Proceedings, The Fourth AAAI Conference on
Human Computation and Crowdsourcing

Learning and Feature Selection under Budget Constraints in Crowdsourcing

Besmira Nushi,[1] **Adish Singla,**[1] **Andreas Krause,**[1] **Donald Kossmann**[1,2]

[1]ETH Zurich, Department of Computer Science, Switzerland
[2]Microsoft Research, Redmond, WA, USA

Abstract

The cost of data acquisition limits the amount of labeled data available for machine learning algorithms, both at the training and the testing phase. This problem is further exacerbated in real-world crowdsourcing applications where labels are aggregated from multiple noisy answers. We tackle classification problems where the underlying feature labels are unknown to the algorithm and a (noisy) label of the desired feature can be acquired at a fixed cost. This problem has two types of budget constraints — the total cost of feature labels available for learning at the *training* phase, and the cost of features to use during the *testing* phase for classification. We propose a novel budgeted learning and feature selection algorithm, B-LEAFS, for jointly tackling this problem in the presence of noise. Experimental evaluation on synthetic and real-world crowdsourcing data demonstrate the practical applicability of our approach.

Introduction

Data acquisition is a costly and time-consuming process in many real-world applications including crowdsourcing, medical diagnosis, and sensor data aggregation. Furthermore, the acquired labels can be noisy due to imperfect answers from the crowd or noisy sensor measurements. In these applications, the performance of machine learning algorithms depends crucially on acquiring the most informative data in a cost-effective way. This motivates the need for designing algorithms that can satisfy budget constraints on the cost of data acquisition and are robust to noisy labels at the same time.

Budget constraints at training phase. We study feature-based classification problems for which the feature labels are unknown in the training phase, which is a commonly encountered setting in crowdsourcing (Welinder et al. 2010b). The algorithm can acquire the label of a desired feature for an instance and each such label has a fixed cost. However, there is a budget constraint for the training phase that limits the total cost of data acquisition by the algorithm. The end goal is to acquire the most informative labels to learn a classifier with low generalization error. This notion of budget constraint is different the one usually considered in active learning (Settles 2012). In active learning, the feature labels for all training instances are known and the budget constraint bounds the cost of acquiring class labels. Also, the data acquisition is more fine-grained in our settings since the algorithm can pick any specific training instance and acquire only a desired set of features. In practical terms, the budget constraints at the training phase map to the amount of budget that can be used in order to train a classifier before employing it to classify new instances.

Budget constraints at testing phase. To perform classification at the testing phase, the algorithm again needs to acquire the feature labels for the test instance. This is often the case in many classification problems where the feature values need to be collected at a given cost for every new instance and can not be computed automatically. For example, in crowdsourcing prediction markets, medical diagnosis, and sensor data aggregation, new predictions in the testing phase are mostly based on newly acquired data for the specific instance that needs to be classified.

This naturally leads to a budget constraint at the testing phase, which in practice bounds the cost of making a prediction, often tackled implicitly via feature selection techniques. However, current techniques for budgeted learning and feature selection (as well as the respective budget constraints) have been studied and designed independently from each other. Several methods apply feature selection strategies only after the labels have been collected for all features during the training phase, which can be prohibitively expensive especially if the number of initial candidate features is high. Then, after learning the predictive parameters of all the features, the least informative features are discarded to reduce the cost of predictions at the testing phase (Hall 2000; Guyon and Elisseeff 2003; Veeramachaneni, Olivetti, and Avesani 2006). In this work, we consider both the constraints simultaneously while learning a classifier.

Noisy labels. The problem of learning with budget constraints at the training phase has been formerly studied in a noise free setting for learning Naïve Bayes classifiers (Deng et al. 2013; Kapoor and Greiner 2005; Lizotte, Madani, and Greiner 2003). However, the proposed algorithms assume that the feature labels are noise-free, which is an unrealistic assumption for real-world applications. Hence, it is neccessary to support more robust learning models for handling noise, which further exacerbates the challenges arising from the above mentioned budget constraints.

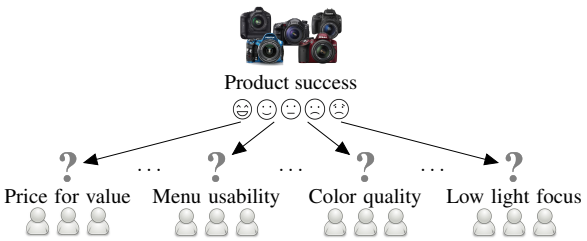


Figure 1: Example of feature-based crowdsourced predictions.

Figure 1 shows an example of making predictions from crowdsourced feature labels under budget constraints.

Example 1. FEATURE-BASED CROWDSOURCED PREDICTIONS *Imagine a company that produces various types of cameras and wants to implement a new strategy for predicting the overall success of its new products based on the customer evaluation of different product features: price for value, menu usability etc. The sales team collects this data via (paid) customer surveys on specific features. However, long surveys are cumbersome and time-consuming for the customers to complete. Moreover, some of the candidate features may be redundant / uninformative, and the evaluation from the customers may be noisy due to subjectivity or human error.*

*Given that there is no previous data available, the company: (i) has a limited budget for collecting data to train a model from its current products (*i.e. *training budget constraint), and (ii) prefers to have only a limited number of features in the surveys for the future products (*i.e. *testing budget constraint). Therefore, the questions of the sales team for this problem are:*

Q1. *What data should be collected in order to build a good training model?*

Q2. *Which features should be included in the final survey for predicting the customer satisfaction from new products.*

Similar questions arise in numerous other problems of making predictions based on crowdsourced feature labels. Analogous applications include medical diagnosis and sensor data aggregation, where the budget constraints respectively relate to the cost of medical tests or sensor measurements.

Our contributions are as follows:

- We propose a novel learning algorithm, B-LEAFS, to *jointly* tackle the problems of **B**udgeted **Lea**rning and **F**eature **S**election for training and testing classifiers that are robust to noisy feature labels.
- B-LEAFS operates in a Bayesian framework, and maintains posterior distributions over all model parameters, thereby enabling us to capture the uncertainty in the model parameters about individual features. The algorithm makes greedy decisions for selecting the next feature label to acquire by exploiting the submodularity of information gain from a feature, conditioned on the current state of learning. In addition, it effectively balances exploration and exploitation by employing Thompson sampling (Thompson 1933).
- We perform extensive experiments, both on synthetic as well as real-world crowdsourcing data, and compare it with state-of-the-art baselines.

