Shreddr: pipelined paper digitization for low-resource organizations

Kuang Chen
Dept. of EECS
UC Berkeley
kuangc@cs.berkeley.edu

Akshay Kannan
Captricity, Inc.
akshayk@captricity.com

Yoriyasu Yano
Captricity, Inc.
yoriy@captricity.com

Joseph M. Hellerstein
Dept. of EECS
UC Berkeley
hellerstein@cs.berkeley.edu

Tapan S. Parikh
School of Information
UC Berkeley
parikh@ischool.berkeley.edu

ABSTRACT

For low-resource organizations working in developing regions, infrastructure and capacity for data collection have not kept pace with the increasing demand for accurate and timely data. Despite continued emphasis and investment, many data collection efforts still suffer from delays, inefficiency and difficulties maintaining quality. Data is often still "stuck" on paper forms, making it unavailable for decision-makers and operational staff. We apply techniques from computer vision, database systems and machine learning, and leverage new infrastructure – online workers and mobile connectivity – to redesign data entry with high data quality. Shreddr delivers self-serve, low-cost and on-demand data entry service allowing low-resource organizations to quickly transform stacks of paper into structured electronic records through a novel combination of optimizations: batch processing and compression techniques from database systems, automatic document processing using computer vision, and value verification through crowd-sourcing. In this paper, we describe Shreddr's design and implementation, and measure system performance with a large-scale evaluation in Mali, where Shreddr was used to enter over a million values from 36,819 pages. Within this case study, we found that Shreddr can significantly decrease the effort and cost of data entry, while maintaining a high level of quality.

1. INTRODUCTION

In many low-resource social service organizations, including those providing community health, micro-finance and other rural development programs in the developing world, collecting accurate and timely information is crucial for improving service delivery, and for monitoring and evaluation (M&E). A number of existing challenges—including limited infrastructure, lack of technical expertise and rapid turnover of technical staff — make data collection time-consuming, error-prone and expensive, especially in the developing world [3].

Several researchers have proposed and developed mobile phone based solutions for data collection [1, 7, 16]. Mobile devices enable remote agents to directly enter information at the point of service, replacing data entry clerks and providing near immediate data availability. However, mobile direct entry usually replace existing paper-based workflows, creating significant training and infrastructure challenges. As a result, going "paperless" is not an option for many organizations [16]. Paper remains the time-tested and preferred data capture medium for many situations, for the following reasons:

- Resource limitations: lack of capital, stable electricity, IT-savvy workers, and system administrators
- Inertia: small changes to deeply ingrained paper-based workflows can increase training costs, errors and lead to apprehension
- Regulation: compliance with paper record-keeping rules from auditing and oversight agencies
- Backup: need to safeguard against power or electronic system failure, and for having a primary source reference [17])

To build end-to-end data infrastructure in these communities, we must address the paper-to-structured-data digitization problem. For many developing world organizations, the default approach for transcribing stacks of paper is to hire on-site data entry workers. However, many organizations cannot afford to hire and retain their own high quality data entry workers. As a result, data is delayed or simply unavailable. For these situations, automated methods of extracting data from paper forms [12] can be immensely beneficial. However, automated form processing methods still have not been widely adopted in the developing world. Automated form processing solutions, where they exist, are still proprietary and difficult to use.

Shreddr is designed as an online, cloud-hosted service to transform paper form images into structured data on demand [3]. Shreddr combines batch processing and compression techniques from database systems, with automatic document processing using computer vision and value verification through crowd-sourced workers. First, the system helps the user extract a paper form's schema and data locations via a web interface. Next, it uses computer vision to align then break up images into constituent image *shreds*. Then, the system orders and groups shreds according to an entropy-based metric, and places the shred groups into *batched* worker interfaces. Lastly, the system assigns work to distributed online workers who iteratively refine shred estimates into final values. While an earlier

DEV '12 March 11-12, Atlanta, GA, USA

paper proposed the general idea, here we describe in detail the design and implementation of Shreddr, including its quality assurance mechanisms and batch worker interfaces. We also measure and analyze system performance through a large-scale case study in Mali, where Shreddr used to enter over one million values from 36,819 pages. In this case study, we found that Shreddr can significantly decrease the amount of effort and cost of entry, while maintaining a high level of quality.

The design of Shreddr advances the current practice in several ways.

1. It allows approximate automation to simplify a large portion of data entry tasks.

2. Working with shreds in the cloud, and a crowd, gives us latitude to control latency and quality at a per-question granularity. For instance, time-sensitive answers can be prioritized, and important answers can be double- or triple-checked.

3. Shredded worker interfaces use cognitive and physical interface *compression* to enable fast, batched data entry and verification.

4. An easy-to-use, web-based interface for managing and performing data entry tasks allows us to make data entry accessible to many more organizations and individuals, at a fraction of the current cost.

In the following sections, we compare and contrast Shreddr to existing approaches, and describe the design and implementation of the Shreddr system. Next, we describe the results we obtained for data quality and effort from a large-case population survey conducted in Mali. Finally, we summarize our findings.

2. RELATED WORK

The most relevant prior work to Shreddr includes research on document recognition, and other systems for optimizing and automating data entry.

2.1 Automatic document processing

One of the first applications of computer technology was for the digitization of paper forms for large enterprises. Platforms like EMC's Captiva [1], and IBM's DataCap [2] can handle millions of forms, ranging from invoices to insurance claims and financial statements for well-resourced organizations. Completely automated end-to-end solutions exist, but are currently financially unattainable for low-resource organizations.

The document processing industry views paper digitization as consisting of three main challenges: (1) recognizing the page being examined, (2) reading values from the page and (3) associating values together across pages [22]. The field of document recognition has provided reliable solutions for a number of these challenges, including: automatic recognition of handwritten numbers, short text with a small number of possible values, and machine produced text. Researchers now focus on issues like reading text in many languages and automatic segmentation and reading of challenging document images. The field, like many enterprise-scale systems, has given relatively little attention to the user-centric requirements and contextual constraints [22] that would make their tools useful in low-resource contexts. Addressing these issues is Shreddr's primary focus.

[1]http://www.emc.com/products/family2/captiva-family.htm

[2]http://www.datacap.com/solutions/applications/forms-processing

2.2 Crowd-sourced digitization

ReCAPTCHA [26] can digitize printed text using a combination of OCR, string-matching and human effort required to gain access to websites and content, aligning website security with document digitization. Shreddr generalizes this approach to arbitrary form data, and includes the ability to recruit paid workers. MicroTask is a commercial crowd-sourced data digitization service [3]. They offer a similar product to Shreddr, but do not offer a "self-serve" interface, dealing primarily with larger scale contracts, competing with more traditional business process outsourcing (BPO) services.

