Kernel Methods for Cooperative Multi-Agent Contextual Bandits

Abhimanyu Dubey [1] **Alex Pentland** [1]

Abstract

Cooperative multi-agent decision making involves a group of agents cooperatively solving learning problems while communicating over a network with delays. In this paper, we consider the kernelised contextual bandit problem, where the reward obtained by an agent is an arbitrary linear function of the contexts' images in the related reproducing kernel Hilbert space (RKHS), and a group of agents must cooperate to collectively solve their unique decision problems. For this problem, we propose COOP-KERNELUCB, an algorithm that provides near-optimal bounds on the per-agent regret, and is both computationally and communicatively efficient. For special cases of the cooperative problem, we also provide variants of COOP-KERNELUCB that provides optimal per-agent regret. In addition, our algorithm generalizes several existing results in the multi-agent bandit setting. Finally, on a series of both synthetic and real-world multi-agent network benchmarks, we demonstrate that our algorithm significantly outperforms existing benchmarks.

1. Introduction

An emerging problem in online learning and multi-agent distributed systems is the *cooperative* multi-agent bandit. It involves a group $\mathcal{V}$ of V agents collectively solving a decision problem while communicating with each other. The problem proceeds in rounds $t = 1, 2, ..., T$, where at any trial $t = 1, 2, ...$, each agent $v \in \mathcal{V}$ is presented with a *decision set* $\mathcal{D}_{v,t}$, and selects an action $\boldsymbol{x}_{v,t} \in \mathcal{D}_{v,t}$. Each agent obtains a stochastic reward $y_{v,t}$, following:

$$y_{v,t} = f(\boldsymbol{x}_{v,t}) + \varepsilon_{v,t},$$

where $\varepsilon_{v,t}$ is i.i.d. noise, and f is an unknown (but fixed) function. The collective objective of the group of agents is to select actions that minimize the expected *group regret*:

$$\mathcal{R}_{\mathcal{G}}(T) = \sum_{v \in \mathcal{V}} \sum_{t=1}^{T} \left(f(\boldsymbol{x}^*_{v,t}) - f(\boldsymbol{x}_{v,t}) \right),$$

where, $\boldsymbol{x}^*_t = \arg\max_{\boldsymbol{x} \in D_{v,t}} f(\boldsymbol{x})$. The research objective for this problem is to design multi-agent algorithms that can leverage communication to improve overall performance (Landgren et al., 2016a; Szorenyi et al., 2013).

Agents communicate via an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $(i, j) \in \mathcal{E}$ if agents i and j are connected. Messages from any agent v are available to agent v' after $d(v, v') - 1$ trials of the bandit, where d is the distance between the agents in $\mathcal{G}$. This gradually creates *heterogeneity* between the information available to each agent, and is the primary technical challenge in algorithm design for this problem. Moreover, recent work assumes that the bandit problem is common (i.e., f is identical for all agents), but this assumption does not hold for most decentralized applications (Boldrini et al., 2018). For instance, in a decentralized supply chain network (Thadakamaila et al., 2004), agents interact with similar but non-identical decision problems, since loads are generally distributed non-uniformly. In this setting, naïvely incorporating observations from neighboring agents may not be beneficial, and algorithms must be carefully designed to optimally leverage cooperation.

A related problem is the online *social network clustering* of bandits, where, at every trial, a randomly selected agent interacts with the bandit (Cesa-Bianchi et al., 2013; Gentile et al., 2014; 2017; Li et al., 2016; 2019). In this formulation, a fixed (but unknown) clustering over the agents is assumed, where agents within a cluster have identical context functions. While the assumptions of linearity and clustering are feasible in the context of social networks (Al Mamunur Rashid et al., 2006), these assumptions may not hold for general multi-agent environments, such as geographically-distributed computational clusters (Cano et al., 2016). In the case when each agent has its own unique decision problem, the clustering approach leads to an $\mathcal{O}(V)$ multiplicative increase in the group regret. Moreover, the *social network clustering* problem is *single-agent*, since at any trial, only one agent interacts with the bandit. This makes it less challenging compared to the multi-agent cooperative bandit, since there is no heterogeneity of information (as

[1]Media Lab and Institute for Data, Systems and Society, Massachusetts Institute of Technology. Correspondence to: Abhimanyu Dubey <dubeya@mit.edu>.

Proceedings of the 37th International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

discussed earlier). Multi-agent settings have been considered for *social network clustering* (Korda et al., 2016), but without delayed feedback (hence, without heterogeneity).

Contributions. In this paper, we study the cooperative multi-agent bandit with delays. We assume that each agent $v \in \mathcal{V}$ interacts with a separate bandit function f_v, where all functions $f_v,\ v \in \mathcal{V}$ have small norm in a known reproducing kernel Hilbert space (RKHS) (Schölkopf & Smola, 2005) specified by a fixed kernel K_x. This is a more general setting compared to the existing *clustering* or identical (i.e., *fully-cooperative*) settings in the literature, and allows us to propose a technique to measure the similarity between the functions f_v via an agent-based similarity kernel, which can be learnt online when it is unknown. Under this formulation, we present COOP-KERNELUCB, an algorithm for the multi-agent contextual bandit problem on networks.

For the context-free multi-agent cooperative bandit problem, existing bounds on group regret scale as $\mathcal{O}\left(\sqrt{TV\log(V\cdot\lambda_{\max}(\mathcal{G}))}\right)$, where, $\lambda_{\max}(\mathcal{G})$ is the maximum eigenvalue of the graph Laplacian for $\mathcal{G}$ (Martínez-Rubio et al., 2019; Landgren et al., 2016b;a). These bounds are obtained by using a communication protocol known as the *running consensus*, that involves agents averaging their beliefs with neighboring agents. This communication protocol restricts these methods to the *fully-cooperative* setting, i.e., all agents have identical arms; it is trivial to see that naive averaging of estimates in non-identical settings can lead to irreducible bias and $\Theta(T)$ regret. COOP-KERNELUCB employs an alternate communication protocol, entitled LOCAL (Suomela, 2013), that involves agents sending messages to each other. Additionally, COOP-KERNELUCB uses an alternative "network" kernel K_z, to measure similarity between agent reward functions $f_1, ..., f_V$. When K_z is known (such as, e.g., in cases when agents correspond to users in a social network), we can use K_z to construct a product kernel $K = K_z \odot K_x$, and use K (instead of K_x) to construct upper confidence bounds.

We consider the case when the decision sets $\mathcal{D}_{v,t}$ can be infinite or continuum action spaces, and $\forall\ v \in \mathcal{V}$, $\|f_v\|_{\mathcal{H}} \leq B$. In this setting, the single-agent IGP-UCB (Chowdhury & Gopalan, 2017) algorithm, assuming the agent interacts with a single function f, obtains a regret of $\widetilde{\mathcal{O}}(\sqrt{VT}(B\sqrt{\Upsilon^x_{VT}}+\Upsilon^x_{VT}))$ (where the $\widetilde{\mathcal{O}}$ hides additional logarithmic dependence on $1/\delta$) when run for a total of VT rounds (i.e., a centralized agent that pulls all arms), where Υ^x_{VT} is the *information gain* after VT rounds, a quantity dependent on the structure of the RKHS of $\mathcal{X}$. We demonstrate that COOP-KERNELUCB, that uses the underlying martingale inequality from IGP-UCB, obtains a regret of $\widetilde{\mathcal{O}}\left(\sqrt{VT\cdot\bar{\chi}(\mathcal{G}_\gamma)}\left(B\sqrt{\Upsilon^x_{VT}\Upsilon^z}+\Upsilon^x_{VT}\Upsilon^z\right)\right)$. Here, Υ^z is a term corresponding to the similarity between functions f_v via the kernel K_z, and $\bar{\chi}(\mathcal{G}_\gamma)$ is a term accounting for the delayed propagation of information in the network $\mathcal{G}$[1]. This bound is achieved by utilizing graph partitions to control the deviation in the confidence bound for each agent.

Our bound is reminiscent of single-agent bounds with additional contexts (Deshmukh et al., 2017; Krause & Ong, 2011), which rely on a *known* K_z. However, in many cases, K_z is (unknown) and requires estimation. For this case, we provide an alternative algorithm via kernel mean embeddings (Christmann & Steinwart, 2010). Against state-of-the-art methods on a variety of real-world and synthetic multi-agent networks, our algorithm exhibits superior performance. Moreover, we present a variant, EAGER-KERNELUCB, of our algorithm (without regret bounds) that comfortably outperforms COOP-KERNELUCB and other benchmarks. This extends the current literature of cooperative bandit estimation from the stochastic multi-armed problem (Landgren et al., 2018; Martínez-Rubio et al., 2019; Landgren et al., 2016a) to a more general class of functions, and provides a technique to determine task similarity over arbitrary cooperative settings.

