Interventional Training for Out-Of-Distribution Natural Language Understanding

Sicheng Yu1,2∗ Jing Jiang1 Hao Zhang3 Yulei Niu4 Qianru Sun1 Lidong Bing1,2
1Singapore Management University  2DAMO Academy, Alibaba Group
3Nanyang Technological University  4Columbia University

Abstract

Out-of-distribution (OOD) settings are used to measure a model’s performance when the distribution of the test data is different from that of the training data. NLU models are known to suffer in OOD settings (Utama et al., 2020b). We study this issue from the perspective of causality, which sees confounding bias as the reason for models to learn spurious correlations. While a common solution is to perform intervention, existing methods handle only known and single confounder (Pearl and Mackenzie, 2018), but in many NLU tasks the confounders can be both unknown and multifactorial. In this paper, we propose a novel interventional training method called Bottom-up Automatic Intervention (BAI) that performs multi-granular intervention with identified multifactorial confounders. Our experiments on three NLU tasks, namely, natural language inference, fact verification and paraphrase identification, show the effectiveness of BAI for tackling different OOD settings.

1 Introduction

From the era of word embeddings (Pennington et al., 2014) to pre-trained language models (Devlin et al., 2019), researchers of natural language understanding (NLU) have tried to push the performance on benchmark datasets. Traditional settings assume independent and identical distribution (IID) in training and testing splits. However, the IID setting cloaks the vulnerability of neural models, i.e., neural models tend to learn non-robust “shortcut” patterns in the training data but fail to make robust predictions on unseen samples. To evaluate the robustness of models, the out-of-distribution (OOD) setting draws the attention of the NLU community.

For example, the task of natural language inference (NLI) determines whether a hypothesis can be entailed from a premise. We can observe that the lexical overlap between the hypothesis and the premise correlates with the entailment label on the benchmark MNLI dataset (Williams et al., 2018) (as shown in the top part of Figure 1). McCoy et al. (2019) proposed an OOD set named HANS for NLI. As shown in the bottom part of Figure 1, HANS does not have the correlation between lexical overlap and the entailment label. NLI models that rely on the lexical overlap heuristic suffer from a significant degradation on HANS (Utama et al., 2020b).

Recently, causal inference has been adopted in NLP to identify robust correlations by analyzing reliable causal effects between variables (Zhang...
In the stock market, the damage can get much worse. The stock market can experience much worse damage.

---

**Natural bias:**
- Probability of entailment proportional to lexical overlap,
- Probability of entailment not proportional to lexical overlap

---

**Confounder**
- Crowdsourced worker preference:
  - Inexperienced workers, experienced workers

---

**Input**

\[ X \]

**Sentence**

\[ Y \]

**Label**

\[ \text{Entailment} \]

---

Figure 2: (a) Causal graph of NLU tasks, (b) intervention operation, and (c) an example of each node in the causal graph on the NLI task, where the data sample is from MNLI (Williams et al., 2018).

Contributions:
1. We analyze the issue of NLU vulnerability from the perspective of causality. (Pearl, 2009, 2010), the crux under a model’s vulnerability is **confounding bias**. We summarize the causal relations behind NLU tasks as a causal graph in Figure 2(a). \( X \) represents the input, e.g., a pair of sentences for NLI, and \( Y \) represents a label to be predicted. \( X \rightarrow Y \) represents the desired relation for a robust NLU model, i.e., how to predict the label with reliable understanding of the input. \( X \leftarrow C \rightarrow Y \) denotes a backdoor path of some unreliable relation between \( X \) and \( Y \) confounded by the confounder \( C \). Examples of \( C \) include nature bias in the dataset (Tang et al., 2020) or crowdsourced workers preference (Geva et al., 2019). For instance, in NLI, \( C \) may represent the degree of lexical overlap between the premise and the hypothesis, which is correlated with the entailment relation in the MNLI dataset (see Figure 1). When crowdsourced workers are engaged to create hypotheses for NLI, \( C \) could be the experience level of a worker, with inexperienced workers more likely to write simple sentences with straightforward meanings. As a result, these examples of \( C \) will make \( X \) and \( Y \) spuriously correlated.

