# Embracing Error to Enable Rapid Crowdsourcing

**Ranjay Krishna**[1], **Kenji Hata**[1], **Stephanie Chen**[1], **Joshua Kravitz**[1],
**David A. Shamma**[2], **Li Fei-Fei**[1], **Michael S. Bernstein**[1]
Stanford University[1], Yahoo! Labs[2]
{ranjaykrishna, kenjihata, stephchen, kravitzj, feifeili, msb}@cs.stanford.edu, aymans@acm.org

## ABSTRACT
Microtask crowdsourcing has enabled dataset advances in social science and machine learning, but existing crowdsourcing schemes are too expensive to scale up with the expanding volume of data. To scale and widen the applicability of crowdsourcing, we present a technique that produces extremely rapid judgments for binary and categorical labels. Rather than punishing all errors, which causes workers to proceed slowly and deliberately, our technique speeds up workers' judgments to the point where errors are acceptable and even expected. We demonstrate that it is possible to rectify these errors by randomizing task order and modeling response latency. We evaluate our technique on a breadth of common labeling tasks such as image verification, word similarity, sentiment analysis and topic classification. Where prior work typically achieves a 0.25× to 1× speedup over fixed majority vote, our approach often achieves an order of magnitude (10×) speedup.

### Author Keywords
Human computation; Crowdsourcing; RSVP

### ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

## INTRODUCTION
Social science [28, 41], interactive systems [15, 30] and machine learning [13, 38] are becoming more and more reliant on large-scale, human-annotated data. Increasingly large annotated datasets have unlocked a string of social scientific insights [17, 8] and machine learning performance improvements [29, 18, 65]. One of the main enablers of this growth has been microtask crowdsourcing [59]. Microtask crowdsourcing marketplaces such as Amazon Mechanical Turk offer a scale and cost that makes such annotation feasible. As a result, companies are now using crowd work to complete hundreds of thousands of tasks per day [39].


*CHI'16*, May 07–May 12, 2016, San Jose, CA, USA

ACM 978-1-4503-3362-7/16/05…$15.00
DOI: http://dx.doi.org/10.1145/2858036.2858115

However, even microtask crowdsourcing can be insufficiently scalable, and it remains too expensive for use in the production of many industry-size datasets [24]. Cost is bound to the amount of work completed per minute of effort, and existing techniques for speeding up labeling (reducing the amount of required effort) are not scaling as quickly as the volume of data we are now producing that must be labeled [63]. To expand the applicability of crowdsourcing, the number of items annotated per minute of effort needs to increase substantially.

In this paper, we focus on one of the most common classes of crowdsourcing tasks [20]: binary annotation. These tasks are yes-or-no questions, typically identifying whether or not an input has a specific characteristic. Examples of these types of tasks are topic categorization (e.g., "Is this article about finance?") [52], image classification (e.g., "Is this a dog?") [13, 38, 36], audio styles [53] and emotion detection [36] in songs (e.g., "Is the music calm and soothing?"), word similarity (e.g., "Are *shipment* and *cargo* synonyms?") [42] and sentiment analysis (e.g., "Is this tweet positive?") [43].

Previous methods have sped up binary classification tasks by minimizing worker error. A central assumption behind this prior work has been that workers make errors because they are not trying hard enough (e.g., "a lack of expertise, dedication [or] interest" [54]). Platforms thus punish errors harshly, for example by denying payment. Current methods calculate the minimum redundancy necessary to be confident that errors have been removed [54, 57, 58]. These methods typically result in a 0.25× to 1× speedup beyond a fixed majority vote [45, 50, 54, 27].

We take the opposite position: that designing the task to encourage some error, or even make errors inevitable, can produce far greater speedups. Because platforms strongly punish errors, workers carefully examine even straightforward tasks to make sure they do not represent edge cases [40, 22]. The result is slow, deliberate work. We suggest that there are cases where we can encourage workers to move quickly by telling them that making some errors is acceptable. Though individual worker accuracy decreases, we can recover from these mistakes post-hoc algorithmically (Figure 1).

We manifest this idea via a crowdsourcing technique in which workers label a rapidly advancing stream of inputs. Workers are given a binary question to answer, and they observe as the stream automatically advances via a method inspired by rapid serial visual presentation (RSVP) [35, 16]. Workers press a key whenever the answer is "yes" for one of the stream items.

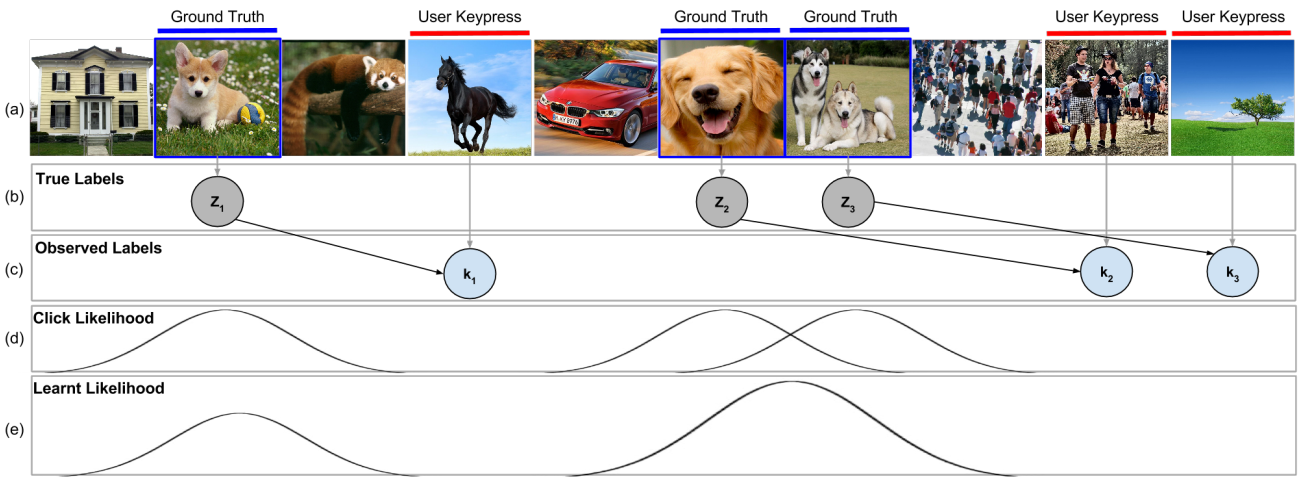


Figure 1: (a) Images are shown to workers at 100ms per image. Workers react whenever they see a dog. (b) The true labels are the ground truth dog images. (c) The workers' keypresses are slow and occur several images after the dog images have already passed. We record these keypresses as the observed labels. (d) Our technique models each keypress as a delayed Gaussian to predict (e) the probability of an image containing a dog from these observed labels.

Because the stream is advancing rapidly, workers miss some items and have delayed responses. However, workers are reassured that the requester expects them to miss a few items. To recover the correct answers, the technique randomizes the item order for each worker and model workers' delays as a normal distribution whose variance depends on the stream's speed. For example, when labeling whether images have a "barking dog" in them, a self-paced worker on this task takes 1.7s per image on average. With our technique, workers are shown a stream at 100ms per image. The technique models the delays experienced at different input speeds and estimates the probability of intended labels from the key presses.

We evaluate our technique by comparing the total worker time necessary to achieve the same precision on an image labeling task as a standard setup with majority vote. The standard approach takes three workers an average of 1.7s each for a total of 5.1s. Our technique achieves identical precision (97%) with five workers at 100ms each, for a total of 500ms of work. The result is an order of magnitude speedup of 10×.

This relative improvement is robust across both simple tasks, such as identifying dogs, and complicated tasks, such as identifying "a person riding a motorcycle" (interactions between two objects) or "people eating breakfast" (understanding relationships among many objects). We generalize our technique to other tasks such as word similarity detection, topic classification and sentiment analysis. Additionally, we extend our method to categorical classification tasks through a ranked cascade of binary classifications. Finally, we test workers' subjective mental workload and find no measurable increase.