2.3 Improving data entry

Lorie *et al.* proposed an automated data entry system that integrates operator intervention with OCR, including a multi-staged verification process to refine the results [12]. This approach focuses on optimizing human attention for correcting OCR in an enterprise environment. Their iterative results improvement model is similar to some high-level aspects of the Shreddr pipeline. Several evaluations of automatic forms processing systems have also been performed, including several focusing on processing flight coupons [13, 27]. Bubble forms have also been evaluated for serving development organizations. However, this approach was constrained to numeric and categorical data entry [25].

There have also been novel mobile data collection solutions specifically designed for low-resource organizations. For example, systems like CAM and Open Data Kit allow for *in situ* data entry directly on mobile devices [7, 17]. DigitalSlate is an elegant approach that uses pen and paper on top of a customized tablet device, providing immediate electronic feedback to handwritten input [20].

3. EXISTING SOLUTIONS

In this section, we describe the prevailing current approaches to data collection in a development context: direct digital entry, and double entry from paper forms. Examination of their relative strengths and weaknesses will motivate the design of Shreddr, as detailed in the next section.

3.1 Replacing paper with direct-entry

Direct digital entry implies digitally capturing data as close to the source as possible, usually by using a mobile device. Direct entry focuses on improving efficiency and feedback. The thinking goes, if a remote agent can enter values directly into a device, then it saves organizations the work of hiring data entry clerks and transferring stacks of paper. Several groups have demonstrated potential accuracy improvements from on-device feedback to the worker, and faster turn-around time through digitization at the point of collection [5, 7, 23].

However, in many situations paper cannot be replaced. Organizations may desire to retain paper documents for reference [17]. More importantly, adoption of new technology and associated workflow changes can incur significant cost, including training and new hardware and software. For some settings, the initial and on-going costs of such an approach are not sustainable over a reasonable time frame [19]. While the novelty of mobile devices and entry may initially capture users' attention, this interest can wane over the longer-term, particularly if the task involves significant drudgery, including entering complicated and long forms using a constrained device like a mobile phone. Low literacy users have also shown preference for traditional analog information capture mediums such as voice or paper [18]. A study of students doing math problems

[3]http://microtask.com

showed that writing modalities less familiar than pen and paper, increasing student cognitive load and reducing performance [8].

3.2 Traditional double data entry

Double data entry (DDE) refers to the practice having the same values keyed in multiple times by different workers to ensure quality. The two (or more) sets of entries are batch-compared after completion to determine the values that need correction. The comparison-and-fix process is most often assisted by specialized software.

Data quality from manual data entry of paper forms can vary widely, ranging from poor to excellent, depending on the data, the workers and the context. A study of paper data collection in South Africa demonstrated completion rates of only about 50% and accuracy rates of 12.8%. [14]. On the other hand, paper-based clinical trials (for pharmaceutical development) achieve error rates of 0.15% [21], by relying on qualified (and often expensive) workers and well-honed double entry practices.

The user's skill level is critical for accurate and efficient data entry. Factors like experience, job engagement level and subject matter expertise are significant user factors affecting data quality. A single pass from trusted workers with known high accuracy can produce results close to that of DDE [9].

4. SHREDDR

In this section, we describe how the Shreddr system approaches the process of digitizing paper forms.

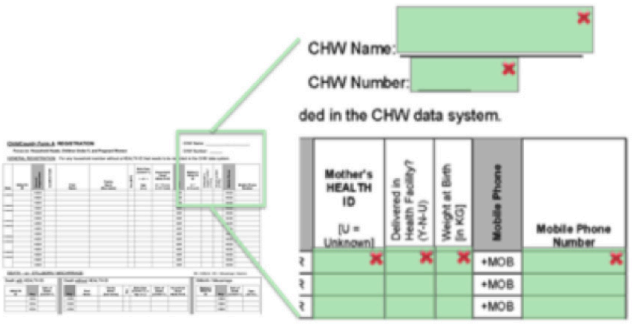


Figure 1: Document definition interface: A form image template on the left; highlighted field areas of interest on the right.

1. First, a user uploads a sample of a form ("template") to the cloud-hosted Shreddr server. Using Shreddr's online, web-based tool, users can highlight data fields and regions of interest (Figure 1). For each field, they provide the system with a few pieces of meta-data, including: name, data-type, unit, significant digits, and possible values. In some cases, this document definition process can also be crowd-sourced [11].

2. Next, users generate images of entered forms ("instances") needing processing and entry. Forms can be imaged using a multitude of methods (e.g. scanner, low-cost digital camera, camera phone, etc.; see Figure 2 for one low-cost approach). Shreddr's quality assurance algorithms ensure accuracy even with low-resolution images (perhaps at additional cost).

3. These images are then uploaded to our website, either individually or using a batch submission interface. This can be done using the website or via email or another file hosting service. Images are aligned and auto-corrected for rotation, lighting, and perspective transformation using computer vision algorithms (Figure 3)

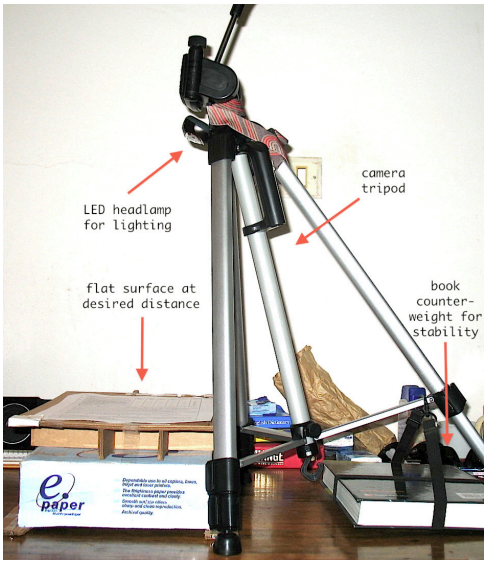


Figure 2: Makeshift copy-stand used to digitize a country-wide survey in Mali.

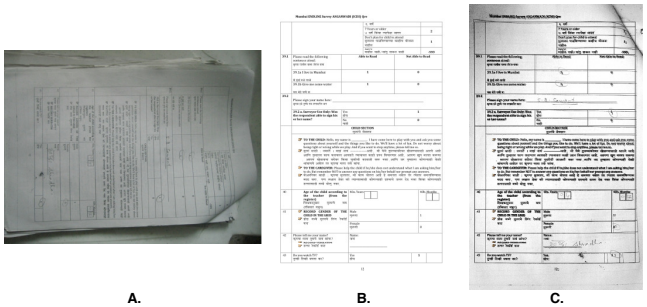


Figure 3: Form image registration outcome: A. The original image, B. The template form, C. the results.

