Reply & Supply: Efficient crowdsourcing when workers do more than answer questions

Thomas C. McAndrew, Elizaveta A. Guseva, James P. Bagrow *

Mathematics & Statistics, Vermont Complex Systems Center, University of Vermont, Burlington, Vermont, United States of America

* james.bagrow@uvm.edu

Abstract

Crowdsourcing works by distributing many small tasks to large numbers of workers, yet the true potential of crowdsourcing lies in workers doing more than performing simple tasks— they can apply their experience and creativity to provide new and unexpected information to the crowdsourcer. One such case is when workers not only answer a crowdsourcer's questions but also contribute new questions for subsequent crowd analysis, leading to a growing set of questions. This growth creates an inherent bias for early questions since a question introduced earlier by a worker can be answered by more subsequent workers than a question introduced later. Here we study how to perform efficient crowdsourcing with such growing question sets. By modeling question sets as networks of interrelated questions, we introduce algorithms to help curtail the growth bias by efficiently distributing workers between exploring new questions and addressing current questions. Experiments and simulations demonstrate that these algorithms can efficiently explore an unbounded set of questions without losing confidence in crowd answers.

1 Introduction

Crowdsourcing has emerged as a powerful new paradigm for accomplishing work by using modern communications technology to direct large numbers of people who are available to complete tasks (workers) to others who need large amounts of work to be completed (crowdsourcers) [1–4]. Crowdsourcing often focuses on tasks that are easy for humans to solve, but may be difficult for a computer. For example, parsing human written text can be a difficult task and optical character recognition systems may be unable to identify all scanned words [5–7]. To address this, the reCAPTCHA [8] system takes scanned images of text which were difficult for computers to recognize and hands them off to Internet workers for recognition. By having many people individually solve quick and easy tasks, reCAPTCHA is able over time to transcribe massive quantities of text. Crowdsourcing in general is especially important as a new vehicle for addressing problems of social good [9–11].

Deciding on an optimal way to assign particular tasks to workers, and in what order, remains an active area of research. For many problems, multiple worker responses to a task must be aggregated to determine a final answer [4] but often, a budget limits the total

crowdsourcing resources available [12–15], either due to financial limits when workers are compensated or time constraints where the speed or size of the crowd limits the number of tasks to be performed or questions to be answered. Most previous work on optimal task assignment takes the form of a Markov Decision Process (MDP) [12, 16]. MDP provides a rigorous mathematical framework to test policies for allocating tasks to workers [17]. Using MDP and other strategies, such as Thompson sampling [18], methods have been introduced to efficiently aggregate responses from workers, including consideration of which workers are most likely to be well suited for a given question based on their past performance on related questions [19–27].

However, to the best of our knowledge, past research has been limited to the case where a fixed set of tasks need to be accomplished, and the response of a worker to a task is only ever to complete the assigned task. In contrast, consider a crowdsourcing problem where workers are able to do more than perform tasks—they may be allowed to propose new questions as well as answers to a given question. The truest expression of crowdsourcing must incorporate the intuition and experience of workers, who are potentially capable of providing the crowdsourcer with far more actionable information for many problem domains [28–30]. While MDP made significant contributions to the design of question assignment algorithms, when the question set is growing due to the crowd, MDP does not naturally account for the hidden state transitions needed to represent newly contributed questions. The lack of research on algorithms accounting for growing question sets reveals a gap in our abilities to efficiently assign questions to workers.

To this end, we study a type of crowdsourcing problem we term *Reply & Supply*. As workers answer a given question (Reply), they are given the opportunity to propose a related question (Supply). Example applications of Reply & Supply include:

- Exploring social networks (*"Are Alice and Bob friends?" "Who else is friends with Alice?" "With Bob?"*)

- Product recommendations (*"Have you bought a camera and laptop together?" "What else would someone buy when buying a camera?"*)

- Image classification (*"Does this photo contain a horse and a mountain?" "What else does it contain?"*)

- Causal attribution (*"Do you think 'hot weather' causes 'violent crime'?" "What causes 'violent crime'?"*)

- Health informatics: Crowdsourcing patient reports to find connections between co-occurring symptoms, new drug interactions, etc. (*"Do you suffer from symptom X?" "What other symptoms do you have?" "Do you take drug Y?" "What other drugs do you take?"*)

In all these examples, new questions can be built by combining crowd-suggested responses with the components of the original question, leading to the creation of a *network structure* of interrelated questions. To explore a social network, for example, if a worker responds that Alice and Bob are friends (Reply) and also proposes that Alice and Carol are friends (Supply), then a new question (*"Are Alice and Carol friends?"*) is formed that other workers can consider and that links to other questions related to Alice. Further, we will show that this network representation naturally generalizes to non-network question sets, and the methods we develop here are fully applicable to both question sets and question nets.

Question networks can be studied with tools from network science that consider the statistical properties governing how theoretical and real-world networks grow and behave [31–38]. One property, the scale-free or heavy-tailed degree distribution [33], where most nodes in the

network have low degree but some very high-degree nodes do exist, holds in many real-world networks. How a scale-free network grows over time introduces biases ('first-mover advantage') that are also inherent in a growing crowdsourced experiment.

In brief, this manuscript makes the following contributions:

1. The introduction of a growing network of linked questions with an accompanying theoretical analysis;

2. The use of Thompson sampling to develop crowd-steering algorithms that enable efficient exploration of an evolving set of tasks or questions without losing confidence in answers;

3. Simulations and real-world crowdsourcing experiments that validate the efficiency and, to some extent, the accuracy of the crowdsourcing performed under the crowd-steering algorithm.

The rest of this paper is organized as follows: Section 2 poses the generic crowdsourcing problem we focus on, analyzes a simple graphical model of how a growing question net is built by a crowd, and uses this model to motivate methods for efficiently assigning questions to workers as the question net grows. Section 3 describes experiments and evaluation metrics to test the proposed theory and methods with both simulated and real-world crowdsourcing tasks. Section 4 presents the results of these experiments and Sec. 5 concludes with a discussion of these results and future work.

2 Methods

Here we introduce a graphical model of a growing question network where questions consider the presence or absence of a relationship between two items (Sec. 2.1). We study the network's properties under a null condition where the crowdsourcer assigns questions to workers randomly without use of a "steering" algorithm to provide guidance (Sec. 2.2). We then use these properties to develop a probability matching algorithm which provides said guidance to the crowdsourcer (Sec. 2.3).

2.1 Crowdsourcing growing question networks

We model a growing set of questions (or tasks) as a graphs where nodes are items and edges or links represent questions relating pairs of items. A question network $G = (V, E)$ is composed of a set of nodes V and a set of edges E, where $|V| = N$ and $|E| = M$. Edge attributes record the answers given by workers, i.e., associated with each edge is a categorical variable storing the counts of worker responses. Those workers may also propose new questions (i.e., new combinations of new or existing items), leading to new nodes and edges. This network model also accommodates non-network question sets, for example by considering each question as a disjoint edge.

As an example of such a network, consider a *synonym proposal task* (SPT) where workers are asked if two words u and v are synonyms. The question is the link (u, v) between two items u and v representing those words. After replying to the question, the worker may also supply another word w which is a synonym for u, for v, or for both words. This grows the question network by introducing new questions linking items (u, w), or items (v, w), or both (u, w) and (v, w). The degree k_i of item i counts the number of questions linking item i to other items.

We focus on cases, such as the SPT, where questions have binary answers, e.g, when workers are asked whether or not a link between two items should exist. Edge attributes on links capture the number of 'yes' and 'no' answers given by workers. However, this graph representation is flexible enough to allow edge attributes to contain any number of dimensions and

there are no restrictions imposed on how workers propose questions. Moreover, this graphical model is capable of representing growing question sets without such relations, for example, a collection of N disjoint questions always containing the response items 'True' and 'False' only may be a two node, N multi-edge graph. While not a particularly meaningful representation, it demonstrates that the algorithms we develop are applicable to general crowdsourcing tasks without modification. Lastly, one can also extend this model to non-binary, multiple choice questions in several ways, including representing questions as hyperedges in a hypergraph.

2.2 Null model

We propose a generative null model for a growing question network [39, 40]. Beginning from a network with one question, a crowdsourcer randomly chooses existing questions to send to workers also chosen at random. Those workers answer the questions and then with some probability also propose new questions. We study the properties of the network under these assumptions to motivate the development of a probability matching algorithm that can allow a crowdsourcer to efficiently explore the growing question network.

The network begins (at time $t = 0$) with two nodes and one undirected link connecting those nodes, representing a single question considering two items. Under the null model, every link (i, j) has an associated *innovation rate* ρ_{ij}. The innovation rate for (i, j) defines the probability a random worker will introduce a new question into the network when presented with question (i, j). If she chooses to innovate, the new question may relate to either or both of the items i and j of the original question the worker was given.

