Contextual semibandits via supervised learning oracles

Authored by:

Alekh Agarwal
Akshay Krishnamurthy
Miro Dudik

Abstract

We study an online decision making problem where on each round a learner chooses a list of items based on some side information, receives a scalar feedback value for each individual item, and a reward that is linearly related to this feedback. These problems, known as contextual semibandits, arise in crowdsourcing, recommendation, and many other domains. This paper reduces contextual semibandits to supervised learning, allowing us to leverage powerful supervised learning methods in this partial-feedback setting. Our first reduction applies when the mapping from feedback to reward is known and leads to a computationally efficient algorithm with near-optimal regret. We show that this algorithm outperforms state-of-the-art approaches on real-world learning-to-rank datasets, demonstrating the advantage of oracle-based algorithms. Our second reduction applies to the previously unstudied setting when the linear mapping from feedback to reward is unknown. Our regret guarantees are superior to prior techniques that ignore the feedback.

1 Paper Body

Decision making with partial feedback, motivated by applications including personalized medicine [21] and content recommendation [16], is receiving increasing attention from the machine learning community. These problems are formally modeled as learning from bandit feedback, where a learner repeatedly takes an action and observes a reward for the action, with the goal of maximizing reward. While bandit learning captures many problems of interest, several applications have additional structure: the action is combinatorial in nature and more detailed feedback is provided. For example, in internet applications, we often recommend sets of items and record information about the user?s interaction with each individual item (e.g., click). This additional feedback is unhelpful unless it relates to the overall reward (e.g., number of clicks), and, as in previous work, we assume a linear relationship. This interaction is known as the semibandit

feedback model. Typical bandit and semibandit algorithms achieve reward that is competitive with the single best fixed action, i.e., the best medical treatment or the most popular news article for everyone. This is often inadequate for recommendation applications: while the most popular articles may get some clicks, personalizing content to the users is much more effective. A better strategy is therefore to leverage contextual information to learn a rich policy for selecting actions, and we model this as contextual semibandits. In this setting, the learner repeatedly observes a context (user features), chooses a composite action (list of articles), which is an ordered tuple of simple actions, and receives reward for the composite action (number of clicks), but also feedback about each simple action (click). The goal of the learner is to find a policy for mapping contexts to composite actions that achieves high reward. We typically consider policies in a large but constrained class, for example, linear learners or tree ensembles. Such a class enables us to learn an expressive policy, but introduces a computational challenge of finding a good policy without direct enumeration. We build on the supervised learning literature, which has developed fast algorithms for such policy classes, including logistic regression and SVMs for linear classifiers and boosting for tree ensembles. We access the policy class exclusively through a supervised learning algorithm, viewed as an oracle. 30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

Algorithm VCEE (Thm. 1) ?-Greedy (Thm. 3) Kale et al. [12] EELS (Thm. 2) Agarwal et al. [1] Swaminathan et al. [22]

p

Regret

KLT log N (LTp)2/3 (K log N)1/3 KLT log N 2/3 (LT)p (K log N)1/3 L K L T log N 4/3 2/3 L T (K log N)1/3

Oracle Calls p T 3/2 K/(L log N) 1 not oracle-based p

1 K L T / log N 1

Weights w? known known known unknown unknown unknown

Table 1: Comparison of contextual semibandit algorithms for arbitrary policy classes, assuming all ? rankings are valid composite actions. The reward is semibandit feedback weighted according p to w . ? ? For known weights, we consider w = 1; for unknown weights, we assume kw k2 ? O(L). In this paper, we develop and evaluate oracle-based algorithms for the contextual semibandits problem. We make the following contributions: 1. In the more common setting where the linear function relating the semibandit feedback to the reward is known, we develop a new algorithm, called VCEE, that extends the oracle-based contextual bandit algorithm of Agarwal p p et al. [1]. We show that VCEE enjoys a regret bound ? KLT log N and O ? L KT log N , depending on the combinatorial structure of between O the problem, when there are T rounds of interaction, K simple actions, N policies, and composite ? 3/2) calls to actions have length L.1 VCEE can handle structured action spaces and makes O(T the supervised learning oracle. 2. We empirically evaluate this algorithm on two large-scale learning-to-rank datasets and compare with other contextual semibandit approaches. These experiments comprehensively demonstrate that effective exploration over a rich policy class can lead to significantly better per-

