INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Crowdsourced Data Validation for ASR Training

Wannaphong Phatthiyaphaibun[1], *Chompakorn Chaksangchaichot*[2], *Thanawin Rakthammanon*[3], *Ekapol Chuangsuwanich*[2], *Sarana Nutanong*[1]

[1]School of Information Science and Technology, VISTEC, Thailand
[2]Department of Computer Engineering Chulalongkorn University, Thailand
[3]Department of Computer Engineering, Kasetsart University, Thailand

wannaphong.p_s21@vistec.ac.th, 6472014221@student.chula.ac.th, thanawin.r@ku.ac.th, ekapol.c@chula.ac.th, snutanon@vistec.ac.th

Abstract

Many ASR engines are based on crowdsourced speech corpora, such as Common Voice. Although crowdsourced data is inexpensive, the utterances obtained from crowdsourcing can be noisy because of uncontrollable factors such as accents, environments, etc. Another issue with the Common Voice corpus is the lack of validators to cover a vast collection of crowdsourced utterances. This issue presents a significant challenge to speech data validation. To mitigate this bottleneck, we propose a machine-learning classifier that predicts the correctness of the data, which can act as either the validator itself or a prescreen for the validator. Our system achieves more than 95% F1-score in the three Common Voice languages, including Thai, Japanese, and Turkish, and performs even better when we have only one human judge involved in the decision. Furthermore, we also found that the data obtained from our method outperformed the current crowdsourcing validation method when used to train the ASR model.

Index Terms: speech recognition, crowdsourcing dataset, speech quality control

1. Introduction

One of the most critical aspects of developing a powerful ASR engine is having an extensive, reliable collection of speech corpora. Recent studies show that the performance of the ASR engine is highly correlated with the number of hours used to train the data [1]. However, developing such a speech corpus can be costly. For example, according to Amazon Mechanical Turk, annotating an hour of English speech data costs around $90 to $150 [2]. One way to mitigate this problem is to use a crowdsourcing method.

The Common Voice Corpus is an excellent example of a speech crowdsourcing effort [3]. Due to the crowdsourced nature of the effort, Common Voice can collect up to 100 languages worldwide. While read-speech crowdsourcing is an economical way to obtain supervised data for constructing an ASR model, more attention is required on quality control. In particular, every read-speech crowdsourcing effort requires a significant portion of labor for utterance validation. In a typical setting, validating the read utterance usually takes at least two validators to agree whether the read speech is acceptable. The attempt to maintain the decision quality with a limited number of volunteer validators often results in a bottleneck on the validation side. For instance, Common Voice reported that among 26,119 hours of recording in all languages, only 17,127 hours of speech data were validated, i.e., more than 30% of the speech data is waiting to be validated. In Thai, Common Voice V12 was reported to have only 163 validated hours among a total of 401 hours, i.e., Only 40% of the data have been already validated and more than a half is still waiting.

There are several works aimed at facilitating the process of crowdsourcing speech datasets. Lee and Glass [4] utilized automatic quality control in the transcription task to help evaluate the annotator's quality, causing an increase in the overall quality of the speech dataset. Novotney and Callison-Burch [2] hired non-expert transcriptionists to help annotate the utterance via Amazon's Mechanical Turk, resulting in better cost optimization with minimal trade-offs on corpus quality. Another work utilized an existing ASR model to generate a transcription on a substantial amount of data before filtering the transcription using a simple heuristic [5]. Yet all these approaches were created to optimize transcription datasets, not read speech, and they do *not* directly address the bottleneck in the crowdsourcing validation process.

In this work, we aim to address this bottleneck and reduce the gap between the numbers of utterance readers and utterance validators. We use a machine-learning-based classifier to facilitate the validation process in crowdsourcing datasets. The machine learning model can be treated as a pre-screening for an utterance validator or even replace the validator, thus reducing the number of utterance validators needed in the system.

Our work contributes to the following points:

- We propose a novel classifier that can act as a pre-screen for the validation process, making utterance validation more efficient.
- We show that this method can be applied to multiple language setups. For this work, we investigated Thai, Japanese, and Turkish. The experimental results demonstrate that our method performs well for all these languages.
- We show that the ASR model trained with the data that was validated using our classifier can achieve better performance.

2. Proposed Method

To reduce the validation burden, we propose to use a classifier which tries to determine whether the recording is of good enough quality and matches with the text prompt. To achieve this we extract features from the utterance and text-prompt and trained the classifier on past validation results. The classifier can also be used in conjunction with a human validator for a final classification which is visualized in Fig. 1. In this section, we will go over the steps of defining the hand-crafted features and the model.

