APEACH: Attacking Pejorative Expressions with Analysis on Crowd-Generated Hate Speech Evaluation Datasets

Anonymous ACL submission

Abstract

Detecting toxic or pejorative expressions in online communities has become one of the main concerns for preventing the users' mental harm. This led to the development of large-scale hate speech detection datasets of various domains, which are mainly built upon web-crawled texts with labels by crowdworkers. However, for languages other than English, researchers might have to rely on only a small-sized corpus due to the lack of data-driven research of hate speech detection. This sometimes misleads the evaluation of prevalently used pretrained language models (PLMs) such as BERT, given that PLMs often share the domain of pretraining corpus with the evaluation set, resulting in over-representation of the detection performance. Also, the scope of pejorative expressions might be restricted if the dataset is built on a single domain text.

To alleviate the above problems in Korean hate speech detection, we propose APEACH, a method that allows the collection of hate speech generated by unspecified users. By controlling the crowd-generation of hate speech and adding only a minimum post-labeling, we create a corpus that enables the generalizable evaluation of hate speech detection regarding text domain and topic. We compare our outcome with prior work on an annotation-based toxic news comment dataset using publicly available PLMs. We check that our dataset is less sensitive to the lexical overlap between the evaluation set and pretraining corpus of PLMs, showing that it helps mitigate the unexpected under/over-representation of model performance. We distribute our dataset publicly online to further facilitate the general-domain hate speech detection in Korean.

1 Introduction

Detecting toxic or pejorative expressions has been a crucial issue in various online communities. [1] In particular, flaming or trolling in online communities is regarded as hostile behavior that can disrupt the public order and cause mental harm to individuals and groups. Attempts to define and detect hate speech from a natural language processing (NLP) perspective have called for timely works, and notable approaches have been suggested so far. In specific, Waseem and Hovy (2016) primarily attacked the judgment on hate speech for Twitter text, and Davidson et al. (2017) further investigated the offensiveness of social media texts beyond binary detection. Recently, Huang et al. (2020) have suggested how demographic matters in hate speech analysis, for the corpora of five different languages, discerning the multilingual tendency.

Creating a hate speech dataset generally involves annotating short documents such as web text, and the context where the hate expressions are from may or may not be given in the process. However, annotating on existing web text has several limitations that deter the dataset's reliability. First, the corpus may incorporate the potential risk of license and personally identifiable information issues that come from the characteristics of online materials. Next, if the text is crawled from a restricted scope of domain, the topics of examples may not be diverse, which can result in the evaluation that focuses only a part of social issues (e.g., gender). Moreover, in view of model training, using a specific domain of web text that might have been used as a pretraining corpus of public language models, may intervene the fair competition between models and mislead the evaluation. Such tendency is more apparent in non-English hate speech de-

[1] Though toxic or pejorative expressions are distinguished with 'hate speech', a more political term, here we interchangeably use both sides of terms for our description and alignment with previous works.

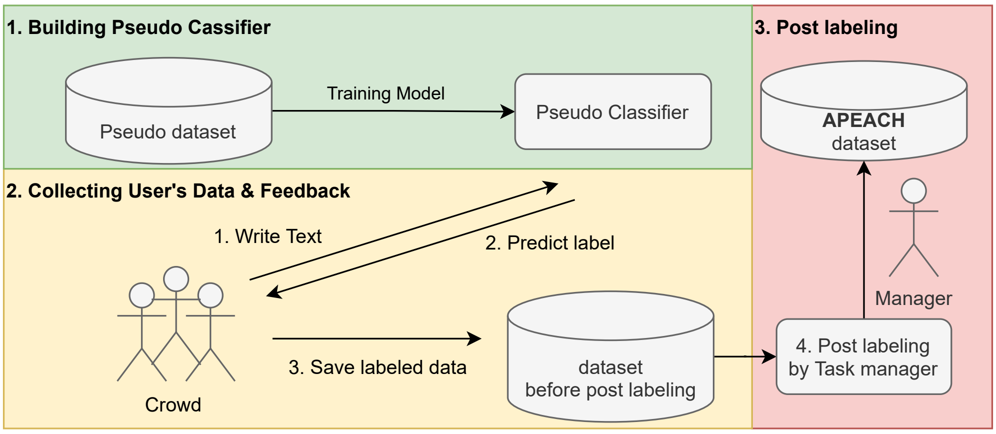

Figure 1: Overall schematic process of the proposed system (APEACH).

tection tasks where the training and evaluation of language models rely upon only a small number of benchmarks. For instance, in Korean, the usual performance checking of pretrained language models (PLMs) adopts BEEP! (Moon et al., 2020), a currently available hand-labeled hate speech dataset in Korean where the domain of raw text is celebrity news comment. People found out that the PLMs trained upon news comments perform better than others which base on news or Wikipedia, but it throws a question to the generalizability and fairness of the evaluation.