4. The system then "shreds" apart the form images into fragments according to the document definition (Figure 4).

5. Shreds are batched by field and data-type (for example: all the numeric age fields) into groups of 30 - 100, depending on field difficulty. We prompt crowd workers using specific instructions like: "Type in this handwriting" (entry), "Verify our guesses" (verify) and "Correct the spelling" (spell-check). (An example entry interface is found in Figure 5. Each shred is tracked by a *decision plan*—a state machine that decides, as estimates come in from workers, when to confirm an answer (depending on the desired accuracy), or what next question to ask.

6. The system heuristically governs data quality using gold standard values and distinct decision logic, based on the data type and desired quality. Asynchronous daemon processes advance the state of decision plans as estimates arrive from workers. We provide more details about the Shreddr quality

Figure 4: Shredding: form image on the left; "shredded" fragments on the right.

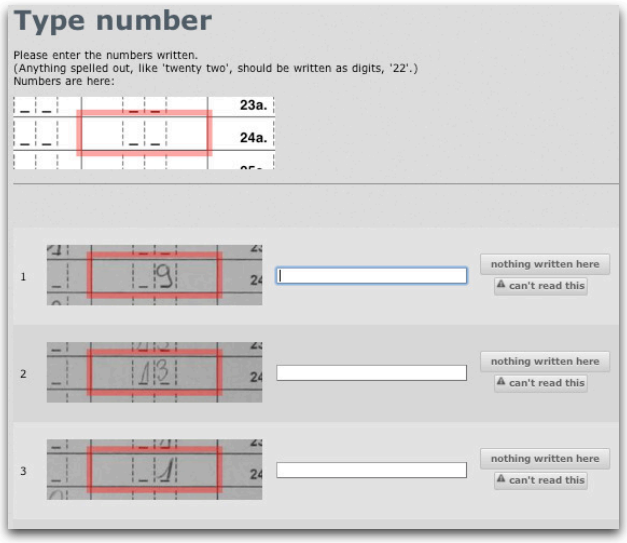


Figure 5: Entry interface.

control algorithms and interfaces, along with their observed performance, in the following sections.

7. Final shred values and the history of decision processes, along with shred and original page images, are securely stored in Shreddr's database. The end user can browse and edit, filter by problematic values (such as those marked for review by data entry operators), and download in CSV format.

5. DESIGN CHOICES

Shreddr is modeled as a data entry assembly line, composed of a series of pipelined digitization steps, interleaving algorithmic control with human input. In this section, we examine several key elements of Shreddr's design.

5.1 Capture and encode separately

Shreddr separates data capture and encoding into distinct steps [3]. The *capturing* step records analog signals from the real world into a persistent digital state, and *encoding* steps iteratively transcribe, clean, translate and annotate the recorded signals according to predefined schemata.

Front-line field workers are the best suited to capture information given their access, local contextual knowledge and familiarity with the community, whereas tasks involving data encoding require more literacy, training and familiarity with specific data vocabularies and schemata. The distinction makes it possible to move tasks involving data encoding to where the knowledge, incentives and capabilities are best suited.

5.2 Data entry as-a-service

Adoption: From the perspective of an organization, data entry typically entails hiring and housing workers or negotiating a contract with a third party data entry provider. With Shreddr, organizations can "subscribe" to the service with little up-front commitment, while incrementally digitizing and re-formulating existing paper-based work flows. Wherever data collection needs to occur, a "data-imager" armed with a single device for imaging and upload is all that is required in the field. Upstream, Shreddr's elastic worker pool can produce structured data from scanned images, within the order of many hours to a few days. These characteristics make it easier for organizations to experiment with and adopt digital solutions for data management and analysis.

Economy of scale: By pooling the work from multiple organizations, Shreddr benefits from an increased economy of scale. A constant flow of work allows Shreddr to further separate and reformulate data encoding tasks in additional assembly line steps. These steps are opportunities for existing and novel techniques to plug in to add value; for example: computer vision algorithms can pre-process document images for legibility.

5.3 Paper independence

Shreddr separates the usage of paper from the usage of data. The main idea, like the concept of *data independence* from the data management literature, is to decouple the organization and layout of data on paper from its usage. In other words, fields on a page must be accessible without the page, and the person who captured the value should not be required to know all its intended usages. This decoupling of value from page has significant benefits:

Access control: Shredding allows different users to see different shred image subsets, for example: (1) research staff can double-check single values, or (2) a subject matter expert can look at just at-risk or "user review"-flagged values.

Because workers only see a single field at a time without knowing what type of document it comes from, we have the potential to safely digitize sensitive data, including personally identifiable information, by presenting fields to on-line workers in a way that does not allow an individual worker to decipher the original content by combining shreds over time. Entry of some values may be restricted to authenticated, trusted workers. Shredding also allows automatic redaction of values from images before they are shown to workers.

Multiple worker pools: Shredding allows more granular control over where data entry work is actually performed. For example, regulations may require sensitive data to be digitized in-country. Similarly, values flagged as difficult to enter by front-line data entry staff can be forwarded to more specialized workers, perhaps working within the organization. In another example, medically related shreds can be formulated as re-CAPTCHA[4] tasks on websites serving the medical community. In other situations, we can rely on general "cognitive surplus" in the developed world, who can now assist development efforts while earning social kudos or other incentives [24].

Lazy evaluation: Shreddr links each answer to the original shredded image in a spreadsheet interface, allowing users to browse and spot-check results, even before entry. As all the source shredded

[4]http://en.wikipedia.org/wiki/ReCAPTCHA

images are retained and available, it may not be worthwhile to immediately (or ever) transcribe open-ended (and difficult and expensive to enter) responses like comments and other open response text fields. These fields can be captured just-in-case, and digitized or viewed on-demand. Similarly, an organization can choose to start with cheap and immediately available low-quality data (such as that available directly from OCR) for real-time monitoring and supervision, while postponing higher quality data, with more estimates and verifications, until it is required for yearly monitoring and evaluation. Using Shreddr, organizations can perform or refine data entry for a specific set of fields whenever they choose.

Datatype	#Shreds	%	Time
Text	186324	18.7%	18.8s
Integer	180144	18.1%	10.1s
Select-one	551542	55.4%	11.0s
Select-many	77284	7.8%	12.7s
Total	995294	100.0 %	12.5s

Table 1: This table shows the number of shreds (and percentage of total), and mean amount of worker time spend per value for the case study dataset.

