Last to Learn Bias: Analyzing and Mitigating a Shortcut in Question Matching

Anonymous ACL submission

Abstract

Recent studies report that even if deep neural models make correct predictions, models may be relying on shortcut rather than understanding the semantics of the text. Previous studies indicate that the shortcut deriving from the biased data distribution in training set makes spurious correlations between features and labels. In this paper, we focus on analyzing and mitigating the biased data distribution in question matching by exploring the model behavior and performance. In particular, we define bias-word as the shortcut, and explore the following questions: (1) Will the bias affect the model? (2) How does the bias affect the model's decision? Our analysis reveals that bias-words make significantly higher contributions to model predictions than random words, and the models tend to assign labels that are highly correlated to the bias-words. To mitigate the effects of shortcut, we propose a simple approach that learns more no-bias-examples first and more bias-examples last. The experiments demonstrate the effectiveness of the proposed approach.

1 Introduction

The task of *question matching* (QM) aims at identifying if a question pair has the same meaning, which benefits many real-world applications, e.g. search engine, intelligent customer services and others. With the development of deep learning (Devlin et al., 2018; Liu et al., 2019; Sun et al., 2019), the pre-trained language models have achieved remarkable results on the task of question matching.

However, recent studies have demonstrated that these models strongly rely on some spurious correlations between features and labels (i.e., shortcut) instead of deep understanding of text for making predictions (Geirhos et al., 2020; Khani and Liang, 2021; Tu et al., 2020; Hendrycks et al., 2020; Wang and Culotta, 2020). The shortcut learning has been studied in various NLP tasks, such as machine reading comprehension (MRC) (Jia and Liang, 2017; Lai et al., 2021; Kaushik and Lipton, 2018; Sugawara et al., 2018, 2020), natural language inference (NLI) (Gururangan et al., 2018; McCoy et al., 2019; Poliak et al., 2018; Du et al., 2021; Kavumba et al., 2021) and question answering (QA) (Ye and Kovashka, 2021; Yu et al., 2020). Previous works mainly examine the shortcut by creating artificial adversarial examples (Jia and Liang, 2017; Sugawara et al., 2018; Niven and Kao, 2019; Kavumba et al., 2019; McCoy et al., 2019; Lai et al., 2021; Kavumba et al., 2021). However, it is not clear that if the studies and improvements on artificial adversarial examples can work well on the distributions from real-world applications (Morris et al., 2020; Bender and Koller, 2020).

To the best of our knowledge, very few studies systematically analyze the shortcut learning phenomena on question matching (QM) task so far. In this paper, we focus on analyzing and mitigating the shortcut in question matching. Instead of creating artificial adversarial examples, our key idea is exploring the biased data distribution to explain the shortcut learning behavior of the question matching models. Specifically we try to answer the following research questions:

- **RQ 1:** What is the bias in the training set of question matching?
- **RQ 2:** Will the bias affect the question matching model?
- **RQ 3:** How does the bias affect the model's decision?
- **RQ 4:** How to mitigate the model's reliance on bias?

In summary, we have the following major findings and contributions:

- We formally define bias-word as the shortcut in the training set of question matching, that is highly correlated to a specific label, and we observe that there is a large proportion of examples containing the bias-words (see Sec. 2).
- We observe that bias-examples are easier to be

learned than others, and bias-words make significantly higher contributions to model predictions than random words (see Sec. 3).
- We find that the models tend to assign labels that highly correlated to the bias-words (see Sec. 4).
- According to the above observations, we propose a simple approach to mitigate the shortcut in question matching, that learns more no-bias-examples first and more bias-examples last. The experiments show the effectiveness of our proposed approach (see Sec. 5).

The remaining of this paper is organized as follows. In Section 2, we answer **RQ 1** and formally define the bias-word and bias-example. In Section 3 and Section 4, we answer **RQ 2** and **RQ 3** respectively, and we conduct extensive analysis on the model behavior on bias-words and bias-examples. Section 5 tries to answer **RQ 4** and proposes a simple approach to mitigate the shortcut in question matching. We conclude our work in Section 6 and discuss the future work.

2 Preliminary

In this section, we firstly introduce the QM datasets on which we perform analysis, then we give definitions about bias-words and bias-examples. At last, we provide the settings of our experiments that used in our experiments.

2.1 Datasets

We conduct our study on three datasets, LCQMC, DuQM and OPPO [1], all of which are about QM task and collected from real-word applications. LCQMC (Liu et al., 2018) is a large-scale Chinese question matching corpus proposed by Harbin Institute of Technology in general domain BaiduZhidao[2]. DuQM [3] is a fine-grained controlled dataset which is aimed to evaluate the robustness of question matching models and generated based on queries in Baidu Search Engine [4]. OPPO is collected from OPPO XiaoBu Dialogue application and we can get it from CCF Big Data & Computing Intelligence Contest. Data statistics are in Tab. 1.

2.2 Definitions

Here we provide the definitions we will use in our analysis and experiments. If we denote W as all words in the data set, the set of examples with a specific word w_i can be formalized as $S(w_i)$, and frequency of w_i can be formalized as f_{w_i}, and

$$f_{w_i} = |S(w_i)| \tag{1}$$

We then define *bias-degree* to measure the degree of word w_i co-occur with category c_j (for QM task, $c_j \in (0, 1)$) and denote is as

$$d_{w_i}^{c_j} = \frac{|S(w_i, c_j)|}{|S(w_i)|} = \frac{|S(w_i, c_j)|}{f_{w_i}} \tag{2}$$

where $|S(w_i, c_j)|$ represents the number of examples with w_i and tagged with c_j.

