Reducing Negative Effects of the Biases of Language Models in Zero-Shot Setting

Xiaosu Wang
xswang19@fudan.edu.cn
Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
China

Yun Xiong*
yunx@fudan.edu.cn
Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
China

Beichen Kang
bckang21@m.fudan.edu.cn
Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
China

Yao Zhang
yaozhang@fudan.edu.cn
Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
China

Philip S. Yu
psyu@cs.uic.edu
Department of Computer Science, University of Illinois at Chicago
USA

Yangyong Zhu
yyzhu@fudan.edu.cn
Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
China

ABSTRACT

Pre-trained language models (PLMs) such as GPTs have been revealed to be biased towards certain target classes because of the prompt and the model's intrinsic biases. In contrast to the fully supervised scenario where there are a large number of costly labeled samples that can be used to fine-tune model parameters to correct for biases, there are no labeled samples available for the zero-shot setting. We argue that a key to calibrating the biases of a PLM on a target task in zero-shot setting lies in detecting and estimating the biases, which remains a challenge. In this paper, we first construct probing samples with the randomly generated token sequences, which are simple but effective in detecting inputs for stimulating GPTs to show the biases; and we pursue an in-depth research on the plausibility of utilizing class scores for the probing samples to reflect and estimate the biases of GPTs on a downstream target task. Furtherly, in order to effectively utilize the probing samples and thus reduce negative effects of the biases of GPTs, we propose a lightweight model Calibration Adapter (CA) along with a self-guided training strategy that carries out distribution-level optimization, which enables us to take advantage of the probing samples to fine-tune and select only the proposed CA, respectively, while keeping the PLM encoder frozen. To demonstrate the effectiveness of our study, we have conducted extensive experiments, where the results indicate that the calibration ability acquired by CA on the probing samples can be successfully transferred to reduce negative effects of the biases of GPTs on a downstream target task, and our approach can yield better performance than state-of-the-art (SOTA) models in zero-shot settings.

CCS CONCEPTS

• **Computing methodologies → Natural language processing**.

KEYWORDS

Biases of Language Models, Probing Samples, Calibration Adapter

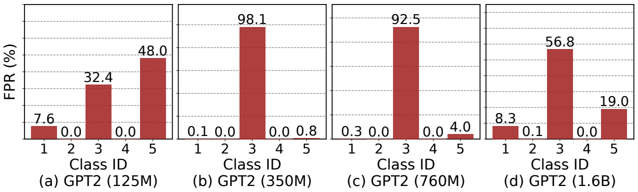
ACM Reference Format:
Xiaosu Wang, Yun Xiong, Beichen Kang, Yao Zhang, Philip S. Yu, and Yangyong Zhu. 2023. Reducing Negative Effects of the Biases of Language Models in Zero-Shot Setting. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM '23), February 27-March 3, 2023, Singapore, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3539597.3570382

1 INTRODUCTION

Benefiting from the development of the prompt-learning technique [3, 20, 25], large-scale pre-trained language models (PLMs), such as GPT2 [22] and GPT3 [3], have achieved impressive zero-shot performance on various natural language understanding tasks [2, 24, 34]. Nonetheless, contextual calibration (CC) [39] has revealed that GPTs [3, 22] are biased towards certain target classes because of the prompt and the model's intrinsic biases, which prevents GPTs from achieving better zero-shot performance. For example, as shown in the Fig. 1(a), for the classes Somewhat Negative (ID: 2) and Somewhat Positive (ID: 4), the false positive rates are both 0.0%; that is to say, there are no samples from other categories that are falsely classified into the class Somewhat Negative or Somewhat Positive by GPT2 (125M). Meanwhile, there are 7.6%, 32.4% and 48.0% samples from other categories that are falsely classified into the classes Very Negative (ID: 1), Neutral (ID: 3) and Very Positive (ID: 5) by GPT2 (125M) respectively. Therefore, we argue that GPT2 (125M) shows biases towards the classes Very Negative, Neutral and Very Positive on the test set of SST5 dataset [28] in zero-shot settings. Fig. 1(b), Fig. 1(c) and Fig. 1(d) illustrate the biases of GPT2 (350M), GPT2 (760M) and GPT2 (1.6B) respectively. In order to cope with the biases and boost zero-shot performance of GPTs, existing

*Corresponding author

WSDM '23, February 27-March 3, 2023, Singapore, Singapore
© 2023 Association for Computing Machinery.
ACM ISBN 978-1-4503-9407-9/23/02…$15.00
https://doi.org/10.1145/3539597.3570382



(a) GPT2 (125M) (b) GPT2 (350M) (c) GPT2 (760M) (d) GPT2 (1.6B)

Figure 1: Illustration of the false positive rate (FPR) for each class predicted by GPT2 of different size on the test set of SST5 dataset [28] in zero-shot settings. Class ID 1, 2, 3, 4 and 5 represent the class Very Negative, Somewhat Negative, Neutral, Somewhat Positive and Very Positive respectively; the number of samples belonging to each class is 279, 633, 389, 510 and 399 respectively. Multiclass data are treated as if binarized under a one-vs-rest transformation. Table 1 shows the prompt template used for SST5 dataset.

researches [12, 16, 39] have demonstrated that calibrating output probabilities of GPTs is an effective technical method.

Intuitively, contextual calibration (CC) [39] proposes that, for a "content-free" sample, such as N/A, [MASK], the empty string, etc., ideally GPTs should score it equally over the target classes; consequently, CC argues that, when feeding in a content-free input, the biases towards certain classes can be reflected and estimated by the actual class scores. For example, when feeding in the content-free sample N/A, GPT2 (125M) should score it as 0.2 Very Negative, 0.2 Somewhat Negative, 0.2 Neutral, 0.2 Somewhat Positive and 0.2 Very Positive (over the class set of SST5 dataset[1]) in an ideal non-biased situation; but, actually, GPT2 (125M) scores N/A as 0.052 Very Negative, 0.007 Somewhat Negative, 0.890 Neutral, 0.002 Somewhat Positive and 0.049 Very Positive. The class Neutral gets the highest score 0.890, indicating that the content-free sample N/A stimulates GPT2 (125M) to show the bias towards the class Neutral; and the actual class scores are considered to be able to reflect and estimate the biases towards corresponding classes of GPTs.