**Contributions**. We make the following contributions:

1. We introduce a rapid crowdsourcing technique that makes errors normal and even inevitable. We show that it can be used to effectively label large datasets by achieving a speedup of an order of magnitude on several binary labeling crowdsourcing tasks.
2. We demonstrate that our technique can be generalized to multi-label categorical labeling tasks, combined independently with existing optimization techniques, and deployed without increasing worker mental workload.

## RELATED WORK

The main motivation behind our work is to provide an environment where humans can make decisions quickly. We encourage a margin of human error in the interface that is then rectified by inferring the true labels algorithmically. In this section, we review prior work on crowdsourcing optimization and other methods for motivating contributions. Much of this work relies on artificial intelligence techniques: we complement this literature by changing the crowdsourcing interface rather than focusing on the underlying statistical model.

Our technique is inspired by rapid serial visual presentation (RSVP), a technique for consuming media rapidly by aligning it within the foveal region and advancing between items quickly [35, 16]. RSVP has already been proven to be effective at speeding up reading rates [72]. RSVP users can react to a single target image in a sequence of images even at 125ms per image with 75% accuracy [46]. However, when trying to recognize concepts in images, RSVP only achieves an accuracy of 10% at the same speed [47]. In our work, we integrate multiple workers' errors to successfully extract true labels.

Many previous papers have explored ways of modeling workers to remove bias or errors from ground truth labels [71, 70, 73, 45, 21]. For example, an unsupervised method for judging worker quality can be used as a prior to remove bias on binary verification labels [21]. Individual workers can also be modeled as projections into an open space representing their skills in labeling a particular image [71]. Workers may have unknown expertise that may in some cases prove adversarial to the task. Such adversarial workers can be detected by jointly learning the difficulty of labeling a particular datum

along with the expertises of workers [70]. Finally, a generative model can be used to model workers' skills by minimizing the entropy of the distribution over their labels and the unknown true labels [73]. We draw inspiration from this literature, calibrating our model using a similar generative approach to understand worker reaction times. We model each worker's reaction as a delayed Gaussian distribution.

In an effort to reduce cost, many previous papers have studied the tradeoffs between speed (cost) and accuracy on a wide range of tasks [68, 6, 67, 49]. Some methods estimate human time with annotation accuracy to jointly model the errors in the annotation process [68, 6, 67]. Other methods vary both the labeling cost and annotation accuracy to calculate a tradeoff between the two [23, 14]. Similarly, some crowdsourcing systems optimize a budget to measure confidence in worker annotations [26, 27]. Models can also predict the redundancy of non-expert labels needed to match expert-level annotations [54]. Just like these methods, we show that non-experts can use our technique and provide expert-quality annotations; we also compare our methods to the conventional majority-voting annotation scheme.

Another perspective on rapid crowdsourcing is to return results in real time, often by using a retainer model to recall workers quickly [1, 33, 31]. Like our approach, real-time crowdsourcing can use algorithmic solutions to combine multiple in-progress contributions [32]. These systems' techniques could be fused with ours to create crowds that can react to bursty requests.

One common method for optimizing crowdsourcing is active learning, which involves learning algorithms that interactively query the user. Examples include training image recognition [60] and attribution recognition [44] with fewer examples. Comparative models for ranking attribute models have also optimized crowdsourcing using active learning [37]. Similar techniques have explored optimization of the "crowd kernel" by adaptively choosing the next questions asked of the crowd in order to build a similarity matrix between a given set of data points [62]. Active learning needs to decide on a new task after each new piece of data is gathered from the crowd. Such models tend to be quite expensive to compute. Other methods have been proposed to decide on a set of tasks instead of just one task [64]. We draw on this literature: in our technique, after all the images have been seen by at least one worker, we use active learning to decide the next set of tasks. We determine which images to discard and which images to group together and send this set to another worker to gather more information.

Finally, there is a group of techniques that attempt to optimize label collection by reducing the number of questions that must be answered by the crowd. For example, a hierarchy in label distribution can reduce the annotation search space [14], and information gain can reduce the number of labels necessary to build large taxonomies using a crowd [11, 4]. Methods have also been proposed to maximize accuracy of object localization in images [61] and videos [66]. Previous labels can also be used as a prior to optimize acquisition of new types of annotations [5]. One of the benefits of our technique is that it can be used independently of these others to jointly improve crowdsourcing schemes. We demonstrate the gains of such a combination in our evaluation.

## ERROR-EMBRACING CROWDSOURCING

Current microtask crowdsourcing platforms like Amazon Mechanical Turk incentivize workers to avoid rejections [22, 40], resulting in slow and meticulous work. But is such careful work necessary to build an accurate dataset? In this section, we detail our technique for rapid crowdsourcing by encouraging less accurate work.

The design space of such techniques must consider which tradeoffs are acceptable to make. The first relevant dimension is accuracy. When labeling a large dataset (e.g., building a dataset of ten thousand articles about housing), *precision* is often the highest priority: articles labeled as on-topic by the system must in fact be about housing. *Recall*, on the other hand, is often less important, because there is typically a large amount of available unlabeled data: even if the system misses some on-topic articles, the system can label more items until it reaches the desired dataset size. We thus develop an approach for producing high precision at high speed, sacrificing some recall if necessary.

The second design dimension involves the task characteristics. Many large-scale crowdsourcing tasks involve closed-ended responses such as binary or categorical classifications. These tasks have two useful properties. First, they are time-bound by users' perception and cognition speed rather than motor (e.g., pointing, typing) speed [10], since acting requires only a single button press. Second, it is possible to aggregate responses automatically, for example with majority vote. Open-ended crowdsourcing tasks such as writing [2] or transcription are often time-bound by data entry motor speeds and cannot be automatically aggregated. Thus, with our technique, we focus on closed-ended tasks.

### Rapid crowdsourcing of binary decision tasks

Binary questions are one of the most common classes of crowdsourcing tasks. Each yes-or-no question gathers a label on whether each item has a certain characteristic. In our technique, rather than letting workers focus on each item too carefully, we display each item for a specific period of time before moving on to the next one in a rapid slideshow. For example, in the context of an image verification task, we show workers a stream of images and ask them to press the spacebar whenever they see a specific class of image. In the example in Figure 2, we ask them to react whenever they see a "dog."

The main parameter in this approach is the length of time each item is visible. To determine the best option, we begin by allowing workers to work at their own pace. This establishes an initial average time period, which we then slowly decrease in successive versions until workers start making mistakes [10]. Once we have identified this error point, we can algorithmically model workers' latency and errors to extract the true labels.

To avoid stressing out workers, it is important that the task instructions convey the nature of the rapid task and the fact

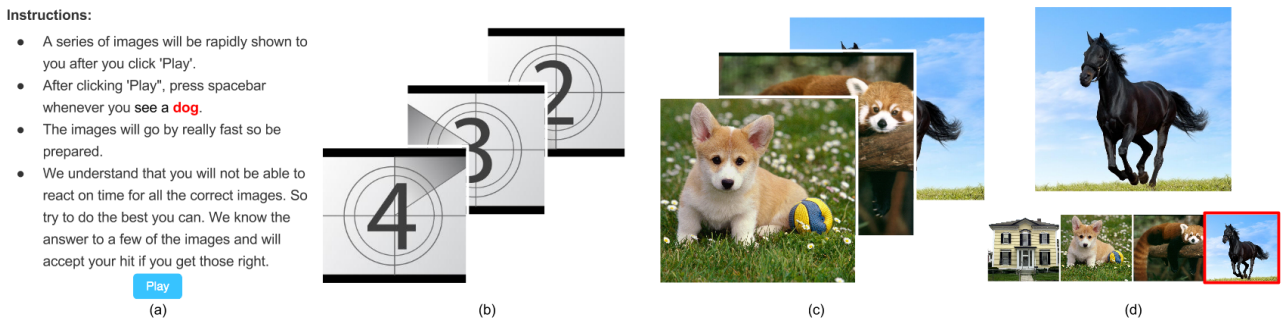


Figure 2: (a) Task instructions inform workers that we expect them to make mistakes since the items will be displayed rapidly. (b) A string of countdown images prepares them for the rate at which items will be displayed. (c) An example image of a "dog" shown in the stream—the two images appearing behind it are included for clarity but are not displayed to workers. (d) When the worker presses a key, we show the last four images below the stream of images to indicate which images might have just been labeled.

that we expect them to make some errors. Workers are first shown a set of instructions (Figure 2(a)) for the task. They are warned that reacting to every single correct image on time is not feasible and thus not expected. We also warn them that we have placed a small number of items in the set that we know to be positive items. These help us calibrate each worker's speed and also provide us with a mechanism to reject workers who do not react to any of the items.

