Controlling Quality and Handling Fraud in Large Scale Crowdsourcing Speech Data Collections

Spencer Rothwell, Ahmad Elshenawy, Steele Carter, Daniela Braga, Faraz Romani, Michael Kennewick, Bob Kennewick

VoiceBox Technologies, Bellevue, WA, USA

spencerr, ahmade, steelec, danielab, farazr, michaelk, bobk {@voicebox.com}

Abstract

This paper presents strategies for measuring and assuring high quality when performing large-scale crowdsourcing data collections for acoustic model training. We examine different types of spam encountered while collecting and validating speech audio from unmanaged crowds and describe how we were able to identify these sources of spam and prevent our data from being tainted. We built a custom Android mobile application which funnels workers from a crowdsourcing platform and allows us to gather recordings and control conditions of the audio collection. We use a 2-step validation process which ensures that workers are paid only when they have actually used our application to complete their tasks. The collected audio is run through a second crowdsourcing job designed to validate that the speech matches the text with which the speakers were prompted. For the validation task, gold-standard test questions are used in combination with expected answer distribution rules and monitoring of worker activity levels over time to detect and expel likely spammers. Inter-annotator agreement is used to ensure high confidence of validated judgments. This process yielded millions of recordings with matching transcriptions in American English. The resulting set is 96% accurate with only minor errors.

Index Terms: unmanaged crowdsourcing, mobile speech data collection, validation, spam, quality control, Acoustic Modeling (AM), Automatic Speech Recognition (ASR).

1. Introduction

Many tasks in Natural Language Processing (NLP) require large amounts of annotated or transcribed language data. It is expensive to create and transcribe this data by hand. Costs can be significantly reduced through the use of crowdsourcing to accomplish simple tasks. These cost reductions often come at the price of introducing noise and consequently lowering data quality. The challenge of crowdsourcing is therefore to efficiently filter out the noise introduced by unreliable workers in order to maintain high data quality. The most common approach for maintaining crowdsourced data quality is to introduce gold-standard work units alongside normal work units. The worker accuracy can then be reliably measured and input from inaccurate workers can be rejected. This prevention mechanism, while adequate for some crowdsourcing jobs, is inadequate or inapplicable for more complicated tasks such as audio collection. Additionally, malicious workers can find innovative ways to circumvent gold-standard work units such as learning the answers to test questions and then reusing those answers under different accounts. We refer to this type of tainted input as spam, which is distinguished from low quality data, the former being intentionally produced and easily spotted because of the regular pattern it shows (e.g. copying and pasting the same string repeatedly as responses for a task); the latter is less obvious and not necessarily intentionally produced (e.g. an utterance about weather in response to a scenario regarding music). Spam becomes especially problematic when collecting high volumes of data because workers have more time to devise strategies for bypassing the existing validation mechanisms. Preventing spam requires a robust approach, combining a number of spam detection strategies to ensure that only high quality responses are collected.

Our work describes quality control and spam detection strategies used while collecting high volumes of crowdsourced audio samples from text prompts. We use a custom designed audio elicitation mobile app which allows us to measure audio properties which are indicative of spam. We also describe crowdsourcing strategies for validating that collected audio matches with given text. Additionally, we describe strategies for preventing hackers from circumventing our quality control and spam prevention tools. All of these strategies combined provide us with large volumes of accurate speech data at very low cost.

Since 2008, there has been a marked increase in the number of NLP conference papers that use crowdsourcing to achieve some goal [1], but there has not been significant attention paid to the issue of cheating and spamming that arises in crowdsourcing. Spam is an important issue to address when one considers using crowdsourcing [13]; without any quality control in place, a substantial portion of the crowd will be comprised of spammers [4, 12]. While it is not uncommon to see issues of quality control addressed in such papers [6, 7, 8, 9], quality control approaches are commonly limited to the use of gold-standard questions, to which answers are known a priori and used to evaluate worker contributions [5, 9, 10]. Other quality control approaches that have been adopted include crowdsourced recruitment, whereby a task is designed to find quality workers whom are later manually selected for further work [11], filtering through majority vote [5], and use of generated completion codes to determine legitimate worker contributions [2].