Specifically, suppose a random worker is given question (u, v) relating items u and v. Under the null model:

1. The worker answers question (u, v) with probability 1.

2. The worker proposes a new item w to study with probability ρ_{uv}:

   a. w is linked to one of the items of the original question with probability γ_{uv}. A single new question, either (u, w) or (v, w) chosen uniformly at random, is introduced;

   b. otherwise, w links to both items of the original question with probability $1 - \gamma_{uv}$. Two new questions, (u, w) and (v, w), are introduced.

3. Repeat from (1) with another sampled question and worker until termination.

This model is tractable but quite basic and does not consider many potential details. For example, it assumes that while questions may have different innovation rates, workers do not. However, for sufficiently large numbers of workers, the average response is always going to be the primary concern, particularly in most crowdsourcing tasks which need to aggregate multiple worker responses to decide upon a final answer for a question. If it is necessary, a crowdsourcer interested in accounting for variation between workers can propose a statistical model for their features, and then use statistical inference to estimate these worker parameters during crowdsourcing (see also the Discussion).

We now prove several *average* properties of this null model. Studying the characteristics of the randomly growing, uncontrolled network informs policies that a crowdsourcer may use to manipulate the network (such as the algorithm we develop in Sec. 2.3). Many of these results are also informative for non-network growing question sets.

The first theorem describes question growth in the random uncontrolled network.

Theorem 1 (Rate of question growth). *The total number of links $M(t)$ as a function of time t is, on average, $M(t) = \eta t + 1$ where $\eta = \langle \rho \rangle \, (2 - \langle \gamma \rangle)$ is termed the **exploration rate**.*

Proof. For the network to grow, a worker must suggest an additional question, which occurs with probability on average $\langle \rho \rangle$ (average of ρ_{ij}). Once the worker commits to a suggestion, one question is added with probability on average $\langle \gamma \rangle$ or two questions are added with probability on average $1 - \langle \gamma \rangle$. Combining these two possibilities, the total number of questions grows on average over one timestep according to

$$M(t + 1) = M(t) + \langle \gamma \rangle \langle \rho \rangle + 2\langle \rho \rangle (1 - \langle \gamma \rangle),$$

with initial condition $M(0) = 1$ representing the single seed question of the network. Making a continuum approximation, this difference equation becomes $M'(t) = \langle \rho \rangle (2 - \langle \gamma \rangle)$, which has solution

$$M(t) = \eta t + 1, \tag{1}$$

where the *exploration rate* $\eta \equiv \langle \rho \rangle (2 - \langle \gamma \rangle)$ plays an important role in the overall network growth.

The number of links grows linearly with a rate η that combines the average rates $\langle \rho \rangle$ and $\langle \gamma \rangle$. Intuitively, the network grows faster if questions are more likely to be innovative (larger $\langle \rho \rangle$), and/or the worker is able to suggest a question for both items at the same time (smaller $\langle \gamma \rangle$).

The solution to the rate equation for question growth can be used to compute the mean number of worker answers per question:

Theorem 2 (Mean answer density). *The mean answer density* (*number of answers per question*) $\langle A \rangle \to 1/\eta$ *as* $t \to \infty$.

Proof. The mean number of answers per question is

$$\langle A \rangle = \frac{\text{total number of answers}}{\text{total number of questions}}. \tag{2}$$

At every time step a question in the network accumulates a single answer from a worker. The denominator of Eq (2) is the solution Eq (1), and so the average density of answers per question is

$$\langle A \rangle = \frac{t}{\eta t + 1} = \frac{1}{\eta + \frac{1}{t}} \to \frac{1}{\eta}$$

as $t \to \infty$.

The mean answer density correlates with the overall uncertainty in the crowdsourcing since there is generally more certainty (but not necessarily correctness) in crowd responses when more workers on average have independently answered questions. Controlling the answer density, and therefore the certainty, now boils down to controlling the exploration rate η. The mean answer density's dependence on η also encapsulates an 'exploration-exploitation' trade-off: lower η leads to higher answer density, but at the cost of less exploration in the network; higher η increases the exploration but lowers answer density and makes more uncertainty in the network. In this null model, the crowdsourcer does not make choices that can exploit this, but tuning between these poles is a key component of the probability matching algorithm we introduce in Sec. 2.3.

The previous two theorems govern global properties of random question networks. We now turn to properties of individual items within the network to explain the unequal distribution of questions attached to items:

Theorem 3 (Rich-get-richer mechanism). *A node i entering the network at time t_i will gain degree, on average, as $k_i(t) = \frac{\eta}{\langle \rho \rangle} \left(\frac{1 + \eta t}{1 + \eta t_i} \right)^{1/2} \mathcal{H}(t - t_i)$, where $\mathcal{H}$ is the Heaviside function.*

Proof. An existing item i only gains a question when the crowdsourcer chooses a question attached to i and the worker answering that question proposes a new question involving i. A question (i, j) associated with item i is selected by the crowdsourcer with probability $k_i(t)/M(t)$, where $k_i(t)$ is the degree (number of questions) of i at time t. After the worker answers question (i, j) she must innovate (probability $\langle \rho \rangle$) with an item w that is not already a neighbor of i (and $w \neq i$) and the new question must be (w, i) (probability $\langle \gamma \rangle / 2$) or it must be two questions (w, i) and (w, j) (probability $1 - \langle \gamma \rangle$). If the worker introduces question (w, j) only (probability $\langle \gamma \rangle / 2$) then i does not gain a new question and so this possibility does not contribute to $k_i(t)$. Combining these possibilities together, $k_i(t)$ evolves on average according to

$$k_i(t) = k_i(t-1) + k_i \frac{\langle \rho \rangle}{M(t-1)} \left(\frac{\langle \gamma \rangle}{2} + (1 - \langle \gamma \rangle) \right). \tag{3}$$

We approximate and simplify this difference equation as before:

$$\frac{dk_i}{dt} = k_i \frac{\langle \rho \rangle}{M(t)} \left(1 - \frac{\langle \gamma \rangle}{2} \right) = \frac{k_i}{2} \left(\frac{\eta}{\eta t + 1} \right); k_i(t_i) = \frac{\eta}{\langle \rho \rangle}, \tag{4}$$

where $k_i(t_i)$ is the initial degree when item i was introduced at some time t_i. Solving Eq (4) results in

$$k_i(t) = \frac{\eta}{\langle \rho \rangle} \left(\frac{1 + \eta t}{1 + \eta t_i} \right)^{1/2} \mathcal{H}(t - t_i). \tag{5}$$

We see from this derivation that the rich-get-richer, preferential attachment mechanism [33] is automatic when questions are chosen at random: an item i is more likely to appear in a sampled question the more questions it has, and therefore items with more questions are more likely to gain further questions than other items. Further, the degree of an item depends critically on two quantities. The first, the ratio of exploration rate η to $\langle \rho \rangle$, equally affects all items in the network. The second, the time of entry t_i, dampens the growth of items that enter the network late and increases the growth of earlier items. This phenomena is often called the 'first mover's advantage', and in the context of crowdsourcing a growing network, items entered earlier in the system accrue more questions than later items.

Using the local estimate of item degree to derive the global degree distribution of the network, we find:

Theorem 4 (Degree Distribution). *The degree distribution of the growing question network*

$$P(k(t)) \rightarrow 2 \left(\frac{\eta}{\langle \rho \rangle} \right)^2 \frac{1}{k^3} \tag{6}$$

as $t \rightarrow \infty$.

Proof. Following [39], begin with the cumulative probability distribution of item i's degree:

$$
\begin{aligned}
P(k_i(t) < k) \quad &= P\left(\frac{1}{\langle \rho \rangle} \left(\frac{1 + \eta t}{1 + \eta t_i} \right)^{1/2} < k \right) \\
&= 1 - P\left(t_i < \eta \left(\frac{1}{k \langle \rho \rangle} \right)^2 (1 + \eta t) - \frac{1}{\eta} \right).
\end{aligned} \tag{7}
$$

Meanwhile, the entry times t_i of items into the network follow a distribution proportional to $\langle \rho \rangle$ uniformly through time:

$$P(t_i = t) \propto \langle \rho \rangle,$$

and, after normalizing, we discover the time of entry follows a uniform distribution. Referring back to Eq (7) and using the integral definition of a cumulative distribution,

$$P(k_i(t) < k) = 1 - \frac{1}{t}\left(\frac{1}{k\langle\rho\rangle}\right)^2 \eta(1+\eta t) - \frac{1}{\eta t}. \tag{8}$$

Lastly, differentiating Eq (8) with respect to k gives the degree distribution:

$$\frac{\partial P(k_i(t) < k)}{\partial k} = P(k(t)) = \frac{2\eta(\eta t + 1)}{t\langle\rho\rangle^2}\frac{1}{k^3} \rightarrow 2\left(\frac{\eta}{\langle\rho\rangle}\right)^2 \frac{1}{k^3} \tag{9}$$

as $t \rightarrow \infty$.

