RELEAF: An Algorithm for Learning and Exploiting Relevance

Cem Tekin, *Member, IEEE*, Mihaela van der Schaar, *Fellow, IEEE*

***Abstract*—Recommender systems, medical diagnosis, network security, etc., require on-going learning and decision-making in real time. These – and many others – represent perfect examples of the opportunities and difficulties presented by Big Data: the available information often arrives from a variety of sources and has diverse features so that learning from all the sources may be valuable but integrating what is learned is subject to the curse of dimensionality. This paper develops and analyzes algorithms that allow efficient learning and decision-making while avoiding the curse of dimensionality. We formalize the information available to the learner/decision-maker at a particular time as a *context vector* which the learner should consider when taking actions. In general the context vector is very high dimensional, but in many settings, the most relevant information is embedded into only a few *relevant* dimensions. If these relevant dimensions were known in advance, the problem would be simple – but they are not. Moreover, the relevant dimensions may be different for different actions. Our algorithm *learns* the relevant dimensions for each action, and makes decisions based in what it has learned. Formally, we build on the structure of a contextual multi-armed bandit by adding and exploiting a *relevance relation*. We prove a general regret bound for our algorithm whose time order depends only on the maximum number of relevant dimensions among all the actions, which in the special case where the relevance relation is single-valued (a function), reduces to $\tilde{O}(T^{2(\sqrt{2}-1)})$; in the absence of a relevance relation, the best known contextual bandit algorithms achieve regret $\tilde{O}(T^{(D+1)/(D+2)})$, where D is the full dimension of the context vector. Our algorithm alternates between exploring and exploiting and does not require observing outcomes during exploitation (so allows for active learning). Moreover, during exploitation, suboptimal actions are chosen with arbitrarily low probability. Our algorithm is tested on datasets arising from network security and online news article recommendations.**

***Index Terms*—Contextual bandits, regret, dimensionality reduction, learning relevance, recommender systems, online learning, active learning.**

I. Introduction

The world is increasingly information-driven. Vast amounts of data are being produced by diverse sources and in diverse formats including sensor readings, physiological measurements, documents, emails, transactions, tweets, and audio or video files and many businesses and government institutions rely on these Big Data in their everyday operations. (Particular applications that have been discussed in the literature include recommender systems [2], neuroscience [3], network monitoring [4], surveillance [5], health monitoring [6], stock market prediction, intelligent driver assistance [7], etc.) To make the best use of these data, it is vital to learn from and respond to the streams of data continuously and in real time. Because data streams are heterogeneous and dynamically evolving over time in unknown and unpredictable ways, making decisions using these data streams online, at run-time, is known to be a very challenging problem [8], [9]. In this paper, we tackle these online Big Data challenges by exploiting a feature that is common to many applications: the data may have many dimensions, but the information that is most important for any given action is embedded into only a few *relevant* dimensions. In general, these relevant dimensions will be different for different actions and are not known in advance – so must be *learned*. We propose and analyze an algorithm that *learns* the relevant dimensions for each action, and makes decisions based in what it has learned.

Our structure builds on contextual multi-armed bandits. We formalize the information obtained from the data streams (perhaps after pre-processing) in terms of "context vectors". Context vectors characterize the information contained in the data generated by the process the learner wishes to control/act on such as the location, and/or data type information (e.g., features/characteristics/modality). The decision maker/learner receives the context vector and takes an action that generates a reward that depends (stochastically) on the context vector. Contexts, actions and rewards are generic terms; the specific meaning depends on the specific Big Data application. For instance, in a network security application [4], contexts are the features of the network packet, actions are the set of predictions about the type of network attacks and the reward is the accuracy of the prediction. In a recommender system [2], contexts are the characteristics (age, gender, purchase history, etc.) of the user, actions are items and the reward is the indicator function of the event that the user buys the item. The problem is to *learn* the rewards (or the distribution of rewards) generated by each action in each context. The context vector is typically high dimensional but in many applications the reward for a particular action will depend only on a few most relevant of these dimensions, embodied in a *relevance relation*. For an action set $\mathcal{A}$ and a type (dimension) set $\mathcal{D}$, the relevance relation is given by $\boldsymbol{\mathcal{R}} = \{\mathcal{R}(a)\}_{a\in\mathcal{A}}$, where $\mathcal{R}(a) \subset \mathcal{D}$. However, whether this is the case and if so, which dimensions are most relevant for a particular action, is not known in advance but must be learned, and decision-making must be adapted to this learning process.

Relevance relations arise naturally in many practical applications. For example, when treating patients with a particular

This work is partially supported by the grants NSF CNS 1016081 and AFOSR DDDAS.

C. Tekin and Mihaela van der Schaar are in Department of Electrical Engineering, UCLA, Los Angeles, CA, 90095. Email: cmtkn@ucla.edu, mihaela@ee.ucla.edu.

A preliminary version of this paper appeared in NIPS 2014 [1].

disease, many contexts may be available – the patients' age, weight, blood tests, imaging, medical history etc. - but often only a few of these contexts are relevant in choosing/not choosing a particular treatment or medication. For instance, surgery may be strongly contra-indicated in patients with clotting problems; drug therapies that require close monitoring may be strongly contra-indicated in patients who do not have committed care-givers, etc. Similarly, in recommender systems, a product recommendation may sometimes depend on many characteristics of the user – gender, occupation, history of past purchases etc. - but will often depend only (or most strongly) on a few characteristics – such as location and home-ownership.

Relevance allows us to avoid the curse of dimensionality: we show that regret bounds depend only on the number of *relevant dimensions*, i.e., D_{rel} – which is typically much less than the full number of dimensions. Our main contributions can be summarized as follows:

- We propose the *Relevance Learning with Feedback* (RELEAF) algorithm that alternates between exploration and exploitation phases. For the general case when $D_{\text{rel}} < D/2$, RELEAF achieves a regret bound of $\tilde{O}(T^{g(D_{\text{rel}})})$,[1] where $g(D_{\text{rel}}) \leq (2D_{\text{rel}}+3)/(2D_{\text{rel}}+4)$, which reduces to a regret bound of $\tilde{O}(T^{2(\sqrt{2}-1)})$ when the relevance relation is a function.
- We derive separate bounds on the regret incurred in exploration and exploitation phases. RELEAF only needs to observe the reward in exploration phases and hence, when observing rewards is costly, active learning can be performed by controlling reward feedback. RELEAF achieves the same time order of regret even when observing rewards is costly.
- The operation of RELEAF involves a confidence parameter, chosen by the user, which can be arbitrarily small. If confidence δ is chosen, then RELEAF will never select suboptimal actions in exploitation steps with probability at least $1-\delta$. This provides performance guarantees, which are important – perhaps vital – in many applications, such as medical treatment.

The rest of the paper is organized as follows. Related work is given in Section II. The problem is formalized in Section III. An algorithm that learns the *relevance relation* between actions and types of contexts is given in Section IV. Then, the regret bounds are proved for this algorithm. Numerical results on several real-world datasets are given in Section V. Finally, conclusions are given in Section VI.

II. Related Work

A. Multi-armed bandits

Our work is a new contextual bandit problem where relevance relations exist. Contextual bandit problems are studied by many others in the past [10]–[15]. The problem we consider in this paper is a special case of the Lipschitz contextual bandit problem [12], [13], where the only assumption is the existence of a known similarity metric between the expected rewards of actions for different contexts. The strengh of this model comes from the fact that there are no stochastic assumptions made on the context arrival process, and the benchmark which the regret is defined against selects the best action for each context. It is known that the lower bound on regret for this problem is $O(T^{(D+1)/(D+2)})$ [12], and there exists algorithms that achieve $\tilde{O}(T^{(D+1)/(D+2)})$ regret [12], [13].[2] Compared to these works, RELEAF only needs to observe rewards in explorations and has a regret whose time order is independent of D. Hence it can still learn the optimal actions fast enough in settings where observations are costly and the context vector is high dimensional. For instance, in Section IV-D we show that the regret of RELEAF is better than the bound of $\tilde{O}(T^{(D+1)/(D+2)})$ in [12], [13] for $D_{\text{rel}} \leq D/2-1$.

	Our work	[12], [13]	[2], [14]	[11], [15]
Relevance relation	yes	no	no	no
Context arrivals	arbitrary	arbitrary	arbitrary	i.i.d.
Reward-context relation	Lipschitz	Lipschitz	linear	joint i.i.d. process
Time order of the regret	depends on D_{rel}	depends on D	independent of D	independent of D

TABLE I
COMPARISON OF OUR WORK WITH OTHER WORK CONTEXTUAL BANDITS.

Another class of contextual bandit problems consider reward functions that are linear in the contexts [2], [14]. Due to this linearity assumption learning reduces to estimating the parameter vector corresponding to each arm, hence the regret bounds do not depend on the dimension of the context space. Several papers [11], [15] impose stochastic assumptions on the process that generates the contexts and the arm rewards. For instance assuming that the contexts and arm rewards are generated by an unknown i.i.d. process, regret independent of the dimension of the context space can be achieved.

The differences between our work and these prior works are summarized in Table I.

B. Dimensionality reduction

Dimensionality reduction methods are often used to find low dimensional representations of high dimensional context vectors (feature vectors) such that the information contained in the low dimensional representation is approximately equal to the information contained in the original context vector [16]. For instance, reduced-rank adaptive filtering [17]–[19] first projects feature vectors onto a lower dimensional subspace, and then adaptively adjusts the filter coefficients over time. In these works a low dimensional representation of the feature vector is learned based on the available data. Compared to this, in our work the relevant dimensions for each action can be different, hence a low dimensional representation that contains information about the rewards of all actions may not exist. An example is a relevance relation $\boldsymbol{\mathcal{R}}$ for which each action only has few relevant dimensions, i.e., $D_{\text{rel}} << D$, but $\bigcup_{a\in\mathcal{A}} \mathcal{R}(a) = D$.

C. Learning with limited number of observations

Examples of related works that consider limited observations while learning are KWIK learning [20], [21] and

[1] $O(\cdot)$ is the Big O notation, $\tilde{O}(\cdot)$ is the same as $O(\cdot)$ except it hides terms that have polylogarithmic growth.

[2] The bounds in [12], [13] are given in terms of *covering* and *zooming* dimensions of the problem instance, but they reduce to the Euclidian dimension for the set of assumptions we have in this paper.

label efficient learning [22]–[24]. For example, [21] considers a bandit model where the reward function comes from a parameterized family of functions and gives a bound on the average regret. An online prediction problem is considered in [22]–[24], where the predictor (action) lies in a class of linear predictors. The benchmark of the context is the best linear predictor. This restriction plays a crucial role in deriving regret bounds whose time order does not depend on D. Similar to these works, RELEAF can guarantee with a high probability that actions with suboptimality greater than a desired $\epsilon > 0$ will never be selected in exploitation steps. However, we do not have any assumptions on the form of the expected reward function other than the Lipschitz continuity.