Once workers start the stream (Figure 2(b)), it is important to prepare them for pace of the task. We thus show a film-style countdown for the first few seconds that decrements to zero at the same interval as the main task. Without these countdown images, workers use up the first few seconds getting used to the pace and speed. Figure 2(c) shows an example "dog" image that is displayed in front of the user. The dimensions of all items (images) shown are held constant to avoid having to adjust to larger or smaller visual ranges.

When items are displayed for less than 400ms, workers tend to react to all positive items with a delay. If the interface only reacts with a simple confirmation when workers press the spacebar, many workers worry that they are too late because another item is already on the screen. Our solution is to also briefly display the last four items previously shown when the spacebar is pressed, so that workers see the one they intended and also gather an intuition for how far back the model looks. For example, in Figure 2(d), we show a worker pressing the spacebar on an image of a horse. We anticipate that the worker was probably delayed, and we display the last four items to acknowledge that we have recorded the keypress. We ask all workers to first complete a qualification task in which they receive feedback on how quickly we expect them to react. They pass the qualification task only if they achieve a recall of 0.6 and precision of 0.9 on a stream of 200 items with 25 positives. We measure precision as the fraction of worker reactions that were within 500ms of a positive cue.

In Figure 3, we show two sample outputs from our interface. Workers were shown images for 100ms each. They were asked to press the spacebar whenever they saw an image of "a person riding a motorcycle." The images with blue bars underneath them are ground truth images of "a person riding a motorcycle." The images with red bars show where workers reacted. The important element is that red labels are often delayed behind blue ground truth and occasionally missed entirely. Both Figures 3(a) and 3(b) have 100 images each with 5 correct images.

Because of workers' reaction delay, the data from one worker has considerable uncertainty. We thus show the same set of items to multiple workers in different random orders and collect independent sets of keypresses. This randomization will produce a cleaner signal in aggregate and later allow us to estimate the images to which each worker intended to react.

Given the speed of the images, workers are not able to detect every single positive image. For example, the last positive image in Figure 3(a) and the first positive image in Figure 3(b) are not detected. Previous work on RSVP found a phenomenon called "attention blink" [7], in which a worker is momentarily blind to successive positive images. However, we find that even if two images of "a person riding a motorcycle" occur consecutively, workers are able to detect both and react twice (Figures 3(a) and 3(b)). If workers are forced to react in intervals of less than 400ms, though, the signal we extract is too noisy for our model to estimate the positive items.

### Multi-Class Classification for Categorical Data

So far, we have described how rapid crowdsourcing can be used for binary verification tasks. Now we extend it to handle multi-class classification. Theoretically, all multi-class classification can be broken down into a series of binary verifications. For example, if there are $N$ classes, we can ask $N$ binary questions of whether an item is in each class. Given a list of items, we use our technique to classify them one class at a time. After every iteration, we remove all the positively classified items for a particular class. We use the rest of the items to detect the next class.

Assuming all the classes contain an equal number of items, the order in which we detect classes should not matter. A simple *baseline approach* would choose a class at random

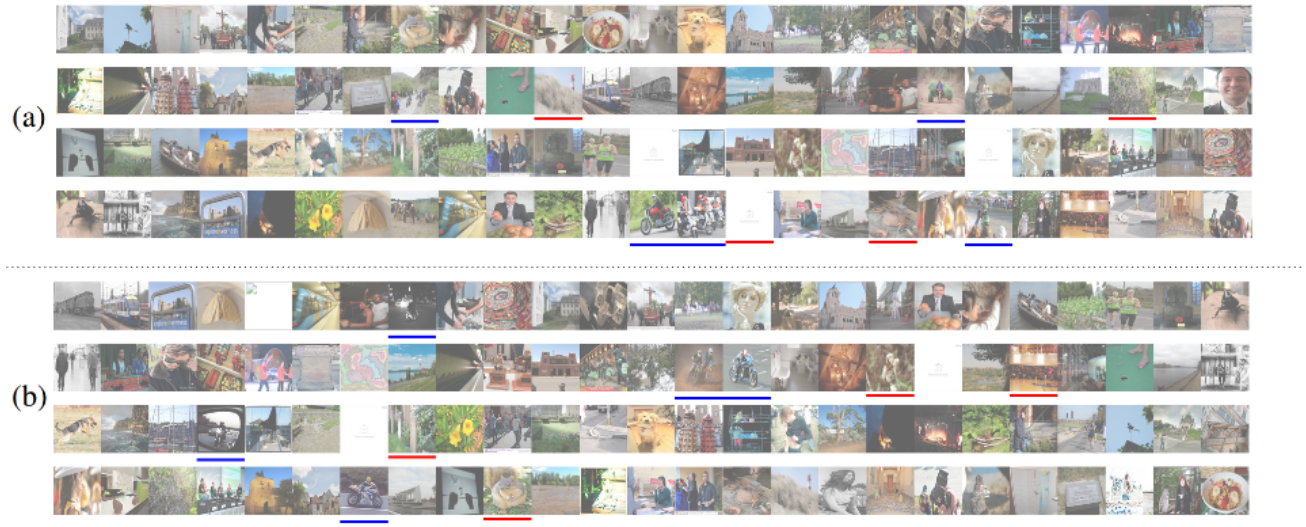


Figure 3: Example raw worker outputs from our interface. Each image was displayed for 100ms and workers were asked to react whenever they saw images of "a person riding a motorcycle." Images are shown in the same order they appeared in for the worker. Positive images are shown with a blue bar below them and users' keypresses are shown as red bars below the image to which they reacted.

and attempt to detect all items for that class first. However, if the distribution of items is not equal among classes, this method would be inefficient. Consider the case where we are trying to classify items into 10 classes, and one class has 1000 items while all other classes have 10 items. In the worst case, if we classify the class with 1000 examples last, those 1000 images would go through our interface 10 times (once for every class). Instead, if we had detected the large class first, we would be able to classify those 1000 images and they would only go through our interface once. With this intuition, we propose a *class-optimized approach* that classifies the most common class of items first. We maximize the number of items we classify at every iteration, reducing the total number of binary verifications required.

## MODEL

To translate workers' delayed and potentially erroneous actions into identifications of the positive items, we need to model their behavior. We do this by calculating the probability that a particular item is in the positive class given that the user reacted a given period after the item was displayed. By combining these probabilities across several workers with different random orders of the same images, these probabilities sum up to identify the correct items.

We use maximum likelihood estimation to predict the probability of an item being a positive example. Given a set of items $\mathcal{I} = \{I_1, \ldots, I_n\}$, we send them to $W$ workers in a different random order for each. From each worker $w$, we collect a set of keypresses $C^w = \{c_1^w, \ldots, c_k^w\}$ where $w \in W$ and $k$ is the total number of keypresses from $w$. Our aim is to calculate the probability of a given item $P(I_i)$ being a positive example. Given that we collect keypresses from $W$ workers:

$$P(I_i) = \sum_w P(I_i|C^w)P(C^w) \tag{1}$$

where $P(C) = \prod_k P(C_k)$ is the probability of a particular set of items being keypresses. We set $P(C_k)$ to be constant, assuming that it is equally likely that a worker might react to any item. Using Bayes' rule:

$$P(I_i|C^w) = \frac{P(C^w|I_i)P(I_i)}{P(C^w)}. \tag{2}$$

$P(I_i)$ models our estimate of item $I_i$ being positive. It can be a constant, or it can be an estimate from a domain-specific machine learning algorithm [25]. For example, to calculate $P(I_i)$, if we were trying to scale up a dataset of "dog" images, we would use a small set of known "dog" images to train a binary classifier and use that to calculate $P(I_i)$ for all the unknown images. With image tasks, we use a pretrained convolutional neural network to extract image features [56] and train a linear support vector machine to calculate $P(I_i)$.