Our theoretical analysis is supported by simulations of growing question networks (Fig 1). We conducted 5,000 simulations and recorded the degree distribution $P(k)$ and degree k of items across different values of exploration rate η and time of item entry t_i. Fig 1(a) validates the slower rate of question accrual for late arriving items, and Fig 1(b) shows the degree distribution's match to theory by the collapse of each curve over multiple values of η.

2.3 Probability matching algorithm for growing question sets and nets

Most algorithms for steering workers towards questions choose questions by defining a metric that captures important characteristics in the system. For example, algorithms stressing accuracy often build metrics that reward higher numbers of answers for questions, achieving a p-value below a pre-defined threshold, or diminishing the variance of questions.

The framework of probability matching, specifically Thompson sampling [18] (TS), is one of the most powerful ways to efficiently choose from a set of dynamic "options" when choices
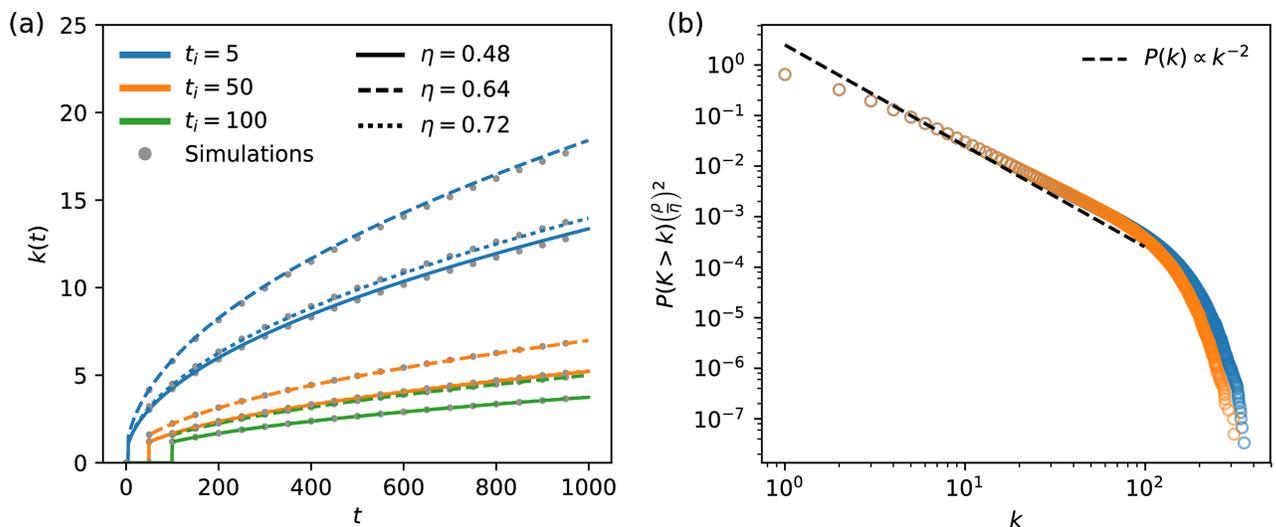
Fig 1. Agreement of theoretical predictions of network growth under the null model with simulations for several different choices of parameters.

https://doi.org/10.1371/journal.pone.0182662.g001

must be made with limited information. Unlike greedy algorithms, one of the strengths of TS is that its stochastic nature prevents choosing locally optimal questions only.

To Thompson sample from a set of options, one assumes a random variable X which follows a distribution $\varphi(x|\theta_i(t))$, where $\theta_i(t)$ is a set of parameters specific to i at time t. One draws an $x_i(t)$ for each option i and selects the option j with the smallest x (or largest x, depending on what x represents), $j = \arg\min_i x_i(t)$. After option j is played (in our case, the worker's answer is received), the parameters for option j are updated. Often x is a Bernoulli random variable and it is natural for φ to be the conjugate Beta distribution with parameters α, β which are updated depending on whether $x = 0$ or $x = 1$.

For specific problems, TS depends on an appropriate reward function. In the context of crowdsourcing, one generally cannot verify the accuracy of crowd answers, so the best choice is to reward certainty or consensus. If the crowd is consistent in their responses for a given question, then that implies the question is being answered as well as possible under current conditions. Thus, in contrast to the Bernoulli Bandit problems typically studied with TS, we do not want to reward 'yes' answers over 'no' answers only. Instead, we want to reward choices that lower the crowdsourcer's measure of uncertainty for questions.

A natural measure of uncertainty for a categorical random variable is the Shannon entropy. However, efficiency is also important to a crowdsourcer. A yes/no question that has 200 responses which are evenly split is very different than a question with 2 responses which is also evenly split, despite having the same entropy. Generally, the crowdsourcer would prefer to assign a worker to the latter question, as there is greater hope of lowering its uncertainty.

This argument guides us to choosing a metric involving both the total number of answers to a question and how evenly distributed those answers were over the categories of that question. We introduce a metric called *link bias* (d) that is sensitive to the uncertainty of a question, but unlike entropy, also accounts for the total number of answers. To begin, the multinomial distribution, with $C - 1$ parameters, naturally models the distribution of a categorical question's total number of answers T across C possible answers, and the Dirichlet distribution, conjugate to the multinomial, can estimate the parameters of the multinomial. Since we expect no available prior information, a non-informative prior can be used. In the case of two categories, which we focus on, the Dirichlet distribution reduces to the Beta distribution ($B(\alpha, \beta)$).

To define question uncertainty, we need a reference point. At a question's peak uncertainty, workers have answered evenly among the question's (C) categories causing an equal proportion of answers per category. In our binary case ($C = 2$), this corresponds to a proportion of $1/C = 1/2$. The link bias d transforms the proportion of answers for question (i, j) to the distance from maximum uncertainty with $d \equiv |2 - p_{ij}(1)|$, where $p_{ij}(1)$ is the fraction of '1' or 'yes' or 'true' answers. When $p_{ij} \sim B(\alpha, \beta)$, the probability density of d becomes

this shoud read: d = | 1/2 - pij(1) |

$$\varphi(d|\alpha, \beta) = \frac{(1 - 2d)^{\alpha-1}(1 + 2d)^{\beta-1} + (1 + 2d)^{\alpha-1}(1 - 2d)^{\beta-1}}{B(\alpha, \beta)2^{\alpha+\beta-2}}, \tag{10}$$

where for simplicity the dependence of α, β on (i, j) has been suppressed. Intuitively, a low link bias ($d \approx 0$) occurs when the crowd is evenly split among possible answers, while a high link bias (at most $d = 1/2$) tells us the crowd converged on a single category.

However, the link bias alone may not sufficiently steer the crowdsourcer to choose questions with a lower number of answers. If needed, we can combine a preference for sampling questions with few answers, with a preference for questions that are uncertain, by weighting

Eq (10) by the current number of answers to define a new 'weighted phi' metric φ_N:

$$\varphi_N(d \mid \alpha, \beta) \equiv \frac{N_{ij}\varphi(d|\alpha, \beta)}{\sum_{uv \in E} N_{uv}}, \tag{11}$$

where N_{ij} is the total number of answers to question (i, j) at the time of sampling.

Thompson sampling of questions via φ or via φ_N defines the two probability matching algorithms we propose. These algorithms handle growing networks of questions automatically and are fully applicable to problems without graphical relations between questions. We will conduct experiments on growing question networks testing the relative performance of both algorithms, and comparing them to other null or control baseline strategies, such as randomly choosing questions.

3 Experiments

We conducted two experiments to test the theoretical analysis and the sampling methods. For the first experiment, we simulated crowdsourcing of a growing question network with a commonly used benchmarking dataset by superimposing two distinct network structures onto a previously conducted crowdsourcing task [12], where questions have been time-ordered to mimic a growing question network, and used this to test three different question sampling algorithms. For the second experiment, we conducted real-world crowdsourcing using the Mechanical Turk crowdsourcing platform [2].

3.1 Experiment 1

To determine the effectiveness of choosing questions based on link bias, we first performed a five-armed experiment using the Recognizing Textual Entailment (RTE) dataset [12], a set of 8,000 binary answers (0 or 1) to 800 unique questions.

For simulating question growth, we superimposed graph structures onto the question set to link the 800 questions together. As mentioned in the introduction, many crowdsourcing problems naturally possess a network structure; here we imposed a structure on the RTE dataset only because it allows us to use the same benchmark dataset that many other researchers have studied. We built 5,000 Erdős-Rényi (ER) and Barabási-Albert (BA) networks [41]. These two options represent two extremes of network structure, and were chosen to test question sampling algorithms over different classes of networks. Briefly, an ER network [31] (specifically the $G(n, m)$ formulation) starts with a set of N nodes and 0 links; a pre-specified number of links M are placed in the network choosing randomly without replacement from all possible $\binom{N}{2}$ pairs of nodes. In contrast, the BA network [33] starts with 2 nodes joined by a single link, nodes are added one at a time until all N nodes are placed, and each new node attaches to m_0 existing nodes in the network. New nodes attach to an existing node i with probability $k_i/\sum_{n \in N} k_n$, a mechanism that is often called *preferential attachment*.