For the special case when actions correspond to making predictions about the context vector (which is equal to the data stream for this special case), our problem is closely related to the problem of active learning. In this problem, obtaining the labels is costly, but the performance of the learning algorithm, i.e., rewards, can only be assessed through the labels, hence actively learning when to ask for the label becomes an important challenge. In stream-based active learning [25]–[28], the learner is provided with a stream of unlabeled instances. When an instance arrives, the learner decides to obtain the label or not. To the best of our knowledge there is no prior work in stream-based active learning that deals with learning relevance relations with sublinear bounds on the regret.

D. Ensemble learning

Numerous ensemble learning methods exists in the literature [4], [29]–[31]. These methods take predictions (actions) from a set of experts (e.g., base classifiers), and combine them with a specific rule to produce a final prediction (action). After the reward of all the actions are observed, the rule to combine the predictions of the experts is updated based on how good each individual expert had performed. The goal is to learn a combination rule such that even if the predictions' of the individual experts are not very accurate, the final prediction is accurate because it takes into account the "opinions" of all experts.

To evaluate the performance of ensemble learning methods analytically, the benchmark is usually taken to be the expert that achieves the highest total reward. Hence the "quality" of the regret bounds depends on the "quality" of the experts. In contrast, our regret bounds are with respect to the best benchmark (that only depends on context arrivals and reward distributions), and can be applied to settings without experts. Moreover, our algorithms work for the *bandit setting*, in which after an action is chosen, only its reward is revealed to the algorithm.

III. Problem Formulation and Preliminaries

A. Notation

For a vector $\boldsymbol{x}$, x_i denotes its ith component. Given a vector $\boldsymbol{v}$, $\boldsymbol{x}_{\boldsymbol{v}} := \{x_i\}_{i\in\boldsymbol{v}}$ denotes the components of $\boldsymbol{x}$ whose positions are in $\boldsymbol{v}$. The time index is $t = 1, 2, \ldots$. When referring to a time dependent variable we use subscript t as the rightmost subscript corresponding to that variable. For instance $\boldsymbol{x}_t$ denotes a vector at time t, $x_{i,t}$ denotes its ith component at time t, and $\boldsymbol{x}_{\boldsymbol{v},t}$ denotes the vector of its components that are in $\boldsymbol{v}$ at time t.

B. Problem formulation

$\mathcal{A}$ is the set of actions, D is the dimension of the context vector, $\mathcal{D} := \{1, 2, \ldots, D\}$ is the set of types, and $\boldsymbol{\mathcal{R}} = \{\mathcal{R}(a)\}_{a\in\mathcal{A}} : \mathcal{A} \to 2^{\mathcal{D}}$ is the (unknown) relevance relation, which maps every $a \in \mathcal{A}$ to a subset of $\mathcal{D}$. We call $D_{\text{rel}} = \max_{a\in\mathcal{A}} |\mathcal{R}(a)|$, the *relevance dimension*. When $D_{\text{rel}} = 1$, we say that $\boldsymbol{\mathcal{R}}$ is a *relevance function*. Elements of $\mathcal{D}$ are denoted by index i. Let $\mathcal{V}_K$, $1 \le K \le D$ be the set of K element subsets of $\mathcal{D}$. We call $\boldsymbol{v} \in \mathcal{V}_K$, a K-tuple of types.

At each time step $t = 1, 2, \ldots$, a context vector $\boldsymbol{x}_t$ arrives to the learner. After observing $\boldsymbol{x}_t$ the learner selects an action $a \in \mathcal{A}$, which results in a random reward $r_t(a, \boldsymbol{x}_t)$. The learner may choose to observe this reward by paying cost $c_O \ge 0$. The goal of the learner is to maximize the sum of the generated rewards minus costs of observations for any time horizon T.

Each $\boldsymbol{x}_t$ consists of D types of contexts, and can be written as $\boldsymbol{x}_t = (x_{1,t}, x_{2,t}, \ldots, x_{D,t})$ where $x_{i,t}$ is called the type i context. $\mathcal{X}_i$ denotes the space of type i contexts and $\mathcal{X} := \mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_D$ denotes the space of context vectors. At any t, we have $x_{i,t} \in \mathcal{X}_i$ for all $i \in \mathcal{D}$. All of our results hold for the case when $\mathcal{X}_i$ is a bounded subset of the real line. The number of elements in $\mathcal{X}_i$ can be finite or infinite. For the sake of notational simplicity we take $\mathcal{X}_i = [0, 1]$ for all $i \in \mathcal{D}$, since the values of context can be rescaled to lie in this range. Then, for the case when the actual context space is finite, $[0, 1]$ will be a superset of the context space. For a context vector $\boldsymbol{x}$, $\boldsymbol{x}_{\mathcal{R}(a)}$ denotes the vector of values of $\boldsymbol{x}$ corresponding to types $\mathcal{R}(a)$. The reward of action a for $\boldsymbol{x} = (x_1, x_2, \ldots, x_D) \in \mathcal{X}$, i.e., $r_t(a, \boldsymbol{x})$, is generated according to an i.i.d. process with distribution $F(a, \boldsymbol{x}_{\mathcal{R}(a)})$ with support in $[0, 1]$ and expected value $\mu(a, \boldsymbol{x}_{\mathcal{R}(a)})$. The learner does not know $F(a, \boldsymbol{x}_{\mathcal{R}(a)})$ and $\mu(a, \boldsymbol{x}_{\mathcal{R}(a)})$ for $a \in \mathcal{A}$, $\boldsymbol{x} \in \mathcal{X}$ a priori.

The following assumption gives a similarity structure between the expected reward of an action and the contexts of the type that is relevant to that action.

Assumption. ***(The Similarity Assumption)*** *For all $a \in \mathcal{A}$, $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$, we have $|\mu(a, \boldsymbol{x}_{\mathcal{R}(a)}) - \mu(a, \boldsymbol{x}'_{\mathcal{R}(a)})| \le L||\boldsymbol{x}_{\mathcal{R}(a)} - \boldsymbol{x}'_{\mathcal{R}(a)}||$, where $L > 0$ is the Lipschitz constant and $||\cdot||$ is the Euclidian norm.*

We assume that the learner knows the L given in the *Similarity Assumption*. While we need this assumption in order to derive our analytic bounds on the performance of the algorithm, as it is common in all contextual bandit algorithms [12], [13], our numerical results in Section V show that the proposed algorithm works well on real-world data sets for which this assumption may not hold. Given a context vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_D)$, the optimal action is $a^*(\boldsymbol{x}) := \arg\max_{a\in\mathcal{A}} \mu(a, \boldsymbol{x}_{\mathcal{R}(a)})$. In order to assess the learner's loss due to unknowns, we compare its performance with the performance of an *oracle benchmark* which knows $a^*(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{X}$. Let $\mu_t(a) := \mu(a, \boldsymbol{x}_{\mathcal{R}(a),t})$. The action chosen by the learner at time t is denoted by α_t. The learner

also decides whether to observe the reward or not, and this decision of the learner at time t is denoted by $\beta_t \in \{0,1\}$. If $\beta_t = 1$, then the learner chooses to observe the reward, else if $\beta_t = 0$, then the learner does not observe the reward. The learner's performance loss with respect to the oracle benchmark is defined as the regret, whose value at time T is given by

$$R(T) := \sum_{t=1}^{T} \mu_t(a^*(\boldsymbol{x}_t)) - \sum_{t=1}^{T} (\mu_t(\alpha_t) - c_O \beta_t). \quad (1)$$

Different from the definitions of regret in related works [12]–[14], there is an additional cost c_O, which is called the *active learning/exploration* cost. Hence the goal of the learner is to maximize its total reward while balancing the active learning costs incurred when observing the rewards. The algorithm we propose in this paper is able to achieve a given tradeoff between the two by actively controlling when to observe the rewards.

A regret that grows sublinearly in T, i.e., $O(T^\gamma)$, $\gamma < 1$, guarantees convergence in terms of the average reward, i.e., $R(T)/T \to 0$. We are interested in achieving sublinear growth with a rate only depending on D_{rel} independent of D.

IV. Online Learning of Relevance Relations

A. Relevance Learning with Feedback

In this section we propose the algorithm *Relevance LEArning with Feedback* (RELEAF), which learns the best action for each context vector by simultaneously learning the relevance relation, and then estimating the expected reward of each action based on the values of the contexts of the relevant types. The feedback, i.e., reward observations, is controlled based on the past context vector arrivals, in a way that the reward observations are only made for actions for which the uncertainty in the reward estimates are high for the current context vector. The controlled feedback feature allows RELEAF to operate as an active learning algorithm. RELEAF has a *relevance parameter* γ_{rel} which is the number of relevant types it will learn for each action. In order to have analytic bounds on the regret, it is required that $\gamma_{\text{rel}} \geq D_{\text{rel}}$. However, the numerical results in Section V show that even with $\gamma_{\text{rel}} = 1$, RELEAF performs very well on several real-world datasets. We assume that RELEAF knows D_{rel} but not $\boldsymbol{\mathcal{R}}$. Hence, in this paper we assume that RELEAF is run with $\gamma_{\text{rel}} = D_{\text{rel}}$. In theory, it is enough for RELEAF to know an upper bound $\bar{D}_{\text{rel}}$ on D_{rel}. Then, the regret of RELEAF will depend on $\bar{D}_{\text{rel}}$. Operation of RELEAF can be summarized as follows:

- Adaptively form partitions (composed of intervals) of the context space of each type in $\mathcal{D}$ and use them to learn the action rewards of similar context vectors together from the history of observations.
- For an action, form reward estimates for $2\gamma_{\text{rel}}$-tuple of intervals corresponding to $2\gamma_{\text{rel}}$-tuple of types. Based on the accuracy of these estimates, either choose to explore and observe the reward (by paying cost c_O for active learning) or choose to exploit the best estimated action (but do not observe the reward) for the current context vector.
- In order to estimate the expected rewards of the actions accurately, find the set of γ_{rel}-tuple of types relevant to each action a. For instance, a γ_{rel}-tuple of types $\boldsymbol{v} \in \mathcal{V}_{\gamma_{\text{rel}}}$ is relevant to action a if $\mathcal{R}(a) \subset \boldsymbol{v}$. Conclude that $\boldsymbol{v}$ is relevant to a if the variation of the reward estimates does not greatly exceed the natural variation of the expected reward of action a over the hypercube corresponding to $\boldsymbol{v}$ formed by intervals of type $i \in \boldsymbol{v}$ (calculated using *Similarity Assumption*).