We model $P(C^w|I_i)$ as a set of independent keypresses:

$$P(C^w|I_i) = P(c_1^w, \ldots, c_k^w|I_i) = \prod_k P(C_k^w|I_i). \tag{3}$$

Finally, we model each keypress as a Gaussian distribution $\mathcal{N}(\mu, \sigma)$ given a positive item. We train the mean $\mu$ and variance $\sigma$ by running rapid crowdsourcing on a small set of items for which we already know the positive items. Here, the mean and variance of the distribution are modeled to estimate the delays that a worker makes when reacting to a positive item.

Intuitively, the model works by treating each keypress as creating a Gaussian "footprint" of positive probability on the images about 400ms before the keypress (Figure 1). The model combines these probabilities across several workers to identify the images with the highest overall probability.

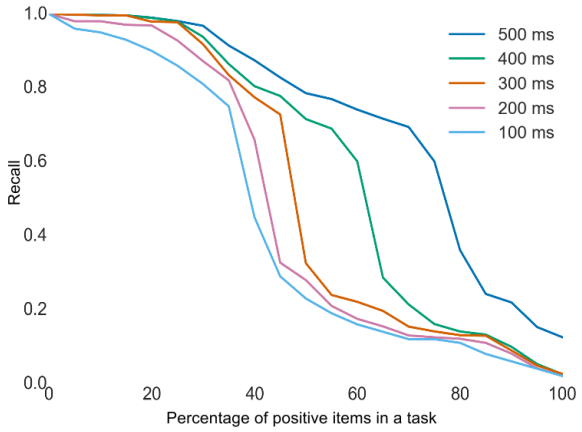


Figure 4: We plot the change in recall as we vary percentage of positive items in a task. We experiment at varying display speeds ranging from 100ms to 500ms. We find that recall is inversely proportional to the rate of positive stimuli and not to the percentage of positive items.

Now that we have a set of probabilities for each item, we need to decide which ones should be classified as positive. We order the set of items $\mathcal{I}$ according to likelihood of being in the positive class $P(I_i)$. We then set all items above a certain threshold as positive. This threshold is a hyperparameter that can be tuned to trade off precision vs. recall.

In total, this model has two hyperparameters: (1) the threshold above which we classify images as positive and (2) the speed at which items are displayed to the user. We model both hyperparameters in a per-task (image verification, sentiment analysis, etc.) basis. For a new task, we first estimate how long it takes to label each item in the conventional setting with a small set of items. Next, we continuously reduce the time each item is displayed until we reach a point where the model is unable to achieve the same precision as the untimed case.

## CALIBRATION: BASELINE WORKER REACTION TIME

Our technique hypothesizes that guiding workers to work quickly and make errors can lead to results that are faster yet with similar precision. We begin evaluating our technique by first studying worker reaction times as we vary the length of time for which each item is displayed. If worker reaction times have a low variance, we accurately model them. Existing work on RSVP estimated that humans usually react about 400ms after being presented with a cue [69, 48]. Similarly, the model human processor [9] estimated that humans perceive, understand and react at least 240ms after a cue. We first measure worker reaction times, then analyze how frequently positive items can be displayed before workers are unable to react to them in time.

*Method.* We recruited 1,000 workers on Amazon Mechanical Turk with 96% approval rating and over 10,000 tasks submitted. Workers were asked to work on one task at a time. Each task contained a stream of 100 images of polka dot patterns of two different colors. Workers were asked to react by pressing the spacebar whenever they saw an image with polka dots of one of the two colors. Tasks could vary by two variables: the *speed* at which images were displayed and the *percentage* of the positively colored images. For a given task, we held the display speed constant. Across multiple tasks, we displayed images for 100ms to 500ms. We studied two variables: *reaction time* and *recall*. We measured the reaction time to the positive color across these speeds. To study recall (percentage of positively colored images detected by workers), we varied the ratio of positive images from 5% to 95%. We counted a keypress as a detection only if it occurred within 500ms of displaying a positively colored image.

*Results.* Workers' reaction times corresponded well with estimates from previous studies. Workers tend to react an average of 378ms ($\sigma$ = 92ms) after seeing a positive image. This consistency is an important result for our model because it assumes that workers have a consistent reaction delay.

As expected, recall is inversely proportional to the speed at which the images are shown. A worker is more likely to miss a positive image at very fast speeds. We also find that recall decreases as we increase the percentage of positive items in the task. To measure the effects of positive frequency on recall, we record the percentage threshold at which recall begins to drop significantly at different speeds and positive frequencies. From Figure 4, at 100ms, we see that recall drops when the percentage of positive images is more than 35%. As we increase the time for which an item is displayed, however, we notice that the drop in recall occurs at a much higher percentage. At 500ms, the recall drops at a threshold of 85%. We thus infer that recall is inversely proportional to the *rate* of positive stimuli and not to the percentage of positive images. From these results we conclude that at faster speeds, it is important to maintain a smaller percentage of positive images, while at slower speeds, the percentage of positive images has a lesser impact on recall. Quantitatively, to maintain a recall higher than 0.7, it is necessary to limit the frequency of positive cues to one every 400ms.

## STUDY 1: IMAGE VERIFICATION

In this study, we deploy our technique on image verification tasks and measure its speed relative to the conventional self-paced approach. Many crowdsourcing tasks in computer vision require verifying that a particular image contains a specific class or concept. We measure precision, recall and cost (in seconds) by the conventional approach and compare against our technique.

Some visual concepts are easier to detect than others. For example, detecting an image of a "dog" is a lot easier than detecting an image of "a person riding a motorcycle" or "eating breakfast." While detecting a "dog" is a perceptual task, "a person riding a motorcycle" requires understanding of the interaction between the person and the motorcycle. Similarly, "eating breakfast" requires workers to fuse concepts of people eating a variety foods like eggs, cereal or pancakes. We test our technique on detecting three concepts: "dog" (easy concept), "a person riding a motorcycle" (medium concept) and "eating breakfast" (hard concept). In this study, we compare how workers fare on each of these three levels of concepts.

| Task | | Conventional Approach | | | Our Technique | | | Speedup |
|---|---|---|---|---|---|---|---|---|
| | | *Time (s)* | *Precision* | *Recall* | *Time (s)* | *Precision* | *Recall* | |
| Image Verification | Easy | 1.50 | 0.99 | 0.99 | 0.10 | 0.99 | 0.94 | **9.00**× |
| | Medium | 1.70 | 0.97 | 0.99 | 0.10 | 0.98 | 0.83 | **10.20**× |
| | Hard | 1.90 | 0.93 | 0.89 | 0.10 | 0.90 | 0.74 | **11.40**× |
| | All Concepts | 1.70 | 0.97 | 0.96 | 0.10 | 0.97 | 0.81 | **10.20**× |
| Sentiment Analysis | | 4.25 | 0.93 | 0.97 | 0.25 | 0.94 | 0.84 | **10.20**× |
| Word Similarity | | 6.23 | 0.89 | 0.94 | 0.60 | 0.88 | 0.86 | **6.23**× |
| Topic Detection | | 14.33 | 0.96 | 0.94 | 2.00 | 0.95 | 0.81 | **10.75**× |

Table 1: We compare the conventional approach for binary verification tasks (image verification, sentiment analysis, word similarity and topic detection) with our technique and compute precision and recall scores. Precision scores, recall scores and speedups are calculated using 3 workers in the conventional setting. Image verification, sentiment analysis and word similarity used 5 workers using our technique, while topic detection used only 2 workers. We also show the time taken (in seconds) for 1 worker to do each task.

*Method.* In this study, we compare the conventional approach with our technique on three (easy, medium and hard) concepts. We evaluate each of these comparisons using precision scores, recall scores and the speedup achieved. To test each of the three concepts, we labeled 10,000 images, where each concept had 500 examples. We divided the 10,000 images into streams of 100 images for each task. We paid workers \$0.17 to label a stream of 100 images (resulting in a wage of \$6 per hour [51]). We hired over 1,000 workers for this study satisfying the same qualifications as the calibration task.