Contextual calibration (CC) [39] shines a light on how to exploit content-free samples to gain calibration ability. For each content-free sample, CC proposes to compute a corresponding affine transform [10, 21] to make the class scores uniform, and then averages the parameters of these learned affine transforms to obtain a final affine transform; the final affine transform model is transfered to calibrate the output probabilities of GPTs on a downstream target task, see Zhao et al. [39] for more details. However, it is important to note that the class scores are contextual: different inputs will lead to different class scores. CC only exploits a few number of content-free samples, e.g. three samples, which is too few to correctly reflect the distribution of content-free data, nor to stimulate GPTs to show all the same biases as GPTs show on a downstream target task. For instance, the three content-free samples (N/A, [MASK] and the empty string) employed by CC are all classified into the class Neutral of SST5 dataset [2] by GPT2 (125M) in zero-shot settings; this result can not reflect the biases towards the other classes (Very Negative and Very Positive) of GPT2 (125M) on the test set of SST5, as shown in the Fig. 1(a). Thereby how to detect and estimate the biases of PLMs remains a challenge. In addition, the small sample size can easily result in overfitting the training data and underfitting the testing data; and furthermore, we argue that the method of averaging the parameters of multiple affine transforms causes CC to have less access to good generalization ability and strong transferability, which both restrict CC from reaching its full potential. As will be shown in Fig. 6, for each unseen content-free sample, CC tries to make the corresponding class scores uniform but achieves much less success than the model (namely CA) proposed in this paper. More comparisons between calibration performance of different models will be shown in Section 5.

In order to overcome the aforementioned flaws, we first construct probing samples with the randomly generated token sequences instead of some manually selected special tokens (e.g. N/A or [MASK]), and these token sequences can be decoded from randomly sampled token ID sequences (Section 3 will show more details); in this way, we can easily obtain a large number of probing samples at low cost, and they are naturally content-free. These constructed probing samples are simple but effective in detecting inputs for stimulating GPTs to show the biases, and we pursue an in-depth research on the plausibility of utilizing class scores for the probing samples to reflect and estimate the biases of GPTs on a downstream target task. Furtherly, to effectively utilize the probing samples and thus reduce negative effects of the biases of GPTs, we propose a novel model Calibration Adapter (CA), which consists of multiple channels corresponding to the target classes respectively. Each channel of CA is designed to estimate the bias towards corresponding class when feeding in an input, and then the actual class scores for the input are corrected by being subtracted from the corresponding bias estimations respectively. Inspired by CC and considering the content-free nature of the probing samples, we propose a self-guided training strategy to carry out distribution-level optimization that aims to make the class scores for probing samples uniform, which enables us to take advantage of the probing samples to fine-tune and select only the proposed CA, respectively, while keeping the PLM encoder frozen. Therefore, our model is lightweight (e.g. 6.3M tunable adapter parameters for SST2 [28] task) as fine-tuning only updates the parameters of CA and the overall model footprint is reduced since several tasks can share a common PLM encoder as backbone. To demonstrate the effectiveness of our study, we have conducted extensive experiments, where the results indicate that the calibration ability acquired by CA on the probing samples can be successfully transfered to reduce negative effects of the biases of GPTs on a downstream target task, and our approach can yield new state-of-the-art performance consistently in zero-shot settings. We will detail our methodology formally in the Section 3 and Section 4.

The main contributions of this work are summarized as follows:

- In this paper, we first construct probing samples with the randomly generated token sequences, which are simple but effective in detecting inputs for stimulating GPTs to show the biases; and we pursue an in-depth research on the plausibility of utilizing class scores for the probing samples to reflect and estimate the biases of GPTs on a downstream target task.

[1] The content-free sample shares the same prompt with SST5 dataset.

[2] As a target task, SST5 shares the same prompt with these content-free samples.

- We propose a lightweight model Calibration Adapter (CA) along with a self-guided strategy that carries out distribution-level optimization, which enables us to take advantage of the probing samples to fine-tune and select only the proposed CA, respectively, while keeping the PLM encoder frozen.
- We conduct extensive validation experiments and analytical studies, and the results demonstrate the rationality and superiority of our research.

2 RELATED WORK

Prompt Learning. Originated from GPT3 [3] and LAMA [19, 20], a series of studies [7, 8, 13, 25, 30] stimulate the knowledge of PLMs that benefits downstream NLP tasks by leveraging natural-language prompts to formalize downstream tasks as language modeling problems, and show that such prompts induce better performances for PLMs on few-shot [15, 17, 26, 27] and zero-shot settings [23, 31, 34–36]. Inspired by this, a new paradigm known as prompt-based learning or prompt-learning has been introduced, which follows the “pre-train, prompt, and predict” process [18]. For instance, to identify the sentiment of a sentence: “A very funny movie.”, we can condition GPT3 on a prompt such as:

“Excellent acting and direction.” has a tone that is positive
“It’s just incredibly dull.” has a tone that is negative
“A very funny movie.” has a tone that is

where the first two lines are two training examples and the last line is the test example (there are no training examples in the zero-shot setting.). The model makes predictions based on which one is more likely to be the subsequent token, “positive” or “negative”.

Calibrating Biases of PLMs. Calibration [9, 18] refers to the ability of a model to make good probabilistic predictions. Contextual calibration (CC) [39] has revealed that GPTs are biased towards certain target classes in zero-shot settings because of the prompt and the model’s intrinsic biases, and proposes to utilize content-free samples to train and obtain a calibration model; then CC transfers the learned calibration model to calibrate three forms of bias: majority label bias, recency bias, and common token bias. However, as the detailed analysis in Section 1, CC can not consumes the content-free data effectively and reasonably, which restricts CC from reaching its full potential. Holtzman et al. [12] argues that all surface forms of a same object will compete for finite probability mass, and a rare surface form will be assigned a much lower probability than a common one. Therefore PMI$_{\mathrm{DC}}$ [12] is proposed to compensate for common token bias by directly factoring out the probability of each answer (such as each class of a target task). Due to the property of causal language models (e.g. GPTs) and the rarity of many possible answers, the unconditional probability of such answers is poorly calibrated for the purposes of a given task; consequently, PMI$_{\mathrm{DC}}$ uses a domain premise string to estimate the unconditional probability of an answer in a given domain. Nonetheless, as will be shown in Fig. 6, PMI$_{\mathrm{DC}}$ falls short of making the class scores for content-free samples uniform. We argue that the domain premise string employed by PMI$_{\mathrm{DC}}$ is selected manually, which introduces large sampling errors and causes the inferior calibration performance of PMI$_{\mathrm{DC}}$. ALC [16] proposes to factor out the probability of each answer proportional to the similarity between a premise and a hand-picked neutral context, and thus calibrate context-independent biases; however, similar to PMI$_{\mathrm{DC}}$, the performance of ALC is also constrained by the manually-selected neutral context. In this paper, we focus on reducing negative effects of the biases of GPTs and boosting the zero-shot performance by calibrating output probabilities of GPTs in prompt-learning.

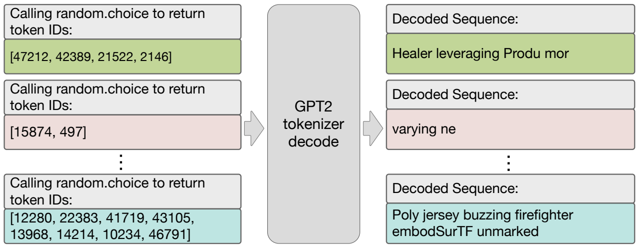

Calling random.choice to return token IDs:
[47212, 42389, 21522, 2146]
Calling random.choice to return token IDs:
[15874, 497]
Calling random.choice to return token IDs:
[12280, 22383, 41719, 43105, 13968, 14214, 10234, 46791]
GPT2 tokenizer decode
Decoded Sequence:
Healer leveraging Produ mor
Decoded Sequence:
varying ne
Decoded Sequence:
Poly jersey buzzing firefighter embodSurTF unmarked

Figure 2: Construction procedure of probing samples.

