Omitted Technical Details for "Justinian's GAAvernor: Robust Distributed Learning with Gradient Aggregation Agent"

Xudong Pan[†], Mi Zhang[†], Duocai Wu[†], Qifan Xiao[†], Shouling Ji[*,‡], and Min Yang[†]

[†]*Fudan University,* [*]*Zhejiang University,* [‡]*Ant Financial*

Please contact {xdpan18, dcwu18, qfxiao16}@fudan.edu.cn if you have questions on this material.

A Sketch of Proofs for Results in Section 4.4[1]

Provable Robustness with a Fixed Byzantine Ratio. Let us start from the basic case of *static Byzantium* of ratio β, where each worker is assumed to play a fixed role during the whole learning process. Denote I as the index set of all benign workers. In this case, we simply implement GAA with linear stateless policy h_ψ. We call h_ψ *stateless* if it has no dependence on the current state s_{t+1}. As linearity allows us to represent h_ψ as a matrix $\Lambda \in \mathbb{R}^{n\times n}$, Line 11 in Algorithm 1 is therefore written as $\alpha_{t+1} \leftarrow \Lambda\alpha_t$. In order to satisfy $\alpha_t \in \mathbb{S}^n$, we further require Λ is *a stochastic matrix*, i.e. $\sum_i \Lambda_{ij} = 1$ and $\Lambda_{ij} \geq 0$ for arbitrary i, j.

By explicitly calculating the derivative of cumulative reward $\mathcal{R}_T = \sum_{t=0}^{T-1} \gamma^t (f(\theta_{t+1}) - f(\theta_t))$ with respect to Λ, we obtain the following update rule (Line 15) for GAA's policy search.

$$\Delta\Lambda = \sum_{t=0}^{T-1} \gamma^t \sum_{i=1}^{n} \nabla f^T(\theta_t) V_i^t \frac{\partial \alpha_t^i}{\partial \Lambda} = \sum_{t=1}^{T-1} \gamma^t \sum_{i=1}^{n} \nabla f^T(\theta_t) V_i^t (\Lambda^{t-1} A_0^i) \tag{1}$$

where $(A_0^i)_{jk} = \alpha_0^k \delta_j^i$ is a constant matrix and Λ_i denotes the i-th row of Λ. Observing the similarity of $\alpha_{t+1} \leftarrow \Lambda\alpha_t$ with a diffusion process on an n-vertex complete graph with transition matrix Λ [8], we immediately arrive the following expressive interpretation of the update rule of Λ.

For simplicity, let us take $T = 2$ and worker i for example. As the updating rule reads $\Delta\Lambda = \gamma \sum_{i=1}^n \nabla f^T(\theta^1) V_i^1 A_0^i$, a reward signal $\nabla f^T(\theta_1) V_i^1$ is then fed back to the transition matrix Λ to update the transition probability from other workers to worker i. It is easy to see, when the signal $\nabla f^T(\theta_1) V_i^1 < 0$, i.e. the proposed updating direction of worker i violates the true gradient direction, the hitting probability of vertex i would decrease and its credit α_t^i would correspondingly be lowered. In one word, GAA will learn to put more credits on workers who submits gradient which is less stray from the true gradient. Moreover, since it would be shortsighted for GAA to base its policy adjustment only on the former iteration, we use the hyper-parameter T to control the time span. In the long term, it is reasonable to expect all the credits would concentrate on benign workers. Formally, we state our observation in the following lemma.

Lemma 1. *Stateless GAA with linear policy Λ in static Byzantium asymptotically learns the optimal policy Λ^*, s.t.*

$$\lim_{k\to\infty} \Lambda_k = \Lambda^* \tag{2}$$

$$(\Lambda^*)_{jl} = \delta_i^j \frac{\mathbf{1}(i \in I)}{(1-\beta)n} \tag{3}$$

With Lma. 1 at hands, we first consider the case when $P_m = \mathcal{D}$ a.e., which allows us to use standard techniques from optimization theory [4] to obtain the following convergence theorem.