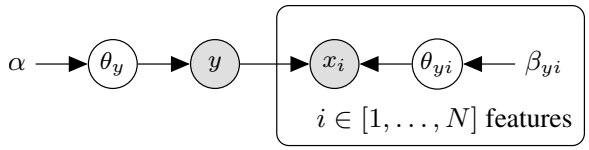


Figure 2: Naïve Bayes model.

Learning Models and Problem Statement

The problem of budgeted learning and feature selecion is tied to the specific classification model that needs to be trained. The goal is to learn a good classifier on a given budget. This section introduces the two models based on which we build our algorithm: Naïve Bayes model (Lowd and Domingos 2005), and Access Path model (Nushi et al. 2015) as a representative of models that handle crowdsourcing errors. Later, we formally define our problem in the context of these two models.

Feature-based classification models

Naïve Bayes model. Figure 2 shows the plate representation of a Naïve Bayes classifier. Each training instance in the Naïve Bayes model is characterized by a class / task variable Y taking values in the domain $\mathcal{Y}$, and a set of N feature variables $X = \{X_1, \ldots, X_N\}$. The Naïve Bayes model considers a noise-free setting, where each feature label is observed exactly once and its value is considered to be correct.

θ_y and θ_{yi} denote the underlying probability distributions for the task variable and the feature variables respectively. In this work, we will deal with categorical tasks and features. Therefore, θ_y and θ_{yi} are both modeled as Dirichlet distributions, where α and β_{yi} are the corresponding hyperparameters. We assume uniform hyperparameters on all the variables. The goal of training in this model is to learn the conditional probabilities of each feature given the task, *i.e.* $p(X_i|Y = y, \beta_{yi}, D)$ where D is the set of feature labels that have been collected so far. These probabilities correspond to the maximum a posteriori (MAP) estimates (Lowd and Domingos 2005) and can be computed exactly when Y is also fully observable (*i.e.* there exists a ground truth for the final task). Otherwise, alternative expectation-maximization (EM) techniques (Dempster, Laird, and Rubin 1977) can be employed to estimate parameters. Predictions are then performed via Bayesian Inference.

The main assumption in this model is the conditional independence of the feature variables given the task. The assumption may not always hold true due to possible correlations between features. This can lead to Naïve Bayes predictions being overconfident when features are not carefully engineered beforehand which further motivates the need for feature selection.

Access Path model. As mentioned earlier, the Naïve Bayes model handles a single label per feature for a given instance which is considered to be correct. In crowdsourcing settings where the labels acquired from the workers may be

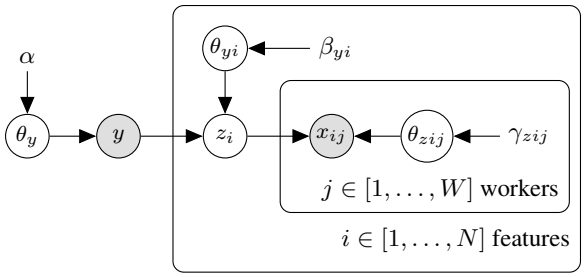


Figure 3: Access Path model.

noisy, it is common to ask the same question / label to multiple workers to ensure quality. In order to support such noisy labels, we adapt the Access Path model from Nushi et. al (2015) as shown in Figure 3 — a natural extension to Naïve Bayes while being able to handle multiple (possibly noisy) labels per feature. The main benefit of this model is that it decouples the noise / information that inherently comes with the features (*i.e.* $p(Z_i|Y = y, \beta_{yi}, D)$) from the noise / information that is further introduced by the workers (*i.e.* $p(X_{ij}|Z_i = z, \gamma_{zij}, D)$) by introducing a middle layer of latent variables Z_i for $i \in 1 \ldots N$. Similar to Dawid and Skene (2015), learning is performed via an expectation-maximization (EM) algorithm (Dempster, Laird, and Rubin 1977), while inference during the prediction step marginalizes over the latent feature variables. Due to the high sparsity of worker participation, this model shares the parameters for the workers within the same feature, which was shown to improve the accuracy of predictions. We follow the same guideline in our experiments as our algorithm runs in training time where the data sparsity is even higher. Nevertheless, if it is known that some of the workers have a sufficient participation to accurately estimate their quality, the model could be further extended to differentiate such workers from the rest by not sharing the θ_{zij} parameters.

This design was originally proposed to represent groups of correlated worker answers (*i.e.* access paths), with the goal of optimizing the prediction costs. In our work, we leverage this model by adapting the notion of worker groups to represent classification features.

Problem statement

We study feature-based classification problems for which the task label $Y = y$ is known during the training phase, and needs to be predicted by the classifier at the testing phase. The feature labels are unknown for both the training and testing instances and can be acquired at a given cost. In practice, there are various problems for which this assumption holds. In Example 1 for instance, the company may already know the success rate of previous products but it may not know the values of individual features and which ones are most informative for predicting the success of new products. Another example is animal species recognition in images as we show in our experiments. For this problem, one can bootstrap a classifier with training examples for which the species in the image is already known but the visual feature labels need to be determined.

Consider a Naïve Bayes model with N conditionally independent feature variables $X = \{X_1, \ldots, X_N\}$ as presented in Figure 2. At the beginning, the classifier has no knowledge about the underlying parameters of θ_{yi}, and we initialize them from the hyperparameters β_{yi}. At a given time, a new feature label is acquired $(Y = y, X_i = x_i)$ for the feature X_i on a task with label $Y = y$. Based on this labeled instance, the algorithm can update the distributions θ_{yi}. For illustration purposes, let us consider the following example: Suppose X_i is a feature variable that takes values in the domain $\{a, b, c\}$ while the task variable Y takes binary values $\{yes, no\}$. The current state of the model has $\theta_{yes,i} \sim Dir(3, 2, 5)$. After observing a new labeled instance $(y = yes, X_i = b)$ the posterior distribution will shift to the new state $\theta_{yes,i} \sim Dir(3, 3, 5)$.

In the context of the Access Path model shown in Figure 3, the algorithm acquires labels to learn the distributions of both the features (θ_{yi}) and the crowd workers (θ_{zij}) at the same time. As it is usually the case in crowdsourcing, on each feature observation we ask a total number of W different crowd workers. This will lead to a cost reduction of W from the available budget B.

Budget B at the training phase. First, the goal of *budgeted learning* is to learn the model parameters during the training phase under a given budget constraint B, with the goal of minimizing the classification error during the testing phase. The budget here limits the number of acquired labels represented by the labeled set D such that $|D| \leq B$. In crowdsourcing applications, this maps to the number of times the learner accesses the crowd. In Example 1, B is the number of times the algorithm asks a customer to evaluate a camera feature. While the learning problem is well understood for infinite budget, the budgeted version is known to be NP-HARD even for simplified variants when a feature is allowed to be queried only once (Lizotte, Madani, and Greiner 2003).