Transferability of Calibration Ability. In deep learning, transferability is corresponding to the ability of deep neural networks to extract transferable knowledge from some source tasks and then adapt the gained knowledge to improve learning in related target tasks [1, 14, 37, 40, 41]. In this paper, we fine-tune and select our model calibration adapter (CA) by exploiting the constructed probing samples (the source domain), and enable CA to acquire the calibration ability; then we transfer CA to reduce negative effects of biases of PLMs and achieve better zero-shot performance on downstream target tasks (the target domains).

3 PROBING SAMPLE

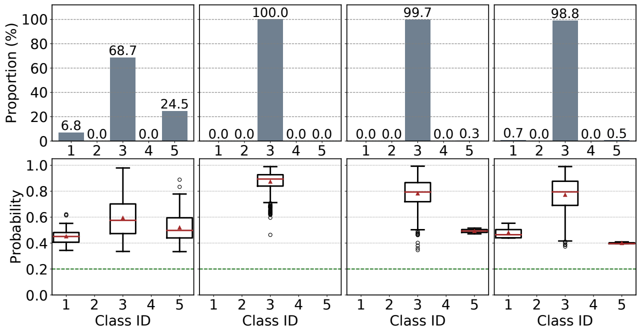
3.1 Constructing Probing Samples

Probing samples are designed to stimulate GPTs to show the biases in zero-shot settings. In this paper, we first construct probing samples with randomly generated token sequences instead of some manually selected special tokens. Specifically, to generate a probing sample, we can call the function *numpy.random.choice* [3] to randomly sample a certain number of token IDs from a given integer range, such as the range $[0, vocabulary\_size)$ [4], and then we decode the token ID sequence into a token sequence; the decoded token sequence is taken as a probing sample. In this way, we can easily obtain a large number of probing samples at low cost, and they are naturally content-free. The construction procedure and some generated probing samples are depicted in Fig. 2.

3.2 Plausibility of the Probing Samples

In this subsection, we pursue an in-depth research on the plausibility of utilizing class scores for the generated probing samples to reflect and estimate the biases of GPTs on a downstream target task. Ideally, given the content-free nature of probing samples, GPTs should score each probing sample equally over the target class set. For comparison, we report the actual prediction made by GPT2 for some probing samples in Fig. 3. In this case, we set the lengths of randomly sampled token ID sequences to 1, 2, 4, 6, 8 and 10

[3] numpy.org/doc/stable/reference/random/generated/numpy.random.choice.html
[4] The value of $vocabulary\_size$ depends on the language model employed, e.g., $vocabulary\_size$ of GPT2 is 50257.



(a) GPT2 (125M) (b) GPT2 (350M) (c) GPT2 (760M) (d) GPT2 (1.6B)

Figure 3: For each subplot, Top: illustration of proportions of probing samples that are classified into each class of SST5 dataset by GPT2 in zero-shot settings; Bottom: illustration of the distributions of the corresponding class probabilities of the probing samples that are classified into each class of SST5 dataset by GPT2 in zero-shot settings. Class ID 1, 2, 3, 4 and 5 represent the class Very Negative, Somewhat Negative, Neutral, Somewhat Positive and Very Positive respectively.

respectively, and for each length of token ID sequence, we generate 100 samples. All 600 randomly generated token ID sequences are decoded into the probing samples used here. Fig. 3 shows that, when feeding in probing samples [5], the biases of GPT2 of different size basically coincide with the biases shown in Fig. 1 respectively. For example, as shown in Fig. 3(a) (top), there are no probing samples that are classified into the class Somewhat Negative (ID: 2) or Somewhat Positive (ID: 4) by GPT2 (125M), and there are 9.8%, 78.0% and 12.2% probing samples that are classified into the classes Very Negative (ID: 1), Neutral (ID: 3) and Very Positive (ID: 5) by GPT2 (125M) respectively; meanwhile, as shown in Fig. 3(a) (bottom), GPT2 (125M) presents high certainties (0.2 in an ideal non-biased situation) for its predictions. In other words, by feeding in probing samples, GPT2 (125M) shows the biases towards the classes: Very Negative, Neutral and Very Positive; which are consistent with the biases of GPT2 (125M) shown in Fig. 1(a). More illustrations about GPT2 of other sizes are shown in the corresponding subplots of Fig. 1 and Fig. 3 respectively. Therefore, we argue that it is reasonable to take advantage of the class scores for probing samples to reflect and estimate the biases of GPTs on a downstream target task.

4 CALIBRATION ADAPTER

In order to effectively utilize the probing samples and thus reduce negative effects of the biases of GPTs, we propose a novel model Calibration Adapter (CA) along with a self-guided training strategy. CA consists of multiple channels corresponding to the target classes respectively. As shown in Fig. 4, each channel of CA is designed to estimate the bias towards corresponding class when feeding in an input, and then the actual class scores for the input are calibrated by being subtracted from the corresponding bias estimations respectively. We will present a single channel of CA in Section 4.2, which is simply extended according to the number of classes of the target task. The self-guided training strategy and the transfer of CA will be described in Section 4.3 and Section 4.4 respectively.

4.1 Normalized Class Probability Generation

Given a text classification task and the prompt template T used for the task, we wrap a text sequence x, e.g., a probing sample (during training phase) or a test sample (during transfer phase), into the prompt template T to form a new text sequence $T(x)$, serving as the input to the employed language model. The class set of the text classification task is notated as $Y = \{y_1, y_2, ..., y_n\}$, where n stands for the number of the classes. When feeding in an input $T(x)$, language models (LMs) can compute a class probability distribution $\mathcal{P} = [p_1, p_2, ..., p_n]$ over the class set Y:

$$p_i = p(y_i|T(x)) \tag{1}$$

where p_i is associated with the class label y_i, $i \in [1, n]$; and LMs classify the sample x into the class with the highest probability:

$$\underset{i}{\operatorname{argmax}}\, p_i \tag{2}$$

As causal language models, GPTs decompose the class probability as:

$$\begin{aligned} p_i &= p(y_i|T(x)) \\ &= \prod_{j=1}^{t_i} p(y_i^j|T(x), y_i^1, ..., y_i^{j-1}) \end{aligned} \tag{3}$$

where y_i^j is the jth token of y_i and t_i is the number of tokens in y_i. Whereafter, the class probability distribution $\mathcal{P} = [p_1, p_2, ..., p_n]$ predicted by the employed language model is renormalized to one:

$$p_i = \frac{p_i}{\sum_j p_j} \tag{4}$$

the normalized class probability distribution is fed into CA as input, and does not require gradient during training phase.