formance than existing approaches. To our knowledge, this is the first thorough experimental evaluation of not only oracle-based semibandit methods, but of oracle-based contextual bandits as well. 3. When the linear function relating the feedback to the reward is unknown, we develop a new algorithm called EELS. Our algorithm first learns the linear function by uniform exploration and then, adaptively, switches to act according to an empirically optimal policy. We prove an ? (LT)2/3 (K log N)1/3 regret bound by analyzing when to switch. We are not aware of other O computationally efficient procedures with a matching or better regret bound for this setting. See Table 1 for a comparison of our results with existing applicable bounds. Related work. There is a growing body of work on combinatorial bandit optimization [2, 4] with considerable attention on semibandit feedback [6, 10, 12, 13, 19]. The majority of this research focuses on the non-contextual setting with a knownprelationship between semibandit feedback and ? KLT) regret against the best fixed composite reward, and a typical algorithm here achieves an O(action. To our knowledge, only the work of Kale et al. [12] and Qin et al. [19] considers the contextual setting, again with known p relationship. The former generalizes the Exp4 algorithm [3] ? KLT) regret,2 but requires explicit enumeration of the policies. to semibandits, and achieves O(The latter generalizes the LinUCB algorithm of Chu et al. [7] to semibandits, assuming that the simple action feedback is linearly related to the context. This differs from our setting: we make no assumptions about the simple action feedback. In our experiments, we compare VCEE against this LinUCB-style algorithm and demonstrate substantial improvements. We are not aware of attempts to learn a relationship between the overall reward and the feedback on simple actions as we do with EELS. While EELS uses least squares, as in LinUCB-style approaches, it does so without assumptions on the semibandit feedback. Crucially, the covariates for its least squares problem are observed after predicting a composite action and not before, unlike in LinUCB. Supervised learning oracles have been used as a computational primitive in many settings including active learning [11], contextual bandits [1, 9, 20, 23], and structured prediction [8]. 1

? notation suppressed factors polylogarithmic in K, L, T and log N . We Throughout the paper, the O(?) analyze finite policy classes, but our work extends to infinite classes by standard discretization arguments. 2 Kale et al. [12] consider the favorable setting where our bounds match, when uniform exploration is valid.

2

2

Preliminaries

Let X be a space of contexts and A a set of K simple actions. Let ? ? (X ! AL) be a finite set of policies, —?— = N , mapping contexts to composite actions. Composite actions, also called rankings, are tuples of L distinct simple actions. In general, there are K!/(K L)! possible rankings, but they might not be valid in all contexts. The set of valid rankings for a context x is defined implicitly through the policy class as {?(x)}?2? .

Let (?) be the set of distributions over policies, and ? (?) be the set of

non-negative weight vectors over policies, summing to at most 1, which we call subdistributions. Let 1(?) be the 0/1 indicator equal to 1 if its argument is true and 0 otherwise.

In stochastic contextual semibandits, there is an unknown distribution D over triples (x, y, ?), where x is a context, y 2 [0, 1]K is the vector of reward features, with entries indexed by simple actions as y(a), and ? 2 [1, 1] is the reward noise, E[?—x, y] = 0. Given y 2 RK and A = (a1 , . . . , aL) 2 AL , we write y(A) 2 RL for the vector with entries y(a‘). The learner plays a T -round game. In each round, nature draws (xt , yt , ?t) ? D and reveals the context xt . The learner selects a valid ranking PL At = (at,1 , at,2 , . . . , at,L) and gets reward rt (At) = ‘=1 w‘? yt (at,‘) + ?t , where w? 2 RL is a possibly unknown but fixed weight vector. The learner is shown the reward rt (At) and the vector of reward features for the chosen simple actions yt (At), jointly referred to as semibandit feedback. The goal is?to achieve ? cumulative reward competitive with all ? 2 ?. For a policy ?, let R(?) := E(x,y,?)?D r ?(x) denote its expected reward, and let ? ? := argmax?2? R(?) be the maximizer of expected reward. We measure performance of an algorithm via cumulative empirical regret, Regret :=