6. CASE STUDY

Shreddr digitized the paper forms obtained from a three-month paper survey of citizens across 570 villages and covering 95 rural towns in Mali in August 2011. The survey had a total of 6-7 pages (there were two versions), and focused on perceptions of governance and accountability among Malian elected officials. In total, we collected approximately 5600 completed surveys.

Originally the project had planned to hire two or three data entry clerks, rent office space, purchase computers and provide in-country training and supervision in order to manually key in all 36,819 total pages that were collected. Based on their estimate of a typical date entry clerk entering three surveys per hour, it would have required two data clerks seven to eight months to finish the job at single-entry quality, assuming five working days per week, and eight working hours per day, covering more then 2,600 hours of total data entry work.

Instead, a staff member collected the paper forms at a centrally located office, where they were imaged and entered using Shreddr. We describe this process in more detail below.

Imaging and uploading: A data manager trained the local staff member to photograph the pages with a point-and-shoot digital camera (Canon Powershot A620) and tripod (as seen in Figure 2). The data manager explained: "I was able to explain the process in 10 minutes to [a data clerk] and leave him unsupervised." Striking an appropriate balance between legibility and upload size, each image was saved in JPG format at 1200 by 1600 resolution and averaged 350KB each. Images were transferred via SD card to a netbook computer running the Picasa image management desktop software[5]. Using Picasa, images were batch-renamed according to a previously agreed-upon convention, and exported to a shared DropBox folder. DropBox[6] is a commercial software service for synchronizing files over the Internet. It provided us delay-tolerant file syncing over a 3G mobile data connection. Each netbook's battery charge lasted about 6 hours. If there was no power, back up batteries were needed or the upload process was stalled.

The data manager estimated that they processed roughly 150 pages (or 22 surveys) per hour, and split their time as follows: 15% organizing and preparing paper forms for imaging, 60% photographing, and 25% organizing and uploading the digital files. At this rate of image creation and upload (150 pages per hour * 350KB per page),a 15KB/sec upload link was needed to keep up with arrival of imaged documents. This rate is usually achievable with a decent GPRS cellular connection. Otherwise, scanned images could be batch-uploaded at a later time, for example overnight. The data manager reported that his biggest pain was removing staples from surveys and having to manually transfer and rename batches of images. Since then, we have improved our image alignment algorithm to handle stapled-page image distortion (see Figure 3). In the future, a dedicated application can assist in the process of organizing, imaging, and uploading images directly from a phone.

Document definition: Our document definition tool has been successfully used by several well-educated and computer-savvy survey researchers. In the Mali case, the principal investigator (PI) of the survey project marked up each of 13 template pages after a single in-person tutorial session. Document definition requires sufficient domain expertise to know which data field should be entered, at what quality and what the data-type and expected answers for each field should be.

In the future, to minimize the chances of erroneous document definition, we can use use automated data-type detection and box detection to simplify and/or validate elements of the document definition process.

Digitization cost and latency: This dataset consisted of a total of about 1 million shredded images that needed to be converted to values. The total number and percentage of shreds per data type, and the mean worker time required to enter them with high quality (including between 3 to 12 workers), can be found in Figure 1.

Our MTurk workers cumulatively spent 2923 hours entering data. Each shred was seen by at least 3 MTurk workers. For comparison, recall that project staff earlier estimated 2600 man-hours for on-site entry at single-pass quality. If we doubled the estimate for double entry quality, and add the costs of facilities, training and equipment for manual workers, we can estimate that Shreddr lowered the total cost of entry by at least half. Note that with automatic value estimation methods like OCR, mark-classification and blank-detection, the total cost could have been even lower.

Cost comparison in terms of actual wages offered is a more crude comparison, as wages vary widely across different geographies, and across different crowd-sourcing platforms and users. The following is a snapshot at the time of writing: project staff said they would pay Malian data entry workers $2 USD/hour. For this study, on a we attempted to pay our mechanical turk workers the same amount on a per task basis (we actually ended up paying less than $2USD/hour per hour to the turk workers, because they completed more tasks then we expected in the allotted time).

The total duration of data entry, with starts and stops, was just over 4 days—compared to the 32 weeks estimated for manual entry by two to three in-house data entry clerks. We were limited by several service interruptions, and by a limitation on the number of concurrent assignments we could have listed at any one time on MTurk (500). At the observed peak rate of about 3000 assignments per hour, we could have completed the dataset within 24 hours (71169 total assignments were actually submitted).

Worker characteristics: 3672 unique workers saw our tasks on MTurk over 10,443 visits; each performing on average 28 minutes of work. 98% of those workers used a downloaded browser (Chrome and Firefox), indicating some technical sophistication. Most workers hailed from India. Among Indian workers, almost half had IP addresses registered in Chennai or Bangalore (Table 2),

[5]http://picasa.google.com
[6]http://dropbox.com

Country/Territory	Visits	Pages/Visit
India	7565	16.7
United States	1939	9.4
United Kingdom	108	32.4
Chennai	1799	21.6
Bangalore	1385	14.9
Total	10443	

Table 2: Worker demographics

both being significant outsourcing hubs.

7. DATA QUALITY

7.1 Sources of error

We first introduce some common techniques for managing data quality and discuss how Shreddr can help.

Incorrect or missing measurements during initial capture: Common methods for improving the robustness of data collection instruments and protocols include: (1) Planting respondents who give known answers among the population; (2) Repeating measurements over the same populations; (3) Adding questions to the data collection instrument to cross-validate important answers; (4) Finding outliers in the answers, and prioritizing the most unlikely answers for review. These methods all establish invariants, which if violated, indicate quality degradation. In order to detect invariants, data should be digitized incrementally and with reasonable turnaround time – both properties supported by the Shreddr approach.

Corruption to digital files or physical paper during transit or storage: Organization often lack proper backup and security measures. The longer paper stacks sit idle without being curated, the more likely it is lost, shuffled, and lose usefulness. Shreddr images documents and stores them in the "cloud" soon after collection, safeguarding against future data corruption and loss.

Mis-keying during transcription: Typically, organizations prevent mis-keying with the following approaches: gold standard values, repeated entries, and entry interface guidance [2]. Keying errors are the primary focus of the discussion below.

7.2 DDE quality

To illustrate the transcription quality achieved with Shreddr, we first examine the process of double data entry (DDE).

