Training Strategies for Promoting Accuracy in Crowdsourced Content Analysis

Ceren Budak[a]*, R. Kelly Garrett[b] and Daniel Sude[c]

[a]*School of Information, University of Michigan, Ann Arbor, USA;* [b]*School of Communication, Ohio State University, Columbus, USA;* [c]*Department of Communication, Tel Aviv University, Tel Aviv, Israel*

* Address correspondence to Ceren Budak, University of Michigan, School of Information, 105 S. State St. Ann Arbor, MI 48109-1285, USA; e-mail: cbudak@umich.edu

Training Strategies for Promoting Accuracy in Crowdsourced Content Analysis

In this work, we evaluate different instruction strategies to improve the quality of crowdcoding for the concept of civility. We test the effectiveness of training, codebooks, and their combination through 2x2 experiments conducted on two different populations—students and Amazon Mechanical Turk workers. In addition, we perform simulations to evaluate the trade-off between cost and performance associated with different instructional strategies and the number of human coders. We find that training improves crowdcoding quality, while codebooks do not. We further show that relying on several human coders and applying majority rule to their assessments significantly improves performance.

Keywords: crowdsourcing; content analysis; civility; randomized experiments; labeling

Introduction

Analysis of texts on a large scale is increasingly important in the social sciences generally and political communication specifically. The analytic goal is to quantify theoretically important attributes of the texts under consideration in a manner that is meaningful, reproducible, and efficient. Crowdsourcing has proven to be an invaluable part of this effort. Splitting a coding task among a large number of non-expert coders—a process often referred to by the portmanteau "crowdcoding"—makes large textual analysis projects more tractable. In some cases, the resulting quantitative data are an end in themselves (e.g., Benoit et al., 2016), while in other cases they function as a training set for other more computationally intensive approaches, such as supervised learning models (for a review, see Grimmer and Stewart, 2013). Crowdcoding performs almost as well as more traditional coding methods for simple concepts (e.g., Lind et al., 2017), and adequate performance with more nuanced concepts is possible under the right conditions (e.g., Budak et al., 2016; Horn, 2019).

Although the viability of crowdcoding is clear, important questions about how it should be implemented remain. As with more traditional approaches to coding, researchers must provide instructions describing the coding task. Some steps are consistently employed across the extant literature (see Krippendorff, 2019; Neuendorf, 2016). For example, coders must have a description of the codes that they will be applying. Beyond that, however, there is considerable variation in what kinds of instructions crowdworkers receive. Sometimes workers are provided with a codebook, which provides detailed descriptions of the concept. It may also include examples or exclusions. This is relatively uncommon (Lind et al., 2017); more often, workers get only a sentence or two describing each code. Workers are also sometimes required to complete a brief training exercise before coding the target content (e.g., Jacobson et al., 2017). This is even less common than providing a codebook.

This article assesses the effectiveness of different instruction strategies for promoting high quality crowdcoding of a latent concept. The test is conducted in the context of political communication and focuses on the concept of civility. Using simple code descriptions as the baseline, we consider the effects of adding a previously validated codebook, of introducing a training exercise, or both. We also use simulations to examine the effect of introducing majority-rule decision making to the coding process and to assess the cost-performance trade-off associated with using different training strategies and varying the number of raters for each text.

Civility

This research is conducted in the context of coding civility. Civility is a complex concept that has been defined in a variety of ways (Stryker et al., 2016). For instance, some scholars focus on insults, extreme language, and emotionality (Gervais, 2015), while others place more emphasis on messages that threaten democracy, deny people

their personal freedoms, or stereotype social groups (Papacharissi, 2004). The concept is also notoriously difficult to code (e.g., Stroud et al., 2015). Although Americans decry incivility in everyday online political talk (Center, 2016; Gardiner et al., 2016), there is considerable variability in which features individuals find uncivil (Kenski et al., 2020). For example, individuals who score high on measures of agreeableness tend to find vulgarity uniquely objectionable. As a result, training coders is time-consuming, often taking several weeks (Coe et al., 2014).

We intentionally focus on a latent concept, which is considerably more difficult to code than a manifest concept, for two key reasons. First, latent concepts, though less obvious, are no less important. Scholars often need to be able to systematically assess the influence of subtle aspects of a message (Neuendorf, 2016, p. 31-33). Second, latent concept coding is especially reliant on having effective instructional strategies. Training someone to code a manifest concept is often as simple as describing the feature to be identified (e.g., Does the author explicitly claim party affiliation?) Coding more nuanced latent concepts, however, requires careful guidance to ensure that coders can systematically assign codes that are consistent with researchers' conceptualizations. It is worth noting, though, that civility is not necessarily representative of all latent concepts.

Our conceptualization of incivility is informed by influential work in this area, focusing on attributes that make a message unnecessarily disrespectful (Coe et al., 2014). This includes features such as name-calling, vulgarity, and accusations that someone is lying. We also explicitly include in our conceptualization statements intended to restrict others' speech, including threats of violence. We treat this as a dichotomous outcome, consistent with other recent scholarship in this area (e.g., Coe et al., 2014, Stroud et al., 2015). Our approach is unique, however, in that we do not split

the concept into different sub-elements before merging to a single dichotomous variable.

Content Analysis

In the social sciences, the goal of any content analysis is to produce valid and reliable measurement of the frequency (and, sometimes, intensity) with which a concept occurs. More recently, computer scientists have embraced the method as a tool for building models that characterize content in ways that do not rely on counts (e.g., sentiment analysis). Whatever the application, the method must identify all instances of the concept to be effective and it should do so in a way that is reproducible, yielding comparable results if the analysis were repeated (Neuendorf, 2016, p.122). Social scientists have used this methodology for many years and best practices are well documented (Krippendorff, 2019). There are detailed guidelines for producing codebooks, training coders, validating codes produced, and reporting results.