How can we address such limitation of annotation-based web text corpus construction? Though collecting a variety of data as in multi-genre natural language inference (NLI) (Williams et al., 2018) might be the most intuitive solution, it requires economic and human resources and still does not resolve the issue of corpus overlap. In this regard, we hypothesized it a reasonable approach to let anonymous paid workers generate toxic expressions, where the expressions are generated from scratch with a minimum guideline.

At a glance, simply opening a web page for text collecting and encouraging user participation seemed to work. However, we noted that neither the data quality nor the open license of the output could be guaranteed in those processes. Thus, we established a crowd-driven hate speech generation scheme using a moderator to ensure the privacy of hate speech authors and obtain a quality-checked corpus at the same time. In specific, we adopt crowd-sourcing platform (as a moderator) and workers for the paid writing, provided with the prompts to guide the generation, to achieve diverse hate speech and prevent participants' disgrace. For the facilitation of text generation and collection process in our research, we devise '**System**', an environment that interacts with the **crowd** (who provides the data) and the **task manager** (who collects the data), which is composed of i) building a hate speech pseudo-classifier and deploying the model, ii) collecting user-generated data and feedback, and iii) post-labeling of the task managers (Figure 1).

Followingly, we obtain **APEACH**, which denotes the collecting scheme and the resulting dataset at the same time. It contains about 3K instances for Korean hate speech detection evaluation; the corpus is well balanced in sentence length and topics, also being aligned with the models trained with the existing hate speech dataset (BEEP!). Most of all, by comparing the model performances of our dataset using publicly available PLMs where the pretraining corpus overlap with BEEP!, we prove that APEACH is less vulnerable to misleading results that might come from the corpus-level similarity with the pretraining corpus.

Our contribution to this field is as follows:

- We present a scheme that collects user-generated hate speech from scratch without undertaking conventional annotation process.
- We build a new evaluation set for Korean hate speech detection that is free from license and privacy issues, and release it for community contribution. [2]
- We conduct a model-based comparison with another human-annotated hate speech benchmark, showing that the generalizability of the

[2] Will be disclosed after the anonymity period.

proposed evaluation set is implied from less overlap with specific pretraining corpus.

2 System

Our system consists of three processes: 1) building pseudo-classifier, 2) collecting users' data & feedback, and 3) post-labeling. First two are to be described in this section.

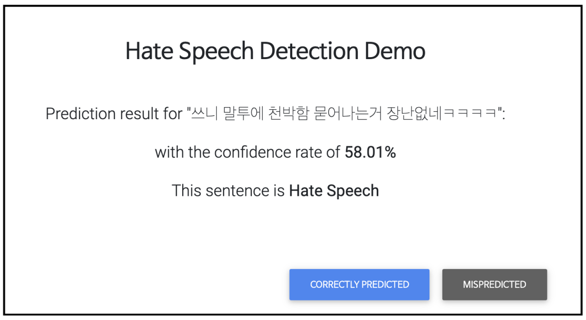

Figure 2: A screen in which users enter input sentences to the deployed model and check the predicted result. At the bottom right is a screen where they can select whether the predicted result is correct (blue) or not (grey). The translation for the input sentence is: "*what a salty! lol* ".

2.1 Building a Pseudo-Classifier

To build the system from scratch, we deploy a pseudo-classifier to compensate for the crowd's loss of concentration coming from repetitive tasks and let them participate actively in the collection process. For this, we primarily created a pseudo-labeled dataset to train the classifier, not for evaluation.As usual hate speech detectors, the pseudo-classifier receives the user-generated text as an input and predicts whether the text incorporates bias or toxicity. However, since this is not for real service but a tool for participants' confirmation on the label of their work, we trained the classifier that displays only a basic performance. We simply create a dictionary of profanity terms to obtain a pseudo-labeled web text dataset, and use it to train a simple binary classifier. The details of dataset construction and model selection are provided in Appendix A.

2.2 Deployment and Text Collection

The pseudo-classifier is deployed to a server to collect user input and feedback. As in Figure 1, when the user enters the test input, the predicted label is displayed. The user further determines whether the corresponding label equals the user's original intention, namely whether it is hate speech or not. The user interface (UI) for the prediction and feedback is exhibited in Figure 2.