Recall that full-DDE (Section 3) is when all values are entered twice, independently. A simple model of full-DDE's expected error rate can be stated as follows: errors in the data that were flagged and incorrectly reconciled, and errors that were not flagged; specifically:

$$\begin{aligned} p(dde) &= p(de\_conflict) * p(judge\_error) \\ &\quad + p(mutual\_mistakes) \end{aligned} \tag{1}$$

The first term models the likelihood that two independent measurements disagreed, and a third attempt to arbitrate was also mistaken. The second term models the likelihood that the two independent measurements made the same mistake on the same answer.

However, errors can quite easily be systematic in nature; for example, both DDE workers misunderstand the entry protocol and skip an entry that should be have the value as "not applicable". Often, even double entry cannot recover these errors.

Often, due to resource and time limitations, partial-DDE is used to check only a portion of the values as a sampling-based quality-control measure. Partial-DDE's expected error rate is:

$$\begin{aligned} p(partial\_dde) &= p(single\_pass\_error) \\ &\quad - portion\_double\_entered \\ &\quad * (p(dde) - p(single\_pass\_error)) \end{aligned} \tag{2}$$

The approach ensures that the single-entry error rate is at a reasonable level, but does not attempt to catch and fix all errors.

7.3 Shreddr quality control

In contrast, here we detail Shreddr's approach.

Gold standards: Shreddr programmatically creates *gold standard* values, from a randomly selected subset of shreds, to act as the first line in quality control. This has been shown to be an effective technique for managing quality in crowd-sourced systems [10, 15]. It is critical that gold standards are correct: gold shreds are randomly assigned as entry tasks to 3-5 different MTurk workers. A gold answer is accepted if all MTurk responses match.

For the remaining shreds, entry batches are injected with a small number of correct gold standards; verify batches are injected with correct and incorrect gold standards. If an answer set from a worker fails a majority of gold standard values, it is rejected.

Gold standards have the added benefit that we use them as ground truth data to compare against regularly-transcribed values, and achieve the same effect as partial-DDE (Equation (2)). Importantly, gold shreds that were attempted but failed must be manually re-assessed by an authority in order to have an unbiased, random sample.

Decision plans: Shreddr currently uses heuristic decision plans for catching mistakes. Below, we detail the two decision plans we used - entry + majority vote (EMV), and double entry + spelling (DES).

Entry + majority-vote (EMV): The decision process starts with a single *entry* estimate for each shred. The estimate is verified by a majority vote over two (or if necessary, three) *verify* tasks. If the majority vote fails, the loop is repeated; if the loop is repeated three times, the shred is marked for further review.

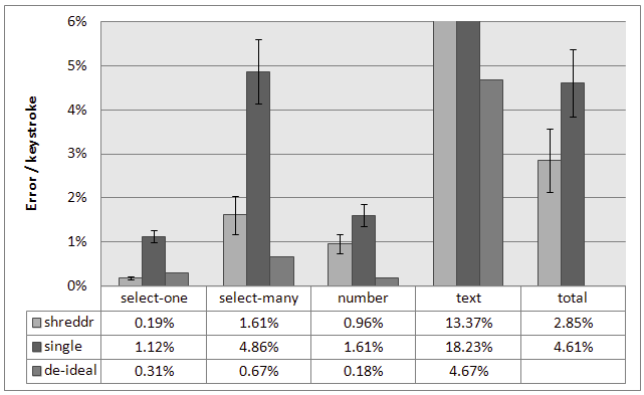
Double entry + spelling-correction (DES): This decision process starts with DDE (Equation (1)): get values from two different workers; if they match after canonicalization, accept the answer. If the answers are close (by edit distance), we ask a third worker to choose the one with better spelling (or submit an alternative correction) and accept that answer. If the answers are not close or the shred is flagged for further review, we obtain another estimate, and so on until we reach convergence.

7.4 Shreddr quality results

We processed all shreds of the Malian case study using the E3V decision plan. In order to measure data quality, we sampled a random subset of 23,255 shreds, and manually curated a ground truth baseline. Ground truth was created using the gold standard creation process (describe above) to generate candidate ground truth values. For each case that the process failed to generate a candidate value (equivalent to gold standard failure), and for each candidate value that did not exactly match the final result, the authors carefully reviewed and manually decided ground truth value.

The industry-standard metric for comparing data quality is errors-per-keystroke. For text or number entry, we used the edit distance compared to the ground truth string. For multiple choice fields, we used the number of items selected incorrectly.

Figure 7 features some example shreds that workers got wrong or were marked "I can't read this": in A and B, the number script is unfamiliar to many workers; in C, French and cursive handwriting was difficult; in D, the marking require some deduction to determine what is the intended answer. Notably, these examples show



	select-one	select-many	number	text	total
shreddr	0.19%	1.61%	0.96%	13.37%	2.85%
single	1.12%	4.86%	1.61%	18.23%	4.61%
de-ideal	0.31%	0.67%	0.18%	4.67%	

Figure 6: Accuracy of Shreddr versus single pass entry and that of idealized double entry.

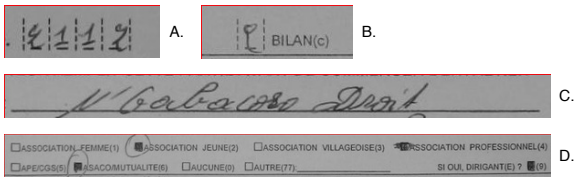


Figure 7: Examples of difficult shreds. In A and B, the number script is unfamiliar to many workers; in C, French and cursive handwriting proved difficult; in D, the markings require some deduction to determine what is the intended answer.

that there exists some fundamentally ambiguous shreds that must be "caught" and escalated to an administrator (and potentially reconciled at the source).

We observed an overall errors-per-keystroke rate of 2.85% (label `shreddr`, docuumented in Figure 6). Shreddr performed well on select-one multiple choice fields: 0.19%, and numbers: 0.96%, as well as select-many multiple choice fields: 1.61%.

The observed text entry error rate of 13.37% requires more explanation. The main difficulty was that most of the responses were in Malian French. (1) MTurk workers' unfamiliarity with the language prevented them from correcting illegible handwriting to actual words and phrases; (2) the handwriting was often in French cursive script which featured unfamiliar serifs and conventions; (3) many MTurk workers did not know how to type accented characters (such as "é").

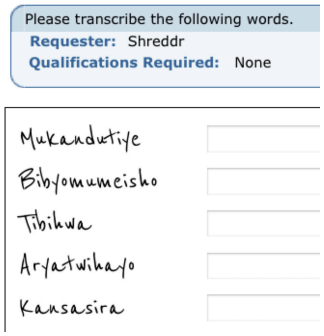


Figure 8: Example interface showing Ugandan last names rendered in a cursive font.