2. Preliminaries

Notation. We use boldface uppercase to represent matrices, i.e., $\boldsymbol{A}$, and lowercase for vectors, i.e., $\boldsymbol{x}$. The RKHS norm of a function f in RKHS $\mathcal{H}$ with kernel K is given by $\|f\|_{\mathcal{H}} = \sqrt{\langle f, f\rangle_{\mathcal{H}}}$. We denote the set $\{a, a+1..., b-1, b\}$ by the shorthand $[a, b]$ and simply as $[b]$ when $a = 1$. We refer to the γ^{th} graph power of a graph $\mathcal{G}$ as $\mathcal{G}_\gamma$ (i.e., $\mathcal{G}_\gamma$ contains an edge (i, j) if there exists a shortest path of length at most γ between i and j in $\mathcal{G}$). We denote a clique in $\mathcal{G}_\gamma$ as a γ-clique in $\mathcal{G}$, and denote $\mathcal{G}$ as γ-complete if $\mathcal{G}_\gamma$ is complete. $N_\gamma(v) \subseteq \mathcal{V}$ denotes the set of nodes at a shortest distance of at most γ from node v in $\mathcal{G}$, referred to as the "γ-neighborhood" of v in $\mathcal{G}$. The distance between two nodes v and v' in $\mathcal{G}$ is given by the shorthand $d_{\mathcal{G}}(v, v')$.

Problem Setup. We consider a multi-agent setting of V agents sitting on the vertices of a network represented by an undirected and connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$[2]. We assume that agents each solve unique instances of kernelised contextual bandit problems. At each step $t = 1, 2, ...$ each agent $v \in V$ obtains, at time t, a decision set $\mathcal{D}_{v,t} \subseteq \mathcal{X} \subset \mathbb{R}^d$, where $\mathcal{X}$ is a compact subset of $\mathbb{R}^d$. In this paper, we assume $\mathcal{D}_{v,t}$ to even be an infinite set or a continuum of actions, however, we discuss the case when it is a finite set of contexts $\boldsymbol{x}^{(1)}_{v,t}, \boldsymbol{x}^{(2)}_{v,t}, ...$ later on, for which tighter regret bounds can be obtained. At each trial t, each agent selects an action

[1] $\bar{\chi}(\mathcal{G}_\gamma)$ denotes the minimum clique number of the γ^{th} power of graph $\mathcal{G}$, i.e., $\mathcal{G}_\gamma$ has an edge (i, j) if there is a path of length at most γ between i and j in $\mathcal{G}$.

[2] Our results and algorithm can be trivially extended to directed or disconnected case, by considering each connected subgraph individually, and considering statistics of the directed graph instead.

$\boldsymbol{x}_{v,t} \in \mathcal{D}_{v,t}$, and receives a reward $y_{v,t} = f_v(\boldsymbol{x}_{v,t}) + \varepsilon_{v,t}$. Where $f_v : \widetilde{\mathcal{X}} \to \mathbb{R}$ is a fixed (but unknown) function, and $\varepsilon_{v,t}$ is additive noise such that the noise sequence $\{\varepsilon_{v,t}\}_{t=1,v\in V}^{\infty}$ is conditionally R-sub-Gaussian.

Single-Agent UCB. Our approach builds on the existing research for upper confidence bounds for bandit kernel learning, a line of research that has seen a lot of interest (Srinivas et al., 2009; Krause & Ong, 2011; Chowdhury & Gopalan, 2017; Valko et al., 2013; Deshmukh et al., 2017). The central idea across all these approaches is to construct an upper confidence bound (UCB) envelope for the true function $f(\cdot)$ using an estimate $\hat{f}_t$, and then chooses an action $\boldsymbol{x}_t \in \mathcal{D}_t$ that maximizes this upper confidence bound, i.e., for some estimate $\hat{f}_t$ of f,

$$\boldsymbol{x}_t = \underset{\boldsymbol{x}\in\mathcal{D}_t}{\arg\max}\left[\hat{f}_t(\boldsymbol{x}) + \sqrt{\beta_t}\sigma_{t-1}(\boldsymbol{x})\right]. \tag{1}$$

Here, β_t is an appropriately chosen "exploration" parameter, and σ_{t-1} can be thought of as the "variance" in the estimate $\hat{f}_t$. Existing UCB-based approaches aim to construct a sequence $(\beta_t)_t$ to ensure a near-optimal tradeoff between exploration and exploitation. The natural choice for $\hat{f}_t$ is the solution to the kernelised ridge regression. Given $\lambda \geq 0$ and $\boldsymbol{X}_{<t} = (\boldsymbol{x}_i, y_i)_{i=1}^t$,

$$\hat{f}_t = \underset{f\in\mathcal{H}}{\arg\min}\ \frac{1}{t}\sum_{(\boldsymbol{x},y)\in\boldsymbol{X}_{<t}}(f(\boldsymbol{x})-y))^2 + \lambda\|f\|_{\mathcal{H}}^2. \tag{2}$$

The solution to the above problem (2) can be written as the following (Valko et al., 2013) (for $\boldsymbol{\kappa}_t(\boldsymbol{x}) = (K(\boldsymbol{x},\boldsymbol{x}_i))_{i=1}^t, \boldsymbol{y}_t = (y_i)_{i=1}^t$ and $\boldsymbol{K}_t = (K(\boldsymbol{x}_i,\boldsymbol{x}_j))_{i,j\in[t]}$):

$$\hat{f}_t(\boldsymbol{x}) = \boldsymbol{\kappa}_t(\boldsymbol{x})^\top(\boldsymbol{K}_t + \lambda\boldsymbol{I})^{-1}\boldsymbol{y}_t. \tag{3}$$

For any particular choice of the sequence $(\beta_t)_t$, various algorithms can be obtained, with different regret guarantees. To provide more insight into the regret bounds obtained by various algorithms, we now provide the definition of Υ^x, i.e., *maximum information gain*.

Definition 1 (Maximum Information Gain (Srinivas et al., 2009; Krause & Ong, 2011)). *For $y_t = f(\boldsymbol{x}_t) + \varepsilon_t$, let $A \subset \mathcal{X}$ be a finite subset such that $|A| = T$. Let $\boldsymbol{y}_A = \boldsymbol{f}_A + \boldsymbol{\varepsilon}_A$ where $\boldsymbol{f}_A = (f(\boldsymbol{x}_i))_{\boldsymbol{x}_i\in A}$ and $\boldsymbol{\varepsilon}_A \sim \mathcal{N}(0,R^2)$. The maximum information gain Υ_T^x after T rounds is:*

$$\Upsilon_T^x \triangleq \max_{A\subset\mathcal{X}:|A|=T} H(\boldsymbol{y}_A) - H(\boldsymbol{y}_A|f). \tag{4}$$

Here $H(\cdot)$ refers to the entropy of a random variable. Furthermore, if we assume that the kernel K_x is bounded; i.e., $K_x(\boldsymbol{x},\boldsymbol{x}) \leq 1\ \forall\boldsymbol{x}\in\mathcal{X}$, the following is true for compact and convex $\mathcal{X}\subset\mathbb{R}^d$. For linear K_x, $\Upsilon_T^x = \mathcal{O}(d\log T)$. For RBF K_x, $\Upsilon_T^x = \mathcal{O}((\log T)^{d+1})$. For Matérn K_x with $\nu > 1$, $\Upsilon_T^x = \mathcal{O}(T^{\frac{d(d+1)}{2\nu+d(d+1)}}(\log T))$.

Remark 1 (UCB Regret for Single-Agent Algorithms). *Let $\delta \in (0,1]$. For continuum-armed $\mathcal{D}_t$, choosing $\beta_t = 2B + 300\Upsilon_{t-1}^x log^3(t/\delta)$ guarantees with probability at least $1-\delta$ a regret of $\widetilde{\mathcal{O}}(\sqrt{T}(B\sqrt{\Upsilon_T^x} + \Upsilon_T^x\ln^{3/2}(T)))$, as demonstrated in the work of (Srinivas et al., 2009) (GP-UCB). This was improved via a new martingale inequality to $\widetilde{\mathcal{O}}(\sqrt{T}(B\sqrt{\Upsilon_T^x} + \Upsilon_T^x))$ in the work of Chowdhury & Gopalan (2017) with the choice of $\beta_t = B + R\sqrt{2(\Upsilon_{t-1}^x + 1 + \ln(1/\delta))}$. An alternative regret bound of $\widetilde{\mathcal{O}}(\sqrt{\tilde{d}T})$ was provided via the Sup-KernelUCB algorithm of Valko et al. (2013) (for finite-armed $\mathcal{D}_t$), where $\tilde{d}$ is the **effective dimension** of K_x, a measure of the intrinsic dimensionality of the RKHS $\mathcal{H}$. $\tilde{d}$ is related to Υ^x as $\Upsilon^x \geq \Omega(\tilde{d}\ln\ln T)$. Further work has focused on improving bounds for various families of kernels, e.g., see (Janz et al., 2020; Scarlett et al., 2017).*

The primary goal in the cooperative learning setting is to provide each agent with stronger estimators that leverage observations from neighboring agents. A suitable baseline, therefore, in this setting, would be that of a centralized agent pulling VT arms in a round-robin manner. Existing single-agent algorithms propose a regret bound of $\widetilde{\mathcal{O}}(\Upsilon_{VT}^x\sqrt{VT})$ in this setting, and this is the comparative regret bound we wish to match. We do not focus on stronger controls for specific kernels or of the information gain Υ^x, and for that we refer the reader to references in Remark 1.