One of the most comprehensive assessments of illegitimate worker contributions in crowdsourcing tasks is from Eickhoff and De Vries [2], who investigate the different types of cheaters, the different ways in which they cheat, and the characteristics of a task they are attracted toward. They demonstrate that the complexity of a task can strongly influence the percentage of spam received, with simpler tasks being much more vulnerable to spammer attention than more complicated tasks. The authors also show how the amount of
work available to a worker can make a difference. If a task has enough units that workers can complete a large volume of work and subsequently have the potential to earn a more substantial amount of money, it will then be hit harder by spammers than a task that does not present workers with much “reuse potential,” or profit-potential. Their work also suggests that gold-standard test question evaluation is insufficient for filtering out spammers. Other techniques, such as filtering by worker location or utilizing recruitment-crowdsourcing, are more effective methods of filtering spammers. These strategies, however, exponentially increase the time needed to complete the task [2].

Similar to [2], Gadiraju et al. [4] identify the different kinds of workers and cheaters that emerge in crowdsourcing, developing a taxonomy of worker types, listed here in increasing order of prevalence: (GP) Gold-Prey, (SD) Smart-Deceivers, (RB) Rule-Breakers, (FD) Fast-Deceivers. Gadiraju et al. demonstrate that each class of spammer exhibits different behaviors. Effective quality control must take into consideration that there are multiple distinct types of spam and spammers, and thus a unilateral approach is insufficient in keeping crowdsourced data clean. Both [2] and [4] show that gold-standard test questions are insufficient in adequately handling spammers. While there are other solutions in place, they are either too slow [2], or require too much post-processing [3] to be feasible on very large scale deployments. The work we do is on the scale of millions, far larger than the magnitude of thousands that these studies are conducted over. Our work here aims to demonstrate the efficacy of common quality control mechanisms on massive-scale projects, as well as novel approaches that only prove necessary when working on this scale.

This paper is organized as follows: section 2 reports the speech data collection and validation; section 3 describes spam types, anti-spam solutions and results; section 4 presents the discussion and conclusions.

2. Speech collection and validation

In order to crowdsource our mobile speech collections, we developed an Android based application named Jibe. It enables us to collect recordings while maintaining control on lead-in or lead-out recording time, ambient noise detection, silence detection, periodic speech audits, etc. This is a standalone app that was distributed to unmanaged crowds (i.e. workers that plug in directly to our platform without any vendor management agreement and without formal screening or testing) via our crowdsourcing platform. In order to access the crowds available in the platform, it was necessary to design a validation system which enabled us to funnel workers off of the platform and into our application. Workers are given our application to download, as well as detailed instructions on how to install and use the application. A unique token is provided to each worker and our application maps each token to a specific collection of prompts and audits to be used, hereby referred to as a session. Upon entering their provided token into the application, the worker is guided through an arbitrary number of predetermined prompts (Figure 1), which are recorded and instantly uploaded to an online hosting service. Periodically throughout a session, workers are presented with what we call an audit, a quality-control implementation which is discussed in detail in Section 3.3. Upon completing their session, a worker is presented with a 9-character completion code. The worker can enter their completion code into the crowdsourcing platform, and we run custom built validation code to verify that the entered code is legitimate by checking it against a log of all generated completion codes. On successful verification, the completion code is deleted from our system so that it cannot be reused. Each session contains 20 prompts and 4 audits, and workers are paid $0.25 for completion of a session. The amount of tokens that a single worker can complete is flexible, determined on a case-by-case basis depending on how much speech data we need to collect.

The second step of the speech data collection is the validation of all recordings obtained via Jibe. After importing the results of our collection, the text strings (prompts) along with the audio paths are exported to a second job on our crowdsourcing platform. In this job, workers are presented with the text and audio and asked if the two match (Figure 2). We refer to this pairing as a unit. If a worker selects yes, they are saying that the two are exact matches. If they select no, then they are asked to select the degree to which the text and audio differ from three categories, minor, significant, and unintelligible. Their answer to this follow-up question is not used in our evaluation and has no bearing on a worker’s accuracy in relation to test questions. The follow-up question is presented to workers solely as a way of increasing the attention that they must pay to the task.