A common solution of deconfounding is **intervention** (Pearl et al., 2016; Pearl and Mackenzie, 2018), which aims to block the backdoor path (or spurious correlation) by cutting off \( C \rightarrow X \) (see Figure 2(b)). The key idea is to stratify \( X \) into different environments (Arjovsky et al., 2019; Teney et al., 2021), i.e., several subsets of training data, according to the identified confounder. Then the model is expected to make environment-agnostic prediction. By doing so, we are controlling \( X \) and thus break the backdoor path by D-Separation (Koller and Friedman, 2009).

Figure 2(c) depicts an example where the NLI training data is stratified into several environments, e.g., one with obvious trend of lexical overlap bias and another does not. Then the NLI model is trained to fit both environments.

However, the confounder \( C \) is not always observed. Furthermore, confounders can be multifactorial in NLU, e.g., it may contain both inherent dataset bias and artifacts from crowdsourced workers. Both scenarios make intervention non-trivial. In this paper, we propose BAI, a bottom-up automatic intervention method, which can (1) identify the unobserved confounder(s) automatically, and (2) perform multi-granular intervention to handle multifactorial confounders. Inspired by Creager et al. (2021), the automatic stratifying mechanism is realized by maximizing the difference between data in different environments.

We further propose a novel **bottom-up** intervention mechanism that aims to address the multifactorial characteristic of \( C \). While most existing debiasing work only considers a single bias, our bottom-up mechanism enables the model to pick up different confounders in two rounds of interventions. Specifically, based on our preliminary experiments, we find that fine-grained partition (i.e., partition with more environments) results in smaller differences between environments, making environment-agnostic learning easier. Thus we start from a fine-grained partition. We then move on to a coarse-grained partition to further block the backdoor effect via \( C \) and make the learning environment-agnostic.

We apply BAI on three OOD benchmarks for NLU tasks. The results show that our method outperforms state-of-the-art methods, e.g., achieving 7 percentage points of absolute gains from the previous best method under OOD setting of Quora Question Pairs (QQP) (Zhang et al., 2019), a benchmark dataset for paraphrase identification.

**Contributions:**
1. We analyze the issue of NLU vulnerability from the perspective of causality analysis;
2. We propose a bottom-up automatic intervention method to perform intervention for unobserved and multifactorial confounders; and
3. Here an environment refers to a subset of training data. A partition is an assignment of the whole training set into multiple environments, e.g., a partition with five environments.

---

\( X \) \( Y \)

---

\( C \)
extensive experiments on three OOD benchmarks demonstrate that our method outperforms state-of-the-art methods.

2 Related Work

OOD Generalization. OOD settings have been studied in recent years in NLU. To tackle dataset bias, most existing work relies on instance reweighting with a bias model for debiasing. Specifically, these methods (Cadene et al., 2019) first design a bias model and then train a target debiased model fused with the bias model. Training instances predicted correctly by the bias model will be down-weighted in the training of the debiased model. Early work mainly revolves around different fusion methods (He et al., 2019; Clark et al., 2019; Utama et al., 2020a; Mahabadi et al., 2020) with known bias. Then researchers started looking into unknown bias by designing the bias model with heuristics, e.g., a model trained with very small amount of data (Utama et al., 2020b) or a model with only the bottom layers of the language model (Ghaddar et al., 2021).

However, instance reweighting based methods rely on either prior knowledge of bias or heuristic design of the bias model. Furthermore, it is pointed out that such bias models may not be able to predict the main model’s reaction of biased samples and reweighting may waste data (Amirkhani and Pilehvar, 2021). In contrast, our method is derived from causal inference (Pearl and Mackenzie, 2018), which is not related to any form of reweighting. Meanwhile, our method does not adopt any bias model (which requires carefully design or prior knowledge of bias).

