Proceedings of the First AAAI Conference on Human Computation and Crowdsourcing

Crowdsourcing Multi-Label Classification for Taxonomy Creation

Jonathan Bragg **Mausam** **Daniel S. Weld**
Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195
{*jbragg, mausam, weld*}*@cs.washington.edu*

Abstract

Recent work has introduced CASCADE, an algorithm for creating a globally-consistent taxonomy by crowdsourcing microwork from many individuals, each of whom may see only a tiny fraction of the data (Chilton et al. 2013). While CASCADE needs only unskilled labor and produces taxonomies whose quality approaches that of human experts, it uses significantly more labor than experts. This paper presents DELUGE, an improved workflow that produces taxonomies with comparable quality using significantly less crowd labor. Specifically, our method for crowdsourcing multi-label classification optimizes CASCADE's most costly step (categorization) using less than 10% of the labor required by the original approach. DELUGE's savings come from the use of decision theory and machine learning, which allow it to pose microtasks that aim to maximize information gain.

Introduction

The presence of large amounts of data is both a boon and bane of modern times. Large datasets enable many novel applications, for example, supervised machine learning and data mining. On the other hand, organizing these datasets for easy access and better understanding requires significant human effort. One such organization practice involves constructing a taxonomy, a hierarchy of categories where each edge denotes an isA relationship (*e.g.*, president isA person). Instances of each category (*items*) are associated with the corresponding node in the taxonomy, for example, the entity Barack Obama is an instance of the president category. WordNet (Miller 1991) and the Linnaean taxonomy are influential examples.

Taxonomizing large datasets and maintaining a taxonomy over time raise significant challenges, since they are a drain on the ontologist(s) responsible for these tasks. A promising answer to this challenge was recently proposed: a distributed crowdsourcing workflow, called CASCADE (Chilton et al. 2013). CASCADE provides a sequence of steps for generating a taxonomy from scratch and for taxonomizing a new item by posing simple questions to unskilled workers on a labor market, such as Amazon Mechanical Turk. Unfortunately, the CASCADE workflow was not optimized. While the overall cost of a CASCADE-produced taxonomy is comparable to one produced by an expert, CASCADE requires about six times as much labor. This suggests that one might be able to refine the workflow, making taxonomy creation both inexpensive and low latency.

Toward this end, we propose DELUGE, a decision-theoretic refinement of CASCADE. DELUGE adopts the high-level skeleton of CASCADE's workflow, but optimizes its most expensive step: assignment of category labels to data items. This step is an example of a multi-label classification problem, an important class of problems which has not previously been optimized in a crowdsourcing setting. Where CASCADE generates a large number of human tasks for each item-label pair, DELUGE saves by ordering the tasks intelligently using a learned model of label and co-occurrence probabilities.

In summary, our paper presents the following primary contributions:

- We present an efficient solution to the novel problem of crowdsourcing multi-label classification. We describe several alternative methods, culminating in a decision-theoretic approach with two components: (1) a probabilistic model that estimates the true value of item-label relationships by allowing workers to be probabilistically accurate, and (2) a controller that chooses, for each item, which questions provide the maximum value of information toward a joint categorization.
- We provide theoretical guarantees for the optimality of our control strategy, as well as an efficient method for selecting batches of labels that makes our approach immediately usable in an online labor market environment.
- We conduct live experiments on Mechanical Turk showing that our best combination of policies requires less than 10% of the labor used by CASCADE for categorization.

Beyond reducing the cost of crowdsourcing multi-label classification and taxonomy creation, our work on DELUGE shows that artificial intelligence and decision-theoretic techniques can be applied to more complex workflows than the simple consensus task-based workflows (Dai et al. 2013; Kamar, Hacker, and Horvitz 2012; Wauthier and Jordan 2011) previously tackled.

```
Procedure BuildTaxonomy (Items):
   ItemLabelMatrix := []
   While TaxonomyNeedsImproving?(ItemLabelMatrix) Do
      Labels := ElucidateLabels(the subset of Items with no label)
      For each Item in Items Do
         ItemLabelMatrix := Categorize(Item, Labels, ItemLabelMatrix)
      Taxonomy = GlobalStructureInference(ItemLabelMatrix)
   Return Taxonomy
```

Figure 1: The general taxonomy creation algorithm. CASCADE and DELUGE differ in their termination conditions and their implementations of `ElucidateLabels` and `Categorize`.

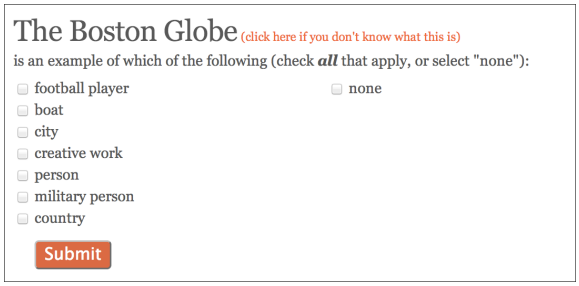


Figure 2: Sample interface for the `Categorize` worker task primitive used in our Mechanical Turk experiment.

Basic Taxonomy Algorithm

Both CASCADE and our refinement, DELUGE, take as input a set of *items* to be categorized, such as photographs or text snippets. Their output is a tree whose interior nodes are each labeled with a text string *label* (category); see Figure 5 for an example.