For simulation purposes, each ER network realization must contain exactly 400 nodes, 800 links, and be connected. BA networks are connected by design; we still enforced the same number of nodes and links as the ER networks. Each simulated crowdsourcing was initialized with one question (a link in the network connecting two corresponding item) chosen at random from the underlying network. During the simulated crowdsourcing, workers answer a question with a 1 with probability equal to the proportion of 1's observed in the original RTE dataset for that question, otherwise the worker answers 0. Next, and with probability $\langle \rho \rangle$, a new node (item) is introduced into the network by selecting randomly from the unseen neighbors of either i or j within that simulation's graph. (This differs slightly from the analytic null

model because there is no $\langle \gamma \rangle$. Instead, two links are formed automatically if the newly introduced item is linked to both i and j in the superimposed network.) If there are no new items to add corresponding to the selected question, this iteration is undone and the algorithm continues. All simulations were run with $\langle \rho \rangle$ = 0.20 for 6,000 time steps.

Simulations were performed independently for each of five arms. The condition of each arm governs how questions are selected by the simulated crowdsourcer:

Random: The first arm of the experiment had a condition where questions (links) were chosen randomly from the pool of already visited links.

Looping: The second arm used a *looping* question sampling algorithm. The first link that entered the system is answered by a worker, then the second link in the system is given to a worker, then the third link and so on. When the algorithm reaches the most recent link within the system it starts again from the oldest link.

Binomial sampling: This strategy selects questions (i, j) based on p-values for a two-sided binomial test that the proportion $p_{ij}(1)$ is significantly different from 1/2. If the p-value of this exact test is small, then it is likely the crowd has already reach consensus on that question and it is not worthwhile to sample that question further. The sampled question was chosen randomly from the set of questions which have a p-value >0.2 and which have received fewer than 10 answers (at the time of sampling)

Thompson sampling with φ: The fourth arm uses Thompson sampling to select links based on link bias (φ).

Thompson sampling with φ_N: As in the fourth arm but links are Thompson sampled with φ_N instead of φ.

This experiment can demonstrate the strengths and weaknesses of selecting links based on these different sampling strategies, and, because it is synthetic, many trials can be conducted while avoiding the costs associated with a new crowdsourcing experiment. Results of Experiment 1 are presented in Sec. 4.

3.2 Experiment 2: Synonym proposal task

This three-armed experiment created new question networks grown from a single seed question (link), and evaluated the φ_N-based Thompson sampling and Binomial sampling versus Random sampling. We paid US-based workers on Amazon's Mechanical Turk crowdsourcing platform [2, 42] to participate in a **synonym validation and proposal experiment**. Synonymy proposal is a good test application for the question sampling algorithms we study because workers can easily understand the question and are capable of proposing new questions (by suggesting new synonyms). Of course, data on synonymy relations are available in lexical resources such as WordNet [43], which we used in this specific task for assessing the accuracy of proposed synonyms (see below), but our primary goal with this experiment is not crowdsourcing a new thesaurus but testing the different question sampling strategies.

In Experiment 2, each worker completes synonymy tasks at a compensation of $0.08 USD per task. Each synonymy task gives a pair of words to a worker and asks whether or not they are synonyms. After a worker answer either 'yes' or 'no', we allow the worker to suggest additional synonyms for each word of the given pair, or a single synonym associated with the combined word pair. A screenshot of the web form used for this task is shown in Fig 2.

Three independent crowdsource networks were built, one for each arm. All three networks began with the same seed question (the word pair *patriotic*, *person*). All other word pairs were proposed by the crowd. The question sampling algorithms draw from all previous worker answers and suggested questions within their respective arms to deliver a question to the next queued worker. The first arm (Random sampling) chooses links using the same methodology as the random arm from Experiment 1, which also closely matches the null model we studied (Sec. 2.2). The second arm (Binomial sampling) selects links according to the Binomial sampling algorithm introduced in Experiment 1. Lastly, the third arm (Thompson sampling) selects links according to Thompson sampling of φ_N. Results for Experiment 2 are presented in Sec. 4.

3.3 Evaluation metrics

For the first experiment, we measure five attributes across the simulated crowdsourcings to compare the different question sampling algorithms. At each time step t, for each simulated network we record network properties f_{nodes}, the fraction of items, and f_{edges}, the fraction of questions:

$$f_{\text{nodes}} = \frac{|V(t)|}{|V(\infty)|}; \ f_{\text{edges}} = \frac{|E(t)|}{|E(\infty)|}, \tag{12}$$

where $V(t)$ is the set of items at time t, $V(\infty)$ is the set of all items at the end of the experiment, $E(t)$ is the set of questions at time t, and $E(\infty)$ is the set of all questions at the end of the experiment.

Next, we record the entropy S and link bias d, averaged over all currently visible questions, to quantify uncertainty in the network:

$$\langle S \rangle = -\frac{1}{|E(t)|} \sum_{ij \in E(t)} \sum_{x \in \{0,1\}} p_{ij}(x) \log_2 p_{ij}(x) \tag{13}$$

and

$$\langle d \rangle = \frac{1}{|E(t)|} \sum_{ij \in E(t)} |2 - p_{ij}(1)|, \tag{14}$$

where $p_{ij}(x)$ is the (Laplace-smoothed) fraction of binary answers of x for question (i, j) (at time t).

The final evaluation metric, mean answer density, measures how many answers are given per question in a particular network (see also Thm. 2):

$$\langle A \rangle = \frac{1}{|E(t)|} \sum_{ij \in E(t)} \sum_{x \in \{0,1\}} N_{ij}(x), \tag{15}$$

where the $N_{ij}(x)$ represents the count of answer x for question (i, j) (at time t).

Validating proposed synonyms. A factor that motivated us to choose the synonym proposal task as our crowdsourcing example is that synonym proposal can, in principle, be validated. Therefore, we will measure both crowd consensus (measured by $\langle S \rangle$ or $\langle d \rangle$) and, as best we can, if Experiment 2's crowdsourcing algorithms lead to different quality rates of synonyms —are we trading off quality for efficiency?

However, measuring synonymy from natural language text is challenging. In principle, all that is needed is a complete thesaurus, meaning a complete lookup table of all words and all their synonyms, perhaps with weights denoting the degree of relatedness between a word and
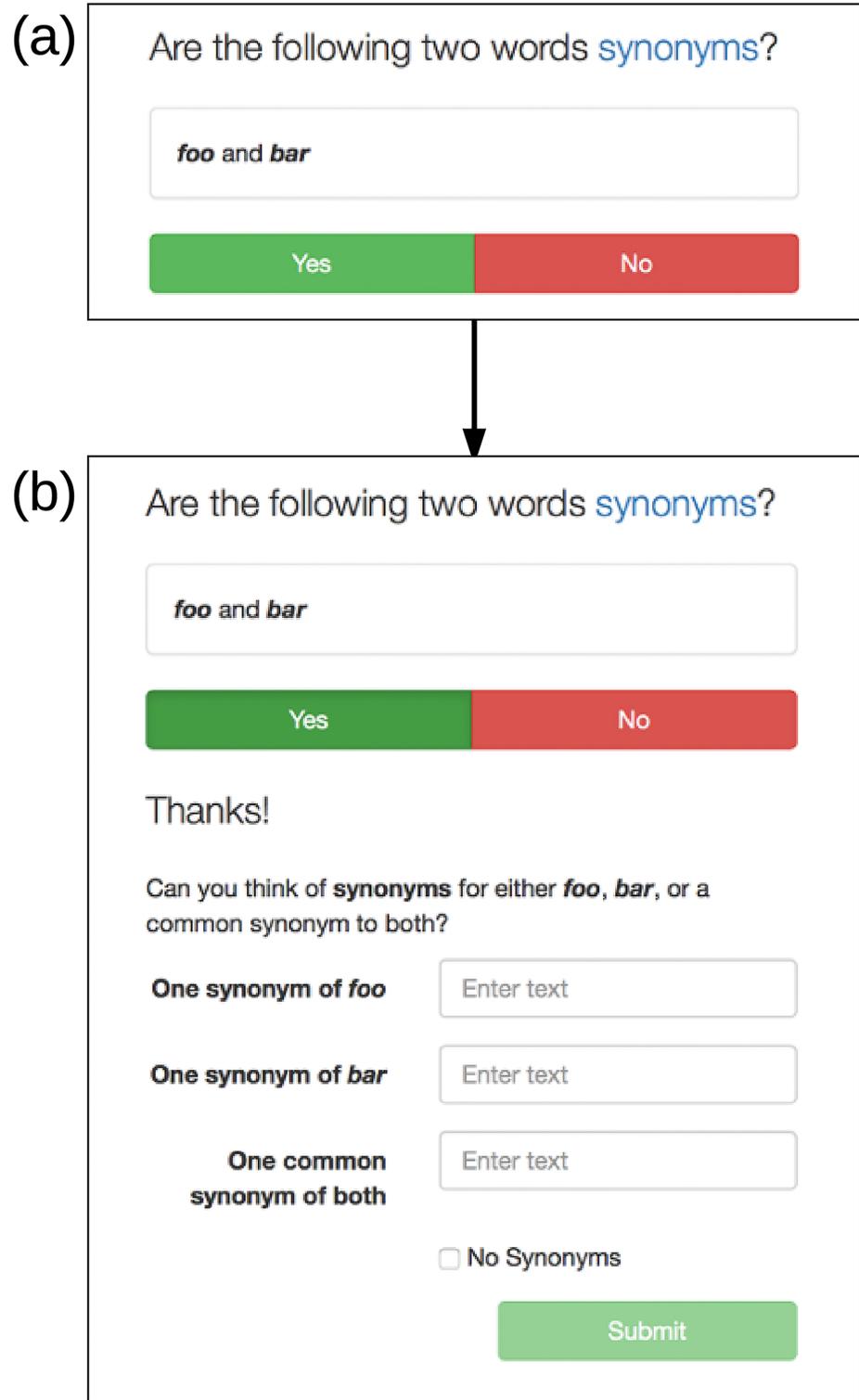
Fig 2. Screenshots of the Mechanical Turk web interface for the synonymy proposal task (Experiment 2). After replying 'yes' or 'no' (a), the form expands for the worker to supply new potential synonym pairs (b).