Causal Intervention. Causality inference (Pearl, 2009, 2010) measures the causal effect between variables and has been widely applied to various scenarios, e.g., social science (Baron and Kenny, 1986), medical science (Hall et al., 1993), and other applications (Niu et al., 2021; Yu et al., 2020; Niu and Zhang, 2021). Recently, causality inference is introduced to the machine learning community and intervention is one of the techniques in causality inference. Intervention (Pearl, 1993) helps to eliminate the effect of confounders (Yang et al., 2021; Yue et al., 2020; Qi et al., 2020; Nan et al., 2021; Zhu et al., 2022; Niu et al., 2022). Invariant Risk Minimization (IRM) (Arjovsky et al., 2019) implements intervention by learning a model invariant to different environments (Arjovsky et al., 2019; Wang et al., 2021). Although IRM has been widely adopted in computer vision (CV) (Krueger et al., 2021; Rosenfeld et al., 2020; Creager et al., 2021; Liu et al., 2021; Wang et al., 2021; Teney et al., 2021), to the best of our knowledge, our proposed BAI is an initial work of applying IRM in NLU. Our work is distinguishable from previous work in two aspects. First, previous work in CV mainly focuses on image with annotated background as confounder while the confounder in NLU is more abstract and vague. Second, our method is the first work considering multiple partitions for handling multifactorial confounders.

3 Method

3.1 Preliminaries

Causal Intervention is the core idea of this paper. We formulate NLU tasks with a causal graph (Pearl and Mackenzie, 2018), which illustrates the causal relationships between variables with a directed acyclic graph. As shown in Figure 2, each node represents a variable, e.g., a pair of sentences or a label for NLU tasks, and each directed edge denotes that the head node has direct effect on the tail node.

Naïve model training, i.e., empirical risk minimization (ERM) (Vapnik, 1991), indiscriminately learns both spurious correlation $X \leftarrow C \rightarrow Y$ and causal correlation $X \rightarrow Y$. Specifically, by applying Bayes’ rule on Figure 2(a), we can obtain:

$$P(Y|X) = \sum_c P(Y|X, c)P(c|X), \quad (1)$$

where the bias is introduced via $P(C|X)$. For example, consider the NLI task. Let $X$ be a pair of two sentences (premise and hypothesis) and $Y$ the entailment label. Let $C$ represents the degree of lexical overlap between the two sentences in $X$, and let $c_1$ and $c_2$ denote two situations: having obvious lexical overlap and having little or no lexical overlap. Typically on IID training data of NLI, $P(c_1|X)$ is larger than $P(c_2|X)$, and thus $P(c_1|X)$ tends to dominate the overall term, $P(Y|X)$. In other words, model tends to learn $P(Y|X)$ from $c_1$ instead of $X$.

In contrast, causal intervention in Figure 2(b) yields:

$$P(Y|do(X)) = \sum_c P(Y|X, c)P(c), \quad (2)$$
where the \( \text{do}(X) \) denotes that intervention is conducted on \( X \). With \( \text{do} \) operation, \( e \) is no longer associated with \( X \) and thus the model treats \( c_1 \) and \( c_2 \) fairly subject to the prior distribution of \( C \).

**Invariant Risk Minimization** (Arjovsky et al., 2019) (IRM) is one of the popular tools for intervention in deep neural networks. Given the stratified environments, IRM targets at a robust model which is invariant to environments. In our paper, we utilize two versions of IRM.

