Reducing Annotation Artifacts in Crowdsourcing Datasets for Natural Language Processing

Donghoon Han
hoonhan.d@kaist.ac.kr
School of Computing, KAIST
Daejeon, South Korea

Juho Kim
juhokim@kaist.ac.kr
School of Computing, KAIST
Daejeon, South Korea

Alice Oh
alice.oh@kaist.edu
School of Computing, KAIST
Daejeon, South Korea

ABSTRACT

Many datasets for natural language processing are generated with crowdsourcing due to its low cost and scalability. However, in the datasets built with crowd workers' generated language, a problem called annotation artifacts arises; a model trained on such datasets learn annotators' writing strategies that are irrelevant to the task itself. Despite the increasing attention, little work dealt with the issue from the perspective of crowdsourcing workflow design. We suggest a simple but powerful adjustment to the dataset collection procedure: instruct workers not to use a word that is highly indicative of annotation artifacts. In the case study of natural language inference dataset construction, the results from two rounds of studies on Amazon's MTurk suggest that applying a word-level constraint reduces the annotation artifacts from the generated dataset by 9.2% in terms of accuracy–gap score at the time cost of 19.7s increase per unit task.

1 INTRODUCTION

In natural language processing (NLP), datasets are often generated to train a model that understands language for diverse tasks such as question answering [14] and identifying logical relationships [3, 18]. One of the widely used approaches when constructing such datasets is crowdsourcing, thanks to its low cost and scalability. In one type of crowdsourcing, workers are asked to write a textual statement to build the dataset when it is difficult to collect a text corpus that fits the purpose of the task [3, 11]. While crowdsourcing has gained popularity in dataset construction, a number of studies have reported that such human-elicited datasets have *annotation artifacts*, a type of dataset bias in which workers' strategies to generate data instances provide a task-irrelevant shortcut to correct prediction [6, 13, 15, 17].

For example, there is a NLP benchmark task named natural language inference (NLI), the goal of which is to correctly classify a pair of statements (so-called premise and hypothesis) according to their logical relationship: *entailment*, *neutral*, or *contradiction*. SNLI, the first large-scale dataset of NLI, was generated with crowdsourcing [3]. Due to the high cost required to collect a number of sentence pairs with clear logical relationships, crowd workers were prompted to write a statement (hypothesis) satisfying a logical relationship given a premise. Table 1 shows example sentence pairs in SNLI. One reported case of annotation artifacts in SNLI is the predominant frequency of negation words like *not* in the *contradiction* class, compared to the other two classes [6, 13]. The skewed distribution of a word over classes might give a clue about the correct answer, which likely causes a model to take the shortcut instead of learning logical relationships from the task. In fact, it has been discovered that the high accuracy of a few neural models is attributed to these annotation artifacts [6, 13, 17].

Premise	A greyhound with a muzzle runs on a racetrack.
Entailment	The dog is running.
Neutral	The greyhound is racing for the rabbit.
Contradiction	The dog is walking around the house.

Table 1: The example instances from SNLI dataset [3]. The goal of the task is to correctly classify the logical relationship given a pair of statements.

A number of approaches have been proposed to resolve annotation artifacts from the existing NLP datasets. Mostly, they suggest either altering the way that a model is trained [1, 2, 4, 7, 10] or augmenting the dataset with adversarial instances [5, 8, 12, 16]. However, research to date has tended to focus on the post-hoc solutions rather than improving the crowdsourcing workflow design. Without fixing the dataset generation scheme, this problem will repeatedly occur. The HCI community has actively introduced new crowdsourcing workflow designs for efficient, accurate, and fair data generation with the power of the crowd. In this research, we extend this line of work to tackle the problem of annotation artifacts from a crowdsourcing workflow design perspective. In order to improve the crowdsourcing workflow design so that a worker cannot use their own strategies contributing to the artifacts, we should understand what affects the annotation artifacts first. In this paper, we attempt to examine the impact of word-level lexical patterns in generation of annotation artifacts, which has not been explicitly investigated by previous research—to the best of our knowledge—despite its importance in any writing tasks.