2.1. Classifier Features

To extract the features from the read-speech and text-prompt pair, we concatenated two groups of features, including *general features*, and *ASR-based Features*, creating a vector with a size

10.21437/Interspeech.2023-389

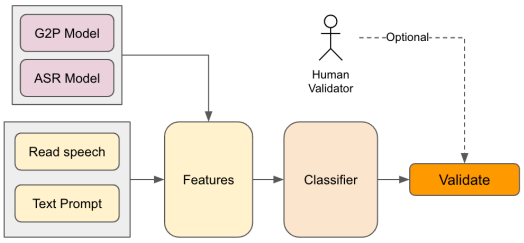

Figure 1: *The full process of our proposed method.*

of 11. The details of each feature are provided below.

2.1.1. General Features

The general features represent the general noise in the audio. This includes information on the voice activity of the audio, the signal-to-noise ratio (SNR), and the duration of the audio. This feature aims to capture any noise present in the audio as well as the portion of speech signal found in the audio. General features include:

- `snr` - SNR Estimation of Speech Signals from the sound [6].
- `dur` - the recording duration in seconds.
- `vad_dur` - the speech duration according to a VAD model [7] in seconds.
- `vad_ratio` - the ratio between the recording duration and the speech duration.

2.1.2. ASR-based Features

In addition to general information, an ASR can be used to help validate the utterance. We incorporate grapheme features, which were designed to capture the errors at the grapheme (or character) level obtained from an ASR model. Besides graphemes, phonemes can also be used which captures the error at the phoneme-level.

- `cer` - the character error rate (CER) of predicted text by the ASR model. Both ground truth text and hypotheses were pre-processed before measuring the error rate, such as removing whitespaces, converting the numbers to their spoken form, and formatting repeated marks for Thai language.
- `len_gt` - the number of words in the ground truth.
- `len_hyp` - the number of words in the model's prediction.
- `gt_word_dur_vad` - the speech duration divided by the number of words in the ground truth.
- `hyp_word_dur_vad` - The speech duration obtained divided by the number of words in the model's prediction.
- `per` - the Phoneme Error Rate. We still uses a standard ASR model which outputs characters. To get the phonemes labels/transcriptions, we use various G2P models. A Thai G2P model was trained on Wiktionary using MariaNMT [8]. For Japanese and Turkish, we use the G2P model from transphone. [9]
- `ipa_ed` - to better capture the sound difference, we can use a distance that takes into account of the closeness of the sound. This feature calculates the phonological feature edit distance proposed by PanPhon [10].

2.2. ASR Model

We use the publicly available Wav2Vec2 XLS-R [11] models that are finetuned to each language for our ASR models. All of the ASR models were trained using the old releases of the Common Voice corpus.

First, for Thai, we use a Wav2Vec2 XLS-R that has already been fine-tuned with Common Voice V7 [12]. For the Japanese version, we use `ttop324/wav2vec2-live-japanese` [13], which was finetuned to various Japanese speech corpora such as Common Voice V6 (train and dev set combined), JSUT corpus, CSS10 dataset, TedxJP-10k dataset, JVS corpus, and JSSS corpus. Lastly, for the Turkish language, we use the model from `m3hrdadfi/wav2vec2-large-xlsr-turkish` [14]. The Turkish model was trained on the Common Voice Corpus V6 (both the train and development sets were used). All of the models were decoded using greedy decoding, and the CER from each model was used as the features as described in the previous section.

2.3. Models

For the machine-learning-based classifier, we conducted our experiments on six different models using a scikit-learn library. The models considered are logistic regression, SVM, kNN, Random Forest, Naive Bayes, and Multi-layer perceptron.

For each model, we perform a grid search over its respective hyperparameters and set of features, and the model that performs best on the validation set is chosen for further evaluation.

3. Experimental Setups

To verify that our proposed method can be used to solve the validator shortage problem, we focus on the Common Voice dataset. We evaluate the effectiveness of our method in two ways:

- The F1-Score of the model with respect to the original Common Voice utterance validation results. We also consider the model performance in a human-in-the-loop manner where it is used in conjunction with an utterance validator.
- The CER was obtained from the training data that was validated by our method instead of the officially validated data.