Conventional applications of the methodology typically focus on analysis of dozens or hundreds of texts. While the method has been scaled up to handle thousands of posts (e.g., Graham and Wright, 2015; Sadeque et al., 2019), crowdcoding is becoming an increasingly popular alternative to handle such cases. Distributing a coding task across a large number of coders allows the process to include thousands (or even tens of thousands) of coding units (Benoit et al., 2016). Even so, large-scale observational studies, such as those using data from social media platforms, can quickly grow too large for crowdcoding alone. In those cases, researchers generally rely on a combination of supervised learning methods and crowdcoding (e.g. Budak et al., 2016). Some studies avoid data labeling all together by relying on dictionary-based or unsupervised approaches (see Grimmer and Stewart, 2013, for a review).

Crowdsourcing Content Analysis

The increasing prominence of large-scale text analysis raises important questions about how to translate the best practices of conventional content analysis into a methodology that can be executed by a large group of loosely connected individuals with limited training. Interest in crowdcoding has grown rapidly, earning it an entry in the recent edition of Krippendorff's influential textbook (2019, pp. 135-138). A Google search for the term generates dozens of relevant results, many of which have been published in the last five years, including articles published in this journal. It is clear from the extant literature that how the coding task is presented to workers can have an effect on the codes they provide (Horn, 2019; Lind et al., 2017; Guo et al., 2019).

Some studies only describe the concept being coded via the labels provided. For example, when creating the training set for the Perspectives API, a system designed to identify "toxic" online comments, the Conversation-AI team asked workers to code statements using a five-point scale anchored by "Very Toxic (a very hateful, aggressive, or disrespectful comment that is very likely to make you leave a discussion)" and "Very healthy contribution (a very polite, thoughtful, or helpful contribution that is very likely to make you want to continue a discussion)" (Perspective, 2020). Other studies used more traditional codebooks (e.g., Jacobson et al., 2017).

Codebooks are intended to be unambiguous and highly detailed, which should help reduce individual differences among coders (Neuendorf, 2016). These documents are developed iteratively. Domain experts draft materials and coders provide feedback based on their attempts to use them in the coding process. Providing a codebook to crowdsourced workers seems likely to have many of the same benefits. Providing more information about how text is to be coded should allow crowdcoding to better

approximate the coding produced using more conventional content analysis methods.

Another element of the conventional coding process is training. Coders are typically required to code a collection of text, meeting to discuss their decisions with the research team and to get feedback. This experience gives them a chance to become more familiar with the concepts of interest and with the codebook itself (Neuendorf, 2016). Modest training tasks have been integrated into some crowdcoding studies. For example, one study asked participants to code two series of six practice statements, providing feedback after each one (Jacobson et al., 2017). As with the codebook, we anticipate that adapting a training task should enhance crowdcoding performance.

Finally, we anticipate that there could be a multiplicative effect, whereby providing both a codebook and training would outperform the sum of the benefits of the individual elements. We test these predictions using a simple experimental design, manipulating which types of instructions we present to participants and observing their performance coding a few dozen statements.

This experimental design provides a rigorous test of the mechanisms of interest. In exchange for that control, however, we sacrifice ecological validity. The experiment fails to replicate several important attributes of crowdcoding. Crowdsourced labor is characterized by high variability in workers' willingness to complete a task: while most workers only code a few messages, a small number will code many more (Ipeirotis, 2010). In our experiments, every participant codes the same number of posts. Another defining attribute of crowdsourcing is the use of majority-rule decision making. This strategy is generally shunned in conventional content analysis—well trained analysts using an effective codebook should each arrive at the same assessment when working independently (Neuendorf, 2016, p. 158). However, aggregating the judgements of non-experts is a key mechanism to use the "wisdom of the crowd" (Surowiecki, 2004).

Furthermore, among the most commonly cited advantages of crowdsourcing are the rapid speed and low cost with which large jobs can be completed.

In order to more fully understand the impact of the various crowdcoding instruction strategies tested here, we turn to simulation. Data collected through our experiments provides a basis for simulating a variety of other worker configurations, including a skewed distribution of worker engagement and varying the number of raters per post. We use these simulations to help us understand which types of instructions and which group configurations work best.

Experiments

We conducted a pair of experiments. Both studies used the same 2 (codebook vs. no codebook) X 2 (training vs. no training) between-participant factorial design, but they relied on different participant recruitment strategies. We compare participants' performance in both conditions in terms of the accuracy relative to ground truth coding created using conventional content analysis methods. As noted, although our experiment design differs from conventional crowdcoding in important ways, the control that it affords helps to ensure that observed differences are due to our treatment.

Participants

Study 1.

For the first study, we recruited participants from a student participant pool at Ohio State University. The pool is composed of undergraduate students who participate in research to earn class credit. Two hundred and forty individuals consented to participate, 233 of whom completed the study. Eight individuals participated more than once; only their first attempt was retained. We also excluded three participants who took more than ten hours to complete the study, leaving 217 cases.

Study 2.

For the second study, we recruited participants using Amazon's crowdsourcing service, Mechanical Turk (MTurk). This platform is regularly used in crowdcoding studies, and past work demonstrates the ability of crowdworkers to produce high-quality labels (Budak et al., 2016; Callison-Burch, 2009; Sorokin and Forsyth, 2008). Experiments done with crowdworkers also reliably replicate the behavior of undergraduate students across a wide variety of behavioral experiments (Berinsky et al., 2012; Buhrmester et al., 2011; Goodman et al., 2013; Mason and Suri, 2012; Paolacci et al., 2010).

We posted a Human Intelligence Task (HIT) briefly describing the study to workers living in the U.S. who had previously completed at least 500 HITs with a minimum approval rating of 98%. The incentive for participating was a $2.50 payment. Seven hundred and eighty-three individuals consented to participate, of which 677 completed the study.

Design and Stimuli

We randomly assigned participants to one of four conditions: codebook, training, codebook plus training, and control (neither codebook nor training). Regardless of condition, participants coded 30 posts as either uncivil or civil (see Appendix A for coding task and Appendix D for post wording). Each post was displayed on a new screen with text describing the label choices shown below it. Each choice was accompanied by a brief description. An uncivil post was described as one that "criticizes other people or their ideas in insulting ways, argues that people who disagree with them should suffer, or tells them to stop expressing opinions." A civil post, in contrast, "takes a stance, even a strong one, without attacking people who disagree or arguing that they should stop talking."