https://doi.org/10.1371/journal.pone.0182662.g002

its synonym and accounting for all possible contexts in which those words may appear. However, without such an exhaustive resource, it can be challenging to determine synonyms, especially when workers may introduce typos, may propose different forms (*runs, running, ran*) of the same root lemma (*run*), or they may propose a multi-word phrase (MWP) which may have a synonymous meaning but where such a meaning is difficult to determine computationally.

Given the challenges of measuring synonymy, we applied two measures to the synonym word pairs (u, v) proposed by workers during the crowdsourcing experiments:

Shared WordNet lemmas The first measure starts by determining for each word w the set of all forms of all its synonym lemmas as encoded in WordNet [43]:

$$L(w) = \bigcup_{s \in \text{synsets}(w)} \bigcup_{\ell \in \text{lemmas}(s)} \ell, \tag{16}$$

where *synsets*(w) is the set of all synonym forms stored in WordNet (we merge sets across parts-of-speech and take *synsets*$(w) = \emptyset$ if w is not present in WordNet). We then say that the two words u and v are synonyms if they share at least one lemma, i.e. that $|L(u) \cap L(v)| > 0$, otherwise they are not synonyms. This is a relatively strict test, and fails to account for many MWPs and natural language concerns such as misspellings, so we expect many (u, v) pairs that workers deem synonyms to be missed by this measure and therefore the actual proportion of synonymous word pairs may be much higher.

Word vector similarity The second measure we employ is based on the meanings encoded by the "word2vec" word embedding algorithm [44]. Word2vec uses a neural network model to learn low-dimensional vector representations of words based on their contextual co-occurrence patterns over a very large text corpus. Supported by the distributional hypothesis [45], the contexts encoded in these vectors are then considered to capture to some extent the meanings and relationships of these words such as, for example, analogous relationships (*Berlin is to Germany as Paris is to France*). Given a pre-trained set of 300-dimensional vectors trained on a 100B word corpus taken from Google News, we define the similarity between two words (or MWPs, if the MWPs are present in the vector data) u and v as their cosine similarity:

$$\text{similarity}(u, v) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \|\mathbf{v}\|}, \tag{17}$$

where $\mathbf{w}$ represents the associated word vector for word or MWP w. If either u or v is not present in the word2vec vector data, we exclude that pair from our analysis (this occurred in Experiment 2 for approximately 17.9% of crowd-proposed word pairs for the Random sampling experiment, 19.8% for Binomial sampling, and 10.9% for Thompson sampling).

4 Results

Experiment 1

[Fig 3](#) displays the five evaluation metrics associated with Experiment 1, averaged over the 5,000 ER and BA networks. (For simulated Binomial sampling only, note that we required questions to have fewer than 30 answers at the time of sampling, not 10 as discussed previously, to provide more simulation statistics.) The Binomial sampling and φ_N Weighted Thompson sampling algorithms outperformed all others in exploration metrics across ER and BA networks. Both methods explored more of the network, and faster, than other methods, as evidenced by $\langle f_{edges} \rangle$ and $\langle f_{nodes} \rangle$. Weighted Thompson sampling performed best at minimizing
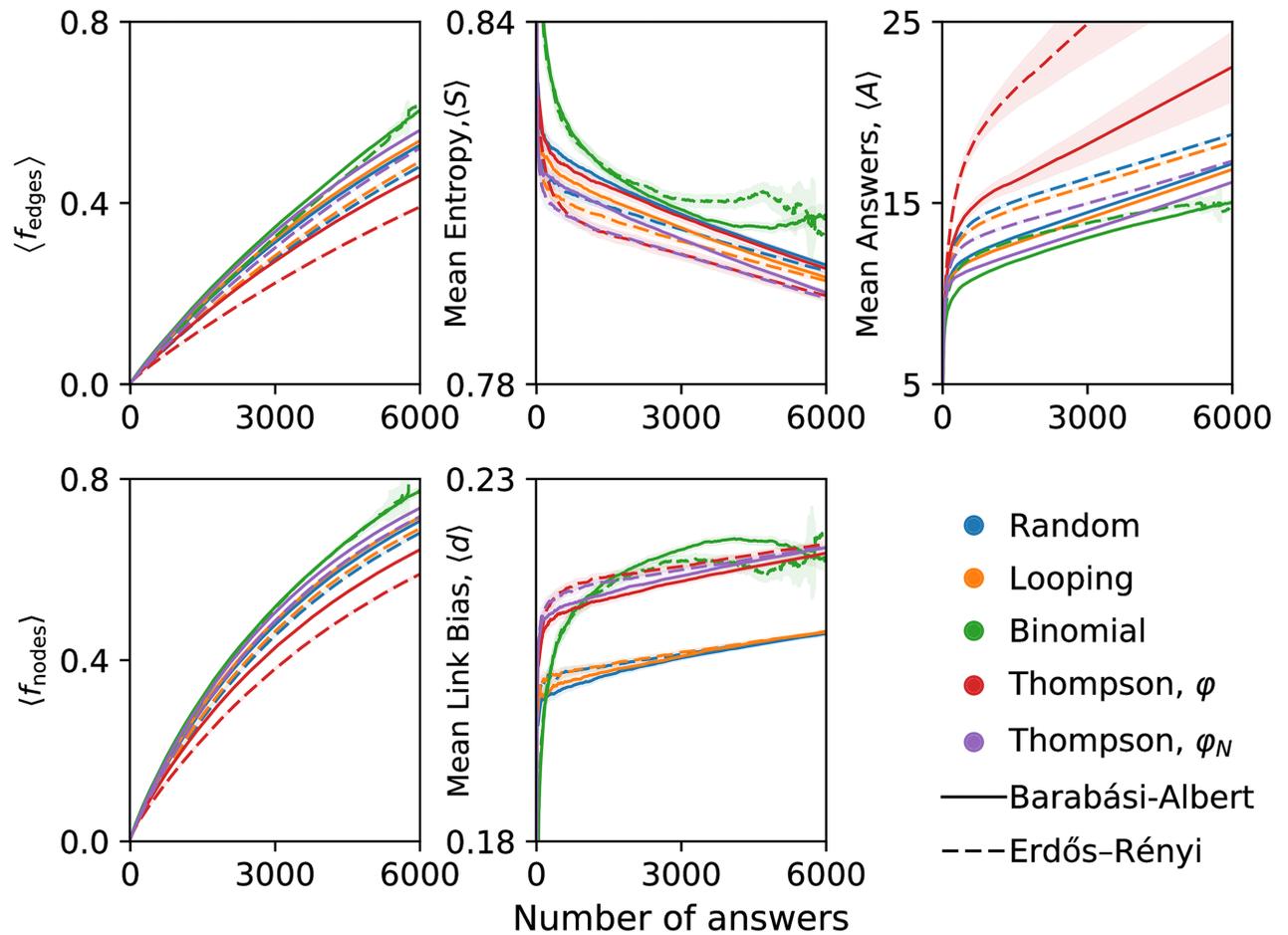
Fig 3. Experiment 1's evaluation metrics for five different question sampling algorithms.

the uncertainty of answers, as measured by lower entropy $\langle S \rangle$ and higher link bias $\langle d \rangle$. In contrast, Binomial and Thompson sampling φ were inconsistent for these two metrics. Lastly, Binomial and φ_N Weighted Thompson sampling also required fewer answers than other algorithms (lower $\langle A \rangle$).

