Optimal Decision Tree with Noisy Outcomes

Abstract

A fundamental task in active learning involves performing a sequence of tests to identify an unknown hypothesis that is drawn from a known distribution. This problem, known as optimal decision tree induction, has been widely studied for decades and the asymptotically best-possible approximation algorithm has been devised for it. We study a generalization where certain test outcomes are noisy, even in the more general case when the noise is persistent, i.e., repeating a test gives the same noisy output, disallowing simple repetition as a way to gain confidence. We design new approximation algorithms for both the non-adaptive setting, where the test sequence must be fixed *a-priori*, and the adaptive setting where the test sequence depends on the outcomes of prior tests. Previous work in the area assumed at most a logarithmic number of noisy outcomes per hypothesis and provided approximation ratios that depended on parameters such as the minimum probability of a hypothesis. Our new approximation algorithms provide guarantees that are nearly best-possible and work for the general case of a large number of noisy outcomes per test or per hypothesis where the performance degrades smoothly with this number. Our results adapt and generalize methods used for submodular ranking and stochastic set cover. We evaluate the performance of our algorithms on two natural applications with noise: toxic chemical identification and active learning of linear classifiers. Despite our theoretical logarithmic approximation guarantees, our methods give solutions with cost very close to the information theoretic minimum, demonstrating the effectiveness of our methods.

1 Introduction

The classic optimal decision tree (ODT) problem involves identifying an initially unknown hypothesis $\bar{x}$ that is drawn from a known probability distribution over a set of m possible hypotheses. We can perform tests in order to distinguish between these hypotheses. Each test produces a binary outcome (positive or negative) and the precise outcome of each hypothesis-test pair is known beforehand.[1] So an instance of ODT can be viewed as a ± 1 matrix with the hypotheses as rows and tests as columns. The goal is to identify hypothesis $\bar{x}$ using the minimum number of tests in expectation.

As a motivating application, consider the following task in medical diagnosis [27]. A doctor needs to diagnose a patient's disease by performing tests. Given an *a priori* probability distribution over possible diseases, what sequence of tests should the doctor perform? Another application is in active learning [11]. Given a set of data points, one wants to learn a classifier that labels the points correctly as positive and negative. There is a set of m possible classifiers which is assumed to contain the true classifier. In the Bayesian setting, which we consider, the true classifier is drawn from some known probability distribution. The goal is to identify the true classifier by querying labels at the minimum number of points (in expectation). Other applications include entity identification in databases [6] and experimental design to choose the most accurate theory among competing candidates [15].

An important issue that is not considered in the classic ODT model is that of unknown or noisy outcomes. In fact, our research was motivated by a dataset involving toxic chemical identification

[1] We consider binary test outcomes only for simplicity: our results also hold for finitely many outcomes.

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

where the outcomes of many hypothesis-test pairs are stated as unknown (one of our experimental results is also on this dataset). Prior work incorporating noise in ODT [15] is restricted to settings with very sparse noise. In this paper, we design approximation algorithms for the noisy optimal decision tree problem in full generality.

We consider a standard model for persistent noise. Certain outcomes (i.e., entries in the hypothesis-test matrix) are random with a known distribution: for simplicity we treat each noisy outcome as an unbiased ± 1 random variable. Our results extend directly to the case when each noisy outcome has a different probability of being ± 1. Persistent noise means that repeating the same test always produces the same ± 1 outcome. We assume that the instance is identifiable, i.e., a unique hypothesis can always be identified irrespective of the noisy outcomes. (This assumption can be relaxed: see §6.)

We consider both non-adaptive policies (where the test sequence is fixed upfront) and adaptive policies (where the test sequence is built incrementally and depends on observed test outcomes). Clearly, adaptive policies perform at least as well as non-adaptive ones.[2] However, non-adaptive policies are very simple to implement (requiring minimal incremental computation) and may be preferred in time-sensitive applications. Our main contributions are:

- an $O(\log m)$-approximation algorithm for non-adaptive ODT with noise.
- an $O(\min(h, r) + \log m)$-approximation algorithm for adaptive ODT with noise, where h (resp. r) is the maximum number of noisy outcomes for any hypothesis (resp. test).
- an $O(\log m)$-approximation algorithm for adaptive ODT with noise when every test has at least $m - O(\sqrt{m})$ noisy outcomes.
- experimental results on applications to toxic chemical identification and active learning.

We note that both non-adaptive and adaptive versions (even for usual ODT) generalize the set cover problem: so an $\Omega(\log m)$ approximation ratio is the best possible (unless P=NP).

Related Work The optimal decision tree problem (without noise) has been extensively studied for several decades [12, 21, 27, 25, 1, 2, 7, 19]. The state-of-the-art result [19] is an $O(\log m)$-approximation, for instances with arbitrary probability distribution and costs. It is also known that ODT cannot be approximated to a factor better than $O(\log m)$, unless P=NP [6].

The application of ODT to Bayesian active learning was formalized in [11]. There are also several results on the *statistical complexity* of active learning. e.g. [4, 20, 31], where the focus is on proving bounds for structured hypothesis classes. In contrast, we consider arbitrary hypothesis classes and obtain *computationally efficient* policies with provable approximation bounds relative to the optimal (instance specific) policy. This approach is similar to that in [11, 16, 13, 15, 10, 23].

The noisy ODT problem was studied previously in [15]. Using a connection to adaptive-submodularity [13], they obtained an $O(\log^2 \frac{1}{p_{min}})$-approximation algorithm for noisy ODT in the presence of very few noisy outcomes; here $p_{min} \leq \frac{1}{m}$ is the minimum probability of any hypothesis.[3] In particular, the running time of the algorithm in [15] is exponential in the number of noisy outcomes per hypothesis, which is polynomial only if this number is at most logarithmic in the number of hypotheses/tests. Our result provides the following improvements (i) the running time is polynomial irrespective of the number of noisy outcomes and (ii) the approximation ratio is better by at least one logarithmic factor. We note that a better $O(\log m)$ approximation ratio (still only for very sparse noise) follows from subsequent work on the "equivalence class determination" problem by [10]. For this setting, our result is also an $O(\log m)$ approximation, but the algorithm is simpler. More importantly, ours is the first result that can handle *any* number of noisy outcomes.

Other variants of noisy ODT have also been considered, e.g. [29, 5, 8], where the goal is to identify the correct hypothesis with at least some target probability. The theoretical results in [8] provide "bicriteria" approximation bounds where the algorithm has a larger error probability than the optimal policy. Our setting is different because we require *zero* probability of error.

Many algorithms for ODT (including ours) rely on some underlying submodularity properties. We briefly survey some background results. The basic submodular cover problem was first considered by

[2] There are also instances where the relative gap between the best adaptive and non-adaptive policies is $\tilde{\Omega}(m)$.

[3] The paper [15] states the approximation ratio as $O(\log 1/p_{min})$ because it relied on an erroneous claim in [13]. The correct approximation ratio, based on [30, 14], is $O(\log^2 1/p_{min})$.

[33], who proved that the natural greedy algorithm is a $(1 + \ln \frac{1}{\epsilon})$-approximation algorithm, where ϵ is the minimal positive marginal increment of the function. [3] obtained an $O(\log \frac{1}{\epsilon})$-approximation algorithm for the submodular ranking problem, that involves simultaneously covering multiple submodular functions; [22] extended this result to also handle costs. [24] studied an adaptive version of the submodular ranking problem. We utilize results/techniques from these papers.

Finally, we note that there is also work on minimizing the *worst-case* (instead of average case) cost in ODT and active learning [28, 32, 17, 18]. These results are incomparable to ours because we are interested in the average case, i.e. minimizing expected cost.

2 Problem Definition

We start with defining the optimal decision tree with noise (ODTN) formally. There is a set of m possible hypotheses with a probability distribution $\{\pi_x\}_{x=1}^m$, from which an unknown hypothesis $\bar{x}$ is drawn. There is also a set $U = [n]$ of binary tests. Each test $e \in U$ is associated with a 3-way partition $T^+(e), T^-(e), T^*(e)$ of the hypotheses, where the test outcome is (a) positive if $\bar{x}$ lies in $T^+(e)$, (b) negative if $\bar{x} \in T^-(e)$, and (c) positive or negative with probability $\frac{1}{2}$ each if $\bar{x} \in T^*(e)$ (these are noisy outcomes). We assume that conditioned on $\bar{x}$, each noisy outcome is independent. We also use $r_x(e)$ to denote the part of test e that hypothesis x lies in, i.e.

$$r_x(e) = \begin{cases} -1 & \text{if } x \in T^-(e) \\ +1 & \text{if } x \in T^+(e) \\ * & \text{if } x \in T^*(e) \end{cases}$$

While we know the 3-way partition $T^+(e), T^-(e), T^*(e)$ for each test $e \in U$ upfront, we are not aware of the actual outcomes for the noisy hypothesis-test pairs. It is assumed that the realized hypothesis $\bar{x}$ can be uniquely identified by performing all tests, regardless of the outcomes of $*$-tests. This means that for every pair $x, y \in [m]$ of hypotheses, there is some test $e \in U$ with $x \in T^+(e)$ and $y \in T^-(e)$ or vice-versa. The goal is to perform an adaptive (or non-adaptive) sequence of tests to identify hypothesis $\bar{x}$ using the minimum *expected* number of tests. Note that expectation is taken over both the prior distribution of $\bar{x}$ and the random outcomes of noisy tests for $\bar{x}$.

