on these four datasets.

To observe the difference of humor recognition performance, we show the accuracy values and the F1 scores during training in Figure 4.3c and Figure 4.3d. We can see there is a 2% decrease of both the accuracy value and F1 score for the model trained on the partial dataset (influential), compared to the full dataset. To verify whether this decrease is cause by the size of the dataset or the most influential training samples, we can see that there is around 1% decrease of both the accuracy value and F1 score for the model trained on the partial dataset (random), which is a



(a) Training loss

(b) Test loss

(c) Accuracy

(d) F1-score

Figure 4.3: The performance of the model trained on the four datasets

relatively less decrease. However, the model trained on the gender swap dataset could achieve a similar performance of the model trained on the original full dataset.

Overall, both the methods of removing influential training samples and swapping the gender entities in the influential training samples do not cause a huge decrease in the model performance, and the gender swap method could achieve a similar performance of the model without debiasing.

Chapter 5

Related Work

5.1 Interpretable Methodology

One direction to interpret the NLP models is designing specific measurement methods to compute the influence of each token on the final output in one sentence [3]. As the increasing number of attention-based NLP models, visualizing the attention weights of each token in a sentence could provide the interpretability to some extent [17, 56]. However, some recent work question and discuss the reasonability of such methods depending on attention weights [23, 52]. For the ubiquitous large-scale transformer-based models, only checking the attention weights is not enough to interpret such models.

Another direction is constructing the saliency map for each input sample based on the gradient. Simonyan et al. visualized the saliency map for each input image by computing the vanilla gradient of each class [45]. Shrikumar et al. proposed the DeepLIFT method to compute the contribution of each feature of the input in a backpropagating manner [44]. Some other work proposed adjusted gradient-based methods to increase the robustness [46, 48]. However, Feng et al. found that the gradient-based methods may produce nonsensical results in the input reduction experiments [16].

There are also some other token-level explainable methods. Li et al. measured the contribution and influence of each token by erasing it [29]. Ribeiro et al. proposed the LIME method to build an interpretable model to the local part of the input sample [41].

Regarding the example-level explainable methods, implementing the influence functions to interpret and debias NLP models is our main focus [26]. Croce et al. implemented the Kernel-based Deep Architecture to explain the prediction based on the activated training samples, by projecting the test sample into a space defined by training samples [12]. Card et al. designed deep weighted averaging classifiers to generate the prediction based on the weighted sum of the training samples, and this weight is related to the distance between the training and test samples

[8]. However, all of these methods except influence functions need additional adjustments of the model to determine influential training samples.

5.2 Humor Data

Constructing high-quality humor datasets is the first step to let the machine recognize and understand humor. Mihalcea and Strapparava built the one-liner jokes dataset, where humorous samples are crawled from social media and non-humorous samples are from the news or proverbs [35]. Yang et al. created the humor dataset with the puns crawled from Pun of the Day website [1] as humorous samples, and non-humorous samples are also from the news [57]. Hasan et al. constructed the multimodal humor dataset UR-FUNNY, which includes audio, text and video data acquired from TED talks [22]. Bertero and Fung developed a dialogue-based humor dataset, where the humorous punchlines in the script of BigBang Theory are annotated [4]. Since some jokes may provoke offensiveness to particular listeners, Meaney et al. combined humor and offense together by labeling each sample not only a humor rating but also a offense rating [34]. There are also some other online humor datasets not released in publications, for example, Reddit Jokes dataset, Wocka dataset [2] and Short Jokes dataset [3].

5.3 Humor Recognition Methodology

Many early work about humor recognition are based on the human-designed features. Mihalcea and Strapparava proposed three typical features (alliteration, synonym and antonymy) to distinguish humorous texts [35]. Mihalcea et al. computed the semantic relatedness to recognize humor, which is calculated by the set-up and the punchline in one sample [36]. Morales and Zhai designed a probabilistic model to classify humorous samples, where the normal texts (e.g., news or Wikipedia) are considered as the background sources [37]. Liu et al. divided one sample into several discourse units and calculated the sentiment association between these units as the feature [31]. As the deep learning architectures being popular, deep neural networks replace the methods of human-designed features gradually. The classical deep learning architectures like convolutional neural network and Transformer are implemented in humor recognition tasks [9, 10, 51]. Fan et al. designed the PACGA structure to recognize humor, which introduces additional phonetic and ambiguity information [15]. Xie et al. implemented the DeBERTa architecture to consider the relative position of each token pair as the extra information to improve the humor recognition performance [54].