We selected the 30 posts from a collection of several thousand messages shared on social media (e.g., Reddit) and online news sites (e.g., the *New York Times*) published in 2016 and 2017. We selected an equal number of civil and uncivil posts (15 each), choosing messages that varied in terms of the effort required to reach consensus during the coding process. These posts had been previously coded by a team of trained coders using a codebook developed by two of the authors based on codebooks used in other high-profile studies of incivility (Coe et al., 2014; Stromer-Galley, 2007; Stroud et al., 2015). Civility was coded dichotomously. Four undergraduate students, led by one of the authors, were trained on the codebook (see Appendix B), working independently to assign codes to the posts before meeting to discuss any disagreements. It is also worth noting that while the selection of our training and test items was purposeful, we could have constructed these datasets in numerous other ways.

After training, the group achieved an acceptable inter-rater reliability (Krippendorff alpha = .77). All coding discrepancies were resolved through discussion, with coders on each side making their case to the group, using the codebook to support their argument, until consensus was reached.[1] The resulting codes provide our ground truth. We mark each code assigned by a participant as "accurate" if it matches the team's assessment.

Procedure

Both studies were administered online using Qualtrics survey software, and each took about 15 minutes to complete (Study 1: $M = 14.5$, $SD = 13.1$; Study 2: $M = 14.4$, $SD = 11.7$). After obtaining consent, we randomly assigned participants to one of four conditions. Regardless of condition, all participants were asked to code 30 comments, which were presented in random order, as either civil or uncivil.

Participants in the codebook and the codebook plus training condition were

presented with a longer version of the codebook upfront followed by a short-form "cheat sheet" for use during the coding task. First, they were presented a brief codebook (572 words) displayed on a single-screen (see Appendix B). This was, verbatim, the codebook used by the team of coders responsible for the ground-truth coding. This document included a definition of incivility, it identified several attributes that make a message uncivil, and it included examples of each attribute. The codebook also included a list of common sources of error, such as attributes that may appear uncivil, but that do not meet our definition of incivility. In this condition, we also included a brief synopsis of the codebook that was specifically adapted to the crowdcoding context, which was shown above each post being coded (see Appendix B). This "cheat sheet" listed several key characteristics used to distinguish between civil and uncivil messages. Participants could mouse over each characteristic to see examples.

Participants in the training condition were asked to practice coding seven posts before moving on to the primary coding task. We presented five uncivil posts and two that were civil. We selected these training posts with the goal of illustrating the key features of (in)civility described in the codebook. After coding each message, participants were provided with feedback on their performance—was the code they assigned correct or not—along with a detailed justification for the correct code (see Appendix C for practice statements and accompanying justifications). At the end of the training session, we informed participants that they would no longer receive feedback on their coding, we reminded them of the characteristics that distinguish civil from uncivil messages, and we encouraged them to be careful going forward.

Dependent Variable

The accuracy of participants' coding of each message is our dependent variable. We assess accuracy by comparing a participant-assigned code to the code assigned by

our team of trained coders (as described in the Design and Stimuli section). When the codes match, the code is labeled "accurate." (See Figure S1 for distributions of participants' average accuracy across the 30 statements for both studies.) Accuracy is the most common way that crowdsourced content analysis approaches are evaluated in the literature (e.g. Kazai et al., 2012; Mitra et al., 2015). However, given the long-standing centrality of inter-rater reliability measures to evaluate content analysis, we also report on inter-rater reliability, measured through Krippendorff alpha. We caution the reader, however. The set of items labeled in our experiments are not a random sample of comments from social media platforms. Instead, they are chosen intentionally to test the ability of labelers to learn and apply this latent concept. Given the challenging nature of the items selected, a lower inter-rater reliability should be expected.

Experiment Results

In both experiments, each participant coded multiple statements, violating the assumption that observations are independent. We use multilevel modeling to account for this, using a random-intercept logistic regression model to estimate the influence of treatment condition on whether a post was coded accurately.

Study 1.

Visual inspection of the accuracy predictions for the first study reveals a pattern that is consistent with our prediction that instruction can be helpful, but estimates are very noisy (see Figure S2). The average treatment effect, however, was non-significant. The four approaches to preparing participants for the coding task yielded statistically indistinguishable levels of accuracy (see Table S1). There were, however, significant differences across conditions after taking into account whether the statement

being assessed was civil or uncivil. Both the training-only condition, $OR = 1.33$, $p = .02$, and the codebook plus training condition, $OR = 1.31$, $p = .02$, resulted in more accurate coding of uncivil posts than the control condition. Furthermore, all three conditions were less accurate than the control when assessing civil posts: codebook-only, $OR = .70$, $p = .03$, training-only, $OR = .63$, $p = .004$, and codebook plus training, $OR = .58$, $p < .001$ (see Fig. 1(a) and Table S2). We do not observe an improvement in inter-rater reliability by condition. Krippendorff Alpha measures for the baseline, codebook, training, and codebook plus training are 0.25, 0.21, 0.25, and 0.24, respectively. We return to these differences in the discussion.

Study 2

The average treatment effect was significant in study two. All three treatment conditions produced more accurate coding than the control: codebook-only, $OR = 1.18$, $p = .002$, training-only, $OR = 1.33$, $p < .001$, and codebook plus training, $OR = 1.37$, $p < .001$ (see Table S3 and Figure S2). Estimated accuracy ranged from .67 (in the control condition) to .73 (in the codebook plus training condition). As in study 1, these effects were conditioned on the civility of the post. All three treatments improved accuracy for uncivil posts and reduced accuracy for civil posts (see Fig. 1(b) and Table S4). We observe only a slight improvement in inter-rater reliability. Krippendorff Alpha measures for the baseline, codebook, training, and codebook plus training are 0.36, 0.38, 0.38, and 0.38, respectively.

Summary of Experiment Results

Taken together, the two studies suggest that providing individuals with more guidance during the coding process improves overall performance, but that this comes at a price. Both the codebook and the training exercise made participants more likely to code all

messages as uncivil, raising accuracy for uncivil posts while lowering it for civil posts. Still the net effect of training in Study 2 was positive, and although Study 1 results were non-significant, the pattern looks similar.