Bias-word. A word highly correlated with a specific label in a data set.[5] To better discuss them, we define bias-word as the word w_i with $f_{w_i} \geq 3$ and $d_{w_i}^{c_j} \geq 0.8$. It is worth mentioning that the bias-words we analyze in this work are originated from LCQMC$_{train}$.

We further define *bias-word*$_0$ and *bias-word*$_1$ as the words highly correlated to category 0 and 1. As shown in Tab. 2, "简便" ("handy") occurs in 35 examples, 33 of which are with category 1, hence it is a bias-word$_1$. Tab. 3 shows that 27.24% (15864/58230) of words are bias-word, and there are more bias-word$_0$ than bias-word$_1$ in LCQMC$_{train}$.

Dataset	Word cnt. q1	Word cnt. q2	Total	Category #0	Category #1	Total
L$_{train}$	6.04	6.36	12.40	100,192	138,574	238,766
L$_{test}$	5.51	5.61	11.12	6,250	6,250	12,500
DuQM	4.66	4.80	9.46	7,318	2,803	10,121
OPPO	4.82	4.71	9.53	7.160	2.840	10,000

Table 1: Data statistics. L$_{train}$ denotes LCQMC training set, and L$_{test}$ denotes LCQMC test set.

	Word	Category #0	Category #1	Total	B-degree
B-word$_0$	漂浮 (float)	5	0	5	1.00
B-word$_1$	简便 (handy)	2	33	35	0.94

Table 2: Examples of bias-word$_0$, and bias-word$_1$. B-word$_0$ represents bias-word$_0$, and B-word$_1$ represent bias-word$_1$.

[1]The datasets can be downloaded from https://luge.ai.
[2]https://zhidao.baidu.com.
[3]https://github.com/baidu/DuReader/tree/master/DuQM.
[4]http://www.baidu.com.

[5]Word is the smallest independent lexical items with own objective or practical meaning. We use Lexical Analysis of Chinese (Jiao et al., 2018) (https://github.com/baidu/lac) for word segmentation in our work.

# Word	# B-word$_0$	# B-word$_1$	# B-word
58,230	11,143	4,721	15,864

Table 3: The statistics of bias-words in LCQMC$_{train}$.

Bias-example. An example with at least one bias-word. As shown in Tab. 4, 41.15% of examples in LCQMC$_{train}$ are bias-examples, which is 25.97%, 32.25% and 24.98% in LCQMC$_{test}$, DuQM and OPPO respectively. Since bias-words occur in almost half of the examples in LCQMC$_{train}$, it is meaningful to study their effects. On the other hand, we define the examples without bias-word as *no-bias-example*.

Dataset	# Examples	# B-exp	%B-exp
LCQMC$_{train}$	238,766	98,260	41.15%
LCQMC$_{test}$	12,500	3,246	25.97%
DuQM	10,121	3,264	32.25%
OPPO	10,000	2,498	24.98%

Table 4: The statistics of bias-examples in datasets. B-exp represents bias-example.

2.3 Experimental setup

Models. We evaluate three popular public available pre-trained models, BERT-base (Devlin et al., 2018)[6], ERNIE-1.0 (Sun et al., 2019)[7], RoBERTa-large (Liu et al., 2019)[8] in our work.

Metrics. Like most binary classification tasks, we use accuracy to evaluate the performance.

Training details. In the training stage, we encode question pairs with a $[SEP]$ and then pass the pooled output to a classifier. We use different learning rates and epochs for different pre-trained models. Specifically, for RoBERTa$_{large}$, the learning rate is 5e-6 and the number of epochs is 3. For BERT$_{base}$ and ERNIE$_{1.0}$, the learning rate is 2e-5, and we set the number of epochs as 2. The batch size is set as 64 and the maximal length of question pair is 64. The proportion of weight decay is 0.01. In addition, we use early stopping to select the best checkpoint. Each model is fine-tuned five times with different seed on LCQMC$_{train}$. We choose the model with the best performance on the LCQMC$_{dev}$ and report average results on LCQMC$_{test}$, DuQM and OPPO.

[6]https://github.com/google-research/bert.
[7]https://github.com/PaddlePaddle/ERNIE.
[8]https://github.com/ymcui/Chinese-BERT-wwm.

3 Will bias affect models?

The dataset statistics in Sec. 2 show that 41.15% of examples in LCQMC$_{train}$ involve bias-words. It is a reasonable assumption that the large proportion of bias-examples would affect the model behavior. To validate our hypothesis, we conduct a behavior analysis about the model's learning and deciding.

3.1 Bias and models' learning

To diagnose the model's behavior during training, we separate LCQMC$_{train}$ into two subsets, bias-examples and no-bias-examples, and re-organize the train examples in 3 orders:

- *bias-first*: firstly bias-examples, then no-bias-examples;
- *bias-last*: firstly no-bias-examples, then bias-examples;
- *random order*: sample the examples randomly.