In specific, in the user feedback phase, two buttons are provided, namely "correctly predicted" and "mispredicted" (Figure 2). If "correctly predicted", the prediction is saved along with the user input as the ground truth. In the case of "mispredicted", the ground truth is saved as a reversed version of the prediction.

3 Dataset

Using the above system, we construct an evaluation dataset with the user-generated data that is inspected with a model prediction.

3.1 Prompts for Text Generation

In general, hate speech includes flamings observed in the web communities, namely the threatening expressions that are represented as text or even in some multi-modal format (Kiela et al., 2020). They can be expressed in hostile or discriminating words. However, letting the participants merely generate the hate speech from scratch might be challenging and misleading.

Topic For effective and efficient data collection, we provide the participants with criteria on various topics of hate speech. We set ten topics that the participants can refer to in generating the text, which is inspired by the code of conduct (COC) of PyCon KR. [3] Each of the topics denotes the main attribute of the hate speech that is to be generated.

1. Behaviors based on gender stereotypes
2. Discrimination or demeaning jokes with one's sexual orientation or identity
3. Discrimination or stereotypes on age, social status, or experience
4. Discrimination based on nationality/ethnicity
5. Racial discrimination
6. Discrimination based on origin or residence
7. Unnecessary or offensive judgments on one's appearance
8. Demeaning or offensive words for illness or disability
9. Forcing or depreciating with eating habits
10. Rude or discriminatory remarks based on others' academic background or major

As shown in Figure 3, workers generate text by selecting one of the topics above. Prompts regarding

[3] `https://www.pycon.kr/en/2020/about/coc/`

Figure 3: Web interface utilized in the crowd-sourcing process.

the topic of hate speech are provided in a dropdown format, while the order of topics is randomly shuffled for each input. Workers enter the input after selecting one of them.

Label We define whether a sentence contains hate expression or not as "label". Besides the topic, the workers are assigned with either they should generate a sentence that contains hate speech or not. The latter case denotes a neutral or seemingly controversial utterance that only shares the topic with hate speeches but is not offensive; e.g., "*I hate those who demeans BLM movements.*" for the topic 'Racial discrimination'. In our study, the hate speech (positive sample) and non-hate speech (negative sample) serve as an element of the balanced dataset for the detection task.

3.2 Post Labeling

For each user input, the pseudo-classifier yields the prediction. For the assigned label, the user input is confirmed based on the user's feedback. For instance, when the assigned label is hate speech, and if the model yields 'non-hate speech', the user may check "mispredicted" and the ground truth is saved as "hate speech". In this process, if the assigned label differs from the saved ground truth, we conclude that the user mislabeled or misunderstood the label, and automatically remove the instance. [4]

We faced some questionable instances that came from the diverse ethical standards of the participants. However, to guarantee the characteristics of the crowd-generated hate speech dataset, such erroneous cases were checked with the minor engagement of task managers. We call this process *post-labeling*, which ensures the quality of the dataset by applying a conventional annotation and voting process in the final decision. In specific, three task managers, who are speakers of contemporary Korean, checked if the user's feedback was appropriate for each instance. Since we regard the user's choice as ground truth, we only dropped the instances that all of the three task managers found irrelevant with the assigned label.

3.3 Dataset Collection

As discussed, the most intuitive way for construction is to collect text inputs from an unspecified crowd using an online platform, e.g., a demo page. Hate speech detector is itself a great contribution to the community, thus getting user feedback with a closed or open beta service is not an unnatural choice, in view of both research progress and industrial development.

However, such an approach incorporates some critical issues on data quality and privacy, which originates in the characteristics of hate speech text. It is widely known that the reliability of a generated corpus is not usually guaranteed if there is a lack of time or budget that it takes in creating the dataset. We guessed that people might not reveal their identity to receive compensation for their toxic expressions, mainly due to the fear of being recognized as a politically incorrect person. To confirm that our guess is correct, a web-based pilot resulted in the collection of text with degraded quality, sometimes violating the license issues or containing personally identifiable information. [5] Also, such an approach might not be approved by research communities and institutional review board (IRB). In order to cope with the limitation of unspecified user-generated hate speech collection discussed above, we create a dataset leveraging the worker pool of the crowd-sourcing platform, while taking the same user generation guideline

[4] Note that this process is a simple but strong checkpoint for the robustness of the labels in the dataset, leveraging the possible human error of the user feedback system (Stumpf et al., 2007). Given that the participants are aware of the pseudo-classification, the system allows them to deem the toxicity of the contents they generate.