Simulation through Bootstrap Sampling

Our experiments provide a rigorous test of how instruction types affect (in)civility coding. However, the design fails to capture several important attributes of the crowdcoding approach. Most notably, traditional coding techniques rely on a pre-determined set of human judges who are assigned a fixed set of posts to code. With crowdcoding, however, a large but indeterminate number of human judges are invited to contribute codes, and individuals determine how much content they will code based on their availability and interest. Since workers do not get paid for partial effort on crowdsourcing platforms such as MTurk, longer tasks require workers to make bigger commitments. For instance, if a HIT is composed of 30 posts to be coded, as ours was, the worker must complete all 30 posts to be paid. This approach is inconsistent with the microtransaction model that is integral to most crowdsourcing platforms. Instead, researchers typically break tasks into smaller pieces. In crowdcoding, that means that each HIT corresponds to coding a single post (e.g., Wulczyn et al., 2017). This allows researchers to capitalize on the long tail of workers willing to work on smaller tasks, rather than being limited to the smaller set of workers willing to complete long tasks.

How would different types of instruction strategies work under such a scheme? Consider, for instance, the training condition. Training can be costly since each worker must complete several practice posts before providing a single novel code. This cost is amortized if individuals code a large number of posts after their training; however, the cost could be exorbitant if a large number of workers drop out after coding only a few posts post-training. This highlights the necessity of examining the cost of instruction

strategies using more realistic (i.e., shorter) HITs and more variability in how many tasks each worker is willing to complete.

There is also a relationship between the number of coders per post and code accuracy. One important decision the researcher makes when publishing a HIT on MTurk is how many workers to request for each HIT. Suppose the goal was to assess the civility of a particular subreddit. To improve the estimate, the researcher can either have more subreddit posts coded or can increase the number of independent evaluations for each post. Past work shows that there is a trade-off between these two strategies and the best strategy depends on the quality of labels required (Ipeirotis et al., 2014). Assuming that the instruction strategies will affect this quality, we are left to wonder: How many workers should be assigned to each civility-coding HIT and does this vary across different types of coder instruction strategies (i.e., codebook, training, or both)?

We address these questions by simulating crowdcoding using data collected through our randomized experiments. We introduce a simple simulation model to:

(1) Examine the effect of group size (how many coders per post) on task accuracy and speed/cost,
(2) Examine the effect of the worker pool (student versus crowdworker) on task accuracy and speed/cost,
(3) Identify the relationship between participation heterogeneity (variation in how many posts each worker codes) and task speed/cost, and
(4) Build a unified model to determine the best instruction practices.

There are four parameters in our simple simulation model: (i) the number of posts to code, (ii) the worker population size, (iii) the number of workers per post specified by the requester, (iv) variance in participation by the workers.

The first two parameters are fixed and directly learned from the data collected

through our experiments. The number of posts to code (denoted as m) is set to 30 in each simulation. We set the population size (denoted as n) to 50—the number of participants in the condition with the smallest size across the two studies. Fixing the population size across conditions and studies allows us to provide a fair comparison between conditions and studies.

The third parameter is similarly straightforward. As noted before, the researcher ("requester" in MTurk parlance) specifies how many unique workers are assigned to each HIT. In our simulation, this determines the number of coding instances that are retrieved per post. Studies commonly set this to an odd value (e.g., three or five) and use majority voting to determine how the post should be coded. The cost of the coding task increases linearly with this parameter. Past work shows that the expected accuracy obtained from a majority vote also increases, albeit sub-linearly (Ipeirotis et al., 2014). In our simulations we vary this number, denoted as k, between 1, 3, 5, ..., 31.

The fourth parameter helps us characterize the heterogeneity of participation by workers in our experiment. As noted, the assignment of HITs to workers is driven by the availability and preferences of workers in the marketplace. Prior scholarship shows that there is a long tail of individuals who have significantly lower activity than the top contributors (Difallah et al., 2018; Ipeirotis, 2010). Following the findings of these studies, we model the tendency to participate using a log-normal distribution, where the standard deviation parameter (σ) dictates the skew of the distribution. We set the mean (μ) to zero and test two different σ values in our experiments: $\sigma = 0$, which generates a uniform distribution, and $\sigma = 2$, which generates a highly skewed distribution. This allows us to determine the impact of worker participation heterogeneity on the cost of civility coding.

Using these four parameters and our experimental data, we simulate the civility

coding process through bootstrap sampling. Consider, for instance, a configuration in which each post is inspected by three workers ($k = 3$) instructed through training (only), drawn from crowdworkers in a setting where workers have high participation heterogeneity ($\sigma = 2$). What would be the result of such a coding process?

We simulate this scenario as follows: We first sub-sample 50 coders out of the 167 in the entire group of crowdworkers assigned to the training condition. This step ensures the comparability of simulations across the two studies and different conditions. Next, we generate a log-normal dataset with a standard deviation of 2 and size of 50 to characterize each worker's tendency to participate. Next, for each of the 30 posts to code, we sample $k = 3$ workers without replacement with weights proportional to the generated participation tendency dataset. This bootstrapping technique identifies, for each post, the set of workers who would accept the HIT. If we denote this set of workers as S_i, then the set of workers in that simulation is then $S = \bigcup_{i=1}^{30} S_i$.

Given the set of workers assigned to each post, we can easily compute the average accuracy across 30 posts as follows: For each post, we identify the code (civil or uncivil) provided by each worker assigned to it and we compute the majority vote. We mark that post as correctly coded if the majority vote matches the ground truth.

Requesters on MTurk commonly set HIT payment rates to match a particular hourly rate. Indeed, workers commonly discuss the pay rate of HITs in worker forums and suggest HITs to fellow workers accordingly. Therefore, we expect the cost of a coding task to correspond in a linear fashion with the total time workers spends on the task. As such, the cost of the coding task is computed as $\sum_{i \in S} c_{train,i} + c_{code,i} * n_i$ where $c_{train,i}$ is the cost of instructing worker i (in other words, the amount of time i takes to complete the training), $c_{code,i}$ is the average time i spends coding a single post, and n_i is the number of posts i codes in that particular simulation. Given the

participation-tendency skew (controlled by σ), we are likely to observe a large number of workers who only code one post ($n_i = 1$) and a small number of workers that work on a substantial fraction of the 30 posts (large n_i).