Our taxonomy creation algorithms use a mixture of algorithmic steps and three task schemata, which are submitted to human workers in the labor market. From a functional perspective, these tasks may be defined as follows:

- `Generate` (t items) $\rightarrow$ t labels:
  Displays t items and asks a worker to suggest a label for each item.
- `SelectBest` (1 item, c labels) $\rightarrow$ 1 label:
  Presents a worker with a single item and c different labels and asks her to pick the best one.
- `Categorize` (1 item, s labels) $\rightarrow$ bit vector of size s:
  Shows a worker a single item and s labels and asks him to indicate which labels apply to the item. (See Figure 2 for our interface corresponding to an instance of this task.)

Since humans are the bottleneck in this approach to taxonomy creation, we seek to minimize the number of tasks requested from the labor market. At the highest level, both CASCADE and DELUGE start by using `Generate` tasks to brainstorm a set of candidate category labels. They then use `SelectBest` tasks to filter out poor labels. Afterwards, `Categorize` tasks identify appropriate labels for all items.

A final, purely algorithmic step, called *global structure inference*, builds a hierarchy from this data by inducing a parent-child relationship between two labels when most of the items in one label are also in the other. Labels with too few items are eliminated, and labels with too great an overlap are merged. In this paper, we make no changes to CASCADE's approach to this final step; see (Chilton et al. 2013) for details.

Figure 1 summarizes this high-level algorithm, but does not specify exactly how the set of category labels should be elucidated, nor does it state how to categorize each item efficiently using a fixed set of labels. We discuss these issues in the next two subsections. As we shall see, CASCADE takes a relatively simple approach to these questions, but more sophisticated techniques can greatly decrease the amount of human labor required.

Elucidating Category Labels

Noting that there would likely be wasteful duplication if one asked humans (via a `Generate` task) to brainstorm candidate labels for *every* one of the items, CASCADE's implementation of `ElucidateLabels` starts by considering only the first few ($m = 32$) items, termed the *initial item set*. CASCADE partitions this initial item set into groups of $t = 8$ and creates a `Generate` task for each, which is sent to $k = 5$ workers. After all $\lceil km/t \rceil$ tasks are completed, CASCADE is left with km candidate labels, not necessarily distinct.

CASCADE's next step is to prune the candidate labels. At this point, each of the m initial items will have up to k distinct suggested labels. For each item, CASCADE submits k `SelectBest` tasks requesting a human to choose which of the labels seems most appropriate. Any labels with two or more votes are retained; after this step, $p \leq 2m$ distinct labels remain (assuming $k = 5$).

In the next section, we use a combinatorial balls-and-urns model to describe an alternative, decision-theoretic method for controlling label elucidation.

Categorizing Items Once Labels are Known

Once labels have been elucidated, CASCADE enters its most costly phase, which incurs $O(np)$ worker tasks, where $n = |Items|$ and $p = |Labels|$. Intuitively, the idea is to iterate through the items and labels, asking k different workers whether a label applies to an item. Chilton *et al.* observed that workers sometimes lack the context to make these decisions, so they proposed categorizing in two sequential phases, which they term *adaptive context filtering*. The first phase iterates through items and labels as described above; every label which receives at least two (out of five) votes progresses to the next phase. In the second phase, workers are only shown labels which made the first round cut, and a label is considered to fit an item if at least four of the five

workers deem it so. Thus, both phases together use between $\lceil knp/s \rceil$ and $2\lceil knp/s \rceil$ worker tasks.

In two sections, we present several improved algorithms for this categorization process, which we have noted is a multi-label classification problem. The first approach generates precisely the same labeling with strictly fewer worker tasks. The second uses substantially fewer workers, with little or no loss in classification accuracy. The final approaches incrementally build probabilistic models of label occurrence and co-occurrence, which they use to optimize the order in which they pose questions to workers.

Pólya's Urn for Label Elucidation

CASCADE's label elucidation step asks workers to brainstorm relevant labels to be added to the taxonomy. CASCADE performs this step on a set of m items, where $m \ll n$, the total number of items to be categorized. The key insight behind elucidating labels for a small number of items is that labels generated for a random subset of items can be globally relevant, and that workers are likely to repeat labels across items. An important control question for optimizing this step involves the choice of m. CASCADE sets $m = 32$ in an ad hoc manner, but ideally we would like to estimate the quality of a set of category labels as it grows in order to determine when elucidating more labels would likely be wasteful.

DELUGE proposes modeling the brainstorming process using a Pólya urn model (Johnson and Kotz 1977), also known as a Chinese Restaurant Process. A very general framework, Pólya urn models are particularly suited for modeling discrete, multi-label distributions where the number of labels is unknown a priori, as is the case for our category labels. The metaphor for this generative model is that of an urn containing colored balls, where colors correspond to labels. In each iteration, a ball is drawn uniformly from the urn and then placed back in the urn along with a new ball. If the drawn ball is black (a specially-designated color), the new ball is a previously unseen color; otherwise, the new ball is the same color as the drawn ball.

As balls are drawn from the urn, the number of colors in the urn increases but the probability of obtaining a new color decreases. Moreover, colors that are drawn frequently have a higher probability of being drawn than other colors. This behavior can be seen from the probabilities that govern draws from the urn. Suppose that there are n_c balls of color c, N non-black balls, and α black balls. Then, the probability of drawing a ball of color c is $n_c/(N+\alpha)$ and the probability of drawing a previously unseen color is $\alpha/(N+\alpha)$. A Pólya urn model is parameterized by α; larger values of α imply higher probability of brainstorming new category labels.

A useful quantity to estimate for determining a stopping condition is the expected number of new labels that would be generated by a fixed number of future worker tasks.