[5] We want to point out that the pilot only provided us with motivation, and was not a part of our experiment. The pilot demo was organized in the very first phase of our project to check how our pseudo-classifier deals with real-world toxic expressions, on a publicly accessible web site, to found out that undisclosed crowds show uncontrollable behaviors. We became aware of legal and ethical standards that are required in hate speech dataset creation, and did not collect or store any logs or data of user input queries (nor they were used in our experiments or analyses). Our final dataset is based on the manual generation of paid workers, not a web-based collection.

and post-labeling scheme.

Compensation through moderator Workers must be identified for the compensation of the hate speech generation, but it may harm the anonymity of the collection phase and eventually affect the natural text generation. Therefore, we enable the crowd-sourcing platform to play a role as a **moderator** between the task managers and workers. In other words, the project is designed in the way that only the moderator manages the workers' profiles, preventing them from being known to the task managers. In this way, we can accommodate both compensation and anonymity for the workers.

Worker selection for dataset quality One of the aims of the evaluation set we construct is to reflect the diversity of contents as much as possible. However, not all the paid workers of the crowd-sourcing platform are qualified for our project. Thus, we had a tutorial to prevent the low-quality generation which might take place in unconstrained user data collections. In detail, we receive ten inputs per worker and count the portion of mislabeled instances to drop the workers with frequent faults. Also, we checked the worker's sincerity with i) if the input is longer than a single character and ii) if the input does not replicate the examples in the guideline. 154 out of 230 workers were finally admitted for participation in the main construction.

Diversity of crowd-generated hate speech In contrast to the anonymous collection where the task managers find it difficult to ask the participants to sincerely generate various topics of texts owing to the lack of compensation, the proposed scheme helps attenuate such limitations with the utilization of topic prompt. It guides the text generation of the workers and helps collect non-hate speech that is less considered in the previous hate speech literature. In addition, to prevent the contents from being biased due to some heavy workers, we let the text generation be a maximum of 40 per participant.

3.4 Dataset Summary

Our construction scheme can simultaneously guarantee data quality, topic variety/distribution, and ethical consideration of crowd-generated hate speech, with the crowd-sourcing platform as a moderator. Sentences are aggregated from the pilot and main collection phase. In the pilot, instances with clear disagreements were not selected for further project, and in the main phase, disagreements among task managers were re-labeled as an opposite class. We provide the detailed information on agreement in Appendix B.

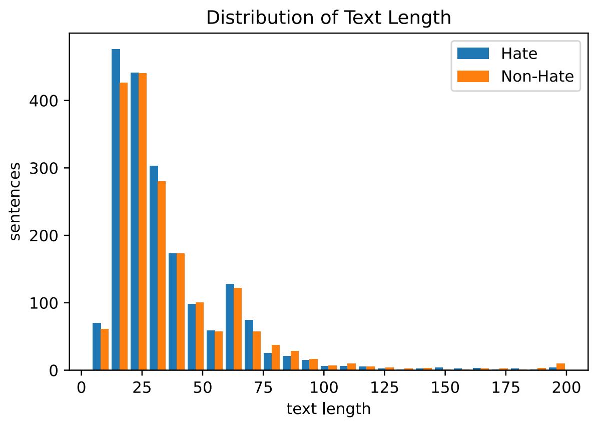

Figure 4: Distribution of text by length and label.

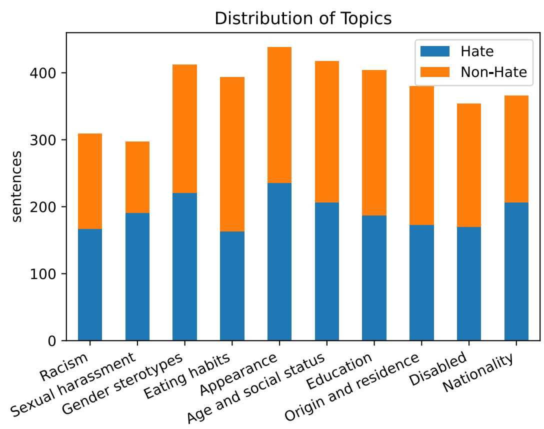

Figure 5: Topic/label distribution

Length distribution In Figure 4, similar length distribution is displayed between hate speech and non-hate speech in APEACH. This suggests that our construction scheme can prevent the biased distribution of the length of hate and non-hate speech. It enables us to further investigate if the hate speech detection model has handled the inductive bias coming from such distribution.

Distribution of topics By shuffling the order of topic prompts per every input, we prevent the bias which comes from the tendency that people habitually select the top candidate in the dropdown interface. In addition, we also confirmed that the two labels are evenly distributed in the dataset by assigning hate speech and non-hate speech in advance in the collection phase (Figure 5).