T X

rt (? ? (xt))

(1)

rt (At).

t=1

The performance of a policy ? is measured by its expected regret, Reg(?) := R(? ?) R(?). Example 1. In personalized search, a learning system repeatedly responds to queries with rankings of search items. This is a contextual semibandit problem where the query and user features form the context, the simple actions are search items, and the composite actions are their lists. The semibandit feedback is whether the user clicked on each item, while the reward may be the click-based discounted cumulative gain (DCG), which is a weighted sum of clicks, with position-dependent weights. We want to map contexts to rankings to maximize DCG and achieve a low regret. We assume that our algorithms have access to a supervised learning oracle, also called an argmax oracle, denoted AMO, that can find a policy with the maximum empirical reward on any appropriate dataset. Specifically, given a dataset D = {xi , yi , vi }ni=1 of contexts xi , reward feature vectors yi 2 RK with rewards for all simple actions, and weight vectors vi 2 RL , the oracle computes AMO(D) := argmax ?2?

n X i=1

hvi , yi (?(xi))i = argmax ?2?

n X L X

vi,‘ yi (?(xi)‘),

(2)

i=1 ‘=1

where ?(x)‘ is the ‘th simple action that policy ? chooses on context x. The oracle is supervised as it assumes known features yi for all simple actions whereas we only observe them for chosen actions. This oracle is the structured

generalization of the one considered in contextual bandits [1, 9] and can be implemented by any structured prediction approach such as CRFs [14] or SEARN [8]. Our algorithms choose composite actions by sampling from a distribution, which allows us to use importance weighting to construct unbiased estimates for the reward features y. If on round t, a composite action At is chosen with probability Qt (At), we construct the importance weighted feature vector y?t with components y?t (a) := yt (a)1(a 2 At)/Qt (a 2 At), which are unbiased estimators of yt (a). For a policy ?, we then define empirical estimates of its reward and regret, resp., as t

?t (?, w) :=

1X hw, y?i (?(xi))i t i=1

and

d t (?, w) := max ?t (? 0 , w) Reg 0 ?

?t (?, w).

d t (?, w?) By construction, ?t (?, w?) is an unbiased estimate of the expected reward R(?), but Reg ? is not an unbiased estimate of the expected regret Reg(?). We use Ex?H [?] to denote empirical expectation over contexts appearing in the history of interaction H. 3

Algorithm 1 VCEE (Variance-Constrained Explore-Exploit) Algorithm Require: Allowed failure probability 2 (0, 1). n o p 1: Q0 = 0, the all-zeros vector. H0 = ;. Define: ?t = min 1/2K, ln(16t2 N/)/(Ktpmin) . 2: for round t = 1, . . . , T do ? t 1 = Qt 1 + (1 P Qt 1 (?))1? . 3: Let ?t 1 = argmax?2? ?t 1 (?, w?) and Q t 1 ? ? ?t 1 (? — xt) (see Eq. (3)), and observe yt (At) and rt (At). 4: Observe xt 2 X, play At ? Q t 1 ? ?t 1 (a 2 A — xt) for each a. 5: Define qt (a) = Q t 1 6: Obtain Qt by solving OP with Ht = Ht 1 [{(xt , yt (At), qt (At)} and ?t . 7: end for Semi-bandit Optimization Problem (OP) With history H and ?

?

c

k1 Regt (?) 0, define b? := kw and kw? k22 ?pmin X Q(?)b? ? 2KL/pmin ?2?