Theorem 1 *Let our Pólya urn contain N colored balls and α black balls. Let the random variable X_d be the number of new colors present in the urn after d future draws. Then,*

$$E[X_d] = \sum_{i=0}^{d-1} \frac{\alpha}{N+\alpha+i}.$$

Recall that CASCADE asks k workers to brainstorm labels for each item. Thus, if we have generated labels for m items, with $n - m = r$ items remaining, we have $N = km$ and $d = kr$. Terminating the label elucidation phase at this point will result in an expected $\sum_{i=0}^{kr-1} \frac{\alpha}{km+\alpha+i}$ missed labels. The expected fractional increase in the total number of labels is this quantity divided by the number of distinct labels seen after the first m items.

Our model provides a principled stopping condition for this phase: terminate when the expected fractional increase in the number of labels is below a desired threshold. In order to operationalize this policy, we compute the maximum-likelihood estimate of α using gradient ascent on the log-likelihood of generating the observed data.

Note that this model assumes that all labels are independent and that workers are equally likely to generate new labels for any particular item. These assumptions are inaccurate due to the underlying label co-occurrence probabilities, as well as potential differences in the number of accessible labels for each item. However, the approximations are reasonable for the `Generate` phase, since we will likely not have enough data to learn parameters for a more complex model in any case. Our model lets us estimate the approximate impact of stopping, which can be used to identify an appropriate termination point for this phase.

Improved Categorization Control Algorithms

CASCADE, like many crowdsourcing workflows, implements voting on binary outcomes by requesting a fixed number of votes k and setting a threshold number of votes T (majority voting is the special case where $T = k/2$). Once the requested number of votes is returned, this procedure returns a positive outcome if and only if the number of positive votes is at least T. The amount of work required by this procedure can be quite large, especially when attempting to scale workflows. In the Adaptive Context Filtering step of its workflow, CASCADE asks k workers to vote on each item and label combination. Supposing there are n items and p labels, this step requires asking for $O(knp)$ votes.

A Lossless Improvement to Threshold Voting

The first observation we make is that given a threshold number of votes T, asking for all k votes is often unnecessary. Once one has received T positive votes, or $k - T + 1$ negative votes, one need not ask for further votes since the answer using k total votes is fully determined to be positive in the former case and negative in the latter case. We call this stopping condition *lossless stopping*; it can be seen as a generalization of the "Ask two people to vote and only ask a third if the first two disagree" policy in TurKit (Little et al. 2009).

One-away Approximation for Threshold Voting

One can further reduce the number of votes required with a simple heuristic method, which we call the *one-away heuristic*, that we hypothesize will result in only a small amount of error compared to the original threshold voting method. (Note that lossless stopping results in no error, compared to

the original.) The one-away heuristic returns true early if we observe $\max\{T-1, 0\}$ positive votes and no negative votes, or returns false early if we observe $\max\{k-T, 1\}$ negative votes and no positive votes. The intuition behind this heuristic is that although the lossless stopping condition may not have been met, we have observed strong evidence for returning an outcome and no evidence in support of the alternative outcome.

A Simple Probabilistic Model

A more powerful way to approach the problem is from the Bayesian perspective. Suppose we have already labeled a large number of items, $I \in \mathcal{I}$, and hence know for each I if label L holds, denoted $\oplus(I, L) = 1$, or does not, denoted $\oplus(I, L) = 0$. Now, when given a new item I' we know nothing about, we can use the previously observed data to calculate the maximum likelihood prior probability of any label $P(\oplus(I', L)) = \sum_{I \in \mathcal{I}} \oplus(I, L)/|\mathcal{I}|$.

In order to update our posterior for $\oplus(I', L)$ after observing a worker's vote, we must model noisy workers. Our worker model uses two parameters to represent how accurately workers are able to detect true positives and true negatives. As in (Raykar et al. 2010), we term these parameters worker *sensitivity* and *specificity*, respectively. We observe that worker specificity is much higher than worker sensitivity due to the sparsity of labels in our dataset, and that representing worker accuracy with two parameters instead of a single shared parameter greatly improves the discriminative ability of our probabilistic models.

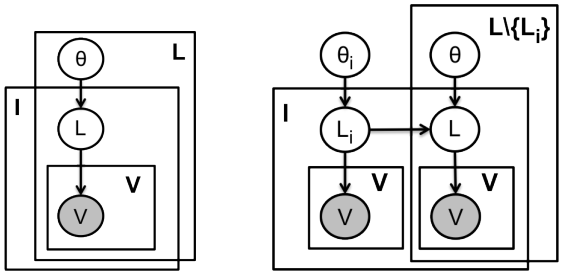
If a worker with sensitivity p_{tp} and specificity p_{tn} answers that a label holds, we can update our posterior by simply multiplying the prior by the likelihood ratio $(p_{tp} + (1 - p_{tn})) \,/\, ((1 - p_{tp}) + p_{tn})$. In this model, the agent always knows the most probable value for $\oplus(I, L)$, and if a utility model associates different costs for false positive and false negative classifications, it can easily trade off between these errors.

We term this baseline probabilistic model the *independent* model, since it naively assumes that labels are independent, as shown in the graphical model in Figure 3a. If we denote the set of labels by $\mathcal{L}$, the independent model has a total of $|\mathcal{L}| + 2$ parameters: one for each label corresponding to the prior probability of that label, and two for a noisy worker model that we assume here is shared among all workers. The marginal label probabilities are

$$P(L \mid \mathbf{v}) \propto P(L)P(\mathbf{v}_L \mid L),$$

where $L \in \mathcal{L}$ is a Boolean random variable corresponding to an outcome $\oplus(I, L)$ for an item and $\mathbf{v}_L \subseteq \mathbf{v}$ is the vector of observed votes associated with that outcome.