Each unit is reviewed by 2 or 4 workers. If the first two workers agree that a unit is valid, then no more workers are needed to review that unit. These units are categorized as high confidence – positive. An internal review by transcription experts found that the crowd correctly judged these with an accuracy rate of 96%, as depicted in Table 1.
Table 1. Worker accuracy rate by confidence level in a sample evaluation of 9000 units in American English (total set was 2 million units)

<table>
<thead>
<tr>
<th>Confidence Level</th>
<th>Positive Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>High conf.</td>
<td>96%</td>
</tr>
<tr>
<td>Medium conf.</td>
<td>71%</td>
</tr>
<tr>
<td>Low conf.</td>
<td>43%</td>
</tr>
<tr>
<td>Negative</td>
<td>95%</td>
</tr>
</tbody>
</table>

If those first two workers agree that a unit is not valid, then no more workers are needed to review that unit. These units are labeled as **negative** and the crowd accurately rates these 95% of the time. If the first two workers disagree, then the unit is reviewed by two more workers automatically within the same job. The goal of this is to split controversial units into two further categories. If three workers say that a unit is valid and one says it is not, it is categorized under **medium confidence** – positive. The accuracy rate of these medium confidence units is substantially worse than high confidence units, at only 71%. Units that receive two positive judgments and two negative judgments and have more than 0.5 confidence are placed into a category called **low confidence** – positive. Only 43% of these are correct. Units with two positive and two negative judgments that have less than 0.5 confidence are added into the **negative** category. The purpose of having these separate categories comes into play when triaging the data to use when training our AM, i.e. we only use the **high confidence positive** set for this purpose. By sorting the units using confidence scores, we reduce the total size of the dataset but this allows us to guarantee 96% accuracy. If we included the lower confidence categories in our final dataset, the overall accuracy would be around 90%. The reason for distinguishing between medium confidence and low confidence categories is a defining factor when determining possible follow-up up steps. Medium confidence units may or may not be re-transcribed and revalidated in an attempt to increase the overall size of the high confidence data set and thus enlarge the AM training set. When we conducted our massive-scale validation campaign, 1.95 million high confidence units out of a total of 2.5 million units (78%) still made it into the set used to train the AM.

In the event of a tie, a confidence rating algorithm is applied in order to determine how to classify the unit. Confidence is calculated as a weighted average of all submitted votes for a given unit. Equation (1) calculates the weight of an individual vote for a unit:

\[ w_i = \frac{\sum_{j=1}^{n} x_{i,j} \cdot p_{i,j}}{n_k} \]  

(1)

In equation (1), \( n \) represents the number of votes received for a unit and \( k \) represents the test question accuracy of the worker who supplied that vote.

Equation (2) takes the average of all voted answers, where each vote is weighted by the result of equation (1).

\[ K = \frac{\sum_{i=1}^{n} w_{i} \cdot p_{i,j}}{n} \]  

(2)

Bad workers, who score less than 70% accuracy on our test questions, are automatically banned from the job and their answers are removed from the data. The units for which we received the bad data are then re-added to the pool of units pending judgments from the crowd.

3. Observed fraud and anti-fraud solutions

When deploying JIBE to the US crowds and internationally, we witnessed three distinct kinds of fraud, as described below.

3.1. Validator bypassing

The first kind of fraud that we observed took place when some workers devised a way to circumvent the validation architecture built into the crowdsourcing platform. This being a relatively lucrative crowdsourcing job, with us paying $0.25 per submission and there being a large amount of work available, our jobs were particularly attractive to hackers and spammers. Hackers and spammers were able to bypass our custom validation protocols and enter any text string into the completion code field and receive pay due to an unforeseen vulnerability of the platform for users to manipulate client-side JavaScript validators. We were able to stop this behavior by implementing a two-stage validation that, though simple, proved effective by deterring this type of fraud.

Essentially, it is a validator wrapped around another validator, where the second-layer of validation only activates upon successful first-layer validation. Successful first-stage validation presents workers with a simple checkbox that must be checked in order to allow submission. If a worker has circumvented first-stage validation, they are not presented with the second-stage validation, thus barring them from submitting their work and contaminating the data-pool.

Our two-stage validator dramatically reduced the amount of fraud we experienced, saving both money and time. In addition, it provided only a trivial amount of extra work for legitimate workers, and our worker-satisfaction ratings received from the crowdsourcing platform indicated that the extra stage was not an inconvenience to legitimate workers.

We ran an experiment to examine the efficacy of our improved validator, the results of which are shown below in Table 2. Our crowdsourced task using our initial 1-stage validator was hit incredibly hard by spammers. A spammer in this instance is a worker who successfully submitted an invalid completion code, circumventing our validation. Completion codes are always of a set length and content, and it is easy to observe when a worker’s submission is illegitimate.

<table>
<thead>
<tr>
<th>Total Workers</th>
<th>Spam</th>
<th>Tainted</th>
<th>Time (h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-Stage</td>
<td>230</td>
<td>84 (56%)</td>
<td>70</td>
</tr>
<tr>
<td>2-Stage</td>
<td>150</td>
<td>9   (6%)</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 2. Encountered spam rates before and after implementation of improved validation method.