We fine-tune three models (BERT, ERNIE and RoBERTa) in these 3 orders and plot the training loss curves in Fig. 1. The training loss curves of all three models present the same tendencies:

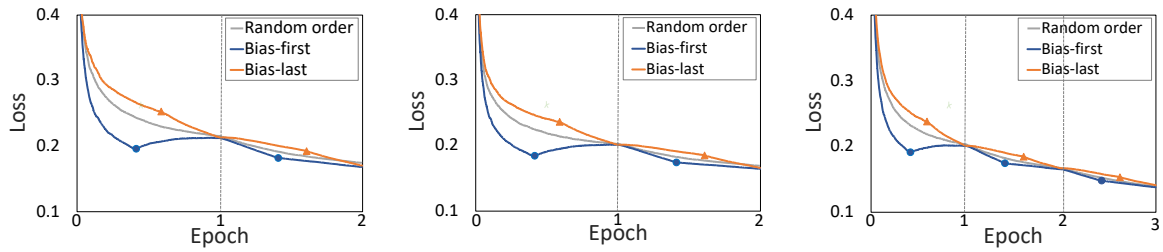
- If bias-first, the loss curve drops more rapidly than random order. After learning all the bias-examples, the loss curve rises slightly and then decreases.
- The tendency of bias-last is contrary: the loss drops more slowly than random order until all the no-bias-examples have been learned, and then the curve decreases faster.
- These two trends repeat for each epoch.

The above observations reflect that models behave different when they learn bias-examples and no-bias-examples: the loss curves of bias-examples drop more sharply than other examples, which indicates that the words highly correlated with specific labels are shortcuts and relatively easy for models to learn. Generally, we validate our hypothesis that the bias feature will affect the model's learning behavior with training loss analysis.

3.2 Bias and models' prediction

The training loss curves illustrate how bias influences model during training. In this part, we provide a quantitatively analysis about bias's impact on model's prediction. If bias is a easier feature for a model to learn, will bias-words make greater contributions when predicting?

The LIME method (Ribeiro et al., 2016) interprets model prediction based on locally approximating the model around a given prediction. In our



(a) Training loss curve of BERT. (b) Training loss curve of ERNIE. (c) Training loss curve of RoBERTa.

Figure 1: Training loss curves of three models on LCQMC$_{train}$, in which ● represents finishing learning bias-examples, and ▲ represents finishing learning no-bias-examples.

work, LIME method serves as a tool to measure the contribution of different words in one input to the final prediction.

To observe the contribution of bias-word in each test sample, we rank the words based on their contributions computing with LIME method. In Fig. 2, we illustrate the probabilities of bias-words with the highest, second, third, and fourth contribution in three test sets. For comparison, we select a same size of words randomly and also plot their probabilities in Fig. 2.

As we reported in Tab. 1, in these three test sets, each sample is composed of around 9 to 11 words. Fig. 2 shows that the random words have a probability of around 46% ranked among the highest 4 contribution words. Compared with random words, the bias-words have significantly higher probability to be ranked among the highest 4, which is about 80% in LCQMC$_{test}$ and DuQM, 68% in OPPO. Specifically, in about 40% of bias-examples of LCQMC$_{test}$, 37% of DuQM, and 25% of OPPO, the bias-word is the word with the highest contribution to final prediction, which is 2~3 times to random words (which is only 13%~15%).

In short, the bias-examples are easier for models to learn, and the bias-words make significantly higher contributions than random words when predicting, which implies that models tend to pay more attention to bias-words during predicting. With the analysis in this section, we can determine that bias is a shortcut and will affect the model behavior. It is therefore substantial to further analyze how it affects the models.

4 How does bias affect models' decision?

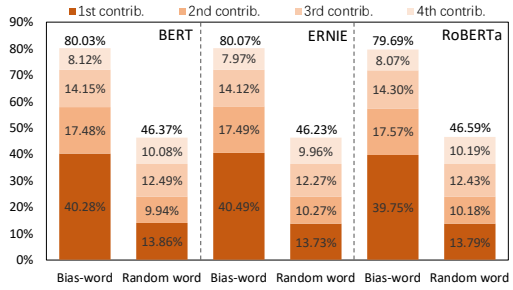
Previous works show that superficial cues exist in many data sets and are widely studied (Bolukbasi et al., 2016; May et al., 2019; Ravfogel et al., 2020; Webster et al., 2020; Kaneko and Bollegala, 2021). However, there are few quantitative analysis to discuss how these cues affect the model's decision. We have proved that bias-words tend to make more contributions to the final prediction than other words. In this section, we will focus on the examples where bias-words make the **greatest** contribution to the final prediction, in which the effect of bias-word would be more significant, to probe the relationship between the bias-word and the predicted category. A reasonable guess is that the models tend to assign the category highly relying on the distribution bias trick, i.e. a bias-word$_1$ with high contribution to the final decision will bring a prediction of category 1 and vice versa.

B-word$_0$	LCQMC$_{test}$		DuQM		OPPO	
	# S	# S_{focus}	# S	# S_{focus}	# S	S_{focus}
BERT		890		1296		799
ERNIE	1777	891	2375	1345	1991	828
RoBERTa		877		1353		793

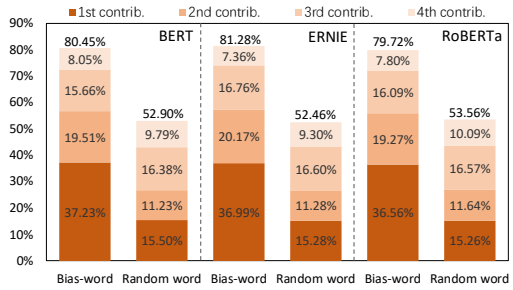
Table 5: Statistics of bias-example$_0$ (S) and focus-bias$_0$ examples (S_{focus}). Focus-bias$_0$ examples represent the examples where models focus on the bias-word$_0$.