[1] `http://www.punoftheday.com/`
[2] `https://github.com/taivop/joke-dataset`
[3] `https://github.com/amoudgl/short-jokes-dataset`

5.4 Gender Bias

As NLP models becoming more and more powerful, it is indispensable to debug and check whether the potential societal stereotypes are contained in NLP datasets or models, like gender bias or social bias. Sun et al. introduced and summarized the current observation and evaluation methods for gender bias [47]. Park et al. investigated the gender bias in abusive language detection models and proposed the debiasing methods [39]. Lu et al. defined a benchmark to measure gender bias in NLP tasks and proposed the counterfactual data augmentation (CDA) method to eliminate such bias [33]. Bolukbasi et al. debiased the word embeddings by removing the word associations related to gender stereotypes [5]. Zhao et al. augmented the training dataset by finding the gender related entities and swapping those to the opposite gender, which is effective to mitigate gender bias in the coreference resolution task [58].

Chapter 6

Conclusion

In this work, we first compare two training strategies, fine-tuning and prompting, for the BERT-based humor recognition model, in both full data and few-shot data settings. We further implement the influence function to interpret the model predictions by observing the most influential training samples. Then we inspect whether the model is biased by checking the gender bias contained in these influential training samples, and adopt two debiasing methods to mitigate the bias of the humor recognition model.

Now we can answer the research questions proposed at the beginning of Chapter 4:

- **RQ1** In the few-shot data setting, the prompting method has a better performance than fine-tuning for humor recognition. But in the full data setting, the performance in the fine-tuning method is slightly better.
- **RQ2** The influence function can be used to interpret the humor recognition model. Removing the training samples with positive/negative influence scores leads to the drop/increase of the prediction confidence.
- **RQ3** For the model trained on the Gender Humor dataset, the influential training samples corresponding to humorous test samples indeed contain gender bias, which means this model tends to depend on the gender-biased samples to predict humor.
- **RQ4** In the few-shot data setting, the prompting method is more sensitive to detect gender-biased training samples. But in the full data setting, the fine-tuning method is better. This conclusion is consistent to that of **RQ1**.
- **RQ5** Swapping the gender related entities in the detected gender-biased and influential training samples is a better approach to debias the humor recognition model.

For future work, we will try to speed up the computation of influence functions, since the current computation speed of influence functions is the bottleneck to interpret the model trained

on a larger dataset. We will also implement this interpretable method for other state-of-the-art models in humor recognition. By observing the properties of the influential training samples, we could check whether the novel mechanisms involved in these models are reasonable and efficient. Considering the bias in humor recognition models, we limit our scope to gender bias in this work, but other types of bias (e.g., racial bias) could also be checked through the same approach.

Bibliography

[1] Naman Agarwal, Brian Bullins, and Elad Hazan. “Second-order stochastic optimization for machine learning in linear time”. In: *The Journal of Machine Learning Research* 18.1 (2017), pp. 4148–4187.

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. “Layer normalization”. In: *arXiv preprint arXiv:1607.06450* (2016).

[3] Yonatan Belinkov and James Glass. “Analysis methods in neural language processing: A survey”. In: *Transactions of the Association for Computational Linguistics* 7 (2019), pp. 49–72.

[4] Dario Bertero and Pascale Fung. “A long short-term memory framework for predicting humor in dialogues”. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2016, pp. 130–135.

[5] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. “Man is to computer programmer as woman is to homemaker? debiasing word embeddings”. In: *Advances in neural information processing systems* 29 (2016).

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. “Language models are few-shot learners”. In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[7] Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. “e-snli: Natural language inference with natural language explanations”. In: *Advances in Neural Information Processing Systems* 31 (2018).

[8] Dallas Card, Michael Zhang, and Noah A Smith. “Deep weighted averaging classifiers”. In: *Proceedings of the conference on fairness, accountability, and transparency*. 2019, pp. 369–378.

[9] Lei Chen and Chong Min Lee. “Convolutional neural network for humor recognition”. In: *arXiv preprint arXiv:1702.02584* (2017).

[10] Peng-Yu Chen and Von-Wun Soo. “Humor recognition using deep learning”. In: *Proceedings of the 2018 conference of the north american chapter of the association for computational linguistics: Human language technologies, volume 2 (short papers)*. 2018, pp. 113–117.