The conventional method of collecting binary labels is to present a crowd worker with a set of items. The worker proceeds to label each item, one at a time. Most datasets employ multiple workers to label each task because majority voting [59] has been shown to improve the quality of crowd annotations. These datasets usually use a redundancy of 3 to 5 workers [55]. In all our experiments, we used a redundancy of 3 workers as our baseline.

When launching tasks using our technique, we tuned the image display speed to 100ms. We used a redundancy of 5 workers when measuring precision and recall scores. To calculate *speedup*, we compare the total worker time taken by all the 5 workers using our technique with the total worker time taken by the 3 workers using the conventional method. Additionally, we vary redundancy on all the concepts to from 1 to 10 workers to see its effects on precision and recall.

*Results.* Self-paced workers take 1.70s on average to label each image with a concept in the conventional approach (Table 1). They are quicker at labeling the easy concept (1.50s per worker) while taking longer on the medium (1.70s) and hard (1.90s) concepts.

Using our technique, even with a redundancy of 5 workers, we achieve a speedup of 10.20× across all concepts. We achieve *order of magnitude* speedups of 9.00×, 10.20× and 11.40× on the easy, medium and hard concepts. Overall, across all concepts, the precision and recall achieved by our technique is 0.97 and 0.81. Meanwhile the precision and recall of the conventional method is 0.97 and 0.96. We thus achieve the same precision as the conventional method. As expected, recall is lower because workers are not able to detect every single true positive example. As argued previously, lower recall can be an acceptable tradeoff when it is easy to find more unlabeled images.

Now, let's compare precision and recall scores between the three concepts. We show precision and recall scores in Figure 5 for the three concepts. Workers perform slightly better at finding "dog" images and find it the most difficult to detect the more challenging "eating breakfast" concept. With a redundancy of 5, the three concepts achieve a precision of 0.99, 0.98 and 0.90 respectively at a recall of 0.94, 0.83 and 0.74 (Table 1). The precision for these three concepts are identical to the conventional approach, while the recall scores are slightly lower. The recall for a more difficult cognitive concept ("eating breakfast") is much lower, at 0.74, than for the other two concepts. More complex concepts usually tend to have a lot of contextual variance. For example, "eating breakfast" might include a person eating a "banana," a "bowl of cereal," "waffles" or "eggs." We find that while some workers react to one variety of the concept (e.g., "bowl of cereal"), others react to another variety (e.g., "eggs").

When we increase the redundancy of workers to 10 (Figure 6), our model is able to better approximate the positive images. We see diminishing increases in both recall and precision as redundancy increases. At a redundancy of 10, we increase recall to the same amount as the conventional approach (0.96), while maintaining a high precision (0.99) and still achieving a speedup of 5.1×.

We conclude from this study that our technique (with a redundancy of 5) can speed up image verification with easy, medium and hard concepts by an order of magnitude while still maintaining high precision. We also show that recall can be compensated by increasing redundancy.

## STUDY 2: NON-VISUAL TASKS

So far, we have shown that rapid crowdsourcing can be used to collect image verification labels. We next test the technique on a variety of other common crowdsourcing tasks: sentiment analysis [43], word similarity [59] and topic detection [34].

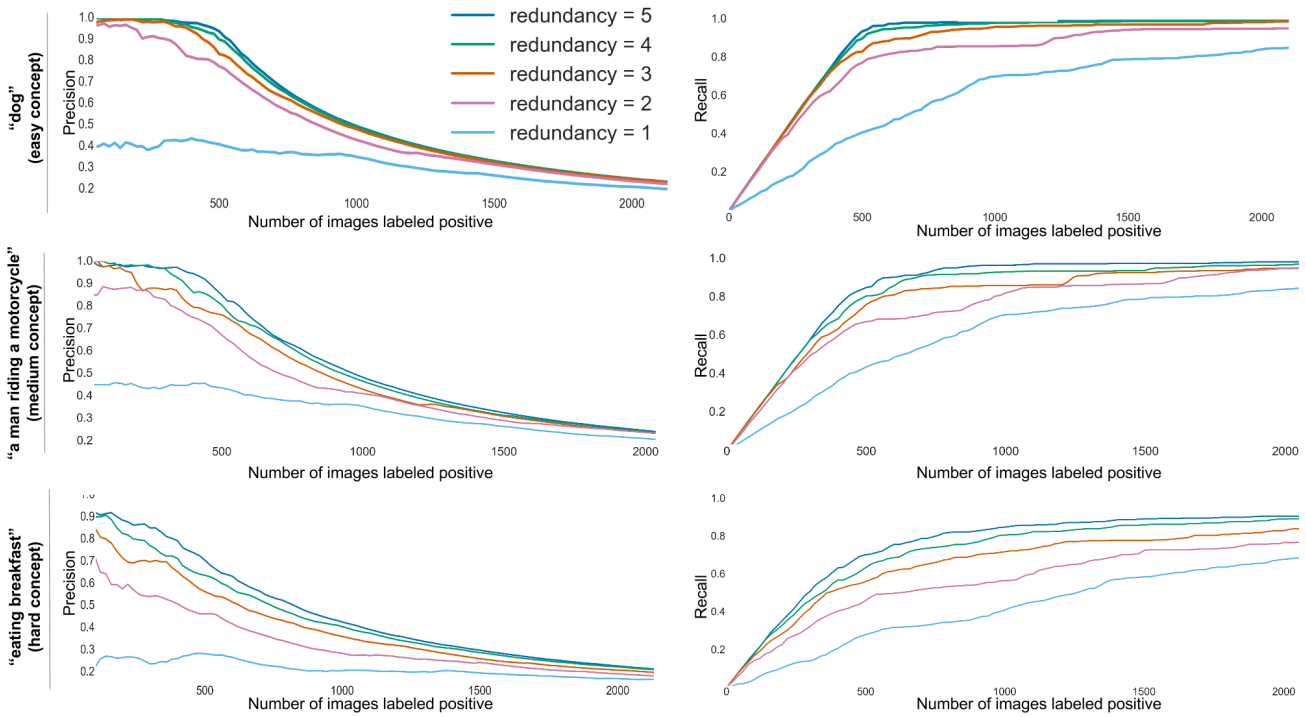


Figure 5: We study the precision (left) and recall (right) curves for detecting "dog" (top), "a person on a motorcycle" (middle) and "eating breakfast" (bottom) images with a redundancy ranging from 1 to 5. There are 500 ground truth positive images in each experiment. We find that our technique works for simple as well as hard concepts.

*Method.* In this study, we measure precision, recall and speedup achieved by our technique over the conventional approach. To determine the stream speed for each task, we followed the prescribed method of running trials and speeding up the stream until the model starts losing precision. For sentiment analysis, workers were shown a stream of tweets and asked to react whenever they saw a positive tweet. We displayed tweets at 250ms with a redundancy of 5 workers. For word similarity, workers were shown a word (e.g., "lad") for which we wanted synonyms. They were then rapidly shown other words at 600ms and asked to react if they see a synonym (e.g., "boy"). Finally, for topic detection, we presented workers with a topic like "housing" or "gas" and presented articles of an average length of 105 words at a speed of 2s per article. They reacted whenever they saw an article containing the topic we were looking for. For all three of these tasks, we compare precision, recall and speed against the self-paced conventional approach with a redundancy of 3 workers. Every task, for both the conventional approach and our technique, contained 100 items.

To measure the cognitive load on workers for labeling so many items at once, we ran the widely-used NASA Task Load Index (TLX) [12] on all tasks, including image verification. TLX measures the perceived workload of a task. We ran the survey on 100 workers who used the conventional approach and 100 workers who used our technique across all tasks.

*Results.* We present our results in Table 1 and Figure 7. For sentiment analysis, we find that workers in the conventional approach classify tweets in 4.25s. So, with a redundancy of 3 workers, the conventional approach would take 12.75s with a precision of 0.93. Using our method and a redundancy of 5 workers, we complete the task in 1250ms (250ms per worker per item) and 0.94 precision. Therefore, our technique achieves a speedup of 10.2×.

Likewise, for word similarity, workers take around 6.23s to complete the conventional task, while our technique succeeds at 600ms. We manage to capture a comparable precision of 0.88 using 5 workers against a precision of 0.89 in the conventional method with 3 workers. Since finding synonyms is a higher-level cognitive task, workers take longer to do word similarity tasks than image verification and sentiment analysis tasks. We manage a speedup of 6.23×.