8? 2 ? :

? x?H E

”

L X ‘=1

:= 100. Find Q 2

1 2KL ? + b? ? Q (?(x)‘ 2 A — x) pmin

? (?)

such that: (4) (5)

Finally, we introduce projections and smoothing of distributions. For any ? 2 [0, 1/K] and any subdistribution P 2 ? (?), the smoothed and projected conditional subdistribution P ? (A — x) is X P ? (A — x) := (1 K?) P (?)1(?(x) = A) + K?Ux (A), (3) ?2?

where Ux is a uniform distribution over a certain subset of valid rankings for context x, designed to ensure that the probability of choosing each valid simple action is large. By mixing Ux into our action selection, we limit the

variance of reward feature estimates y?. The lower bound on the simple action probabilities under Ux appears in our analysis as pmin , which is the largest number satisfying Ux (a 2 A)

pmin /K

for all x and all simple actions a valid for x. Note that pmin = L when there are no restrictions on the action space as we can take Ux to be the uniform distribution over all rankings and verify that Ux (a 2 A) = L/K. In the worst case, pmin = 1, since we can always find one valid ranking for each valid simple action and let Ux be the uniform distribution over this set. Such a ranking can be found efficiently by a call to AMO for each simple action a, with the dataset of a single point (x, 1a 2 RK , 1 2 RL), where 1a (a0) = 1(a = a0).

3

Semibandits with known weights

We begin with the setting where the weights w? are known, and present an efficient oracle-based algorithm (VCEE, see Algorithm 1) that generalizes the algorithm of Agarwal et al. [1]. The algorithm, before each round t, constructs a subdistribution Qt 1 2 ? (?), which is used to ? t 1 by placing the missing mass on the maximizer of empirical reward. The form the distribution Q ? ?t 1 (see composite action for the context xt is chosen according to the smoothed distribution Q t 1 Eq. (3)). The subdistribution Qt 1 is any solution to the feasibility problem (OP), which balances exploration and exploitation via the constraints in Eqs. (4) and (5). Eq. (4) ensures that the distribution has low empirical regret. Simultaneously, Eq. (5) ensures that the variance of the reward estimates y? remains sufficiently small for each policy ?, which helps control the deviation between empirical and expected regret, and implies that Qt 1 has low expected regret. For each ?, the variance constraint is based on the empirical regret of ?, guaranteeing sufficient exploration amongst all good policies. OP can be solved efficiently using AMO and a coordinate descent procedure obtained by modifying the algorithm of Agarwal et al. [1]. While the full algorithm and analysis are deferred to Appendix E, several key differences between VCEE and the algorithm of Agarwal et al. [1] are worth highlighting.

4

One crucial modification is that the variance constraint in Eq. (5) involves the marginal probabilities of the simple actions rather than the composite actions as would be the most obvious adaptation to our setting. This change, based on using the reward estimates y?t for simple actions, leads to substantially lower variance of reward estimates for all policies and, consequently, an improved regret bound. Another important modification is the new mixing distribution Ux and the quantity pmin . For structured composite action spaces, uniform exploration over the valid composite actions may not provide sufficient coverage of each simple action and may lead to dependence on the composite action space size, which is exponentially worse than when Ux is used. The regret guarantee for Algorithm 1 is the following: Theorem 1. For any 2 (0, 1), with probability at least 1 , VCEE achieves rekw? k22 p ? gret O kw? k1 L KT log(N/) / pmin . Moreover, VCEE can be efficiently implemented with p ? T 3/2 K / (pmin log(N/)) calls to a supervised learning oracle AMO. O

In Table 1, we compare this result to other applicable regret bounds in pthe most common setting, ? KLT log N) regret bound, where w? = 1 and all rankings are valid (pmin = L). VCEE enjoys a O(which is the best bound amongst oracle-based approaches, representing an exponentially better L-dependence over the purely bandit feedback variant [1] and a polynomially better T -dependence over an ?-greedy scheme (see Theorem 3 in Appendix A). This improvement over ?-greedy is also verified by our experiments. Additionally, our bound matches that of Kale et al. [12], who consider the harder adversarial setting but give an algorithm that requires an exponentially worse running time, ?(N T), and cannot be efficiently implemented with an oracle.