4.1 Influence of bias-word on predicted labels

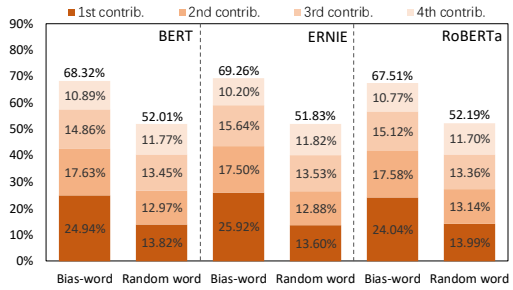
Although bias-words tend to contribute more, not all bias-words make great contribution during predicting. To explore the impact of bias-word on predicted label, an effective method is to observe the prediction result when bias-word contributes the most, in which the effect of bias-word would be more significant. For convenience, we define **focus-bias examples** as the examples in which bias-words make the greatest contribution, and we present the statistics of bias-examples and focus-bias examples in Tab. 5 and Tab. 6.



(a) Results on LCQMC$_{test}$.



(b) Results on DuQM.



(c) Results on OPPO.

Figure 2: Probability of bias-words with the 1st, 2nd, 3rd, 4th contribution on three test sets.

Tendency to predict c_j. We define T_{c_j} as the tendency of model to predict of category c_j

$$T_{c_j} = \frac{|S_{pred}(c_j)|}{|S_{true}(c_j)|} \qquad c_j \in (0, 1) \qquad (3)$$

where $|S_{true}(c_j)|$ and $|S_{pred}(c_j)|$ represent the number of test examples with true label c_j and predicted as c_j.

Results analysis. To evaluate the influence of bias-word on predicted label, we calculate the tendency of "normal" bias-examples and focus-bias examples and denote them as T_{c_j} and $T_{c_j}^{focus}$.

Fig. 3(a) to Fig. 3(c) demonstrates the influences of bias-word$_0$ on three test sets. In DuQM (Fig. 3(b)), it is obvious that T_0^{focus} is higher than T_0 by 5%~7% with all three models, which implies that when the bias-word$_0$ contributes the most, models have a high tendency to predict of 0. The same result is shown in LCQMC$_{test}$ (Fig. 3(a)). However, on OPPO, T_0 is slightly higher (0.01~0.02) compared with T_0^{focus}. We suppose that it is resulted by co-influencing of other shortcut and we provide an extensive experiment to discuss it in Sec. 4.2. Fig. 3(d) to Fig. 3(f) are about the influence of bias-word$_1$. As shown in Fig. 3(f), models tend to predict 1 when they concentrate on bias-word$_1$ on OPPO, T_1^{focus} is higher than T_1 by 16% averaged between three models). The comparisons on DuQM present different results on three datasets. On LCQMC$_{test}$, T_1^{focus} is higher than T_1 with BERT and RoBERTa.

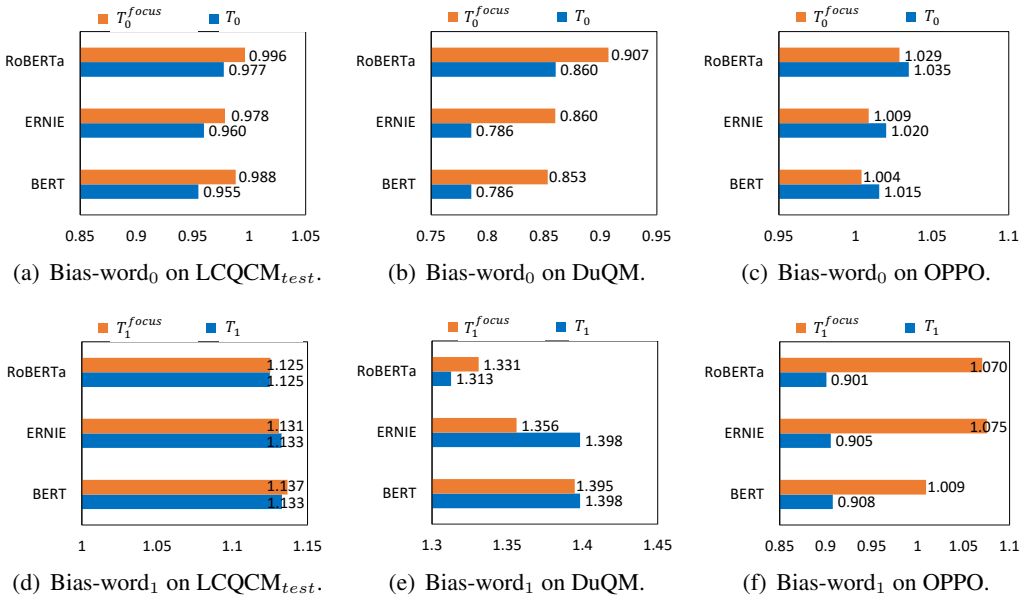
In short, when models pay more attention to bias-words, they tend to assign labels relying on the distribution bias they learn from training set. To explore why the tendency to 0 is not obvious on on OPPO (Fig. 3(c)), we will provide a further discussion about the influence of other shortcut.

B-word$_1$	LCQMC$_{test}$		DuQM		OPPO	
	# S	# S_{focus}	# S	# S_{focus}	# S	S_{focus}
BERT		984		557		265
ERNIE	1543	1009	1095	541	602	267
RoBERTa		993		514		255

Table 6: Statistics of bias-example$_1$ (S) and focus-bias$_1$ examples (S_{focus}). Focus-bias$_1$ examples represent the examples where models focus on the bias-word$_1$.