It is entirely possible to transcribe difficult text and achieve high quality results, with both poorly printed text , as well as ambiguous English handwriting on MTurk . We corroborated these results in a separate quality assessment. We tested unfamiliar words using standard characters, by asking MTurk workers to enter Ugandan last names rendered in a handwriting-style font (shown in Figure 8). The experiment offered tasks with 25 names to type for $0.02 USD, and then doubled the number of names per task to 50, 100, and so on, until workers were unwilling to perform the task. We measured the errors-per-keystroke rate of all completed tasks to be 1.28%, in only a single-pass.

To best address this issue, we simply need to find more data entry workers proficient in French typing, and preferably with some familiarity with Malian dialect. In the future, we can require workers have French knowledge through a "qualification". We can also recruit specific worker pools for more specialized tasks, potentially while increasing the cost.

Character-separation can also be an important quality control mechanism. We noted that when the space to write a short response is placed in a character-segmented box, like this:

`Name:  |_|_|_|`

the results were significantly better than those for open-ended response fields.

7.5 Quality comparison

Data collection scenarios differ significantly across different contexts. Recall in Section 3, that data entry quality can range across several orders of magnitude, making it very difficult to make "fair" comparisons. A poorly designed instrument, under-trained enumerators, or less skilled data entry workers can each have a significant impact on the resulting data quality. For our comparison, we try to hold contextual factors constant and compare the outcomes of Shreddr digitization, to that of hypothetical single and double entry scenarios.

In Figure 6, the variables `shreddr` refers to Shreddr's finalized results; `single` refers to the same work that was assign to MTurk workers using the same interfaces and infrastructure as Shreddr, except without Shreddr's additional decision plan quality control mechanisms; `de-ideal` is the hypothetical best case for double entry quality

Vs. single pass: We see that Shreddr outperforms single entry across every data type. For categorical fields, Shreddr is an order of magnitude more accurate; in select-one accuracy, Shreddr is the range of the lowest reported error rates from the clinical trials literature [9, 21]. Number fields saw 40% fewer errors than single pass entry, and for text fields, the advantage was a smaller 27% fewer errors. The relatively smaller improvement obseved for text fields, is indicative of a high baseline error rate, given workers' lack of familiarity with typing in French and/or recognizing words in the Malian dialect.

Vs. ideal double entry: Recall Equation (1):

$$p(dde) = p(de\_conflict) * p(judge\_error) + p(mutual\_mistakes)$$

To calculate `de-ideal`, we make two simplifying assumptions: (1) the data accuracy of reconciliation is equal to that of single entry; and (2) in general, mistakes are independently and uniformly distributed. The latter implies that we can model the likelihood of a worker choosing a particular wrong answer as a random drawing from the *domain* of possible choices, repeated for each decision (a *keystroke*). This means:

	select-1	select-many	number	text
de_conflict	3.39%	8.96%	5.50%	25.60%
keystrokes	1.00	1.57	1.26	2.75
domain	4.13	6.92	*10*	*26*

Table 3: $p(de\_conflict)$ is the empirical % of values that were flagged by two independent single entry passes; average keystrokes empirical averages representing the number of categorical values chosen or the length of characters typed; domain size is the total number of categorical values available or 10 and 26 for number and text fields, respectively.

$$\begin{aligned} p(judge\_error) &= p(single\_pass\_error) \\ p(mutual\_mistakes) &= p(single\_pass\_error) \\ &\quad * \frac{1}{domain\_size^{keystrokes}} \end{aligned}$$

So, our model becomes:

$$\begin{aligned} &p(dde) = \\ &\quad p(de\_conflict) * p(single\_pass\_error) \\ &\quad + p(single\_pass\_error) * \frac{1}{domain\_size^{keystrokes}} \end{aligned} \quad (3)$$

We parameterize the model with a combination of empirical measurements and derivations, summarized Table 3. $p(de\_conflict)$ are the empirical % values that were flagged by two independent single entry passes; *average keystrokes* are the empirical averages representing the number of categorical values chosen or the length of characters typed; *domain size* is the total number of categorical values available or 10 and 26 for number and text fields, respectively. For example, for select-one fields: $3.39\% * 1.01\% + 1.01\% * (1/4.1^1) = 0.31\%$.

We were pleased to see that Shreddr's select-one error rates were below what would be expected from idealized double entry (select-many, number and text types are roughly 2.5, 5.5 and 3 times the ideal rate, respectively). Recall the difficult shreds shown in Figure 7: we must keep in mind that the ideal double entry error rate does not account for the baseline error rate due to input ambiguity; entered values where we simply cannot say what the source intended.

7.6 Error independence

For catching random errors, independent measurements from roughly 3 workers is sufficient. However, errors are often systematic—correlated to other errors. For example, a single worker may tend to make the same type of mistake for same type of question over and over again due to misunderstanding task instructions, or a limitation in their knowledge (for example, of spelling, or of a specific language or dialect). Digitization approaches leveraging fewer workers have a higher likelihood of encountering systematic errors from users who agree on wrong answers. In contrast, Shreddr distributes the work across thousands of workers. As such, for a given question, the probability that a random pair of workers will repeatedly make the same systematic error is very low, as low as the general prevalence of mistaken knowledge within the user population. This is not to say that MTurk workers are necessarily better individually, but with N workers, errors will tend to cancel each other out as N grows.

We can test this hypothesis by measuring the effects of worker distribution in Shreddr's generation of gold standard values, described in Section 7. After gold standard values are created, they are used to police other work. We keep a running rate of worker disagrees vs. agreement percentage, and automatically disable the gold standard if the rate rises above a conservative threshold T. A reject essentially means the value is wrong or ambiguous.

	1 worker	1 or 2 workers	3 workers
P(disabled)	7.9%	0.76%	0.04%
% answered by	1.3%	16%	84%

Table 4: Probabilities of a gold standard value being answered by, and being disabled, by number of unique workers

We examined the distribution of workers who participated in generating our gold standard data (Table 4). Recall that over three thousand different workers provided input to our case study dataset. Still, there was a 1.3% chance that a single worker answered all 3 gold standard estimates (separately), and a 16% chance that a single worker provided at least 2 of 3. These worker-distribution scenarios, per shred, is akin to that of DDE or direct entry.

When a gold standard was created with input from three different workers, its likelihood of being disabled is 0.04%; this increases to 0.76% if fewer than 2 workers provided the input, and increase much higher to 7.9% if only 1 worker provided all 3 estimates. In other words, a value that was agreed on by three different workers is exponentially more likely to be correct than if fewer (but still independent) workers provided the opinion. As a result, we theorize that Shreddr can catch a much greater number of systematic errors with its wider network of users.