Other results address the non-contextual p setting, where the optimal bounds for both stochastic [13] and adversarial [2] semibandits are ?(KLT). Thus, our bound may be optimal when pmin = ?(L). However, these p results apply even without requiring p all rankings to be valid, so they improve on our bound by a L factor when pmin = 1. This L discrepancy may not be fundamental, but it seems unavoidable with some degree of uniform exploration, as in all existing contextual bandit algorithms. A promising avenue to resolve this gap is to extend the work of Neu [18], which gives high-probability bounds in the noncontextual setting without uniform exploration. To summarize, our regret bound is similar to existing results on combinatorial (semi)bandits but represents a significant improvement over existing computationally efficient approaches.

4

Semibandits with unknown weights

We now consider a generalization of the contextual semibandit problem with a new challenge: the weight vector w? is unknown. This setting is substantially more difficult than the previous one, as it is no longer clear how to use the semibandit feedback to optimize for the overall reward. Our result shows that the semibandit feedback can still be used effectively, even when the transformation is unknown. Throughout, we assume that the true weight vector w? has bounded norm, i.e., kw? k2 ? B. One restriction required by our analysis is the ability to play any ranking. Thus, all rankings must be valid in all contexts, which is a natural restriction in domains such as information retrieval and recommendation. The uniform distribution over all rankings is denoted U .

We propose an algorithm that explores first and then, adaptively, switches to exploitation. In the exploration phase, we play rankings uniformly at random, with the goal of accumulating enough information to learn the weight vector w? for effective policy optimization. Exploration lasts for a variable length of time governed by two parameters n? and ? . The n? parameter controls the minimum number of rounds of the exploration phase and is O(T 2/3), similar to ?-greedy style schemes [15]. The adaptivity is implemented by the ? parameter, which imposes a lower bound on the eigenvalues of the 2nd-moment matrix of reward features observed during exploration. As a result, we only transition to the exploitation phase after this matrix has suitably large eigenvalues. Since we make no assumptions about the reward features, there is no bound on how many rounds this may take. This is a departure from previous explore-first schemes, and captures the difficulty of learning w? when we observe

the regression features only after taking an action. After the exploration phase of t rounds, we perform least-squares regression using the observed reward features and the rewards to learn an estimate w ? of w? . We use w ? and importance weighted 5

Algorithm 2 EELS (Explore-Exploit Least Squares) Require: Allowed failure probability 2 (0, 1). Assume kw? k2 ? B. p 2/3 1/3 2/3 1: Set n? T (K ln(N/)/L) max{1, (B L) } 2: for t = 1, . . . , n? do 3: Observe xt , play At ? U (U is uniform over all rankings), observe yt (At) and rt (At). 4: end for P n? P 2 t) 5: Let V? = 2n?1K 2 yt (b) U1(a,b2A t=1 a,b2A yt (a) (a,b2At) . 6: V? 2V? + 3 ln(2/ n)/(2n?). o 1/3 7: Set ? max 6L2 ln(4LT /), (T V? /B)2/3 (L ln(2/)) . P n? T 8: Set ? t=1 yt (At)yt (At) . 9: while min (?) ? ? do 10: t t + 1. Observe xt , play At ? U , observe yt (At) and rt (At). 11: Set ? ? + yt (At)yt (At)T . 12: end while Pt 13: Estimate weights w ? ? 1 (i=1 yi (Ai)ri (Ai)) (Least Squares). 14: Optimize policy ? ? argmax?2? ?t (?, w) ? using importance weighted features. 15: For every remaining round: observe xt , play At = ? ? (xt). reward features from the exploration phase to find a policy ? ? with maximum empirical reward, ?t (?, w). ? The remaining rounds comprise the exploitation phase, where we play according to ? ?. The remaining question is how to set ? , which governs the length of the exploration phase. The ideal setting uses the unknown parameter V := E(x,y)?D Vara?Unif(A) [y(a)] of the distribution D, where Unif(A) is the uniform distribution over all simple actions. We form an unbiased estimator V? of V and derive an upper bound V? . While the optimal ? depends on V , the upper bound V? suffices. For this algorithm, we prove the following regret bound. Theorem 2. For any 2 (0, 1) and T K ln(N/)/ min{L, (BL)2 }, with probability at least 1 , ? T 2/3 (K log(N/))1/3 max{B 1/3 L1/2 , BL1/6 } . EELS can be implemented EELS has regret O efficiently with one call to the optimization oracle. The theorem shows that we can achieve sublinear regret without dependence on the composite action space size even when the weights are unknown. The p only applicable alternatives from the literature are displayed in Table 1, specialized to B = ?(L). First, oracle-based contextual bandits [1] achieve a better T -dependence, but both the regret and the number of oracle calls grow exponentially with L. Second, the deviation bound of Swaminathan et al. [22], which exploits the reward structure but not the semibandit feedback, leads to an algorithm with regret that is polynomially worse in its dependence on L and B (see Appendix B). This observation is consistent with non-contextual results, which show that the value of semibandit information is only in L factors [2]. Of course EELS has a sub-optimal dependence on T , although this is the best we are aware of for a computationally efficient algorithm in this setting. It is an interesting open question to achieve p poly(K, L) T log N regret with unknown weights.