4.2 Channel of CA

4.2.1 Bias Feature Generation.

As shown in Fig. 4, the normalized class probabilities $\mathcal{P}$ for the sample x are filled to the diagonals of a diagonal matrix, namely $\text{diag}(\mathcal{P})$; then CA projects the class probabilities into the bias feature space through a linear transformation:

$$[\vec{f_1}, \vec{f_2}, ..., \vec{f_n}] = \text{diag}(\mathcal{P})\mathbf{W}_1 + \vec{b}_1 \tag{5}$$

where $\mathbf{W}_1 \in \mathbb{R}^{n \times d_1}$ and $\vec{b}_1 \in \mathbb{R}^{d_1}$ are the trainable parameters of the linear transformation, d_1 is the potentially different transformation size; the bias feature $\vec{f_i}$ is associated with the class label y_i. In order to estimate the bias towards y_i, the bias feature $\vec{f_i}$ is fed as input into the channel of CA that corresponds to the class y_i.

4.2.2 Estimating the Biases.

A channel of CA consists of a fully connected layer and a linear regression with an activation (Act) [11] and a Dropout (Drop) [29] in between:

$$\begin{aligned} \vec{h}_i &= \text{Drop}(\text{Act}(\vec{f_i}\mathbf{W}_2^i + \vec{b}_2^i)) \\ s_i &= \vec{h}_i \vec{w}^i + \epsilon^i \end{aligned} \tag{6}$$

where $\mathbf{W}_2^i \in \mathbb{R}^{d_1 \times d_2}$ and $\vec{b}_2^i \in \mathbb{R}^{d_2}$ are the trainable parameters of the fully connected layer, which enables CA with a larger model capacity to ensure excellent transferability of model representation capabilities, and d_2 is the potentially different transformation size; $\vec{w}^i \in \mathbb{R}^{d_2}$ and $\epsilon^i \in \mathbb{R}$ are the trainable parameters of the linear regression, which obtains the bias estimation s_i corresponding to the class y_i.

[5] They share the same prompt template (shown in Table 1) with SST5 dataset.

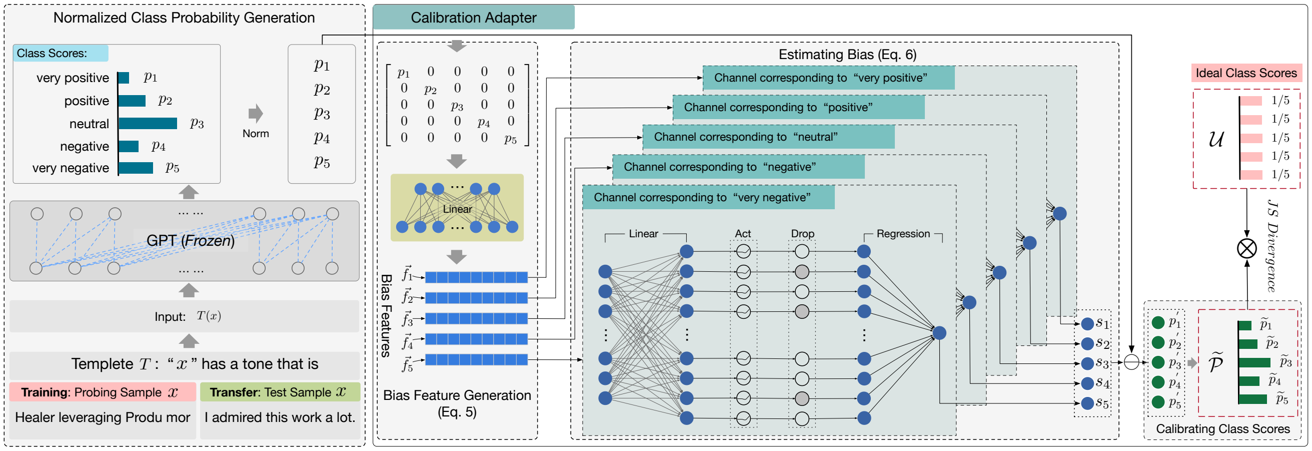

Normalized Class Probability Generation
Class Scores:
very positive
positive
neutral
negative
very negative
p_1 p_2 p_3 p_4 p_5
Norm
GPT (Frozen)
Input: $T(x)$
Templete T: " x " has a tone that is
Training: Probing Sample x
Transfer: Test Sample x
Healer leveraging Produ mor
I admired this work a lot.
Calibration Adapter
Linear
Bias Features
$\vec{f}_1$ $\vec{f}_2$ $\vec{f}_3$ $\vec{f}_4$ $\vec{f}_5$
Bias Feature Generation (Eq. 5)
Estimating Bias (Eq. 6)
Channel corresponding to "very positive"
Channel corresponding to "positive"
Channel corresponding to "neutral"
Channel corresponding to "negative"
Channel corresponding to "very negative"
Linear
Act
Drop
Regression
s_1 s_2 s_3 s_4 s_5
Ideal Class Scores
$\mathcal{U}$
1/5
JS Divergence
p'_1 p'_2 p'_3 p'_4 p'_5
$\widetilde{\mathcal{P}}$
$\widetilde{p}_1$ $\widetilde{p}_2$ $\widetilde{p}_3$ $\widetilde{p}_4$ $\widetilde{p}_5$
Calibrating Class Scores

Figure 4: Illustration of calibration adapter. In this case, SST5 serves as the downstream target task.

4.2.3 *Calibrating Class Scores.*

The class probabilities that are the input to CA are first subtracted by the corresponding bias estimations and then normalized to get the calibrated class scores:

$$p_i^{'} = p_i - s_i$$
$$\widetilde{p}_i = \frac{\exp(p_i^{'}/\tau)}{\sum_j \exp(p_j^{'}/\tau)} \tag{7}$$

where a small temperature controler τ and a Softmax function are employed to prevent overfitting; $\widetilde{p}_i$ is the calibrated class score associated with the class y_i.

4.3 Self-guided Training Strategy

Inspired by CC [39] and considering the content-free nature of the probing samples, each probing sample should be scored equally over the target class set Y in an ideal non-biased situation. As a consequence, we propose a self-guided training strategy to carry out a distribution-level optimization that aims to make the class scores for probing samples uniform. Since the impact of calibration during training phase should be measured at the distribution level, we choose Jensen-Shannon (JS) divergence as a metric to assess the similarity between the ideal class score distribution $\mathcal{U} = [\frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n}]$ and the calibrated class score distribution $\widetilde{\mathcal{P}} = [\widetilde{p}_1, \widetilde{p}_2, ..., \widetilde{p}_n]$ for a probing sample:

$$\ell = \mathcal{JS}(\mathcal{U}, \widetilde{\mathcal{P}}) \tag{8}$$

where the length of $\mathcal{U}$ is n, and $\mathcal{U}$ does not require gradient. The final training objective $\mathcal{L}$ is as follow (K means the number of probing samples in a batch):

$$\mathcal{L} = \sum^{K} \ell \tag{9}$$

The smaller the loss on the development set of probing samples, the better the performance of CA is considered to be.