Theorem 1. *In β-ratio static Byzantium, Algorithm 1 will asymptotically converge to the global optimum in $O(1/\sqrt{t})$. Formally, we have*

$$f(\theta_t) - f(\theta^*) < \frac{2RM}{\sqrt{(1-\beta)nt}} + \frac{S\eta R^2}{t} + O(e^{-t}) \tag{4}$$

where t is steps of gradient descent and R is the diameter of parameter space.

Then, we extend Thm. 1 to the case when $\mathrm{KL}(P_m||\mathcal{D})$ to prove Thm. 1. Expanding the loss in its expectation form, we have

$$|\mathcal{E}_{x\sim\mathcal{D}}[f(\theta^*, x)] - \mathcal{E}_{x\sim P_m}[f(\theta^*, x)]| \tag{5}$$

$$\leq \|f\|_\infty \int_x d|\mathcal{D} - P_m|(x) \leq \frac{\sqrt{2}}{2}\sqrt{\mathrm{KL}(\mathcal{D}||P_m)} \tag{6}$$

[1]This material accompanies the paper titled "Justinian's GAAvernor: Robust Distributed Learning with Gradient Aggregation Agent", which is fortunately to appear in USENIX Security 2020.

where the last inequality is an application of Pinsker's upper bound on total variation [5]. A similar inequality can be obtained for $f(\theta_t)$. With the triangle inequality on the left side of (4), we prove Thm. 1.

Provable Robustness with a Fluctuated Byzantine Ratio. Again we first assume $P_m = \mathcal{D}$ a.e.. As static Byzantium is a special case of *randomized Byzantium* (i.e., where the Byzantine ratio fluctuates over time), we will then focus on analyzing GAA's robustness in randomized Byzantium at large. In the following analysis, we denotes the index set of always benign workers as I, which is formally defined as $I := \{i \in [n] : p_i \equiv 0 \wedge r_i(0) = 1\}$. It is reasonable to assume $|I| \geq 1$, since otherwise all the workers are under control of the adversary which allows it to crack the learning process trivially.

Previous analysis on static Byzantium may tempt us to replace the $(1-\beta)n$ term in Thm. 1 directly with $|I|$, i.e. the cardinality of set I. However, such a naive way is in general improper since Lma. 1 is not valid any longer due to the randomness in randomized Byzantium.

Inspired by Eq. 9, we first identify the following adversarial n-armed bandit problem [3] in GAA's learning process.

Definition 1 (GAA with Adversarial n-Armed Bandit)**.** *For each iteration $t = 1, 2, \ldots$,*

1. *GAA chooses $I_t \in [n]$ from the multinomial distribution with parameter α_t.*
2. *Adversary selects n gradients $\{V_i^t\}_{i=1}^n$ and assign reward $r_{i,t} := \frac{\nabla f(\theta)^T V_i^t}{M^2} \in [0,1]$ to each arm i of the bandit.*
3. *GAA receives the reward $r_{I_t,t}$, while the reward of the other arms are not observed.*

Therefore, instead of arguing GAA learns its optimal policy asymptotically, we turn to establish an upper bound on the *pseudo-regret* of GAA in the adversarial bandit game defined above. Intuitively, the pseudo-regret of GAA up to iteration t is defined as the difference between the expectation of actual cumulative reward (i.e. $\mathbb{E}_{I_t} \sum_t r_{I_t,t}$) and the retrospectively optimal cumulative reward (i.e. $\max_i \sum_t r_{i,t}$). The intermediate result on its regret bound is established in the following proposition.

Proposition 1. *With proper initialization of parameters, the pseudo-regret of GAA when playing with the adversarial n-armed bandit in Def. 1 by policy $(\alpha_t)_{t>0}$ in Algorithm 1 is upper bounded by $\sqrt{tn\ln n}$.*

By noticing the optimal cumulative reward is attained exactly by choosing the always benign worker, we immediately establish the lower bound on the cumulative variance in gradients as a corollary of Prop. 1.