8. EFFICIENCY

The idea for Shreddr began with a conversation in a rural Ugandan health clinic. We noted that a data entry clerk was mouthing numbers while inputting "yes/no" values. When asked, he pointed to the sequence of consecutive "yes" answers on the paper form, and explained that by memorizing the number of "yes" answers in the span, he could enter that many without looking back at the paper form. We recognized that he was performing on-the-fly data compression: specifically, he was unknowingly applying the equivalent of run-length encoding[7].

Repeated runs of the same value allow the worker to *compress* the input into one operation, like run-length-encoding enables vector operations over stored data. Data compression algorithms like run-length-encoding reduce the consumption of storage and and network bandwidth. In the same way, we can engineer the data entry task to reduce the consumption of human cognitive bandwidth.

8.1 Shredded interfaces

Shredding filled-in documents create the opportunity to create worker-optimized interfaces. The freedom to work with images representing individual values is key. Both physical and cognitive operations can benefit. We use information entropy, the metric for data compress-ability, combined with Fitts' Law [6], the metric for physical efficiency, to measure the effectiveness of a compressed data entry interface.

Order by field: Figure 5, as mentioned before, is an example of our entry interface. In it, we show only values from a single field—in this case, a 2-digit number. This interface is advantageous in terms of the required physical movements: the value image is co-located with the entry field, rather than, say, on a desk; as well as entropy: the worker only has to think about numbers, rather than

[7] http://en.wikipedia.org/wiki/Run-length_encoding

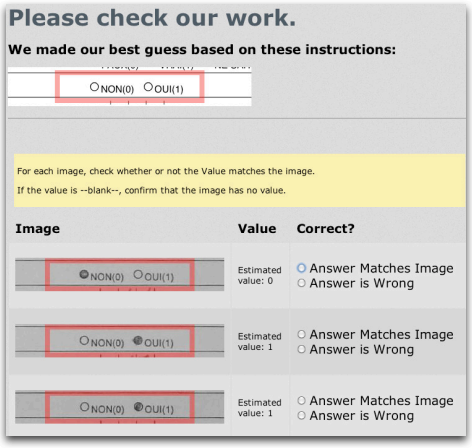


Figure 9: Verification interface - list-style.

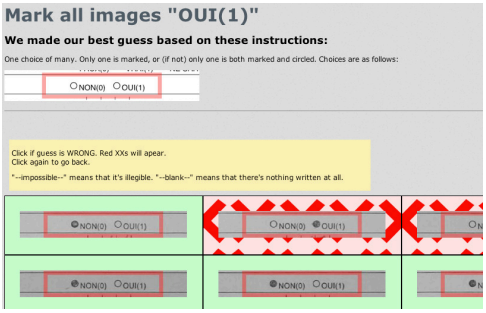


Figure 10: Value-ordered verification interface - grid-style.

switch mental context between different value domains. This also allows them to center their hands on the numeric keypad, or on the numeric portion of a standard keyboard.

Reformulation and order by value: Figures 9 and 10 show examples of our verification interfaces: the first lists our best estimate alongside the image and prompts the worker to confirm whether an answer is correct, the second pre-sorts our best estimates by value, and lays out only those that we believe to be a particular value, and prompts the worker to click those that do not match. The list interface reduces the active domain size to a binary decision. The value-ordered grid interface reduces the task to one of pattern matching.

Efficiency analysis: Figure 11 shows durations in seconds of how long a MTurk worker spent per value, by data type and question effort. We present question effort as the number of choices to select from for checkbox and radiobuttons, and the number of keystrokes for handwritten text or numbers.

We can see that for the same number of characters, integers take less time than text, providing more evidence that a smaller domain size (10 digits) is faster then a larger one (all letters). The durations of radio buttons are consistently below that of checkboxes, because checkboxes are can have multiple answers and thus a longer amount of time. By the same argument, we can also gain speed by turning a data entry task into a verification, which we have shown can be faster and more accurate [4].

9. DISCUSSION

In this section we discuss some other benefits, trade-offs and future possibilities building on the Shreddr approach.

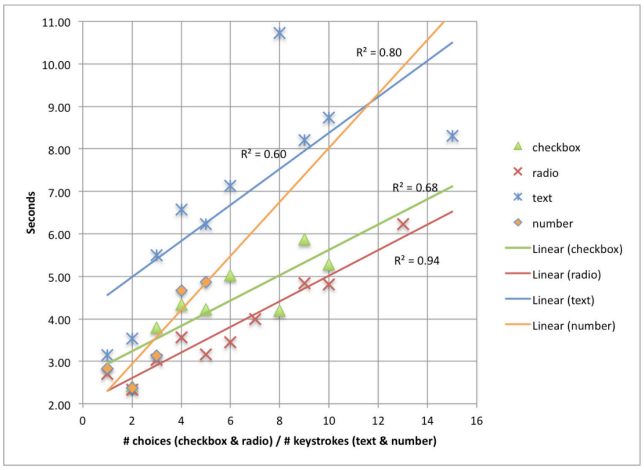


Figure 11: Durations in seconds of how long a MTurk worker spent per value, by data type and question difficulty

Shredded context: Shredded entry and traditional entry provide different types of contextual benefit. Shredded entry allows *domain* context: a worker can see many examples of field's values at once. For example, when entering many handwritten village name that should be the same, but is written by different hands, a worker can pattern-match, decide on an optimal spelling, and quickly enter the same value for all matches. Traditional ordered entry provides *correlational* context, which enables additional understanding of the data by presenting it in the context of the rest of the form, including other fields pertaining to the same record. For example, while entering "age", if the worker sees the "weight", and thinks, "a five-year-old cannot weigh 100 kilograms", then the worker has caught a semantic incongruity. That many data entry workers say they do not think about the meaning behind what they are transcribing [2], suggests that domain context is much more useful than the correlational context for maintaining data entry efficiency and quality. We can also automatically find semantic incongruities using statistical techniques [4].

Maximizing bits-per-decision In data systems research, streamlining dataflows is about finding bottlenecks—mismatches in impedance. The orders of magnitude time-difference between the rates of human and computer decision-making indicate that we should optimize around the amount of latency human computation introduces into a computing system. As data digitization becomes more and more automated, we must ask, "What diminishing roles do humans have in a mostly digital system"? Our belief is that we should treat human computation as input for significant events, and use it during computational fix-points or to break ties, or to answer very hard problems. It follows that human-computer hybrid data flows must reformulate the human task to maximize the effective *bits-per-human-decision*.