The proposed self-guided training strategy enables us to take advantage of the probing samples to fine-tune and select only the proposed model CA, respectively, while keeping the PLM encoder frozen. Therefore, since without further pre-training or fine-tuning a PLM, and several tasks can share a common PLM encoder as backbone, our research provides a lightweight solution to reduce negative effects of the biases of PLMs.

4.4 Transfer

After being fine-tuned and selected on the training set and the development set of the probing samples (as the source domain) respectively, CA is directly transfered to calibrate the output probabilities of GPTs on a downstream target task (as the target domain) without further fine-tuning. CA classifies a test sample into the class with the highest calibrated class score:

$$\underset{i}{\text{argmax}}\, \widetilde{p}_i \tag{10}$$

5 EXPERIMENTS

5.1 Probing Samples for Training

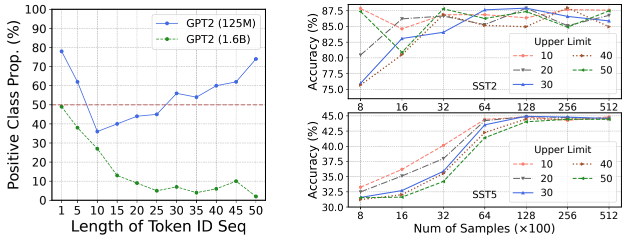
Reconsidering the construction of probing samples described in Section 3, the remaining questions are: (1) What should be the sampling probability associated with each token id? (2) What should be the length of a token ID sequence? (3) How many probing samples should be constructed?

5.1.1 *What should be the sampling probability associated with each token id?*

In this paper, we focus on improving the zero-shot performance of downstream target tasks; wherein neither do we know the data distributions of the downstream target tasks, nor can we make any assumptions about the data distributions of the downstream target tasks. Therefore, we simply sample a token ID according to a uniform distribution over all token IDs.

5.1.2 *What should be the length of a token ID sequence?*

To better understand the effect of probing samples decoded from token ID sequences of different lengths in stimulating the biases of GPTs, we conduct a pilot study with the prompt used by SST2 dataset [28]. In this case, we set the lengths of token ID sequences to 1, 5, 10, 15, 20, 25, 30, 35, 40, 45 and 50 respectively, and with each length, we randomly generate 100 token ID sequences; each randomly generated token ID sequence is decoded into a token sequence, where the decoded token sequences are regarded as probing samples used here. The probing samples share the same prompt template (shown in Table 1) with SST2 dataset. Fig. 5(a) shows the results, and we can summarize the following conclusions: (1) When the length of



(a) Illustration of proportions of probing samples decoded from token ID sequences of different lengths and classified into class Positive of SST2 dataset [28] by GPT2 (125M) and GPT2 (1.6B) in zero-shot settings.

(b) The effects of different numbers of randomly generated token ID sequences with different upper limits of length on the performance of calibration adapter (CA). GPT2 (1.6B) is used for zero-shot inference here.

Figure 5: Preliminary study of probing samples for training.

the token ID sequences is relatively small, such as less than 30, the biases stimulated by probing samples decoded from token ID sequences of different lengths are quite different. Specifically, as shown in Fig. 5(a), when the length of the token ID sequences is fixed to 1, there are 78% probing samples that are classified into the classes Positive by GPT2 (125M), indicating that GPT2 (125M) shows the bias towards the class Positive; however, when the length of the token ID sequences is fixed to 10, there are 36% probing samples that are classified into the classes Positive by GPT2 (125M), indicating that GPT2 (125M) shows the bias towards the class Negative. (2) When the length of the token ID sequences exceeds 30, the biases stimulated by the probing samples decoded from token ID sequences do not change significantly anymore. Therefore, we argue that the randomly generated token ID sequences should not be fixed to have the same length. We propose to set the length of a randomly generated token ID sequence to a random value that does not exceed a given upper limit, and on account of zero-shot settings, we sample an integer according to a uniform distribution over all positive integers up to the given upper limit as the length of a token ID sequence. We will discuss the value of the upper limit next.

5.1.3 How many probing samples should be constructed? In order to determine the upper limit of the lengths of randomly generated token ID sequences, and the number of probing samples that should be constructed, we conduct preliminary experiments on the test sets of SST2 dataset and SST5 dataset respectively. In this case, we set the length of a randomly generated token ID sequence to a random value that does not exceed a given upper limit u, which is one of the following values $\{10, 20, 30, 40, 50\}$. With a upper limit u, we randomly generate m and one-tenth of m token ID sequences, which are decoded to constitute the training set and development set of probing samples respectively; m is one of the following values $\{800, 1600, 3200, 6400, 12800, 25600, 51200\}$. In zero-shot settings, for the samples in the test set of a target task (SST2 or SST5), the output probabilities of GPT2 (1.6B) over the class set of the target task are calibrated by our model CA, which is fine-tuned and selected on the training set and development set of the above generated probing samples respectively; and the probing samples share same prompt template (shown in Table 1) with the target task. The calibration performance of CA is reported in Fig. 5(b), and we have the following conclusions: (1) Overall, increasing the number of probing samples can significantly improve the performance of calibration adapter (CA). When the number of probing samples exceeds 12800, the performance gain of CA is no longer significant, and the performance of CA tends to stabilize. (2) CA can achieve the best performance when the upper limit of the lengths of randomly generated token ID sequences is set to 30, which can be better understood through the observations described in the previous subsection, namely that the biases stimulated by the probing samples decoded from token ID sequences whose lengths exceed 30 do not change significantly anymore. Based on the above analysis, for all of the following experiments (unless specified otherwise), we empirically set the upper limit of the lengths of randomly generated token ID sequences to 30, and set the number of probing samples in the training set and development set to 12800 and 1280 respectively.

5.2 Experiment Settings

In this paper, we adopt following pre-trained language models (PLMs): GPT2 [22] and GPT3 [3]. We construct training and development sets of probing samples for each task according to the discussions in Subsection 5.1. GeLU [11] is employed as the activation function, and the dropout rate 0.2 is used to randomly zero some of the elements of the input tensor during training phase. We generally set the value of the temperature controller τ to 0.1 or 0.01 depending on the target task. The potentially transformation sizes d_1 and d_2 are set to 1024 and 3072 respectively, and the trainable parameters of each channel of CA are randomly initialized.

5.3 Evaluation Datasets

We conduct a systematic study across 7 popular English datasets: the 4-way AGNews [38] is a topic classification task, SST-2 [28] and SST-5 [28] are various granularities of sentiment classification tasks, the 6-way TREC [32] is a question classification task, the 3-way CB [6] and binary RTE [5] from SuperGLUE [33] are textual entailment tasks, and BoolQ (BQ) [4] poses yes/no (i.e. Boolean) questions based on a multi-sentence passage. Table 1 shows the prompt templates used for each dataset.

5.4 Baselines

LM. LM represents the performance of the employed PLMs without being calibrated, namely as defined in Equation 2.