5

Proof sketches

We next sketch the arguments for our theorems. Full proofs are deferred to the appendices. Proof of Theorem 1: The result generalizes Agarwal et. al [1], and the proof structure is similar. For the regret bound, we use Eq. (5) to control the deviation of the empirical reward estimates which d t . A careful

inductive argument leads to the following bounds: make up the empirical regret Reg d t (?) + c0 Reg(?) ? 2Reg

kw? k22 KL?t kw? k1

and

? 2

d t (?) ? 2Reg(?) + c0 kw k2 KL?t . Reg kw? k1

Here c0 is a universal constant and ?t is defined in the pseudocode. Eq. (4) guarantees low empirical ? ?t t , and the above inequalities also ensure small population regret. regret when playing according to Q 6

PT kw? k2 The cumulative regret is bounded by kw? k21 KL t=1 ?t , which grows at the rate given in Theorem 1. The number of oracle calls is bounded by the analysis of the number of iterations of coordinate descent used to solve OP, via a potential argument similar to Agarwal et al. [1]. Proof of Theorem 2: We analyze the exploration and exploitation phases individually, and then optimize n? and ? to balance these exploration phase, the expected per-round regret p terms. For the p can be bounded by either kw? k2 KV or kw? k2 L, but the number of rounds depends on the minimum eigenvalue min (?), with ? defined in Steps 8 and 11. However, the expected per-round 2nd-moment matrix, E(x,y)?D,A?U [y(A)y(A)T], has all eigenvalues at least V . Thus, after t rounds, we expect min (?) tV , so exploration lasts about ? /V rounds, yielding roughly p p ? Exploration Regret ? ? kw? k2 min{ KV , L}. V Now our choice of

?

produces a benign dependence on V and yields a T 2/3 bound.

For the exploitation phase, we bound the error between the empirical reward estimates ?t (?, w) ? and the true reward R(?). Since we know min (?) ? in this phase, we obtain r r p p K log N L ? Exploitation Regret ? T kw k2 +T min{ KV , L}. n? ? The first term captures the error from using the importance-weighted y? vector, while the second uses a bound on the error kw ? w? k2 from the analysis of linear regression (assuming min (?) ?).

This high-level argument ignores several important details. First, we must show that using V? instead of the optimal choice V in the setting of ? does not affect the regret. Secondly, since the termination condition for the exploration phase depends on the random variable ?, we must derive a highprobability bound on the number of exploration rounds to control the regret. Obtaining this bound requires a careful application of the matrix Bernstein inequality to certify that ? has large eigenvalues.