We repeat this process 1,000 times to identify expected accuracy and timing, as well as confidence intervals for both measures.

Simulation Results

Figure 2 provides a summary of crowdcoding accuracy in the simulations. We provide two figures corresponding to simulations on (a) crowdworkers with uniform participation and (b) students with uniform participation assumptions. Plots generated assuming a skewed distribution are nearly identical and are omitted for brevity. Skewed participation only affects the timing (i.e., cost) of coding since participation tendency is independent from worker accuracy (see Figure S5 for all time estimates). We refer interested readers to Supplementary Materials for inter-rater reliability simulation results.

We start by noting the similarity of results across the two plots shown in Figure 2. Perhaps most striking is the relative strength of training over no training. Training consistently outperforms other options in terms of accuracy when holding group size constant. If, for instance, a researcher aims to achieve an accuracy measure of at least 0.75 (y-axis) using crowdworkers, the most efficient option is to use training with three workers, which achieves an accuracy of 0.77 (see subplot b). Neither the control nor codebook condition ever reach that level of accuracy. The condition that combined the codebook and training is able to reach similar accuracy levels, but this approach takes significantly more time. Under uniform participation assumptions, the training-only condition is expected to take about two and a half hours (9,368 seconds) while training plus codebook would take about 4 hours (14,469 seconds). A similar pattern of

increased costs is observed under the skewed distribution assumption (see Figure S5).

Regarding the model parameters, we make several observations. First, majority voting, as opposed to relying on a single coder, significantly improves performance across all populations and instruction strategies. The benefits of this approach, however, level off as group size grows. In the control condition with crowdworkers, a group size of five is optimal; after that, group performance begins to decline. Indeed, performance of the largest groups is comparable to that of a single worker. For the other conditions, there is no penalty for large groups, but we do observe diminishing returns. For example, increasing the group size from $k = 1$ to $k = 3$, we achieve at least a 0.05 increase in the training condition across both sub-figures (e.g. 0.73 to 0.78 for Figure 2(a)). The increase from $k = 29$ to $k = 31$ is insignificant.

Next, we consider differences between the students or crowdworkers, by comparing Figures 2(a) and 2(b). Notably, the accuracy profiles are distinct across the two populations. For instance, increasing group size significantly improves all conditions with the student sample while the effect is flat for the control group among crowdworkers.[2] We also observe that the student population has a higher accuracy ceiling. The cost profiles are, however, similar for the two populations (see Figure S5).

Thus far, our focus has been primarily on accuracy. We observed that training outperforms other techniques but generally takes longer, especially when compared to the control group. How does that cost vary with the skew of participation? To answer this question, we compare the simulations under the two participation distributions. As expected, distribution does not influence accuracy; it only effects the time required to complete the work. This effect is observed for tasks with high start-up costs (codebook and training). In skewed participation simulations, coders with high participation tendencies complete a large number of tasks and therefore amortize the start-up cost of

the instruction process. As noted, the coding task is expected to take two and a half hours in total for the training condition with three crowdworkers ($k = 3$) when participation is uniform. The same task is expected to take almost an hour less (5,978 seconds or about an hour and forty minutes) when participation is skewed. As we increase the group size (e.g., $k = 31$), the cost differences between the two participation assumptions disappear. Such large groups would almost certainly result in each worker being picked at least once in each simulation, irrespective of the participation tendencies, leading to comparable costs.

While simulations using data from students and those based on uniform-participation among crowdworkers provide informative reference points, simulations of crowdworkers with a skewed participation distribution are closest to the types of tasks researchers are likely to implement when using crowdcoding. Therefore, we conclude with a simple summary of the best strategy under these conditions. We identify the optimal strategy using utility, which we define in terms of a researcher's willingness to pay (WTP) for a 1% increase in accuracy. For instance, a WTP of \$10 means that the researcher is willing to pay \$750 for a coding task with 75% accuracy, whereas a WTP of \$1 corresponds to a budget of \$75 for the same accuracy (regardless of how many items are to be coded).

For a given WTP, we can compute the utility of each crowdcoding approach as $WTP * accuracy - codingCost$. We compute *codingCost* by multiplying the task completion time across all coders (in hours) by a \$15/hour pay rate. We then vary WTP and identify the best strategy both in terms of instruction strategy and the optimal number of coders. Results are given in Figure 3. We see that, for a very low WTP (\$1), the best strategy is to forego instruction altogether.

In the vast majority of cases, however, the best strategy is the training-only approach. Importantly, neither codebook-only nor codebook plus training is *ever* the

optimal choice, regardless of how much the researcher is willing to pay for accurate coding. Although Figure 2 shows that the returns for increasing the number of coders per item diminish for large *k*, increasing group size is still appropriate if WTP is high enough. For instance, for a WTP of $50 (which amounts to a valuation of $3,750 for 75% accuracy), the optimal utility is reached through training with 25 coders per item.

Discussion

When coding a latent concept such as civility, experiments and simulations indicate that crowdcoding without explicitly teaching workers how to complete the task is almost never the best option. Even when budgets are tight, training can improve accuracy. Of course, financial constraints are inevitable, and it is up to researchers to decide if improving accuracy by a few percent is worth the additional expense. For example, if a $100 premium to maximize accuracy is outside a project's budget, the best option is to forego instruction. But given that accurate coding is a defining goal of content analysis, our results suggest that training will tend to be a better choice.

Although training does tend to improve accuracy, its influence plays out in complicated ways. Perhaps most notable is the fact that training increased participants' willingness to label any post uncivil. Consequently, their accuracy when labeling civil items declined slightly while their performance on uncivil items improved. In short, this reveals an important trade-off between false-positives and false-negatives when coding (in)civility. Most importantly, though, the net effect was positive.

Although the value of instructing coders may sound obvious to scholars trained in conventional content analysis, it is a striking contrast to current crowdcoding practices. Most crowdcoding provides little guidance beyond the labels themselves, yet our simulation suggests that even a short training exercise often yields accuracy improvements of 10% or better. This is a notable payoff for an easy-to-implement and

relatively inexpensive change in how researchers teach workers about the crowdcoding task. At the same time, in contrast to common content analysis practices, the types of training used here is remarkably lightweight. For instance, the training for the incivility labeling process descried by Coe et al. (2014) took 6 weeks to complete.