[11] R Dennis Cook and Sanford Weisberg. *Residuals and influence in regression*. New York: Chapman and Hall, 1982.

[12] Danilo Croce, Daniele Rossini, and Roberto Basili. “Auditing deep learning processes through kernel-based explanatory models”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 2019, pp. 4037–4046.

[13] Anirban Dasgupta, Petros Drineas, Boulos Harb, Vanja Josifovski, and Michael W Mahoney. “Feature selection methods for text classification”. In: *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2007, pp. 230–239.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).

[15] Xiaochao Fan, Hongfei Lin, Liang Yang, Yufeng Diao, Chen Shen, Yonghe Chu, and Tongxuan Zhang. “Phonetics and ambiguity comprehension gated attention network for humor recognition”. In: *Complexity* 2020 (2020).

[16] Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. “Pathologies of neural models make interpretations difficult”. In: *arXiv preprint arXiv:1804.07781* (2018).

[17] Andrea Galassi, Marco Lippi, and Paolo Torroni. “Attention in natural language processing”. In: *IEEE Transactions on Neural Networks and Learning Systems* 32.10 (2020), pp. 4291–4308.

[18] Tianyu Gao, Adam Fisch, and Danqi Chen. “Making pre-trained language models better few-shot learners”. In: *arXiv preprint arXiv:2012.15723* (2020).

[19] Tushar Ghorpade and Lata Ragha. “Featured based sentiment classification for hotel reviews using NLP and Bayesian classification”. In: *2012 International Conference on Communication, Information & Computing Technology (ICCICT)*. IEEE. 2012, pp. 1–5.

[20] David Gunning, Eric Vorm, Jennifer Yunyan Wang, and Matt Turek. “DARPA’s explainable AI (XAI) program: A retrospective”. In: *Applied AI Letters* 2.4 (2021), e61. DOI: https://doi.org/10.1002/ail2.61. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ail2.61. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/ail2.61.

[21] Xiaochuang Han, Byron C Wallace, and Yulia Tsvetkov. “Explaining black box predictions and unveiling data artifacts through influence functions”. In: *arXiv preprint arXiv:2005.06676* (2020).

[22] Md Kamrul Hasan, Wasifur Rahman, Amir Zadeh, Jianyuan Zhong, Md Iftekhar Tanveer, Louis-Philippe Morency, et al. “Ur-funny: A multimodal language dataset for understanding humor”. In: *arXiv preprint arXiv:1904.06618* (2019).

[23] Sarthak Jain and Byron C Wallace. “Attention is not explanation”. In: *arXiv preprint arXiv:1902.10186* (2019).

[24] Immanuel Kant. “Critique of Judgment, ed. and trans”. In: *WS Pluhar, Indianapolis: Hackett* (1790).

[25] Aurangzeb Khan, Baharum Baharudin, Lam Hong Lee, and Khairullah Khan. “A review of machine learning algorithms for text-documents classification”. In: *Journal of advances in information technology* 1.1 (2010), pp. 4–20.

[26] Pang Wei Koh and Percy Liang. “Understanding black-box predictions via influence functions”. In: *International conference on machine learning*. PMLR. 2017, pp. 1885–1894.

[27] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. “Recurrent convolutional neural networks for text classification”. In: *Twenty-ninth AAAI conference on artificial intelligence*. 2015.

[28] Herbert M Lefcourt and Rod A Martin. *Humor and life stress: Antidote to adversity*. Springer Science & Business Media, 2012.

[29] Jiwei Li, Will Monroe, and Dan Jurafsky. “Understanding neural networks through representation erasure”. In: *arXiv preprint arXiv:1612.08220* (2016).

[30] Zachary C Lipton. “The Mythos of Model Interpretability: In machine learning, the concept of interpretability is both important and slippery.” In: *Queue* 16.3 (2018), pp. 31–57.

[31] Lizhen Liu, Donghai Zhang, and Wei Song. “Modeling sentiment association in discourse for humor recognition”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 2018, pp. 586–591.

[32] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. “Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing”. In: *arXiv preprint arXiv:2107.13586* (2021).

[33] Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and Anupam Datta. “Gender bias in neural natural language processing”. In: *Logic, Language, and Security*. Springer, 2020, pp. 189–202.