6

Experimental Results

Our experiments compare VCEE with existing alternatives. As VCEE generalizes the algorithm of Agarwal et al. [1], our experiments also provide insights into oracle-based contextual bandit approaches and this is the first detailed empirical study of such algorithms. The weight vector w? in our datasets was known, so we do not evaluate EELS. This section contains a high-level description of our experimental setup, with details on our implementation, baseline algorithms, and policy classes deferred to Appendix C. Software is available at

http://github.com/akshaykr/oracle_cb. Data: We used two large-scale learning-to-rank datasets: MSLR [17] and all folds of the Yahoo! Learning-to-Rank dataset [5]. Both datasets have over 30k unique queries each with a varying number of documents that are annotated with a relevance in {0, . . . , 4}. Each query-document pair has a feature vector (d = 136 for MSLR and d = 415 for Yahoo!) that we use to define our policy class. For MSLR, we choose K = 10 documents per query and set L = 3, while for Yahoo!, we set K = 6 and L = 2. The goal is to maximize the sum of relevances of shown documents (w? = 1) and the individual relevances are the semibandit feedback. All algorithms make a single pass over the queries. Algorithms: We compare VCEE, implemented with an epoch schedule for solving OP after 2i/2 rounds (justified by Agarwal et al. [1]), with two baselines. First is the ?-G REEDY approach [15], with a constant but tuned ?. This algorithm explores uniformly with probability ? and follows the empirically best policy otherwise. The empirically best policy is updated with the same 2i/2 schedule. We also compare against a semibandit version of L IN UCB [19]. This algorithm models the semibandit feedback as linearly related to the query-document features and learns this relationship, while selecting composite actions using an upper-confidence bound strategy. Specifically, the algorithm maintains a weight vector ?t 2 Rd formed by solving a ridge regression problem with the semibandit feedback yt (at,‘) as regression targets. At round t, the algorithm uses document features {xa }a2A and chooses the L documents with highest xTa ?t + ?xTa ?t 1 xa value. Here, ?t is the feature 2nd-moment matrix and ? is a tuning parameter. For computational reasons, we only update ?t and ?t every 100 rounds. Oracle implementation: L IN UCB only works with a linear policy class. VCEE and ?-G REEDY work with arbitrary classes. Here, we consider three: linear functions and depth-2 and depth-5
7

Dataset: MSLR
Average reward
1.0
Dataset: Yahoo!
3.1
0.8
2.3
0.6
3.0
0.4
2.2
0.2 0.0 0.0
10000 0.2 ?-Lin
VC-Lin
2.9
20000
0.430000 0.6 Number of interactions (T) ?-GB2 VC-GB2
10000 ?-GB5
0.8 20000 VC-GB5

1.0 30000 LinUCB

Figure 1: Average reward as a function of number of interactions T for VCEE, ?-G REEDY, and L IN UCB on MSLR (left) and Yahoo (right) learning-to-rank datasets. gradient boosted regression trees (abbreviated Lin, GB2 and GB5). Both GB classes use 50 trees. Precise details of how we instantiate the supervised learning oracle can be found in Appendix C. Parameter tuning: Each p algorithm has a parameter governing the explore-exploit tradeoff. For VCEE, we set ?t = c 1/KLT and tune c, in ?-G REEDY we tune ?, and in L IN UCB we tune ?. We ran each algorithm for 10 repetitions, for each of ten logarithmically spaced parameter values.

Results: In Figure 1, we plot the average reward (cumulative reward up to round t divided by t) on both datasets. For each t, we use the parameter that achieves the best average reward across the 10 repetitions at that t. Thus for each t, we are showing the performance of each algorithm tuned to maximize reward over t rounds. We found VCEE was fairly stable to parameter tuning, so for VC-GB5 we just use one parameter value (c = 0.008) for all t on both datasets. We show confidence bands at twice the standard error for just L IN UCB and VC-GB5 to simplify the plot.

Qualitatively, both datasets reveal similar phenomena. First, when using the same policy p class, VCEE consistently outperforms ?-G REEDY. This agrees with our theory, as VCEE achieves T -type regret, while a tuned ?-G REEDY achieves at best a T 2/3 rate. Secondly, if we use a rich policy class, VCEE can significantly improve on L IN UCB, the empirical state-of-the-art, and one of few practical alternatives to ?-G REEDY. Of course, since ?-G REEDY does not outperform L IN UCB, the tailored exploration of VCEE is critical. Thus, the combination of these two properties is key to improved performance on these datasets. VCEE is the only contextual semibandit algorithm we are aware of that performs adaptive exploration and is agnostic to the policy representation. Note that L IN UCB is quite effective and outperforms VCEE with a linear class. One possible explanation for this behavior is that L IN UCB, by directly modeling the reward, searches the policy space more effectively than VCEE, which uses an approximate oracle implementation.