Budget K at the testing phase. Second, the goal of *feature selection* is to select a set of the best K features for classification in the testing phase where $K \leq N$. Hence, K is a budget constraint applicable to the testing phase. In Example 1, K corresponds to the number of camera features that will be included in the final survey design. For the Naïve Bayes model, the feature selection problem is also NP-HARD (Feige 1998), with viable approximation guarantees possible through greedy selection approaches (Krause and Guestrin 2005b; Sviridenko 2004). To the best of our knowledge, this is the first work that considers both challenges while learning a classifier, in both the noise-free and noisy setting.

Budgeted Learning and Feature Selection

Existing approaches. The purpose of any budget allocation strategy for training is to produce a classifier that in the future is going to make good predictions. The trivial approach would be to uniformly allocate the budget among all features without distinguishing the good features from the bad ones. This is also known as the ROUNDROBIN algorithm, and as shown in our experimental evaluation and previous work, is generally inferior to other schemes as some of the features are more informative than others. Other strategies continue

allocating budget to the same feature as long as this reduces the loss on the task variable (*i.e.*, the conditional entropy $H(Y|X_i)$) after this i-th feature is observed in isolation to the others. BIASEDROBIN (Lizotte, Madani, and Greiner 2003) is one representative algorithm in this spectrum and was shown to be competitive (Deng et al. 2013) when there exists only a relatively small number of candidate features. However, for a larger number of candidates this algorithm (i) fails to deal with correlated features, and (ii) delays the selection of the best candidates. These issues are then handled by another group of GREEDY selection methods which select in each iteration the i-th feature that reduces the overall loss of the whole model when added to the currently selected feature set X_S (*i.e.*, the conditional entropy $H(Y|X_{S\cup\{i\}})$). These methods are known to be efficient with strong theoretical guarantees (Krause and Guestrin 2005a) for fixed and known feature parameters learned from sufficient historical data. Our work is inspired from similar greedy selection techniques but adapted for the case of unknown feature parameters that are updated as the data is collected. A closely related approach, referred to as TSGREEDY in our experiments, proposed in (Schnitzler, Yu, and Mannor 2015), employs Thompson Sampling by greedily selecting in each iteration a whole set of K features from a sampled model. As discussed further during our experimental evaluation, this approach has a high exploration cost when feature labels are noisy and is applicable only to the Naïve Bayes model.

Limitations and challenges. The fundamental issue with the aforementioned techniques is related to the fact that the decision-making on which feature to observe next is based on the point estimates of the feature parameters for the feature variables X_i. If the collected data for the feature is not sufficient, these estimates might not be good representatives of the underlying parameter distribution. If so, the observed value of the feature may be far away from its expected value which can then lead to selecting non-informative features and making wrong classifications in the testing phase.

B-LEAFS for Naïve Bayes

Our approach. The key idea of our approach is to use the posterior distributions over the parameters (*e.g.* θ_y and θ_{yi} for Naïve Bayes) in order to encode the uncertainty about the true parameter value. Based on this Bayesian framework, we design the B-LEAFS algorithm inspired by Thompson Sampling (Thompson 1933; Agrawal and Goyal 2012), which is a natural way to balance exploration and exploitation for comparing various probability distributions. Next, we describe how we apply these ideas for Naïve Bayes and the Access Path model.

The B-LEAFS strategy (Algorithm 1) performs one iteration per feature selection / search (Lines 10-19). The k-the iteration is allowed to spend a maximum budget of $b_k = \frac{B}{2^k}$ when $k-1$ number of features have already been added to the best set. The reasoning behind this allocation is based on the experimental observation that the early iterations of the algorithm explore more and require more budget than the later ones. This iteration-based constraint ensures that each feature search does not spend more than a maximum amount of budget before including a new item in the best set even if the feature is not sufficiently credible. Every iteration performs the following steps:

Model sampling. (Line 13) On each feature label acquisition, B-LEAFS samples a set of parameters from the current posterior parameter distribution θ. Model sampling is essential for our method as it helps to balance exploration vs. exploitation trade-offs. The more data we observe from a certain feature, the more likely it is for the sampled parameter to be close to its mean value. Consequently, the early decisions of B-LEAFS will tend to explore more features while the later decisions will mostly focus on exploiting the current best ones. This is the reason why the initial iterations require more budget than the others. For Naïve Bayes, sampling is trivial and is performed as in Algorithm 2. First, we compute the Dirichlet posterior distributions θ from the feature value counts in the current dataset D, and then sample a model $\tilde{\theta}$ from the resulting distributions.

Submodular feature selection. (Line 14) Similar to the Thompson Sampling algorithm, we decide on which features to observe next based on the sampled model parameters $\tilde{\theta}$. More precisely, we select the feature i^* that brings the most information in addition to what the current best set S already provides, which prevents the algorithm from includ-

ALGORITHM 1. B-LEAFS

1 **Input:** feature variables $X = \{X_1, \ldots, X_N\}$
2 training budget constraint B
3 testing constraint K features
4 credibility constraint δ
5 **Output:** parameters $\tilde{\theta} = \{\tilde{\theta}_y, \tilde{\theta}_{yi}\ \forall y \in \mathcal{Y}, i \in [1, \ldots, N]\}$
6 best feature set S s.t. $|S| = K$
7 **Initialize:** $S = \emptyset$, $D = \emptyset$
8 uniform priors $\alpha, \beta = \{\beta_{yi}\ \forall y \in \mathcal{Y}, i \in [1, \ldots, N]\}$
9 **for** $(k = 1; k \leq K; k++)$ **do**
10 $b_k = \lfloor \frac{B}{2^k} \rfloor$ // $\frac{B}{2^{k-1}}$ for $k = K$
11 $i^* = \texttt{null}$
12 **while** $((\neg$ISCREDIBLE$(i^*, D, \beta, \delta))$ *and* $(b_k > 0))$ **do**
13 $\tilde{\theta} =$ SAMPLEMODEL(D, α, β)
14 $i^* = \arg\max_{i \in [1,\ldots,N] \setminus S}$ IG$(Y; X_{S\cup\{i\}}|\tilde{\theta})$
15 Draw $y \sim P(Y)$
16 Observe x_{i^*} for a task s.t. $Y = y$
17 $D = D \cup \{(y, x_{i^*})\}$
18 $b_k = b_k - 1$, $B = B - 1$
19 $S = S \cup \{i^*\}$
20 $\tilde{\theta} = \text{MAP}(D, \alpha, \beta)$
21 **return** $\tilde{\theta}$, S