4 Experiment

We exploit our corpus to evaluate the hate speech detection models trained with a widely used Korean hate speech benchmark, BEEP! (Moon et al., 2020). In specific, we compare APEACH (ours) and BEEP! dev set as an evaluation corpus, to check the generalizability and performance tendency using each set.

4.1 Korean Pretrained Language Models

We adopt publicly available Korean pretrained language models for the reproducibility. The characteristics of each model are as follows:

- **KoBERT** [6], **DistilKoBERT** [7]: KoBERT is a PLM that follows the BERT (Devlin et al., 2019) training scheme, a self-supervised learning scheme based on Transformers (Vaswani et al., 2017) architecture, with Korean Wikipedia data. In addition, DistilKoBERT is a light-weighted version of KoBERT using distillation, adopting Huggingface's DistilBERT (Sanh et al., 2019) model.
- **KoELECTRA** [8]: An ELECTRA (Clark et al., 2020)-based Korean language model pretrained with 'Modu Corpus' released by the National Institute of Korean Language (NIKL) (National Institute of Korean Languages, 2020), Korean Wikipedia, NamuWiki [9], and various news articles.
- **KcBERT** (Lee, 2020) [10]: A Korean BERT model trained with 12GB of NAVER politics news comments. [11]
- **SoongsilBERT** [12]: In addition to the news comments data used by KcBERT, SoongsilBERT utilizes college community data and Modu Corpus.

The architecture and fine-tuning configuration of each PLM are provided in Appendix C.

[6] https://github.com/SKTBrain/KoBERT
[7] https://github.com/monologg/DistilKoBERT
[8] https://github.com/monologg/KoELECTRA
[9] A Large-scale Korean open encyclopedia. https://namu.wiki/
[10] https://github.com/Beomi/KcBERT
[11] https://bit.ly/3o1d7lk
[12] https://github.com/jason9693/Soongsil-BERT

4.2 Training Data

We used BEEP! training set for fine-tuning the above PLMs. In detail, BEEP! is a human-annotated corpus where the intensity of hate speech is tagged with the labels of 'hate', 'offensive', and 'none', built upon celebrity news comments on a Korean online news platform. The instances with 'hate' labels include hostile expressions, stigmatization, or sexual harassment, and 'offensive' instances include sarcastic or inhumane expressions.

Although the construction scheme of train and dev set of BEEP! differs from our dataset, we want to compare the tendency between each set regarding hate speech detection models, utilizing both datasets. Nevertheless, since APEACH merely suffices the scale of an evaluation corpus, we first fine-tune the models with BEEP! train set.

4.3 Evaluation

We formulate BEEP! dev set and APEACH as both binary classification using F1 scores. For APEACH, labels regarding hate and non-hate speech serve as positive and negative samples, respectively. In contrast, since BEEP! was initially formulated as a ternary task, we reformulate it into 'hate'+'offensive' and 'none' for the consistency of the binary setting.

4.4 Results

The model-wise experimental results using BEEP! and APEACH are in Table 1. It was encouraging that the models trained with BEEP! training set shows reasonable performance even in our dataset, which implies that our criteria for dataset generation are largely aligned with the existing work.

Influence of corpus domain In the case of BEEP!, KcBERT-Large displays the highest performance, while in APEACH, KoELECTRA which generally scores lower in BEEP! shows almost the same performance as KcBERT. This implies that the style and domain of the dominant corpus used for pretraining of each PLM influence the downstream task performance.

Performance per topic We observed the deviation of the inference accuracy by ten topics presented in the guideline (Table 2). This deviation seems to come from the difference between the construction scheme of the training corpus and the evaluation corpus. In detail, providing the random order of prompts while generating the hate speech

Model	BEEP! dev set	APEACH (ours)	Relative difference
KoBERT	0.8030	0.7885	-1.81%
DistillKoBERT	0.7570	0.7715	1.92%
KoELECTRA-V3	0.7920	0.8101	2.29%
KcBERT-Base	0.8088	0.8086	-0.02%
KcBERT-Large	**0.8295**	0.8116	-2.16%
SoongsilBERT-Base	0.8261	**0.8424**	1.97%
SoongsilBERT-Small	0.8149	0.8228	0.97%
Composition	Hate + offensive : 311 None : 160	Hate : 1,922 Non-hate : 1,848	

Table 1: F1 score of binary classification performance for each model. The fine-tuning was conducted with the Beep! train set. At the bottom, we provide the number of sentences according to the labels of each dataset.