Relevance Learning with Feedback (RELEAF):

Input: L, ρ, δ, γ_{rel}.
Initialization: $\mathcal{P}_{i,1} = \{[0,1]\}$, $i \in \mathcal{D}$. Run **Initialize**(i, $\mathcal{P}_{i,1}$, 1), $i \in \mathcal{D}$.
while $t \geq 1$ **do**
  Observe $\boldsymbol{x}_t$, find $\boldsymbol{p}_t$ that $\boldsymbol{x}_t$ belongs to.
  Set $\mathcal{U}_t := \bigcup_{i \in \mathcal{D}} \mathcal{U}_{i,t}$, where $\mathcal{U}_{i,t}$ (given in (5)), is the set of under explored actions for type i.
  if $\mathcal{U}_t \neq \emptyset$ **then**
    (**Explore**) $\beta_t = 1$, select α_t randomly from $\mathcal{U}_t$, observe $r_t(\alpha_t, \boldsymbol{x}_t)$.
    Update sample mean reward of α_t corresponding to $2\gamma_{\text{rel}}$-tuples of intervals: for all $\boldsymbol{q} \in Q_t$, given in (2). $\bar{r}^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q}, \alpha_t) = (S^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q}, \alpha_t)\bar{r}^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q}, \alpha_t) + r_t(\alpha_t, \boldsymbol{x}_t))/(S^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q}, \alpha_t) + 1)$.
    Update counters: for all $\boldsymbol{q} \in Q_t$, $S^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q}, \alpha_t) + +$.
  else
    (**Exploit**) $\beta_t = 0$, for each $a \in \mathcal{A}$ calculate the set of candidate relevant contexts $\text{Rel}_t(a)$ given in (6).
    for $a \in \mathcal{A}$ **do**
      if $\text{Rel}_t(a) = \emptyset$ **then**
        Randomly select $\hat{c}_t(a)$ from $\mathcal{V}_{\gamma_{\text{rel}}}$.
      else
        For each $i \in \text{Rel}_t(a)$, calculate $\text{Var}_t(\boldsymbol{v}, a)$ given in (7).
        Set $\hat{c}_t(a) = \arg\min_{\boldsymbol{v} \in \text{Rel}_t(a)} \text{Var}_t(\boldsymbol{v}, a)$.
      end if
      Calculate $\bar{r}^{\hat{c}_t(a)}(\boldsymbol{p}_{\hat{c}_t(a),t} a)$ as given in (8).
    end for
    Select $\alpha_t = \arg\max_{a \in \mathcal{A}} \bar{r}^{\hat{c}_t(a)}(\boldsymbol{p}_{\hat{c}_t(a),t} a)$.
  end if
  for $i \in \mathcal{D}$ **do**
    $N^i(p_{i,t}) + +$.
    if $N^i(p_{i,t}) \geq 2^{\rho l(p_{i,t})}$ **then**
      Create two new level $l(p_{i,t}) + 1$ intervals p, p' whose union gives $p_{i,t}$.
      $\mathcal{P}_{i,t+1} = \mathcal{P}_{i,t} \cup \{p, p'\} - \{p_{i,t}\}$.
      Run **Initialize**(i, $\{p, p'\}$, t).
    else
      $\mathcal{P}_{i,t+1} = \mathcal{P}_{i,t}$.
    end if
  end for
  $t = t + 1$
end while

Initialize(i, $\mathcal{B}$, t):
for $p \in \mathcal{B}$ **do**
  Set $N^i(p) = 0$, $\bar{r}^{(\boldsymbol{v}(\boldsymbol{q}),i)}((\boldsymbol{q}, p), a) = 0$, $S^{(\boldsymbol{v}(\boldsymbol{q}),i)}((\boldsymbol{q}, p), a) = 0$ for all $2\gamma_{\text{rel}}$-tuple of types $(\boldsymbol{v}(\boldsymbol{q}), i)$ that contain type i, for all $a \in \mathcal{A}$ such that $(\boldsymbol{q}, p) \in \boldsymbol{\mathcal{P}}_{(\boldsymbol{v}(\boldsymbol{q}),i),t}$.
end for

Fig. 1. Pseudocode for RELEAF.

In order to learn fast, RELEAF exploits the similarities between the context vectors of the relevant types[3] given in the *Similarity Assumption* to estimate the rewards of the actions.

[3]RELEAF only needs to know L but not $\boldsymbol{\mathcal{R}}$. Even if L is not known, it can use a slowly increasing function $\hat{L}(t)$ as an estimate for L so that a sublinear regret bound will hold for a time horizon T such that $\hat{L}(T) \geq L$.

The key to success of our algorithm is that this estimation is good enough if relevant tuples of types for each action are correctly identified. Since in Big Data applications D can be very large, learning the D_{rel}-tuple of types that is relevant to each action greatly increases the learning speed.

RELEAF adaptively forms the partition of the space for each type in $\mathcal{D}$, where the partition for the context space of type i at time t is denoted by $\mathcal{P}_{i,t}$. All the elements of $\mathcal{P}_{i,t}$ are disjoint intervals of $\mathcal{X}_i$ whose lengths are elements of the set $\{1, 2^{-1}, 2^{-2}, \ldots\}$.[4] An interval with length 2^{-l}, $l \geq 0$ is called a level l interval, and for an interval p, $l(p)$ denotes its level, $s(p)$ denotes its length. By convention, intervals are of the form $(a, b]$, with the only exception being the interval containing 0, which is of the form $[0, b]$.[5] Let $p_{i,t} \in \mathcal{P}_{i,t}$ be the interval that $x_{i,t}$ belongs to, $\boldsymbol{p}_t := (p_{1,t}, \ldots, p_{D,t})$ and $\boldsymbol{\mathcal{P}}_t := (\mathcal{P}_{1,t}, \ldots, \mathcal{P}_{D,t})$. For $\boldsymbol{v} \in \mathcal{V}_K$, $1 \leq K \leq D$, let $\boldsymbol{p}_{\boldsymbol{v},t}$ denote the elements of $\boldsymbol{p}_t$ corresponding to types in $\boldsymbol{v}$, and let $\boldsymbol{\mathcal{P}}_{\boldsymbol{v},t} = \times_{i \in \boldsymbol{v}} \mathcal{P}_{i,t}$.

The pseudocode of RELEAF is given in Fig. 1. RELEAF starts with $\mathcal{P}_{i,1} = \{\mathcal{X}_i\} = \{[0, 1]\}$ for each $i \in \mathcal{D}$. As time goes on and more contexts arrive for each type i, it divides $\mathcal{X}_i$ into smaller and smaller intervals. Then, these intervals are used to create $2\gamma_{\text{rel}}$-dimensional hypercubes corresponding to $2\gamma_{\text{rel}}$-tuples of types, and past observations corresponding to context vectors lying in these hypercubes are used to form sample mean reward estimates of the expected action rewards. The intervals are created in a way to balance the variation of the sample mean rewards due to the number of past observations that are used to calculate them and the variation of the expected rewards in each hypercube formed by the intervals. For each interval $p \in \mathcal{P}_{i,t}$, RELEAF keeps a counter for the number of type i context arrivals to p. When the value of this counter exceeds $2^{\rho l(p)}$, where $\rho > 0$ is an input of RELEAF called the *duration parameter*, p is destroyed and two level $l(p) + 1$ intervals, whose union gives p are created. For example, when $p_{i,t} = (k2^{-l}, (k+1)2^{-l}]$ for some $0 < k \leq 2^l - 1$ if $N_t^i(p_{i,t}) \geq 2^{\rho l}$, RELEAF sets

$$\begin{aligned}&\mathcal{P}_{i,t+1} = \mathcal{P}_{i,t} - \{p_{i,t}\} \\ &\cup \{(k2^{-l}, (k + 1/2)2^{-l}], ((k + 1/2)2^{-l}, (k + 1)2^{-l}]\}.\end{aligned}$$

Otherwise $\mathcal{P}_{i,t+1}$ remains the same as $\mathcal{P}_{i,t}$. It is easy to see that the lifetime of an interval increases exponentially in its duration parameter.

We next describe the control numbers RELEAF keeps for each type i, the counters and sample mean rewards RELEAF keeps for $2\gamma_{\text{rel}}$-tuples of intervals ($2\gamma_{\text{rel}}$-dimensional hypercubes) corresponding to a $2\gamma_{\text{rel}}$-tuple of types to determine whether to explore or exploit and how to exploit. Let $\mathcal{V}_K(i)$ be the set of K-tuples of types that contains type i. For each $\boldsymbol{v} \in \mathcal{V}_K(i)$, we have $i \in \boldsymbol{v}$.

Let $\mathcal{D}_{-\boldsymbol{v}} := \mathcal{D} - \{\boldsymbol{v}\}$. For type i, let $Q_{i,t} := \{\boldsymbol{p}_{\boldsymbol{v},t} : \boldsymbol{v} \in \mathcal{V}_{2\gamma_{\text{rel}}}(i)\}$ be the set of $2\gamma_{\text{rel}}$-tuples of intervals that includes an interval belonging to type i at time t, and let

$$Q_t := \bigcup_{i \in \mathcal{D}} Q_{i,t}. \tag{2}$$

To denote an element of $Q_{i,t}$ or Q_t we use index $\boldsymbol{q}$. For any $\boldsymbol{q} \in Q_t$, the tuple of types corresponding to the tuple intervals in $\boldsymbol{q}$ is denoted by $\boldsymbol{v}(\boldsymbol{q})$. For instance if $\boldsymbol{q} = (q_{i_1}, q_{i_2}, \ldots, q_{i_{2\gamma\text{rel}}})$, then $\boldsymbol{v}(\boldsymbol{q}) = (i_1, i_2, \ldots, i_{2\gamma\text{rel}})$. The decision to explore or exploit at time t is solely based on $\boldsymbol{p}_t$. For events $A_1, \ldots, A_K$, let $\mathrm{I}(A_1, \ldots, A_k)$ denote the indicator function of event $\bigcap_{k=1:K} A_k$. Let

$$S_t^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q}, a) := \sum_{t'=1}^{t} \mathrm{I}\left(\alpha_{t'} = a, \beta_{t'} = 1, \boldsymbol{p}_{\boldsymbol{v}(\boldsymbol{q}),t'} = \boldsymbol{q}\right),$$

be the number of times a is selected and the reward is observed when the context values corresponding to types $\boldsymbol{v}(\boldsymbol{q})$ are in $\boldsymbol{q}$ and $\boldsymbol{q} \in \boldsymbol{\mathcal{P}}_{\boldsymbol{v}(\boldsymbol{q}),t}$. Also let

$$\begin{aligned}&\bar{r}_t^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q}, a) \\ &:= \frac{\sum_{t'=1}^{t} r_{t'}(a, \boldsymbol{x}_{t'}) \mathrm{I}\left(\alpha_{t'} = a, \beta'_t = 1, \boldsymbol{p}_{\boldsymbol{v}(\boldsymbol{q}),t'} = \boldsymbol{q}\right)}{S_t^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q}, a)},\end{aligned}$$

be the sample mean reward of action a for $2\gamma_{\text{rel}}$-tuple of intervals $\boldsymbol{q}$.