4.2 Word-overlap: another shortcut for QM models

In real-world scenarios, the mechanism of a model's decision is complicated. Different shortcuts may interact together to give the final prediction. In this work, we argue that QM models are also affected by **word overlap** shortcut. Word overlap is a shortcut which have been discussed in many MRC and NLI works (McCoy et al., 2019; Lai et al., 2021; Kaushik and Lipton, 2018). For QM task, the QM models tend to predict 0 if a sentence pair has low word overlap, i.e., there are few common words between them, and vice versa. As the result of OPPO shown in Tab. 7, even if models focus on bias-word$_0$, the tendency to 0 is not significant. We attribute the phenomenon to the word overlapping bias in the QM task. To eliminate the influence of word-overlapping, we design a experiment on the examples in which the question pairs with high overlapping. We use Levenshtein edit distance to measure the overlapping degree. The



(a) Bias-word$_0$ on LCQCM$_{test}$. (b) Bias-word$_0$ on DuQM. (c) Bias-word$_0$ on OPPO.

(d) Bias-word$_1$ on LCQCM$_{test}$. (e) Bias-word$_1$ on DuQM. (f) Bias-word$_1$ on OPPO.

Figure 3: Tendency to predict 0 of bias-word$_0$ and predict 1 of bias-word$_1$.

Model	**Dist.** $\leqslant 1$		**Dist.** $\leqslant 2$		**Dist.** $\leqslant 3$		**Dist.** $\leqslant 4$		**Dist.** $\leqslant 5$	
	T_0	T_0^{focus}	T_0	T_0^{focus}	T_0	T_0^{focus}	T_0	T_0^{focus}	T_0	T_0^{focus}
BERT	0.78	0.80	0.79	0.80	0.84	0.85	0.89	0.90	0.94	0.96
ERNIE	0.78	0.82	0.80	0.81	0.85	0.87	0.90	0.92	0.95	0.97
RoBERTa	0.86	0.92	0.86	0.91	0.89	0.92	0.94	0.96	0.98	1.00
$\overline{\Delta}$	0.0385		0.0243		0.0177		0.0215		0.0183	

Table 7: Tendency to predict 0 with edit distance less than 6. $\overline{\Delta}$ denotes the mean of T_0^{focus}-T_0 on BERT, ERNIE and RoBERTa.

long edit distance examples with category 0 suffers from overlapping shortcut.

Results analysis. We report the models' prediction tendency with short edit distance in Tab. 7. From the Tab. 7, we can observe that models have a higher tendency to predict 0 on focus-bias examples than "normal" bias-examples, which implies that models tend to predict 0 if we try to eliminate the word-overlap bias. Specifically, comparison with normal bias-examples, the average T_0^{focus} of three models with edit distance 1 increases by 0.0385, which is 0.0243, 0.0177, 0.0215 and 0.0183 for edit distance of 2, 3, 4 and 5. The less word-overlap the samples has, the more significant the impact of bias is.

Generally, we can deduce that models tend to assign labels relying on the distribution bias trick. With eliminating the influence of word-overlap, the models' prediction tendency towards 0 becomes significant on OPPO. Besides the bias-word shortcut we study in this work, QM models are also affected by many other shortcuts and they influence the models' behaviors together.

5 How to mitigate model's reliance on bias?

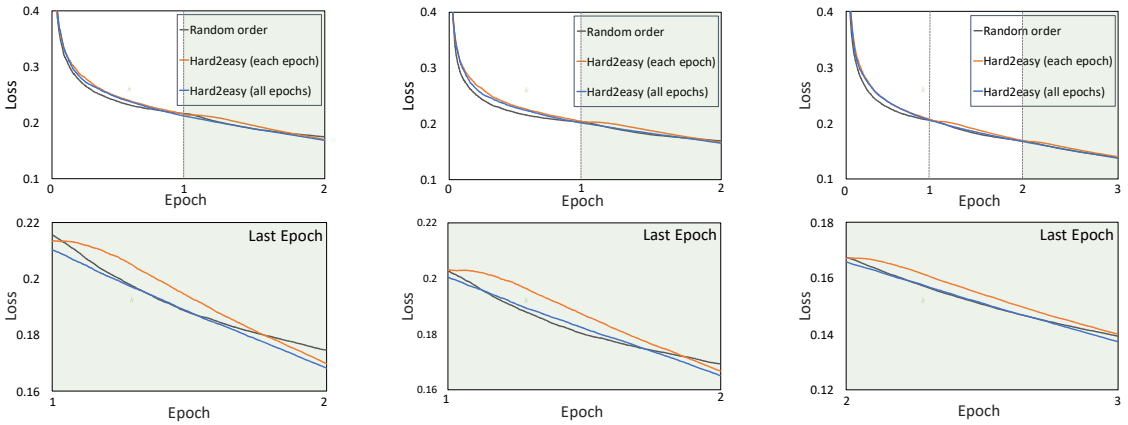
Previous work argues that models tend to learn the bias at a very early stage of training (Lai et al., 2021). Likewise, the training loss curve in Fig. 1 reflects that the loss curve of bias-examples drops more rapidly than random shuffling. These findings imply that model tends to find superficial cues firstly, which is the easiest way to fit the training data. Motivated by these observations, we propose a training strategy that the order of training samples is re-organized in a *hard-to-easy* form to mitigate the models' reliance on bias.