3.1. Dataset Preparation

For the classifier dataset, we use Common Voice V9 for all languages which include Thai, Japan, and Turkish. For each language in the Common Voice data, we focused on three files. First, `validated.tsv` contains audio samples that was marked valid by having more upvotes than downvotes. These were treated as positive samples for our classifier. Second, `invalidated.tsv` contains audio samples that were marked invalid by having more downvotes that upvotes. They were treated as the negative samples. Third, `others.tsv` contains audio samples that have not been checked by a sufficient number of validators. This data was also included in the training of a downstream ASR described in section 3.3 For all experiments, we removed any samples that were involved in the ASR-based features to avoid data leakage.

We split the modified Common Voice V9 into train, validation, and test set, and use validation set for model selection.

Languages	Setups	No. Sample	No. Votes	Votes per sample	Precision	Recall	F1-Score
Th	Baseline	14485	30547	2.1088	-	-	-
	MODEL		0	0	95.56%	99.68%	97.57%
	MODEL+H		14485	1	99.71%	98.74%	99.22%
Jp	Baseline	22577	92176	4.0827	-	-	-
	MODEL		0	0	91.82%	98.87%	95.21%
	MODEL+H		22577	1	98.07%	94.76%	96.38%
Tr	Baseline	24266	73937	3.0469	-	-	-
	MODEL		0	0	98.01%	99.85%	98.92%
	MODEL+H		24266	1	99.68%	98.98%	99.33%

Table 1: *The F1-score of the classifier evaluated with different setups.* ***Th*** *denotes Thai,* ***Jp*** *denotes Japanses, and* ***Tr*** *denotes Turkish.*

3.2. Evaluation Setups

To measure the effectiveness of our classifier, we evaluate our classifier performance using a newer version of Common Voice, which is V12. Furthermore, since we investigate whether including human judgment can improve the performance, we run experiments in two scenarios:

- **MODEL** - This is the performance of using only the best classifier without any further human validation.
- **MODEL+H** - instead of fully relying on the classifier, we classify the utterance as valid only if both the model and a single human vote agree on their choice. If the model and human cannot reach a consensus, then we treat those samples as invalid.

Although the **MODEL** part can be evaluated without any human feedback, this is not the case for **MODEL+H** as they require human judgment. To simulate human judgment, we utilize the number of upvotes and downvotes obtained from the Common Voice dataset. For each Common Voice audio sample, we simulate a human judgment by sampling the vote choice, where

$$P(\text{valid}) = \frac{\text{number of upvotes}}{(\text{number of upvotes} + \text{number of downvotes})}$$

and $P(\text{invalid}) = 1 - P(\text{valid})$ accordingly. This configuration is used for all available languages.

3.3. Downstream Evaluation

In addition to testing how well the classifier works, we also investigate whether our method can have any potentially adverse effect if it replaces the current validation process in terms of ASR training. To do this, we fine-tune Wav2Vec2 XLS-R using the Common Voice V12 dataset that was validated using the validation methods mentioned above. The reported CER obtained from both models is then compared with two setups. The first baseline model, marked as **No validation** in Table 2, was finetuned on all data mentioned in 3.1. The other baseline was finetuned only on `validated.tsv` (marked as **CV Validated** in table 2). We evaluate the CER on the Common Voice V12 test set. Note that we removed all Common Voice data that appeared in the classifier training data to avoid data leakage.

4. Results

4.1. Classifier Results

As shown in Table 1, our model alone can achieve the F1-score of more than 95% in all languages. Furthermore, when the model was combined with a single human judgment, the performance improved even more. This suggests that our framework works well either when using the model alone or when there's only one human judgment needed.

Additionally, to show that our method can improve the efficiency of the validation process, we also provide the average number of validators used in the process. The baseline refers to the current crowdsourcing process used by Common Voice, where at least two validators are required for each sample.

4.2. Downstream ASR Results

Table 2 reports the CER of the ASR model when the data was trained on different setups. According to the results, the model trained using the data that our classifier validated has a lower CER than the baseline model. Additionally, our method significantly outperforms the model trained with only validated Common Voice data. Interestingly, the ASR performance is not entirely dependent on the amount of data. When trained with no validation at all, the ASR performance can be worse than using some validation. In the worst case like in Jp, the model failed to converge. However, the differences between Model and Model+H seem to suggest that the slight difference in quality seems to not have a noticeable effect on ASR performance.