At time t, RELEAF assigns a *control number* to each $i \in \mathcal{D}$ denoted by

$$D_{i,t} := \frac{2 \log(tD^*|\mathcal{A}|/\delta)}{(Ls(p_{i,t}))^2}, \tag{3}$$

where

$$D^* = \binom{D - 1}{2\gamma_{\text{rel}} - 1}. \tag{4}$$

This number depends on the cardinality of $\mathcal{A}$, the length of the active interval that type i context is in at time t and a *confidence parameter* $\delta > 0$, which controls the accuracy of sample mean reward estimates. $D_{i,t}$ is a sufficient number of reward observations from an action, which guarantees that the estimated reward for that action will be sufficiently close to the expected reward for the context at time t. By sufficiently close we mean that when i is the relevant type of context for the action, the difference between the true expected reward of that action and the estimated expected reward will be less than a constant factor of the length of the interval that contains the type i context due to the Similarity Assumption. The control function ensures that within each hypercube, the rate of exploration only increases logarithmically in time. It also guarantees that each action is explored at least $\sim 1/s(p_{i,t})^2$ times, which guarantees that the regret due to exploitations in each hypercube is small enough to achieve a sublinear regret bound (see Theorem 1).

Then, it computes the set of under-explored actions for type i as

$$\begin{aligned}\mathcal{U}_{i,t} := \Big\{&a \in \mathcal{A} : S_t^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q}, a) < D_{i,t} \\ &\text{for some } \boldsymbol{q} \in Q_i(t)\Big\},\end{aligned} \tag{5}$$

[4]Setting interval lengths to powers of 2 is for presentational simplicity. In general, interval lengths can be set to powers of any integer greater than 1.

[5]Endpoints of intervals will not matter in our analysis, so our results will hold even when the intervals have common endpoints.

and then, the set of under-explored actions as $\mathcal{U}_t := \bigcup_{i\in\mathcal{D}} \mathcal{U}_{i,t}$. The decision to explore or exploit is based on whether or not $\mathcal{U}_t$ is empty, as follows:

(i) If $\mathcal{U}_t \neq \emptyset$, RELEAF randomly selects an action $\alpha_t \in \mathcal{U}_t$ to explore, and observes its reward $r_t(\alpha_t, \boldsymbol{x}_t)$. Reward observation costs c_O, which is the active learning cost. Then, it updates the sample mean rewards and counters for all $\boldsymbol{q} \in Q_t$,

$$\bar{r}_{t+1}^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q},\alpha_t) = \frac{S_t^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q},\alpha_t)\bar{r}_{t+1}^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q},\alpha_t) + r_t(\alpha_t,\boldsymbol{x}_t)}{S_t^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q},\alpha_t)+1},$$
$$S_{t+1}^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q},\alpha_t) = S_t^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q},\alpha_t)+1.$$

(ii) If $\mathcal{U}_t = \emptyset$, RELEAF exploits by estimating the relevant γ_{rel}-tuple of types $\hat{c}_t(a)$ for each $a \in \mathcal{A}$ and forming sample mean reward estimates for action a based on $\hat{c}_t(a)$. It first computes the set of *candidate relevant tuples of types* for each $a \in \mathcal{A}$. For each $\boldsymbol{v} \in \mathcal{V}_{\gamma_{\text{rel}}}$, let $\mathcal{V}_{2\gamma_{\text{rel}}}(\boldsymbol{v})$ be the set of $2\gamma_{\text{rel}}$-tuples of types such that $\boldsymbol{v} \cap \boldsymbol{w} = \boldsymbol{v}$ for $\boldsymbol{w} \in \mathcal{V}_{2\gamma_{\text{rel}}}(\boldsymbol{v})$.

$$\text{Rel}_t(a) := \Big\{ \boldsymbol{v} \in \mathcal{V}_{\gamma_{\text{rel}}} : |\bar{r}_t^{\boldsymbol{w}}(\boldsymbol{p}_{\boldsymbol{w},t},a) - \bar{r}_t^{\boldsymbol{w}'}(\boldsymbol{p}_{\boldsymbol{w}',t},a)| \leq 3L\sqrt{\gamma_{\text{rel}}} \max_{i\in\boldsymbol{v}} s(p_{i,t}), \forall \boldsymbol{w},\boldsymbol{w}' \in \mathcal{V}_{2\gamma_{\text{rel}}}(\boldsymbol{v}) \Big\}. \quad (6)$$

The intuition is that if the tuple of types $\boldsymbol{v}$ contains the tuple of types $\mathcal{R}(a)$ that is relevant to a, then independent of the values of the contexts of the other types, the variation of the pairwise sample mean reward of a over $\boldsymbol{p}_{\boldsymbol{w},t}$ must be very close to the variation of the expected reward of a in $\boldsymbol{p}_{\boldsymbol{v},t}$ for $\boldsymbol{w} \in \mathcal{V}_{2D_{\text{rel}}}(\boldsymbol{v})$ in exploitation steps.

If $\text{Rel}_t(a)$ is empty, this implies that RELEAF failed to identify the relevant tuple of types, hence $\hat{c}_t(a)$ is randomly selected from $\mathcal{V}_{\gamma_{\text{rel}}}$. If $\text{Rel}_t(a)$ is nonempty, RELEAF computes the maximum variation

$$\text{Var}_t(\boldsymbol{v},a) := \max_{\boldsymbol{w},\boldsymbol{w}'\in\mathcal{V}_{2\gamma_{\text{rel}}}(\boldsymbol{v})} |\bar{r}_t^{\boldsymbol{w}}(\boldsymbol{p}_{\boldsymbol{w},t},a) - \bar{r}_t^{\boldsymbol{w}'}(\boldsymbol{p}_{\boldsymbol{w}',t}a)|, \quad (7)$$

for each $\boldsymbol{v} \in \text{Rel}_t(a)$. Then it sets $\hat{c}_t(a) = \min_{\boldsymbol{v}\in\text{Rel}_t(a)} \text{Var}_t(\boldsymbol{v},a)$. This way, whenever $\mathcal{R}(a) \subset \boldsymbol{v}$ for some $\boldsymbol{v} \in \text{Rel}_t(a)$, even if $\boldsymbol{v}$ is not selected as the estimated relevant tuple of types, the sample mean reward of a calculated based on the estimated relevant tuple of types will be very close to the sample mean of its reward calculated according to $\mathcal{R}(a)$. After finding the estimated relevant tuple of types $\hat{c}_t(a)$ for $a \in \mathcal{A}$, the sample mean rewards of the actions are computed as

$$\bar{r}_t^{\hat{c}_t(a)}(\boldsymbol{p}_{\hat{c}_t(a),t},a) := \frac{\sum_{\boldsymbol{w}\in\mathcal{V}_{2\gamma_{\text{rel}}}(\hat{c}_t(a))} \bar{r}_t^{\boldsymbol{w}}(\boldsymbol{p}_{\boldsymbol{w},t},a) S_t^{\boldsymbol{w}}(\boldsymbol{p}_{\boldsymbol{w},t},a)}{\sum_{\boldsymbol{w}\in\mathcal{V}_{2\gamma_{\text{rel}}}(\hat{c}_t(a))} S_t^{\boldsymbol{w}}(\boldsymbol{p}_{\boldsymbol{w},t},a)}. \quad (8)$$

Then, RELEAF selects

$$\alpha_t = \arg\max_{a\in\mathcal{A}} \bar{r}_t^{\hat{c}_t(a)}(\boldsymbol{p}_{\hat{c}_t(a),t},a).$$

Different from explorations, since the reward is not observed in exploitations, sample mean rewards and counters are not updated.

B. Why sample mean reward estimates for $2\gamma_{rel}$-tuple of intervals are required?

Assume that RELEAF knows D_{rel}, hence $\gamma_{\text{rel}} = D_{\text{rel}}$. Then, RELEAF computes sample mean reward estimates for $2D_{\text{rel}}$-tuples of intervals corresponding to different types and uses them to learn the action with the highest reward by learning the relevant D_{rel}-tuples of types. However, is it possible to learn the action with the highest reward by only forming sample mean estimates for D_{rel}-tuples of intervals? For instance consider the case when $D_{\text{rel}} = 1$ and the following greedy learning algorithm called Greedy-RELEAF, outlined as follows:

(i) Form sample mean reward estimates of each action a for each type $i \in \mathcal{D}$, i.e., $\bar{r}_t^i(p,a)$, $p \in \mathcal{P}_{i,t}$ based only on the context arrivals corresponding to type i; (ii) In exploitation steps choose the action with the highest sample mean reward over all sets of intervals in $\boldsymbol{p}_t$, i.e., $\arg\max_{a\in\mathcal{A}} \max_{i\in\mathcal{D}} \bar{r}_t^i(p_i(t),a)$. The following lemma shows that there exists a context arrival process for which the regret of Greedy-RELEAF will be linear in time.

Lemma 1. *Let $\mathcal{A} = \{a,b\}$, $\mathcal{D} = \{i,j\}$, $\mathcal{R}(a) = i$, $\mathcal{R}(b) = j$. $x_i(t) = x$ for all t and $x_j(t) = 1$ with probability 0.8 and $x_j(t) = 0$ with probability 0.2 for all t independently. Assume that $\mu(a,x) = 0.5$ and $\mu(b,x_j(t)) = x_j(t)$. Then, we have $R(T) = O(T)$.*

Proof: Given that Greedy-RELEAF explores sufficiently many times, at an exploitation step t when the context vector is $(x,0)$, we have

$$\text{P}\left(|\bar{r}_t^i(p_i(t),a) - 0.5| < 0.1, |\bar{r}_t^i(p_i(t),b) - 0.8| < 0.1, |\bar{r}_t^j(p_j(t),a) - 0.5| < 0.1\right) \geq 0.5$$

for any $p_i(t)$ containing x and $p_j(t)$ containing 0. At such a t Greedy-RELEAF will select action b with probability at least 0.5, resulting in an expected regret of at least 0.5^2. Assume that the context vector arrivals are such that $(x,0)$ appears in more than 50% of the time for all T large enough. Then, the regret of Greedy-RELEAF will be linear in T. ■

For the problem instance given in Lemma 1, RELEAF will calculate and compare sample mean rewards $\bar{r}_t^{i,j}((p_i(t),p_j(t)),a)$ for pairs of intervals corresponding to different types instead of directly forming sample mean rewards for intervals of each type; hence in exploitations it can identify that the type relevant to action a is i and action b is j with a very high probability. We will prove this in the following subsection by deriving a sublinear in time regret bound for RELEAF for the case when $D_{\text{rel}} = 1$. A general regret bound for $1 \leq D_{\text{rel}} < D/2$ is proven in our online technical report [32].