Asking 150 workers to complete our task, over half of the received submissions were clearly fraudulent, meeting neither the length nor content criteria expected from a valid completion code. The “Tainted” column in the table shows the number of received judgments that were intercepted while the task was running, meaning these intercepted spammed units were fed back into the work pool so that legitimate workers could complete the work instead.

Our 2-stage validator encountered profoundly less spam. Only 6% of our final results were illegitimate. The task was also completed much more quickly, as the crowdsourcing platform did not have to waste any time flagging workers and...
feeding back spammed data into the work pool. The reduction in spam resulted in significant cost savings as well.

3.2. Spam in speech recordings

A common way of cheating during our audio collections involved instances where workers would completely ignore the prompts shown to them. This resulted in collections of recordings that consisted entirely of background noise, such as a TV, or unrelated speech, such as the worker having a conversation with their friend.

We were able to stem this type of fraud by randomly inserting controlled “audits” throughout a session. These audits consisted of phrases that were tested internally by five people of varying accents through our server side ASR system. Each person spoke the phrase into the app five times. A phrase was only allowed for use as an audit if the ASR was able to correctly identify it 100% of the time for all five people. When a worker submits their recording for an audit, the audio is sent to our ASR, and we use the Levenshtein edit-distance algorithm [14] on the highest confidence ASR hypothesis. We then calculate a score of how “correct” the worker was on their audit, which is determined by the calculated edit-distance normalized by the length of the audit string. If the calculated score falls within an acceptable, arbitrary range of error, the audit is considered “passed” and the worker is allowed to continue working. If a worker fails a set amount of audits, the application blocks that user from doing more recordings and they are barred from participating in future speech collections.

3.3 Test questions learning

After validating hundreds of thousands of units, we experienced a sophisticated kind of fraud that drove us to implement several new steps in our pipeline to prevent this fraud in future jobs. Before experiencing this particular fraud, we composed a set of several hundred test questions. The idea for this was simply to allow for enough test questions that one could be presented for every ten units and thereby allow individual workers to complete thousands of real units if they so desired. After running our jobs many times, however, we saw the accuracy of the crowd sharply decrease and the cost of jobs take an unexpected spike. After an investigation, we concluded that this was because a sophisticated spammer had infiltrated our jobs by completing a majority of our test questions, thereby reducing the quality of our data set. We identified in a finalized follow-up set of 2,500,000 units. A systematic spam rate significantly from this expected ratio. The next step we took was to implement a set of revolving test questions. This required the writing of several thousand additional test questions, which were used to validate several million more units of data with high confidence. We never expose the entire set of test question at one time. We use a random subset of the test questions every time we deploy a job. This prevents the spammer from hacking one job and being able to immediately use the answers for that particular job’s test questions to systematically spam any other job. Before implementing the above described methods, we identified a systematic spam rate of approximately 40% in a finalized set of 375,000 units. After implementing these methods, no systematic spam was identified in a finalized follow-up set of 2,500,000 units. A useful method to identify spam accounts is to look at their activity. We can analyze graphs of worker activity over time and see a distinction between legitimate and spam accounts.

We can clearly see the pattern of the legitimate user resembles human activity in that it is not uniform over time (Figure 3); the worker took periodic breaks. The activity of the spam-bot is much more uniform and resembles a flat line (Figure 4). It spent roughly 7 hours completing its task while taking no breaks and had no significant variation in activity within that timeframe.

4. Conclusion

In this paper we described our process for collecting and validating large volumes of speech data for commercial level acoustic model training through unmanaged crowds. Crowdsourcing presents many challenges with regard to maintaining data quality due to fraud. We were able to improve the observed accuracy rate to 96% by combining a number of spam prevention strategies and quality control steps. We designed a mobile application for speech recordings (Jibe) which was used in conjunction with our crowdsourcing platform through a 2-step validation system, which could not easily be bypassed by hackers. Using this application gave us the ability to implement additional spam prevention techniques that would otherwise be unavailable through the crowdsourcing platform alone. We implemented periodic ASR based speech audits and monitoring of audio quality metrics to enforce audio content and quality. On top of this, we sent all recordings through a second stage of validation which harnessed the crowd in filtering collected audio that did not exactly match their corresponding text prompt. The end result of this process yielded consistently high quality speech data on a large scale and at low cost. These anti-fraud techniques are language agnostic and ensure high quality data collections and validations for commercial level ASR products.
References