5.1 Hard-to-Easy in each epoch

To alleviate this shortcut learning behavior of models, a straightforward idea is that model's training starts from no-bias-examples and gradually moves

Model	Approach	LCQMC$_{test}$	DuQM	OPPO
BERT	Random sampling	86.93$_{\pm 0.41}$	67.65$_{\pm 0.82}$	81.40$_{\pm 0.48}$
	Hard2easy (each epoch)	87.36$_{\pm 0.78}$	68.54$_{\pm 1.42}$	**81.71**$_{\pm 0.39}$
	Hard2easy (all epochs)	**87.38**$_{\pm 0.48}$	**68.82**$_{\pm 0.85}$	81.60$_{\pm 0.35}$
ERNIE	Random sampling	86.72$_{\pm 0.65}$	70.88$_{\pm 1.72}$	82.23$_{\pm 0.21}$
	Hard2easy (each epoch)	87.27$_{\pm 0.45}$	70.40$_{\pm 2.10}$	82.26$_{\pm 0.31}$
	Hard2easy (all epochs)	**87.65**$_{\pm 0.54}$	**71.48**$_{\pm 0.61}$	**82.45**$_{\pm 0.25}$
RoBERTa	Random sampling	87.60$_{\pm 0.94}$	73.92$_{\pm 0.50}$	82.56$_{\pm 0.25}$
	Hard2easy (each epoch)	**87.78**$_{\pm 0.26}$	74.10$_{\pm 0.80}$	82.48$_{\pm 0.38}$
	Hard2easy (all epochs)	87.74$_{\pm 0.27}$	**74.32**$_{\pm 0.45}$	**82.75**$_{\pm 0.34}$

Table 8: Accuracy (%) of random sampling, hard2easy (each epoch) and hard2easy (all epochs) on three test sets. Each experiment is repeated five times with different random seeds and we report mean and standard deviation here. The experimental settings are same as we described in Sec. 2. It is worth mentioning that we fine-tune BERT and ERNIE for 2 epochs, RoBERTa for 3 epochs, in which all models can converge.



(a) Training loss curves on BERT. (b) Training loss curves on ERNIE. (c) Training loss curves on RoBERTa.

Figure 4: Training loss curves of three models. To better present how curves converge, we provide detailed figures of last epoch below.

on to bias-examples. If we present more no-bias-examples at the early stage, the models will be prevented from fitting bias feature and forced to learn other semantic features.

Implementation. The conventional manner of training neural model is to perform mini-batch stochastic gradient descent (mini-batch SGD) and the examples in each mini-batch are chosen randomly. In our proposed training strategy, we pre-sample training examples in a hard-to-easy form: the proportion of bias-examples in the example set we have sampled grows linearly, until all the bias-examples are selected. Appendix A contains more details about our sampling procedure to get a linearly-increasing hard-to-easy order. The experimental setup is same as we descried in Sec. 2.3.

Results analysis. The experiment results are shown in column *hard2easy (each epoch)* of Tab. 8. Compared to random sampling, a hard-to-easy order improves the accuracy of BERT on all three test sets, which is by 0.43% on LCQMC$_{test}$, 0.89% on DuQM, and 0.31% on OPPO. The effects on ERNIE and RoBERTa are not significant. We guess that hard-to-easy (each epoch) for more than one epoch results in a non-consistent increasing on proportion of bias-examples, which increases from 0% to 41.15% in the first epoch (the proportion of bias-examples in LCQMC$_{train}$ is 41.15%) and fluctuates in next epochs (see Fig. 5).

Hard-to-easy in each epoch is effective for BERT, but does not improve the performance of ERNIE and RoBERTa. If we train the model more than one epoch, this strategy would not increase the proportion of bias-examples linearly. To overcome this limitation, we optimize our strategy and propose the training strategy *hard-to-easy all epochs*.

Model	Approach	LCQMC$_{test}$	DuQM	OPPO
BERT	Random sampling	40.28	37.23	24.94
	Hard2easy(all epochs)	40.28	36.69$_{\downarrow}$	23.61$_{\downarrow}$
ERNIE	Random sampling	40.49	36.99	25.92
	Hard2easy(all epochs)	40.17$_{\downarrow}$	36.47$_{\downarrow}$	25.62$_{\downarrow}$
RoBERTa	Random sampling	39.75	36.56	24.04
	Hard2easy(all epochs)	38.80$_{\downarrow}$	35.76$_{\downarrow}$	23.84$_{\downarrow}$

Table 9: Probability (%) that bias-word makes the greatest contribution to final prediction. We compare the results between random order and our method hard2easy (all epochs). Our strategy reduces the contribution of bias-words to all models' prediction on all test sets.

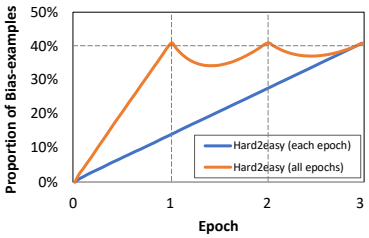


Figure 5: The proportion of bias-examples in the examples we have sampled.

5.2 Hard-to-Easy all epochs

To achieve hard-to-easy in more intuitive way, as shown in Fig. 5, we re-order the bias-examples for the whole training process, i.e., the proportion of bias-examples in the training set we have sampled grows linearly from the start to the end of training.