ALGORITHM 2. SAMPLEMODEL- Naïve Bayes

1 **Input:** collected data D, priors α, β
2 **Output:** parameters $\tilde{\theta} = \{\tilde{\theta}_y, \tilde{\theta}_{yi}\ \forall y \in \mathcal{Y}, i \in [1, \ldots, N]\}$
3 Sample $\tilde{\theta} \sim P(\theta|D, \alpha, \beta)$
4 **return** $\tilde{\theta}$

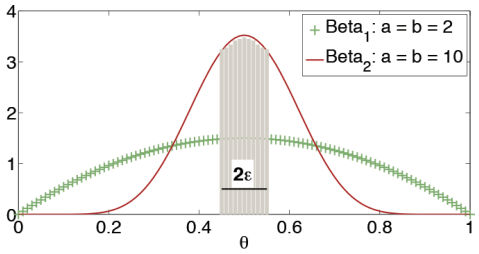


Figure 4: Example of binary parameter distributions

ing correlated features. Note that we use information gain as a decision-theoretic objective function which has been shown to be submodular for both the models described in the previous section due to the feature conditional independence assumption. The benefit of using a submodular set function is that it enables the application of efficient greedy approximation schemes (Sviridenko 2004). Moreover, information gain can be approximately computed via sampling from the model network as shown in (Krause and Guestrin 2005a).

Parameter credibility check. (Line 12) Before adding a feature to the best set, B-LEAFS checks whether: (i) enough budget has been invested for the feature search, and (ii) the posterior distributions of this feature, θ_{yi^*}, are sufficiently credible and concentrated around the mean. For instance, imagine two binary features whose conditional distribution $P(X_i|Y)$ follows a Bernoulli distribution. Suppose also that the conjugate priors for these features are Beta distributions as shown in Figure 4. Both these distributions have a mean value of $\mu = 0.5$. However, Beta_1 has only a few observations and the mass of distribution within a credible interval $[\mu - \epsilon; \mu + \epsilon]$ as shown in the shaded area is smaller than for Beta_2. Formally, we define the credibility of a feature X_i to be within a 2ϵ credibility interval around its mean as follows:

$$q_i = \sum_{y \in Y} \frac{P(Y=y)}{|\mathcal{X}_i|} \sum_{x \in \mathcal{X}_i} \int_{\mu-\epsilon}^{\mu+\epsilon} \text{Beta}(a^x_{yi}, a^0_{yi} - a^x_{yi}) d\theta^x_{yi} \tag{1}$$

Here, a^x_{yi} is the number of times value $x \in \mathcal{X}_i$ has been observed when $Y = y$, and a^0_{yi} is the total number of examples for which $Y = y$. Note that a^x_{yi} represents concentration parameters of the Dirichlet distribution for θ_{yi}. The normalization by $\frac{P(Y=y)}{|\mathcal{X}_i|}$ ensures that the credibility of the whole feature is weighed according to the distribution of Y and the feature variable cardinality. The function ISCREDIBLE returns true if $q_i \geq 1 - \delta$, and false otherwise. The function is generic and can use other notions of parameter concentration such as variance.

In our implementation, we fix $\epsilon = 0.05$ while varying the credibility parameter δ according to the budget constraints. Our initial guideline with this respect is that the algorithm should spend more than $\frac{B}{N}$ budget units on the selected features which is the amount of budget that ROUNDROBIN would use. Without making any assumptions on the conditional distribution of $P(X_i|Y)$, one can compute a corresponding δ_L value for this guideline. Furthermore, one can also compute a corresponding δ_U value for the ideal case when the algorithm would spend on average a budget of $\frac{B}{K}$ on each selected feature. The closer δ is to δ_U, the tighter the credibility requirements are. As the algorithm also requires budget for exploration, δ_U is an unrealistic requirement. In our experiments, we observe that $\delta = \frac{\delta_U - \delta_L}{2}$ is sufficiently stringent to detect informative features.

k	#observations per feature								Selected features (S)
1	4	5	7	5	3	6	2	28	$\{X_8\}$
2	20	34	14	14	13	10	12	28	$\{X_2, X_8\}$
3	36	34	16	19	16	13	15	28	$\{X_1, X_2, X_8\}$
4	36	34	19	32	20	13	18	28	$\{X_1, X_2, X_4, X_8\}$

Table 1: Example of running B-LEAFS on the `nursery` dataset for $B = 200$ and $K = 4$. The optimal set of features without training phase budget constraints is $\{X_1, X_2, X_4, X_8\}$.

Discussion. The B-LEAFS algorithm is based on a generic class of greedy selection algorithms for maximizing monotone submodular set functions (Sviridenko 2004; Feige 1998). The crucial difference here is that the objective function (*i.e.* information gain in our case) is computed on the sampled model for balancing exploration and exploitation. The data collection then (Line 17) follows a traditional greedy selection, while the feature selection (Line 19) instead appends new features only when they are sufficiently credible, which is the major distinction of B-LEAFS from the traditional Thompson Sampling algorithm. Both decisions are guided by the marginal increase of information gain after observing one additional feature value.

Example 2. In Table 1 we show an example of running B-LEAFS on the `nursery` dataset from the UCI repository (Blake and Merz 1998). In this example we required the algorithm to spend a maximum budget of $B = 200$ while selecting $K = 4$ (out of 8) features. Each element in the collected data vector represents the number of times a feature has been observed. The algorithm adds one feature per iteration (marked in gray) in the set S and most of the budget is spent in the first two iterations of the algorithm. After the last iteration, the algorithm manages to recover the optimal set of features $\{X_1, X_2, X_4, X_8\}$ which is the set of features that would have been selected from a model trained with all the available data in the dataset. Note that the amount of data assigned to the algorithm in this case ($B = 200$) is only 2% of the overall data available in `nursery`.

B-LEAFS does not spend more budget on a feature once the feature has been added to S. One could observe that the final model θ could still benefit from reaccessing such features for further improving their parameters. However, once the best set of features is known, such refinements can be achieved in a more cost-efficient way by periodically retraining the model with new instances in the testing phase.