The ineffectiveness of providing crowdworkers with a codebook compared to having them complete a simple training exercise is also noteworthy. Perhaps surprisingly, combining training with a codebook is *not* a good option, at least when coding civility. Adding a codebook increases costs without improving accuracy. This finding is striking given the reliance on codebooks in past work. This is not to say that codebooks are irrelevant. To the contrary, they remain an important tool by which researchers articulate and document the concept of interest (Krippendorff, 2019), and in the context of crowdcoding, having a codebook may facilitate the selection of training items. But providing crowdworkers with a codebook appears largely ineffective.

It is worthwhile to briefly comment on the inter-rater reliability patterns observed throughout our experiments. Inter-rater reliability, measured using Krippendorff alpha, were below the threshold used for conventional content analysis, but they are typical of other tests of crowdcoding (e.g., Lind et al., 2017). This is likely to be a by-product of our item selection process. Items were selected to provide a rigorous test of crowdworkers' ability to provide accurate labels, even for content that conventionally trained coders found difficult. We do not observe a consistent improvement in inter-rater reliability measures across conditions. While there is a significant improvement in *accuracy* when training is compared to baseline in study 1, for instance, no improvement is observed in the alpha scores. This shows that a higher accuracy does not necessarily translate to higher inter-rater reliability, and vice versa.

In practice, our recommendations will make crowdcoding somewhat more difficult. For example, providing additional instructions requires that the process be split into two parts (e.g., training and coding), with one functioning as a prerequisite. But the task is not so difficult as to be an obstacle to conducting this type of research. Indeed, all major crowdsourcing platforms support adding qualification tasks (e.g., training) to a HIT (e.g., coding a post). For example, see blog.mturk.com/tutorial-identifying-workers-that-will-be-good-at-your-task-66dccb92b42f).

In keeping with other work on crowdcoding (e.g., Guo et al, 2019), but contrary to the best practices typically associated with conventional content analysis, our empirical evidence also suggests that using majority rule with an odd number of coders is a good choice when crowdcoding. Exactly how many workers to use depends on how much value the research team places on accuracy, but the biggest improvement comes when increasing group size to three or five. For example, the difference between one coder and three is considerably larger than the difference between 29 and 31.

Reflecting on why majority rule is appropriate in crowdcoding and not in traditional content analysis is informative. We suspect that the reason is related to differences in the composition of the groups responsible for coding. Given a small set of coders, it is infeasible to collect large numbers of codes for every post, especially in a large dataset. Furthermore, reliance on the same small group of people for every code creates risk that the results will reflect idiosyncratic biases of the group. Requiring that coders assign the same codes even when they work independently helps ensure that data are not biased in this way. With crowdcoding, however, the pool of potential coders is much larger, providing an alternative mechanism for keeping biases in check. Majority rule with a diverse and changing set of coders helps to ensure that errors and biases cancel one another out. This is particularly important when coding latent concepts, such

as incivility, which are more likely to be influenced by contextual factors, including culture, personal experience, or education (Aroyo et al., 2019).

This logic also suggests that when using majority rule, researchers must ensure that the set of workers recruited to the coding task is large and heterogeneous. Crowdcoding with small or homogeneous groups of workers poses the same risks found in conventional content analysis. That is, the codes may be unduly swayed by workers' biases. There are a variety of ways to ensure crowdworker diversity. One mechanism that may be useful when coding political content is to use quotas to ensure partisan diversity among coders (e.g., see Budak et al., 2016).

It is possible that majority rule is uniquely effective for crowdcoding civility precisely because the concept is understood so differently by different people (Kenski et al., 2020). If perceptions of what makes a statement uncivil were more consistent, the benefit of this approach might be smaller. Still, the literature on content analysis makes it clear that divergent understanding of seemingly simple concepts is common (Krippendorff, 2019), suggesting that our recommended approach is widely applicable.

Another striking pattern in our data concern differences between students and crowdworkers. Among student coders, increasing group size helped in all conditions. Among crowdworkers, in contrast, increasing group size was uniquely beneficial when training was provided. We speculate that this may be because without training, coding errors are more highly correlated among crowdworkers than among students. It seems to us plausible that MTurk workers have developed a highly consistent set of labeling habits through their on-going work in the space. In such a context, group size would have little influence on accuracy. Fortunately, it appears that training changes this pattern. When it is provided, increasing group size yields significant dividends.

As with all research, this work has important limitations. The experiments are conducted using a single latent concept. Civility is widely studied, making it a useful test case, and we have no reason to expect the patterns observed here to be bound to this concept. Still, replicating with other concepts would be valuable. We also acknowledge that the test corpus is considerably smaller than would be found in a real content analysis. Indeed, a primary appeal of crowdcoding is that it is scalable to very large datasets. The lack of ecological validity does not render our results meaningless, but it is an important limitation. Constraining the number of items on which accuracy scores are computed means, among other things, that we are unable to detect any potential longer-term benefits of the training task. Similarly, the small curated corpus may lack some of the diversity that would be observed in a real-world dataset. Replicating this approach on a larger scale would be useful. Finally, we do not know how dependent these results are on the specific training items selected. Although their selection was purposeful, the set of potential alternative training items is infinite. It is likely that other training items (e.g., a different mix of civil and uncivil items) could influence the effectiveness of the approach. Further refinement of the process of selecting training items is needed.

Looking to the future, we challenge the research community to consider ways in which we can further improve crowdcoding. Can we improve the training task? For example, perhaps there are different types of training, more systematic ways of identifying training items, or better ways of providing feedback to coders. Perhaps scholars with a background in computer science can find ways to enhance training through automatic tailoring. For example, can a system be devised to automatically vary the set of training items presented based on the types of mistakes a worker makes? And what about the coding task itself? Can we recreate the interactive aspects of in-person

training in a crowdcoding environment, allowing workers to talk with one another as they learn? Would it be useful to continue to provide periodic feedback throughout the coding process, reinforcing the training conducted at the start of the task? Other factors that shape the effectiveness of crowdcoding may be harder to control, such as the level of measurement or whether the concept is latent or manifest, as these factors are strongly informed by the research question. But we should continue to explore other ways that we may be able to improve crowdcoding. Crowdcoding is still in its infancy, and lessons learned now have the potential to pay big dividends moving forward.