In our algorithms and analysis, it will be convenient to work with an **expanded set** of hypotheses M. For a binary vector $b \in \{\pm 1\}^U$ and hypothesis $x \in [m]$, we say b is consistent with x and denote $b \sim x$, if $b_e = r_x(e)$ for each $e \in U$ with $r_x(e) \neq *$. Let $M = \{(b, x) \in \{\pm 1\}^U \times [m] : b \sim x\}$, and $M_x \subseteq M$ be all copies associated with a particular hypothesis $x \in [m]$; note that $\{M_x\}_{x=1}^m$ is a partition of M. Each "expanded" hypothesis $(b, x) \in M$ corresponds to the case where the true hypothesis $\bar{x} = x$ and the test-outcomes are given by b. We assign the probability $q_{b,x} = \pi_x / 2^{h_x}$ to each $(b, x) \in M$, where h_x is the number of *-tests for x. Note that conditioned on $\bar{x} = x$, the probability of observing outcomes b is exactly 2^{-h_x}; so $\Pr[\bar{x} = x \text{ and test outcomes are } b] = q_{b,x}$. For any $(b, x) \in M$ and $e \in U$, define $r_{b,x}(e) = b(e)$ to be the observed outcome of test e if $\bar{x} = x$ and test-outcomes are b. For every expanded hypothesis $(b, x) \in M$ and test $e \in U$, define

$$T_{b,x}(e) = \begin{cases} T^+(e) & \text{if } r_{b,x}(e) = -1 \\ T^-(e) & \text{if } r_{b,x}(e) = +1 \end{cases}, \tag{1}$$

which is the subset of (original) hypotheses that can *definitely* be ruled-out based on test e if $\bar{x} = x$ and the test-outcomes are given by b. Note that hypotheses in $T^*(e)$ are never part of $T_{b,x}(e)$ as their outcome on test e can be positive/negative (so they cannot be ruled-out). For every hypothesis $(b, x) \in M$, define a monotone submodular function $f_{b,x} : 2^U \to [0, 1]$:

$$f_{b,x}(S) = |\bigcup_{e \in S} T_{b,x}(e)| \cdot \frac{1}{m-1}, \qquad \forall S \subseteq U, \tag{2}$$

which equals the fraction of the $m - 1$ hypotheses (excluding x) that have been ruled-out based on the tests in S if $\bar{x} = x$ and test-outcomes are given by b. Assuming $\bar{x} = x$ and test-outcomes are given by b, hypothesis x is uniquely identified after tests S if and only if $f_{b,x}(S) = 1$.

A **non-adaptive** policy is specified by just a permutation of tests. The policy performs tests in this sequence and eliminates incompatible hypotheses until there is a unique compatible hypothesis (which is $\bar{x}$). Note that the number of tests performed under such a policy is still random (depends

on $\bar{x}$ and outcomes of noisy tests). An **adaptive** policy chooses tests incrementally, depending on prior test outcomes. The *state* of a policy is a tuple (E, d) where $E \subseteq U$ is a subset of tests and $d \in \{\pm 1\}^E$ denotes the observed outcomes on tests in E. An adaptive policy is specified by a mapping $\Phi : 2^U \times \{\pm 1\}^U \to U$ from states to tests, where $\Phi(E, d)$ is the next test to perform at state (E, d). Equivalently, we can view a policy as decision tree with nodes corresponding to states, labels at nodes representing the test performed at that state and branches corresponding to the ± 1 outcome at the current test. As the number of states can be exponential, we cannot hope to specify arbitrary adaptive policies. Instead, we want implicit policies Φ, where given *any* state (E, d), the test $\Phi(E, d)$ can be computed *efficiently*. This would imply that the total time taken under any outcome is polynomial. We note that an optimal policy Φ^* can be very complex and the map $\Phi^*(E, d)$ may not be efficiently computable. We will still compare the performance of our (efficient) policy to Φ^*.

In this paper, we consider the **persistent noise** model. That is, repeating a test e with $\bar{x} \in T^*(e)$ always produces the same outcome. An alternative model is non-persistent noise, where each run of test e with $\bar{x} \in T^*(e)$ produces an independent random outcome. The persistent noise model is more appropriate to handle missing data. It also contains the non-persistent noise model as a special case (by introducing multiple tests with identical partitions). One can easily obtain an adaptive $O(\log^2 m)$-approximation for the non-persistent model using existing algorithms for noiseless ODT [7] and repeating each test $O(\log m)$ times. The persistent-noise model that we consider is much harder.

3 Non-Adaptive Algorithm

Our algorithm is based on a reduction to the submodular ranking problem [3], defined below.

Submodular Function Ranking (SFR) An instance of SFR consists of a ground set U of elements and a collection of monotone submodular functions $\{f_1, ..., f_m\}$, $f_x : 2^U \to [0, 1]$, with $f_x(\emptyset) = 0$ and $f_x(U) = 1$ for all $x \in [m]$. Additionally, there is a weight $w_x \geq 0$ for each $x \in [m]$. A solution is a permutation of the elements U. Given any permutation σ of U, the **cover time** of function f is $C(f, \sigma) := \min\{t \mid f(\cup_{i \in [t]} \sigma(i)) = 1\}$ where $\sigma(i)$ is the i^{th} element in σ. In words, it is the earliest time when the value of f reaches the unit threshold. The goal is to find a permutation σ of $[n]$ with minimal total cover time $\sum_{x \in [m]} w(x) \cdot C(f_x, \sigma)$. We will use the following result:

Theorem 3.1 ([3])**.** *There is an $O(\log \frac{1}{\epsilon})$-approximation for SFR where ϵ is minimum marginal increment of any function.*

The non-adaptive ODTN problem can be expressed as an instance of SFR as follows. The elements are the tests U. For each hypothesis-copy $(b, x) \in M$ there is a function $f_{b,x}$ (see (2)) with weight $q_{b,x}$. Based on the definition of these functions, the parameter $\epsilon = \frac{1}{m-1}$. To see the equivalence, note that a solution to non-adaptive ODTN is also a permutation σ of U and hypothesis x is uniquely identified under outcome (b, x) exactly when function $f_{b,x}$ has value one. Moreover, the objective of the ODTN problem is the expected number of tests in σ to identify the realized hypothesis $\bar{x}$, which equals

$$\sum_{x=1}^{m} \pi_x \sum_{b \sim x} 2^{-h_x} \cdot C_{b,x}(\sigma) = \sum_{(b,x) \in M} q_{b,x} \cdot C_{b,x}(\sigma),$$

where $C_{b,x}(\sigma)$ is the cover-time of function $f_{b,x}$. It now follows that this SFR instance is equivalent to the non-adaptive ODTN instance. However, we cannot apply Theorem 3.1 directly to obtain an $O(\log m)$ approximation. This is because we have an exponential number of functions (note $|M|$ can be exponential in m), which means that a direct implementation of the algorithm from [3] requires exponential time. Nevertheless, we show that a variant of the SFR algorithm can be used to obtain:

Theorem 3.2. *There is an $O(\log m)$-approximation for non-adaptive ODTN.*

The SFR algorithm [3] is a greedy-style algorithm that at any point, having already chosen tests E, assigns a score to each test $e \in U \setminus E$ of

$$G_E(e) := \sum_{(b,x) \in M : f_{b,x}(E) < 1} q_{b,x} \frac{f_{b,x}(\{e\} \cup E) - f_{b,x}(E)}{1 - f_{b,x}(E)} = \sum_{(b,x) \in M} q_{b,x} \cdot \Delta_E(b, x, e), \qquad (3)$$

$$\Delta_E(b,x,e) = \begin{cases} \frac{f_{b,x}(\{e\}\cup E) - f_{b,x}(E)}{1 - f_{b,x}(E)}, & \text{if } f_{b,x}(E) < 1; \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

where $\Delta_E(b,x,e)$ is the "gain" of test e for the hypothesis-copy b,x. At each step, the algorithm chooses the test of maximum score. However, we do not know how to compute the score (3) in polynomial time. Instead, using the fact that $G_E(e)$ is the expectation of $\Delta_E(b,x,e)$ over the hypothesis-copies $(b,x) \in M$, we will show that we can obtain an approximate maximizer by sampling. Moreover, Theorem 3.1 also holds when we choose a test with approximately maximum score: this follows directly from the analysis in [22]. This sampling approach is still not sufficient because it can fail when the value $G_E(e)$ is very small. A key observation is, when the score $G_E(e)$ is small for all tests e then it must be that, with high probability the already-performed tests E uniquely identify hypothesis $\bar{x}$. Hence the future tests won't affect the expected cover time by much.

As the realized hypothesis $\bar{x}$ can always be identified uniquely, for any pair $x, y \in [m]$ of hypotheses, there is a test where x and y have opposite outcomes (i.e. one is $+$ and the other $-$). So there is a set $\mathcal{L}$ of at most $\binom{m}{2}$ tests where hypothesis $\bar{x}$ will be uniquely identified by performing all the tests in $\mathcal{L}$.

The non-adaptive ODTN algorithm (Non-Adap) involves two phases. In the first phase, we run the SFR algorithm using sampling to get estimates $\bar{G_E}(e)$ of the scores $G_E(e)$ at each step; let $e^* = \arg\max_{e \in U} \bar{G_E}(e)$ denote the chosen test. If at some step, the maximum sampled score is less than m^{-5} then we go to the second phase where we perform all the tests in $\mathcal{L}$ and stop. The number of samples used to obtain each estimate is polynomial in m; so the overall runtime is polynomial.

We use the following sampling lemma (Chernoff bound).

Lemma 3.3. *Let X be a $[0,1]$ bounded random variable with $\mathbb{E}X \geq \Omega(m^{-5})$. Let $\bar{X}$ denote the average of m^6 many independent samples of X. Then $\Pr\left[\bar{X} \notin [\frac{1}{2}\mathbb{E}X, 2\mathbb{E}X]\right] \leq e^{-\Omega(m)}$.*

Proof. Let $X_1, ..., X_N$ be i.i.d. samples of random variable$X$ where $N = m^6$ is the number of samples. Letting $Y = \sum_{i \in [N]} X_i$, the usual Chernoff bound implies for any $\delta \in (0,1)$,

$$\Pr\left(Y \notin [(1-\delta)\mathbb{E}Y, (1+\delta)\mathbb{E}Y]\right) \leq \exp(-\frac{\delta^2}{2} \cdot \mathbb{E}Y).$$

Setting $\delta = \frac{1}{2}$ and using the assumption $\mathbb{E}Y = N \cdot \mathbb{E}X = \Omega(m)$, the lemma follows. □

Lemma 3.4. *Consider any step with $\bar{S} := \max_{e \in U} \bar{G_E}(e)$. If $\bar{S} \geq m^{-5}$ then $G_E(e^*) \geq \frac{S}{2}$.*

Proof. Let $S := \max_{e \in U} G_E(e)$. We first claim that $S \geq \frac{1}{2}m^{-5}$ w.h.p: otherwise, Lemma 3.3 would imply $\bar{S} < m^{-5}$ (we can artificially increase the mean to satisfy the "large mean" assumption in that lemma, which only makes proving the upper tail harder).