Topic	F1 Score
Nationality	0.8519
Age and social status	0.8700
Eating habits	0.8182
Appearance	0.8114
Gender stereotypes	0.7993
Sexual harassment	0.7610
Racism	0.8511
Origin and residence	0.8393
Disabled	0.8525
Education	0.9035

Table 2: SoongsilBERT-Base's F1 score of binary classification according to topics.

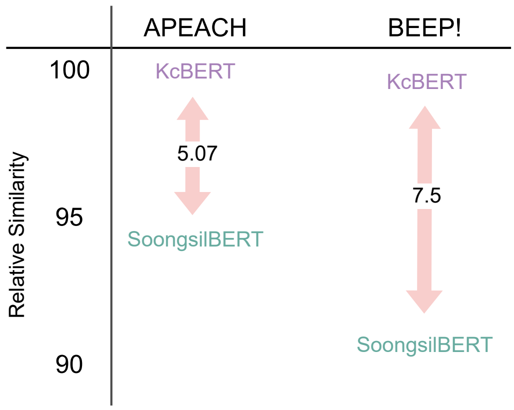

Figure 6: Averaged TF-IDF cosine similarity between the evaluation datasets and PLM training corpora. We calculated four scores using two pretraining corpora and two evaluation datasets. Four scores are normalized by the maximum value. TF-IDF vectors are generated with the pretraining corpus of KcBERT and Soongsil-BERT (provided by the authors), and the dev set of BEEP! and APEACH.

data yielded diverse topics which are difficult to obtain unless the annotation corpora are collected from multiple web communities. We did not yet exactly find why F1 scores regarding gender stereotypes and sexual harassment are lower than other categories despite the dominance of gender-related instances in BEEP! corpus. One possibility is that the style of text regarding gender and sexuality differs much from that of BEEP!, yielding discrepancy between train and evaluation set. Ironically, this is one evidence of the domain coverage of our dataset not only regarding topics but also style, as to be investigated in the following section.

5 Analysis

5.1 Domain Generalizability

As discussed above, APEACH tackles domain dependency issues of annotation-based corpus construction scheme by i) letting the crowd generate the hate speech based on prompts and ii) not specifying the style of text that is created. These two are difficult to be guaranteed by annotating crawled web data from just several communities, and it is a current limitation of BEEP! which serves as a unique hate speech benchmark for Korean.

In detail, for BEEP! dev set where the text comes from the news comments domain, KoELECTRA, which lacks news comments in pretraining, shows relatively lower performance, while the tendency differs in APEACH. This suggests that APEACH allows the investigation of the performance of fine-tuned hate speech detection models with less dependency on the domain of the corpus used for pretraining. Besides, for SoongsilBERT, a model trained with an augmented corpus upon the KcBERT's, the tendency for BEEP! and APEACH differs, implying that the domain-specificity of BEEP! dev set might have over-represented the advantage of KcBERT on news comment text. In other words, the better performance of KcBERT in BEEP! shows that crawl and annotation-based dataset construction can bring dependency on specific domains, which acts as a limitation in the domain generalizability in the evaluation.

Quantification of domain generalizability To quantify these ideas, we validated the domain generalizability of each evaluation set by calculating the TF-IDF similarity between each set and the PLM pretraining corpus (Figure 6). Here, the TF-IDF word dictionary is built upon all whitespaced words of four corpus (APEACH, BEEP! dev set, KcBERT pretraining corpus, and SoongsilBERT pretraining corpus), and 1% of KcBERT and SoongsilBERT pretraining corpus are randomly sampled for the feasibility of computing cosine similarity between the evaluation set instances. As a result, in the BEEP! dev set, the performance gap between KcBERT and SoongsilBERT is significant, whereas in APEACH, the gap is relatively less displayed. This implies that APEACH is less sensitive to the overlap with PLM pretraining corpus, suggesting the generalizability of the evaluation.

Qualitative analysis Among APEACH, we investigated the toxic examples regarding 'Gender stereotypes' and 'Sexual harassment', which were observed most significant among BEEP! train set. Two subtypes of texts were excerpted, namely the web comment style (that resembles BEEP!) and other formal texts. WARNING: This part contains contents that may offend the readers.