We conducted a controlled study on Amazon's Mechanical Turk (MTurk) [1] to examine the impact of lexical patterns in generation of annotation artifacts. We recruited 15 unique workers for NLI data collection task in each of the two different conditions. In condition *Baseline*, we collected the data in almost the same way as SNLI was collected, while in condition *SW* (single-word), workers are instructed to include a given word when writing. The constraint word is chosen based on the data of condition *Baseline*, the semantic meaning of which is regarded as the least associated with the class. Interestingly, a model trained on the data of condition *SW* exhibits a significantly reduced degree of annotation artifacts compared to that of condition *Baseline* from 18.91% to 9.71% in terms of accuracy–gap score, the metric that we devised to measure the degree of annotation artifact. This result implies that certain words are inherently correlated with a specific class, thus

[1] https://www.mturk.com/

the class-specific lexical patterns can possibly cause annotation artifacts.

Despite the significant reduction of annotation artifacts by introducing a single-word constraint, another issue arises; condition *SW* takes about twice more time than *Baseline* on average. To understand the relationship between task design, data quality, and task time, we collect an additional set of data from 15 workers on MTurk, giving a choice between five words to include in text generation (instead of one) as a constraint. Considering the task designs from previous studies with different degree of freedom [3, 8], we discover that there exists a trade-off between task time and the degree of annotation artifact. This result indicates that with more degree of freedom given to workers, annotators leverage strategies so that annotation artifacts can deteriorate.

This research has the following core contributions:

- We provide evidence that the lexical patterns are atttributed to the presence of annotation aritfacts from the dataset collected by crowdsourcing with writing.
- We show that by simply adjusting the task design in that a worker must inclue a constraint word in their writings, annotation artifacts can be significantly reduced from the dataset. However, as a trade-off, the task time increases along with less degree of freedom given to the workers.
- We publish the experiment data [2] of 2.7k instances with validation results for replication and further investigation of workers' behaviors in different workflow designs.

2 STUDY 1

In this section, we describe our empirical study using MTurk to reveal the effect of lexical patterns on annotation artifacts and present the results of the study. We chose NLI as the domain of our empirical study. Our method differs substantially from previous research using model-based approaches to mitigating annotation artifacts. But the end goal is the same, so we compare our method with the model-based approaches.

2.1 Condition

2.1.1 Baseline. As a baseline for comparison with the proposed conditions, we collected the data on the data collection interface reconstructed based on the description of SNLI's [3].

2.1.2 Single-word constraint (SW). We hypothesize that the class-specific lexical patterns are a major cause of annotation artifacts, and this pattern is present even in a small NLI dataset. In this condition, we start with the data collected in the condition *Baseline*, and we instruct the user to include a specific word, the *constraint word* when writing the hypothesis sentences. Similar to previous work [6, 13], we use pointwise mutual information (PMI) to select the word. PMI given a word w and a class c is defined as follows:

$$PMI(w, c) = log \frac{p(w, c)}{p(w)p(c)}$$

This metric measures the degree of association between a word w and class c; the lower the PMI is, the less frequent the word w is used in the class c than other classes. A word which is used more than 10 times over all classes and of the lowest PMI value among the words in a class was selected.

[2] https://doi.org/10.6084/m9.figshare.12962480.v3

2.2 Data collection

For each condition, 15 unique participants on MTurk were recruited with the following qualifications: (1) residents of the U.S., (2) at least 500 HITs approved, and (3) HIT approval rate greater than 98%. The participants of the condition *Baseline* and *SW* were paid \$5 (\$12.57/hr) and \$6 (\$6.76/hr), respectively. A participant was not allowed to join a task of multiple conditions.

During the task, a participant was asked to write a hypothesis statement of each class given a premise. We used 15 premises for the experiment, which were randomly sampled from premises of SNLI dataset prior to the experiment. Following the sentence writing task, participants were asked to respond to a questionnaire designed to understand the task load and get feedback. After completing the data collection in each condition, we validated the data on MTurk to filter out invalid instances with majority voting. Annotators were paid \$0.05 per single annotation task where the median time was 9s.

2.3 Evaluation

Given an arbitrary dataset A, we first trained a hypothesis-only classifier—a classifier trained and tested only on the hypotheses—on the training set of A. We then measured the difference between the performance of the trained model and a random classifier on dataset B. The difference is regarded as the degree of annotation artifact present in A which also gives a clue about B. For convenience, we named this metric as *performance–gap of A on B* in this paper. This metric can capture the degree of annotation artifact in that when dataset A has annotation artifact that is also artifact in B, the performance–gap score would be significantly larger than zero by leveraging the artifact for testing on B. The more severe annotation artifact that two datasets share, the higher performance–gap score is achieved. Further, to measure the degree of annotation artifact in a single dataset, we slightly adjusted the metric and named it *performance–gap of A*, of which the only difference is that the classifier is tested on the test set of A instead of the whole set. In a similar manner, if dataset A has annotation artifact, the performance–gap score would be significantly larger than zero.