Consider first any $e \in U$ with $G_E(e) < S/4$. By Lemma 3.3, it follows that $\Pr[\bar{G_E}(e) > S/2] \leq e^{-\Omega(m)}$. Now consider the test e' with $G_E(e') = S$. Again, by Lemma 3.3, it follows that $\Pr[\bar{G_E}(e') \leq S/2] \leq e^{-\Omega(m)}$. This means that w.h.p., test e^* has $\bar{G_E}(e^*) \geq \bar{G_E}(e') > S/2$ and again by Lemma 3.3, $G_E(e^*) > S/4$. □

Lemma 3.5. *Consider the step in our algorithm with $\max_{e \in U} \bar{G_E}(e) < m^{-5}$. Then the probability that the realized hypothesis $\bar{x}$ is uniquely identified by the tests E is $1 - m^{-3}$.*

Proof. We prove the contrapositive. Suppose that the probability z of *not* identifying the realized hypothesis $\bar{x}$ after E is more than m^{-3}. Let $p_x(y) = \Pr_{b \sim x}[y \text{ not ruled out by } E | \bar{x} = x]$ denote the probability that when $\bar{x} = x$, hypothesis y is not ruled out after performing tests E. Note that

$$z = \sum_{x=1}^{m} \pi_x \cdot \Pr_{b \sim x}[E \text{ doesn't rule out } [m] \setminus x] \leq \sum_{x=1}^{m} \pi_x \sum_{y \in [m] \setminus x} p_x(y) \leq m \sum_{x=1}^{m} \left(\pi_x \cdot \max_{y \in [m] \setminus x} p_x(y)\right)$$

It follows that there is some $x \in [m]$ with $\pi_x \cdot \max_{y \in [m] \setminus x} p_x(y) \geq \frac{z}{m^2}$. So there is some $x, y \in [m]$ with $\pi_x \cdot p_x(y) \geq \frac{z}{m^2} \geq m^{-5}$, where we used the assumption that $z \geq m^{-3}$. Recall that there is some test e^* that separates x and y deterministically. Let $B' = \{b \sim x : y \notin \cup T_{b,x}(e)\}$. Note that

$\sum_{b\in B'} q_{b,x} = \pi_x \cdot p_x(y)$. For any $b \in B'$ we have (i) $y \notin \cup T_{b,x}(e)$ $f_{b,x}(E)$ and (ii) $y \in T_{b,x}(e^*)$ (this is true for *all* $b \sim x$). Therefore $\Delta_E(b, x, e^*) \geq \frac{1}{m-1}$ for all $b \in B'$, which implies:

$$g_E(e^*) \geq \sum_{b\sim x} q_{b,x}\Delta_E(b, x, e^*) \geq \sum_{b\in B'} q_{b,x}\frac{1}{m-1} = \frac{\pi_x \cdot p_x(y)}{m-1} \geq m^{-5},$$

as desired. □

Proof of Theorem 3.2. We bound the expected costs (number of tests) from phase 1 and 2 separately. By Lemma 3.4, the test chosen in each step of phase 1 is a 2-approximate maximizer (w.h.p.) of the score used in the ASR algorithm. So the expected cost in phase 1 is at most $O(\log m)$ times the optimum. By Lemma 3.5, the probability of running phase 2 is at most m^{-3}. As there are $|\mathcal{L}| \leq m^2$ tests in phase 2, the expected cost is $o(1)$. So we obtain an $O(\log m)$-approximation algorithm.

4 Adaptive Algorithms

Our adaptive algorithm maintains the posterior probability of each hypothesis based on the previous test outcomes, and uses these probabilities to calculate a "score" for each test. The score of a test has two components (i) a term that prioritizes splitting the candidate hypotheses in a balanced way and (ii) terms that correspond to the expected number of hypotheses eliminated. We maintain the following information at each point in the algorithm: already performed tests $E \subseteq U$, compatible hypotheses $H \subseteq [m]$ and (posterior) probability p_x for each $x \in H$. Given values $\{p_x : x \in [m]\}$, to reduce notation we use the shorthand $p(S) = \sum_{x\in S} p_x$ for any subset $S \subseteq [m]$.

Algorithm 1 ODTN$_r$

initially $E \leftarrow \emptyset$, $H \leftarrow [m]$ and $p_x \leftarrow \pi_x$ for all $x \in [m]$.
while $|H| > 1$ **do**
  for any test $e \in U$, let $L_e(H)$ be the smaller cardinality set among $T^+(e) \cap H$ and $T^-(e) \cap H$
  select test $e \in U \setminus E$ that maximizes:

$$p\left(L_e(H)\right) + \frac{|T^-(e)\cap H|}{|H|-1}\cdot p\left(T^+(e)\cap H\right) + \frac{|T^+(e)\cap H|}{|H|-1}\cdot p\left(T^-(e)\cap H\right) + \frac{1}{2}\frac{|H\setminus T^*(e)|}{|H|-1}\cdot p\left(T^*(e)\cap H\right). \tag{5}$$

  if outcome of test e is $+$ **then**
    update $H \leftarrow H \setminus T^-(e)$
  else
    update $H \leftarrow H \setminus T^+(e)$
  end if
  $E \leftarrow E \cup \{e\}$
  for $x \in H$ **do**
    if $r_x(e) = *$ **then**
      $p_x \leftarrow p_x/2$
    end if
  end for
end while

Theorem 4.1. *Algorithm 1 is an $\mathcal{O}(r + \log m)$-approximation algorithm for adaptive ODTN, where r is the maximum number of noisy outcomes per test.*

The high-level idea is to view any ODT instance $\mathcal{I}$ as a suitable instance $\mathcal{J}$ of adaptive submodular ranking (ASR). Then we will use an existing framework of analysis of ASR from [24].

An equivalent ASR instance $\mathcal{J}$. This involves the expanded hypothesis set M where each hypothesis $(b, x) \in M$ occurs with probability $q_{b,x} = \pi_x/2^{h_x}$. Each hypothesis (b, x) is also associated with: (i) submodular function $f_{b,x} : 2^U \to [0, 1]$ and (ii) feedback function $r_{b,x} : U \to \{+, -\}$ where $r_{b,x}(e)$ is the outcome of test e under hypothesis (b, x). The goal in the ASR instance is to adaptively select a subset $S \subseteq U$ such that the value $f_{b,x}(S) = 1$ for the realized hypothesis (b, x). The objective is to minimize the expected cost $\mathbb{E}[|S|]$.

Lemma 4.2. *The ASR instance $\mathcal{J}$ is equivalent to ODT instance $\mathcal{I}$.*

Proof. We will show that any feasible decision tree for $\mathcal{J}$ (resp. $\mathcal{I}$) is also feasible for instance $\mathcal{I}$ (resp. $\mathcal{I}$) with the same objective. In one direction, let $\mathcal{T}$ be a decision tree for $\mathcal{J}$. For any hypothesis $(b, x) \in M$ let $P_{b,x}$ denote the unique path traced in $\mathcal{T}$ and let $S_{b,x}$ denote the tests performed. Then we have $f_{b,x}(S_{b,x}) = 1$ which means $\bigcup_{e \in S_{b,x}} T_{b,x}(e) = [m] \setminus x$. Now consider $\mathcal{T}$ as a decision tree for the ODT instance $\mathcal{I}$. *Condition* on hypothesis $x \in [m]$ and outcomes b on the $*$-tests for x, which occurs with probability $q_{b,x} = \pi_x / 2^{h_x}$. Then, it is clear that the feedback from any test e is $r_{b,x}(e)$ and so the path traced in $\mathcal{T}$ is just $P_{b,x}$. Moreover, the set of incompatible hypotheses based on test e is $T_{b,x}(e)$. So the set of incompatible hypotheses at the end of $P_{b,x}$ is $\bigcup_{e \in S_{b,x}} T_{b,x}(e) = [m] \setminus x$, which means x is identified. Taking an expectation over all x and b, it follows that $\mathcal{T}$ is a feasible decision tree for $\mathcal{I}$ with cost at most that for instance $\mathcal{J}$.

In the other direction, let $\mathcal{T}'$ be any decision tree for instance $\mathcal{I}$. Again *condition* on hypothesis $x \in [m]$ and outcomes b on the $*$-tests for x (with probability $q_{b,x}$). Then a unique path $P'_{b,x}$ is traced in $\mathcal{T}'$, and let $S'_{b,x}$ denote the tests on this path. As before, the set of incompatible hypotheses at the end of $P'_{b,x}$ is $\bigcup_{e \in S'_{b,x}} T_{b,x}(e) = [m] \setminus x$ because x is identified. Now consider $\mathcal{T}'$ as a decition tree for ASR instance $\mathcal{J}$. Under hypothesis b, x, it is clear that path $P'_{b,x}$ is traced and so tests $S'_{b,x}$ are selected. It follows that $f_{b,x}(S'_{b,x}) = 1$ which means that hypothesis b, x is covered at the end of $P'_{b,x}$. So $\mathcal{T}'$ is a feasible decision tree for $\mathcal{J}$. Taking expectations, the cost for $\mathcal{J}$ is at most that for instance $\mathcal{I}$. □

Now, we present an algorithm for the ASR instance $\mathcal{J}$ that we will show is equivalent to running Algorithm 1 on instance $\mathcal{I}$. Crucially, the ASR algorithm is almost identical to that studied in prior work [24] and therefore we can essentially re-use the analysis from that paper to prove our bound. Recall that the expanded hypotheses $M = \cup_{x=1}^{m} M_x$ where M_x are all copies of hypothesis $x \in [m]$. To reduce notation, we use $q(S) = \sum_{(b,x) \in S} q_{b,x}$ for any subset $S \subseteq M$. Also note that hypothesis (b, x) is covered when $f_{b,x}(E) = 1$ which implies identifying hypothesis $x \in [m]$.

Algorithm 2 Algorithm for ASR instance $\mathcal{J}$.

initially $E \leftarrow \emptyset, H' \leftarrow M$.
while $H' \neq \emptyset$ **do**
  $H \leftarrow \{x \in [m] \,:\, M_x \cap H' \neq \emptyset\}$.
  for any test $e \in U$, let $L'_e(H') = \{(b, x) \in H' : x \in L_e(H)\} = H' \cap \left(\cup_{x \in L_e(H)} M_x\right)$
  select test $e \in U \setminus E$ that maximizes:

$$q\left(L'_e(H')\right) \;+\; \sum_{(b,x) \in M_x \cap H'} q_{b,x} \cdot \frac{f_{b,x}(e \cup E) - f_{b,x}(E)}{1 - f_{b,x}(E)}. \tag{6}$$

  remove incompatible and covered hypotheses from H' based on the feedback from e.
  $E \leftarrow E \cup \{e\}$
end while

We now prove the equivalence between Algorithms 1 and 2. The *state* of either algorithm is represented by the set E of tests performed along with their outcomes.