If $|I| \geq 1$, we have

$$\sum_{k=1}^{t} \|\nabla f(\theta_t) - \sum_i \alpha_i^k V_i^k\|_2^2 \leq 2M^2\left(\frac{t}{S} + \sqrt{tn\ln n}\right) \tag{7}$$

for each $t > 0$.

With the variance bound above, we can derive the following theorem.

Theorem 2. *In randomized Byzantium with $|I| \geq 1$, Algorithm 1 will asymptotically converge to the global optimum with convergence rate satisfying the following inequality*

$$f(\theta_t) - f(\theta^*) < \frac{2RM + M^2\sqrt{Sn\ln n}}{\sqrt{t}} + \frac{S\eta R^2}{t} \tag{8}$$

where t is steps of gradient descent and R is an absolute constant.

With the similar techniques at the end of the analysis for the previous case, we are able to extend Thm. 2 with Pinsker's inequality to prove Thm. 2.

B Omitted Proofs

B.1 Proof for Proposition 2

Proof.

- **Brute GAR.** Consider the slightest violation with $n = 2m$. Therefore it is easy to see, if some adversary takes $B_1 = B_2 = \ldots = B_m = E$, according to the algorithm in Sec. 3.2, $\mathcal{C}^* = \{B_i\}_{i=1}^m$. Correspondingly, $\mathcal{F}$ yields E as an arbitrary vector.
- **GeoMed.** In the case of $n = 2m$, we still take $B_1 = B_2 = \ldots = B_m = E$. As GeoMed works with $\sum_{j\neq i}\|V_i - V_j\|$, we can see for any Byzantine worker, the term is $\sum_{i=1}^m \|V_i - E\|$, while for innocent worker j, it is $m\|V_j - E\| + \sum_{1\leq i\leq m, i\neq j}\|V_i - V_j\|$. With the triangular inequality $\|V_i - V_j\| \geq \|V_i - E\| - \|V_j - E\|$, it is easy to see $\sum_{i=1}\|V_i - E\| \leq m\|V_j - E\| + \sum_{1\leq i\leq m, i\neq j}\|V_i - V_j\|$, which indicates $\mathcal{F} = E$.
- **Krum.** The slightest violation $n = 2m+2$ indicates $n - m - 2 = m$. Therefore, only if $B_1 = B_2 = \ldots = B_m = E$, any Byzantine worker will be assigned with smallest score 0. Therefore, $\mathcal{F} = E$.

□

B.2 Proof for Lemma 1

Proof. Recall with explicit calculation, in the main paper, we obtain the form of the derivative of cumulative reward w.r.t. Λ as follows.

$$\Delta\Lambda = \sum_{t=0}^{T-1}\gamma^t \sum_{i=1}^{n} \nabla f^T(\theta_t) V_i^t \frac{\partial \alpha_t^i}{\partial \Lambda} \tag{9}$$

$$= \sum_{t=1}^{T-1}\gamma^t \sum_{i=1}^{n} \nabla f^T(\theta_t) V_i^t (\Lambda^{t-1} A_0^i) \tag{10}$$

where $(A_0^i)_{jk} = \alpha_0^k \delta_j^i$ is a constant matrix and Λ_i denotes the i-th row of Λ.

Following the principle of projected gradient ascent, the corresponding updating rule for the policy parameter Λ is

$$\tilde{\Lambda}_{k+1} \leftarrow \Lambda_k + \Delta\Lambda_k \tag{11}$$

$$\Lambda_{k+1} \leftarrow P\tilde{\Lambda}_{k+1} \tag{12}$$

where P in the second line is a projection operator from $\mathbb{R}^{n\times n}$ to the set of stochastic matrices, which in practice, is implemented with a row-wise soft-max operation.