Finally, for topic detection, workers spend significant time analyzing articles in the conventional setting (14.33s on average). With 3 workers, the conventional approach takes 43s. In comparison, our technique delegates 2s for each article. With a redundancy of only 2 workers, we achieve a precision of 0.95, similar to the 0.96 achieved by the conventional approach. The total worker time to label one article using our technique is 4s, a speedup of 10.75×.

The mean TLX workload for the control condition was 58.5 ($\sigma$ = 9.3), and 62.4 ($\sigma$ = 18.5) for our technique. Unexpectedly, the difference between conditions was not significant ($t(99) = -0.53$, $p = 0.59$). The temporal demand scale item appeared to be elevated for our technique (61.1 vs. 70.0), but this difference was not significant ($t(99) = -0.76$, $p = 0.45$).

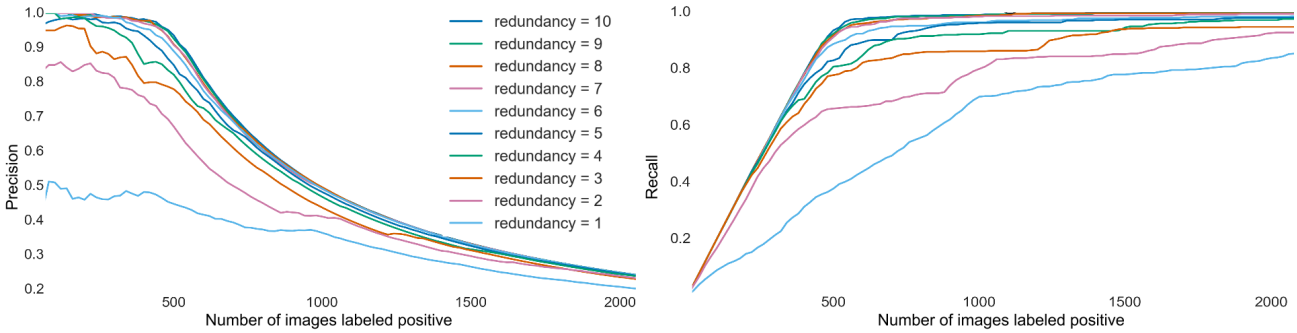


Figure 6: We study the effects of redundancy on recall by plotting precision and recall curves for detecting "a person on a motorcycle" images with a redundancy ranging from 1 to 10. We see diminishing increases in precision and recall as we increase redundancy. We manage to achieve the same precision and recall scores as the conventional approach with a redundancy of 10 while still achieving a speedup of 5×.

We conclude that our technique can be used to scale crowdsourcing on a variety of tasks without statistically increasing worker workload.

## STUDY 3: MULTI-CLASS CLASSIFICATION

In this study, we extend our technique from binary to multi-class classification to capture an even larger set of crowdsourcing tasks. We use our technique to create a dataset where each image is classified into one category ("people," "dog," "horse," "cat," etc.). We compare our technique with a conventional technique [13] that collects binary labels for each image for every single possible class.

*Method.* Our aim is to classify a dataset of 2,000 images with 10 categories where each category contains between 100 to 250 examples. We compared three methods of multi-class classification: (1) a *naive approach* that collected 10 binary labels (one for each class) for each image, (2) a *baseline approach* that used our interface and classified images one class (chosen randomly) at a time, and (3) a *class-optimized approach* that used our interface to classify images starting from the class with the most examples. When using our interface, we broke tasks into streams of 100 images displayed for 100ms each. We used a redundancy of 3 workers for the conventional interface and 5 workers for our interface. We calculated the precision and recall scores across each of these three methods as well as the cost (in seconds) of each method.

*Results.* (1) In the *naive approach*, we need to collect 20,000 binary labels that take 1.7s each. With 5 workers, this takes 102,000s ($170 at a wage rate of $6/hr) with an average precision of 0.99 and recall of 0.95. (2) Using the *baseline approach*, it takes 12,342s ($20.57) with an average precision of 0.98 and recall of 0.83. This shows that the baseline approach achieves a speedup of 8.26× when compared with the naive approach. (3) Finally, the *class-optimized approach* is able to detect the most common class first and hence reduces the number of times an image is sent through our interface. It takes 11,700s ($19.50) with an average precision of 0.98 and recall of 0.83. The class-optimized approach achieves a speedup of 8.7× when compared to the naive approach. While the speedup between the baseline and the class-optimized methods is small, it would be increased on a larger dataset with more classes.

## APPLICATION: BUILDING IMAGENET

Our method can be combined with existing techniques [14, 60, 44, 3] that optimize binary verification and multi-class classification by preprocessing data or using active learning. One such method [14] annotated ImageNet (a popular large dataset for image classification) effectively with a useful insight: they realized that its classes could be grouped together into higher semantic concepts. For example, "dog," "rabbit" and "cat" could be grouped into the concept "animal." By utilizing the hierarchy of labels that is specific to this task, they were able to preprocess and reduce the number of labels needed to classify all images. As a case study, we combine our technique with their insight and evaluate the speedup in collecting a subset of ImageNet.

*Method.* We focused on a subset of the dataset with 20,000 images and classified them into 200 classes. We conducted this case study by comparing three ways of collecting labels: (1) The naive approach asked 200 binary questions for each image in the subset, where each question asked if the image belonged to one of the 200 classes. We used a redundancy of 3 workers for this task. (2) The optimal-labeling method used the insight to reduce the number of labels by utilizing the hierarchy of image classes. (3) The combined approach used our technique for multi-class classification combined with the hierarchy insight to reduce the number of labels collected. We used a redundancy of 5 workers for this technique with tasks of 100 images displayed at 250ms.

*Results.* (1) Using the naive approach, this would result in asking 4 million binary verification questions. Given that each binary label takes 1.7s (Table 1), we estimate that the total time to label the entire dataset would take 6.8 million seconds ($11,333 at a wage rate of $6/hr). (2) The optimal-labeling method is estimated to take 1.13 million seconds ($1,888) [14]. (3) Combining the hierarchical questions with our interface, we annotate the subset in 136,800s ($228). We achieve a precision of 0.97 with a recall of 0.82. By combining our 8× speedup with the 6× speedup from intelligent question selection, we achieve a 50× speedup in total.