3. Cooperative Kernelized Bandits

Network Contexts. Recall that for any agent v, the rewards y_v are generated following $y_{v,t} = f_v(\boldsymbol{x}_{v,t}) + \varepsilon_{v,t}$. To provide a relationship between different f_v, we assume that the functions $f_v, v\in\mathcal{V}$ are parameteric functionals of some function $F : \mathcal{X}\times\mathcal{Z}\to\mathbb{R}$ for a known *network context* space $\mathcal{Z}$ such that $\forall\ v\in\mathcal{V}, \exists\ \boldsymbol{z}_v\in\mathcal{Z}$ such that $\forall\boldsymbol{x}\in\mathcal{X}$,

$$f_v(\boldsymbol{x}) = F(\boldsymbol{x},\boldsymbol{z}_v). \tag{5}$$

Kernel Assumptions. We denote the space $\mathcal{X}\times\mathcal{Z}$ as $\widetilde{\mathcal{X}}$, and the overall input $(\boldsymbol{x},\boldsymbol{z})$ as $\tilde{\boldsymbol{x}}$. Furthermore, we assume that the function F has a small norm in the reproducing kernel Hilbert space (RKHS, Schölkopf & Smola (2005)) $\mathcal{H}_K$ associated with a PSD kernel $K : \widetilde{\mathcal{X}}\times\widetilde{\mathcal{X}}\to\mathbb{R}$. $\mathcal{H}_K$ is completely specified by the kernel $K(\cdot,\cdot)$, and via an inner product $\langle\cdot,\cdot\rangle_K$ following the reproducing property. As is typical with the kernelized bandit literature, we assume a known bound on the RKHS norm of F, i.e. $\|F\|_K \leq B$, and we assume that the kernel has finite variance, i.e. $K(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{x}}) \leq 1, \forall\ \tilde{\boldsymbol{x}}\in\widetilde{\mathcal{X}}$[3].

[3]These are typically made assumptions in the contextual bandit literature, and avoid scaling of the regret bounds. In the linear case, the first assumption corresponds to having a bound on the norm of the context vectors (Chowdhury & Gopalan, 2017), and the second

Finally, we must impose constraints on the interaction of the inputs $\boldsymbol{x}$ and $\boldsymbol{z}$ via two kernels $K_x(\cdot,\cdot)$ and $K_z(\cdot,\cdot)$. We assume that K is a composition of two separate positive-semidefinite kernels, K_z and K_x such that $K_z : \mathcal{Z}\times\mathcal{Z} \to \mathbb{R}$, i.e., operating on the network contexts, and $K_x : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ operates on the action contexts. Our regret bounds assume that the overall kernel K is formed via the Hadamard product of K_x and K_z:

$$K\left((\boldsymbol{z},\boldsymbol{x}),(\boldsymbol{z}',\boldsymbol{x}')\right) = K_x(\boldsymbol{x},\boldsymbol{x}')K_z(\boldsymbol{z},\boldsymbol{z}'). \tag{6}$$

Remark 2 (Kernel Compositions)**.** *For the development in the paper, we restrict ourselves to the Hadamard composition, however, it is important to note that this is not a limitation of our technique, and other compositions can be explored. See the Appendix for details on the sum ($K_z \oplus K_x$), and Kronecker ($K_z \otimes K_x$) compositions.*

Remark 3 (Independent vs. Pooled Modeling)**.** *When $\mathcal{D}_{v,t}$ are countably finite, an alternate formulation is the "independent" assumption (Li et al., 2010), where a separate model is considered for each "arm". We assume the "pooled" environment (Abbasi-Yadkori et al., 2011), (i.e., where all "arms" are modeled together), however it is easy to extend results to the former setting, by assuming arm-dependent network contexts (see appendix).*

The *network* kernel, K_z, determines how "similar" agent functions f_v are. For example, if all agents solve the same bandit problem, i.e., $f_v = f \; \forall v \in V$, then the appropriate choice for this is to set $\mathcal{Z} = \{1\}$, and $\boldsymbol{z}_v = 1$ for all $v \in V$, and hence, $K = K_x$. Alternatively, in many internet applications, users (which may correspond to agents) are arranged in an online social network (say, $\mathcal{G}_{\text{net}}$), and $\boldsymbol{z}_v$ can be a network embedding of user v in $\mathcal{G}_{\text{net}}$. Typically, however, $\mathcal{Z}$ can be defined more generally with a corresponding positive semi-definite (PSD) kernel K_z.

3.1. LOCAL Communication

Existing research on distributed bandit learning has largely focused on two communication protocols - the first being a centralized setting (Liu & Zhao, 2010; Wang et al., 2020), as is standard in distributed computation, where a central server acts as an intermediary between "client" agents (i.e., a star-graph communication), and the second being the *running consensus* protocol, where agents are arranged in a network structure, but communication is done by repeatedly averaging (a weighted version) of observations with neighbors (Landgren et al., 2016a;b).

In this paper, we use the LOCAL communication protocol (Suomela, 2013; Fraigniaud, 2016; Linial, 1992), which has recently seen an increase in interest in the decentralized multi-agent bandit literature (Cesa-Bianchi et al., 2019a;b).

is to ensure the methods are scale-free (Agrawal & Goyal, 2012).

At an abstract level, it can be seen as a generalization of the centralized communication protocol to a server-free, arbitrary graph setting. The protocol assumes that pulling a bandit arm and communication occur sequentially within each trial t, i.e., first, each agent $v \in \mathcal{V}$ pulls an arm $\tilde{\boldsymbol{x}}_{v,t}$ and receives a reward $y_{v,t}$ from the respective bandit environment. The agent then sends the message $\boldsymbol{m}_{v,t} = \langle t, v, \tilde{\boldsymbol{x}}_{v,t}, y_{v,t}\rangle$ to its neighbors in $\mathcal{G}$. This message is forwarded from agent to agent γ times (taking one trial of the bandit problem each between forwards), after which it is dropped. The *time-to-live* (delay) parameter γ is a common technique to control communication complexity in this setting. Each agent $v \in V$ therefore also receives messages $\boldsymbol{m}_{v',t-d(v,v')}$ from all the nodes v' such that $d(v,v') \leq \gamma$.

3.2. Cooperative Kernel-UCB

In this section we present the primary algorithm, cooperative Kernel-UCB. The central ideas in the development of the algorithm are (a) to leverage the similarity of the agent kernels (as specified by K_z) and (b) to control the variance estimates $\sigma^2_{v,t-1}$ between agents by delayed diffusion of rewards.

Using an Augmented Kernel. For each agent $v \in \mathcal{V}$ we construct an upper confidence bound (UCB) envelope for the true function $f_v(\cdot) = F(\cdot, \boldsymbol{z}_v)$ over the space $\widetilde{\mathcal{X}}$. This is done by using the composition kernel K instead of the action kernel K_x, which allows us to take the network context $\boldsymbol{z}_v$ into account. The agent then chooses an action that maximizes the upper confidence bound, following the typical approach in UCB-based algorithms. For any $v \in V, \boldsymbol{x} \in \mathcal{D}_{v,t}$, the UCB can be given by,

$$\boldsymbol{x}_{v,t} = \underset{\boldsymbol{x}\in\mathcal{D}_{v,t}}{\arg\max}\left[\widehat{F}_{v,t}\left(\boldsymbol{z}_v,\boldsymbol{x}\right) + \sqrt{\beta_{v,t}}\sigma_{v,t-1}\left(\boldsymbol{z}_v,\boldsymbol{x}\right)\right]. \tag{7}$$

Here $\widehat{F}_{v,t}(\cdot,\boldsymbol{z}_v)$ is the agent's estimate for f_v at time t, and the second term denotes the exploration bonus. Using $\boldsymbol{x}_{v,t}$, the agent can construct the *aggregate* optimal context $\tilde{\boldsymbol{x}}_{v,t} = (\boldsymbol{z}_v, \boldsymbol{x}_{v,t})$. $\widehat{F}_{v,t}$ is obtained by solving:

$$\widehat{F}_{v,t} = \arg\min_{f\in\mathcal{H}_K} \tfrac{1}{n_v(t)}\left(\textstyle\sum_{(\tilde{\boldsymbol{x}},y)\in\widetilde{\boldsymbol{X}}_{v,t}}\left(f(\tilde{\boldsymbol{x}})-y\right)^2\right) + \lambda\|f\|^2_{\mathcal{H}_K}. \tag{8}$$

Here, $\widetilde{\boldsymbol{X}}_{v,t} = (\tilde{\boldsymbol{x}}_i, y_i)_{i=1}^{n_v(t)}$ denotes the $n_v(t)$ total action-reward pairs available at time t. Note that this comprises not just personal observations, but additional observations available via the messages received until that time. The solution to the above problem (8) is given as:

$$\widehat{F}_{v,t}(\tilde{\boldsymbol{x}}) = \boldsymbol{\kappa}_{v,t}(\tilde{\boldsymbol{x}})^\top\left(\boldsymbol{K}_{v,t} + \lambda\boldsymbol{I}\right)^{-1}\boldsymbol{y}_{v,t}. \tag{9}$$

Here, $\boldsymbol{\kappa}_{v,t}(\tilde{\boldsymbol{x}}) = \left(K\left(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{x}}_i\right)\right)_{i=1}^{n_v(t)}$ denotes the vector of kernel values between the input vector $\boldsymbol{x}$ and all previously stored data by agent v, and similarly $\boldsymbol{y}_{v,t} = (y_{v,i})_{i=1}^{n_v(t)}$ denotes the vector of rewards. The matrix $\boldsymbol{K}_{v,t}$ denotes the

$n_v(t) \times n_v(t)$ matrix of kernel evaluations of every pair of samples $\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j \in \widetilde{\boldsymbol{X}}_{v,t}$ possessed by agent v. To construct the sequence $(\beta_{v,t})_t$, following result motivates the upper confidence bound.