Binomial sampling slightly outperformed φ_N Weighted Thompson sampling in many metrics. However, Binomial sampling has a distinct drawback: the thresholds used to sample questions may lead to a situation where no questions meet its sampling criteria. This is visible in the simulation curves, which are quite noisy due to individual simulations which terminated too early. Of course, this can be fixed by any of several means, such as falling back to random sampling when no questions meet the criteria, or tuning the cutoffs used in Binomial sampling. But Thompson sampling avoids these complexities entirely.

The overall performance of Binomial sampling and φ_N-based Thompson sampling in simulated crowdsourcing nominates them as candidate algorithms for Experiment 2's real crowdsourcing.

Experiment 2

Fig 4 shows the constructed networks for each arm of the Synonym Proposal Task (the task is described in Fig 2). Qualitatively, all three networks appeared similar. Quantitatively, (Table 1)
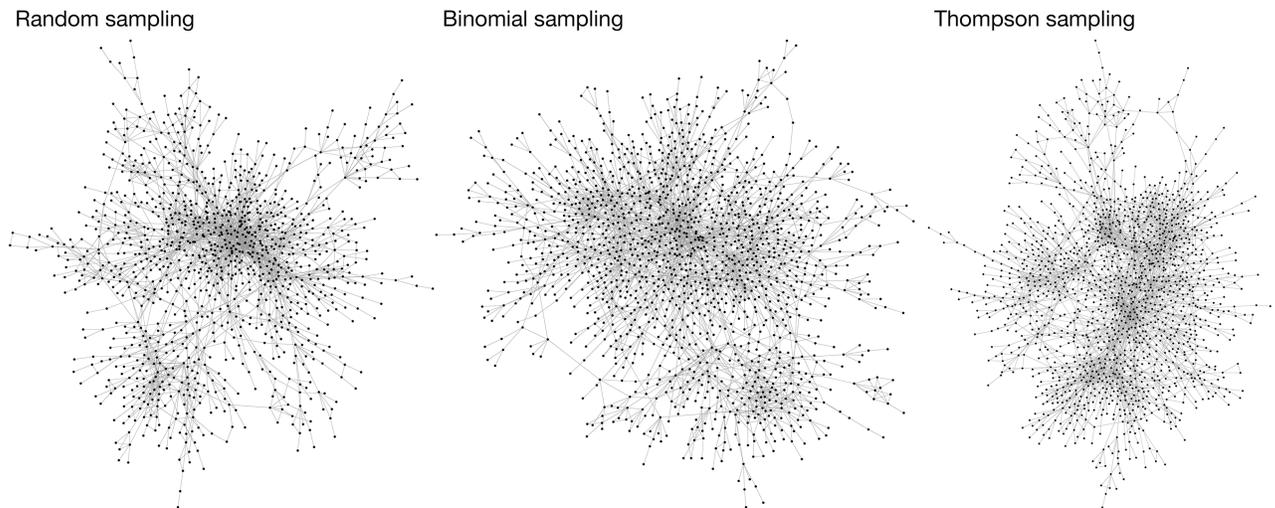
Fig 4. Comparison of question networks for the synonymy proposal task under random sampling, binomial sampling, and φ_N Thompson sampling.

both Binomial and Thompson sampling were able to explore more of the network (discovering more items and questions) than Random sampling with more efficiency (lower mean number of answers $\langle A \rangle$). The explored networks appeared similar by a number of network metrics, although the network generated via Binomial sampling has a lower average degree and higher average shortest path length. Lastly, Binomial and Thompson sampling were comparable to Random sampling in crowd consensus on individual answers, having similar levels of entropy and link bias. Both of these statistics measured how skewed the worker answers were in favor of 'yes, they are synonyms' or 'no, they are not synonyms'.

Taken together, both Binomial and Thompson sampling maintained a comparable level of certainty (measured by consensus or consistency in worker responses) in the network with fewer answers needed on average than Random sampling.

To further understand the answer density of the different sampling methods, we computed the distribution of the number of answers N_{ij} to question (i, j) in Fig 5. Here Random sampling clearly separated from the other two sampling strategies, and Random sampling ended with

Table 1. Summary statistics for the three arms of Experiment 2. Both Binomial and Thompson sampling are more efficient than Random sampling (lower $\langle A \rangle$) without losing the crowd's average consensus on answers, measured by $\langle S \rangle$ and $\langle d \rangle$.

	Random	Binomial	Thompson
N(items)	1134	1537	1509
N(questions)	2413	2887	3020
N(responses)	5043	4993	5034
$\langle A \rangle$	2.090	1.729	1.667
Average degree, $\langle k \rangle$	4.256	3.757	4.003
Clustering coefficient, $\langle CC \rangle$	0.265	0.220	0.243
Eccentricity, $\langle e \rangle$	10.304	11.247	11.840
Shortest path length, $\langle \ell \rangle$	5.732	6.346	5.982
Entropy, $\langle S \rangle$	0.560	0.553	0.551
Link bias, $\langle d \rangle$	0.361	0.371	0.390
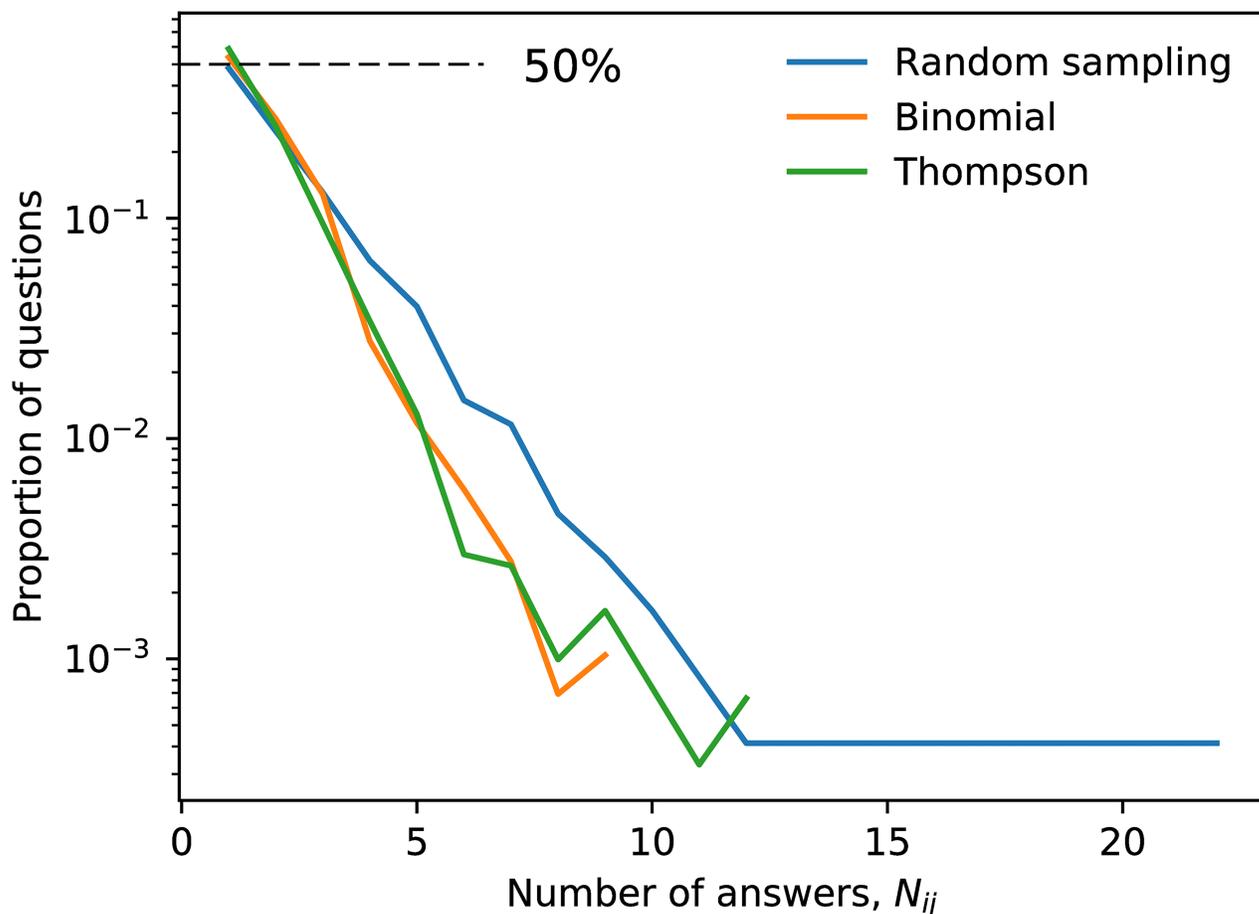
Fig 5. The distributions of the total number of answers per question at the end of crowdsourcing, for each arm of Experiment 2. The efficiency of Binomial and Thompson sampling compared with Random sampling is clear. In all arms, approximately 50% of proposed questions are answered only once.

more questions with more answers than the other sampling strategies. We also note that all three arms finished with many questions with few answers: approximately 50% of questions at the end of the experiment had a single answer. We discuss this further in Sec. 5.