ALGORITHM 3. SAMPLEMODEL- Access Path model

1 **Input:** collected data D, priors α, β, γ
2 **Output:** parameters $\tilde{\theta} = \{\tilde{\theta}_y, \tilde{\theta}_{yi}, \tilde{\theta}_{zij}\ \forall y, z, i, j\}$
3 **for** $(t = 1; t \leq T; t++)$ **do**
4 $\quad \theta_y^{(t)} \sim P(\theta_y^{(t-1)} | \alpha, D)$
5 $\quad$ **for** $(i = 1; i \leq N; i++)$ **do**
6 $\quad\quad$ **foreach** $d \in D$ **do**
7 $\quad\quad\quad z_i^{(t)}[d] \sim P(Z_i | \theta_{yi}^{(t-1)}, D)$ //update Z_i for instance d
8 $\quad\quad \theta_{yi}^{(t)} \sim P(\theta_{yi} | z_i^{(t)}[1 \ldots |D|], \beta_{yi}, D)$
9 $\quad\quad$ **for** $(j = 1; j \leq W; j++)$ **do**
10 $\quad\quad\quad \theta_{zij}^{(t)} \sim P(\theta_{zij} | z_i^{(t)}[1 \ldots |D|], \gamma_{zij}, D)$
11 **return** $\theta^{(t)}$

B-LEAFS for the Access Path model

B-LEAFS for the Access Path model follows the same structure as Algorithm 1. However, due to the introduction of the feature layer with latent variables Z_i, model sampling and the parameter credibility check are accordingly adjusted.

Model sampling. Sampling from the Access Path model parameters entails sampling from θ_y, as well as θ_{yi} and θ_{zij}. In contrast to the Naïve Bayes model, here the actual counts that involve the Z_i variables cannot be observed. We overcome this problem by applying Gibbs sampling (Geman and Geman 1984) as a Markov Chain Monte Carlo algorithm for computing approximate observations from a joint distribution of random variables. Gibbs sampling is a suitable choice for the Access Path model given that the hidden variables θ_y, θ_{yi}, and Z_i are characterized by conditional probability distributions as depicted in Figure 3.

As shown in Algorithm 3, we implement these ideas for alternatively sampling: (i) the value of the latent feature variables Z_i (Line 7), and (ii) the model parameters θ_{yi} and θ_{zij} (Line 8 and 10). Each iteration samples one latent variable at a time given the state of the rest of the other variables present in the model. Note that the Z_i feature variables need to be sampled for all the data that has been collected so far which requires multiple passes through the data. However, for more practical applications, the algorithm can still benefit from parallelizing sampling across features thanks to the conditional independence assumption.

Parameter credibility check. Similar to what we discussed earlier, B-LEAFS checks the credibility of the feature parameters by using the definition in Equation 1. For the Access Path Model we ensure that both the feature and crowd parameters (θ_{yi} and θ_{zij}) satisfy this condition. Depending on the amount of crowdsourcing redundancy W and noise, one may also enforce different δ-credibility requirements for θ_{yi} and θ_{zij}. For simplicity, in all our experiments we apply the same δ to both the parameters.

Experimental Evaluation

Datasets. In this section, we show experimental results on two types of open and publicly available data sources: datasets from the UCI Machine Learning Repository (Blake and Merz 1998) and the CUB-200 birds classification dataset (Welinder et al. 2010a). Both datasets consist of categorical features and tasks. UCI datasets are labeled by domain experts and contain a single expert label per feature.

CUB-200 instead, was created as part of a large-scale crowdsourced data collection for bird species recognition. The dataset contains 6,033 images allocated over 200 different species. There are 288 candidate binary features in total, and the authors collected 5-10 crowdsourcing labels per feature per image. Therefore, for this task workers solved tasks with image-related questions like: "*Is the color of the bird's beak yellow?*". The collected answers then are used as labels for the corresponding features. The dataset does not have any ground truth on the true feature values. However, one can measure the amount of disagreement which is on average 3%-10% per image. Some features have a higher disagreement than others which can reach up to 50% for features that are hard to distinguish.

Baselines

We compare our approach with the following three different baselines from related work:

ROUNDROBIN (Lizotte, Madani, and Greiner 2003) uniformly allocates the budget across features in a round-robin fashion, applying thereby the pure-exploration strategy.

BIASEDROBIN (Lizotte, Madani, and Greiner 2003) continues observing the same feature as long as this action reduces the expected value of loss on the task variable, *e.g.* the conditional entropy $H(Y|X_i)$. Note that this notion of loss does not take into account sets of features but only the current feature in isolation. This may lead to selecting redundant features that are correlated with each other. Once the expected value of the loss does not decrease anymore, the algorithm starts exploring other features. Based on this design, the approach tends to quickly exploit features that seem to be more informative without exploring the whole candidate set. However, this optimistic behavior may prevent the algorithm from exploiting features that have not been encountered yet before the budget is exhausted.

TSGREEDY (Schnitzler, Yu, and Mannor 2015) is the closest approach to our work although it is applicable only for the Naïve Bayes model. It applies a traditional Thompson Sampling algorithm which runs $T = \frac{B}{K}$ iterations. Each iteration involves three main steps: (1) model sampling (2) submodular selection of K features via the GREEDY algorithm (3) observing the K features and updating the current model. In contrast to our approach, this algorithm selects K features at a time and does not check the credibility of feature parameters before including them in the best set. As a result, the algorithm has a higher exploration cost especially in the presence of noise. For comparison purposes, we adapted TS-GREEDY for the Access Path model by employing the same Gibbs Sampling algorithm as in Algorithm 3.

Since none of these algorithms involve explicit feature selection, for a fair comparison, we first learn a model from the collected data and then run a submodular GREEDY feature selection scheme on the resulting model (Krause and Guestrin 2005a) similar to the feature observation step in B-

Algorithm	#observations per feature	Selected features (S)	Error
RoundRobin	12 11 11 11 11 11 11 11 11	$\{\underline{X_2}, X_6\}$	0.10
BiasedRobin	18 10 7 16 6 15 7 18 3	$\{X_1, X_8\}$	0.11
TsGreedy	6 17 13 12 2 15 4 16 15	$\{\underline{X_2}, X_8\}$	0.08
B-Leafs	1 45 10 0 2 38 2 1 0	$\{\underline{X_2}, X_6\}$	0.05

Table 2: Example of running all the algorithms on the `breast-cancer` dataset for $B = 100$ and $K = 2$. For the same dataset, the optimal set of features without training phase budget constraints is $\{\underline{X_2}, \underline{X_3}\}$ and results to a 0.04 prediction error.