Avg. Following PMI_{DC} [12], the average strategy (Avg) extends from Equation 3 and is defined as: $\underset{i}{\text{argmax}}\, \frac{1}{t_i} p(y_i|T(x))$, where t_i is the number of tokens in the class y_i.

Unc and PMI_{DC}. PMI_{DC} [12] proposes to compensate for common token bias by directly factoring out the probability of each class:

$$\underset{i}{\text{argmax}}(\log p(y_i|T(x)) - \log p(y_i|x_{domain})) \tag{11}$$

where x_{domain} is a domain premise chosen for a specific task. As shown in Table 1, we employ the same domain premise for each task as in PMI_{DC}. Unc [12] ignores the premise x completely, and only uses a domain premise: $\underset{i}{\text{argmax}}\, p(y_i|x_{domain})$

Table 1: The templates used for each task, along with an example instance. For each dataset, we employ the same prompts as in PMI$_{\text{DC}}$ [12]. The domain premises are marked in $[\cdot]_{\text{D}}$, and parts specific to each domain premise are underlined. The correct candidate answers for each instance are marked in $[\cdot]_{\text{A}}$.

Task Type	Dataset	Templete
Text Classification	AGNews	title: Touchy Times at Midas\n summary: The auto maintenance company has a simple business but a complicated prognosis.[\n topic:]$_{\text{D}}$[business]$_{\text{A}}$
	SST2	"A smart , sweet and playful romantic comedy ."[The quote has a tone that is]$_{\text{D}}$[positive]$_{\text{A}}$
	SST5	"Daring , mesmerizing and exceedingly hard to forget ."[The quote has a tone that is]$_{\text{D}}$[very positive]$_{\text{A}}$
	TREC	What is the oldest city in the United States ?[The answer to this question will be]$_{\text{D}}$[a location.]$_{\text{A}}$
Entailment	RTE	The girl was found in Drummondville.\n question: Drummondville contains the girl.[true or false?\n answer:]$_{\text{D}}$[false]$_{\text{A}}$
	CB	question: Given that "A: Your turn. B: Okay. Uh, I don't think they should abolish it." Is "they should abolish it" true, false, or neither?\n[the answer is:]$_{\text{D}}$[false]$_{\text{A}}$
Boolean QA	BoolQ	title: The Resident (TV series)\n question: Is the tv show the resident over for the season?[\n answer:]$_{\text{D}}$[yes]$_{\text{A}}$

Table 2: Performance comparison when using GPT2 and GPT3 for zero-shot inference. (Acc: %).

PLMs	Methods	AGNews	SST-2	SST-5	CB	TREC	RTE	BQ
GPT2 (125M)	Unc	25.0	49.9	18.1	08.9	22.6	52.7	**62.2**
	LM	57.4	63.6	27.4	48.2	23.0	51.6	58.8
	Avg	57.4	63.6	24.4	48.2	14.4	51.6	58.8
	PMI$_{\text{DC}}$	63.0	67.1	30.0	50.0	36.4	49.8	51.1
	ALC	60.6	66.3	28.8	48.2	38.4	50.9	56.1
	CC	55.4	63.4	30.0	10.7	**46.0**	48.7	38.0
	CA	**64.6**	**78.4**	**39.2**	**51.8**	42.0	**54.2**	**62.2**
GPT2 (350M)	Unc	25.0	49.9	17.6	08.9	22.6	47.3	**62.2**
	LM	64.3	80.2	18.5	50.0	28.8	53.1	60.8
	Avg	64.3	80.2	27.2	50.0	12.2	53.1	60.8
	PMI$_{\text{DC}}$	64.4	86.2	39.3	50.0	21.6	54.9	49.7
	ALC	64.4	84.9	27.9	50.0	32.8	54.2	57.6
	CC	64.3	84.0	41.3	14.3	43.6	49.8	50.5
	CA	**69.6**	**87.3**	**42.1**	**57.1**	**50.8**	**56.7**	**62.2**
GPT2 (760M)	Unc	25.0	49.9	17.6	08.9	22.6	47.3	**62.2**
	LM	60.7	77.0	20.3	48.2	22.8	53.1	58.0
	Avg	60.7	77.0	26.7	48.2	22.6	53.1	58.0
	PMI$_{\text{DC}}$	64.1	85.6	22.0	50.0	**44.0**	54.2	46.7
	ALC	63.0	84.2	21.3	50.0	37.2	**55.6**	52.6
	CC	55.5	83.1	42.3	08.9	40.6	53.8	55.2
	CA	**64.6**	**86.5**	**43.0**	**53.6**	43.4	54.9	**62.2**
GPT2 (1.6B)	Unc	25.0	49.9	17.6	08.9	22.6	47.3	**62.2**
	LM	64.8	84.0	30.4	50.0	22.8	47.7	56.3
	Avg	64.8	84.0	29.1	50.0	24.0	47.7	56.3
	PMI$_{\text{DC}}$	65.4	87.5	40.8	50.0	32.8	53.4	49.5
	ALC	64.9	86.7	35.7	50.0	46.8	50.5	53.6
	CC	60.0	82.0	43.2	17.9	37.3	48.5	49.8
	CA	**67.6**	**87.9**	**44.9**	**58.9**	**54.2**	**55.6**	**62.2**
GPT3 (2.7B)	Unc	25.0	49.9	18.1	08.9	13.0	47.3	**62.2**
	LM	69.0	53.7	20.0	51.8	29.4	48.7	58.5
	Avg	69.0	53.8	20.4	51.8	19.2	48.7	58.5
	PMI$_{\text{DC}}$	67.9	72.3	23.5	57.1	57.2	51.6	53.5
	CC	63.2	71.4	34.7	50.0	38.8	49.5	56.9
	CA	**69.5**	**79.1**	**35.8**	**58.9**	**59.0**	**54.5**	**62.2**
GPT3 (175B)	Unc	25.0	49.9	17.6	08.9	22.6	47.3	37.8
	LM	75.4	63.6	27.0	48.2	47.2	56.0	62.5
	Avg	75.4	63.6	27.3	48.2	25.4	56.0	62.5
	PMI$_{\text{DC}}$	74.7	71.4	29.6	50.0	58.4	64.3	**64.0**
	CC	73.9	**75.8**	31.9	48.2	57.4	57.8	61.0
	CA	**76.2**	71.7	**36.1**	**73.2**	**59.4**	**66.1**	**64.0**

ALC. ALC [16] factors out the probability of each answer proportional to the similarity between a premise x and a hand-picked neutral context. As discussed in ALC, the domain premise x_{domain} defined in PMI$_{\text{DC}}$ can be employed as a neutral context used here due to its neutral nature. ALC can be defined as:

$$\underset{i}{\operatorname{argmax}}(\log p(y_i|T(x)) - g(T(x), x_{domain}) \log p(y_i|x_{domain})) \quad (12)$$

where $g(T(x), x_{domain})$ is the similarity estimation. Following ALC, we consider Total Variation Distance to estimate the similarity:

$$g(T(x), x_{domain}) = 1 - 0.5 \times \|p_v(T(x)) - p_v(x_{domain})\|_1 \quad (13)$$

p_v indicates the probability vector output by the employed PLM across the vocabulary for the first token given the corresponding context. Limited by the API provided by OpenAI [6], and following the paper [16], ALC is only used to calibrate the outputs of GPT2.