Next, we examined the synonym "quality" of the SPTs, using the synonymy measures introduced in Sec. 3.3. We limited these calculations to proposed word pairs examined by at least three crowd workers to ensure sufficient answers from the crowd. Fig 6(a) shows the proportion of word pairs that share at least one WordNet lemma: Both Binomial and Thompson sampling have slightly higher proportions than Random sampling, at over 12% compared with approximately 11%. This indicates that quality was not lost when using a more efficient sampling strategy. Of course, 11–12% of word pairs sharing a lemma seems low, but recall that shared lemmas is a very strict measure that is likely to miss many synonymous word pairs and so we do not conclude that the majority of the crowd answers are "wrong." Furthermore, to better understand the shared lemma proportion, we constructed a randomized control by shuffling the word pairs (preserving the total frequencies of individual words) and re-measured the proportion of shared lemmas. We found a significant drop in the proportion to approximately 1% (the error bars on these proportions are shown in Fig 6(a) but are quite small).
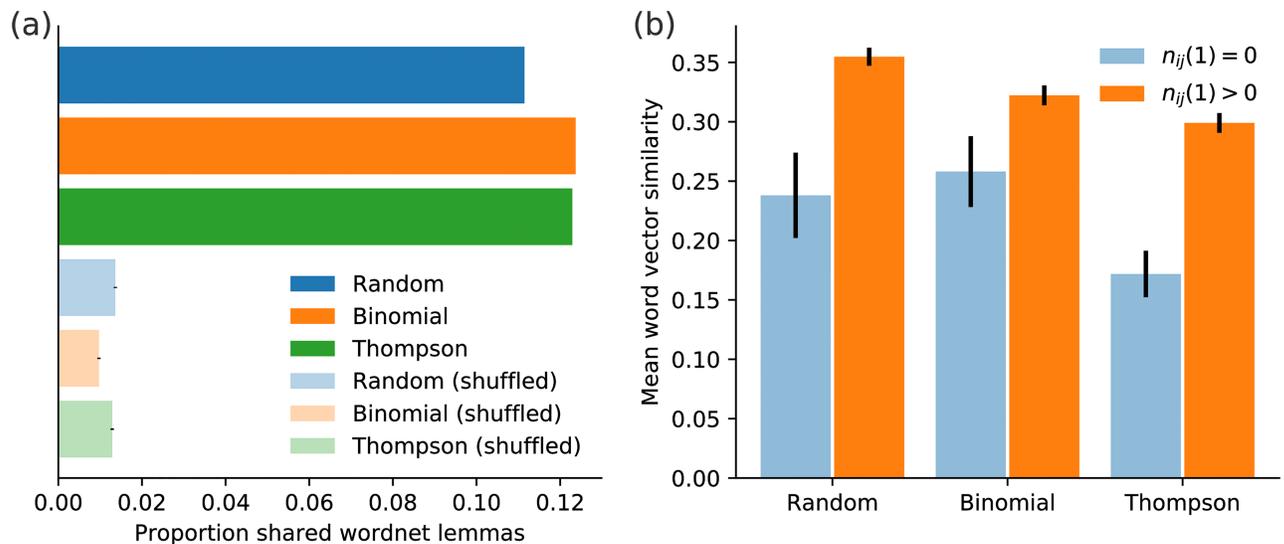
Fig 6. Measures of synonymy for Experiment 2's crowdsourced word pairs. Synonymy for proposed word pairs was estimated using (a) shared WordNet lemmas, (b) cosine similarity between word2vec word embedding vectors (see Sec. 3.3). Approximately 11–12% of crowdsourced word pairs share one or more WordNet lemmas (a strict measure of synonymy), and Binomial and Thompson sampling achieved slightly higher rates than Random sampling. As a control, the word pairs proposed by the crowd were randomized, and the proportion of word pairs with shared lemmas dropped significantly. Meanwhile, regardless of sampling algorithm, the word pairs proposed by the crowd also had significantly higher word vector similarity when at least one member of the crowd agreed that the pair were synonymous ($n_{ij}(1) > 0$), as opposed to no members agreeing the pair were synonymous ($n_{ij}(1) = 0$). This further underscores the estimated quality of the proposed questions and answers and that Binomial and Thompson sampling methods do not appear to trade off quality for efficiency. (To avoid ambiguous answers, we considered word pairs that received at least three answers from workers in these calculations, and panel (a) considers those word pairs with $n_{ij}(1) > 0$.)

https://doi.org/10.1371/journal.pone.0182662.g006

Likewise, Fig 6(b) shows word vector similarities for the three sampling methods, decomposed into word pairs where at least one worker agreed they were synonyms versus no workers agreeing they were synonyms. The crowd-proposed word pairs flagged as synonyms had similarities significantly higher than those not flagged as synonyms. There is a small drop in vector similarity for Binomial and Thompson sampling compared with Random sampling, likely balancing out the small increase in WordNet shared lemma proportion shown in Fig 6(a). We conclude that overall there is no loss in quality, at least as indicated by these measures, when using more efficient sampling algorithms.

Taken together, while we only have one crowdsourcing realization for each arm, it is reasonable to conclude from Experiment 2 that both Binomial sampling and Thompson sampling achieved much higher rates of exploration (more items) and greater efficiency (fewer answers per question) than Random sampling without losing confidence or accuracy in question responses.

5 Discussion

We studied the problem of efficient assignment of crowdsourcing tasks to workers when those workers are also able to propose tasks themselves. Using workers to contribute new tasks and not merely perform predetermined tasks helps unlock the true potential of crowdsourcing. We formulated a growing question network model for this problem, prove theoretical properties of this system, and developed and validated sampling algorithms that can guide workers to grow the network efficiently, while only sacrificing at most minimal confidence in their responses.

Modeling the evolution of the uncontrolled question network teaches us how to better design crowdsourcing policies. For example, by monitoring the innovation rate (ρ) and exploration rate (η) of the growing question network, a crowdsourcer may be able to better and more efficiently control the question network as it grows. At the same time, the rich-get-richer growth of items (older items are attached to a larger fraction of questions), implies that crowdsourcers should pay special attention to the newest items entering the network, to balance out the inherent bias in favor of older items.

Thompson sampling is fast, easy to implement, and flexible enough to capture the preferences of different crowdsourcers, but it is only one potential policy for question selection. More rigorous question selection techniques can be implemented which may outperform the proposed techniques, but with potentially more restrictions. The Thompson sampling algorithms proposed here work for both question nets but also non-network question sets, and can naturally accommodate both growing and static questions sets and nets. Further, statistical inference of question parameters and worker features [15], based on extensions of the null model analyzed in Sec. 2.3, can be used by the crowdsourcer to better pair workers with questions.

There remains considerable room for improvement. For example, in Fig 5, approximately 50% of questions in Experiment 2 received a single answer, regardless of arm. This means that even with the current algorithms the crowd is still supplying an inordinate amount of questions that are being left mostly unconsidered. Of course, some of this may be unavoidable; if there is too much Supply, then the crowd will invariably fall behind. This is further compounded by the inherent bias in favor of older questions. Thompson and Binomial sampling helped curtail this "first-mover-advantage" bias in the growing network but did not necessarily eliminate it. This is the fundamental challenge (and appeal) of this crowdsourcing problem, and more work focused on these issues is needed.

In the future, we will address more detailed schemes for question selection. Questions that contain more than a binary (true/false) response should be further investigated, although the only adaptation of the Thompson sampling algorithm is in the choice of metric to Thompson sample from. Different network structures may arise for different crowdsourcing problems, and assessing the accuracy of the network inferred by the crowdsourcing, and not necessarily the accuracy of individual links, will also be investigated. These and many other interesting and important questions remain in the new problem of crowdsourcing with growing question nets and sets.

Acknowledgments

Author Contributions

Conceptualization: James P. Bagrow.

Data curation: Elizaveta A. Guseva, James P. Bagrow.

Formal analysis: Thomas C. McAndrew, James P. Bagrow.

Funding acquisition: James P. Bagrow.

Investigation: Thomas C. McAndrew, Elizaveta A. Guseva, James P. Bagrow.

Methodology: Thomas C. McAndrew, James P. Bagrow.

Project administration: James P. Bagrow.

Resources: James P. Bagrow.

Supervision: James P. Bagrow.