10. CONCLUSION

Shreddr allows field workers to capture information using existing and familiar paper forms, which are upload as images and iteratively digitized, using a combination of automated and human-assisted techniques. Shreddr's approach segments the work-flow to leverage pipeline and task parallelism. The key mechanisms are to separate data capture from encoding, to shred images to create opportunities for vector operations in the interface, and to leverage on-line workers to achieve low latency and efficient task allocation

at scale.

We presented a case study of the system in action, in which we digitized a million values from a large-scale citizen survey conducted in Mali. Some types of values indicated that we still have work to do, particularly for dealing with specialized languages and in escalating difficult values to the appropriate expert workers. Other data types (select-one) already showed great results that achieve comparable quality to, and more efficient then, the standard practices used for clinical trials and other rigorous evaluations.

In the future, we plan to apply more computer vision and machine learning optimizations, allowing our workers to focus on more specialized, more difficult (and hence higher paying) work. We also plan to expand these interfaces to better support and incentivize in-house, in-country and volunteer workers.

11. ACKNOWLEDGMENTS

A big thank you to Melissa Ho and Jessica Gottleib for their patience, feedback, and trust. We are grateful to Andy Kanter, Eric Brewer, Kurtis Heimerl, Maneesh Agrawala, Matt Berg for advice and support. Of course, this could not have been possible without the team at Captricity: Andrea Spillmann, Jake Abernathy, Jeff Lin, Jon Barron, Lee Butterman, Nick Jalbert, Rob Carroll, Trevor Smith – thanks for sharing this vision.

12. REFERENCES

[1] OpenRosa. `http://openrosa.org/`.

[2] K. Chen, J.M. Hellerstein, and T.S. Parikh. Designing adaptive feedback for improving data entry accuracy. In *Proc. of UIST*, pages 239–248. ACM, 2010.

[3] K. Chen, J.M. Hellerstein, and T.S. Parikh. Data in the first mile. In *Proc. of CIDR 11*, 2011.

[4] Kuang Chen, Harr Chen, Neil Conway, Joseph M. Hellerstein, and Tapan S. Parikh. Usher: Improving data quality with dynamic forms. *IEEE Transactions on Knowledge and Data Engineering*, 23:1138–1153, 2011.

[5] Brian DeRenzi, Neal Lesh, Tapan Parikh, Clayton Sims, Werner Maokla, Mwajuma Chemba, Yuna Hamisi, David S hellenberg, Marc Mitchell, and Gaetano Borriello. E-imci: improving pediatric health care in low-income countries. In *Proc. of SIGCHI*, 2008.

[6] P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *J. of Exp. Psychology*, 47(6), 1954.

[7] Carl Hartung, Yaw Anokwa, Waylon Brunette, Adam Lerer, Clint Tseng, and Gaetano Borriello. Open data kit: Building information services for developing regions. In *Proc. of ICTD*, 2010.

[8] W. Johnson, H. Jellinek, L. Klotz Jr, R. Rao, and S.K. Card. Bridging the paper and electronic worlds: the paper user interface. In *Proc. of SIGCHI*, pages 507–512. ACM, 1993.

[9] C.K. Jørgensen and B. Karlsmose. Validation of automated forms processing: A comparison of teleform with manual data entry. *Computers in biology and medicine*, 28(6):659–667, 1998.

[10] A. Kittur, E.H. Chi, and B. Suh. Crowdsourcing user studies with mechanical turk. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 453–456. ACM, 2008.

[11] G. Little and Y.A. Sun. Human ocr: Insights from a complex human computation process. 2011.

[12] RA Lorie, VP Riyaz, and TK Truong. A system for automated data entry from forms. In *Proc. of Pattern Recognition*, volume 3, pages 686–690. IEEE, 1996.

[13] J. Mao, R. Lorie, and K. Mohiuddin. A system for automatically reading iata flight coupons. In *icdar*, page 153. Published by the IEEE Computer Society, 1997.

[14] K.S. Mate, B. Bennett, W. Mphatswe, P. Barker, and N. Rollins. Challenges for routine health system data management in a large public programme to prevent mother-to-child hiv transmission in south africa. *PLoS One*, 4(5):e5483, 2009.

[15] D. Oleson, A. Sorokin, G. Laughlin, V. Hester, J. Le, and L. Biewald. Programmatic gold: Targeted and scalable quality assurance in crowdsourcing. 2011.

[16] Tapan S. Parikh. *Designing an architecture for delivering mobile information services to the rural developing world*. PhD thesis, University of Washington, Seattle, WA, USA, 2007.

[17] T.S. Parikh, P. Javid, et al. Mobile phones and paper documents: evaluating a new approach for capturing microfinance data in rural india. In *Proc. of SIGCHI*, pages 551–560. ACM, 2006.

[18] S Patnaik, E Brunskill, and W Thies. Evaluating the accuracy of data collection on mobile phones: A study of forms, sms, and voice. In *ICTD*, 2009.

[19] A.L. Ratan and M. Gogineni. Cost realism in deploying technologies for development. 2008.

[20] AR Ratan, S. Chakraborty, K. Toyama, P. Chitnis, K.S. Ooi, M. Phiong, and M. Koenig. Managing microfinance with paper, pen and digital slate. *Proc. of Info. & Comm. Tech. and Development (ICTD 2010), London, IEEE*, 2010.

[21] Robin A. Reynolds-Haertle and Ruth McBride. Single vs. double data entry in cast. *Controlled Clinical Trials*, 13(6), 1992.

[22] E. Saund. Scientific challenges underlying production document processing. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 7874, page 1, 2011.

[23] Y. Schwartzman and T.S. Parikh. Using cam-equipped mobile phones for procurement and quality control at a rural coffee cooperative. *MobEA V: Mobile Web in the Developing World*, 2007.

[24] Clay Shirky. *Cognitive Surplus: Creativity and Generosity in a Connected Age*. Penguin, 2010.

[25] G. Singh, L. Findlater, K. Toyama, S. Helmer, R. Gandhi, and R. Balakrishnan. Numeric paper forms for ngos. In *Information and Communication Technologies and Development (ICTD), 2009 International Conference on*, pages 406–416. IEEE, 2009.

[26] L. Von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum. recaptcha: Human-based character recognition via web security measures. *Science*, 321(5895):1465, 2008.

[27] S. Zhao and Z. Wang. A high accuracy rate commercial flight coupon recognition system. In *Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on*, pages 82–86. IEEE, 2003.