1. ㅅㅂ옷 꼬라지 하고는.. 게이냐? (*Wtf what a fashion... are you a gay?*)
2. 여자 연예인들은 다들 뒤로 스폰끼고 구멍장사하는거 아닌가? (*Isn't it a routine for idol girls to have a sugar daddy and fuck around?*)
3. 아무래도 아이를 돌보거나 그 밖의 집안일에 관해서는 남성들보다는 여성들이 좀 더 신경 써야 하는 것이 당연하다고 생각합니다. (*I think it's natural for women to pay more attention to caring for their children or other household chores than men.*)
4. 남자면 그냥 해도 될텐데 고집을 많이 부리네요. 불편해도 좀 참아야 다들 편한데 (*If you are a man, you can just do it; you have so much words. Even if it's uncomfortable, just endure it.*)

We checked that the SoongsilBERT-Base trained upon BEEP! train set correctly infers the toxicity of sentence (1-2) but fails to detect the harm of (3-4), where the stereotype is implicated in a polite and formal manner. This shows how the constructed dataset helps domain-generalized evaluation of hate speech detection, compared to the previous approach which adopts a single-domain text.

5.2 Mitigating Train-Test Overlap

Regarding contents, KcBERT was pretrained based on large-scale politics news comments. BEEP! deals mainly with celebrity news comments, not politics, but shares a similar domain. Therefore, a potential token overlap exists between KcBERT's pretraining corpus and BEEP!'s train/dev set, as we checked previously. This seems to boost the score of KcBERT significantly when evaluated with the BEEP! dev set.

We attempt to mitigate this in APEACH, which contains only crowd-generated thus unique utterances, preventing the over-representation of KcBERT shown in BEEP! dev set. APEACH guarantees such generalizability by generating text in a free-style manner based on topic prompts. Accordingly, we confirm that it fits with the evaluation of PLMs pretrained with the wider range of corpus (SoongsilBERT), compared to the BEEP! dev set. Through this, we tackle again the risk of train-test overlap for the hate speech data constructed with crawling and annotation, and emphasize that APEACH mitigates this issue. This property also guarantees the utility of APEACH as a training set, and we add corresponding results in Appendix D.

6 Conclusion

In this work, we introduce a crowd-driven generation scheme in constructing an evaluation set for hate speech detection, distinct from the existing corpus construction schemes based on crawling and annotation. After a managed human text generation that ensures both the participants' anonymity and the reliability of the corpus, we report a thorough analysis of the created data, accompanied by a comparison with the prior work in Korean. The resulting corpus, APEACH, displays the potential of adopting crowd-driven generation in hate speech dataset construction, achieving generalizability and topic variety. Though there is headroom for the scalability of the corpus, we believe that the proposed scheme can be utilized to make up the evaluation set and training set of domain-agnostic hate speech detection.

Ethical Consideration and Societal Implications

Our study aims at the construction of hate speech corpus distinguished from the conventional scheme of crawling and annotation. This not only lessens the annotators' mental damage, which is probable in reading other people's toxic comments, but also mitigates the potential issue of license and privacy in distributing the corpus. First, we obtain the texts written by 'workers' acknowledged by the moderator, not unknown 'users', to make an appropriate compensation (≈$ 0.2 per sentence) and encourage the high-quality generation. Second, we guarantee anonymity but accept only qualified people, to prevent the case that the text is copy and pasted from other sources. Last, by ordering the omission of personally identifiable information in the generation process, we avoid the danger of information leaks in the final dataset. Our study is approved by institutional review board, where the specification is to be revealed after acceptance.

Our work incorporates several limitations and potential harms as well. First, using our scheme does not guarantee the hate speech dataset that satisfies everybody, since the intuition of workers differs significantly across various groups of people. Also, though our workers are selected after a pilot study, they may not be fully equipped with the ethical guideline. Thus, their decision might not always be ideal, which brings the degradation of reliability to the final label. Last, our binary scheme for hate speech lacks score-based decision and span notation which are up-to-date in the hate speech community, providing only hard-labeled instances of anonymous workers. However, we think the strength of our dataset is in initiating a generation-based hate speech detection corpus that allows crowd participation with lessened privacy and license concerns. We also want to state that the intended use of our dataset is to evaluate pretrained language models' ability to detect toxic language, with less dependency on the type of pretraining corpora, the domain of text, and the topic and length of sentences.

References

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, ICWSM '17, pages 512–515.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Homa Hosseinmardi, Sabrina Arredondo Mattson, Rahat Ibn Rafiq, Richard Han, Qin Lv, and Shivakant Mishra. 2015. Detection of cyberbullying incidents on the instagram social network. *arXiv preprint arXiv:1503.03909*.

Xiaolei Huang, Linzi Xing, Franck Dernoncourt, and Michael J. Paul. 2020. Multilingual Twitter corpus and baselines for evaluating demographic bias in hate speech recognition. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1440–1448, Marseille, France. European Language Resources Association.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. In *Advances in Neural Information Processing Systems*, volume 33, pages 2611–2624. Curran Associates, Inc.