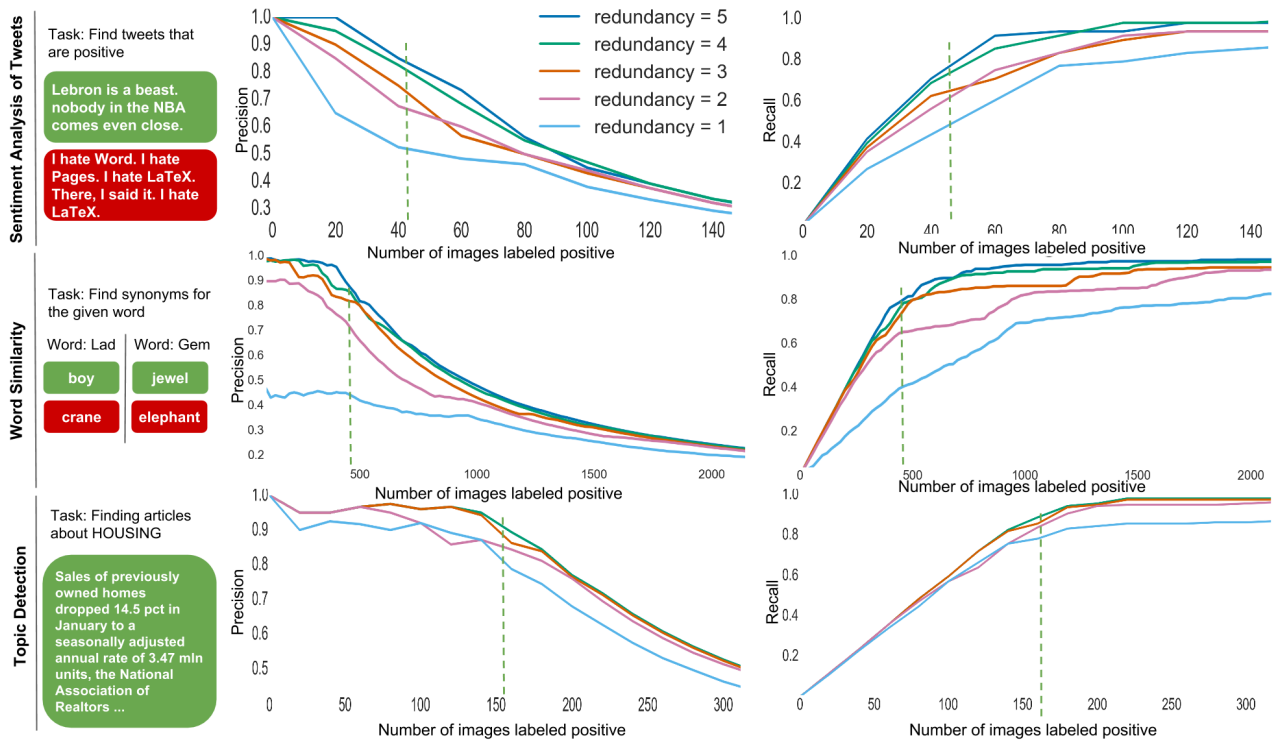


Figure 7: Precision (left) and recall (right) curves for sentiment analysis (top), word similarity (middle) and topic detection (bottom) images with a redundancy ranging from 1 to 5. Vertical lines indicate the number of ground truth positive examples.

## DISCUSSION

We focused our technique on positively identifying concepts. We then also test its effectiveness at classifying the absence of a concept. Instead of asking workers to react when they see a "dog," if we ask them to react when they do *not* see a "dog," our technique performs poorly. At 100ms, we find that workers achieve a recall of only 0.31, which is much lower than a recall of 0.94 when detecting the presence of "dog"s. To improve recall to 0.90, we must slow down the feed to 500ms. Our technique achieves a speedup of 2× with this speed. We conclude that our technique performs poorly for anomaly detection tasks, where the presence of a concept is common but its absence, an anomaly, is rare. More generally, this exercise suggests that some cognitive tasks are less robust to rapid judgments. Preattentive processing can help us find "dog"s, but ensuring that there is no "dog" requires a linear scan of the entire image.

To better understand the active mechanism behind our technique, we turn to concept typicality. A recent study [19] used fMRIs to measure humans' recognition speed for different object categories, finding that images of most typical examplars from a class were recognized faster than the least typical categories. They calculated typicality scores for a set of image classes based on how quickly humans recognized them.In our image verification task, 72% of false negatives were also atypical. Not detecting atypical images might lead to the curation of image datasets that are biased towards more common categories. For example, when curating a dataset of dogs, our technique would be more likely to find usual breeds like "dalmatians" and "labradors" and miss rare breeds like "romagnolos" and "otterhounds." More generally, this approach may amplify biases and minimize clarity on edge cases. Slowing down the feed reduces atypical false negatives, resulting in a smaller speedup but with a higher recall for atypical images.

## CONCLUSION

We have suggested that crowdsourcing can speed up labeling by encouraging a small amount of error rather than forcing workers to avoid it. We introduce a rapid slideshow interface where items are shown too quickly for workers to get all items correct. We algorithmically model worker errors and recover their intended labels. This interface can be used for binary verification tasks like image verification, sentiment analysis, word similarity and topic detection, achieving speedups of 10.2×, 10.2×, 6.23× and 10.75× respectively. It can also extend to multi-class classification and achieve a speedup of 8.26×. Our approach is only one possible interface instantiation of the concept of encouraging some error; we suggest that future work may investigate many others. Speeding up crowdsourcing enables us to build larger datasets to empower scientific insights and industry practice. For many labeling goals, this technique can be used to construct datasets that are an order of magnitude larger without increasing cost.

## ACKNOWLEDGEMENTS

This work was supported by a National Science Foundation award 1351131. Thanks to Joy Kim, Juho Kim, Geza Kovacs, Niloufar Salehi and reviewers for their feedback.

## REFERENCES


1. Michael S Bernstein, Joel Brandt, Robert C Miller, and David R Karger. 2011. Crowds in two seconds: Enabling realtime crowd-powered interfaces. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, 33–42.
2. Michael S Bernstein, Greg Little, Robert C Miller, Björn Hartmann, Mark S Ackerman, David R Karger, David Crowell, and Katrina Panovich. 2010. Soylent: a word processor with a crowd inside. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. ACM, 313–322.
3. Arijit Biswas and Devi Parikh. 2013. Simultaneous active learning of classifiers & attributes via relative feedback. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 644–651.
4. Jonathan Bragg, Mausam Daniel, and Daniel S Weld. 2013. Crowdsourcing multi-label classification for taxonomy creation. In *First AAAI conference on human computation and crowdsourcing*.
5. Steve Branson, Kristjan Eldjarn Hjorleifsson, and Pietro Perona. 2014. Active annotation translation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 3702–3709.
6. Steve Branson, Catherine Wah, Florian Schroff, Boris Babenko, Peter Welinder, Pietro Perona, and Serge Belongie. 2010. Visual recognition with humans in the loop. In *Computer Vision–ECCV 2010*. Springer, 438–451.
7. Donald E Broadbent and Margaret HP Broadbent. 1987. From detection to identification: Response to multiple targets in rapid serial visual presentation. *Perception & psychophysics* 42, 2 (1987), 105–113.
8. Moira Burke and Robert Kraut. 2013. Using Facebook after losing a job: Differential benefits of strong and weak ties. In *Proceedings of the 2013 conference on Computer supported cooperative work*. ACM, 1419–1430.
9. Stuart K Card, Allen Newell, and Thomas P Moran. 1983. The psychology of human-computer interaction. (1983).
10. Justin Cheng, Jaime Teevan, and Michael S Bernstein. 2015. Measuring Crowdsourcing Effort with Error-Time Curves. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 1365–1374.
11. Lydia B Chilton, Greg Little, Darren Edge, Daniel S Weld, and James A Landay. 2013. Cascade: Crowdsourcing taxonomy creation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1999–2008.
12. Lacey Colligan, Henry WW Potts, Chelsea T Finn, and Robert A Sinkin. 2015. Cognitive workload changes for nurses transitioning from a legacy system with paper documentation to a commercial electronic health record. *International journal of medical informatics* 84, 7 (2015), 469–476.
13. Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 248–255.
14. Jia Deng, Olga Russakovsky, Jonathan Krause, Michael S Bernstein, Alex Berg, and Li Fei-Fei. 2014. Scalable multi-label annotation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 3099–3102.
15. Ethan Fast, Daniel Steffee, Lucy Wang, Joel R Brandt, and Michael S Bernstein. 2014. Emergent, crowd-scale programming practice in the IDE. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*. ACM, 2491–2500.
16. Li Fei-Fei, Asha Iyer, Christof Koch, and Pietro Perona. 2007. What do we perceive in a glance of a real-world scene? *Journal of vision* 7, 1 (2007), 10.
17. Eric Gilbert and Karrie Karahalios. 2009. Predicting tie strength with social media. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 211–220.
18. Ross Girshick, Jeff Donahue, Trevor Darrell, and Jagannath Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 580–587.
19. Marius Cătălin Iordan, Michelle R Greene, Diane M Beck, and Li Fei-Fei. 2015. Basic level category structure emerges gradually across human ventral visual cortex. *Journal of cognitive neuroscience* (2015).
20. Panagiotis G Ipeirotis. 2010. Analyzing the Amazon Mechanical Turk Marketplace. *XRDS: Crossroads, The ACM Magazine for Students* 17, 2 (2010), 16–21.
21. Panagiotis G Ipeirotis, Foster Provost, and Jing Wang. 2010. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*. ACM, 64–67.
22. Lilly C Irani and M Silberman. 2013. Turkopticon: Interrupting worker invisibility in amazon mechanical turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 611–620.
23. Suyog Dutt Jain and Kristen Grauman. 2013. Predicting sufficient annotation strength for interactive foreground segmentation. In *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, 1313–1320.
24. Tatiana Josephy, Matt Lease, and Praveen Paritosh. 2013. CrowdScale 2013: Crowdsourcing at Scale workshop report. (2013).