In Fig. 4, we compare the training loss curves of random order and hard-to-easy. For all three models, in the first epoch, the loss curves of two hard-to-easy are above random order, since we present more challenging examples at the early stage; in each epoch, the loss curves of hard-to-easy (each epoch) are firstly slightly above all epochs and then the two curves overlap; due to the increasing proportion of no-bias-examples in the beginning of each epoch, the curves of each epoch have a slight rising in epochs' start; in the last epoch (green figures below), the loss curves of all epochs converge to the lowest value.

Results analysis. We report the results in column *hard2easy (all epochs)* of Tab. 8. Across all models and test sets, hard-to-easy (all epochs) outperforms random sampling and hard-to-easy (each epoch). Across three models, the improvement of BERT is highest, which is 0.45% on LCQMC$_{test}$, 1.17% on DuQM, 0.20% on OPPO. Especially on DuQM, in which the proportion of bias-examples is highest, our strategy brings the greatest improvement for all models, which is 1.17% for BERT, 0.60% for ERNIE, 0.40% for RoBERTa.

Besides the model's performance, we are also concerned about whether our strategy helps models shift attention from bias-words. We compare the contribution of bias-words with random sampling and hard-to-easy (all epochs) in Tab. 9. Our strategy reduces all three models' attention on bias-words successfully across all three test sets. For example, when RoBERTa predicts on LCQMC$_{test}$, the contribution of bias-words decreases by 0.95%, which represents the model pay less attention on bias-words with our strategy.

We provide a example to explain how our strategy helps model focus on wright words in Appendix B. In conclusion, with our simple strategy, the performance of the models are improved on all three test sets. Moreover, the contribution of bias-words become less significant after applying our strategy. It is an effective approach to mitigate the bias shortcut in QM datasets that we re-organize the training order from hard to easy.

6 Conclusion

In this paper, we explore the biased data distribution to explain the shortcut learning behavior of the QM models. Specifically, we observe that bias-examples are easier being learned than others, and bias-words make significantly higher contributions to model predictions than random words. Besides, we observe that the models tend to assign labels that are highly correlated to the bias-words. According to our observation, we propose a simple approach to mitigating the shortcut in QM task, that learns more no-bias-examples first but more bias-examples last, and the experiment results demonstrate the effectiveness of our proposed approach. In the future work, we will apply this analysis framework and mitigation approach to other NLP tasks.

References

Emily M Bender and Alexander Koller. 2020. Climbing towards nlu: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29:4349–4357.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Mengnan Du, Varun Manjunatha, R. Jain, Ruchi Deshpande, Franck Dernoncourt, Jiuxiang Gu, Tong Sun, and Xia Hu. 2021. Towards interpreting and mitigating shortcut learning behavior of nlu models. In *NAACL*.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. *arXiv preprint arXiv:2004.06100*.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. *arXiv preprint arXiv:1707.07328*.

Zhenyu Jiao, Shuqi Sun, and Ke Sun. 2018. Chinese lexical analysis with deep bi-gru-crf network. *arXiv preprint arXiv:1807.01882*.

Masahiro Kaneko and Danushka Bollegala. 2021. Debiasing pre-trained contextualised embeddings. *arXiv preprint arXiv:2101.09523*.

Divyansh Kaushik and Zachary C Lipton. 2018. How much reading does reading comprehension require? a critical investigation of popular benchmarks. *arXiv preprint arXiv:1808.04926*.

Pride Kavumba, Benjamin Heinzerling, Ana Brassard, and Kentaro Inui. 2021. Learning to learn to be right for the right reasons. *arXiv preprint arXiv:2104.11514*.

Pride Kavumba, Naoya Inoue, Benjamin Heinzerling, Keshav Singh, Paul Reisert, and Kentaro Inui. 2019. When choosing plausible alternatives, clever hans can be clever. *arXiv preprint arXiv:1911.00225*.

Fereshte Khani and Percy Liang. 2021. Removing spurious features can hurt accuracy and affect groups disproportionately. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 196–205.

Yuxuan Lai, Chen Zhang, Yansong Feng, Quzhe Huang, and Dongyan Zhao. 2021. Why machine reading comprehension models learn shortcuts? *arXiv preprint arXiv:2106.01024*.

Xin Liu, Qingcai Chen, Chong Deng, Huajun Zeng, Jing Chen, Dongfang Li, and Buzhou Tang. 2018. Lcqmc: A large-scale chinese question matching corpus. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1952–1962.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Chandler May, Alex Wang, Shikha Bordia, Samuel R Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. *arXiv preprint arXiv:1903.10561*.

R Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *arXiv preprint arXiv:1902.01007*.

John X Morris, Eli Lifland, Jack Lanchantin, Yangfeng Ji, and Yanjun Qi. 2020. Reevaluating adversarial examples in natural language. *arXiv preprint arXiv:2004.14174*.

Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. *arXiv preprint arXiv:1907.07355*.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. *arXiv preprint arXiv:1805.01042*.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. *arXiv preprint arXiv:2004.07667*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " 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*, pages 1135–1144.

Saku Sugawara, Kentaro Inui, Satoshi Sekine, and Akiko Aizawa. 2018. What makes reading comprehension questions easier? *arXiv preprint arXiv:1808.09384*.

Saku Sugawara, Pontus Stenetorp, Kentaro Inui, and Akiko Aizawa. 2020. Assessing the benchmarking capacity of machine reading comprehension datasets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8918–8927.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Lifu Tu, Garima Lalwani, Spandana Gella, and He He. 2020. An empirical study on robustness to spurious correlations using pre-trained language models. *Transactions of the Association for Computational Linguistics*, 8:621–633.