One subtlety concerns the treatment of past data. Since the agent has no access to gold data, it does not know the true labels, $\oplus(I, L)$, even when it has seen the assessments of many workers. We use expectation maximization (EM) to estimate the values of these latent labels together with the parameters of our model. We perform a Bayesian estimate of our parameters to avoid problems when categorization is just starting, by assuming weak symmetric Beta priors and computing a maximum a posteriori estimate.



(a) Independent (b) MLNB

Figure 3: Generative probabilistic models for multi-label classification. The I, L, and V plates correspond to items, labels, and votes, respectively. The MLNB model in (b) is the model for predicting label L_i; there are $|\mathcal{L}|$ such models.

Modeling Label Co-occurrence

The assumption of label independence made by the previous model is a substantial approximation. For example, an item which has been categorized as "person" is more likely to be a member of the actor category than to be a member of the location category. This observation is especially pertinent to taxonomies with deep hierarchical structure, but is true for any set of overlapping labels.

It is natural, therefore, to learn a joint model of label probabilities. In this model, when a worker responds that an item is in a given category, the posterior for *all* other categories can be updated. This update will also affect the choice of which label has the highest value of information by the control strategy we will define.

There are many ways to represent a complex joint distribution. As a baseline approach, we explore a simple model, which we term the *multi-label naive Bayes* (MLNB) model. For each label in this model, we construct a star graph with directed edges from that label to all other labels; the graphical model in Figure 3b shows the star graph for label L_i. Using notation we defined for the independent model, the marginal label probabilities for the MLNB model are

$$P(L \mid \mathbf{v}) \propto P(L)P(\mathbf{v}_L \mid L) \prod_{L' \in \mathcal{L} \setminus \{L\}} \sum_{L'} P(L' \mid L)P(\mathbf{v}_{L'} \mid L').$$

Calculating marginal probabilities for all labels requires $O(|\mathcal{L}|^2)$ computations per item and involves summing out the latent label variables represented as child nodes in the graphical model. This approximation models pairwise co-occurrences between labels directly but ignores the higher order interactions.

In order to estimate parameters for the MLNB model in an efficient manner, we reuse parameters and label predictions obtained by running EM for the independent model. We approximate the additional $2(|\mathcal{L}^2| - |\mathcal{L}|)$ conditional label probabilities $P(L' \mid L)$ using these predictions as the expected fraction of items with labels L and L' out of those with label L. Knowing that a small fraction of labels applies to any particular item, we use Laplace smoothing to bias these estimates against positive co-occurrence.

We have also explored a more sophisticated probabilistic graphical model that combines our generative independent

model with a pairwise Markov network over the label variables for an item. We found that this model, which requires approximate inference methods, is too slow to be useful in a live crowdsourcing setting and does not produce significant gains over the MLNB model; we do not describe it further in this paper.

Choosing Which Questions to Ask

One may also consider different control strategies for choosing the next question(s) to dispatch to a worker. CASCADE employed a simple *round-robin* strategy, but we advocate using a *greedy* search that asks about the label(s) where a worker's vote(s) would provide the greatest value of information.

More formally, each time DELUGE asks a worker for new votes, its goal is to select a set of votes that will result in the greatest expected decrease in the uncertainty of our label predictions. Information theory provides us with a useful criterion for measuring the amount of uncertainty in the distribution of label predictions, the joint entropy

$$H(\mathcal{L}) = -\sum_{\mathbf{l} \in \operatorname{dom} \mathcal{L}} P(\mathbf{l}) \log P(\mathbf{l}).$$

The domain of $\mathcal{L}$ consists of all possible assignments to the variables in $\mathcal{L}$, and $\mathbf{l}$ denotes one of those assignments. Let $\mathcal{A} \subset \mathcal{V}$, where $\mathcal{V}$ denotes the unbounded set of possible future votes. The expected uncertainty of the distribution of label predictions after receiving the votes in $\mathcal{A}$ is the conditional entropy

$$H(\mathcal{L} \mid \mathcal{A}) = -\sum_{\substack{\mathbf{l} \in \operatorname{dom} \mathcal{L} \\ \mathbf{a} \in \operatorname{dom} \mathcal{A}}} P(\mathbf{l}, \mathbf{a}) \log P(\mathbf{l} \mid \mathbf{a}).$$

We are interested in maximizing the difference of these two quantities, known as the expected information gain, or the mutual information $I(\mathcal{L}; \mathcal{A}) = H(\mathcal{L}) - H(\mathcal{L} \mid \mathcal{A})$.

Unfortunately, calculating the optimal set, $\mathcal{A}$, that maximizes information gain is intractable due to the combinatorial nature of the problem. However, we are able to select a near-optimal set by exploiting the combinatorial concept of *submodularity*. Nemhauser, Wolsey, and Fisher (1978) show that the greedy algorithm for optimizing a submodular function provides a solution that is guaranteed to be within a constant factor of $(1 - 1/e) \approx 63\%$ of optimal. While information gain is not, in general, submodular, it does satisfy this property under our modeling assumption that workers' errors are not correlated, *i.e.*, that votes in $\mathcal{V}$ are independent given the true values of $\mathcal{L}$ (Krause and Guestrin 2005). Krause and Guestrin provide a greedy algorithm for selecting a near-optimal subset of variables under this assumption, and prove that one cannot achieve a tighter bound unless $\mathbf{P} = \mathbf{NP}$.