Lemma 4.3. *If Algorithms 1 and 2 are at the same state (i.e. performed the same set of tests E and observed the same outcomes) then the set $H \subseteq [m]$ in the two algorithms is identical.*

Proof. Let H_1 and H_2 denote the set H in Algorithms 1 and 2 respectively. We will show that $H_1 = H_2$. Consider any $x \in H_1$. We must have observed $r_x(e)$ on every test $\{e \in E : r_x(e) \neq *\}$. As there is a scenario-copy corresponding to *every* outcome on the $*$-tests for x, we will have some (b, x) that is compatible with the observations on all tests in E. This means $(b, x) \in H'$ and so $x \in H_2$. On the other hand, consider $x \in H_2$, i.e. there is some $(b, x) \in H'$. Then we must have observed $r_{b,x}(e)$ on every test $e \in E$. In particular, we must have observed $r_x(e)$ on each $\{e \in E : r_x(e) \neq *\}$, which means that $x \in H_1$. □

Lemma 4.4. *If Algorithms 1 and 2 are at the same state (i.e. performed the same set of tests E and observed the same outcomes) then $p_x = \sum_{(b,x)\in H'\cap M_x} q_{b,x}$ for all $x \in H$.*

Proof. This can be proved by induction. Note that at the beginning we clearly have $p_x = \pi_x = \sum_{(b,x)\in M_x} q_{b,x}$ for all $x \in [m]$. Consider any state in the two algorithms when we have performed tests E and observed the same outcomes. Let $H \subseteq [m]$ denote the compatible hypotheses at this state (which is the same in both algorithms by Lemma 4.3) and e be the next test performed under both algorithms. Let $\bar{H} \subseteq H$ and $\bar{H}' \subseteq H'$ denote the compatible hypotheses after this test. We will prove inductively the expression for p_x for any $x \in \bar{H}$ by considering different cases for $r_x(e)$.

If $r_x(e) \neq *$ and the outcome of e is different from $r_x(e)$ then $x \notin \bar{H}$ and there is nothing to prove.

If $r_x(e) \neq *$ and the outcome of e is $r_x(e)$ then $x \in \bar{H}$ and $r_{b,x}(e) = r_x(e)$ for all copies (b, x) of x. So $M_x \cap H' = M_x \cap \bar{H}'$ and the expression for p_x remains the same.

If $r_x(e) = *$ then p_x decreases by a factor of 2 in Algorithm 1. By definition of the copies M_x of hypothesis x, it is clear that exactly half the copies $(b, x) \in M_x \cap H'$ have $r_{b,x}(e) = +$ and the rest have $r_{b,x}(e) = -$. So, irrespective of the observed outcome of e, we have $|M_x \cap \bar{H}'| = \frac{1}{2}|M_x \cap H|$. This implies $q\left(\bar{H}' \cap M_x\right) = \frac{\pi_x}{2^{h_x}}|\bar{H}' \cap M_x| = \frac{1}{2}q\left(\bar{H} \cap M_x\right) = p_x$ (after the update). □

Lemma 4.5. *Consider any state (E, H') of Algorithm 2 and $(b, x) \in H'$. The following are true:*

1. $\bigcup_{d\in E} T_{b,x}(d) = [m] \setminus H$. *So* $f_{b,x}(E) = \frac{m-|H|}{m-1}$.
2. *For any* $e \in U$, $f_{b,x}(E \cup e) - f_{b,x}(E) = \frac{|H\cap T_{b,x}(e)|}{m-1}$.
3. *If* $x \in H \cap T^+(e)$ *then* $f_{b,x}(E \cup e) - f_{b,x}(E) = \frac{|H\cap T^-(e)|}{m-1}$.
4. *If* $x \in H \cap T^-(e)$ *then* $f_{b,x}(E \cup e) - f_{b,x}(E) = \frac{|H\cap T^+(e)|}{m-1}$.
5. *If* $x \in H \cap T^*(e)$ *then*

$$\sum_{(b,x)\in H'_x} q_{b,x}\left(f_{b,x}(E \cup e) - f_{b,x}(E)\right) = \frac{1}{2}\sum_{(b,x)\in H'_x} q_{b,x}\left(\frac{|H \cap T^-(e)|}{m-1} + \frac{|H \cap T^+(e)|}{m-1}\right),$$

where $H'_x = M_x \cap H'$.

Proof. (1) As $(b, x) \in H'$, the outcome of each test $d \in E$ must have been $r_{b,x}(d)$. By definition, $T_{b,x}(d)$ consists of all hypotheses $y \in [m]$ with $r_y(d) \neq *$ and $r_y(d) \neq r_{b,x}(d)$. So $\bigcup_{d\in E} T_{b,x}(d) \subseteq [m]$ is precisely the set of incompatible hypotheses at this state. In other words, $\bigcup_{f\in E} T_{b,x}(f) = [m] \setminus H$ and $f_{b,x}(E) = \frac{|\bigcup_{d\in E} T_{b,x}(d)|}{m-1} = \frac{m-|H|}{m-1}$.

(2) The set of hypotheses in H that are compatible with (b, x) after test e are $H \setminus T_{b,x}(e)$. So based on (1) we can write:

$$f_{b,x}(E\cup e) - f_{b,x}(E) = \frac{m - |H \setminus T_{b,x}(e)|}{m-1} - \frac{m-|H|}{m-1} = \frac{|H \setminus T_{b,x}(e)| - |H|}{m-1} = \frac{|H \cap T_{b,x}(e)|}{m-1}.$$

(3-4) These follow directly from (2) using the definition of $T_{b,x}(e)$.

(5) It is easy to see (as observed before) that half the scenario-copies $b, x \in H' \cap M_x = H'_x$ have $r_{b,x}(e) = +$ (which implies $T_{b,x}(e) = T^-(e)$) and the rest have $r_{b,x}(e) = -$ (which implies $T_{b,x}(e) = T^+(e)$). So using (2),

$$\sum_{(b,x)\in H'_x} q_{b,x}\left(f_{b,x}(E \cup e) - f_{b,x}(E)\right) = \sum_{\substack{(b,x)\in H'_x \\ b(e)=-}} q_{b,x}\left(\frac{|H \cap T^+(e)|}{m-1}\right) + \sum_{\substack{(b,x)\in H'_x \\ b(e)=+}} q_{b,x}\left(\frac{|H \cap T^-(e)|}{m-1}\right),$$

which equals the claimed expression as exactly half the copies in H'_x have $b(e) = +$. □

Lemma 4.6. *The decision tree produced by Algorithm 1 on $\mathcal{I}$ is the same as that produced by Algorithm 2 on $\mathcal{J}$.*

Proof. Consider any state which is common to both algorithms, with E being the already chosen tests, $H \subseteq [m]$ the compatible hypotheses and $H' \subseteq M$ the compatible uncovered scenario-copies. We will prove:

1. The score of each test e is the same for both algorithms (at this state). Therefore the same test is chosen in both algorithms.
2. The stopping criteria for both algorithms is the same, i.e. Algorithm 1 stops at this state iff Algorithm 2 stops.

This clearly suffices to prove the lemma.

Proving 1: We need to show that (5) and (6) are equal. Consider any $x \in H$. By Lemma 4.4, we have $p_x = \sum_{(b,x)\in H'_x} q_{b,x}$. Now using Lemma 4.5, we have the following three cases:

- If $x \in H \cap T^+(e)$ then
$$\sum_{(b,x)\in H'_x} q_{b,x} \left(\frac{f_{b,x}(E \cup e) - f_{b,x}(E)}{1 - f_{b,x}(E)} \right) = p_x \cdot \frac{|H \cap T^-(e)|}{|H| - 1}.$$
- If $x \in H \cap T^-(e)$ then
$$\sum_{(b,x)\in H'_x} q_{b,x} \left(\frac{f_{b,x}(E \cup e) - f_{b,x}(E)}{1 - f_{b,x}(E)} \right) = p_x \cdot \frac{|H \cap T^+(e)|}{|H| - 1}.$$
- If $x \in H \cap T^*(e)$ then
$$\sum_{(b,x)\in H'_x} q_{b,x} \left(\frac{f_{b,x}(E \cup e) - f_{b,x}(E)}{1 - f_{b,x}(E)} \right) = \frac{p_x}{2} \cdot \frac{|H \cap T^-(e)| + |H \cap T^+(e)|}{|H| - 1}.$$

Summing over all $x \in H$, it follows that the second term in (6) equals the sum of the last three terms in (5). We now show that the first term in (6) equals the first term in (5):

$$\sum_{x\in L_e(H)} p_x = \sum_{x\in L_e(H)} \Big(\sum_{(b,x)\in H'} q_{b,x} \Big) = \sum_{(b,x)\in L'_e(H')} q_{b,x} = q\left(L'_e(H')\right),$$

where the first equality is by Lemma 4.4 and the second equality is by definition of $L_e(H)$ and $L'_e(H)$. It now follows that the score of each test at every state is equal in both algorithms.

Proving 2: We show that both algorithms stop at the same states. Algorithm 1 stops when $|H| = 1$ and Algorithm 2 stops when $H' = \emptyset$. Note that scenario-copy (b, x) is covered exactly when $H = \{x\}$. As H' consists of all compatible uncovered scenario-copies, it follows that $|H| = 1$ iff $H' = \emptyset$. □

Approximation ratio for Algorithm 2. We will now prove

Theorem 4.7. *Algorithm 2 is an $O(r + \log m)$-approximation algorithm for ASR instance $\mathcal{J}$.*

The proof is very similar to the analysis in [24]. So we only provide an outline of the overall proof, while emphasizing the differences. For $k = 0, 1, \cdots$, define the following quantities:

- $A_k \subseteq M$ is the set of uncovered hypotheses in ALG at time $L \cdot 2^k$, and $a_k = q(A_k)$.
- Y_k is the set of uncovered hypotheses in OPT at time 2^{k-1}, and $y_k = q(Y_k)$.

Here $L = O(r + \log m)$. The key step is to show:

$$a_k \leq 0.2a_{k-1} + 3y_k, \qquad \text{for all } k \geq 1. \tag{7}$$

As shown in [24], this implies Theorem 4.7.In order to prove (7) we use the quantity:

$$Z := \sum_{t>L2^{k-1}}^{L2^k} \sum_{(E,H')\in R(t)} \max_{e\in U\setminus E} \left(\sum_{(b,x)\in L'_e(H')} q_{b,x} \; + \sum_{(b,x)\in H'} q_{b,x} \cdot \frac{f_{b,x}(e\cup E) - f_{b,x}(E)}{1 - f_{b,x}(E)} \right) \quad (8)$$

Above, $R(t)$ denotes the set of states (E, H') that occur at time t in ALG. (7) will be proved by separately lower and upper bounding Z.