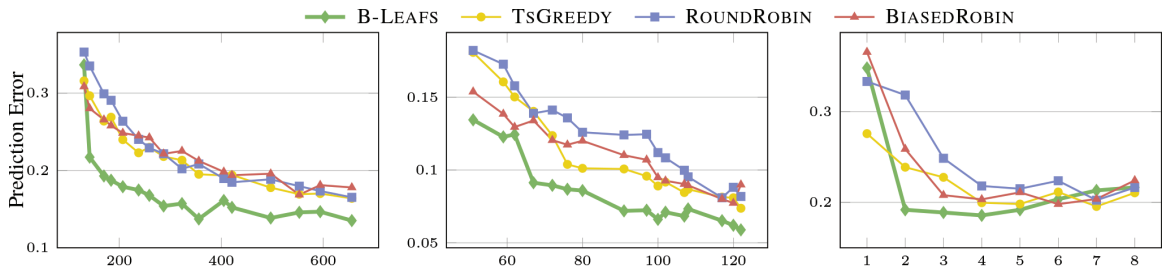


Figure 5: Budgeted learning and feature selection on Naïve Bayes. B-Leafs can make better predictions and is able to distinguish the most informative features. All techniques are most beneficial for tighter test budget constraint (i.e. lower K).

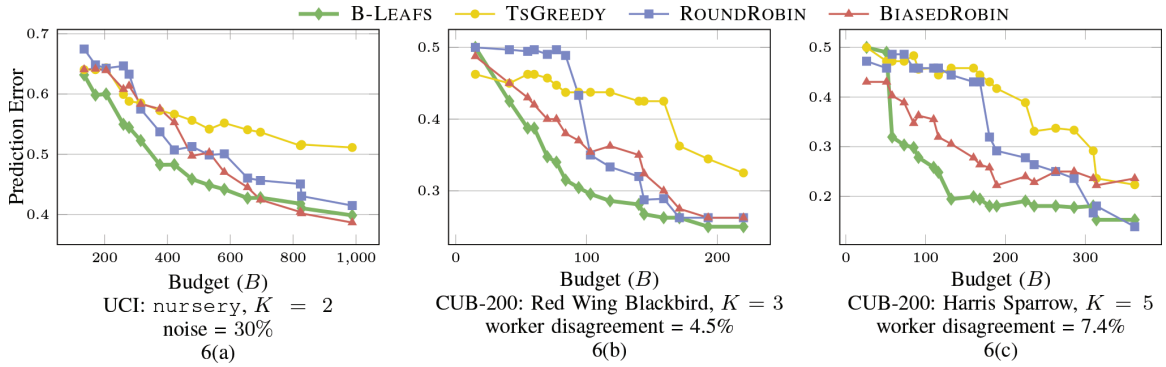


Figure 6: Budgeted learning and feature selection on the Access Path model. B-Leafs outperforms the other strategies on both synthetic and real-world crowdsourcing data. All strategies are comparable for high budget except TsGreedy which spends a high budget on exploration.

Leafs (Line 14). In every step, the greedy algorithm selects the feature i that maximizes the marginal increase in information gain $\text{IG}(Y; X_{S\cup\{i\}})$ after adding this feature to the current best set.

For all experiments, we keep 80% of the data for training and 20% for testing. All methods select feature labels from the training set and the resulting model with only the selected features is then evaluated on the testing set. This process is repeated 16 times for each experiment.

Example 3. To illustrate the differences between the different approaches, in Table 2 we show an example of running all approaches on the `breast-cancer` UCI dataset by assigning a budget constraint of $B = 100$ and $K = 2$. The optimal set of features for this dataset is $\{\underline{X_2}, \underline{X_3}\}$. Although none of the methods is able to recover the full best set, the differences between the various algorithms can be observed in the final set of selected features S and the amount of budget that is spent to learn each of these features in the collected data vector. BiasedRobin misses the selection of $\underline{X_2}$ as it quickly decides to exploit the previous feature. TsGreedy is able to identify $\underline{X_2}$ as one of the best features but does not spend sufficient budget on it which leads to a higher error. In fact, the budget in TsGreedy is more uniformly distributed. B-Leafs instead manages to have a lower error rate given that it is able to better refine the parameters of the features in S by investing more budget in them and spending

significantly less budget in less informative features (*e.g.* X_9 is the least informative feature here).

Evaluation on the Naïve Bayes model. We first show results on the Naïve Bayes model. In Figures 5(a) and 5(b) we show the error rate of the resulting classifiers in testing phase on two different UCI datasets while varying the training budget constraint B. The `nursery` dataset has $N = 8$ categorical features while the `breast-cancer` dataset has $N = 10$ categorical features. B-LEAFS is able to make more accurate predictions at a lower cost. More interestingly, for the `breast-cancer` dataset it is able to identify the top most informative features even though the best features of this dataset are comparably informative. BIASEDROBIN instead quickly overexploits some features and fails to select the best set. Finally, TSGREEDY is slower in identifying the best features due its longer explorative behavior.

In the experiment in Figure 5(c), we vary the testing phase constraint K, *i.e.* the number of features that can be used for prediction. In the `nursery` dataset, only two of the features are informative for classification which explains why the error improvement saturates for all algorithms when $K > 2$. In general, we also observe that our approach (as well as BIASEDROBIN and TSGREEDY) are more beneficial when it is required to select a small number of features. Otherwise, if K is comparable to the size of the candidate set N, their performance converges to the simplistic ROUNDROBIN.

Evaluation on the Access Path model. To evaluate our approach in a crowdsourcing setting, we initially add synthetic noise of 30% to the UCI datasets. In the experiment shown in Figure 6(a), for instance, we synthetically generate 5 feature labels based on the actual feature label in the dataset. In 70% of the cases, the label will correspond to the true value. The rest of the labels, are uniformly picked from the rest of the possible feature values. Multiclass datasets with non-binary features like `nursery` are highly sensitive to such noise. The interesting observation here is that, TSGREEDY's performance significantly deteriorates in noisy settings, although it is fairly comparable to other baselines for noise-free observations. B-LEAFS then is more beneficial for lower rather than higher training budget constraints.

Figures 6(b) and 6(c) show results on the CUB-200 datasets. For brevity, we include evaluation outcomes on two different bird species. These experiments involve one-vs-all classification tasks for birds that belong to the same category (*e.g.* different kinds of sparrows) which is a more challenging task than classifying birds that belong to different categories. For each species, we include in the candidate feature set the top K most informative features and $N - K$ other randomly selected ones ($N = 20$). Results show that B-LEAFS outperforms the prediction error of BIASEDROBIN. All strategies are comparable for high budget except TSGREEDY, which again has a long exploration phase.