This greedy algorithm accumulates a set of future votes $\mathcal{A}$ by adding votes $V \in \mathcal{V}$ one at a time with a greedy heuristic. While we are interested in the set of votes that maximizes the information gain for $\mathcal{L}$, the greedy heuristic selects a vote V by ranking them according to the quantity $H(V \mid \mathcal{A}) - H(V \mid \mathcal{L})$. In general, these conditional entropies require that we represent the full joint distribution over $\mathcal{L}$, which is intractable for even a small number of labels. Fortunately, we can refine this heuristic using an additional conditional independence assumption of our models, which simplifies $H(V \mid \mathcal{L})$ to the local conditional entropy $H(V \mid L_V)$, where $L_V \in \mathcal{L}$ is the label corresponding to vote V.

Theorem 2 *Let each vote $V \in \mathcal{V}$ be independent of all other votes given the label L_V, and let $\mathcal{A}$ be the set of future votes accumulated by the greedy algorithm thus far. Also let V_L denote an arbitrary future vote for some label L. Then, the set $\mathcal{A}$ constructed by successively adding future vote V^* by the strategy*

$$V^* \in \operatorname*{argmax}_{L \in \mathcal{L}} H(V_L \mid \mathcal{A}) - H(V_L \mid L)$$

is within $(1 - 1/e)$ of optimal.

The proof follows from applying Krause and Guestrin's result to our model. Note that when the greedy algorithm selects the first vote, $\mathcal{A}$ is initially empty and thus $H(V \mid \mathcal{A})$ is simply $H(V)$.

This theorem yields a surprising result: selecting a near-optimal single question to ask a worker requires only the local entropies $H(V)$ and $H(V \mid L_V)$. DELUGE leverages this greedy strategy, along with the MLNB model of label co-occurrence, to optimize the categorization process.

Experiments

In our experiments, our goal is to compare the various strategies from the categorization control section. We first compare the simple improvements to threshold voting by analyzing the cost savings for each strategy (lossless, one-away) and threshold setting $T = \{2, 3, 4\}$, along with the quality of the taxonomy produced. Next, we evaluate the probabilistic models on their predictive performance and compare that against the original strategy from CASCADE.

Dataset

In order to better analyze the effect of different categorization algorithms, we controlled for variation in label elucidation policies and adopted a fixed set of candidate category labels from the literature. Specifically, we took a subset of the fine-grained entity tags described in (Ling and Weld 2012) by eliminating low probability tags and all those for organizations, events, and facilities; this process yielded a manageable set of 33 labels. We then over-generated items for each of these labels, and constructed a random subset of 100 items.

Our worker vote collection process involved emulating a run of the `Categorize` procedure from CASCADE, called on these 100 items and 33 categories. We had a total of $k = 15$ workers from Mechanical Turk vote on batches of seven labels per Human Intelligence Task (HIT). The interface for our HITs, shown in Figure 2, uses form validation to ensure that a worker either selects at least one label, or deliberately indicates that none of the displayed labels apply to the item in question. Each HIT cost \$0.04 and the total amount paid to workers was \$300. The purpose of gathering this data was to allow us to compare different control strategies, controlling for worker error, since each control strategy would be seeing the same worker responses.

Method	T	F-score	Votes	% savings
Lossless	2	0.83	130	21
One-away	2	0.82	96	42
Lossless	3	0.84	102	38
One-away	3	0.83	69	58
Lossless	4	0.75	72	56
One-away	4	0.70	38	77

Table 1: Comparison of threshold voting methods. Mean F-score and number of votes per item.

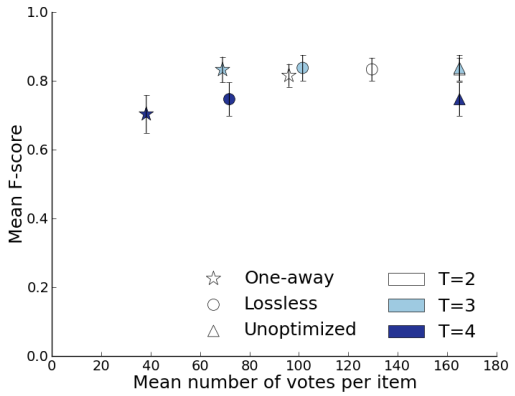


Figure 4: F-score vs. cost for threshold voting improvements

Threshold Approaches

In the first experiment, we compared the threshold voting modifications to the naive version of threshold voting implemented in CASCADE. Since CASCADE uses various threshold settings in the adaptive context filtering, we tested with thresholds of $T = \{2, 3, 4\}$ out of 5 total votes. Table 1 shows the number of votes per item used by lossless stopping and the one-away heuristic, and the fraction of votes saved compared to the original approach in CASCADE, which used $33 \times 5 = 165$ votes per item. Note that lossless stopping, which returns exactly the same predictions as the threshold voting procedure from CASCADE, is able to save up to 56% of the votes when $T = 4$.

In order to better understand the impact of the one-away heuristic on classification performance, in Figure 4 we plot F-score performance vs. the number of votes used for lossless stopping and the one-away heuristic. For threshold values $T = \{2, 3\}$, the one-away strategy significantly lowers the already reduced cost associated with lossless stopping without introducing a statistically significant decrease in F-score. The decrease in F-score for $T = 4$ is statistically significant ($p < 0.01$ using a two-tailed paired t-test), but can be attributed to poor recall. The one-away heuristic at this threshold setting returns false if the first vote is negative, which is suboptimal since worker sensitivity is significantly lower than worker specificity.

In addition to classification performance, we are also interested in how our improvement methods impact the quality of the final output taxonomy. Visual inspection for errors in the output taxonomies did not reveal a decrease in quality when using the one-away heuristic. Figure 5 shows a high-quality taxonomy produced by the one-away heuristic with threshold $T = 3$.