Given the input \( X \), model \( f \) and the partition of environments \( \mathcal{E} \), the original version of IRM (Arjovsky et al., 2019) minimizes the objective:

\[
\text{IRM}_{e1} = \sum_{e \in \mathcal{E}} \text{XE}(f(X^e), Y) + \lambda \cdot \| \nabla_{w_{\text{ref}}} \text{XE}(w \cdot f(X^e), Y) \|^2, \tag{3}
\]

where \( X^e \) denotes the data in the environment of \( e \) and \( \text{XE} \) denotes cross-entropy loss. \( w \) is a fixed dummy classifier. The second term measures the optimality of \( w \) for each environment to encourage the model to make environment-invariant predictions. This version of IRM is unstable due to the second-order derivatives.

Another version of IRM (Teney et al., 2021) adopted in our paper initializes individual classifier \( \mathbf{W}_e \) for each environment \( e \) while all environments share one feature extractor. Here we denote the model for the environment \( e \) as \( f^e = \mathbf{W}_e \circ \Phi \) where \( \Phi \) is a feature extractor, e.g., BERT. The corresponding loss is written as:

\[
\text{IRM}_{e2} = \sum_{e \in \mathcal{E}} \text{XE}(f^e(X^e), Y) + \lambda \cdot \text{Var}(\mathbf{W}_e). \tag{4}
\]

The second term is the variance of classifier weights, which encourages optimal classifiers for different environments to be close to each other\(^4\).

### 3.2 Bottom-up Automatic Intervention

To implement intervention on NLU tasks with the unobserved and multi-factorial confounder, we propose a Bottom-up Automatic Intervention (BAI) method using IRM. Figure 3 and Algorithm 1 (in Appendix) show the overall pipeline of BAI. It consists of two components: **automatic stratification** and **bottom-up intervention**. The automatic stratification component generates partition of environments by maximizing the difference between data in different environments based on a reference model. The bottom-up intervention component performs intervention at two levels of granularity.

**Automatic Stratification** generates the partition of environments with unobserved confounder. A good partition is achieved when a reference model behaves differently under different environments. Inspired by Creager et al. (2021), we first train a reference model \( f_{\text{ref}} \) through the naïve trained BERT (Devlin et al., 2019). Note the second term of Eq. 3 is to make environment-invariant prediction, that is, to minimize the difference of data behavior across environments. Inversely, our goal is to maximize the difference of data behavior by magnifying the second term of IRM.

As shown in Figure 3(a), we initialize an environment matrix \( \mathbf{M} \in \mathbb{R}^{D \times N} \) indicating the belonging of each training sample to each environment, where \( D \) and \( N \) denote the number of training data and pre-defined environments, respectively. \( \mathbf{M}_{i,j} \) is the probability of \( i \)-th sample belonging to \( j \)-th environment. IRM\(_{e2}\) is not applicable since the naïve trained reference model only has one classifier. Thus we derive \( \mathbf{M} \) by fixing the refer-

\(^4\)More details about IRM in Appendix Section ??.
ence model \( f_{\text{ref}} \) and maximizing the second term of IRMv1 as follows:

\[
\max_{M} \sum_{e \in E} \| \nabla w \mid w = 1.0 \mid X E(w \cdot f_{\text{ref}}(X^e), Y) \|_2^2, \tag{5}
\]

where \( E \) is the partition of environments determined by \( M \). Note that \( \max \) operation makes the backward-propagation of gradients from \( M \) infeasible. To address this issue, we deploy the Gumbel Softmax trick (Jang et al., 2016) to re-formulate the discrete sampling as:

\[
\mathcal{E} = g(M) = \text{Gumbel-Softmax}(M). \tag{6}
\]

We term the environment matrix with \( n \) environments as \( M_n \). Specifically, we deploy automatic stratifying to extract two environments matrices, i.e., fine-grained \( M_{n_1} \) and coarse-grained \( M_{n_2} \) \((n_1 > n_2)\), for bottom-up intervention.

**Bottom-Up Intervention** adopts multi-granular partitions for intervention in a bottom-up fashion, to derive a robust model \( f_{\text{int}} \). As shown in Figure 3(b), bottom-up intervention consists of two rounds of intervention deployed by IRMv2 due to its stability and scalability.