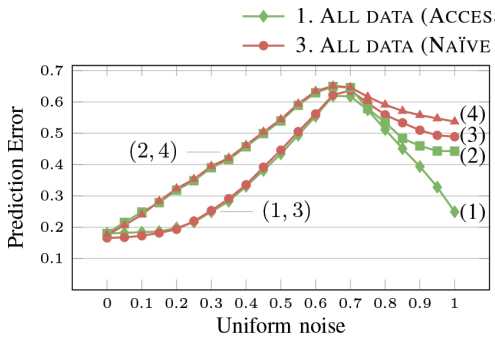
Noise impact. Finally, in the experiments in Figure 11, we study the behavior of both learning models under two different noise regimes: *uniform* noise and *biased* noise. For both regimes we synthetically generate 5 feature labels for both the training and the testing splits of the dataset. In the uniform noise case, the label either replicates the true feature value or is uniformly picked with a varying probability (horizontal axis) from the possible feature values different from the true value. This simulates random but non-biased worker mistakes. The biased noise regime instead imitates situations when workers could have a strong bias in consistently confusing the true value for another erroneous feature value. Note that, both noise regimes are present in real-world crowdsourcing platforms. Moreover, in these experiments we also show the discrepancy of the prediction error of models trained with all the available data in the dataset (ALL DATA) and the models trained with only the data collected by B-LEAFS with a a budget constraint of $B = 1000$. In both settings, we select $K = 2$ best features.

Figure 11(a) shows that in the uniform noise regime both learning models have comparable accuracy when noise is lower than 0.65. However, for higher levels of noise, the Access Path model is able to better recover from the uniform noise mistakes by inversely interpreting the feature labels. Also, the B-LEAFS algorithm is able to construct a better model if applied together with the Access Path model for noisy feature labels. Similar observations are also depicted in Figure 11(b) for biased noise. However, these results also show that biased mistakes are more difficult to recover as the biased noise may lead to the selection of a non-optimal set of features. This explains the discrepancy between the models built from B-LEAFS and the ALL DATA setting.

Related Work

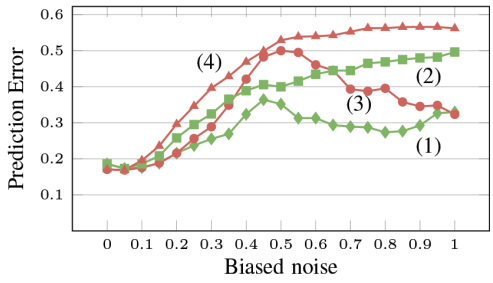
Learning under budget constraints. Budgeted learning has previously been studied in the context of Naïve Bayes classifiers (Deng et al. 2013; 2007; Lizotte, Madani, and Greiner 2003). The problem is however discussed orthogonal to feature selection and assumes accurate feature labels. (Kapoor and Greiner 2005) formally define the joint problem but the proposed algorithms first learn the parameters of the whole candidate set and then adaptively select features via a bounded decision tree. While this adaptive approach improves the classification accuracy, it requires a significant amount of data for building a plausible decision tree. Recently, (Kusner et al. 2014) study the optimization of the computation time of feature extraction during training.

Another line of research is related to best-arm identification under budget constraints (Lin, Li, and Chen 2015; Ding et al. 2013). Nevertheless, these methods are not designed for learning classifiers but rather for making a bounded selection of actions that maximize the joint reward. Bandit algorithms have been lately applied in the crowdsourcing setting (Biswas et al. 2015; Zhou, Chen, and Li 2014; Tran-Thanh et al. 2014; 2013; Abraham et al. 2013) for worker selection purposes in the context of expert crowdsourcing. Our work addresses similar exploration-exploitation trade-offs related to feature selection rather than worker selection for training and testing classifiers in a cost-efficient way. The main difference between the two problems is the fact that for the same example a worker can be accessed only once while a feature can be labeled multiple times from various workers. Both problems are however complementary to each other and integrating both into a single framework is an important avenue for future work.



UCI: nursery, $K = 2$, $B = 1000$

7(a)



UCI: nursery, $K = 2$, $B = 1000$

7(b)

Yet another optimization aspect relevant to crowdsourcing is related to applications where the task label is unknown both at training and testing time and is acquired at a given cost. Although the key ideas of B-Leafs (*i.e.* model sampling, credibility check, greedy feature selection) are valid even for these applications, the algorithm still needs to consider an additional cost-quality trade-off between labeling features for a new instance or labeling features for instances that are already in the dataset. This problem has been studied in recent work (Lin, Mausam, and Weld 2016; 2014) in the context of (re)active learning in crowdsourcing where the task labels are unknown but the feature labels are cost-free. Balancing this trade-off for the generic case of unknown task and feature labels would combine related aspects of active learning and budgeted learning.

Learning from crowdsourced data. The rapid advances in crowdsourcing applications created new opportunities for the community to collect and annotate data. However, crowdsourcing processes for large-scale data acquisition are expensive and prone to noisy annotations. This motivated the need for applying active learning techniques (Lin, Mausam, and Weld 2016; Mozafari et al. 2014; Zheng, Scott, and Deng 2010) for reducing the cost of data collection. These approaches mainly focus on actively selecting tasks and workers and assume that feature labels are cost-free. The closest study to our method is presented by (Schnitzler, Yu, and Mannor 2015). The authors propose a traditional Thompson Sampling algorithm for selecting observations from noisy sensors. As we show in the experimental section, the algorithm requires a longer exploration phase, which increases the cost of selecting the best features.

In this work, we use the Access Path model for learning from crowdsourced data. Similar approaches group workers (Venanzi et al. 2014) and tasks (Kamar, Kapoor, and Horvitz 2015) to overcome the challenges of data sparsity and task-dependent biases. Furthermore, other worker selection and crowd access optimization strategies (Ho, Jabbari, and Vaughan 2013; Karger, Oh, and Shah 2011) can be leveraged to achieve a fine-grained optimization customized to the pool of crowd workers.

Finally, recent work has focused on crowdsourcing feature engineering and discovery for machine learning classifiers (Zou, Chaudhuri, and Kalai 2015; Cheng and Bernstein 2015; Singla, Tschiatschek, and Krause 2016). Our ideas are complementary to feature discovery as they can be applied to refine the learning model after obtaining a viable candidate feature set from the crowd.

Conclusion

In this work, we studied the problem of building machine learning models from crowdsourced data under training and testing budget constraints. In particular, we focused on feature-based classification models and proposed a novel budgeted learning and feature selection algorithm which naturally balances exploration-exploitation trade-offs. This approach adaptively acquires crowdsourced feature labels for learning classifiers that can make accurate predictions at a lower cost. In the future, we plan to integrate these ideas with further optimization strategies related to worker / task selection, and provide end-to-end guidelines to the process of learning in crowdsourcing applications.