- **person (41)** — Jesus, Kenny G, The Pope, Mel ...
  - **actor (9)** — Madonna, Meg Ryan, Eminem, Ha ...
    - **director (2)** — Mel Gibson, Brad Pitt
  - **religious leader (8)** — The Pope, rabbi, Pope John Pa ...
  - **politician (8)** — Queen Elizabeth, Hillary Clin ...
  - **artist (8)** — Eminem, Madonna, Monet, Georg ...
    - **musician (5)** — Eminem, Kenny G, George Harri ...
  - **football player (6)** — Peyton Manning, Joe Montana, ...
  - **author (4)** — Martha Stewart, John Locke, I ...
  - **organization founder (3)** — Martha Stewart, Steve Jobs, U ...
  - **military person (3)** — Saddam Hussein, Frederick the ...
- **location (15)** — Fiji, Indonesia, Shanghai, Wa ...
  - **country (7)** — Fiji, Greenland, Indonesia, F ...
    - **island (6)** — Fiji, Whidbey Island, Greenla ...
  - **city (4)** — Shanghai, Washington, Florida ...
  - **river (3)** — The Charles River, The Potoma ...
- **vehicle (12)** — Honda Accord, Hummer, MiG-23, ...
  - **car (6)** — Chevy Tahoe, Ford Taurus, Hum ...
  - **truck (3)** — Chevy Silverado, tractor trai ...

Figure 5: The one-away policy with threshold $T = 3$ used only 42% of the labor required by CASCADE, yet produced an excellent taxonomy (excerpt shown).

Inference-based Approaches

We hypothesized that scaling multi-label classification and taxonomy creation to a large number of items requires a probabilistic approach. To empirically determine the effectiveness of our approaches, we compared the performance of various inference and control strategies using the votes gathered from Mechanical Turk.

In our experiments, we tested three inference methods (MLNB, Independent, and Majority) and two control strategies (greedy and round-robin). MLNB and Independent inference methods were described in the previous section, and Majority performs simple majority vote estimation that defaults to a negative answer and breaks ties in favor of a positive answer (we found that these simple modifications improved results for our dataset). The greedy control strategy uses the heuristic from Theorem 2 to select labels that maximize information gain, while the round-robin strategy fills in votes layer by layer (*e.g.*, it asks once about each label before asking twice about any label). Majority with round-robin is our reconstruction of the original CASCADE approach.

In order to test how our models will perform when scaling in the number of items, we evaluate the performance of our models using leave-one-out cross-validation for the 100 items. We estimate model parameters using 99 items and five worker votes for each item-label pair in the training set.

Figure 6 shows the results of this experiment. MLNB and Independent show a clear improvement over the simple round-robin method, and MLNB in particular reaches high levels of performance very quickly. The improvement of MLNB over Independent is highly statistically significant at the 0.05 significance level (using a two-tailed paired t-test) for the first 47 votes, lending credence to our hypothesis that co-occurrence information aids classification. Furthermore, points where Independent crosses slightly above MLNB are not statistically significant. We note that the probabilistic

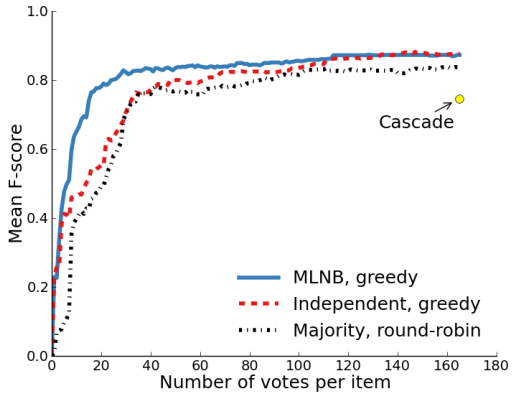


Figure 6: Performance vs. number of votes (Mechanical Turk data). CASCADE does not fall on the Majority line, as it uses a different threshold ($T = 4$).

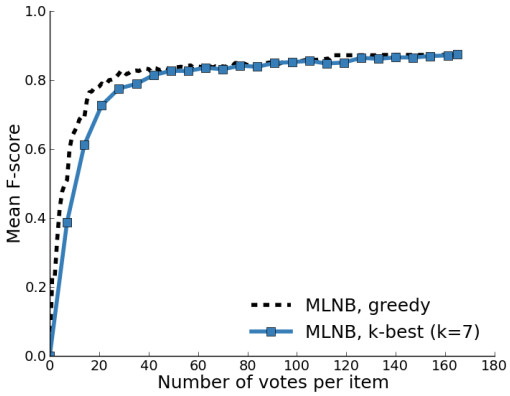


Figure 7: Performance vs. number of votes for selecting batches of $k = 7$ (Mechanical Turk data, MLNB with single label selection shown for comparison).

models sometimes request votes in excess of the votes collected for an item-label pair, in which case we simply use the next-best label; this behavior does not happen frequently enough to impact our statistically significant results.

So, which control strategy is best? On this dataset, CASCADE's voting policy (accept a label if four out of five workers think it applies) required 165 worker tasks per item and yielded an F-score of 75% when compared to gold-truth data. In contrast, our one-away strategy with threshold $T = 3$ had an F-score of 83% and used only 42% as much labor. Our probabilistic approaches are anytime and can be stopped after any number of worker tasks. MLNB with a greedy control strategy produced an F-score around 76% after only 16 tasks per item, which is less than 10% as much labor as CASCADE required to achieve similar performance.

Batching Tasks

In order to be practically useful in a crowdsourcing setting, our control strategies need to be able to group tasks together so that a worker can answer multiple questions about an item at once; see Figure 2 for an example. Theorem 2 provides a method for choosing batches of labels, by accumulating a set of votes using the greedy heuristic. An alternative simple control strategy, which we term k-best, simply selects the top k labels ranked by the greedy heuristic before any votes have been accumulated.