C. Regret analysis of RELEAF for $D_{rel} = 1$

In this section we derive analytical regret bounds for RELEAF. For simplicity of exposition, we prove our bounds for the special case when $D_{\text{rel}} = 1$, i.e., when the relevance relation is a function, and RELEAF is run with $\gamma_{\text{rel}} = D_{\text{rel}}$. Although $D_{\text{rel}} = 1$ is the simplest special case, our numerical results on real-world datasets in Section V shows that RELEAF performs very well with $\gamma_{\text{rel}} = 1$.

Let $\tau(T) \subset \{1, 2, \ldots, T\}$ be the set of time steps in which RELEAF exploits by time T. $\tau(T)$ is a random set which depends on context arrivals and the randomness of the action selection of RELEAF. The regret $R(T)$ defined in (1) can be written as a sum of the regret incurred during explorations (denoted by $R_{\mathrm{O}}(T)$) and the regret incurred during exploitations (denoted by $R_{\mathrm{I}}(T)$). Computing the two regrets separately gives more flexibility when choosing the parameter of RELEAF according to the objective of the learner. Although the definition of the regret in (1), allows us to write regret as $R_{\mathrm{O}}(T) + R_{\mathrm{I}}(T)$, the learner can set the parameters of RELEAF according to other objectives such as minimizing $R_{\mathrm{I}}(T)$ subject to $R_{\mathrm{O}}(T) \leq K$ for a fixed T and $K > 0$, or minimizing the time order of the regret when it is a more general function of regret in explorations and exploitations, i.e., $f(R_{\mathrm{O}}(T), R_{\mathrm{I}}(T))$. For instance, in an online prediction problem, if the cost of accessing the true label (exploration) is small, but the cost of making a prediction error in an exploitation step is very large, the learner can trade off to have higher rate of explorations.

The following theorem gives a bound on the regret of RELEAF in exploitation steps.

Theorem 1. *Let RELEAF run with relevance parameter $\gamma_{rel} = 1$, duration parameter $\rho > 0$, confidence parameter $\delta > 0$ and control numbers*

$$D_{i,t} := \frac{2\log(t|\mathcal{A}|D/\delta)}{(Ls(p_{i,t}))^2},$$

for $i \in \mathcal{D}$. Let $R_{inst}(t)$ be the instantaneous regret at time t, which is the loss in expected reward at time t due to not selecting $a^*(\boldsymbol{x}_t)$. When the relevance relation is such that $D_{rel} = 1$, then, with probability at least $1-\delta$, we have

$$R_{inst}(t) \leq 8L(s(p_{\mathcal{R}(\alpha_t),t}) + s(p_{\mathcal{R}(a^*(\boldsymbol{x}_t)),t})),$$

for all $t \in \tau(T)$, and the total regret in exploitation steps is bounded above by

$$\begin{aligned} R_I(T) &\leq 8L \sum_{t\in\tau(T)} (s(p_{\mathcal{R}(\alpha_t),t} + s(p_{\mathcal{R}(a^*(\boldsymbol{x}_t)),t})) \\ &\leq 16LD2^{2\rho}T^{\rho/(1+\rho)}, \end{aligned}$$

for arbitrary context vectors $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T$. Hence $R_I(T)/T = O(T^{-1/(1+\rho)})$, and $\lim_{T\to\infty} R_I(T) = 0$.

Proof: The proof is given in Appendix A. ∎

Theorem 1 provides both context arrival process dependent and worst case bounds on the exploitation regret of RELEAF. By choosing ρ arbitrarily close to zero, $R_{\mathrm{I}}(T)$ can be made $O(T^{\gamma})$ for any $\gamma > 0$. While this is true, the reduction in regret for smaller ρ not only comes from increased accuracy, but it is also due to the reduction in the number of time steps in which RELEAF exploits, i.e., $|\tau(T)|$. By definition, time t is an exploitation step if

$$\begin{aligned} S_t^{(i,j)}(p_{i,t}, p_{j,t}, a) &\geq \frac{2\log(t|\mathcal{A}|D/\delta)}{L^2 \min\{s(p_{i,t})^2, s(p_{j,t})^2\}} \\ &= \frac{2^{2\max\{l(p_{i,t}),l(p_{j,t})\}+1}\log(t|\mathcal{A}|D/\delta)}{L^2}, \end{aligned}$$

for all $\boldsymbol{q} = (p_{i,t}, p_{j,t}) \in Q_t$, $i, j \in \mathcal{D}$. This implies that for any $\boldsymbol{q} \in Q_{i,t}$ which has the interval with maximum level equal to l, $\tilde{O}(2^{2l})$ explorations are required before any exploitation can take place. Since the time a level l interval can stay active is $2^{\rho l}$, it is required that $\rho \geq 2$ so that $\tau(T)$ is nonempty.

The next theorem gives a bound on the regret of RELEAF in exploration steps.

Theorem 2. *Let RELEAF run with γ_{rel}, ρ, δ and $D_{i,t}$, $i \in \mathcal{D}$ values as stated in Theorem 1. When the relevance relation is such that $D_{rel} = 1$, we have*

$$\begin{aligned} R_O(T) \leq{}& \frac{960D^2(c_O+1)\log(T|\mathcal{A}|D/\delta)}{7L^2}T^{4/\rho} \\ &+ \frac{64D^2(c_O+1)}{3}T^{2/\rho}, \end{aligned}$$

with probability 1, for arbitrary context vectors $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T$. Hence $R_O(T)/T = O(T^{(4-\rho)/\rho})$, and $\lim_{T\to\infty} R_O(T) = 0$ for $\rho > 4$.

Proof: The proof is given in Appendix B. ∎

Based on the choice of the duration parameter ρ, which determines how long an interval will stay active, it is possible to get different regret bounds for explorations and exploitations. Any $\rho > 4$ will give a sublinear regret bound for both explorations and exploitations. The regret in exploitations increases in ρ while the regret in explorations decreases in ρ.

Theorem 3. *Let RELEAF run with γ_{rel}, δ and $D_{i,t}$, $i \in \mathcal{D}$ values as stated in Theorem 1 and $\rho = 2+2\sqrt{2}$. Then, the time order of exploration and exploitation regrets are balanced up to logarithmic orders. With probability at least $1-\delta$ we have both $R_I(T) = \tilde{O}(T^{2/(1+\sqrt{2})})$ and $R_O(T) = \tilde{O}(T^{2/(1+\sqrt{2})})$.*

Proof: The time order of the exploitation regret is increasing in ρ from the result of Theorem 1, and the time order of the exploration regret is decreasing in ρ from the result of Theorem 2. The time orders of both regrets are be balanced when $\rho/(1+\rho) = 4/\rho$, which gives the result. ∎

Another interesting case is when actions with suboptimality greater than $\epsilon > 0$ must never be chosen in any exploitation step by time T. When such a condition is imposed, RELEAF can start with partitions $\mathcal{P}_{i,1}$ that have intervals with high levels such that it explores more at the beginning to have more accurate reward estimates before any exploitation. The following theorem gives the regret bound of RELEAF for this case.

Theorem 4. *Let RELEAF run with relevance parameter $\gamma_{rel} = 1$, duration parameter $\rho > 0$, confidence parameter $\delta > 0$, control numbers*

$$D_{i,t} := \frac{2\log(t|\mathcal{A}|D/\delta)}{(Ls(p_{i,t}))^2},$$

and with initial partitions $\mathcal{P}_{i,1}$, $i \in \mathcal{D}$ consisting of intervals with levels $l_{\min} = \lceil \log_2(3L/(2\epsilon)) \rceil$. When the relevance relation is such that $D_{rel} = 1$, then, with probability $1-\delta$, we have

$$R_{inst}(t) \leq \epsilon,$$

for all $t \in \tau(T)$,

$$R_I(T) \leq 16L2^{2\rho}T^{\rho/(1+\rho)},$$

and

$$R_O(T) \leq \frac{81L^4}{\epsilon^4}\left(\frac{960D^2(c_O+1)\log(T|\mathcal{A}|D/\delta)}{7L^2}T^{4/\rho} + \frac{64D^2(c_O+1)}{3}T^{2/\rho}\right),$$

for arbitrary context vectors $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T$. *Bounds on* $R_I(T)$ *and* $R_O(T)$ *are balanced for* $\rho = 2 + 2\sqrt{2}$.

Proof: The proof is given in Appendix C. ∎

D. Regret bound for RELEAF for $D_{rel} < D/2$

Similar to the analysis in the previous subsection, RELEAF achieves sublinear in D_{rel} regret for any $D_{\text{rel}} < D/2$.

Theorem 5. *Let RELEAF run with relevance parameter* $\gamma_{rel} = D_{rel}$, *duration parameter* $\rho > 0$, *confidence parameter* $\delta > 0$ *and control numbers*

$$D_{i,t} := \frac{2\log(t|\mathcal{A}|D^*/\delta)}{(Ls(p_{i,t}))^2},$$

for $i \in \mathcal{D}$, *where* D^* *is given in (4). Then, with probability at least* $1-\delta$ *we have* $R_I(T) = \tilde{O}(T^{g(D_{rel})})$ *and* $R_O(T) = \tilde{O}(T^{g(D_{rel})})$, *where*

$$g(D_{rel}) := \frac{2 + 2D_{rel} + \sqrt{4D_{rel}^2 + 16D_{rel} + 12}}{4 + 2D_{rel} + \sqrt{4D_{rel}^2 + 16D_{rel} + 12}}.$$

Proof: The proof is given in our online technical report [32]. ∎

The bound on the regret given in Theorem 5 matches the bound in Theorem 3 for $D_{\text{rel}} = 1$.

Remark 1. *The regret bound in Theorem 5 is better than the* generic *regret bound* $\tilde{O}(T^{(D+1)/(D+2)})$ *for contextual bandit algorithms [12], [13] that does not exploit the existence of relevance relations when* $D_{rel} \leq D/2 - 1$.

V. Numerical Results

In this section, we numerically compare the performance of our learning algorithm with state–of–the–art learning techniques, including ensemble learning methods and other multi-armed bandit algorithms for two real-world datasets: (i) network intrusion detection, (ii) webpage recommendation. The purpose of simulations for the first dataset is to show that RELEAF can learn to make accurate prediction without the need of base classifiers, which are required by ensemble learners. The purpose of simulations for the second dataset is to show that RELEAF can learn to make accurate recommendations based on the context vectors of the users, by only observing the click information for the recommended webpage. An extended numerical results section, which includes additional information about the datasets and additional simulation results can be found in our online technical report [32].