CC. CC [39] computes an affine transform for each content-free input to make the class scores uniform respectively, and then averages the parameters of these learned affine transforms to attain a final affine transform, which is transfered to correct the output probabilities of GPTs on a downstream target task. Following CC, three content-free samples are employed here, namely N/A, [MASK] and the empty string.

5.5 Key Results and Analyses

Zero-shot results for GPT2 and GPT3 are reported in Table 2. There are several observations drawn from the results.

First, the overall comparison indicate that our model CA can consistently achieve SOTA performance (38 out of 42) while remaining lightweight (e.g. 6.3M and 15.8M tunable adapter parameters for SST2 task and SST5 task respectively); the calibration ability acquired by CA on the probing samples can be successfully transfered to calibrate the output probabilities of GPTs on a downstream target task, which contributes to reducing negative effects of the biases of GPTs and boosting zero-shot performance.

Second, our model CA adapts smoothly to GPTs of different sizes, where some improvement is pretty significant; for instance, when using GPT3 (175B) for zero-shot inference, CA improves the accuracy on the CB dataset from 50.0% to 73.2%. Even when the baseline models perform well, the improvement is still decent.

Third, let's focus on the worst-case performance of each prime model. Except for CA, calibration performed by all other models can sometimes cause a negative impact on the zero-shot performance of GPTs. For instance, CC causes the accuracy on CB to drop from 50% to 17.9% when using GPT2 (1.6B) for inference, and PMI$_{\text{DC}}$ and ALC causes the accuracy on BQ to drop from 60.8% to 49.7% and 57.6% respectively when using GPT2 (350M) for inference. For GPTs of various sizes, not only does CA cause no negative impacts, but its

[6] beta.openai.com/docs/api-reference/completions/create

Table 3: The mean and standard deviation for 5 randomly sampled sets of 2 examples used for few-shot inference. SST2 dataset is employed here. Avg is excluded, as it is equivalent to LM due to using single-token answers. (Acc: %).

PLMs	Unc	LM	PMI_{DC}	ALC	CC	CA
GPT2 (125M)	$49.9_{\pm 0.0}$	$60.7_{\pm 9.9}$	$61.8_{\pm 10.4}$	$61.5_{\pm 10.3}$	$61.1_{\pm 5.0}$	$\mathbf{75.4}_{\pm 3.6}$
GPT2 (350M)	$49.9_{\pm 0.0}$	$74.1_{\pm 13.6}$	$77.8_{\pm 12.3}$	$77.1_{\pm 12.8}$	$82.1_{\pm 4.6}$	$\mathbf{88.3}_{\pm 1.5}$
GPT2 (760M)	$49.9_{\pm 0.0}$	$74.8_{\pm 12.0}$	$73.9_{\pm 13.1}$	$74.6_{\pm 13.2}$	$77.4_{\pm 8.9}$	$\mathbf{88.2}_{\pm 1.4}$
GPT2 (1.6B)	$49.9_{\pm 0.0}$	$71.6_{\pm 9.4}$	$70.5_{\pm 11.0}$	$70.9_{\pm 10.2}$	$68.2_{\pm 8.5}$	$\mathbf{82.6}_{\pm 3.2}$
GPT3 (2.7B)	$49.9_{\pm 0.0}$	$58.1_{\pm 7.9}$	$69.1_{\pm 13.0}$	-	$78.4_{\pm 2.2}$	$\mathbf{81.6}_{\pm 3.0}$
GPT3 (175B)	$49.9_{\pm 0.0}$	$79.1_{\pm 10.0}$	$64.9_{\pm 10.1}$	-	$78.1_{\pm 13.6}$	$\mathbf{86.7}_{\pm 5.4}$

calibration performance is extremely competitive even when suboptimal. We argue that the special content-free samples employed by CC and the neutral domain premises employed by PMI_{DC} and ALC are both selected manually, which introduces large sampling errors, limiting the transferability of CC, PMI_{DC} and ALC respectively. Our model CA learns calibration knowledge from a large number of probing samples, and the delicate model design along with the proposed self-guided training strategy makes CA highly transferable and consistently superior.

To summarize, the results in this experiment not only confirm that it is reasonable to take advantage of the class scores for probing samples to reflect and estimate the biases of GPTs on a downstream target task, but also fully illustrate that our model CA is able to consume the probing data effectively and achieve superior calibration ability and transferability.

5.6 Few-shot

While this paper is focus on zero-shot settings, CA is just as applicable to few-shot scenarios. Table 3 reports the mean and standard deviation of 5 randomly sampled sets of 2 examples that are used by each model for SST2 task. The overall trend clearly favors CA.

5.7 Generalization

In this subsection, we investigate the generalization ability of different calibration models on unseen probing samples. In this case, according to the construction strategy discussed in Subsection 5.1, we construct 1280 probing samples, which constitute the test set used here. The classes of SST5 dataset are taken as the target classes, and SST5 shares the same prompt template with the probing samples used here. For each probing sample, Jensen-Shannon (JS) divergence serves as a metric to assess the similarity between the ideal class score distribution and the actual class score distribution (with or without being calibrated). The results reported in Fig. 6 show that CA has better generalization ability on probing samples, which reflects that CA can consume the probing data more effectively.

5.8 Ablation Study

We finally conduct two ablations on CA. Firstly, we further analyze how effective CA is at utilizing probing data. To do so, we compare the accuracy of CA and CC when consuming the same number of content-free probing samples. We evaluate this ablation on SST2 and SST5, and use GPT2 (1.6B) for zero-shot inference. We find that CA can effectively acquire calibration knowledge as the number of consumed probing samples increases, but CC cannot (Fig. 7(a)).

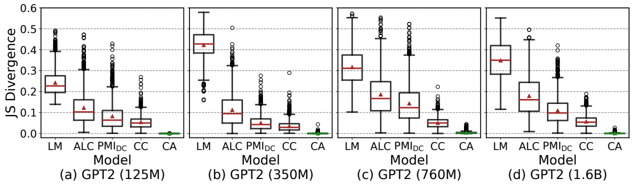
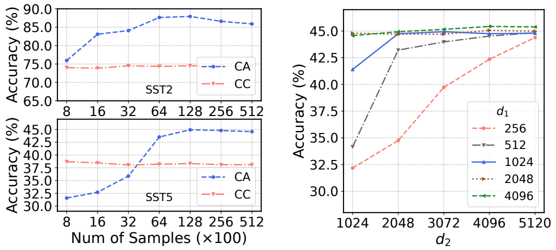


Figure 6: Illustration of the distributions of similarities between the ideal class score distributions and corresponding actual class score distributions predicted by GPT2 and calibrated by different models on a test set of probing samples.



(a) Utilizing probing samples. (b) Capacity of CA.