Lemma 1 (Chowdhury & Gopalan (2017))**.** *Let $\widetilde{\mathcal{X}} \subset \mathbb{R}^d$, and $F : \widetilde{\mathcal{X}} \to \mathbb{R}$ be a member of the RKHS of real-valued functions on $\widetilde{\mathcal{X}}$ with kernel K, and RKHS norm bounded by B. Then, with probability at least $1-\delta$, the following holds for all $\tilde{\boldsymbol{x}} \in \widetilde{\mathcal{X}}$, and simultaneously for all $t \geq 1, v \in \mathcal{V}$:*

$$\Delta_{v,t}(\tilde{\boldsymbol{x}}) \leq \sigma^2_{v,t-1}(\tilde{\boldsymbol{x}}) \left(B + R\sqrt{\ln \tfrac{\det(\lambda \boldsymbol{I} + \boldsymbol{K}_{v,t})}{\delta^2} + 2\ln(V)} \right).$$

Where $\Delta_{v,t}(\tilde{\boldsymbol{x}}) = \left| F(\tilde{\boldsymbol{x}}) - \widehat{F}_{v,t}(\tilde{\boldsymbol{x}}) \right|$ and we denote $\sigma^2_{v,t-1}(\tilde{\boldsymbol{x}}) = K(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}) - \boldsymbol{\kappa}_{v,t}(\tilde{\boldsymbol{x}})^\top (\boldsymbol{K}_{v,t} + \lambda \boldsymbol{I})^{-1} \boldsymbol{\kappa}_{v,t}(\tilde{\boldsymbol{x}})$ as the "variance" proxy for the UCB for brevity. This confidence bound is derived using the stronger, kernelized self-normalized concentration inequality from Chowdhury & Gopalan (2017), that holds *simultaneously* for all $t \geq 1$, and hence prevents the second logarithmic term, in contrast to (Srinivas et al., 2009) for continuum-armed $\mathcal{D}_{v,t}$.

Controlling Drift via Clique Partitions. The fundamental idea in controlling regret is to bound the per-round regret incurred by any agent by the UCB "variance" term $\sigma^2_{v,t-1}(\tilde{\boldsymbol{x}}_{v,t})$, and the algorithm attempts to bound $\sum_{v \in V} \sigma^2_{v,t-1}(\tilde{\boldsymbol{x}}_{v,t})$ by a quantity smaller than $O(\sqrt{V})$ (i.e., improve over non-cooperative behavior). Our approach is to obtain this rate by partitioning $\mathcal{G}$ into G subgraphs $\mathcal{G}'_1, ..., \mathcal{G}'_G$, and ensuring that the variance terms are similar for each agent within a subgraph for all t.

Our partitioning solution is a conservative one: let $\mathbf{C}$ be a clique covering of the γ-power of $\mathcal{G}$. For any clique $\mathcal{C} \in \mathbf{C}$, we restrict each agent $v \in \mathcal{C}$ to only accept observations from agents that belong to $\mathcal{C}$ as well. This ensures that at any t, any agent $v \in \mathcal{C}$ has an upper bound on $\sigma^2_{v,t-1}$ that depends only on $\mathcal{C}$. Therefore we can control the group regret within each clique $\mathcal{C}$, leading to a factor of $\sqrt{\bar{\chi}(\mathcal{G}_\gamma)}$ (instead of V) in the regret, where $\bar{\chi}(\cdot)$ is the clique number.

Remark 4 (Computational complexity)**.** *As outlined in Valko et al. (2013), it is possible to perform an $\mathcal{O}(1)$ update of the Gram matrix, via the Schur decomposition (l. 30-34, Algorithm 1). This update can also be applied when K_z is unknown and approximated online, see Section 3.3.*

Algorithm 1 COOP-KERNELUCB

```
 1: Input: Graph $\mathcal{G}_\gamma$ with clique cover $\boldsymbol{C}_\gamma$, kernels
    $K_x(\cdot,\cdot), K_z(\cdot,\cdot)$, $\lambda$, explore param. $\eta$, buffers $\boldsymbol{B}_v = \phi$.
 2: for For each iteration $t \in [T]$ do
 3:   for For each agent $v \in V$ do
 4:     if $t = 1$ then
 5:       $\boldsymbol{x}_{v,t} \leftarrow$RANDOM$(D_{v,t})$.
 6:     else
 7:       $\boldsymbol{x}_{v,t} \leftarrow \arg\max_{\boldsymbol{x} \in D_{v,t}} \left( \hat{f}_{v,t}(\boldsymbol{z}_v, \boldsymbol{x}) + \frac{\eta}{\sqrt{\lambda}} \sigma_{v,t-1}(\boldsymbol{z}_v, \boldsymbol{x}) \right)$.
 8:     end if
 9:     $\tilde{\boldsymbol{x}}_{v,t} \leftarrow (\boldsymbol{z}_v, \boldsymbol{x}_{v,t}), y_{v,t} \leftarrow$PULL$(\tilde{\boldsymbol{x}}_{v,t})$.
10:     if $t = 1$ then
11:       $(\boldsymbol{K}_{v,t})^{-1} \leftarrow 1/K(\tilde{\boldsymbol{x}}_{v,t}, \tilde{\boldsymbol{x}}_{v,t}) + \lambda$.
12:       $\boldsymbol{y}_v \leftarrow [y_{v,0}]$.
13:       $\boldsymbol{\kappa}_v = (K(\cdot, \tilde{\boldsymbol{x}}_{v,t}))$.
14:     else
15:       $\boldsymbol{B}_v \leftarrow \boldsymbol{B}_v \cup (\tilde{\boldsymbol{x}}_{v,t}, y_{v,t})$.
16:     end if
17:     $\boldsymbol{m}_{v,t} \leftarrow \langle t, v, \tilde{\boldsymbol{x}}_{v,t}, y_{v,t} \rangle$.
18:     SENDMESSAGE$(\boldsymbol{m}_{v,t})$.
19:     for $\langle t', v', \tilde{\boldsymbol{x}}', y' \rangle$ in RECVMESSAGES$(v, t)$ do
20:       if $v' \in$ CLIQUE$(v, \mathbf{C}_\gamma)$ then
21:         $\boldsymbol{B}_v \leftarrow \boldsymbol{B}_v \cup (\tilde{\boldsymbol{x}}', y')$.
22:       end if
23:     end for
24:     for $(\tilde{\boldsymbol{x}}', y') \in \boldsymbol{B}_v$ do
25:       $\boldsymbol{y}_v \leftarrow [\boldsymbol{y}_v, y']$.
26:       $\boldsymbol{\kappa}_v = (\boldsymbol{\kappa}_v, K(\cdot, \tilde{\boldsymbol{x}}'))$.
27:       $\boldsymbol{K}_{22} \leftarrow \left( K(\tilde{\boldsymbol{x}}', \tilde{\boldsymbol{x}}') + \lambda - (\boldsymbol{\kappa}_v)^\top (\boldsymbol{K}_{v,t})^{-1} \boldsymbol{\kappa}_v \right)^{-1}$.
28:       $\boldsymbol{K}_{11} \leftarrow \left( (\boldsymbol{K}_{v,t})^{-1} + \boldsymbol{K}_{22} (\boldsymbol{K}_{v,t})^{-1} \boldsymbol{\kappa}_v (\boldsymbol{\kappa}_v)^\top (\boldsymbol{K}_{v,t})^{-1} \right)$.
29:       $\boldsymbol{K}_{12} \leftarrow -\boldsymbol{K}_{22} (\boldsymbol{K}_{v,t})^{-1} \boldsymbol{\kappa}_v^\tau$.
30:       $\boldsymbol{K}_{21} \leftarrow -\boldsymbol{K}_{22} (\boldsymbol{\kappa}_v)^\top (\boldsymbol{K}_{v,t})^{-1}$.
31:       $(\boldsymbol{K}_{v,t})^{-1} \leftarrow [\boldsymbol{K}_{11}, \boldsymbol{K}_{12}; \boldsymbol{K}_{21}, \boldsymbol{K}_{22}]$.
32:     end for
33:     $\boldsymbol{B}_v = \phi$.
34:     $\hat{f}_{v,t+1} \leftarrow (\boldsymbol{\kappa}_v)^\top (\boldsymbol{K}_{v,t})^{-1} \boldsymbol{y}_v$.
35:     $s_{v,\rho+1} \leftarrow \sqrt{K(\cdot,\cdot) - (\boldsymbol{\kappa}_v)^\top (\boldsymbol{K}_{v,t})^{-1} \boldsymbol{\kappa}_v}$.
36:   end for
37: end for
```

Algorithm 1 presents the COOP-KERNELUCB algorithm with a tunable exploration parameter η, which can be different from the parameter $\beta_{v,t}$ used in the analysis, as is typical in this setting (Gentile et al., 2014; Chowdhury & Gopalan, 2017). We now present a regret bound for this algorithm.