For evaluation, we used a model which consists of a single softmax classification layer on top of ALBERT-Base [9]. For statistical testing in the analyses, the two-tailed t-test is used.

2.4 Results

2.4.1 Annotation artifacts. Table 2 presents the degree of annotation artifact measured by the performance–gap scores.

Baseline vs. SW. The comparison between two conditions revealed that the annotation artifact was significantly reduced in condition *SW*. The accuracy–gap score of data in condition *SW* is significantly smaller than that of condition *Baseline* ($t(198) = 8.982, p \ll 0.01$), and the same holds for the F1–gap score ($t(198) = 9.652, p \ll 0.01$). Another observation is that the performance–gap scores of condition *Baseline* on *SW* and those of *SW* on *Baseline* are similar and have non-zero values. This indicates that datasets generated in both conditions share a certain amount of annotation artifacts.

(a) Accuracy–gap scores

A \ B	*Baseline*	*SW*	*MW*
Baseline	(18.91, 8.12)	(7.56, 4.70)	(7.09, 4.14)
SW	(6.72, 2.84)	(9.71, 6.23)	(9.98, 3.89)
MW	(7.18, 4.11)	(10.06, 4.92)	(11.06, 7.64)

(b) F1–gap scores

A \ B	*Baseline*	*SW*	*MW*
Baseline	(18.01, 10.22)	(5.81, 6.10)	(5.72, 6.10)
SW	(5.30, 3.64)	(6.33, 6.49)	(7.22, 4.59)
MW	(5.26, 6.06)	(5.39, 6.16)	(8.12, 8.55)

Table 2: The degree of annotation artifacts measured by performance–gap scores of A on B. The first and second value indicate the mean and standard deviation of performance–gap scores measured for 100 testings.

2.4.2 Task load. Following the addition of a word constraint, however, we find that the task of condition *SW* becomes significantly more difficult than *Baseline*. First, the overall time taken for a user to complete the task significantly increased in condition *SW* compared to the baseline. While participants of condition *Baseline* spent 23m 35s to write all 45 statements, it took 53m 15s for participants of condition *SW* to complete the task on average ($t(28) = -4.080, p \ll 0.01$). In addition, users' feedback on the task collected via the questionnaire supports our claim. Four among 15 participants in condition *SW* left comments that they sometimes felt it was impossible to write a sentence using a constraint word. These observations support the increment of difficulty and imply that it mostly comes from writing instances of *entailment* and *neutral* classes.

3 STUDY 2

While the annotation artifact is significantly reduced in Section 2, another problem arises; the task becomes too laborious. In fact, the constraint that requires a user to include a specific word in writings seems to be the major factor for the increased task load. Thus, by slightly increasing the degree of freedom given to users, we further explore the design space to investigate the relationship between the degree of annotation artifact, task time, and task design.

3.1 Condition

3.1.1 Multi-word constraint (MW). To give a less difficult restriction to the users than condition *SW*, we provided five words among which users can select one as their constraint word in condition *MW*. Similar to condition *SW*, the five words with the least PMI values are presented as candidate constraint words.

3.2 Data collection and Evaluation

The data for condition *MW* was collected in the same way as described in Section 2.2. The participants who had already joined the task of condition *Baseline* and *SW* were not allowed to participate in this task again. For data collection, we paid $6 ($9.35/hr) to a participant. For data validation, the reward was $0.05 per task, and the median time taken for a single annotation task was 8s.

3.3 Results

3.3.1 Annotation artifacts. Figure 1 briefly depicts the trade-off relationship between the degree of annotation artifact and the task time. As a reference, we included the counterfactually-augmented dataset published in [8] for our analysis. According to their results from the experiment with BiLSTM, annotation artifacts almost vanish in the dataset, while workers spent about four minutes for the unit crowdsourcing task of dataset augmentation. The peformance–gap scores of this dataset is ($M = 7.03, S.D. = 6.10$) and ($M = 6.13, S.D. = 7.00$), in terms of accuracy and F1-macro score, respectively. Thus, we consider the experiment result of this dataset as one extreme of the design space.