Setups	CER			Train Size (min)		
	Th	Jp	Tr	Th	Jp	Tr
No Validation	10.98	98.08	9.74	516	1317	900
CV Validated	14.82	16.20	10.06	247	226	813
Model	10.57	**9.89**	**9.27**	464	1232	897
Model+H	**10.34**	10.50	9.42	462	1080	894

Table 2: *The CER of the Wav2Vec2 XLS-R model when finetuned using the dataset obtained from a different validation method. We also provide the amount of training data in minutes for each language setups.*

5. Discussion

5.1. Validator Efficiency

In this section, we explore the efficiency of the validation process when utilizing our method. From Table 1, we can clearly see that our method can save up to four times the number of validators with minimal F1-score reduction. Furthermore, the ASR model obtained from the data validated by our model alone requires zero validators while achieving better performance than the data validated by more than two humans. Even though we utilized only one human to validate the process, not only can we

obtain better data quality but we can also save up to four times the number of votes. This clearly suggests that our method can reduce the number of validations required while maintaining the quality of the data.

5.2. Feature Importance

Since we also use Random Forest as one of our machine learning algorithms, we investigate the feature importance of the model to see which features are the most significant. For both Thai and Japanese, we found that `cer` is the most significant feature, while for Turkish, the most dominant feature appears to be `ipa_ed`. However, when we take all languages into account, we notice that both `ipa_ed` and `snr` are among the top 3 most dominant features. Our manual inspection of some of the Thai common voice data reveals that many of the `invalidated.tsv` samples contain some kinds of noise (850 out of 940 samples) or samples that do not exactly match with the text prompt (529 out of 940 samples). Therefore, `ipa_ed` and `snr` are reasonable features here.

5.3. Threshold Usage

When using our classifier in Section 2, there is a wide range of thresholds that can be used. This raises the question of whether using a higher threshold can result in better data quality by lowering false negatives. Therefore, we conduct a simple experiment to see if there is any significant improvement when we raise a threshold from a default value of 0.5 to 0.8. The result is shown in Table 3.

Metrics	Threshold	Th	Jp	Tr
Train Size (min)	0.5	462	1232	897
	0.8	406	1053	883
Classifier (F1-Score)	0.5	97.57%	95.21%	98.92%
	0.8	98.61%	96.27%	99.26%
ASR (CER)	0.5	10.57	9.98	9.27
	0.8	10.98	10.38	9.52

Table 3: *The performance of the classifier and the downstream ASR model when using different classifier thresholds.*

From the results, there are slight improvements in F1. However, this does not translate to better ASR performance. When we raise the classifier's threshold, there are fewer training data available for the ASR model, and this could result in an overall degradation of the model's performance.

5.4. Replacing the ASR-based features with features from a Phoneme Recognizer

Our method assumes an existing ASR model for that language. To ensure that our approach also works without this assumption, we conduct an experiment where we remove the need for the ASR model in our classifier and replace it with an off-the-shelf phoneme recognition model [15]. Note that the phoneme recognition model's training data does not overlap with the downstream ASR model's training data. With this change, our ASR-based features now only include `per` and `ipa_ed`. The results are shown in table 4. The results show only small differences when using the phoneme recognizer. Thus, if a specific language does not contain a strong ASR model, we may utilize an off-the-shelf phoneme recognition instead.

Setups	Metrics	Th	Jp	Tr
MODEL	Train Size (min)	482	1246	898
	Classifier (F1-Score)	97.07%	94.30%	98.85%
	ASR (CER)	10.34	9.90	9.48
MODEL+H	Train Size (min)	480	1078	894
	Classifier (F1-Score)	99.11%	96.44%	99.41%
	ASR (CER)	10.28	10.19	9.76

Table 4: *The performance of the classifier and the downstream ASR model in the setups where we replace the ASR model with Phoneme Recognition Model.*

6. Future Works

There are several rooms to improve our current frameworks. To this end, we have proposed a framework that facilitates validators' work by utilizing a machine-learning-based classifier as a pre-screening criteria. However, there are several areas that can be explored further, and we grouped them as follows:

- **Bias from human judgment** - Although Common Voice has a clear standard for defining what speech utterances are valid or invalid, the final judgment still depends on the validator's bias. For example, given that the speech is read properly but the accent makes the speech sound harder to interpret, should we keep this utterance? If we can define the validation rules more clearly, we can engineer better-handcrafted features.
- **Cross-lingual Setups** - Another area to explore is how we can create a classifier that is generalizable across multiple languages. Our work showed that it is possible to leverage crowdsourced existing data to train a classifier used for validating utterances. However, this is specific to only one language, and we will need to train a new model for each new language. If the classifier can be used across different language domains, it will be more efficient.