Theorem 1 (Group Regret under Delayed Communication)**.** *Let $\mathbf{C}$ be a minimal clique covering of $\mathcal{G}_\gamma$. When $\mathcal{D}_{v,t}$ is continuum-armed, Algorithm 1 incurs a per-agent average regret that satisfies, with probability at least $1-\delta$,*

$$\widehat{\mathcal{R}}(T) = \mathcal{O}\left(\sqrt{\bar{\chi}(\mathcal{G}_\gamma) \cdot \frac{T}{V}} \left(R \cdot \widehat{\Upsilon}_T + \sqrt{\widehat{\Upsilon}_T} \left(B + R\sqrt{2 \log \frac{V\lambda}{\delta}} \right) \right) \right).$$

Here $\widehat{\Upsilon}_T = \max_{\mathcal{C} \in \boldsymbol{C}} \left[\log\det \left(\frac{1}{\lambda} \boldsymbol{K}_{\mathcal{C},T} + \boldsymbol{I} \right) \right]$ is the overall information gain, and for any clique $\mathcal{C} \in \mathbf{C}$, the matrix $\boldsymbol{K}_{\mathcal{C},T}$ is the Gram matrix formed by actions from all agents within $\mathcal{C}$ until time T, i.e. $(\tilde{\boldsymbol{x}}_{v,t})_{v \in \mathcal{C}, t \in [T]}$.

We first discuss the leading factors in the bound. Compared to single-agent bounds, a coarse approximation of our rate reveals an additional factor of $\mathcal{O}\left(\sqrt{\bar{\chi}(\mathcal{G}_\gamma)}\right)$. This factor arises from the delayed spread of information, and is equal to the minimum clique number of the γ power graph of the communication network. When $\mathcal{G}$ is γ-complete (i.e., $\mathcal{G}_\gamma$ is complete), $\bar{\chi}(\mathcal{G}_\gamma) = 1$, providing us the best rate. An example topology when this condition is realized include $\gamma/2$-star graphs (one node at the center, and 'spikes' of $\gamma/2$ nodes). Conversely, since we assume $\mathcal{G}$ is connected, in the worst-case graph, $\bar{\chi}(\mathcal{G}_\gamma) = \lceil V/\gamma \rceil$. in which case the regret bound is equivalent to that obtained when all agents run in isolation. This is achieved, for example, in a line graph. Now, we formalize the idea of heterogeneity among agents.

Definition 2 (Heterogeneity)**.** *In a multi-agent setting with V agents and context vector space $\mathcal{Z}$ with kernel K_z, let $\boldsymbol{K}_z$ be the matrix of pairwise interactions, i.e., $\boldsymbol{K}_z = (K(\boldsymbol{z}_v, \boldsymbol{z}_{v'}))_{v,v' \in \mathcal{V}}$. Then, the corresponding **heterogeneity** Υ_z for this setting is defined as $\Upsilon_z = rank(\boldsymbol{K}_z)$.*

For the composition considered in our paper, we can derive a regret bound in terms of Υ^x_{VT}, i.e., the information gain from VT actions $\boldsymbol{x}$ across agents and heterogeneity Υ_z.

Corollary 1. *When $K = K_z \odot K_x$, Algorithm 1 incurs a per-agent average regret, w. p. at least $1-\delta$,*

$$\widehat{\mathcal{R}}(T) = \widetilde{\mathcal{O}}\left(\Upsilon_z \cdot \Upsilon_{VT} \cdot \sqrt{\bar{\chi}(\mathcal{G}_\gamma) \cdot \frac{T}{V} \cdot \log\left(\frac{V\lambda}{\delta}\right)}\right).$$

Remark 5. *The regret bound displays a smooth interaction between the network structure, communication delays and agent similarity[4]. Corollary 1 implies that the communication effectively acts as a "mask" on the underlying cooperative performance, which is controlled primarily by the proximity of the network contexts themselves. For instance, when network contexts are identical (fully-cooperative), $\Upsilon_z = 1$, and then the network structure entirely determines the regret (via $\bar{\chi}(\mathcal{G}_\gamma)$). Conversely, if $\boldsymbol{K}_z$ is full-rank, then $\Upsilon_z = V$, and agents cannot leverage cooperation. In this case, no improvement can be obtained regardless of the density of $\mathcal{G}$.*

Examples. We now provide a few examples to illustrate the problem setting. Consider the case when K_x and K_z are both linear. In this case, the algorithm can be understood as a weighted variant of the linear UCB algorithm: for an observation $\boldsymbol{x}_{v,t}$ from an agent v to v', the "weighted" observation is given by $(\boldsymbol{z}_v^\top \boldsymbol{z}_{v'}) \cdot \boldsymbol{x}_{v,t}^\top \boldsymbol{x}_{v,t}$, and hence the "weight" is $(\boldsymbol{z}_v^\top \boldsymbol{z}_{v'})$. In an ideal implementation, the vectors $\boldsymbol{z} \in \mathbb{S}_{d-1}(1)$, i.e., the unit sphere in d dimensions, such that $\boldsymbol{z}^\top \boldsymbol{z} = 1$, ensuring that personal observations are given a weight of 1. Alternatively, when both K_z and K_x are RBF kernels, we observe an additive effect, i.e., $K = \exp\left(-\frac{\|\boldsymbol{z}_v - \boldsymbol{z}_{v'}\|^2}{2\sigma_z^2} - \frac{\|\boldsymbol{x}_v - \boldsymbol{x}_{v'}\|^2}{2\sigma_x^2}\right) = \exp\left(-\frac{\|\hat{\boldsymbol{x}}_v - \hat{\boldsymbol{x}}_{v'}\|^2}{2}\right)$, where $\hat{\boldsymbol{x}} = \begin{bmatrix} \boldsymbol{x}/\sigma_x \\ \boldsymbol{v}/\sigma_v \end{bmatrix}$. Note that the additional factor incurred in comparison to single-agent learning is $\Upsilon_z = \mathcal{O}(d \ln V)$ for RBF K_z, and for any action or network kernel, the regret can be obtained via Remark 1.

[4]We demonstrate this smooth relationship for the product kernel, i.e. $K = K_z \odot K_x$, however, alternate relationships are worth exploring. For more details and results on some forms of kernel compositions, see Appendix.

3.3. Approximating Network Contexts

The previous analysis assumes the availability of the underlying *network context* vectors $\boldsymbol{z}_v$ for each agent (or at least oracle access to the kernel K_z), however, for many applications, this information is not available, and must be estimated from the contexts themselves. Our approach is based on *kernel mean embeddings* (Blanchard et al., 2011; Christmann & Steinwart, 2010; Deshmukh et al., 2017).

Consider the network context space $\mathcal{Z}$ to be the RKHS $\mathcal{H}_{K_x}$, and we assume that the contexts $\boldsymbol{x}_{v,t}$ for each agent are drawn from an underlying probability density $\mathcal{P}_v$. The idea is to use $\boldsymbol{z}_v$ as a representation of $\mathcal{P}_v$, so that we can (with an appropriate metric), use K_z as a measure of "similarity" of the context distributions. For this, we look towards *kernel mean embeddings* of the distributions $\mathcal{P}_v$ in the RKHS $\mathcal{H}_{K_x}$. This implies that the augmented context $\tilde{\boldsymbol{x}}_{v,t}$ at any time t for any agent $v \in V$ is $(\Psi(\mathcal{P}_v), \boldsymbol{x}_{v,t})$, where $\Psi(\mathcal{P}_v) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{P}_v}[\phi_x(\boldsymbol{x})] = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{P}_v}[K_x(\cdot, \boldsymbol{x})]$ is the kernel mean embedding of $\mathcal{P}_v$ in $\mathcal{H}_{K_x}$. Using this, we can define the kernel K_z as follows.

$$K_z\left(\Psi(\mathcal{P}_v), \Psi(\mathcal{P}_{v'})\right) = \exp\left(-\|\Psi(P_v) - \Psi(P_{v'})\|_2/2\sigma_z^2\right). \tag{10}$$

We can estimate this from the available context via the *empirical mean kernel embedding* $\widehat{\Psi}_t(\mathcal{P}_v) = \frac{1}{t}\sum_{i=1}^t K_x(\cdot, \boldsymbol{x}_{v,i})$. Now, we can calculate the *empirical kernel approximation* $\widehat{K}_{z,t}(\cdot,\cdot)$ at time t:

$$\widehat{K}_{z,t}(\mathcal{P}_v, \mathcal{P}_{v'}) = \exp\left(-\text{MMD}_{\mathcal{H}}(\widehat{\Psi}_t(\mathcal{P}_v), \widehat{\Psi}_t(\mathcal{P}_{v'}))/2\sigma_z^2\right), \tag{11}$$

The empirical maximum mean discrepancy (MMD) (Gretton et al., 2012) is the measure employed to measure the divergence of the embeddings in $\mathcal{H}_{K_x}$, and is given by:

$$\text{MMD}^2_{\mathcal{H}}(\widehat{\Psi}_t(\mathcal{P}_v), \widehat{\Psi}_t(\mathcal{P}_{v'})) = \sum_{\tau,\tau'}^{t,t}(K_x(\boldsymbol{x}_{v,\tau}, \boldsymbol{x}_{v,\tau'}) + K_x(\boldsymbol{x}_{v',\tau}, \boldsymbol{x}_{v',\tau'}) - 2K_x(\boldsymbol{x}_{v,\tau}, \boldsymbol{x}_{v',\tau'})). \tag{12}$$

Our next result describes how the approximation (constructed from t samples each) $\widehat{K}_t = \widehat{K}_{z,t} \odot K_x$ deviates from the true kernel $K = K_z \odot K_x$ under this model.