From Figure 1, we can interpret that while counterfactually–augmented data [8] succeeded at reducing annotation artifacts, the task design is too inefficient in terms of unit task time. The accuracy–gap ($t(198) = -1.365, p = 0.174$) and F1–gap ($t(198) = -1.669, p = 0.097$) scores are both increasing in condition *MW* with the higher degree of freedom to users than condition *SW*, while the increment of accuracy–gap is not statistically significant (Table 2). The observed increase of annotation artifact in condition *MW* than *SW* is possibly attributed to the selection bias, the workers' strategies to choose a word that is similar to the premise sentences, considering that the set of premises provided to users are identical.

3.3.2 Task load. The time taken for completing the task of condition *MW* is 38m 30s on average, which is significantly more than the task time of condition *Baseline* ($t(28) = -2.6207, p = 0.014$), while less compared to condition *SW* ($t(28) = 2.043, p = 0.051$). Also, two of 15 users explicitly left comments that they felt the task demanding. To sum up, as we expected with providing multiple options, users may find this task easier than the task of condition *SW*.

4 DISCUSSION

In this section, we discuss how annotation artifacts could be attributed to lexical patterns. Then, we discuss the trade-off between work load and dataset quality by controlling the degree of freedom. Furthermore, we suggest future work for crowdsourcing workflows for data generation.

4.1 Annotation artifacts can be attributed to lexical patterns.

The results of Study 1 (Table 2) revealed that annotation artifacts are significantly reduced in condition *SW* than condition *Baseline*, following the introduction of a word constraint. The possible explanation for this result is that the word-level constraints successfully prevented users from leveraging the word-level strategies, which signals that the generation of annotation artifacts is partly attributed to the lexical patterns.

Another interesting observation is that the performance–gap scores of condition *Baseline* on *SW* and that of *SW* on *Baseline* were found to be similar. The small difference between the two performance measures is likely to be related to the limitation of a word-level constraint; there can exist diverse levels of annotation artifacts, such as syntactic level, and the word-level intervention

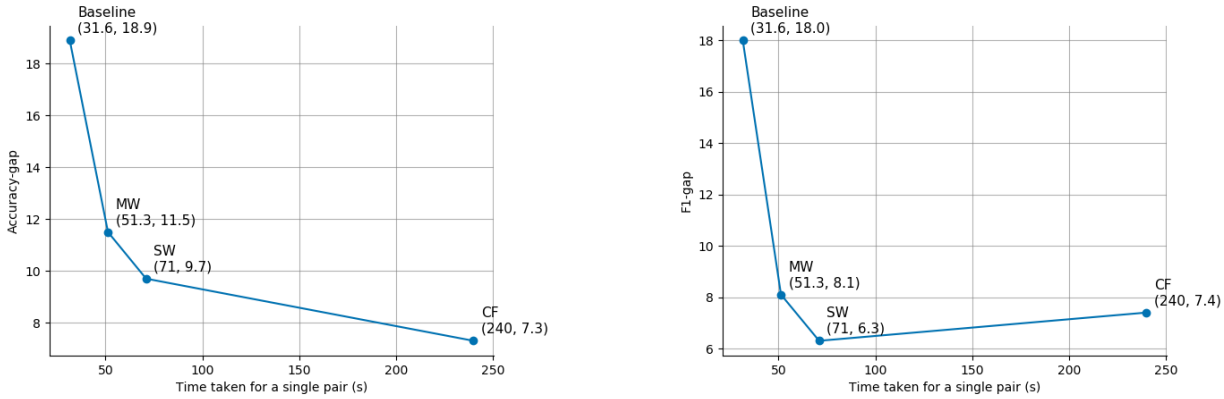

Figure 1: Two graphs showing the relationship between the time taken for a unit task and the degree of annotation artifact measured by performance–gap scores

to the workflow is not sufficient to fully eliminate the annotation artifact from the dataset. Thus, further research could investigate the diverse factors influencing annotation artifacts and the diverse types of annotation artifacts.

4.2 Controlling the degree of freedom leads to the trade-off between task load and dataset quality.

The results of study 2 (Figure 1) reveal the trade-off that the higher degree of freedom decreases the task load at the cost of an increase in annotation artifacts present in the dataset. However, controlling the degree of freedom in a more sophisticated manner could lead to a more optimized solution in terms of dataset quality and task load. The participants in condition *SW* explicitly mentioned that writing an entailment is especially difficult compared to writing the other two classes. Based on these observations, we can maintain the degree of freedom in other classes while increasing the degree for only the *entailment* class. As such, we can achieve a more optimized task design with more sophisticated control of degree of freedom.

4.3 Future work

The results of study 2 (Figure 1) suggest the possibility of designing a data collection task to reduce the annotation artifact while workers are continuously incentivized to contribute. On top of the focus of this work on the impact of word-level constraints, further research could examine the role of more diverse degress of variation on task design into annotation artifacts.