Lemma 4.8 ([24]). *We have $Z \geq L \cdot (a_k - 3y_k)/3$.*

The proof of this lower bound is identical to that in [24], although the definition of $L'_e(H')$ is different here. The proof of the upper bound however requires new ideas, as described next.

Lemma 4.9. *We have $Z \leq a_{k-1} \cdot (1 + \ln m + r + \log m)$.*

Proof. For any hypothesis $(b, x) \in A_{k-1}$ (i.e. uncovered in ALG by time $L2^{k-1}$) let $\sigma_{b,x}$ be the path traced by (b, x) in ALG's decision tree, starting from time $2^{k-1}L$ and ending at 2^kL or when (b, x) gets covered. Recall that for any $L2^{k-1} < t \leq L2^k$, any hypothesis in H' for any state in $R(t)$ appears in A_{k-1}. So only hypotheses in A_{k-1} can contribute to Z and we rewrite (8) as:

$$\begin{aligned} Z \quad &= \sum_{(b,x)\in A_{k-1}} q_{b,x} \cdot \sum_{e\in\sigma_{b,x}} \left(\frac{f_{b,x}(e\cup E) - f_{b,x}(E)}{1 - f_{b,x}(E)} + \mathbb{1}[(b,x) \in L'_e(H')] \right) \\ &\leq \sum_{(b,x)\in A_{k-1}} q_{b,x} \cdot \left(\sum_{e\in\sigma_{b,x}} \frac{f_{b,x}(e\cup E) - f_{b,x}(E)}{1 - f_{b,x}(E)} \; + \sum_{e\in\sigma_{b,x}} \mathbb{1}[(b,x) \in L'_e(H')] \right) \quad (9) \end{aligned}$$

Above, for any $e \in \sigma_{b,x}$ we use (E, H') to denote the state at which e is selected.

Fix any hypothesis $(b, x) \in A_{k-1}$. For the first term, we use Lemma 4.10 below and the definition of ϵ. This implies $\sum_{e\in\sigma_{b,x}} \frac{f_{b,x}(e\cup E) - f_{b,x}(E)}{1 - f_{b,x}(E)} \; \leq \; 1 + \ln \frac{1}{\epsilon} \leq 1 + \ln m$ as parameter $\epsilon \geq 1/m$ for $f_{b,x}$.

Now, we bound the second term by proving the inequality below:

$$\sum_{e\in\sigma_{b,x}} \mathbb{1}[(b,x) \in L'_e(H)] \leq r + \log m \quad (10)$$

To prove this inequality, consider hypotheses in H'. Now, if hypothesis $(b, x) \in L'_e(H)$ when ALG selects test e, then x would be in $L_e(H)$. Suppose $L_e(H) = T^+(e) \cap H$; the other case is identical. Let $D_e(H) = T^-(e) \cap H$ and $S_e(H) = T^*(e) \cap H$. As $x \in L_e(H)$, it must be that path $\sigma_{b,x}$ follows the + branch out of e. Also, the number of candidate hypotheses on this path after test e is

$$|L_e(H)| + |S_e(H)| \leq \frac{|L_e(H)|}{2} + \frac{|D_e(H)|}{2} + |S_e(H)| = \frac{|H|}{2} + \frac{|S_e(H)|}{2} \leq \frac{|H|}{2} + \frac{r}{2}.$$

The first inequality uses the definition of $L_e(H)$ and the last inequality uses the bound of r on the number of hypotheses with $*$ outcomes. Hence, each time that $(b, x) \in L'_e(H)$ along path $\sigma_{b,x}$, the number of candidate hypotheses changes as $|H_{new}| \leq \frac{1}{2}|H_{old}| + \frac{r}{2}$. This implies that after $\log_2 m$ such events, $|H| \leq r$. Let $\sigma'_{b,x}$ denote the portion of path $\sigma_{b,x}$ after $|H|$ drops below r. Note that each time $(b, x) \in L'_e(H)$ we have $L_e(H) \neq \emptyset$: so $|H|$ reduces by at least one after each such test e. Hence $\sum_{e\in\sigma'_{b,x}} \mathbb{1}[(b, x) \in L'_e(H)] \leq r$. As the portion of path $\sigma_{b,x}$ until $|H| \leq r$ contributes at most $\log_2 m$ to the left-hand-side in (10), the total is at most $r + \log_2 m$ as needed. □

Lemma 4.10. *Let $f : 2^U \to [0, 1]$ be any monotone function with $f(\emptyset) = 0$ and $\epsilon = \min\{f(S \cup \{e\}) - f(S) \; : \; e \in U, S \subseteq U, f(S \cup \{e\}) - f(S) > 0\}$. Then, for any sequence $\emptyset = S_0 \subseteq S_1 \subseteq \cdots S_k \subseteq U$ of subsets, we have $\sum_{t=1}^{k} \frac{f(S_t) - f(S_{t-1})}{1 - f(S_{t-1})} \; \leq \; 1 + \ln \frac{1}{\epsilon}$.*

Setting $L = 15(1 + \ln m + r + \log_2 m)$ and applying Lemmas 4.8 and 4.9 completes the proof of (7) and hence Theorem 4.7.

Tight Example: our analysis above is tight, as shown by the following instance with $r = m$ where the algorithm's cost is $\Omega(m)$ times the optimum. The instance has $m = 6k$ scenarios, which are partitioned into $A = \{a_1, \cdots a_{3k}\}$, $B = \{b_1, \cdots, b_{3k-3}\}$ and $\{c, c', c''\}$. The probability of hypothesis c is $1-\epsilon$ and each of the other hypotheses has probability $\frac{\epsilon}{6k-1}$. We will use $\epsilon = k^{-3} \to 0$. We also have four types of tests (unspecified hypotheses have $*$ outcomes).

- α-tests: for each $j \in [k]$, test α_j is $+$ on $\{c, c'\}$ and $-$ on $\{a_{3j-2}, a_{3j-1}, a_{3j}\}$.
- β-tests: for each $j \in [k-1]$, test β_j is $+$ on $\{c, c'\}$ and $-$ on $\{b_{3j-2}, b_{3j-1}, b_{3j}\}$.
- γ-tests: for each pair $s, t \neq c$ of hypotheses there is a test that is $+$ on s and $-$ on t.
- Test δ is $+$ on A and $-$ on all other scenarios. Test δ' is $+$ on $\{c', c''\} \cup B$ and $-$ on all other scenarios.

Bound on the Optimal Cost: We first perform tests δ and δ'. If both outcomes are $-$ then we identify hypothesis c uniquely (which happens with probability $1-\epsilon$). Otherwise, we continue to perform all the γ-tests which suffices to identify the realized hypothesis (this happens only with probability ϵ). The expected cost is at most $2 + \epsilon \cdot m^2 = \mathcal{O}(1)$ using $\epsilon = 1/k^3$.

Cost of our Algorithm: We will only describe the sequence of tests under realized hypothesis c, which will provide a lower bound on the algorithm's cost. We claim that the algorithm will select tests $\alpha_1, \beta_1, \alpha_2, \beta_2, \cdots \alpha_{k-1}, \beta_{k-1}$ by suitable tie-breaking. We show this by induction. Consider the candidate hypotheses H at any point in this sequence. Note that the alternating choice of α and β tests implies that we will always have $c, c', c'' \in H$ and either $|H \cap A| = 3 + |H \cap B|$ or $|H \cap A| = |H \cap B|$. In either case, the "lighter" side of test δ is always $H \cap A$ (possibly by breaking ties) and the lighter side of test δ' is $H \cap (B \cup \{c', c''\})$. In particular, $c \notin L_e(H)$ for test $e \in \{\delta, \delta'\}$. The score of $e \in \{\delta, \delta'\}$ in Equation (5) is:

$$p(L_e(H)) \;+\; \frac{|T^+(e) \cap H|}{|H| - 1} \cdot p(T^-(e) \cap H) \;+\; \frac{|T^-(e) \cap H|}{|H| - 1} \cdot p(T^+(e) \cap H) \quad \leq \quad \epsilon + \frac{1}{2} + \epsilon.$$

Above we used the fact that hypothesis c does not lie in the lighter side and it has probability $1-\epsilon$ (so the total remaining probability is at most ϵ). On the other hand, the score of all remaining α and β tests will be equal (by symmetry) and has value at least $1-\epsilon$ as c lies in the lighter side of these tests. Finally, all γ-tests have a score of at most $\epsilon + \frac{1}{2}$. So the algorithm can choose any remaining α or β test at this point, proving the inductive step. Thus the expected cost of our algorithm is at least $(1-\epsilon)(2k-2) = \Omega(m)$.

4.1 $\mathcal{O}(h + \log m)$-Approximation Algorithm

Here we observe that directly applying the ASR algorithm from [24] on instance $\mathcal{J}$ leads to this bound. This corresponds to changing $L'_e(H')$ in the score (6) to the smaller of the following sets:

$$\{(b, x) \in H' : r_{b,x}(e) = +\} \text{ and } \{(b, x) \in H' : r_{b,x}(e) = -\},$$

which corresponds to the smaller-cardinality part of H'.

Theorem 4.11. *There is an $\mathcal{O}(h + \log m)$-approximation algorithm for adaptive ODTN, where h is the maximum number of noisy outcomes for a scenario.*

Proof. Consider the ASR instance $\mathcal{J}$. By applying the ASR algorithm, we obtain an $O(\log M + \log 1/\epsilon) = O(h + \log m)$ approximation ratio, because $M \leq 2^h \cdot m$ and $\epsilon \geq 1/m$. □

Corollary 4.11.1. *There is an $\mathcal{O}(\min(h, r) + \log m)$-approximation algorithm for adaptive ODTN, where h is the maximum number of noisy outcomes per scenario, and r is the maximum number of noisy outcomes per test.*

We note that our analysis is tight (up to constant factors).

5 Adaptive Algorithm with Very Few Known Outcomes

In this section we consider instances with a very large number of noisy outcomes, and obtain an approximation algorithm that relies on different ideas. Formally, we define an α**-sparse** ($\alpha \leq 1/2$) instance as follows. There is a constant C such that $\max\{|T^+(e)|, |T^-(e)|\} \leq C \cdot m^\alpha$ for all tests $e \in U$. Our main result here is:

Theorem 5.1. *There is an adaptive $O(\log m)$-approximation algorithm for ODT with noise on any $\alpha \leq \frac{1}{2}$ sparse instance.*

We first make some simple observations related to α-sparse instances.