Zhao Wang and Aron Culotta. 2020. Robustness to spurious correlations in text classification via automatically generated counterfactuals. *arXiv preprint arXiv:2012.10040*.

Kellie Webster, Xuezhi Wang, Ian Tenney, Alex Beutel, Emily Pitler, Ellie Pavlick, Jilin Chen, Ed Chi, and Slav Petrov. 2020. Measuring and reducing gendered correlations in pre-trained models. *arXiv preprint arXiv:2010.06032*.

Keren Ye and Adriana Kovashka. 2021. A case study of the shortcut effects in visual commonsense reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3181–3189.

Sicheng Yu, Yulei Niu, Shuohang Wang, Jing Jiang, and Qianru Sun. 2020. Counterfactual variable control for robust and interpretable question answering. *arXiv preprint arXiv:2010.05581*.

A Sampling procedure of hard-to-easy

Algorithm 1 Sampling procedure of hard2easy

Define:

N : Training epoch

T : Training set

T_{bias} : Bias-examples in T

$T_{no-bias}$: No-bias-examples in T

Initialize:

$T_{Hard2easy} \leftarrow \emptyset$

Set α to ensure that the remaining examples in T_{bias} or $T_{no-bias}$ at the end are as few as possible.

Process of sampling:

for $i = 1$ *to* $N \times Size(T)$ **do**
  if $Size(T_{bias}) == 0$ **then**
    Insert $T_{no-bias}$ into $T_{Hard2easy}$;
    Break;
  if $Size(T_{no-bias}) == 0$ **then**
    Insert T_{bias} into $T_{Hard2easy}$;
    Break;
  $k \leftarrow 100 - (\alpha \times i)$;
  $Num \leftarrow RandInit(0, 100)$;
  if $Num \geq k$ **then**
    Sample example from T_{bias};
    Append example to $T_{Hard2easy}$;
  else
    Sample example from $T_{no-bias}$;
    Insert example into $T_{Hard2easy}$;

return $T_{Hard2easy}$

We pre-define the training order with algorithm shown in Alg. 1, which helps us organize the training samples in a hard-to-easy form. We divide the training set T into two sets T_{bias} and $T_{no-bias}$. With tuning k, the probability of sampling from $T_{no-bias}$ decreases, so as to present more no-bias-examples at the early stage and more bias-examples at the late stage. The k decreases linearly as the number of samples increases and the slope is α. Until the end of sampling, either T_{bias} or $T_{no-bias}$ will have remaining examples. In order to fit the size of training set, we need tune the value of α to ensure that the remaining examples in T_{bias} or $T_{no-bias}$ are as few as possible.

B The effect of hard-to-easy on words' contribution

As our strategy shows the highest improvement in DuQM, we conduct a case study on it. We filter out 106 examples where RoBERTa predicts wrongly with random sampling but correctly with hard-to-easy (all epochs). Out of the 48 examples which were predicted wrongly and focused incorrectly, 31 examples model focuses correctly after employing hard-to-easy (all epochs) and makes a correct prediction.

As the example shown in Fig. 6, RoBERTa focuses on "cervical spondilosis" with random order; if we re-order the training examples with hard-to-easy all epochs, the most important words are "serious" and "common". The model detects the differences and predicts correctly after employing our strategy.

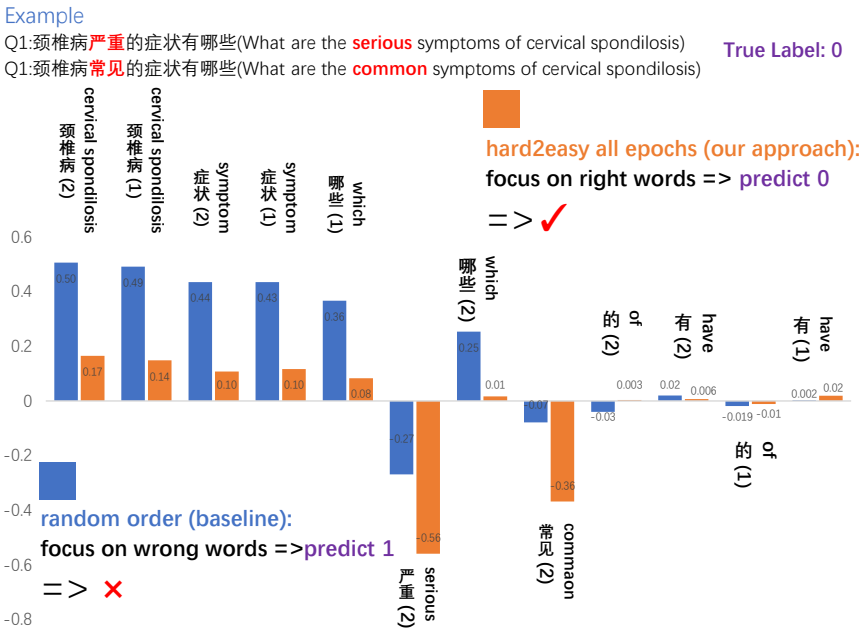


Figure 6: With random sampling, RoBERTa focuses on wrong words (which are bias-words) and predicts incorrectly. With our hard-to-easy (all epochs), the contribution of right words increase significantly and model makes a right prediction.