Conclusions

This research builds on a growing body of scholarship concerned with analyzing text at scale. Content analysis is a challenging task, and the performance achieved here is encouraging. Even with modest sized groups (three to five coders per item) coding after training frequently achieved 75%+ accuracy. We identify two important strategies for improving accuracy. First, crowdcoding works better when workers are required to complete a brief training exercise at the start of the coding process. Second, crowdcoding—in contrast to traditional content analysis—is more accurate when several workers code every item and majority rule is applied. These recommendations are relatively simple and inexpensive to implement.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant Nos. IIS-1717688 and IIS-1717965. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. Thanks to Kate Kenski, Teresa Lynch, and Ashely Muddiman for valuable feedback on earlier versions

of this manuscript, to Craig Bossley, Logan Fadeley, Maggie Sullivan, and Kelsey Yappel for preparing the ground-truth coding, and to Ashwin Rajadesingan for developing the online coding tool.

Notes

[1] When generating the ground truth codes, posts about which coders disagreed were somewhat more likely to be coded as uncivil after discussion. One of the most common sources of disagreement was coders' differing familiarity with the political environment. For example, politically sophisticated coders were more likely to recognize political misinformation. Other common sources of errors included the use of derogatory slang unfamiliar to some coders, and brief uncivil expressions embedded in lengthy messages that were otherwise civil.

[2] The accuracy increases slightly for $k = 3$ and then drops. While a drop in accuracy as we increase group size is counter-intuitive, it is theoretically possible. If, for instance, only two out of 31 coders have the right answer to a given question, increasing the group size from three to five is guaranteed to result in a drop in expected accuracy. This is because it is impossible to achieve non-zero accuracy for the majority vote when at most two out of five people can have the right answer.

References

Aroyo, L., Dixon, L., Thain, N., Redfield, O., and Rosen, R. (2019). Crowdsourcing subjective tasks: The case study of understanding toxicity in online discussions. In *Companion Proceedings of the 2019 World Wide Web Conference*, pages 1100–1105. ACM.

Benoit, K., Conway, D., Lauderdale, B. E., Laver, M., and Mikhaylov, S. (2016). Crowd-sourced text analysis: Reproducible and agile production of political data. *American Political Science Review*, 110(2):278–295.

Berinsky, A. J., Huber, G. A., and Lenz, G. S. (2012). Evaluating online labor markets for experimental research: Amazon.com's Mechanical Turk. *Political Analysis*, 20(3):351–368.

Budak, C., Goel, S., and Rao, J. M. (2016). Fair and balanced? quantifying media bias through crowdsourced content analysis. *Public Opinion Quarterly*, 80(S1):250–271.

Buhrmester, M., Kwang, T., and Gosling, S. D. (2011). Amazon's mechanical turk a new

source of inexpensive, yet high-quality, data? *Perspectives on Psychological Science*, 6(1):3–5.

Callison-Burch, C. (2009). Fast, cheap, and creative: evaluating translation quality using amazon's mechanical turk. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, volume 1, pages 286–295. Association for Computational Linguistics.

Center, P. R. (2016). The political environment on social media. Report, Pew Research Center. Coe, K., Kenski, K., and Rains, S. A. (2014). Online and uncivil? patterns and determinants of incivility in newspaper website comments. *Journal of Communication*, 64(4):658–679.

Difallah, D., Filatova, E., and Ipeirotis, P. (2018). Demographics and dynamics of mechanical turk workers. In *Proceedings of the eleventh ACM international conference on web search and data mining*, pages 135–143.

Gardiner, B., Mansfield, M., Anderson, I., Holder, J., Louter, D., and Ulmanu, M. (2016). The Dark Side of Guardian Comments. *The Guardian, 2016.*

Gervais, B. T. (2015). Incivility Online: Affective and Behavioral Reactions to Uncivil Political Posts in a Web-based Experiment. *Journal of Information Technology & Politics*, 12(2):167– 185.

Goodman, J. K., Cryder, C. E., and Cheema, A. (2013). Data collection in a flat world: The strengths and weaknesses of Mechanical Turk samples. *Journal of Behavioral Decision Making*, 26(3):213–224.

Guo, L., Mays, K., Lai, S., Jalal, M., Ishwar, P., & Betke, M. (2019). Accurate, Fast, But Not Always Cheap: Evaluating "Crowdcoding" as an Alternative Approach to Analyze Social Media Data. *Journalism & Mass Communication Quarterly, 97*(3), 811-834. https://doi.org/10.1177/1077699019891437

Graham, T. and Wright, S. (2015). A tale of two stories from "below the line": Comment fields at the guardian. *The International Journal of Press/Politics*, 20(3):317–338.

Grimmer, J. and Stewart, B. M. (2013). Text as data: The promise and pitfalls of automatic content analysis methods for political texts. *Political Analysis*, 21(3):267–297.

Horn, A. (2019). Can the online crowd match real expert judgments? how task complexity and coder location affect the validity of crowd-coded data. *European Journal of Political Research*, 58(1):236–247.

Ipeirotis, P. G. (2010). Analyzing the amazon mechanical turk marketplace. *XRDS: Cross- roads, The ACM Magazine for Students*, 17(2):16–21.

Ipeirotis, P. G., Provost, F., Sheng, V. S., and Wang, J. (2014). Repeated labeling using multiple noisy labelers. *Data Mining and Knowledge Discovery*, 28(2):402–441.

Jacobson, M. R., Whyte, C. E., and Azzam, T. (2017). Using crowdsourcing to code open-ended responses: A mixed methods approach. *American Journal of Evaluation*, 39(3):413–429.

Kazai, G., Kamps, J., & Milic-Frayling, N. (2012). The face of quality in crowdsourcing relevance labels: Demographics, personality and labeling accuracy. In *Proceedings of the 21st ACM international conference on Information and knowledge management* (pp. 2583-2586).