A. Datasets

Network Intrusion (NI) [33]: The network intrusion dataset from UCI archive [33] consists of a series of TCP connection records, labeled either as normal connections or as attacks. The data consists of 42 features, and we take 15 of them as types of contexts. Taken features are normalized to lie in $[0, 1]$. The prediction action belongs to the set $\{attack, noattack\}$. Reward is 1 when the prediction is correct and 0 otherwise.
Webpage Recommendation (WR) [2]: This dataset contains webpage recommendations of Yahoo! Front Page which is an Internet news website. Each instance of this dataset consists of (i) IDs of the recommended items and their features, (ii) context vector of the user, and (iii) user click information. For a recommended webpage (item), reward is 1 if the user clicks on the item and 0 otherwise. The context vector for each user is generated by mapping a higher dimensional set of features of the user including features such as gender, age, purchase history, etc. to $[0, 1]^5$. The details of this mapping is given in [2]. We select 5 items and consider $T = 10000$ user arrivals.

B. Learning algorithms

Next we briefly summarize the algorithms considered in our evaluation:

RELEAF: Our algorithm given in Fig. 1 with control numbers $D_{i,t}$ divided by 5000 to reduce the number of explorations.[6]

RELEAF-ALL: Same as RELEAF except that reward of the selected action is observed in every time step. This version is useful when the reward of the selected action can be observed with no cost.

RELEAF-FO: Same as RELEAF except that it observes the rewards of all actions instead of the reward of the selected action. We refer to this version of our algorithm as RELEAF with full observation (RELEAF-FO).

Contextual zooming (CZ) [13]: This algorithm adaptively creates balls over the joint action and context space, calculates an index for each ball based on the history of selections of that ball, and at each time step selects an action according to the ball with the highest index that contains the action-context pair.

Hybrid-ϵ [34]: This algorithm is the contextual version of ϵ-greedy, which forms context-dependent sample mean rewards for the actions by considering the history of observations and decisions for groups of contexts that are similar to each other.

LinUCB [2]: This algorithm computes an index for each action by assuming that the expected reward of an action is a linear combination of different types of contexts. The action with the highest index is selected at each time step.

Ensemble Learning Methods Average Majority (AM) [4], Adaboost [29], Online Adaboost [30] and Blum's Variant of Weighted Majority (Blum) [31]: The goal of ensemble learning is to create a strong (high accuracy) classifier by combining predictions of base classifiers. Hence all these methods require

[6]The theoretical bounds are proven to hold for worst-case context vector arrivals and reward distributions. In practice, the relevance relation and the order of action rewards are identified correctly with much less explorations.

base classifiers (trained a priori) that produce predictions (or actions) based on the context vector.

AM simply follows the prediction of the majority of the classifiers and does not perform active learning. Adaboost is trained a priori with 1500 instances, whose labels are used to compute the weight vector. Its weight vector is fixed during the test phase (it is not learning online); hence no active learning is performed during the test phase. In contrast, Online Adaboost always receives the true label at the end of each time slot. It uses a time window of 1000 past observations to retrain its weight vector. Similar to Online Adaboost, Blum also learns its weight vector online. The key differences between our algorithm and the methods that we compare against are given in Table II.

C. Network intrusion simulations

In this section we compare the performance of RELEAF, RELEAF-ALL and RELEAF-FO with other learning methods described in Section V-B. For the ensemble learning methods, the base classifiers are logistic regression classifiers, each trained with 5000 different instances from the NI. Comparison of performances in terms of the error rate is given in Table III. We see that RELEAF-FO has the lowest error rate at 0.68%, more than two times better than any of the ensemble learning methods. All the ensemble learning methods we compare against use classifiers to make predictions, and these classifiers require a priori training. In contrast, RELEAF and RELEAF-FO do not require any a priori training, learn online and require only a small number of label observations (i.e. they can perform active learning).

CZ performs very poorly in this simulation because its learning rate is sensitive to Lipschitz constant that is given as an input to the algorithm which we set equal to 0.5. Numerical results related to the performance of CZ and RELEAF for different L values can be found in our online technical report [32]. LinUCB performs the best in terms of the overall rate of error, but if we consider the error rate of RELEAF in exploitations it is better than LinUCB. This highlights the finding of Theorem 1 regarding RELEAF, which states that highly suboptimal actions are not chosen in exploitations with a high probability.

Algorithm	error %	exploitation error %	number of label observations
AM	3.07	N/A	0
Adaboost	3.1	N/A	1500
Online Adaboost	2.25	N/A	all
Blum	1.64	N/A	all
CZ	53	N/A	all
Hybrid-ϵ	8.8	N/A	all
LinUCB	0.27	N/A	all
RELEAF	1.19	0.24	398
RELEAF-ALL	1.07	0.22	all
RELEAF-FO	0.68	0.24	229

TABLE III
COMPARISON OF THE ERROR RATES OF RELEAF-FO WITH ENSEMBLE LEARNING METHODS FOR NETWORK INTRUSION DATASET.

Abbreviation	CTR
CZ	3.79
Hybrid-ϵ	6.41
LinUCB	6.06
RELEAF-ALL	6.62

TABLE IV
COMPARISON OF THE CLICK THROUGH RATES (CTRS) OF RELEAF, CZ, HYBRID-ϵ AND LINUCB FOR WEBPAGE RECOMMENDATION DATASET.

D. Webpage recommendation simulations

In this dataset only the click behavior of the user for the recommended item is observed. Moreover, it is reasonable to assume that the click behavior feedback is always available (no costly observations). The ensemble learning methods require availability of experts recommending actions and full reward feedback including the rewards of the actions that are not selected, to update the weights of the experts, hence they are not suitable for this dataset. In contrast, multi-armed bandit methods are more suitable since only the feedback about the reward of the chosen action is required. Hence we only compare RELEAF-ALL, CZ, LinUCB and Hybrid-ϵ for this dataset. We compare the click through rates (CTRs), i.e., average number of times the recommended item is clicked, of all algorithms in Table IV. We observe that RELEAF-ALL has the highest CTR.

VI. CONCLUSION

In this paper we formalized the problem of learning the best action (prediction, recommendation etc.) to be taken based on the current streaming Big Data by online learning the relevance relation between types of contexts and actions. We proposed an algorithm that (i) has sublinear regret with time order independent of D, (ii) only requires reward observations in explorations, (iii) for any $\epsilon > 0$, does not select any ϵ suboptimal actions in exploitations with a high probability. We illustrated the properties of the proposed algorithm via extensive numerical simulations on real-data, showed that it achieves high average reward and identifies the set of relevant types. The proposed algorithm can be used in a variety of application (including applications requiring active learning) such as medical diagnosis, recommender systems and stream mining problems. An interesting future research direction is learning both relevant types of contexts and relevant type of actions for multi-armed bandit problems with high dimensional action and context spaces.

REFERENCES

[1] C. Tekin and M. van der Schaar, "Discovering, learning and exploiting relevance," in *NIPS*, 2014.

[2] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proc.of the 19th International Conference on World Wide Web*. ACM, 2010, pp. 661–670.

[3] J. Djolonga, A. Krause, and V. Cevher, "High-dimensional gaussian process bandits," in *NIPS*, 2013, pp. 1025–1033.

[4] J. Gao, W. Fan, and J. Han, "On appropriate assumptions to mine data streams: Analysis and practice," in *Seventh IEEE International Conference on Data Mining (ICDM)*, 2007, pp. 143–152.

[5] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 8, pp. 747–757, 2000.

Algorithm	Base classifiers	Prior training	Online Learning	Active learning
AM [4]	required	no	no	no
Adaboost [29]	required	required	no	no
Online Adaboost [29], **Blum** [31]	required	required	yes	no
CZ [13], **Hybrid-ϵ** [34] , **LinUCB** [2]	not required	not required	yes	no
RELEAF	not required	not required	yes	yes

TABLE II
PROPERTIES OF RELEAF, ENSEMBLE LEARNING METHODS AND OTHER CONTEXTUAL BANDIT ALGORITHMS.

[6] V. S. Tseng, C.-H. Lee, and J. Chia-Yu Chen, "An integrated data mining system for patient monitoring with applications on asthma care," in *Computer-Based Medical Systems, 2008. CBMS'08. 21st IEEE International Symposium on*. IEEE, 2008, pp. 290–292.

[7] S. Avidan, "Support vector tracking," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 8, pp. 1064–1072, 2004.

[8] R. Ducasse, D. S. Turaga, and M. van der Schaar, "Adaptive topologic optimization for large-scale stream mining," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 4, no. 3, pp. 620–636, 2010.

[9] "Ibm smarter planet project," *http://www-03.ibm.com/software/products/en/infosphere-streams*.

[10] E. Hazan and N. Megiddo, "Online learning with prior knowledge," in *Learning Theory*. Springer, 2007, pp. 499–513.

[11] J. Langford and T. Zhang, "The epoch-greedy algorithm for contextual multi-armed bandits," *NIPS*, vol. 20, pp. 1096–1103, 2007.

[12] T. Lu, D. Pál, and M. Pál, "Contextual multi-armed bandits," in *International Conference on Artificial Intelligence and Statistics*, 2010, pp. 485–492.

[13] A. Slivkins, "Contextual bandits with similarity information," in *COLT*, 2011.

[14] W. Chu, L. Li, L. Reyzin, and R. E. Schapire, "Contextual bandits with linear payoff functions," in *International Conference on Artificial Intelligence and Statistics*, 2011, pp. 208–214.

[15] M. Dudik, D. Hsu, S. Kale, N. Karampatziakis, J. Langford, L. Reyzin, and T. Zhang, "Efficient optimal learning for contextual bandits," *arXiv preprint arXiv:1106.2369*, 2011.

[16] A. Srivastava and X. Liu, "Tools for application-driven linear dimension reduction," *Neurocomputing*, vol. 67, pp. 136–160, 2005.

[17] A. M. Haimovich and Y. Bar-Ness, "An eigenanalysis interference canceler," *Signal Processing, IEEE Transactions on*, vol. 39, no. 1, pp. 76–84, 1991.

[18] Y. Hua, M. Nikpour, and P. Stoica, "Optimal reduced-rank estimation and filtering," *Signal Processing, IEEE Transactions on*, vol. 49, no. 3, pp. 457–469, 2001.

[19] R. C. de Lamare and R. Sampaio-Neto, "Adaptive reduced-rank processing based on joint and iterative interpolation, decimation, and filtering," *Signal Processing, IEEE Transactions on*, vol. 57, no. 7, pp. 2503–2514, 2009.