In our experiments, we found that k-best offers the best trade-off between classification performance and computational complexity. Figure 7 shows that for $k = 7$ (the same number used by CASCADE and our own live experiment), MLNB with k-best control results in a small decrease in performance compared to MLNB with single label selection. This difference is statistically significant (at the 0.05 significance level using a two-tailed paired t-test) only until about 35 votes per item, and the batched version of MLNB still outperforms Independent with single label selection.

The accumulative greedy method failed to produce significant performance gains over k-best. Moreover, computing the greedy heuristic for $k > 1$ is computationally intensive, requiring approximation of conditional entropies for the MLNB model. One possible reason the accumulative method fails to improve performance is that labels within a batch must be distinct in our setting (it is not beneficial to ask the same worker the same question more than once). Given this restriction, k-best is an effective heuristic that incurs no additional cost compared to selection of single labels.

Simulation Study

An intelligent control procedure must be robust to noise due to worker quality. In order to assess the behavior of our techniques on more complex classification problems where the workers may be more error-prone, we simulated workers with 60% sensitivity and 80% specificity. We perform this experiment using the gold-truth item-label answers in a purely simulation setting. The overall higher performance of our results in Figure 8 despite less accurate workers (average sensitivity and specificity for workers in our dataset was 76% and 98%, respectively) can be attributed to discrepancies between the gold-truth answers supplied by the authors and the collective decisions made by workers on Mechanical Turk. Figure 8 shows the same statistically significant ordering of the models as we saw with real worker votes, suggesting that our results generalize to a wide array of multi-label classification tasks.

Related Work

Our research fits into the broad theme of using AI techniques for optimization of crowdsourced tasks (Weld, Mausam, and Dai 2011). A large body of work has optimized simple tasks such as classification with noisy workers (Dai et al. 2013; Kamar, Hacker, and Horvitz 2012; Raykar et al. 2010; Sheng, Provost, and Ipeirotis 2008; Wauthier and Jordan 2011; Welinder et al. 2010; Whitehill et al. 2009). Relatively less research has gone into optimizing more complex workflows such as CASCADE's that have a much richer space of possible outcomes. A notable exception is the optimization of iterative improvement workflows using decision-theoretic control (Dai et al. 2013).

Closely related work on optimizing the categorization

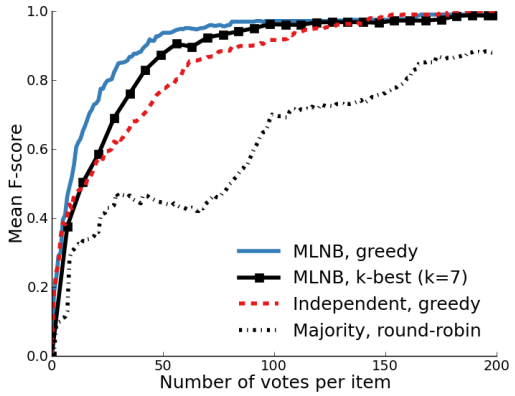


Figure 8: Performance vs. number of votes for a more difficult simulated task (sensitivity = 0.6, specificity = 0.8).

of items within a taxonomy takes a graph-theoretic approach (Parameswaran et al. 2011), but does not consider a probabilistic framework for modeling noisy workers, a critical component of crowdsourcing systems. Moreover, that approach assumes labels are organized in a taxonomy that is known a priori, and does not model label co-occurrence, which our experiments show dramatically improves labeling efficiency and accuracy.

Other related research investigates selecting the next best question from a set of known questions. Often, the goal is to use active learning to improve the accuracy of a classifier, by selecting questions based on label uncertainty or model uncertainty (Sheng, Provost, and Ipeirotis 2008; Wauthier and Jordan 2011). Our approach to multi-label classification seeks to ask questions that optimize the value of information within a graphical model (Krause and Guestrin 2005), rather than to optimize performance on an external task.

Our approach to label elucidation is related to work on collaborative and social tagging. (Golder and Huberman 2006) uses a Pólya urn model to explain why relative tag proportions tend to stabilize over time for bookmarks on the Delicious website. (Chi and Mytkowicz 2008) investigates tagging on Delicious as well, using information-theoretic measures to model the developing vocabulary of tags and the effectiveness of the set of tags for document retrieval. In a crowd labor setting, (Lin, Mausam, and Weld 2012b) uses a Chinese Restaurant Process model to optimize free response question answering.

Other instances of the use of AI within crowdsourcing include assigning tasks to workers (Donmez, Carbonell, and Schneider 2010; Tran-Thanh et al. 2012), solving consensus tasks using worker responses and a machine learning model (Kamar, Hacker, and Horvitz 2012), selecting workers based on skill (Shahaf and Horvitz 2010), and choosing between multiple workflows for the same task (Lin, Mausam, and Weld 2012a).

Conclusions

Machine learning and decision-theoretic techniques offer the potential for dramatically reducing the amount of human labor required in crowdsourced applications. However, to date, most work has focused on optimizing relatively simple workflows, such as consensus task and iterative improvement workflows. Taxonomy generation is an important task, which requires a complex workflow to create a globally consistent interpretation of a large dataset from workers who typically have only a narrow view of a small data subset. Since previous work on crowdsourcing taxonomy creation, CASCADE, was both promising yet labor intensive, it is a natural target for decision-theoretic optimization.