Kenski, K., Coe, K., and Rains, S. A. (2020). Perceptions of uncivil discourse online: An examination of types and predictors. *Communication Research*, 47(6):795–814.

Krippendorff, K. (2019). *Content Analysis: An Introduction to Its Methodology*. Sage Publications, Los Angeles, fourth edition.

Lind, F., Gruber, M., and Boomgaarden, H. G. (2017). Content analysis by the crowd: Assessing the usability of crowdsourcing for coding latent constructs. *Communication Methods and Measures*, 11(3):191–209.

Mason, W. and Suri, S. (2012). Conducting behavioral research on amazon's mechanical turk. *Behavior Research Methods*, 44(1):1–23.

Mitra, T., Hutto, C. J., & Gilbert, E. (2015). Comparing person-and process-centric strategies for obtaining quality data on amazon mechanical turk. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (pp. 1345-1354).

Neuendorf, K. A. (2016). *The content analysis guidebook.* Sage Publications, Thousand Oaks, CAs, 2nd edition.

Paolacci, G., Chandler, J., and Ipeirotis, P. G. (2010). Running experiments on amazon mechanical turk. *Judgment and Decision making*, 5(5):411–419.

Papacharissi, Z. (2004). Democracy online: civility, politeness, and the democratic potential of online political discussion groups. *New Media & Society*, 6(2):259–283.

Perspective (2020). What if technology could help improve conversations online? https://www.perspectiveapi.com/

Sadeque, F., Rains, S., Shmargad, Y., Kenski, K., Coe, K., and Bethard, S. (2019). Incivility detection in online comments. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 283–291, Minneapolis, Minnesota. Association for Computational Linguistics.

Sorokin, A., & Forsyth, D. (2008, June). Utility data annotation with amazon mechanical turk. In *2008 IEEE computer society conference on computer vision and pattern recognition workshops* (pp. 1-8). IEEE.

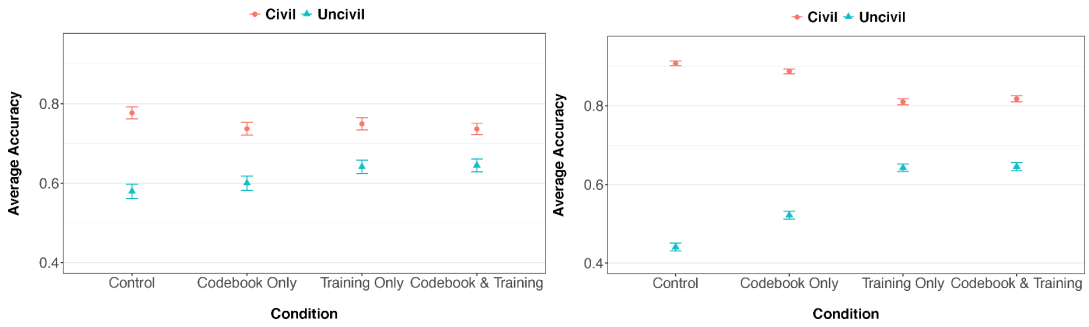
Stromer-Galley, J. (2007). Measuring deliberation's content: A coding scheme. *Journal of Public Deliberation*, 3(1): Article 12.

Stroud, N. J., Scacco, J. M., Muddiman, A., and Curry, A. L. (2015). Changing deliberative norms on news organizations' Facebook sites. *Journal of Computer-Mediated Communication*, 20(2):188–203.

Stryker, R., Conway, B. A., and Danielson, J. T. (2016). What is political incivility? *Communication Monographs*, 83(4):535–556.

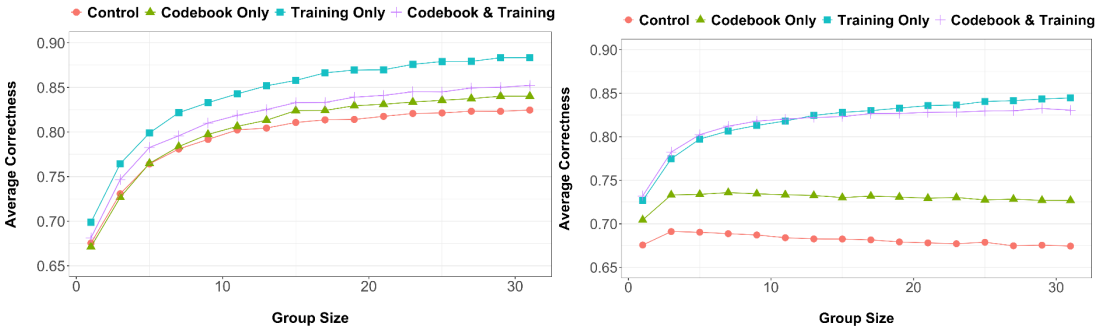
Surowiecki, J. (2004). *The wisdom of crowds: why the many are smarter than the few and how collective wisdom shapes business, economies, societies, and nations*. Doubleday, New York.

Wulczyn, E., Thain, N., and Dixon, L. (2017). Ex machina: Personal attacks seen at scale. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, page 1391–1399, Republic and Canton of Geneva, CHE.



(a) Students **(b) Crowdworkers**

Figure 1. Average accuracy broken down by question type (whether the ground truth code is civil or uncivil).



(a) Students (b) Crowdworkers

Figure 2. Average accuracy for different instruction strategies. We observe, for instance, that expected coding accuracy across all 30 items is roughly 0.68 (x-axis) for both populations when a single worker is assigned to each item for the Control condition (red curve with round markers). Each curve includes 16 points, corresponding to an increase of 2 workers at each step (1, 3, …, 31). Here we show the accuracy results for when the participation is uniform. Skewed participation does not affect accuracy and thus the corresponding plots are omitted for brevity.

Figure 3. What is the best strategy for crowdcoding on MTurk given a particular Willingness To Pay (WTP)? WTP corresponds to how much a research is willing to pay for each 1% increase in accuracy across entire coding task. For instance, a WTP of $50 means that the researcher is willing to pay $50 for each 1% increase in accuracy and therefor has a maximum budget of $5,000. This plot summarizes the two components of the strategy: Which instruction strategy to use (top sub-figure) and how many individuals code each item (bottom subfigure).