25. Ece Kamar, Severin Hacker, and Eric Horvitz. 2012. Combining human and machine intelligence in large-scale crowdsourcing. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems, 467–474.
26. David R Karger, Sewoong Oh, and Devavrat Shah. 2011. Budget-optimal crowdsourcing using low-rank matrix approximations. In *Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on*. IEEE, 284–291.
27. David R Karger, Sewoong Oh, and Devavrat Shah. 2014. Budget-optimal task allocation for reliable crowdsourcing systems. *Operations Research* 62, 1 (2014), 1–24.
28. Aniket Kittur, Ed H Chi, and Bongwon Suh. 2008. Crowdsourcing user studies with Mechanical Turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM, 453–456.
29. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.
30. Ranjitha Kumar, Arvind Satyanarayan, Cesar Torres, Maxine Lim, Salman Ahmad, Scott R Klemmer, and Jerry O Talton. 2013. Webzeitgeist: design mining the web. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 3083–3092.
31. Gierad Laput, Walter S Lasecki, Jason Wiese, Robert Xiao, Jeffrey P Bigham, and Chris Harrison. 2015. Zensors: Adaptive, Rapidly Deployable, Human-Intelligent Sensor Feeds. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 1935–1944.
32. Walter Lasecki, Christopher Miller, Adam Sadilek, Andrew Abumoussa, Donato Borrello, Raja Kushalnagar, and Jeffrey Bigham. 2012. Real-time captioning by groups of non-experts. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*. ACM, 23–34.
33. Walter S Lasecki, Kyle I Murray, Samuel White, Robert C Miller, and Jeffrey P Bigham. 2011. Real-time crowd control of existing interfaces. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, 23–32.
34. David D. Lewis and Philip J. Hayes. 1994. Guest Editorial. *ACM Transactions on Information Systems* 12, 3 (July 1994), 231.
35. Fei Fei Li, Rufin VanRullen, Christof Koch, and Pietro Perona. 2002. Rapid natural scene categorization in the near absence of attention. *Proceedings of the National Academy of Sciences* 99, 14 (2002), 9596–9601.
36. Tao Li and Mitsunori Ogihara. 2003. Detecting emotion in music. In *ISMIR*, Vol. 3. 239–240.
37. Liang Liang and Kristen Grauman. 2014. Beyond comparing image pairs: Setwise active learning for relative attributes. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 208–215.
38. Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft COCO: Common objects in context. In *Computer Vision–ECCV 2014*. Springer, 740–755.
39. Adam Marcus and Aditya Parameswaran. 2015. *Crowdsourced data management: industry and academic perspectives*. Foundations and Trends in Databases.
40. David Martin, Benjamin V Hanrahan, Jacki O'Neill, and Neha Gupta. 2014. Being a turker. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*. ACM, 224–235.
41. Winter Mason and Siddharth Suri. 2012. Conducting behavioral research on Amazons Mechanical Turk. *Behavior research methods* 44, 1 (2012), 1–23.
42. George A Miller and Walter G Charles. 1991. Contextual correlates of semantic similarity. *Language and cognitive processes* 6, 1 (1991), 1–28.
43. Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval* 2, 1-2 (2008), 1–135.
44. Amar Parkash and Devi Parikh. 2012. Attributes for classifier feedback. In *Computer Vision–ECCV 2012*. Springer, 354–368.
45. Mausam Daniel Peng Dai and S Weld. 2010. Decision-theoretic control of crowd-sourced workflows. In *In the 24th AAAI Conference on Artificial Intelligence (AAAI10*. Citeseer.
46. Mary C Potter. 1976. Short-term conceptual memory for pictures. *Journal of experimental psychology: human learning and memory* 2, 5 (1976), 509.
47. Mary C Potter and Ellen I Levy. 1969. Recognition memory for a rapid sequence of pictures. *Journal of experimental psychology* 81, 1 (1969), 10.
48. Adam Reeves and George Sperling. 1986. Attention gating in short-term visual memory. *Psychological review* 93, 2 (1986), 180.
49. Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C Berg, and Fei-Fei Li. 2014. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision* (2014), 1–42.

50. Olga Russakovsky, Li-Jia Li, and Li Fei-Fei. 2015. Best of both worlds: human-machine collaboration for object annotation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2121–2131.
51. Niloufar Salehi, Lilly C Irani, and Michael S Bernstein. 2015. We Are Dynamo: Overcoming Stalling and Friction in Collective Action for Crowd Workers. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 1621–1630.
52. Robert E Schapire and Yoram Singer. 2000. BoosTexter: A boosting-based system for text categorization. *Machine learning* 39, 2 (2000), 135–168.
53. Prem Seetharaman and Bryan Pardo. 2014. Crowdsourcing a reverberation descriptor map. In *Proceedings of the ACM International Conference on Multimedia*. ACM, 587–596.
54. Victor S Sheng, Foster Provost, and Panagiotis G Ipeirotis. 2008. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 614–622.
55. Aashish Sheshadri and Matthew Lease. 2013. Square: A benchmark for research on computing crowd consensus. In *First AAAI Conference on Human Computation and Crowdsourcing*.
56. K. Simonyan and A. Zisserman. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR* abs/1409.1556 (2014).
57. Padhraic Smyth, Michael C Burl, Usama M Fayyad, and Pietro Perona. 1994. Knowledge Discovery in Large Image Databases: Dealing with Uncertainties in Ground Truth.. In *KDD Workshop*. 109–120.
58. Padhraic Smyth, Usama Fayyad, Michael Burl, Pietro Perona, and Pierre Baldi. 1995. Inferring ground truth from subjective labelling of venus images. (1995).
59. Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y Ng. 2008. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics, 254–263.
60. Zheng Song, Qiang Chen, Zhongyang Huang, Yang Hua, and Shuicheng Yan. 2011. Contextualizing object detection and classification. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 1585–1592.
61. Hao Su, Jia Deng, and Li Fei-Fei. 2012. Crowdsourcing annotations for visual object detection. In *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*.
62. Omer Tamuz, Ce Liu, Serge Belongie, Ohad Shamir, and Adam Tauman Kalai. 2011. Adaptively learning the crowd kernel. *arXiv preprint arXiv:1105.1033* (2011).
63. Bart Thomee, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. 2016. YFCC100M: The New Data in Multimedia Research. *Commun. ACM* 59, 2 (2016). DOI:**http://dx.doi.org/10.1145/2812802**
64. Sudheendra Vijayanarasimhan, Prateek Jain, and Kristen Grauman. 2010. Far-sighted active learning on a budget for image and video recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 3035–3042.
65. Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2014. Show and tell: A neural image caption generator. *arXiv preprint arXiv:1411.4555* (2014).
66. Carl Vondrick, Donald Patterson, and Deva Ramanan. 2013. Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision* 101, 1 (2013), 184–204.
67. Catherine Wah, Steve Branson, Pietro Perona, and Serge Belongie. 2011. Multiclass recognition and part localization with humans in the loop. In *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2524–2531.
68. Catherine Wah, Grant Van Horn, Steve Branson, Subhrajyoti Maji, Pietro Perona, and Serge Belongie. 2014. Similarity comparisons for interactive fine-grained categorization. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 859–866.
69. Erich Weichselgartner and George Sperling. 1987. Dynamics of automatic and controlled visual attention. *Science* 238, 4828 (1987), 778–780.
70. Peter Welinder, Steve Branson, Pietro Perona, and Serge J Belongie. 2010. The multidimensional wisdom of crowds. In *Advances in neural information processing systems*. 2424–2432.
71. Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in neural information processing systems*. 2035–2043.
72. Jacob O Wobbrock, Jodi Forlizzi, Scott E Hudson, and Brad A Myers. 2002. WebThumb: interaction techniques for small-screen browsers. In *Proceedings of the 15th annual ACM symposium on User interface software and technology*. ACM, 205–208.
73. Dengyong Zhou, Sumit Basu, Yi Mao, and John C Platt. 2012. Learning from the wisdom of crowds by minimax entropy. In *Advances in Neural Information Processing Systems*. 2195–2203.