[20] L. Li, M. L. Littman, T. J. Walsh, and A. L. Strehl, "Knows what it knows: a framework for self-aware learning," *Machine Learning*, vol. 82, no. 3, pp. 399–443, 2011.

[21] K. Amin, M. Kearns, M. Draief, and J. D. Abernethy, "Large-scale bandit problems and {KWIK} learning," in *ICML*, 2013, pp. 588–596.

[22] N. Cesa-Bianchi, C. Gentile, and F. Orabona, "Robust bounds for classification via selective sampling," in *ICML*, 2009, pp. 121–128.

[23] S. M. Kakade, S. Shalev-Shwartz, and A. Tewari, "Efficient bandit algorithms for online multiclass prediction," in *ICML*, 2008, pp. 440–447.

[24] E. Hazan and S. Kale, "Newtron: an efficient bandit algorithm for online multiclass prediction." in *NIPS*, 2011, pp. 891–899.

[25] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby, "Selective sampling using the query by committee algorithm," *Machine learning*, vol. 28, no. 2-3, pp. 133–168, 1997.

[26] N. Cesa-Bianchi, G. Lugosi, and G. Stoltz, "Minimizing regret with label efficient prediction," *Information Theory, IEEE Transactions on*, vol. 51, no. 6, pp. 2152–2162, 2005.

[27] O. Dekel, C. Gentile, and K. Sridharan, "Selective sampling and active learning from single and multiple teachers," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 2655–2697, 2012.

[28] N. Zolghadr, G. Bartók, R. Greiner, A. György, and C. Szepesvári, "Online learning with costly features and labels," in *Proc. NIPS*, 2013, pp. 1241–1249.

[29] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," in *Computational Learning Theory*. Springer, 1995, pp. 23–37.

[30] W. Fan, S. J. Stolfo, and J. Zhang, "The application of adaboost for distributed, scalable and on-line learning," in *Proc. of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1999, pp. 362–366.

[31] A. Blum, "Empirical support for winnow and weighted-majority algorithms: Results on a calendar scheduling domain," *Machine Learning*, vol. 26, no. 1, pp. 5–23, 1997.

[32] C. Tekin and M. van der Schaar, "RELEAF: An algorithm for learning and exploiting relevance," *http://arxiv.org/pdf/1502.01418v1.pdf*, 2015.

[33] K. Bache and M. Lichman, "UCI machine learning repository," http://archive.ics.uci.edu/ml, University of California, Irvine, School of Information and Computer Sciences, 2013.

[34] D. Bouneffouf, A. Bouzeghoub, and A. L. Gançarski, "Hybrid-ε-greedy for mobile context-aware recommender system," in *Advances in Knowledge Discovery and Data Mining*. Springer, 2012, pp. 468–479.

APPENDIX A
PROOF OF THEOREM 1

Let $A := |\mathcal{A}|$. We first define a sequence of events which will be used in the analysis of the regret of RELEAF. For $p \in \mathcal{P}_{\mathcal{R}(a),t}$, Let $\pi(a,p) = \mu(a, x^*_{\mathcal{R}(a)}(p))$, where $x^*_{\mathcal{R}(a)}(p)$ is the context at the geometric center of p. For $j \in \mathcal{D}_{-\mathcal{R}(a)}$, let

$$\text{INACC}_t(a,j) := \left\{ |\bar{r}_t^{(\mathcal{R}(a),j)}((p_{\mathcal{R}(a),t}, p_{j,t}), a) - \pi(a, p_{\mathcal{R}(a),t})| > \frac{3}{2} Ls(p_{\mathcal{R}(a),t}) \right\},$$

be the event that the pairwise sample mean corresponding to pair $(\mathcal{R}(a), j)$ of types is *inaccurate* for action a. Let $\text{ACC}_t(a) := \bigcap_{j \in \mathcal{D}_{-\mathcal{R}(a)}} \text{INACC}_t(a,j)^C$, be the event that all pairwise sample means corresponding to pairs $(\mathcal{R}(a), j)$, $j \in \mathcal{D}_{-\mathcal{R}(a)}$ are accurate. Consider $t \in \tau(T)$. Let $\text{WNG}_t(a) := \{\mathcal{R}(a) \notin \text{Rel}_t(a)\}$, be the event that the type relevant to action a is not in the set of candidate relevant types, and $\text{WNG}_t := \bigcup_{a \in \mathcal{A}} \text{WNG}_t(a)$, be the event that the type relevant to some action a is not in the set of candidate relevant types of that action. Finally, let $\text{CORR}_T := \bigcap_{t \in \tau(T)} \text{WNG}_t^C$, be the event that the relevant types for all actions are in the set of candidate relevant types at all exploitation steps.

We first prove several lemmas related to Theorem 1. The next lemma gives a lower bound on the probability of CORR_T.

Lemma 2. *For RELEAF, for all $a \in \mathcal{A}$, $t \in \tau(T)$, we have $\text{P}(\textit{INACC}_t(a,j)) \leq \frac{2\delta}{ADt^4}$. for all $j \in \mathcal{D}_{-\mathcal{R}(a)}$, and $\text{P}(\textit{CORR}_T) \geq 1 - \delta$ for any T.*

Proof: For $t \in \tau(T)$, we have $\mathcal{U}_t = \emptyset$, hence

$$S_t^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q}, a) \geq \frac{2\log(tAD/\delta)}{(Ls(p_{\mathcal{R}(a),t}))^2},$$

for all $a \in \mathcal{A}$, $\boldsymbol{q} \in Q_i(t)$ and $i \in \mathcal{D}$. Due to the *Similarity Assumption*, since rewards in $\bar{r}_t^{(\mathcal{R}(a),j)}((p_{\mathcal{R}(a),t}, p_{j,t}), a)$ are sampled from distributions with mean between $[\pi(a, p_{\mathcal{R}(a),t}) -$

$\frac{L}{2}s(p_{\mathcal{R}(a),t}), \pi(a, p_{\mathcal{R}(a),t}) + \frac{L}{2}s(p_{\mathcal{R}(a),t})]$, using a Chernoff bound we get

$$\begin{aligned}\mathrm{P}(\mathrm{INACC}_t(a,j)) &\leq 2\exp\left(-2(Ls(p_{\mathcal{R}(a),t}))^2\frac{2\log(tAD/\delta)}{(Ls(p_{\mathcal{R}(a),t}))^2}\right)\\ &\leq 2\delta/(ADt^4).\end{aligned}$$

We have $\mathrm{WNG}_t(a) \subset \bigcup_{j\in\mathcal{D}_{-\mathcal{R}(a)}} \mathrm{INACC}_t(a,j)$. Thus

$$\mathrm{P}(\mathrm{WNG}_t(a)) \leq 2\delta/(At^4), \text{ and } \mathrm{P}(\mathrm{WNG}_t) \leq 2\delta/t^4.$$

This implies that

$$\begin{aligned}\mathrm{P}(\mathrm{CORR}_T^C) &\leq \sum_{t\in\tau(T)} \mathrm{P}(\mathrm{WNG}_t)\\ &\leq \sum_{t\in\tau(T)} \frac{2\delta}{t^4} \leq \sum_{t=3}^{\infty}\frac{2\delta}{t^4} \leq \delta.\end{aligned}$$

■

Lemma 3. *When CORR_T happens we have for all $t \in \tau(T)$*

$$|\bar{r}_t^{\hat{c}_t(a)}(p_{\hat{c}_t(a),t}, a) - \mu(a, x_{\mathcal{R}(a),t})| \leq 8Ls(p_{\mathcal{R}(a),t}).$$

Proof: From Lemma 2, CORR_T happens when

$$|\bar{r}_t^{(\mathcal{R}(a),j)}((p_{\mathcal{R}(a),t}, p_{j,t}), a) - \pi(a, p_{\mathcal{R}(a),t})| \leq \frac{3L}{2}s(p_{\mathcal{R}(a),t}),$$

for all $a \in \mathcal{A}$, $j \in \mathcal{D}_{-\mathcal{R}(a)}$, $t \in \tau(T)$. Since $|\mu(a, x_{\mathcal{R}(a),t}) - \pi(a, p_{\mathcal{R}(a),t})| \leq Ls(p_{\mathcal{R}(a),t})/2$, we have

$$|\bar{r}_t^{(\mathcal{R}(a),j)}((p_{\mathcal{R}(a),t}, p_{j,t}), a) - \mu(a, x_{\mathcal{R}(a),t})| \leq 2Ls(p_{\mathcal{R}(a),t}), \tag{A.1}$$

for all $a \in \mathcal{A}$, $j \in \mathcal{D}_{-\mathcal{R}(a)}$, $t \in \tau(T)$. Consider $\hat{c}_t(a)$. Since it is chosen from $\mathrm{Rel}_t(a)$ as the type with the minimum variation, we have on the event CORR_T

$$\begin{aligned}&|\bar{r}_t^{(\hat{c}_t(a),k)}((p_{\hat{c}_t(a),t}, p_{k,t}), a) - \bar{r}_t^{(\hat{c}_t(a),j)}((p_{\hat{c}_t(a),t}, p_{j,t}), a)|\\ &\leq 3Ls(p_{\mathcal{R}(a),t}),\end{aligned}$$

for all $j, k \in \mathcal{D}_{-\hat{c}_t(a)}$. Hence we have

$$\begin{aligned}&|\bar{r}_t^{\mathcal{R}(a)}(p_{\mathcal{R}(a),t}, a) - \bar{r}_t^{\hat{c}_t(a)}(p_{\hat{c}_t(a),t}, a)|\\ &\leq \max_{k,j}\Big\{|\bar{r}_t^{(\mathcal{R}(a),k)}((p_{\mathcal{R}(a),t}, p_{k,t}), a)\\ &\quad -\bar{r}_t^{(\hat{c}_t(a),j)}((p_{\hat{c}_t(a),t}, p_{j,t}), a)|\Big\}\\ &\leq \max_{k,j}\Big\{|\bar{r}_t^{(\mathcal{R}(a),k)}((p_{\mathcal{R}(a),t}, p_{k,t}), a)\\ &\quad -\bar{r}_t^{(\mathcal{R}(a),\hat{c}_t(a))}((p_{\mathcal{R}(a),t}, p_{\hat{c}_t(a),t}), a)|\\ &\quad +|\bar{r}_t^{(\hat{c}_t(a),\mathcal{R}(a))}((p_{\hat{c}_t(a),t}, p_{\mathcal{R}(a),t}), a)\\ &\quad -\bar{r}_t^{(\hat{c}_t(a),j)}((p_{\hat{c}_t(a),t}, p_{j,t}), a)|\Big\}\\ &\leq 6Ls(p_{\mathcal{R}(a),t}).\end{aligned} \tag{A.2}$$