Lemma 2. *For an RKHS $\mathcal{H}$, assume that $\|f\|_\infty \le d$ for all $f \in \mathcal{H}$ with $\|f\|_{\mathcal{H}} \le 1$. Then, the following is true with probability at least $1-\delta$ for all $\boldsymbol{x}_i, \boldsymbol{x}_j \in \widetilde{\mathcal{X}}$:*

$$\left|\log\left(\frac{\widehat{K}_{z,t}(\boldsymbol{x}_i, \boldsymbol{x}_j)}{K_z(\boldsymbol{x}_i, \boldsymbol{x}_j)}\right)\right| \le \frac{1}{\sigma_z^2}\left(\sup_{\mathcal{P}\in\mathfrak{P}_{\mathcal{X}}} \mathfrak{R}_t(\mathcal{H}, \mathcal{P}) + 2d\sqrt{\frac{1}{2t}\log\frac{1}{\delta}}\right).$$

Here $\mathfrak{R}_t(\mathcal{H}, \mathcal{P}_v)$ denotes the t-sample Rademacher average (Bartlett & Mendelson, 2002) of $\mathcal{H}$ under $\mathcal{P}_v \in \mathfrak{P}_{\mathcal{X}}$.

Lemma 2 implies the *consistency* of the empirical Kernel estimator, i.e., for any $v, v' \in \mathcal{V}$, $\widehat{K}_{z,t} \to K_z$ with probability 1 as $t \to \infty$. To obtain K_z we can employ any other PSD kernel K_P on $\mathcal{X}$ besides K_x as well.

Remark 6 (Regret of Simultaneous Estimation)**.** *At any instant, the empirical heterogeneity is locally controlled, i.e., for a clique cover $\mathbf{C}$ of $\mathcal{G}_\gamma$, $\Upsilon_z \leq \max_{\mathcal{C}\in\mathbf{C}} |\mathcal{C}|^2$. This follows directly from Theorem 2 of Krause & Ong (2011) and the fact that for any agent in a clique $\mathcal{C}$, the empirical kernel approximation only takes $1/2\,(|\mathcal{C}| \cdot (|\mathcal{C}|-1))$ distinct values at any instant. This implies that sparse network settings can easily be shown to benefit from cooperation (i.e., when $|\mathcal{C}| = \mathcal{O}(V^{1/4})$), but future work can address stronger controls on the group regret.*

4. Extensions

Fully-Cooperative Setting. In the fully cooperative setting, the *network contexts* for all agents are identical, and each agent is solving the same contextual bandit problem. When the decision set is fixed (i.e. $D_{v,t} = D \subset \mathcal{X}$ for all v, t, Cesa-Bianchi et al. (2019b)) we can derive a variant of COOP-KERNELUCB that provides optimal performance. The central idea of the algorithm is for centrally-positioned agents to essentially follow Algorithm 1, however, the agents that are positioned peripherally in $\mathcal{G}$ mimic the actions (obtained after the appropriate delay) of these centrally positioned agents. We partition the set of agents $\mathcal{G}$ into "central" and "peripheral" agents such that each peripheral agent is connected to at least one central agent in $\mathcal{G}_\gamma$. This algorithm is defined as DIST-KERNELUCB as this algorithm is not decentralized. It just remains to define the partition, which we do after introducing some notation.

Definition 3. *An independent set of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a set of vertices $\mathcal{V}' \subseteq \mathcal{V}$ such that no two vertices in $\mathcal{V}'$ are connected. A maximal independent set $\mathcal{V}^*$ is the largest independent set, and independence number $\alpha(\mathcal{G}) = |\mathcal{V}^*|$.*

We set the "central" agents $\mathcal{V}_C$ of $\mathcal{G}$ as the maximal weighted independent set of $\mathcal{G}_\gamma$ (where, for any node $v \in \mathcal{V}$, the weight $w_v = N_\gamma(v)$), and set the complement $\mathcal{V}_P = \mathcal{V} \setminus V_C$ as the "peripheral" set. Each peripheral agent p is assigned the central agent it is connected to (denoted as cent(p)), and in case any peripheral agent is connected to more than one central agent, we assign it to the central agent with maximum degree in $\mathcal{G}_\gamma$. The set of peripheral agents assigned to a central agent c is denoted by $\pi(c)$. We can then make the following claims about regret incurred in this setting.

Theorem 2. *$\mathcal{D}_{v,t}$ is continuum-armed,* DIST-KERNELUCB *incurs a per-agent average regret that satisfies, with probability at least $1-\delta$,*

$$\widehat{\mathcal{R}}(T) = \widetilde{\mathcal{O}}\left(\Upsilon_z \cdot \Upsilon_{VT} \cdot \sqrt{\alpha(\mathcal{G}_\gamma) \cdot \frac{T}{V} \cdot \log\left(\frac{V\lambda}{\delta}\right)}\right).$$

Here, $\alpha(\mathcal{G}_\gamma)$ refers to the independence number of $\mathcal{G}_\gamma$.

The central concept utilized in this case is to partition the network in a manner that allows for a group of agents to make identical (albeit delayed) decisions. The regret analysis uses the property that the vertex cover of the elements of V_C spans $\mathcal{G}_\gamma$, and one can bound the total regret by simply bounding the regret incurred by each "central" agent, since for any "peripheral" agent v, the regret incurred is only a constant larger than the correponding regret incurred by the "central" agent, i.e., $\mathcal{R}_v(T) \leq \widetilde{O}(\sqrt{\gamma}) + \mathcal{R}_{\text{cent}(v)}(T)$. The full proof and algorithm pseudocode are in the appendix.

Remark 7. *In addition to the tighter average per-agent regret (since $\alpha(\mathcal{G}_\gamma) \leq \bar{\chi}(\mathcal{G}_\gamma)$), we can make a stronger claim about the individual regret for any agent as well. Both these regret bounds match the rates mentioned for the context-free case in (Cesa-Bianchi et al., 2019b; Bar-On & Mansour, 2019). Moreover, when $\gamma = 1$, then the bound on the group regret matches the lower bound shown in the nonstochastic case (Cesa-Bianchi et al., 2019a).*

Eager Estimation. In addition to the algorithms mentioned above, we also consider a (potentially stronger) variant of COOP-KERNELUCB that does not take delays into account at all, and simply updates its observation set as soon as it obtains any new information (from any communicating agent). Consequently, for any arbitrary $\mathcal{G}$ and γ, this can lead to significant *drift* in the Gram matrices for any pair of agents, making this algorithm (dubbed EAGER-KERNELUCB) significantly more challenging to analyze. We defer the analysis therefore to future work and present empirical evaluations of this variant in this paper. This algorithm can be understood as Algorithm 1 run with all observations (i.e., lines 20-22 in Algorithm 1 are ignored), although we present the complete pseudocode in the appendix.

5. Experiments

The central aspect we wish to experimentally understand is the behavior of the algorithm with respect to network structures and delay in cooperative learning (alternatively, a detailed experimental comparison of single-agent KernelUCB under Gaussian noise can be found in (Srinivas et al., 2009; Krause & Ong, 2011)), and hence our experimental benchmark setup focuses on these aspects as well. We conduct two major lines of experimentation, the first on synthetically generated random networks, and the second on real-world networks subsampled from the SNAP network datasets (Leskovec & Sosič, 2016).