Proposition 1. *The optimal value $OPT \geq \Omega(m^{1-\alpha})$.*

Proof. By definition of α-sparse instances, the maximum number of candidate hypotheses that can be eliminated after performing a single test is m^α. As we need to eliminate $m-1$ hypotheses irrespective of the realized hypothesis $\bar{x}$, we need to perform at least $\frac{m-1}{m^\alpha} = \Omega(m^{1-\alpha})$ tests under every $\bar{x}$. □

Proposition 2. *Consider any $W \subseteq U$ and $X \subseteq [m]$. For any $y \in X$, $\kappa(y) = |\{e \in W : r_y(e) \neq *\}|$ denotes the number of tests in W for which y has a ± 1 outcome. Then, the number of hypotheses in X with $\kappa(y) > |W|/2$ is at most $2Cm^\alpha$.*

Proof. Let $X' = \{y \in X : \kappa(y) > |W|/2\}$. Then:

$$|X'| \cdot \frac{|W|}{2} < \sum_{y \in X} \kappa(y) = \sum_{e \in W} |\{y \in X : r_y(e) \neq *\}| \leq |W| \cdot Cm^\alpha.$$

Rearraging, we get $|X'| \leq 2Cm^\alpha$ as needed. □

Main Idea. Consider a decision tree obtained by the following naive algorithm: when A is the current subset of "alive" scenarios, we choose T s.t. $|T^+ \cap A| + |T^- \cap A|$ is maximized. In general, this tree can have very high cost compared to OPT, so we truncate it to reduce the cost to $O(\log m \cdot OPT)$. Meanwhile, our truncated tree is guaranteed to rule out all but a "small" number ($O(\sqrt{m})$) of scenarios, hence it won't be too costly to perform a "brute-force" search on the remaining ones.

5.1 Relation to Stochastic Set Cover

We now establish a crucial relation to the stochastic set cover (SSC) problem [26, 22] and also state a useful result on SSC. This forms the basis of our algorithm.

An instance of SSC consists of a groundset V and k stochastic sets $S_1, \cdots S_k$ each of which is a subset of V. The distribution of each set S_i is known to the algorithm and the distributions of different sets are independent of each other. The instantiation of each set is only known after it is selected. The goal is to find an adaptive policy that minimizes the expected number of sets to cover V. A natural adaptive greedy algorithm is known to be an $O(\log m)$-approximation where $m = |V|$ [26]. At any point in this policy, if $A \subseteq V$ denotes the uncovered elements then we choose the set S_{i^*} that maximizes the expected number of new elements covered, i.e. $i^* = \arg\max_{i=1}^k \mathbb{E}[|S_i \cap A|]$. At any such step, we call a set β-greedy if it has expected coverage at least a $1/\beta$ fraction of the maximum. We need an extension of this result that involves picking β-greedy sets at just some constant fraction of the steps. Formally, call an SSC policy (β, ρ) greedy if for every $t \geq 1$, at least t/ρ steps among the first t steps involve picking a β-greedy set. By modifying the analysis in [22] slightly, we obtain:

Theorem 5.2. *Any (β, ρ) greedy policy for stochastic set cover is an $O(\beta\rho \log m)$-approximation.*

We now derive a lower bound on the optimal ODTN value in terms of certain SSC instances. For any $x \in [m]$, let $SSC(x)$ denote the stochastic set cover instance with groundset $V = [m] \setminus \{x\}$ (i.e. all hypotheses other than x) and sets U (i.e. all tests) where

$$S_e(x) = \begin{cases} T^+(e) \text{ with prob. } 1 & \text{if } x \in T^-(e) \\ T^-(e) \text{ with prob. } 1 & \text{if } x \in T^+(e) \\ T^-(e) \text{ or } T^+(e) \text{ with prob. } \frac{1}{2} \text{ each} & \text{if } x \in T^*(e) \end{cases}, \qquad \forall e \in U.$$

Lemma 5.3. $OPT \geq \sum_{x\in[m]} \pi_x \cdot OPT_{SSC(x)}$.

Proof. Consider any feasible decision tree $\mathcal{T}$ for the ODTN instance and any hypothesis $x \in [m]$. If we *condition* on $\bar{x} = x$ then $\mathcal{T}$ corresponds to a feasible adaptive policy for $SSC(x)$. This is because (i) for any outcome-vector $b \sim x$, the tests performed in $\mathcal{T}$ must rule-out all the hypotheses $[m]\setminus x$ and (ii) the hypotheses ruled-out by any test e (conditioned on $\bar{x} = x$) is a random subset that has the same distribution as $S_e(x)$. Formally, if $P_{b,x}$ denotes the path traced in $\mathcal{T}$ under test outcomes $b \sim x$ then this policy for $SSC(x)$ has cost $\sum_{b\sim x} 2^{-h_x} \cdot |P_{b,x}|$ as $\Pr[\text{observing outcomes } b|\bar{x} = x] = 2^{-h_x}$. So $OPT_{SSC(x)} \leq \sum_{b\sim x} 2^{-h_x} \cdot |P_{b,x}|$. Taking expectations over $x \in [m]$ the lemma follows. □

5.2 A Low-Cost Membership Oracle

One subroutine in our algorithm is an "oracle" called "Member", which takes a (small) subset $Z \subseteq [m]$ as input, and decides whether $\bar{x} \in Z$.

1. Initialize $Z' \leftarrow Z$.
2. While $|Z'| \geq 2$, do:
   i. Perform any test $e \in U$ with both $T^+(e) \cap Z', T^-(e) \cap Z' \neq \emptyset$,
   ii. Let R be the set of scenarios ruled out, let $Z' \leftarrow Z'\setminus R$.
3. When $|Z'| = 1$, let $Z' = \{z\}$. % Determine whether $\bar{x} = z$.
   (a) $Y \leftarrow [m]\setminus\{z\}$, $k \leftarrow 0$,
   (b) while $Y \neq \emptyset$ and $k \leq 4\log m$, do:
   choose any test $e \in U$ with $z \notin T^*(e)$, suppose that $z \in T^+(e)$.[4] If the outcome of test e is inconsistent with z, then declare "$\bar{x} \notin Z$" and stop; otherwise let $Y \leftarrow Y\setminus T^-(e)$ and $k \leftarrow k + 1$.
   (c) Let $W \subseteq U$ denote the tests performed in step 3b and
   $$S = \{y \in [m] : r_y(e) = r_z(e) \text{ for at least } 2\log m \text{ tests } e \in W\}.$$
   (d) For each $y \in S$, perform any test that deterministically separates y and z.
   (e) If we never received an outcome inconsistent with z in step 3d, then declare "$\bar{x} = z$"; otherwise declare "$\bar{x} \notin Z$".

One caveat in step 3b is that it may happen that there are not $4\log m$ many tests e with $z \notin T^*(e)$. However, in this case, the algorithm must have already exhausted *all* tests with $z \notin T^*(e)$ and because z is still consistent with all observed outcomes, we know that $\bar{x} = z$.

Lemma 5.4. *If $\bar{x} \in Z$, then Member(Z) declares $\bar{x} = z$ with probability one; otherwise, Member(Z) declares $\bar{x} \notin Z$ with probability $1-m^{-2}$. Moreover, the expected cost of Member(Z) is $O(|Z|+m^{\alpha})$.*

Proof. If $\bar{x} \in Z$ then it is clear that Member(Z) declares $\bar{x} = z$. Now consider the case $\bar{x} \notin Z$. Recall that $z \in Z$ denote the unique hypothesis that is still compatible after step 2.

Case 1 If $\bar{x} \notin S$ then we have $\bar{x} \in T^*(e)$ for at least $2\log m$ tests $e \in W$. As z has a deterministic outcome for each test in W, the probability that all outcomes in W are consistent with z is at most m^{-2}. So with probability $1 - m^{-2}$ we must have observed that z is not compatible in step 3b and declare $\bar{x} \notin Z$.

Case 2 If $\bar{x} \in S$ then we will identify correctly that $\bar{x} \neq z$ in step 3c as one of these tests separates $\bar{x}$ and z deterministically. So in this case we will always declare $\bar{x} \notin Z$.

[4] The case $z \in T^-(e)$ is identical.

In order to bound the cost, note that the number of tests performed are: $|Z|$ in step 2, $4\log m$ in step 3b and $|S|$ in step 3d. The key step is to bound $|S|$, for which we use Proposition 2 with tests W and hypotheses $X = [m]$. In the notation of Proposition 2, $S = \{y \in X : \kappa(y) > |W|/2\}$ as $|W| = 4\log m$. It now follows that $|S| \leq 2Cm^{\alpha}$. Hence the total number of tests is $|Z| + 4\log m + |S| = O(|Z| + m^{\alpha})$. □

5.3 The Main Algorithm

Now define the main algorithm $\mathcal{A}_0$:

1. Initialize: compatible hypotheses $A \leftarrow [m]$, weights $w_x = 0$ for $x \in A$, number of tests $t \leftarrow 0$.
2. While $|A| \geq 2$,
   (a) If t is a power of 2, let $Z \subseteq A$ be the subset of $2Cm^{\alpha}$ scenarios with lowest w_x. Invoke Member(Z): if it identifies some Z-hypothesis as $\bar{x}$ then stop.
   (b) Else:
      i. Perform test $e \in U$ maximizing $\frac{1}{2}(|T^{+}(e) \cap A| + |T^{-}(e) \cap A|)$.
      ii. Update the weights: $w_x \leftarrow w_x + 1$ for each for each $x \in T^{*}(e)$.
      iii. Remove incompatible hypotheses from A (based on the test e outcome).
      iv. $t \leftarrow t + 1$.
3. Output the unique hypothesis in A as $\bar{x}$.

Using Lemma 5.4, it is clear that $\bar{x}$ is identified correctly with probability $1 - m^{-2}$. We now analyze the cost. Note that the membership oracle is invoked $O(\log m)$ times as the total number t of tests used is always at most m. Moreover, each time it is invoked on $O(m^{\alpha})$ scenarios. So the total number of tests due to step 2a is $O(m^{\alpha}\log m)$. In the rest of this section, we will bound the expected cost due to step 2b and ignore the cost of performing tests due to step 2a.

Truncated Decision Tree. Let $\mathcal{A}$ denote the decision tree corresponding to our algorithm. We only consider tests that correspond to step 2b. For any expanded hypothesis $(b, x) \in M$ let $P_{b,x}$ denote the path traced in $\mathcal{A}$. We will work with a truncated decision tree $\bar{\mathcal{A}}$, defined as follows.