We first generate fine-grained partition \( \mathcal{E}_{n_1} \) and coarse-grained partition \( \mathcal{E}_{n_2} \) from \( M_{n_1} \) and \( M_{n_2} \) (see Figure 3(b)), where the number of environments in \( \mathcal{E}_{n_1} \) is larger than that in \( \mathcal{E}_{n_2} \). Second, we start from the fine-grained partition \( \mathcal{E}_{n_1} \) and train the intervened robust model \( f_{\text{int}} \). Similarly, we decompose \( f_{\text{int}} = W \circ \Phi \) where \( \Phi \) is feature extractor, e.g., BERT, and \( W \) is a set of learned classifiers. We use \( W_e \) to represent the classifier exclusive to environment \( e \) and \( W \{ \mathcal{E} \} \) to denote the set of classifiers for \( \mathcal{E} \) partition, that is, \( W \{ \mathcal{E}_{n_1} \} = \{ W_e \mid e \in \mathcal{E}_{n_1} \} \) represents all classifiers for partition \( \mathcal{E}_{n_1} \). The feature extractor and the classifiers of \( \mathcal{E}_{n_1} \) in bottom fine-grained intervention are optimized by:

\[
\min_{\Phi, W} \sum_{e \in \mathcal{E}_{n_1}} \text{XE}(f_{\text{int}}^e(X^e), Y) + \lambda \text{Var} (W_e), \tag{7}
\]

Then we conduct the intervention of coarse-grained partition \( \mathcal{E}_{n_2} \). To prevent the catastrophic forgetting, i.e., the intervention with new partition may make the model forget the invariant property on previous partition, we incorporate the idea from continual learning (Li and Hoiem, 2017; Rebuffi et al., 2017). Specifically, we fix the parameter of model \( f_{\text{int}} \) including the feature extractor and \( n_1 \) classifiers for \( \mathcal{E}_{n_1} \). Then we augment \( n_2 \) classifiers for the new partition \( \mathcal{E}_{n_2} \), resulting in \( n_1 + n_2 \) classifiers. Here we only optimize the \( n_2 \) augmented classifiers during training as:

\[
\min_{W} \sum_{e \in \mathcal{E}_{n_2}} \text{XE}(f_{\text{int}}^e(X^e), Y) + \lambda \text{Var} (W_e), \tag{8}
\]

where the first term is based on the new partition \( \mathcal{E}_{n_2} \) while the second term computes the variance of classifier weights across all \( n_1 + n_2 \) classifiers.

**Inference** is based on the design of IRMv2 (Teney et al., 2021). Since we are not able to distinguish which environment the input data belongs to, we simply average the weight of \( n_1 + n_2 \) classifiers for inference:

\[
\hat{Y} = f_{\text{int}}^\Phi(X) = \bar{W} \cdot \Phi(X), \tag{9}
\]

where \( \bar{W} \) denotes the mean weight of all classifiers.

The overall pipeline of BAI is summarized in Algorithm 1.

**Algorithm 1 BAI Training**

1: **Input:** Dataset \( D \), reference model \( f_{\text{ref}} \)
2: **Output:** \( f_{\text{int}} = W \circ \Phi \)
3: Initialize environment matrix \( M_{n_1}, M_{n_2} \)
4: Update \( M_{n_1}, M_{n_2} \) with Eq. 5 and Eq. 6
5: Initialize \( f_{\text{ref}} \)
6: for \( X \) in \( D \) do
7: Get environment \( e \in \mathcal{E}_{n_1} \) of \( X \) from \( M_{n_1} \)
8: Update \( \Phi \) and \( W_e \) with Eq. 7
9: end for
10: for \( X \) in \( D \) do
11: Get environment \( e \in \mathcal{E}_{n_2} \) of \( X \) from \( M_{n_2} \)
12: Update \( W_e \) with Eq. 8
13: end for

**4 Experiment**

**4.1 NLU Tasks and Benchmarks**

We apply our method on three NLU tasks to evaluate the effectiveness of our method. Specifically, we train on the original training set and evaluate on both the IID and the OOD evaluation sets. The accuracy is reported for all the benchmark datasets.