Acknowledgements. The authors would like to thank Ece Kamar for helpful discussions. This work was supported in part by the Swiss National Science Foundation, and NanoTera.ch program as part of the Opensense II project. Adish Singla acknowledges support by a Facebook Graduate Fellowship.

References

Abraham, I.; Alonso, O.; Kandylas, V.; and Slivkins, A. 2013. Adaptive crowdsourcing algorithms for the bandit survey problem. In *COLT 2013*, 882–910.

Agrawal, S., and Goyal, N. 2012. Analysis of thompson sampling for the multi-armed bandit problem. In *COLT*, 39.1–39.26.

Biswas, A.; Jain, S.; Mandal, D.; and Narahari, Y. 2015. A truthful budget feasible multi-armed bandit mechanism for crowdsourcing time critical tasks. In *AAMAS*, 1101–1109.

Blake, C., and Merz, C. J. 1998. UCI repository of machine learning databases.

Cheng, J., and Bernstein, M. S. 2015. Flock: Hybrid crowd-machine learning classifiers. In *CSCW*, 600–611.

Dempster, A. P.; Laird, N. M.; and Rubin, D. B. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)* 1–38.

Deng, K.; Bourke, C.; Scott, S. D.; Sunderman, J.; and Zheng, Y. 2007. Bandit-based algorithms for budgeted learning. In *ICDM*, 463–468.

Deng, K.; Zheng, Y.; Bourke, C.; Scott, S. D.; and Masciale, J. 2013. New algorithms for budgeted learning. *Machine Learning* 90(1):59–90.

Ding, W.; Qin, T.; Zhang, X.; and Liu, T. 2013. Multi-armed bandit with budget constraint and variable costs. In *AAAI*.

Feige, U. 1998. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)* 45(4):634–652.

Geman, S., and Geman, D. 1984. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 721–741.

Guyon, I., and Elisseeff, A. 2003. An introduction to variable and feature selection. *JMLR*.

Hall, M. A. 2000. Correlation-based feature selection for discrete and numeric class machine learning. In *ICML*.

Ho, C.; Jabbari, S.; and Vaughan, J. W. 2013. Adaptive task assignment for crowdsourced classification. In *ICML*, 534–542.

Kamar, E.; Kapoor, A.; and Horvitz, E. 2015. Identifying and accounting for task-dependent bias in crowdsourcing. In *HCOMP*, 92–101.

Kapoor, A., and Greiner, R. 2005. Learning and classifying under hard budgets. In *ECML*, 170–181.

Karger, D. R.; Oh, S.; and Shah, D. 2011. Budget-optimal crowdsourcing using low-rank matrix approximations. In *Allerton*, 284–291.

Krause, A., and Guestrin, C. 2005a. Near-optimal nonmyopic value of information in graphical models. In *UAI*, 324–331.

Krause, A., and Guestrin, C. 2005b. A note on the budgeted maximization of submodular functions. Technical Report CMU-CALD-05-103, Carnegie Mellon University.

Kusner, M. J.; Chen, W.; Zhou, Q.; Xu, Z. E.; Weinberger, K. Q.; and Chen, Y. 2014. Feature-cost sensitive learning with submodular trees of classifiers. In *AAAI*, 1939–1945.

Lin, T.; Li, J.; and Chen, W. 2015. Stochastic online greedy learning with semi-bandit feedbacks. In *NIPS*, 352–360.

Lin, C. H.; Mausam; and Weld, D. S. 2014. To re(label), or not to re(label). In *HCOMP*.

Lin, C. H.; Mausam; and Weld, D. S. 2016. Re-active learning: Active learning with relabeling. In *AAAI*.

Lizotte, D. J.; Madani, O.; and Greiner, R. 2003. Budgeted learning of naive-bayes classifiers. In *UAI*, 378–385.

Lowd, D., and Domingos, P. M. 2005. Naive bayes models for probability estimation. In *ICML*, 529–536.

Mozafari, B.; Sarkar, P.; Franklin, M. J.; Jordan, M. I.; and Madden, S. 2014. Scaling up crowd-sourcing to very large datasets: A case for active learning. *PVLDB* 8(2):125–136.

Nushi, B.; Singla, A.; Gruenheid, A.; Zamanian, E.; Krause, A.; and Kossmann, D. 2015. Crowd access path optimization: Diversity matters. In *HCOMP*, 130–139.

Schnitzler, F.; Yu, J. Y.; and Mannor, S. 2015. Sensor selection for crowdsensing dynamical systems. In *AISTATS*, 829–837.

Settles, B. 2012. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 6(1):1–114.

Singla, A.; Tschiatschek, S.; and Krause, A. 2016. Noisy submodular maximization via adaptive sampling with applications to crowdsourced image collection summarization. In *AAAI*.

Sviridenko, M. 2004. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters* 32(1):41–43.

Thompson, W. R. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 285–294.

Tran-Thanh, L.; Venanzi, M.; Rogers, A.; and Jennings, N. R. 2013. Efficient budget allocation with accuracy guarantees for crowdsourcing classification tasks. In *AAMAS*, 901–908.

Tran-Thanh, L.; Stein, S.; Rogers, A.; and Jennings, N. R. 2014. Efficient crowdsourcing of unknown experts using bounded multi-armed bandits. *Artificial Intelligence* 214:89–111.

Veeramachaneni, S.; Olivetti, E.; and Avesani, P. 2006. Active sampling for detecting irrelevant features. In *ICML*, 961–968.

Venanzi, M.; Guiver, J.; Kazai, G.; Kohli, P.; and Shokouhi, M. 2014. Community-based bayesian aggregation models for crowdsourcing. In *WWW*, 155–164.

Welinder, P.; Branson, S.; Mita, T.; Wah, C.; Schroff, F.; Belongie, S.; and Perona, P. 2010a. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology.

Welinder, P.; Branson, S.; Belongie, S. J.; and Perona, P. 2010b. The multidimensional wisdom of crowds. In *NIPS*, 2424–2432.

Zheng, Y.; Scott, S. D.; and Deng, K. 2010. Active learning from multiple noisy labelers with varied costs. In *ICDM*, 639–648.

Zhou, Y.; Chen, X.; and Li, J. 2014. Optimal PAC multiple arm identification with applications to crowdsourcing. In *ICML*, 217–225.

Zou, J. Y.; Chaudhuri, K.; and Kalai, A. T. 2015. Crowdsourcing feature discovery via adaptively chosen comparisons. In *HCOMP*, 198.