7. Conclusions

We propose a framework that uses a machine-learning classifier to facilitate the validation process of crowdsourced speech data. Our classifier was trained using the data from the Common Voice Corpus, and we evaluated the results in two setups: when our classifier acted as a single validator and when our classifier was used together with a single human judgment. The results suggested that our model can achieve up to 95% accuracy, and the performance improved further when the system was used with only one human vote. Furthermore, we also evaluate the quality of the data obtained when validated using our proposed method. The ASR model trained with the data validated using our classifier alone achieved a better CER when compared with the data validated by the Common Voice crowdsourcing method while saving up to four times the number of votes. Our method also worked well with a phoneme recognizer for languages where there might be ASR available. Additionally, we also investigated the effect of raising the classifier threshold, but we did not see any improvement on the downstream ASR when raising the threshold.

8. References

[1] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," *arXiv preprint arXiv:2212.04356*, 2022.

[2] S. Novotney and C. Callison-Burch, "Cheap, fast and good enough: Automatic speech recognition with non-expert transcription," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Los Angeles, California: Association for Computational Linguistics, Jun. 2010, pp. 207–215. [Online]. Available: https://aclanthology.org/N10-1024

[3] R. Ardila, M. Branson, K. Davis, M. Kohler, J. Meyer, M. Henretty, R. Morais, L. Saunders, F. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," in *Proceedings of the Twelfth Language Resources and Evaluation Conference*. European Language Resources Association, pp. 4218–4222. [Online]. Available: https://aclanthology.org/2020.lrec-1.520

[4] C. ying Lee and J. Glass, "A transcription task for crowdsourcing with automatic quality control," in *Proc. Interspeech 2011*, 2011, pp. 3041–3044.

[5] O. Kapralova, J. Alex, E. Weinstein, P. Moreno, and O. Siohan, "A big data approach to acoustic model training corpus selection," in *Conference of the International Speech Communication Association (Interspeech)*, 2014.

[6] C. Kim and R. M. Stern, "Robust signal-to-noise ratio estimation based on waveform amplitude distribution analysis," in *Interspeech 2008*. ISCA, pp. 2598–2601. [Online]. Available: https://www.isca-speech.org/archive/interspeech_2008/kim08e_interspeech.html

[7] F. Taguchi, "py-webrtcvad wrapper for trimming speech clips," 2022. [Online]. Available: https://github.com/F-Tag/python-vad

[8] M. Junczys-Dowmunt, R. Grundkiewicz, T. Dwojak, H. Hoang, K. Heafield, T. Neckermann, F. Seide, U. Germann, A. Fikri Aji, N. Bogoychev, A. F. T. Martins, and A. Birch, "Marian: Fast neural machine translation in C++," in *Proceedings of ACL 2018, System Demonstrations*. Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 116–121. [Online]. Available: http://www.aclweb.org/anthology/P18-4020

[9] X. Li, F. Metze, D. Mortensen, S. Watanabe, and A. Black, "Zero-shot learning for grapheme to phoneme conversion with language ensemble," in *Findings of the Association for Computational Linguistics: ACL 2022*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 2106–2115. [Online]. Available: https://aclanthology.org/2022.findings-acl.166

[10] D. R. Mortensen, P. Littell, A. Bharadwaj, K. Goyal, C. Dyer, and L. Levin, "PanPhon: A resource for mapping IPA segments to articulatory feature vectors," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. The COLING 2016 Organizing Committee, pp. 3475–3484. [Online]. Available: https://aclanthology.org/C16-1328

[11] A. Babu, C. Wang, A. Tjandra, K. Lakhotia, Q. Xu, N. Goyal, K. Singh, P. von Platen, Y. Saraf, J. Pino, A. Baevski, A. Conneau, and M. Auli, "XLS-R: Self-supervised Cross-lingual Speech Representation Learning at Scale," in *Proc. Interspeech 2022*, 2022, pp. 2278–2282.

[12] VISTEC-depa AI Research Institute of Thailand, "wav2vec2-large-xlsr-53-th (revision 3155938)," 2023. [Online]. Available: https://huggingface.co/airesearch/wav2vec2-large-xlsr-53-th

[13] ttop324, "wav2vec2-live-japanese," 2021. [Online]. Available: https://huggingface.co/ttop324/wav2vec2-live-japanese

[14] M. Farahani, "wav2vec2-large-xlsr-turkish," Mar. 2021. [Online]. Available: https://huggingface.co/m3hrdadfi/wav2vec2-large-xlsr-turkish

[15] Q. Xu, A. Baevski, and M. Auli, "Simple and Effective Zero-shot Cross-lingual Phoneme Recognition," in *Proc. Interspeech 2022*, 2022, pp. 2113–2117.