**Natural Language Inference** aims to classify the relationship between two sentences, i.e., a premise and a hypothesis, into three classes: “entailment”, “contradiction” and “neutral”. It has been observed that NLI models may rely on the lexical overlap bias (McCoy et al., 2019). We adopt
### 4.3 Comparison with SOTAs

In this section, we compare our method with the following baselines: **Naïve Fine-tuning** (Devlin et al., 2019) directly fine-tunes the pre-trained language model on the downstream NLU tasks; **Reweighting** (Clark et al., 2019) reweights each training sample according to the confidence on bias.

For standard hyperparameters for the training of NLU model, we use the same configuration as Utama et al. (2020a,b), i.e., 3 epochs of training, learning rate of $5e^{-5}$ for NLI and $2e^{-5}$ for fact verification and paraphrase identification. Unlike previous methods (Clark et al., 2019; Grand and Belinkov, 2019; Clark et al., 2020; Sanh et al., 2020; Ghaddar et al., 2021) which are directly evaluated on the OOD set, we only perform checkpoint selection on the OOD set. We choose hyperparameters exclusive to our method according to the analysis on the NLI task (see RQ3) and deploy the same configuration for the other two tasks to avoid hyperparameter tuning. Specifically, we set the learning rate to $1e^{-2}$ for automatic stratification to optimize the environment matrix, and $n_1 = 5$ and $n_2 = 2$ for bottom-up intervention. We also fix $\lambda$ to $1e2$. Note the coarse-grained partition may require multiple turns of training to achieve better performance. The average results over 5 runs with different random seeds are reported.
model; **Product-of-Expert** (Hinton, 2002) trains the robust model fused with the bias model by sum of logits; **Learned-Mixin** (Clark et al., 2019) utilizes a different fusion method. **Regularized-Confidence** (Utama et al., 2020a) enhances the model in a knowledge distillation fashion; Unknown bias version methods in Utama et al. (2020b) adopt the bias model trained only with a small number of data; **Forgettable Examples** (Yaghoobzadeh et al., 2021) trains the model with an additional round with the forgotten data; **Self-Debiasing** (Ghaddar et al., 2021) utilizes bottom layers of model as the bias model; **EIIL** (Creager et al., 2021) is the IRM method that inspired this paper, which is originally applied to CV.

Table 1 summarizes the performance comparison between BAI and the above SOTA methods. Overall, BAI achieves the top performance on all the OOD sets. Specifically, BAI significantly outperforms naïve Fine-tuning by doubling the accuracy on PAWS (65.0% vs. 33.5%), which demonstrates that BAI with causality-theoretic basis is effective for OOD generalization on NLU tasks. Also, BAI surpasses SOTA methods with 6.9% gains over previous best result on PAWS, which shows the superiority of BAI over reweighting based methods.

We also observe a trade-off between IID and OOD on MNLI and QQP across most of the methods, i.e., performance gains on OOD are achieved with the sacrifice of IID performance. It is because naïve fine-tuning fits IID training data well. Interestingly, the IID test data of FEVER benefits from debiasing methods, which suggests that the data distribution of the IID test data may be different from that of the training data.

### 4.4 Ablation Studies

In this section, we conduct extensive ablation studies to evaluate the components in our BAI and answer the following research questions.

**RQ1: How does each component of BAI contribute to the performance gains?**

**Answer:** We design four ablative settings: (a) Replacing the learned environment matrix with a randomly initialized one; (b) Removing the regularizer term in Eq. 7 and 8; (c) Replacing bottom-up intervention with single intervention, i.e., removing Eq. 8. (d) Using the same number of classifiers on naïve fine-tuning model as our BAI.