NIKL National Institute of Korean Languages. 2020. NIKL CORPORA 2020 (v.1.0).

Junbum Lee. 2020. Kcbert: Korean comments bert. In *Proceedings of the 32nd Annual Conference on Human and Cognitive Language Technology*, pages 437–440.

Jihyung Moon, Won Ik Cho, and Junbum Lee. 2020. BEEP! Korean corpus of online news comments for toxic speech detection. In *Proceedings of the Eighth International Workshop on Natural Language Processing for Social Media*, pages 25–31, Online. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Simone Stumpf, Vidya Rajaram, Lida Li, Margaret Burnett, Thomas Dietterich, Erin Sullivan, Russell Drummond, and Jonathan Herlocker. 2007. Toward harnessing user feedback for machine learning. In *Proceedings of the 12th international conference on Intelligent user interfaces*, pages 82–91.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*, 30:5998–6008.

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

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

A Pseudo Classifier

A.1 Dataset

We use the texts of an online community (YourSSU) [13] and news comments for the tagging of pseudo-gold labels. About 500,000 sentences were finally collected and were pre-processed with, e.g., removing duplication. The labeling policy of the dataset partly follows Hosseinmardi et al. (2015). We initially constructed a dictionary made up of about 200 profanity terms (toxic words), and labeled a text as "Hate" if it contains any of the listed terms. In specific, the toxic words were chosen referring to the criteria on politics/sexual/racism/religion, from the COC in PyCon KR. Profanity terms that are used as simple expressives, e.g., `fxxx`, as an interjection, were not included in the dictionary to prevent the false alarm.

A.2 Model

We deployed with the DistilKoBERT, [14] namely the distilled version of KoBERT. The performance of this model was measured with an F1 score of 0.5857 using APEACH.

A.3 Demo Page

The demo page shown in Figure 2 was implemented using Flask, [15] and was uploaded to the server using Heroku. [16] Due to the model size being too large to deploy on Heroku, we conducted quantization for linear operation and exported it using TorchScript.

B Post-Labeling Information

During the post-labeling process of APEACH, the number of sentences accepted/rejected by the task-manager is summarized in Table 3. The column 'tutorial session' in Table 3 denotes the session that was conducted to guarantee the dataset quality (Section 3.2). The workers who have passed the tutorial session participated in the main session. APEACH is a dataset that includes both tutorial and main session.

[13]Online community of a Korean university https://yourssu.com/

[14]https://github.com/monologg/DistilKoBERT

[15]https://flask.palletsprojects.com/en/

[16]https://www.heroku.com/

	Tutorial session		Main session	
	Non-hate	**Hate**	**Non-hate**	**Hate**
Accept	453	478	1386	1499
Reject	38	52	116	1
Total	491	530	1502	1500

Table 3: The number of accepted/rejected sentences in the post-labeling process. The workers with more sincerity (less faults) in the tutorial finally participated in the main session.

	Validation	
Train	**APEACH**	**BEEP! dev**
APEACH	-	KoELECTRA: 0.7502 KcBERT-Large: 0.7893
BEEP! train	KoELECTRA: 0.8101 KcBERT-Large: 0.8116	KoELECTRA: 0.7916 KcBERT-Large: 0.8295

Table 4: Evaluation results (F1 score) on APEACH and BEEP! with KoELECTRA and KcBERT-Large. Since APEACH does not have a specific training set, we exclude the case where the training set and the evaluation set are both APEACH.

C Training Details

Fine-tuning the pretrained language models bases on Huggingface Transformers (Wolf et al., 2020) 3.3.1 and PyTorch (Paszke et al., 2019) 1.6.0. For all the pretrained models, checkpoints available in Huggingface Model Hub were used. [17] GPU used for the fine-tuning is 1x NVIDIA TITAN RTX. We train the models for 10 epochs, with the batch size 32 and the learning rate of 5e-05.

D Training with APEACH

To check the quality of APEACH as both training and evaluation set, we trained two pretrained language models (KcBERT, KoELECTRA) with our APEACH and BEEP! train set, and the evaluation results with APEACH and BEEP! dev set are shown in Table 4. We obtained similar results from both models in training with BEEP! and evaluating with APEACH, while about 0.04 difference was displayed in evaluating with BEEP! dev set, which tells that BEEP! dev is more sensitive to pre-training corpora. However, we observed that the F1 score shifts down almost 0.04 lower for both models when the training is also done with APEACH. We first assume that the size of training set (BEEP! train - 8K, APEACH - 3.7K) matters, and also conjecture that the different composition of BEEP! and APEACH influences.

[17]https://huggingface.co/models