Validation: Thomas C. McAndrew, Elizaveta A. Guseva, James P. Bagrow.

Visualization: Thomas C. McAndrew, Elizaveta A. Guseva, James P. Bagrow.

Writing – original draft: James P. Bagrow.

Writing – review & editing: Thomas C. McAndrew, James P. Bagrow.

References

1. Howe J. The rise of crowdsourcing. Wired magazine. 2006; 14(6):1–4.

2. Kittur A, Chi EH, Suh B. Crowdsourcing user studies with Mechanical Turk. In: Proceedings of the SIG-CHI conference on human factors in computing systems. ACM; 2008. p. 453–456.

3. Brabham DC. Crowdsourcing as a model for problem solving an introduction and cases. Convergence: the international journal of research into new media technologies. 2008; 14(1):75–90. https://doi.org/10.1177/1354856507084420

4. Kamar E, Hacker S, Horvitz E. 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; 2012. p. 467–474.

5. MacLean DL, Heer J. Identifying medical terms in patient-authored text: a crowdsourcing-based approach. Journal of the American Medical Informatics Association. 2013; 20(6):1120–1127. https://doi.org/10.1136/amiajnl-2012-001110 PMID: 23645553

6. Holley R. Crowdsourcing: how and why should libraries do it? D-Lib Magazine. 2010; 16(3):4.

7. Karnin ED, Walach E, Drory T. Crowdsourcing in the document processing practice. Springer; 2010.

8. Von Ahn L, Maurer B, McMillen C, Abraham D, Blum M. reCAPTCHA: Human-based character recognition via web security measures. Science. 2008; 321(5895):1465–1468. https://doi.org/10.1126/science.1160379 PMID: 18703711

9. Pickard G, Pan W, Rahwan I, Cebrian M, Crane R, Madan A, et al. Time-critical social mobilization. Science. 2011; 334(6055):509–512. https://doi.org/10.1126/science.1205869 PMID: 22034432

10. Tang JC, Cebrian M, Giacobe NA, Kim HW, Kim T, Wickert DB. Reflecting on the DARPA red balloon challenge. Communications of the ACM. 2011; 54(4):78–85. https://doi.org/10.1145/1924421.1924441

11. Naroditskiy V, Rahwan I, Cebrian M, Jennings NR. Verification in Referral-Based Crowdsourcing. PLOS ONE. 2012; 7(10):1–7. https://doi.org/10.1371/journal.pone.0045924

12. Li Q, Ma F, Gao J, Su L, Quinn CJ. Crowdsourcing High Quality Labels with a Tight Budget. In: Proceedings of the Ninth ACM International Conference on Web Search and Data Mining. ACM; 2016. p. 237–246.

13. Karger DR, Oh S, Shah D. Budget-optimal task allocation for reliable crowdsourcing systems. Operations Research. 2014; 62(1):1–24. https://doi.org/10.1287/opre.2013.1235

14. Tran-Thanh L, Venanzi M, Rogers A, Jennings NR. Efficient budget allocation with accuracy guarantees for crowdsourcing classification tasks. In: Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems. International Foundation for Autonomous Agents and Multiagent Systems; 2013. p. 901–908.

15. Tran-Thanh L, Stein S, Rogers A, Jennings NR. Efficient crowdsourcing of unknown experts using multi-armed bandits. In: European Conference on Artificial Intelligence; 2012. p. 768–773.

16. Ipeirotis PG, Gabrilovich E. Quizz: targeted crowdsourcing with a billion (potential) users. In: Proceedings of the 23rd international conference on World wide web. ACM; 2014. p. 143–154.

17. Puterman ML. Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons; 2014.

18. Chapelle O, Li L. An empirical evaluation of Thompson sampling. In: Advances in neural information processing systems; 2011. p. 2249–2257.

19. Donmez P, Carbonell JG, Schneider J. Efficiently learning the accuracy of labeling sources for selective sampling. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM; 2009. p. 259–268.

20. Hung NQV, Tam NT, Tran LN, Aberer K. An evaluation of aggregation techniques in crowdsourcing. In: Web Information Systems Engineering—WISE 2013. Springer; 2013. p. 1–15.

21. Khattak FK, Salleb-Aouissi A. Quality control of crowd labeling through expert evaluation. In: Proceedings of the NIPS 2nd Workshop on Computational Social Science and the Wisdom of Crowds; 2011.

22. Kazai G, Kamps J, Milic-Frayling N. Worker types and personality traits in crowdsourcing relevance labels. In: Proceedings of the 20th ACM international conference on Information and knowledge management. ACM; 2011. p. 1941–1944.

23. Whitehill J, Wu Tf, Bergsma J, Movellan JR, Ruvolo PL. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In: Advances in neural information processing systems; 2009. p. 2035–2043.

24. Ross J, Irani L, Silberman M, Zaldivar A, Tomlinson B. Who are the crowdworkers?: shifting demographics in Mechanical Turk. In: CHI'10 extended abstracts on Human factors in computing systems. ACM; 2010. p. 2863–2872.

25. Ipeirotis PG, Provost F, Wang J. Quality management on Amazon Mechanical Turk. In: Proceedings of the ACM SIGKDD workshop on human computation. ACM; 2010. p. 64–67.

26. Rajan V, Bhattacharya S, Celis LE, Chander D, Dasgupta K, Karanam S. Crowdcontrol: An online learning approach for optimal task scheduling in a dynamic crowd platform. In: Proceedings of ICML Workshop: Machine Learning Meets Crowdsourcing; 2013.

27. Abraham I, Alonso O, Kandylas V, Slivkins A. Adaptive Crowdsourcing Algorithms for the Bandit Survey Problem. In: COLT; 2013. p. 882–910.

28. Bongard JC, Hines PD, Conger D, Hurd P, Lu Z. Crowdsourcing predictors of behavioral outcomes. IEEE Transactions on Systems, Man, and Cybernetics: Systems. 2013; 43(1):176–185. https://doi.org/10.1109/TSMCA.2012.2195168

29. Bevelander KE, Kaipainen K, Swain R, Dohle S, Bongard JC, Hines PD, et al. Crowdsourcing novel childhood predictors of adult obesity. PloS one. 2014; 9(2):e87756. https://doi.org/10.1371/journal.pone.0087756 PMID: 24505310

30. Salganik MJ, Levy KEC. Wiki Surveys: Open and Quantifiable Social Data Collection. PLOS ONE. 2015; 10(5):1–17. https://doi.org/10.1371/journal.pone.0123483

31. Erdős P, Rényi A. On the strength of connectedness of a random graph. Acta Mathematica Hungarica. 1961; 12(1–2):261–267.

32. Erdős P, Rényi A. On random graphs I. Publ Math Debrecen. 1959; 6:290–297.

33. Barabási AL, Albert R. Emergence of scaling in random networks. Science. 1999; 286(5439):509–512. https://doi.org/10.1126/science.286.5439.509 PMID: 10521342

34. Albert R, Barabási AL. Statistical mechanics of complex networks. Reviews of modern physics. 2002; 74(1):47. https://doi.org/10.1103/RevModPhys.74.47

35. Watts DJ, Strogatz SH. Collective dynamics of 'small-world' networks. Nature. 1998; 393(6684):440–442. https://doi.org/10.1038/30918 PMID: 9623998

36. Strogatz SH. Exploring complex networks. Nature. 2001; 410(6825):268–276. https://doi.org/10.1038/35065725 PMID: 11258382

37. Newman MEJ. The structure and function of complex networks. SIAM review. 2003; 45(2):167–256. https://doi.org/10.1137/S003614450342480

38. Newman MEJ, Girvan M. Finding and evaluating community structure in networks. Physical review E. 2004; 69(2):026113. https://doi.org/10.1103/PhysRevE.69.026113

39. Barabási AL, Albert R, Jeong H. Mean-field theory for scale-free random networks. Physica A: Statistical Mechanics and its Applications. 1999; 272(1):173–187.

40. Bagrow JP, Sun J, ben-Avraham D. Phase transition in the rich-get-richer mechanism due to finite-size effects. Journal of Physics A: Mathematical and Theoretical. 2008; 41(18):185001. https://doi.org/10.1088/1751-8113/41/18/185001

41. Newman MEJ. Networks: an introduction. Oxford university press; 2010.

42. Buhrmester M, Kwang T, Gosling SD. Amazon's Mechanical Turk a new source of inexpensive, yet high-quality, data? Perspectives on psychological science. 2011; 6(1):3–5. https://doi.org/10.1177/1745691610393980 PMID: 26162106

43. Miller GA. WordNet: a lexical database for English. Communications of the ACM. 1995; 38(11):39–41. https://doi.org/10.1145/219717.219748

44. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J. Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems; 2013. p. 3111–3119.

45. Harris ZS. Distributional structure. Word. 1954; 10(2–3):146–162. https://doi.org/10.1080/00437956.1954.11659520