[34] JA Meaney, Steven Wilson, Luis Chiruzzo, Adam Lopez, and Walid Magdy. “Semeval 2021 task 7: Hahackathon, detecting and rating humor and offense”. In: *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*. 2021, pp. 105–119.

[35] Rada Mihalcea and Carlo Strapparava. “Making computers laugh: Investigations in automatic humor recognition”. In: *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*. 2005, pp. 531–538.

[36] Rada Mihalcea, Carlo Strapparava, and Stephen Pulman. “Computational models for incongruity detection in humour”. In: *International Conference on Intelligent Text Processing and Computational Linguistics*. Springer. 2010, pp. 364–374.

[37] Alex Morales and ChengXiang Zhai. “Identifying humor in reviews using background text sources”. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2017, pp. 492–501.

[38] Anton Nijholt, Oliviero Stock, Alan Dix, and John Morkes. “Humor modeling in the interface”. In: *CHI'03 extended abstracts on Human factors in computing systems*. 2003, pp. 1050–1051.

[39] Ji Ho Park, Jamin Shin, and Pascale Fung. “Reducing gender bias in abusive language detection”. In: *arXiv preprint arXiv:1808.07231* (2018).

[40] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. “Pre-trained models for natural language processing: A survey”. In: *Science China Technological Sciences* 63.10 (2020), pp. 1872–1897.

[41] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. “" Why should i trust you?" Explaining the predictions of any classifier”. In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016, pp. 1135–1144.

[42] Timo Schick and Hinrich Schütze. “It's not just size that matters: Small language models are also few-shot learners”. In: *arXiv preprint arXiv:2009.07118* (2020).

[43] Arthur Schopenhauer. “The world as will and idea (Vols. I, II, & III)”. In: *Haldane, RB, & Kemp, J.(3 Vols.). London: Kegan Paul, Trench, Trubner* 6 (1883).

[44] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. “Learning important features through propagating activation differences”. In: *International conference on machine learning*. PMLR. 2017, pp. 3145–3153.

[45] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. “Deep inside convolutional networks: Visualising image classification models and saliency maps”. In: *arXiv preprint arXiv:1312.6034* (2013).

[46] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. “Smoothgrad: removing noise by adding noise”. In: *arXiv preprint arXiv:1706.03825* (2017).

[47] Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. “Mitigating gender bias in natural language processing: Literature review”. In: *arXiv preprint arXiv:1906.08976* (2019).

[48] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. “Axiomatic attribution for deep networks”. In: *International conference on machine learning*. PMLR. 2017, pp. 3319–3328.

[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. “Attention is All you Need”. In: *NIPS 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett. 2017, pp. 5998–6008. URL: http://papers.nips.cc/paper/7181-attention-is-all-you-need.

[50] Zeerak Waseem and Dirk Hovy. “Hateful symbols or hateful people? predictive features for hate speech detection on twitter”. In: *Proceedings of the NAACL student research workshop*. 2016, pp. 88–93.

[51] Orion Weller and Kevin Seppi. “Humor detection: A transformer gets the last laugh”. In: *arXiv preprint arXiv:1909.00252* (2019).

[52] Sarah Wiegreffe and Yuval Pinter. “Attention is not not explanation”. In: *arXiv preprint arXiv:1908.04626* (2019).

[53] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. “Google’s neural machine translation system: Bridging the gap between human and machine translation”. In: *arXiv preprint arXiv:1609.08144* (2016).

[54] Yubo Xie, Junze Li, and Pearl Pu. “HumorHunter at SemEval-2021 Task 7: Humor and offense recognition with disentangled attention”. In: *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*. 2021, pp. 275–280.

[55] Yubo Xie, Junze Li, and Pearl Pu. “Uncertainty and surprisal jointly deliver the punchline: Exploiting incongruity-based features for humor recognition”. In: *arXiv preprint arXiv:2012.12007* (2020).

[56] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. “Show, attend and tell: Neural image caption generation with visual attention”. In: *International conference on machine learning*. PMLR. 2015, pp. 2048–2057.

[57] Diyi Yang, Alon Lavie, Chris Dyer, and Eduard Hovy. “Humor recognition and humor anchor extraction”. In: *Proceedings of the 2015 conference on empirical methods in natural language processing*. 2015, pp. 2367–2376.

[58] Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. “Gender bias in coreference resolution: Evaluation and debiasing methods”. In: *arXiv preprint arXiv:1804.06876* (2018).