As reported in Table 2, the settings (a) and (d) prove that the environment partition is vital in our

<table>
<thead>
<tr>
<th>Ablative Setting</th>
<th>Dev</th>
<th>HANS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naïve FT</td>
<td>84.5</td>
<td>62.4</td>
</tr>
<tr>
<td>(a) Randomized Environment</td>
<td>84.0</td>
<td>62.4</td>
</tr>
<tr>
<td>(b) w/o Regularizer</td>
<td>83.0</td>
<td>66.8</td>
</tr>
<tr>
<td>(c) One Intervention</td>
<td>83.9</td>
<td>69.9</td>
</tr>
<tr>
<td>(d) Naïve FT+Multiple Classifiers</td>
<td>84.4</td>
<td>62.6</td>
</tr>
<tr>
<td>Full Method</td>
<td>82.3</td>
<td>72.7</td>
</tr>
</tbody>
</table>

Table 2: **RQ1.** Results of ablative settings on MNLI. “FT” denotes Fine-tuning.

<table>
<thead>
<tr>
<th>Stratifying Method</th>
<th>Dev</th>
<th>HANS</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Stratifying</td>
<td>84.5</td>
<td>62.4</td>
</tr>
<tr>
<td>(1) Domain Information</td>
<td>84.2</td>
<td>63.2</td>
</tr>
<tr>
<td>(2) Confidence</td>
<td>84.0</td>
<td>67.7</td>
</tr>
<tr>
<td>(3) Lexical Overlap</td>
<td>83.8</td>
<td>65.6</td>
</tr>
<tr>
<td>Automatic Stratifying (Ours)</td>
<td>83.9</td>
<td>69.9</td>
</tr>
</tbody>
</table>

Table 3: **RQ2.** Results of alternative methods for environment stratification on MNLI.

method and the improvement of our method is not from the added parameters \(^{3}\). Result of (b) reveals that both the regularizer term and the design of one classifier for one environment contribute to the gains in our method. Finally, the full method with bottom-up intervention outperforms (c), which demonstrates the effectiveness of multi-granular intervention.

![Figure 4: RQ3. The accuracies of one round of intervention on MNLI with different numbers of environments.](image)

**RQ2: Is there any other solution for stratification?**

**Answer:** Yes. We evaluate several alternative methods for partition on MNLI according to the attached information of training samples: (1) Domain information, *i.e.*, “fiction”, “government”, “slate”, “telephone” and “travel”; (2) Confidence of predic-

\(^{3}\)BAI introduces 0.008% more parameters compared to that of Naïve Fine-tuning.
We first analyze the situation of only one environment in the final intervention is better and observe that applying the partition with two environments in the final intervention is better and increasing the turns of intervention only brings confidence and lexical overlap shows considerable improvements compared to that of no stratifying, which demonstrates the two factors are indeed related to the confounder of MNLI. Note that the automatic stratifying method is designed for unobserved confounder, which outperforms the simple heuristics in settings (2) and (3) without using any prior knowledge of bias.

**RQ3: How to set the number of environments?**

**Answer:** We first analyze the situation of only one round of intervention and visualize the performance trend in Figure 4. Note that setting the number of environments as one equals to naive fine-tuning, i.e., no stratification. Overall, there is a trade-off in the results between Dev and HANS, i.e., IID and OOD performances. This phenomenon is particularly prominent in $E_2$. The reason is that only one intervention forces the model to focus on only one confounder. In this case, it forces the model to pay much attention on the harder samples, i.e., the confounder of crowdsourced worker preference, leading to significant performance drop on dev set (see RQ4 for more details). With the number of environments increasing, the gaps between the environments are also smaller, i.e., the OOD performance of ten environments is close to that of the naive fine-tuning.