Figure 7: Illustration for ablation studies.

We also study how the capacity of CA affects accuracy. In Fig. 7(b), we use GPT2 (1.6B) for zero-shot inference, and show the accuracy for SST5 for different capacity of CA, which depends on d_1 and d_2 in Equation 6. Overall, increasing the capacity of CA can effectively improve the performance of CA; when the capacity of CA exceeds a certain limit (such as d_1 exceeds 1024 and d_2 exceeds 3072), the performance gain is no longer significant. Therefore, we empirically set d_1 and d_2 to 1024 and 3072 respectively.

6 CONCLUSION

In this paper, we first construct probing samples with the randomly generated token sequences, and we pursue an in-depth research on the plausibility of utilizing class scores for the probing samples to reflect and estimate the biases of GPTs on a downstream target task. We propose a lightweight model Calibration Adapter (CA) along with a self-guided training strategy that carries out distribution-level optimization, which enables us to take advantage of the probing samples to fine-tune and select only the proposed CA, respectively, while keeping the PLM encoder frozen. Extensive experiments and analytical studies are conducted to demonstrate the rationality and superiority of our research.

ACKNOWLEDGMENT

This work is supported in part by Chinese National Key Laboratory of Science and Technology on Information System Security, the National Natural Science Foundation of China Projects No. U1936213, and the NSF under grants III-1763325, III-1909323, III-2106758, and SaTC-1930941.

REFERENCES

[1] Yoshua Bengio. 2012. Deep Learning of Representations for Unsupervised and Transfer Learning. *ICML workshop*.

[2] Gregor Betz, Kyle Richardson, and Christian Voigt. 2021. Thinking Aloud: Dynamic Context Generation Improves Zero-Shot Reasoning Performance of GPT-2. *ArXiv* abs/2103.13033 (2021).

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

[4] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

[5] Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL Recognising Textual Entailment Challenge. *MLCW*.

[6] Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The CommitmentBank: Investigating projection in naturally occurring discourse. *Sinn und Bedeutung*.

[7] Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Haitao Zheng, Zhiyuan Liu, Juan-Zi Li, and Hong-Gee Kim. 2021. Prompt-Learning for Fine-Grained Entity Typing. *ArXiv* abs/2108.10604 (2021).

[8] Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making Pre-trained Language Models Better Few-shot Learners. *ACL-IJCNLP*.

[9] Leon J. Gleser. 1996. Measurement, Regression, and Calibration. *Technometrics* (1996).

[10] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On Calibration of Modern Neural Networks. *ICML* (2017).

[11] Dan Hendrycks and Kevin Gimpel. 2016. Bridging Nonlinearities and Stochastic Regularizers with Gaussian Error Linear Units. *ArXiv* (2016).

[12] Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. 2021. Surface Form Competition: Why the Highest Probability Answer Isn't Always Right. In *EMNLP*.

[13] Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Juan-Zi Li, and Maosong Sun. 2021. Knowledgeable Prompt-tuning: Incorporating Knowledge into Prompt Verbalizer for Text Classification. *ArXiv* abs/2108.02035 (2021).

[14] Junguang Jiang, Yang Shu, Jianmin Wang, and Mingsheng Long. 2022. Transferability in Deep Learning: A Survey. *CoRR* (2022).

[15] Katharina Kann, Kyunghyun Cho, and Samuel R. Bowman. 2019. Towards realistic practices in lowresource natural language processing: The development set. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing* (2019), 3342–3349.

[16] Sawan Kumar. 2022. Answer-level Calibration for Free-form Multiple Choice Question Answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Dublin, Ireland, 665–679.

[17] Sawan Kumar and Partha P. Talukdar. 2021. Reordering Examples Helps during Priming-based Few-Shot Learning. *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021* (2021), 4507–4518.

[18] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ArXiv* abs/2107.13586 (2021).

[19] Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How Context Affects Language Models' Factual Predictions. *Automated Knowledge Base Construction* (2020).

[20] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language Models as Knowledge Bases?. In *EMNLP-IJCNLP*.

[21] John Platt. 1999. Probabilistic Outputs for Support vector Machines and Comparisons to Regularized Likelihood Methods. *Advances in Large Margin Classifiers*.

[22] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog* (2019).

[23] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-Shot Text-to-Image Generation. *ICML* abs/2102.12092 (2021).

[24] Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang A. Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M SAIFUL BARI, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Stella Rose Biderman, Leo Gao, T. G. Owe Bers, Thomas Wolf, and Alexander M. Rush. 2021. Multitask Prompted Training Enables Zero-Shot Task Generalization. *ArXiv* abs/2110.08207 (2021).

[25] Timo Schick and Hinrich Schütze. 2021. Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference. In *EACL*.

[26] Timo Schick and Hinrich Schütze. 2021. It's Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners. In *NAACL*.

[27] Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical Networks for Few-shot Learning. In *NIPS*. 4077–4087.

[28] Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*.

[29] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* (2014).

[30] Yi Sun, Yu Zheng, Chao Hao, and Hangping Qiu. 2021. NSP-BERT: A Prompt-based Zero-Shot Learner Through an Original Pre-training Task-Next Sentence Prediction. *ArXiv* abs/2109.03564 (2021).

[31] Nandan Thakur, Nils Reimers, Andreas Ruckl'e, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogenous Benchmark for Zero-shot Evaluation of Information Retrieval Models. *ArXiv* abs/2104.08663 (2021).

[32] Ellen M. Voorhees and Dawn M. Tice. 2000. Building a question answering test collection. *SIGIR*.

[33] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. *NeurIPS*.

[34] Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2021. Finetuned Language Models Are Zero-Shot Learners. *ArXiv* abs/2109.01652 (2021).

[35] Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. 2019. Zero-Shot Learning—A Comprehensive Evaluation of the Good, the Bad and the Ugly. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41 (2019), 2251–2265.

[36] Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking Zero-shot Text Classification: Datasets, Evaluation and Entailment Approach. *EMNLP-IJCNLP*.

[37] Shuxi Zeng, Murat Ali Bayir, Joel Pfeiffer, Denis Xavier Charles, and Emre Kıcıman. 2021. Causal Transfer Random Forest: Combining Logged Data and Randomized Experiments for Robust Prediction. *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (2021).

[38] Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level Convolutional Networks for Text Classification. *NeurIPS*.

[39] Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate Before Use: Improving Few-shot Performance of Language Models. In *Proceedings of the 38th International Conference on Machine Learning*, Vol. 139. 12697–12706.

[40] Yichao Zhou, Ying Sheng, Nguyen Ha Vo, Nick Edmonds, and Sandeep Tata. 2022. Learning Transferable Node Representations for Attribute Extraction from Web Documents. *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining* (2022).

[41] Yongchun Zhu, Zhenwei Tang, Yudan Liu, Fuzhen Zhuang, Ruobing Xie, Xu Zhang, Leyu Lin, and Qing He. 2022. Personalized Transfer of User Preferences for Cross-domain Recommendation. *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining* (2022).