Combining (A.1) and (A.2), we get

$$|\bar{r}_t^{\hat{c}_t(a)}(p_{\hat{c}_t(a),t}, a) - \mu(a, x_{\mathcal{R}(a),t})| \leq 8Ls(p_{\mathcal{R}(a),t}).$$

■

Since for $t \in \tau(T)$, $\alpha_t = \arg\max_{a\in\mathcal{A}} \bar{r}_t^{\hat{c}_t(a)}(p_{\hat{c}_t(a),t}, a)$, using the result of Lemma 3, we conclude that

$$\begin{aligned}&\mu_t(\alpha_t)\\ &\geq \mu_t(a^*(\boldsymbol{x}_t)) - 8L(s(p_{\mathcal{R}(\alpha_t),t}) + s(p_{\mathcal{R}(a^*(\boldsymbol{x}_t)),t})),\end{aligned}$$

Thus, the regret in exploitation steps is

$$\begin{aligned}&8L\sum_{t\in\tau(T)}\big(s(p_{\mathcal{R}(\alpha_t),t}) + s(p_{\mathcal{R}(a^*(\boldsymbol{x}_t)),t})\big)\\ &\leq 16L\sum_{t\in\tau(T)}\max_{a\in\mathcal{A}} s(p_{\mathcal{R}(a),t}) \leq 16L\sum_{t\in\tau(T)}\sum_{i\in\mathcal{D}} s(p_{i,t})\\ &\leq 16LD\max_{i\in\mathcal{D}}\left(\sum_{t\in\tau(T)} s(p_{i,t})\right).\end{aligned}$$

We know that as time goes on RELEAF uses partitions with smaller and smaller intervals, which reduces the regret in exploitations. In order to bound the regret in exploitations for any sequence of context arrivals, we assume a worst case scenario, where context vectors arrive such that at each t, the active interval that contains the context of each type has the maximum possible length. This happens when for each type i contexts arrive in a way that all level l intervals are split to level $l+1$ intervals, before any arrivals to these level $l+1$ intervals happen, for all $l = 0, 1, 2, \ldots$. This way it is guaranteed that the length of the interval that contains the context for each $t \in \tau(T)$ is maximized. Let $l_{\max}$ be the level of the maximum level interval in $\mathcal{P}_i(T)$. For the worst case context arrivals we must have

$$\sum_{l=0}^{l_{\max}-1} 2^l 2^{\rho l} < T \Rightarrow l_{\max} < 1 + \log_2 T/(1+\rho),$$

since otherwise maximum level hypercube will have level larger than $l_{\max}$. Hence we have

$$\begin{aligned}&16LD\max_{i\in\mathcal{D}}\left(\sum_{t\in\tau(T)} s(p_{i,t})\right) \leq 16LD\sum_{l=0}^{1+\log_2 T/(1+\rho)} 2^l 2^{\rho l} 2^{-l}\\ &= 16LD\sum_{l=0}^{1+\log_2 T/(1+\rho)} 2^{\rho l} \leq 16LD2^{2\rho}T^{\rho/(1+\rho)}.\end{aligned}$$

Appendix B
Proof of Theorem 2

Recall that time t is an exploitation step only if $\mathcal{U}_t = \emptyset$. In order for this to happen we need $S_t^{\boldsymbol{v}(\boldsymbol{q})}(\boldsymbol{q}, a) \geq D_{i,t}$ for all $\boldsymbol{q} \in Q_i(t)$. There are $D(D-1)$ type pairs. Whenever action a is explored, all the counters for these $D(D-1)$ type pairs are updated for the pairs of intervals that contain types of contexts present at time t, i.e. $\boldsymbol{q} \in Q_t$. Now consider a hypothetical scenario in which instead of updating the counters of all $\boldsymbol{q} \in Q_t$, the counter of only one of the randomly selected interval pair is updated. Clearly, the exploration regret of this hypothetical scenario upper bounds the exploration regret of the original scenario. In this scenario for any $p_i \in \mathcal{P}_{i,t}$, $p_j \in \mathcal{P}_{j,t}$, we have

$$S_t^{(i,j)}((p_i, p_j), a) \leq \frac{2\log(tAD/\delta)}{L^2\min(s(p_i), s(p_j))^2} + 1.$$

We can go one step further and consider a second hypothetical scenario where there is only two types i and j, for which the actual regret at every exploration step is magnified

(multiplied) by $D(D-1)$. The maximum possible exploration regret of the second scenario (for the worst case of type i and j context arrivals) upper bounds the exploration regret of the first scenario. Hence, we bound the regret of the second scenario. Let $l_{\max}$ be the maximum possible level for an active interval for type i by time T. We must have $\sum_{l=0}^{l_{\max}-1} 2^{\rho l} < T$, which implies that $l_{\max} < 1 + \log_2 T/\rho$. Next, we consider all pairs of intervals for which the minimum interval has level l. For each type j interval p_j that has level l, there exists no more than $\sum_{k=l}^{l_{\max}} 2^k$ type i intervals that have lengths greater than or equal to l. Consider a level k type i interval p_i such that $l \leq k < 1 + \log_2 T/\rho$. Then for the pair of intervals (p_i, p_j) the exploration regret is bounded by $(c_O+1)\left(2\log(tAD/\delta)/(2^{-2k}L^2)+1\right)$. Hence, the worst case exploration regret is bounded by

$$
\begin{aligned}
R_{\text{O}}(T) &\leq (c_O+1)D^2\left(2\sum_{l=0}^{1+\log_2 T/\rho} 2^l \sum_{k=l}^{1+\log_2 T/\rho} 2^k\left(\frac{2\log(tAD/\delta)}{2^{-2k}L^2}+1\right)\right)\\
&= (c_O+1)D^2\left(\frac{4\log(tAD/\delta)}{L^2}\sum_{l=0}^{1+\log_2 T/\rho} 2^l \sum_{k=l}^{1+\log_2 T/\rho} 2^{3k}\right.\\
&\left.+2\sum_{l=0}^{1+\log_2 T/\rho} 2^l \sum_{k=l}^{1+\log_2 T/\rho} 2^k\right)\\
&\leq \frac{4D^2(c_O+1)\log(tAD/\delta)}{L^2}\times\frac{240}{7}T^{4/\rho}\\
&+\frac{64D^2(c_O+1)}{3}T^{2/\rho}.
\end{aligned}
$$

Appendix C
Proof of Theorem 4

To achieve ϵ-optimality in every exploitation step it is sufficient to have

$$
\begin{aligned}
&\text{INACC}_t(a,j)^C = \left\{|\bar{r}_t^{(\mathcal{R}(a),j)}((p_{\mathcal{R}(a),t},p_{j,t}),a)-\pi(a,p_{\mathcal{R}(a),t})|\right.\\
&\left.<\frac{3}{2}Ls(p_{\mathcal{R}(a),t})\right\},\\
&\subset\left\{|\bar{r}_t^{(\mathcal{R}(a),j)}((p_{\mathcal{R}(a),t},p_{j,t}),a)-\pi(a,p_{\mathcal{R}(a),t})|<\epsilon\right\},
\end{aligned}
$$

for $t \in \tau(T)$. This is satisfied when $l_{\min} \geq \log_2(3L/(2\epsilon))$. Starting with level $l_{\min}$ intervals instead of level 0 intervals decreases the exploitation regret of ORL-CF. Hence the regret bound in Theorem 1 is an upper bound on the exploitation regret.

For any sequence of context arrivals, we have the following bound on the level of the interval with the maximum level,

$$l_{\max} < 1 + l_{\min} + \log_2 T/\rho.$$

Continuing similarly with the proof of Theorem 2, we have

$$
\begin{aligned}
R_{\text{O}}(T) &\leq (c_O+1)D^2\left(2\sum_{l=0}^{1+\log_2 T/\rho} 2^{l_{\min}}2^l \sum_{k=l}^{1+\log_2 T/\rho} 2^{l_{\min}}2^k\right.\\
&\left.\left(2^{4l_{\min}}\frac{2\log(tAD/\delta)}{2^{-2l_{\min}}2^{-2k}L^2}+1\right)\right)\\
&= (c_O+1)D^2\left(\frac{4\log(tAD/\delta)}{L^2}\sum_{l=0}^{1+\log_2 T/\rho} 2^l \sum_{k=l}^{1+\log_2 T/\rho} 2^{3k}\right.\\
&\left.+2^{2l_{\min}}2\sum_{l=0}^{1+\log_2 T/\rho} 2^l \sum_{k=l}^{1+\log_2 T/\rho} 2^k\right)\\
&\leq 2^{4l_{\min}}\left(\frac{4D^2(c_O+1)\log(tAD/\delta)}{L^2}\times\frac{240}{7}T^{4/\rho}\right.\\
&\left.+\frac{64D^2(c_O+1)}{3}T^{2/\rho}\right).
\end{aligned}
$$

Cem Tekin Cem Tekin is an Assistant Professor in Electrical and Electronics Engineering Department at Bilkent University, Turkey. From February 2013 to January 2015, he was a Postdoctoral Scholar at University of California, Los Angeles. He received the B.Sc. degree in electrical and electronics engineering from the Middle East Technical University, Ankara, Turkey, in 2008, the M.S.E. degree in electrical engineering: systems, M.S. degree in mathematics, Ph.D. degree in electrical engineering: systems from the University of Michigan, Ann Arbor, in 2010, 2011 and 2013, respectively. His research interests include machine learning, multi-armed bandit problems, data mining, multi-agent systems and game theory. He received the University of Michigan Electrical Engineering Departmental Fellowship in 2008, and the Fred W. Ellersick award for the best paper in MILCOM 2009.

Mihaela van der Schaar Mihaela van der Schaar is Chancellor Professor of Electrical Engineering at University of California, Los Angeles. Her research interests include network economics and game theory, online learning, dynamic multi-user networking and communication, multimedia processing and systems, real-time stream mining. She is an IEEE Fellow, a Distinguished Lecturer of the Communications Society for 2011-2012, the Editor in Chief of IEEE Transactions on Multimedia and a member of the Editorial Board of the IEEE Journal on Selected Topics in Signal Processing. She received an NSF CAREER Award (2004), the Best Paper Award from IEEE Transactions on Circuits and Systems for Video Technology (2005), the Okawa Foundation Award (2006), the IBM Faculty Award (2005, 2007, 2008), the Most Cited Paper Award from EURASIP: Image Communications Journal (2006), the Gamenets Conference Best Paper Award (2011) and the 2011 IEEE Circuits and Systems Society Darlington Award Best Paper Award. She received three ISO awards for her contributions to the MPEG video compression and streaming international standardization activities, and holds 33 granted US patents.