We further analyze the multiple interventions. We conduct experiments with the number of interventions in different orders or combinations. The experiment results are summarized in Table 4. We observe that applying the partition with two environments in the final intervention is better and increasing the turns of intervention only brings

<table>
<thead>
<tr>
<th>Order &amp; Combination</th>
<th>Dev</th>
<th>HANS</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E_2 \rightarrow E_5$</td>
<td>81.7</td>
<td>70.1</td>
</tr>
<tr>
<td>$E_5 \rightarrow E_3$</td>
<td>83.7</td>
<td>71.4</td>
</tr>
<tr>
<td>$E_5 \rightarrow E_3 \rightarrow E_2$</td>
<td>81.3</td>
<td>73.5</td>
</tr>
<tr>
<td>$E_5 \rightarrow E_2$ (Config in Table 1)</td>
<td>81.1</td>
<td>73.3</td>
</tr>
</tbody>
</table>

Table 4: RQ3. Results of different orders and combinations of environment numbers on MNLI, arrows represent the intervention order.
 marginal improvements. Thus, we simply fix \( \mathcal{E}_5 \rightarrow \mathcal{E}_2 \) for all tasks in our paper.

**RQ4:** What is each environment like?

**Answer:** Figure 5 inspects each environment in two partitions, *i.e.*, \( \mathcal{E}_5 \) and \( \mathcal{E}_2 \), on MNLI and summarizes the characteristic for each environment. \( \mathcal{E}_2 \) can be regarded as a coarse variant of \( \mathcal{E}_5 \), *i.e.*, the first environment of \( \mathcal{E}_2 \) partition combines four environments of \( \mathcal{E}_5 \). We can see that both partitions contain environments with distinct characteristics. \( \mathcal{E}_2 \) focuses more on crowdsourced worker preference while \( \mathcal{E}_5 \) shows each environment with more diverse situation for the nature bias, *i.e.*, lexical overlap bias.

We further investigate the crowdsourced worker preference in \( \mathcal{E}_2 \), *i.e.*, the difficulty of the samples in these two environments is distinguishable. Samples in the second environment are more challenging compared to the first one. As depicted in Figure 5 (ii), reasoning of easy samples is straightforward, *i.e.*, nice versus not nice and do not like. In contrast, hard examples require a deep understanding of the semantic meaning. For instance, the hard samples with contradiction and entailment as labels expect the model to have the ability to identify the current situation, *e.g.*, no name for now, and the usual situation, *e.g.*, name is usually mentioned in the past. The above inspection reveals that BAI helps to generate meaningful and multifactorial partition.

**RQ5:** Whether BAI is model-agnostic?

**Answer:** We apply the same hyperparameters and partitions on BERT to RoBERTa (Liu et al., 2019). The results in Table 5 demonstrate that the proposed BAI can be applied on more advanced language model than BERT.

<table>
<thead>
<tr>
<th>Method</th>
<th>MNLI</th>
<th>FEVER</th>
<th>QQP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IID</td>
<td>OOD</td>
<td>IID</td>
</tr>
<tr>
<td>Naïve F.T.</td>
<td>87.3</td>
<td>69.8</td>
<td>87.1</td>
</tr>
<tr>
<td>BAI</td>
<td>85.3</td>
<td>76.7</td>
<td>91.2</td>
</tr>
</tbody>
</table>

Table 5: RQ5. Results of BAI with RoBERTa as backbone. “Naïve F.T.” denotes Naïve Fine-tuning

6 Limitations

The limitations of this paper are twofold. First, the proposed method is only evaluated on natural language understanding tasks. Thus the effectiveness on natural language generation tasks and sequence labeling tasks is not guaranteed. Similarly, the optimal hyper-parameters for other tasks may also differ from the selections stated in this paper. Second, the performance trade-off (see Table 1) is non-negligible on the IID set compared to the OOD set. It is not desirable when the model is applied to the normal scenario, *e.g.*, the confounders provide shortcuts for model inference.

Acknowledgments

The authors gratefully acknowledge the support by the Lee Kuan Yew Fellowship awarded by Singapore Management University.

References