This paper presents DELUGE, a refinement of the CASCADE algorithm with novel approaches to the subproblems of label elucidation and multi-label classification. For the former, we introduce a combinatorial Pólya urn model that allows us to calculate the relative cost of stopping the label generation phase early. For the problem of classifying items with a fixed set of labels, we present four models: lossless, one-away, a simple probabilistic model, and the MLNB model of label co-occurrence. The latter two models support a greedy control strategy that chooses the most informative label to ask a human to evaluate, within a constant factor of the optimal next label. We also provide a batching strategy, making our approach to multi-label classification both highly general and practically useful.

Using a new dataset of fine-grained entities, we performed live experiments on Mechanical Turk to evaluate the relative effectiveness of the approaches to multi-label classification. While CASCADE's voting policy required 165 worker tasks per item, our approaches achieve superior performance using much less labor. In particular, DELUGE uses MLNB with a greedy control strategy to exceed CASCADE's performance after only 16 tasks per item, or less than 10% as much labor.

We envision extending this work in a number of ways. Our probabilistic models do not distinguish between individual workers, since we focus on comparing different representations of the underlying distribution on labels. However, learning individual noisy worker models would likely improve results for these models. Another line of inquiry involves exploration of the design implications of this work. For example, our anytime probabilistic approaches could be used to pose questions with a dynamic interface that updates as a worker provides responses. Finally, we hope that our work inspires other researchers to tackle the design and optimization of workflows for more complex problem domains.

Acknowledgements

We thank Lydia Chilton, Christopher Lin, Shih-Wen Huang, and Ben Taskar for useful discussions. This work was supported by the WRF / TJ Cable Professorship, Office of Naval Research grant N00014-12-1-0211, and National Science Foundation grants IIS 1016713 and IIS 1016465.

References

Chi, E. H., and Mytkowicz, T. 2008. Understanding the efficiency of social tagging systems using information theory. In *Proceedings of the Nineteenth ACM Conference on Hypertext and Hypermedia (HT '08)*, 81–88.

Chilton, L. B.; Little, G.; Edge, D.; Weld, D. S.; and Landay, J. A. 2013. Cascade: Crowdsourcing taxonomy creation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*.

Dai, P.; Lin, C. H.; Mausam; and Weld, D. S. 2013. POMDP-based control of workflows for crowdsourcing. *Artificial Intelligence* 202(0):52–85.

Donmez, P.; Carbonell, J. G.; and Schneider, J. 2010. A probabilistic framework to learn from multiple annotators with time-varying accuracy. In *Proceedings of the SIAM International Conference on Data Mining (SDM '10)*, 826–837.

Golder, S. A., and Huberman, B. A. 2006. Usage patterns of collaborative tagging systems. *Journal of Information Science* 32(2):198–208.

Johnson, N. L., and Kotz, S. 1977. *Urn models and their application: an approach to modern discrete probability theory*. Wiley New York.

Kamar, E.; Hacker, S.; and Horvitz, E. 2012. Combining human and machine intelligence in large-scale crowdsourcing. In *Proceedings of the Eleventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS '12)*.

Krause, A., and Guestrin, C. 2005. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence (UAI '05)*.

Lin, C.; Mausam; and Weld, D. S. 2012a. Dynamically switching between synergistic workflows for crowdsourcing. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI '12)*.

Lin, C. H.; Mausam; and Weld, D. S. 2012b. Crowdsourcing control: Moving beyond multiple choice. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence (UAI '12)*.

Ling, X., and Weld, D. S. 2012. Fine-grained entity resolution. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI '12)*.

Little, G.; Chilton, L. B.; Goldman, M.; and Miller, R. C. 2009. TurKit: Tools for iterative tasks on mechanical turk. In *Human Computation Workshop (HCOMP '09)*.

Miller, G. 1991. WordNet: An on-line lexical database. *International Journal of Lexicography* 3(4):235–312.

Nemhauser, G. L.; Wolsey, L. A.; and Fisher, M. L. 1978. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming* 14(1):265–294.

Parameswaran, A. G.; Sarma, A. D.; Garcia-Molina, H.; Polyzotis, N.; and Widom, J. 2011. Human-assisted graph search: It's okay to ask questions. *Proceedings of the VLDB Endowment* 4(5):267–278.

Raykar, V. C.; Yu, S.; Zhao, L. H.; and Valadez, G. 2010. Learning from crowds. *Journal of Machine Learning Research* 11:1297–1322.

Shahaf, D., and Horvitz, E. 2010. Generalized task markets for human and machine computation. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI '10)*.

Sheng, V. S.; Provost, F.; and Ipeirotis, P. G. 2008. Get another label? Improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the Fourteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '08)*.

Tran-Thanh, L.; Stein, S.; Rogers, A.; and Jennings, N. R. 2012. Efficient crowdsourcing of unknown experts using multi-armed bandits. In *Proceedings of the Twentieth European Conference on Artificial Intelligence (ECAI '12)*, 768–773.

Wauthier, F., and Jordan, M. 2011. Bayesian bias mitigation for crowdsourcing. In *Advances in Neural Information Processing Systems 24 (NIPS '11)*. 1800–1808.

Weld, D. S.; Mausam; and Dai, P. 2011. Human Intelligence *Needs* Artificial Intelligence. In *Human Computation Workshop (HCOMP '11)*.

Welinder, P.; Branson, S.; Belongie, S.; and Perona, P. 2010. The multidimensional wisdom of crowds. In *Advances in Neural Information Processing Systems 23 (NIPS '10)*. 2424–2432.

Whitehill, J.; Ruvolo, P.; Wu, T.; Bergsma, J.; and Movellan, J. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in Neural Information Processing Systems 22 (NIPS '09)*. 2035–2043.