Fix any $(b, x) \in M$. Let $\theta(t)$ denote the fraction of the first t tests in $P_{b,x}$ that are $*$-tests for x.[5]

$$\text{Define } t_{b,x} = \max\left\{t \in \{2^0, 2^1, \cdots, 2^{\log m}\} \ : \ \theta(t') \geq \frac{1}{\rho} \text{ for all } t' \leq t\right\}. \tag{11}$$

We let $\rho = 4$. If $t_{b,x} > |P_{b,x}|$ then reset $t_{b,x} = |P_{b,x}|$.

$\bar{\mathcal{A}}$ is the subtree of $\mathcal{A}$ consisting of the first $t_{b,x}$ tests along path $P_{b,x}$, for each $(b, x) \in M$.

Relating costs of $\mathcal{A}$ and $\bar{\mathcal{A}}$. We now show that the expected cost of $\mathcal{A}$ is at most twice that of $\bar{\mathcal{A}}$. We will show that for each $(b, x) \in M$, the number of tests performed $|P_{b,x}| \leq 2t_{b,x}$. Again, we only count tests from step 2b. We only need to consider the case that $t_{b,x} < |P_{b,x}|/2$. Let $t'_{b,x} = 2t_{b,x}$ which is a power-of-2. By (11) we know that there is some $t_{b,x} < k \leq t'_{b,x}$ with $\theta(k) < 1/\rho$. Hence $\theta(t'_{b,x}) < \frac{2}{\rho} < \frac{1}{2}$. Consider the point in the algorithm after performing the first $t'_{b,x}$ tests (call them W) on $P_{b,x}$. Let X be the compatible hypotheses after the $t'_{b,x}$-th test on $P_{b,x}$. Because $\theta(t'_{b,x}) < 1/2$, at most $|W|/2$ tests in W are $*$-tests for hypothesis x: in other words the weight $w_x \leq |W|/2$ at this point in the algorithm. Let $X' = \{y \in X : W \text{ has at most } \frac{|W|}{2} * \text{-tests for } y\}$. Using Proposition 2 with W and X, it follows that $|X'| \leq 2Cm^{\alpha}$. Hence the number of hypotheses $y \in X$ with $w_y \geq |W|/2$ is at most $2Cm^{\alpha}$, and so $x \in Z$ (recall that Z consists of $2Cm^{\alpha}$ hypotheses with the lowest weight).

[5] Only tests from step 2b are counted.

Bounding cost of $\bar{\mathcal{A}}$. Here we use the relation to stochastic set cover. Recall the instances $SSC(x)$ for $x \in [m]$. A key observation is:

Proposition 3. *Consider step 2b in the main algorithm, where A denotes the set of compatible hypotheses and e is the chosen test. For any $x \in T^*(e)$,*

$$\frac{1}{2}\left(|T^+(e) \cap A| + |T^-(e) \cap A|\right) = \mathbb{E}[|S_e(x) \cap (A \setminus x)|] \;\geq\; \frac{1}{2} \cdot \max_{d \in U} \mathbb{E}[|S_d(x) \cap (A \setminus x)|].$$

Proof. This follows easily by the choice of test e in step 2b. Suppose (for contradiction) there is some $d \in U$ with $\mathbb{E}[|S_d(x) \cap (A \setminus x)|] > 2 \cdot \mathbb{E}[|S_e(x) \cap (A \setminus x)|$.

- If $x \in T^*(d)$ then $\mathbb{E}[|S_d(x) \cap (A \setminus x)|] = \frac{1}{2}\left(|T^+(d) \cap A| + |T^-(d) \cap A|\right) > |T^+(e) \cap A| + |T^-(e) \cap A|$, which violates the choice of e.
- If $x \in T^+(d)$ then $\mathbb{E}[|S_d(x) \cap (A \setminus x)|] = |T^-(d) \cap A| > |T^+(e) \cap A| + |T^-(e) \cap A|$, which again violates the choice of e.

In either case, we obtain a contradiction. □

Fix any hypothesis $x \in [m]$ and consider decision tree $\bar{\mathcal{A}}_x$ obtained by *conditioning* $\bar{\mathcal{A}}$ on $\bar{x} = x$. The truncation (11) and Proposition 3 together imply that $\bar{\mathcal{A}}_x$ is a $(2, \rho)$ greedy policy for $SSC(x)$. Now, using Theorem 5.2, the expected cost of $\bar{\mathcal{A}}_x$ is $O(\log m) \cdot OPT_{SSC(x)}$. Taking expectations over $x \in [m]$, the expected cost of $\bar{\mathcal{A}}$ is $O(\log m) \sum_{x=1}^{m} \pi_x \cdot OPT_{SSC(x)}$, which is $O(\log m) \cdot OPT$ by Lemma 5.3.

Combined with the fact that the cost of $\mathcal{A}$ is at most twice that of $\bar{\mathcal{A}}$, the expected number of tests due to step 2b in the main algorithm is $O(\log m) \cdot OPT$. Adding the contribution from step 2a, we obtain an expected cost of

$$O(\log m) \cdot (m^{\alpha} + OPT) \leq_{(\text{as } \alpha < \frac{1}{2})} O(\log m) \cdot (m^{1-\alpha} + OPT) \leq_{(\text{Prop. 1})} O(\log m) \cdot OPT.$$

This completes the proof of Theorem 5.1.

6 Extensions

Instances that are not perfectly identifiable. We have assumed that for every pair x, y of hypotheses, there is some test that distinguishes them deterministically. Without this assumption, we can still obtain similar results by slightly changing the stopping criterion as follows. Define a *similarity graph* G on m nodes (corresponding to hypotheses) with an edge (x, y) if there is *no* test separating x and y deterministically. For each $x \in [m]$, let D_x be the star centered at x. We now want a policy using minimum number of tests that identifies a star containing $\bar{x}$. In the noise-less case, this problem has been studied in [23, 9] and an adaptive $O(d + \log m)$-approximation algorithm is known [24] where $d = \max_{i=1}^{m} |D_i|$. For us, $d = 1 +$ max-degree(G). Theorems 3.2 and 4.11.1 can be extended to obtain approximation ratios of $O(d \log m)$ and $O(d + \min(h, r) + \log m)$ respectively.

Non-binary outcomes. We can also handle tests with an arbitrary set Σ of outcomes (instead of ± 1). This requires extending the outcomes b to be in Σ^U and applying this change to the definitions of sets $T_{b,x}$ (1) and submodular function $f_{b,x}$ (2).

For the non-adaptive version, we will apply the approach using submodular ranking using different submodular functions $f_{b,x}$. In particular, for each $(b, x) \in M$ and region D_i, we define a submodular function:

$$f^i_{b,x}(S) = |\bigcup_{e \in S} T_{b,x}(e) \cap \overline{D_i}| \cdot \frac{1}{m - |D_i|}, \qquad \forall S \subseteq U,$$

where $\overline{D_i} = [m] \setminus D_i$. Assuming $\bar{x} = x$ and test-outcomes are given by b, we know $x \in D_i$ after tests S if and only if $f^i_{b,x}(S) = 1$. We define $f_{b,x}$ to be the "OR combination" of functions $f^i_{b,x}$ where $x \in D_i$. As in [24],

$$f_{b,x}(S) = \Pi_{i: x \in D_i}(1 - f^i_{b,x}(S)), \qquad \forall S \subseteq U.$$

Crucially, this OR combination ensures that $f_{b,x}(S) = 1$ iff some $f^i_{b,x}(S) = 1$; see [18]. The marginal increment parameter $\epsilon = m^{-d}$ as $|\{i : x \in D_i| \leq d$ for all $x \in [m]$. The non-adaptive algorithm is then identical to that in § 3 and we obtain an $O(\log \frac{1}{\epsilon}) = O(d \log m)$ approximation ratio.

For the adaptive version, following the noise-less algorithm in [24], we run a two-phase algorithm. In the first phase, we identify some subset $N \subseteq [m]$ containing $\bar{x}$ with $|N| \leq d$. This can be done using the algorithm in § 4 with the following submodular function for each $(b, x) \in M$.

$$f_{b,x}(S) == |\bigcup_{e \in S} T_{b,x}(e)| \cdot \frac{1}{m-d}, \qquad \forall S \subseteq U.$$

The expected cost of the resulting algorithm is $O(\min(r, h)+\log m)\cdot OPT$ using an identical analysis. Then, in the second phase, we run a simple splitting algorithm that iteratively selects any test that splits the current set H of candidate hypotheses, until H is contained in some region. The expected cost of this phase is at most $d\cdot OPT$. Combining both phases, we obtain an $O(d+\min(h, r)+\log m)$-approximation algorithm.

Non-uniform noise distribution. Our results extend directly to the case where each noisy outcome has a different probability of being ± 1. Suppose that the probability of every noisy outcome is between δ and $1 - \delta$. Then Theorems 3.2 and 4.11.1 continue to hold (irrespective of δ), and Theorem 5.1 holds with a slightly worse $O(\frac{1}{\delta} \log m)$ approximation ratio.

7 Experiments

We implemented our algorithms, and performed experiments on real-world and synthetic data sets. We compared our algorithms' cost (expected number of tests) with an information theoretic lower bound on the optimal cost and show that the difference is negligible. Thus, despite our logarithmic approximation ratios, the practical performance can be much better.

Chemicals with Unknown Test Outcomes One natural application of ODT is identifying chemical or biological materials. We considered a data set called WISER[6], which includes 400+ chemicals (hypothesis) and 78 binary tests. Every chemical has either positive, negative or unknown result on each test. To ensure every pair of chemical can be distinguished, we removed the chemicals that are not identifiable from each other to be left with 255 chemicals.

Random Binary Classifiers with Margin Error We construct a dataset containing 100 two-dimensional points, by picking each of their attributes uniformly in $[-1000, 1000]$. We also choose 2000 random triples (a, b, c) to form linear classifiers $\frac{ax+by}{\sqrt{a^2+b^2}} + c \leq 0$, where $a, b \leftarrow N(0, 1)$ and $c \leftarrow U(-1000, 1000)$. The point labels are binary and we introduce noisy outcomes based on the distance of each point to a classifier. Specifically, for each threshold $d \in \{0, 5, 10, 20, 30\}$ we define dataset CL-d that has a noisy outcome for any classifier-point pair where the distance of the point to the boundary of the classifier is smaller than d. In order to ensure that the instances are perfectly identifiable, we remove "equivalent" classifiers and we are left with 234 classifiers.