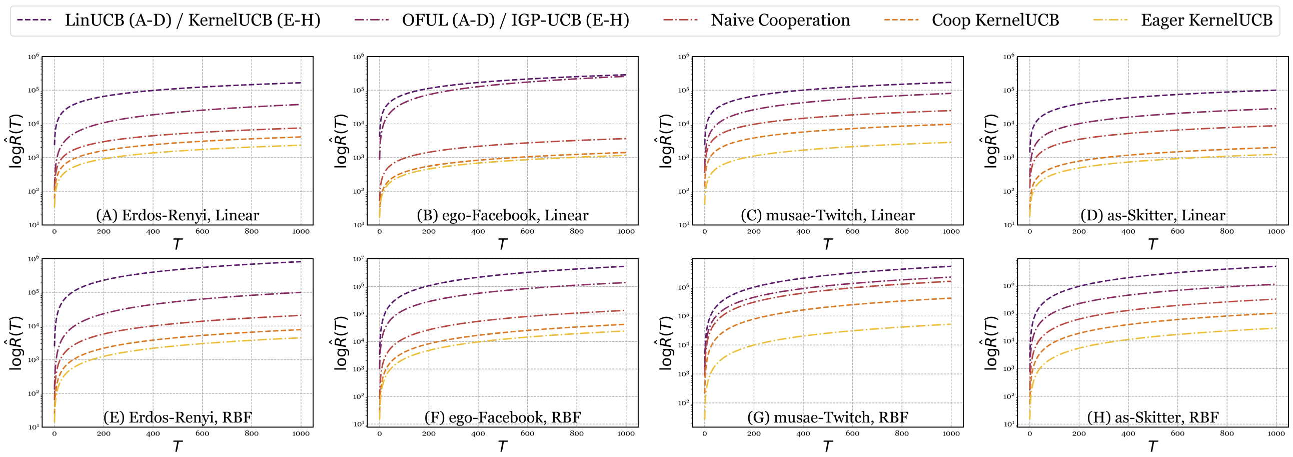

Figure 1. An experimental comparison of COOP-KERNELUCB and its variants with benchmark techniques for contextual bandits. Each experiment is averaged over 100 trials. The top row denotes the linear kernel, and the bottom row denotes the RBF kernel.

Comparison Environments. We compare in two benchmark setups. In order to compare performance with linear methods, our first setup assumes K_x is the linear kernel, and K_z is a clustering of the agents given by the independent sets of the γ^{th} power of the underlying connectivity graph $\mathcal{G}$ (K_z not known to the algorithms *a priori*, and $\gamma = \text{diameter}(\mathcal{G})/2$). This is done to motivate the central application scenario where the network connectivity and task similarity are correlated. The second setup is where K_x and K_z are both randomly initialized Gaussian kernels (where K_z is again unknown to our method). We run both setups on graphs of $V = 200$ nodes, $D_{v,t}$ is a set of 8 randomly generated contexts for all $v \in \mathcal{V}, t \in T$ and dimensionality $d = 10$ for $\mathcal{X}$ and $\mathcal{Z}$ (for setup 2). For the kernel estimation task, we set $\sigma = 1$, and we set $\lambda = 1$.

Network Structures. We run experiments on two network structures - (a) synthetic, randomly generated networks and (b) real-world networks. For the synthetic networks, we generate random connected Erdos-Renyi networks (Erdős & Rényi, 1960) of size $V = 200$ with $p = 0.7$. For the synthetic networks, we subsample V nodes and their corresponding edges (for $V = 200$) from the `ego-Facebook`, `musae-Twitch`, and `as-Skitter` networks, in order to represent a diverse set of networks found in social networks, peer-to-peer distribution and autonomous systems.

Benchmark Methods. In the linear setting, we compare against single-agent LinUCB (Li et al., 2010) (where every agent runs LinUCB independently), OFUL (Abbasi-Yadkori et al., 2011) and COOP-KERNELUCB and EAGER-KERNELUCB. In the kernel setting, we compare against single-agent KernelUCB (Valko et al., 2013), IGP-UCB (Chowdhury & Gopalan, 2017). Additionally, an important benchmark we compare against is *Naive Cooperation*, where agents run IGP-UCB (kernel) and LinUCB (linear) but include observations from neighbors as their own (without reweighting).

Results Summary. Figure 1 describes the regret achieved on each of the 4 benchmark networks for both linear and RBF (Gaussian) settings. Each plot is obtained after averaging the results for 100 trials, where the bandit contexts were refreshed every trial. We first highlight the general trend observed. Since the baseline techniques do not utilize cooperation at all, we expect them to provide a per-agent regret that scales linearly, instead of the $\mathcal{O}(1/\sqrt{V})$ dependence for our algorithms, which is obtained in our results as well. Among our algorithms, we see that COOP-KERNELUCB and DIST-KERNELUCB perform similarly for the Erdos-Renyi outperforms *Naive Cooperation* in both the linear and kernel settings, which can be attributed to the fact that naive cooperation does not take agent similarities into account. We observe that EAGER-KERNELUCB consistently outperforms other algorithms, across all benchmark tasks.

Our motivation for this algorithm stems from work in the delayed feedback regime for the stochastic (context-free) bandit (Joulani et al., 2013), which suggests that incorporating observations *as soon as* they are available can provide optimal regret. While it is challenging to derive a provably optimal algorithm in the contextual setting (and more challenging in the multi-agent case), we simply extended the "as soon as" heuristic in EAGER-KERNELUCB. The observed empirical regret suggests to us that EAGER-KERNELUCB obtains $O(\sqrt{\frac{T}{V}(\gamma + \alpha(\mathcal{G}_\gamma))})$ regret, lower than the other variants of COOP-KERNELUCB. The other variations between different graph families can be attributed to the difference in connectivity (instead of the kernel approximation).

6. Discussion and Related Work

This paper is inspired by and draws from concepts in several (often disparate) subfields within the literature. We discuss our contributions with respect to these areas sequentially.

Cooperative Multi-Agent Learning. Cooperative bandit learning with delays has maintained the setting that all agents solve the same bandit problem (i.e., fully cooperative), which our work generalizes as a first step. In the nonstochastic (multi-armed) case (without delays), this problem was first studied in the work of Awerbuch & Kleinberg (2008), where they proposed an algorithm with a per-agent regret bound of $O(\sqrt{(1+KV^{-1})\ln T})$, which matches (up to logarithmic factors) our version of the bound in the same setting (with contexts). In the multi-armed case, (Landgren et al., 2016a;b; 2018; Martínez-Rubio et al., 2019) provide algorithms whose regret scales as a function of the graph Laplacian of $\mathcal{G}$, using a *consensus* protocol (Bracha & Toueg, 1985). Our algorithms are based on a message-passing framework (i.e., LOCAL), which maintains the same communication complexity, while providing significantly better regret guarantees. Moreover, we can express the consensus protocol as an instance (albeit restricted) of our algorithm, when $K_x(i,j) = \mu_i \mathbf{1}\{i = j\}, \mu_i \in \mathbb{R}$ is a scaled simplex, and $K_z(i,j) = A_{ij}^{d(i,j)}$ is the power of the graph Laplacian. Algorithms for the nonstochastic non-contextual case with delays have been developed in (Cesa-Bianchi et al., 2019b; Bar-On & Mansour, 2019), that propose algorithms with per-agent average regret scaling as $\widetilde{\mathcal{O}}(\sqrt{\alpha(\mathcal{G}_\gamma)TV^{-1}})$ and individual regret (for agent $v \in V$) scaling as $\widetilde{\mathcal{O}}(\sqrt{(1+K|\pi(\text{cent}(v))|^{-1})T})$, which match the regret achieved by DIST-KERNELUCB in the fully cooperative *contextual* setting. A minimax regret bound for the nonstochastic context-free of $O(\sqrt{(\gamma+K)T})$ is also provided in (Cesa-Bianchi et al., 2019b), improving on the work of Neu et al. (2010), which our work improves up to smaller network factors ($\sqrt{\bar{\chi}(\mathcal{G}_\gamma)}$). When we compare our regret bounds with the algorithm-agnostic delayed feedback regret bounds provided by (Joulani et al., 2013) for the single-agent case, we observe the same relationship.

Leveraging Social Contexts. There has been extensive research in leveraging *social* side-observations across the bandit literature. Cesa-Bianchi et al. (2013) provide an algorithm called *GoB.Lin* that assumes an outer-product relationship between information flow in the network (via the graph Laplacian) and context information. This is exactly an instance of our framework, where the kernel $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is described by $\tilde{\boldsymbol{\phi}}(\boldsymbol{x}_i) A_{\otimes}^{-1} \tilde{\boldsymbol{\phi}}(\boldsymbol{x}_j)$ (in their notation), extended to the (kernel) multi-agent case with delayed feedback. In their setting, our regret bounds match exactly those of GoB.Lin. The clustering formulation can also be seen as a variant of the kernel framework, where $K_z(\boldsymbol{z}_v, \boldsymbol{z}_{v'}) = 1$ if agents belong to the same cluster, and 0 otherwise. The clustering is not known *a priori*, however, and the work of (Gentile et al., 2014; 2017; Li & Zhang, 2018) provides algorithms with tight regret guarantees for this case (our kernel embedding technique is similar in this regard). Again, we highlight that while multi-agent decision-making has been studied in the social network case (with non-identical contexts) (Korda et al., 2016; Wang et al., 2020), none, to our knowledge, consider general graph communication with delays.

Kernel Methods for Bandit Optimization. A theoretical treatment of kernelised bandit learning was first explored in the work of Valko et al. (2013), which was built on the LinUCB (Li et al., 2010) and SupLinUCB (Chu et al., 2011), that were in turn inspired by the early work of Auer et al. (2002). Our work is an improvement on the single-agent algorithms provided by Valko et al. (2013) owing to the martingale inequality presented in (Chowdhury & Gopalan, 2017), who use their result to construct improved versions single-agent Gaussian Process bandit algorithms (Krause & Ong, 2011; Srinivas et al., 2009). Our work also improves on the multi-task framework introduced by (Deshmukh et al., 2017) to the multi-agent setting with delays, along with a stronger regret bound, and an approximation guarantee for the *kernel mean embedding* approach to estimate task similarity. Recent results (Calandriello et al., 2019; Janz et al., 2020) on bandit optimization for certain kernel families can certainly be used to construct algorithmic variants with stronger guarantees on the context kernel K_x.