7

Discussion

This paper develops oracle-based algorithms for contextual semibandits both with known and unknown weights. In both cases, our algorithms achieve the best known regret bounds for computationally efficient procedures. Our empirical evaluation of VCEE, clearly demonstrates the advantage of sophisticated oracle-based approaches over both parametric approaches and naive exploration. To our knowledge this is the first detailed empirical evaluation of oracle-based contextual bandit or semibandit learning. We close with some promising directions for future work: p ? KLT log N) regret even with structured action spaces? 1. With known weights, can we obtain O(This may require a new contextual bandit algorithm that does not use uniform smoothing. p 2. With unknown weights, can we achieve a T dependence while exploiting semibandit feedback?

Acknowledgements This work was carried out while AK was at Microsoft Research. 8

2 References

[1] A. Agarwal, D. Hsu, S. Kale, J. Langford, L. Li, and R. E. Schapire. Taming the monster: A fast and simple algorithm for contextual bandits. In ICML, 2014. [2] J.-Y. Audibert, S. Bubeck, and G. Lugosi. Regret in online combinatorial optimization. Math of OR, 2014. [3] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. SIAM Journal on Computing, 2002. [4] N. Cesa-Bianchi and G. Lugosi. Combinatorial bandits. JCSS, 2012. [5] O. Chapelle and Y. Chang. Yahoo! learning to rank challenge overview. In Yahoo! Learning to Rank Challenge, 2011. [6] W. Chen, Y. Wang, and Y. Yuan. Combinatorial multi-armed bandit: General framework and applications. In ICML, 2013. [7] W. Chu, L. Li, L. Reyzin, and R. E. Schapire. Contextual bandits with linear payoff functions. In AISTATS, 2011. [8] H. Daum? III, J. Langford, and D. Marcu. Search-based structured prediction. MLJ, 2009. [9] M. Dud?k, D. Hsu, S. Kale, N. Karampatziakis, J. Langford, L. Reyzin, and T. Zhang. Efficient optimal learning for contextual bandits. In UAI, 2011. [10] A. Gy?rgy, T. Linder, G. Lugosi, and G. Ottucs?k. The on-line shortest path problem under partial monitoring. JMLR, 2007. [11] D. J. Hsu. Algorithms for active learning. PhD thesis, University of California, San Diego, 2010. [12] S. Kale, L. Reyzin, and R. E. Schapire. Non-stochastic bandit slate problems. In NIPS, 2010. [13] B. Kveton, Z. Wen, A. Ashkan, and C. Szepesv?ri. Tight regret bounds for stochastic combinatorial semi-bandits. In AISTATS, 2015. [14] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In ICML, 2001. [15] J. Langford and T. Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In NIPS, 2008. [16] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In WWW, 2010. [17] MSLR. Mslr: Microsoft learning to rank dataset. projects/mslr/.

http://research.microsoft.com/en-us/

[18] G. Neu. Explore no more: Improved high-probability regret bounds for non-stochastic bandits. In NIPS, 2015. [19] L. Qin, S. Chen, and X. Zhu. Contextual combinatorial bandit and its application on diversified online recommendation. In ICDM, 2014. [20] A. Rakhlin and K. Sridharan. Bistro: An efficient relaxation-based method for contextual bandits. In ICML, 2016. [21] J. M. Robins. The analysis of randomized and nonrandomized AIDS treatment trials using a new approach to causal inference in longitudinal studies. In Health Service Research Methodology: A Focus on AIDS, 1989. [22] A. Swaminathan, A. Krishnamurthy, A. Agarwal, M. Dud?k, J. Langford, D. Jose, and I. Zitouni. Off-policy evaluation for slate recommendation. arXiv:1605.04812v2, 2016. [23] V. Syrgkanis, A. Krishnamurthy, and R. E. Schapire. Efficient algorithms for adversarial contextual learning. In ICML, 2016.

9