Objective function. In this research, a constraint word is selected based on the PMI value which measures the extent to how skewed the usage of a word is over classes. We chose this metric as we believed that the distorted distribution of a word can act as an indicator of annotation artifact. Assuming a dataset designer puts emphasis on the diversity in the dataset among other things, one can try another objective function to choose a word that is located far from the sentence on the embedding space. As dataset construction inevitably includes value-laden decisions, investigating different objective functions and their impact could present valuable insights.

Degree of freedom. We can control the degree of freedom in more diverse dimensions such as financial compensation and time. For example, from the task design of condition *MW*, a designer may not want to harm the degree of freedom while hoping the annotator to choose a specific word from the provided options. Although this is almost impossible in our experiment setting, one can adjust the task design in practice by distinguishing the amount of monetary reward according to the preference and needs.

Constraint type. While we chose to focus on the word-level constraint regarding its applicability over various domains, another type of constraints such as syntactic patterns can be considered. Considering the previous study that a neural model of NLI exhibits poor performance on several syntactic heuristics, we speculate that a certain type of syntactic patterns in the dataset can be attributed to annotation artifacts. As such, considering the diverse candidates that possibly affect the generation of the annotation artifact, our study design can be adopted to investigate the role of particular factors by adjusting the type of constraint.

5 CONCLUSION

In this paper, we investigated the impact of word–level constraints in generation of annotation artifact. Interestingly, the single–word constraint into the crowdsourcing workflow succeeded at reducing annotation artifact while writing task becomes too laborious. The second study with multi–word constraint revealed the trade–off between annotation artifact and task time.

The present study lays the groundwork for future research into the relationship between crowdsourcing task design and annotation artifact. However, we believe that there is still abundant room for further investigation on this issue, as suggested in Future work (Section 4.3). We expect that these studies can ultimately remove the annotation artifact, thus the model trained on the dataset can learn the task itself, not a shortcut. In addition, we would like to argue the importance of crowdsourcing workflow design for dataset generation, as the generated dataset might negatively affect the reliability of model's performance.

REFERENCES

[1] Yonatan Belinkov, Adam Poliak, Stuart M Shieber, Benjamin Van Durme, and Alexander M Rush. 2019. Don't Take the Premise for Granted: Mitigating Artifacts in Natural Language Inference. *arXiv preprint arXiv:1907.04380* (2019).

[2] Yonatan Belinkov, Adam Poliak, Stuart M Shieber, Benjamin Van Durme, and Alexander M Rush. 2019. On adversarial removal of hypothesis-only bias in natural language inference. *arXiv preprint arXiv:1907.04389* (2019).

[3] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326* (2015).

[4] Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't Take the Easy Way Out: Ensemble Based Methods for Avoiding Known Dataset Biases. *arXiv preprint arXiv:1909.03683* (2019).

[5] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 6904–6913.

[6] 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* (2018).

[7] He He, Sheng Zha, and Haohan Wang. 2019. Unlearn dataset bias in natural language inference by fitting the residual. *arXiv preprint arXiv:1908.10763* (2019).

[8] Divyansh Kaushik, Eduard Hovy, and Zachary C Lipton. 2019. Learning the difference that makes a difference with counterfactually-augmented data. *arXiv preprint arXiv:1909.12434* (2019).

[9] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942* (2019).

[10] Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. 2020. End-to-End Bias Mitigation by Modelling Biases in Corpora. ACL.

[11] Fabrizio Morbini, Eric Forbell, and Kenji Sagae. 2014. Improving classification-based natural language understanding with non-expert annotation. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*. 69–73.

[12] Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2019. Adversarial nli: A new benchmark for natural language understanding. *arXiv preprint arXiv:1910.14599* (2019).

[13] 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* (2018).

[14] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250* (2016).

[15] Roy Schwartz, Maarten Sap, Ioannis Konstas, Leila Zilles, Yejin Choi, and Noah A Smith. 2017. Story cloze task: Uw nlp system. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*. 52–55.

[16] Rishi Sharma, James Allen, Omid Bakhshandeh, and Nasrin Mostafazadeh. 2018. Tackling the story ending biases in the story cloze test. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 752–757.

[17] Masatoshi Tsuchiya. 2018. Performance impact caused by hidden bias of training data for recognizing textual entailment. *arXiv preprint arXiv:1804.08117* (2018).

[18] Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426* (2017).