7. Conclusion

In this paper we presented COOP-KERNELUCB, an kernelized algorithm for decentralized, multi-agent cooperative contextual bandits and proved regret bounds of $\widetilde{O}(\sqrt{\bar{\chi}(\mathcal{G}_\gamma)T/V})$ on the average pseudo-regret, and supported our theoretical developments with experimental performance. However, there are several aspects of the kernelised cooperative bandit problem that are left as open problems. An interesting first direction is to establish suitable *lower bounds* on the group regret in cooperative decision-making with delays. An $\Omega(\sqrt{VT})$ lower bound (Bubeck et al., 2012) can be derived for a single-agent playing VT trials sequentially, however each of the $v \in V$ agents can greatly reduce their uncertainty at every trial when cooperating, hence understanding the limits of cooperation is an interesting endeavor. Next, we presented a variant EAGER-KERNELUCB of our algorithm that does not attempt to control the drift between the agent Gram matrices, that outperforms our main algorithm. We conjecture that a regret guarantee of the order $\widetilde{O}(\sqrt{(\gamma+\alpha(\mathcal{G}_\gamma))T/V})$ exists for this algorithm in the linear case, and proving this under suitable assumptions is an interesting future direction as well. Finally, extending this line of research into the Bayesian case is also worth exploring.

References

Abbasi-Yadkori, Y., Pál, D., and Szepesvári, C. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, pp. 2312–2320, 2011.

Agrawal, S. and Goyal, N. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on Learning Theory*, pp. 39–1, 2012.

Al Mamunur Rashid, S. K. L., Karypis, G., and Riedl, J. Clustknn: a highly scalable hybrid model-& memory-based cf algorithm. 2006.

Auer, P., Cesa-Bianchi, N., and Fischer, P. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

Awerbuch, B. and Kleinberg, R. Online linear optimization and adaptive routing. *Journal of Computer and System Sciences*, 74(1):97–114, 2008.

Bar-On, Y. and Mansour, Y. Individual regret in cooperative nonstochastic multi-armed bandits. *arXiv preprint arXiv:1907.03346*, 2019.

Bartlett, P. L. and Mendelson, S. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

Blanchard, G., Lee, G., and Scott, C. Generalizing from several related classification tasks to a new unlabeled sample. In *Advances in neural information processing systems*, pp. 2178–2186, 2011.

Boldrini, S., De Nardis, L., Caso, G., Le, M. T., Fiorina, J., and Di Benedetto, M.-G. mumab: A multi-armed bandit model for wireless network selection. *Algorithms*, 11(2): 13, 2018.

Bracha, G. and Toueg, S. Asynchronous consensus and broadcast protocols. *Journal of the ACM (JACM)*, 32(4): 824–840, 1985.

Bubeck, S., Cesa-Bianchi, N., et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5 (1):1–122, 2012.

Calandriello, D., Carratino, L., Lazaric, A., Valko, M., and Rosasco, L. Gaussian process optimization with adaptive sketching: Scalable and no regret. *arXiv preprint arXiv:1903.05594*, 2019.

Cano, I., Weimer, M., Mahajan, D., Curino, C., and Fumarola, G. M. Towards geo-distributed machine learning. *arXiv preprint arXiv:1603.09035*, 2016.

Cesa-Bianchi, N., Gentile, C., and Zappella, G. A gang of bandits. In *Advances in Neural Information Processing Systems*, pp. 737–745, 2013.

Cesa-Bianchi, N., Cesari, T. R., and Monteleoni, C. Cooperative online learning: Keeping your neighbors updated. *arXiv preprint arXiv:1901.08082*, 2019a.

Cesa-Bianchi, N., Gentile, C., and Mansour, Y. Delay and cooperation in nonstochastic bandits. *The Journal of Machine Learning Research*, 20(1):613–650, 2019b.

Chowdhury, S. R. and Gopalan, A. On kernelized multi-armed bandits. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 844–853. JMLR. org, 2017.

Christmann, A. and Steinwart, I. Universal kernels on non-standard input spaces. In *Advances in neural information processing systems*, pp. 406–414, 2010.

Chu, W., Li, L., Reyzin, L., and Schapire, R. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 208–214, 2011.

Deshmukh, A. A., Dogan, U., and Scott, C. Multi-task learning for contextual bandits. In *Advances in neural information processing systems*, pp. 4848–4856, 2017.

Erdős, P. and Rényi, A. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960.

Fraigniaud, P. Locality in distributed graph algorithms, 2016.

Gentile, C., Li, S., and Zappella, G. Online clustering of bandits. In *International Conference on Machine Learning*, pp. 757–765, 2014.

Gentile, C., Li, S., Kar, P., Karatzoglou, A., Zappella, G., and Etrue, E. On context-dependent clustering of bandits. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1253–1262. JMLR. org, 2017.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

Janz, D., Burt, D. R., and González, J. Bandit optimisation of functions in the mat\'ern kernel rkhs. *arXiv preprint arXiv:2001.10396*, 2020.

Joulani, P., Gyorgy, A., and Szepesvári, C. Online learning under delayed feedback. In *International Conference on Machine Learning*, pp. 1453–1461, 2013.

Korda, N., Szörényi, B., and Shuai, L. Distributed clustering of linear bandits in peer to peer networks. In *Journal of machine learning research workshop and conference proceedings*, volume 48, pp. 1301–1309. International Machine Learning Society, 2016.

Krause, A. and Ong, C. S. Contextual gaussian process bandit optimization. In *Advances in neural information processing systems*, pp. 2447–2455, 2011.

Landgren, P., Srivastava, V., and Leonard, N. E. On distributed cooperative decision-making in multiarmed bandits. In *2016 European Control Conference (ECC)*, pp. 243–248. IEEE, 2016a.

Landgren, P., Srivastava, V., and Leonard, N. E. Distributed cooperative decision-making in multiarmed bandits: Frequentist and bayesian algorithms. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 167–172. IEEE, 2016b.

Landgren, P., Srivastava, V., and Leonard, N. E. Social imitation in cooperative multiarmed bandits: Partition-based algorithms with strictly local information. In *2018 IEEE Conference on Decision and Control (CDC)*, pp. 5239–5244. IEEE, 2018.

Leskovec, J. and Sosič, R. Snap: A general-purpose network analysis and graph-mining library. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(1):1–20, 2016.

Li, L., Chu, W., Langford, J., and Schapire, R. E. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pp. 661–670, 2010.

Li, S. and Zhang, S. Online clustering of contextual cascading bandits. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Li, S., Karatzoglou, A., and Gentile, C. Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 539–548, 2016.

Li, S., Chen, W., and Leung, K.-S. Improved algorithm on online clustering of bandits. *arXiv preprint arXiv:1902.09162*, 2019.

Linial, N. Locality in distributed graph algorithms. *SIAM Journal on computing*, 21(1):193–201, 1992.

Liu, K. and Zhao, Q. Distributed learning in multi-armed bandit with multiple players. *IEEE Transactions on Signal Processing*, 58(11):5667–5681, 2010.

Martínez-Rubio, D., Kanade, V., and Rebeschini, P. Decentralized cooperative stochastic multi-armed bandits. In *Advances in Neural Information Processing Systems*, 2019.

Neu, G., Antos, A., György, A., and Szepesvári, C. Online markov decision processes under bandit feedback. In *Advances in Neural Information Processing Systems*, pp. 1804–1812, 2010.

Scarlett, J., Bogunovic, I., and Cevher, V. Lower bounds on regret for noisy gaussian process bandit optimization. *arXiv preprint arXiv:1706.00090*, 2017.

Schölkopf, B. and Smola, A. Support vector machines and kernel algorithms. In *Encyclopedia of Biostatistics*, pp. 5328–5335. Wiley, 2005.

Srinivas, N., Krause, A., Kakade, S. M., and Seeger, M. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

Suomela, J. Survey of local algorithms. 45(2):24, 2013.

Szorenyi, B., Busa-Fekete, R., Hegedus, I., Ormandi, R., Jelasity, M., and Kegl, B. Gossip-based distributed stochastic bandit algorithms. In Dasgupta, S. and McAllester, D. (eds.), *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 19–27, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL http://proceedings.mlr.press/v28/szorenyi13.html.

Thadakamaila, H., Raghavan, U. N., Kumara, S., and Albert, R. Survivability of multiagent-based supply networks: a topological perspect. *IEEE Intelligent Systems*, 19(5): 24–31, 2004.

Valko, M., Korda, N., Munos, R., Flaounas, I., and Cristianini, N. Finite-time analysis of kernelised contextual bandits. *arXiv preprint arXiv:1309.6869*, 2013.

Wang, Y., Hu, J., Chen, X., and Wang, L. Distributed bandit learning: Near-optimal regret with efficient communication. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SJxZnR4YvB.