Distributions For the distribution over the hypotheses we have considered permutations of power law distribution ($\Pr[X = x; \alpha] = \beta x^\alpha$) for $a = 0, -0.5$ and 1. Note that, $\alpha = 0$ corresponds to uniform distribution. To be able to compare the results across different classifiers' datasets meaningfully. We have considered the same permutation in each distribution.

Algorithms and Results The following algorithms is implemented on Mac OS with 16GB RAM in Python 3 : the adaptive $O(r + \log m)$-approximation (ODTN$_r$), the adaptive $O(h + \log m)$-approximation (ODTN$_h$), the non-adaptive $O(\log m)$-approximation (Non-Adap) and a slightly adaptive version of Non-Adap (Low-Adap). Algorithm Low-Adap considers the same sequence of tests as Non-Adap while (adaptively) skipping non-informative tests based on observed outcomes. To take randomness of unknown outcomes into account, we run each algorithm 100 times. Tables 1,

[6] https://wiser.nlm.nih.gov

Tables 3 and Tables 5 show the expected costs of different algorithms on all datasets when the parameter α in the distribution over hypothesis is $0, -0.5$ and -1 correspondingly. These tables also report values of an information theoretic lower bound (the entropy) on the optimal cost (Low-BND). We also report the sample standard deviation of all algorithms in Tables 2, Tables 4 and Tables 6. Since the approximation ratio of some of our algorithms are dependent on maximum number of unknown outcomes per hypothesis (h) and maximum number of unknown outcomes per test (r), we also have included these parameters as well as their average values . From the results We can see that ODTN$_r$ consistently outperforms the other algorithms in every distribution and is very close to the lower bound.

Data / Algorithm	Wiser	Classifer-0	Classifer-5	Classifer-10	Classifer-20	Classifer-30
Low-BND	**7.994**	**7.870**	**7.870**	**7.870**	**7.870**	**7.870**
ODTN$_r$	8.357	7.910	7.927	7.915	7.962	8.000
ODTN$_h$	9.707	7.910	7.979	8.211	8.671	8.729
Non-Adap	11.568	9.731	9.831	9.941	9.996	10.204
Low-Adap	9.152	8.619	8.517	8.777	8.692	8.803

Table 1: Cost of Different Algorithms for $\alpha = 0$ (Uniform Distribution).

Data / Algorithm	Wiser	Classifer-0	Classifer-5	Classifer-10	Classifer-20	Classifer-30
ODTN$_r$	0.008	0	0	0.002	0.003	0.006
ODTN$_h$	0.01	0	0	0	0.004	0.01
Non-Adap	1.463	0.937	1.047	1.092	1.056	1.158
Low-Adap	0.0317	0.0685	0.0541	0.0760	0.0206	0.0550

Table 2: Standard Deviation of Different Algorithms for $\alpha = 0$ (Uniform Distribution).

Data / Algorithm	Wiser	Classifer-0	Classifer-5	Classifer-10	Classifer-20	Classifer-30
Low-BND	7.702	7.582	7.582	7.582	7.582	7.582
ODTN$_r$	8.177	7.757	7.780	7.789	7.831	7.900
ODTN$_h$	9.306	7.757	7.829	8.076	8.497	8.452
Non-Adap	11.998	9.504	9.500	9.694	9.826	9.934
Low-Adap	8.096	7.837	7.565	7.674	8.072	8.310

Table 3: Cost of Different Algorithms for $\alpha = -0.5$.

Data / Algorithm	Wiser	Classifer-0	Classifer-5	Classifer-10	Classifer-20	Classifer-30
ODTN$_r$	0.008	0	0	0.002	0.005	0.005
ODTN$_h$	0.008	0	0.003	0.004	0.008	0.007

Table 4: Standard Deviation of Different Algorithms for $\alpha = -0.5$.

Algorithm \ Data	Wiser	Classifer-0	Classifer-5	Classifer-10	Classifer-20	Classifer-30
Low-BND	6.218	6.136	6.136	6.136	6.136	6.136
ODTN_r	7.367	6.998	7.121	7.150	7.299	7.357
ODTN_h	8.566	6.998	7.134	7.313	7.637	7.915
Non-Adap	11.976	9.598	9.672	9.824	10.159	10.277
Low-Adap	9.072	8.453	8.344	8.609	8.683	8.541

Table 5: Cost of Different Algorithms for $\alpha = -1$.

Algorithm \ Data	Wiser	Classifer-0	Classifer-5	Classifer-10	Classifer-20	Classifer-30
ODTN_r	0.004	0	0.001	0.001	0.003	0.003
ODTN_h	0.005	0	0.001	0.003	0.005	0.004

Table 6: Standard Deviation of Different Algorithms for $\alpha = -1$.

Parameters \ Data	Wiser	Classifer-0	Classifer-5	Classifer-10	Classifer-20	Classifer-30
r	245	0	5	7	12	13
Avg-r	30.690	0	1.12	2.21	4.43	6.54
h	45	0	3	6	8	8
Avg-h	9.39	0	0.48	0.94	1.89	2.79

Table 7: Maximum and Average Number of Stars per Hypothesis and per Test in Different Datasets.

References

[1] M. Adler and B. Heeringa. Approximating optimal binary decision trees. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 1–9. Springer, 2008.

[2] E. M. Arkin, H. Meijer, J. S. Mitchell, D. Rappaport, and S. S. Skiena. Decision trees for geometric models. *International Journal of Computational Geometry & Applications*, 8(03):343–363, 1998.

[3] Y. Azar and I. Gamzu. Ranking with submodular valuations. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 1070–1079. SIAM, 2011.

[4] M. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, pages 65–72, 2006.

[5] G. Bellala, S. K. Bhavnani, and C. Scott. Active diagnosis under persistent noise with unknown noise distribution: A rank-based approach. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, pages 155–163, 2011.

[6] V. T. Chakaravarthy, V. Pandit, S. Roy, P. Awasthi, and M. K. Mohania. Decision trees for entity identification: Approximation algorithms and hardness results. *ACM Trans. Algorithms*, 7(2):15:1–15:22, 2011.

[7] V. T. Chakaravarthy, V. Pandit, S. Roy, and Y. Sabharwal. Approximating decision trees with multiway branches. In *International Colloquium on Automata, Languages, and Programming*, pages 210–221. Springer, 2009.

[8] Y. Chen, S. H. Hassani, and A. Krause. Near-optimal bayesian active learning with correlated and noisy tests. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, pages 223–231, 2017.

[9] Y. Chen, S. Javdani, A. Karbasi, J. A. Bagnell, S. S. Srinivasa, and A. Krause. Submodular surrogates for value of information. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 3511–3518, 2015.

[10] F. Cicalese, E. S. Laber, and A. M. Saettler. Diagnosis determination: decision trees optimizing simultaneously worst and expected testing cost. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 414–422, 2014.

[11] S. Dasgupta. Analysis of a greedy active learning strategy. In *Advances in neural information processing systems*, pages 337–344, 2005.

[12] M. Garey and R. Graham. Performance bounds on the splitting algorithm for binary testing. *Acta Informatica*, 3:347–355, 1974.

[13] D. Golovin and A. Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *J. Artif. Intell. Res.*, 42:427–486, 2011.

[14] D. Golovin and A. Krause. Adaptive submodularity: A new approach to active learning and stochastic optimization. *CoRR*, abs/1003.3967, 2017.

[15] D. Golovin, A. Krause, and D. Ray. Near-optimal bayesian active learning with noisy observations. In *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada.*, pages 766–774, 2010.

[16] A. Guillory and J. A. Bilmes. Average-case active learning with costs. In *Algorithmic Learning Theory, 20th International Conference, ALT 2009, Porto, Portugal, October 3-5, 2009. Proceedings*, pages 141–155, 2009.

[17] A. Guillory and J. A. Bilmes. Interactive submodular set cover. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 415–422, 2010.

[18] A. Guillory and J. A. Bilmes. Simultaneous learning and covering with adversarial noise. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 369–376, 2011.

[19] A. Gupta, V. Nagarajan, and R. Ravi. Approximation algorithms for optimal decision trees and adaptive tsp problems. *Mathematics of Operations Research*, 42(3):876–896, 2017.

[20] S. Hanneke. A bound on the label complexity of agnostic active learning. In *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, pages 353–360, 2007.

[21] L. Hyafil and R. L. Rivest. Constructing optimal binary decision trees is NP-complete. *Information Processing Lett.*, 5(1):15–17, 1976/77.

[22] S. Im, V. Nagarajan, and R. V. D. Zwaan. Minimum latency submodular cover. *ACM Transactions on Algorithms (TALG)*, 13(1):13, 2016.

[23] S. Javdani, Y. Chen, A. Karbasi, A. Krause, D. Bagnell, and S. S. Srinivasa. Near optimal bayesian active learning for decision making. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, April 22-25, 2014*, pages 430–438, 2014.

[24] P. Kambadur, V. Nagarajan, and F. Navidi. Adaptive submodular ranking. In *Integer Programming and Combinatorial Optimization - 19th International Conference, IPCO 2017, Waterloo, ON, Canada, June 26-28, 2017, Proceedings*, pages 317–329, 2017 (full version: https://arxiv.org/abs/1606.01530).

[25] S. R. Kosaraju, T. M. Przytycka, and R. Borgstrom. On an optimal split tree problem. In *Workshop on Algorithms and Data Structures*, pages 157–168. Springer, 1999.

[26] Z. Liu, S. Parthasarathy, A. Ranganathan, and H. Yang. Near-optimal algorithms for shared filter evaluation in data stream systems. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 133–146, 2008.

[27] D. W. Loveland. Performance bounds for binary testing with arbitrary weights. *Acta Inform.*, 22(1):101–114, 1985.

[28] M. J. Moshkov. Greedy algorithm with weights for decision tree construction. *Fundam. Inform.*, 104(3):285–292, 2010.

[29] M. Naghshvar, T. Javidi, and K. Chaudhuri. Noisy bayesian active learning. In *50th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2012, Allerton Park & Retreat Center, Monticello, IL, USA, October 1-5, 2012*, pages 1626–1633, 2012.

[30] F. Nan and V. Saligrama. Comments on the proof of adaptive stochastic set cover based on adaptive submodularity and its implications for the group identification problem in "group-based active query selection for rapid diagnosis in time-critical situations". *IEEE Trans. Information Theory*, 63(11):7612–7614, 2017.

[31] R. D. Nowak. Noisy generalized binary search. In *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada.*, pages 1366–1374, 2009.

[32] A. M. Saettler, E. S. Laber, and F. Cicalese. Trading off worst and expected cost in decision tree problems. *Algorithmica*, 79(3):886–908, 2017.

[33] L. A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982.