First, we consider the case when k = 0. As $\alpha_{0,0}$ is initially set as $(1/n,\dots,1/n)$ and $\Lambda_0 = \frac{1}{n}\mathbf{1}_{n\times n}$ in Algorithm 1, we therefore have

$$\Delta\Lambda_{0,i} = \sum_{t=0}^{T-1} \gamma^t \nabla f^T(\theta_t) V_i^t \mathbf{1}_n \tag{13}$$

which is exactly the cumulative variation in gradient for worker i. Therefore, by our assumption on the behavior of benign worker and Byzantine worker, we claim for $i, r \in I$ and $j \notin I$, we have

$$\Delta\Lambda_{0,i} \succ \Delta\Lambda_{0,j} \tag{14}$$

$$\Delta\Lambda_{0,i} = \Delta\Lambda_{0,r} \succ \mathbf{0} \tag{15}$$

By applying the updating rule (Eq. 11), a direct result follows as

$$\begin{aligned} \Lambda_{1,ii} &> \Lambda_{1,ji} \\ \Lambda_{1,ii} &= \Lambda_{1,rr} \\ \Lambda_{1,ii} > \Lambda_{0,ii} &= \frac{1}{n} \end{aligned} \tag{16}$$

for each $i \in I$ and $j \neq i$. Meanwhile, we have elements in each row of Λ remain identical.

Intuitively, the relations in inequality 16 present the basic case for the following lemma.

Lemma 2. *For iteration $k \geq 1$ and arbitrary $i, r \in I$, $j \notin I$, we always have*

$$\Lambda_{k,ii} > \Lambda_{k,ji} \tag{17}$$

$$\Lambda_{k,ii} = \Lambda_{k,rr} \tag{18}$$

$$\tag{19}$$

where elements in each row of Λ_k remain identical.

Proof. We proceed by induction on k. Since the basic cases have been verified above, we prove under the assumption that when these relations are valid with each $k' \leq k-1$.

Recall in Algorithm 1, α_0 always inherits the value from the last episode (c.f. line 15). Therefore, the α_0 in episode k (denoted as $\alpha_{0,k}$) is recursively calculated as

$$\alpha_{0,k} = \Lambda_{k-1}\alpha_{T-1,k-1} \tag{20}$$

By applying the recursive relation repetitively, we have

$$\alpha_{0,k} = \Lambda_{k-1}^T \alpha_{0,k-1} = \Lambda_{k-1}^T \dots \Lambda_0^T \alpha_{0,0} \tag{21}$$

Therefore we have $\alpha_{0,k}^i = \alpha_{0,k}^r > \alpha_{0,k}^j$, with a simple observation $(\Lambda\alpha)^i > (\Lambda\alpha)^j$ if $\alpha^i > \alpha^j$. Therefore, we claim matrix ΛA_0^i also satisfies the first two relation in (17), which can be seen from a simple calculation $(\Lambda_{k-1}A_0^i)_{lm} = \sum_j \Lambda_{k-1,lj}\alpha_0^m\delta_j^i = \Lambda_{k-1,li}\alpha_0^m$. Therefore, with the induction assumption and the derived relation on α, it is easy to verify the claim above. Repetitively apply the relation, we can therefore prove for each $t = 1,\dots,T-1$, $(\Lambda^{t-1}A_0^i)$ also satisfies the first two relation in (17), so as $\Delta\Lambda_k$. Finally, combined with the (14) and the nonnegative property of $(\Lambda^{t-1}A_0^i)$, it is easy to see $(\Delta\Lambda_k)_i \succ 0$. Thus, we have proved the lemma. □

With Lma. 2, we notice that each diagonal element of Λ_k with index in I increases by iteration k, while the projection operator is order-preserving. Therefore, we conclude Λ_{jj} for $j \notin I$ would vanish and thus we have $\lim_{k\to\infty}(\Lambda_k)_{jl} \to \delta_i^j \frac{\mathbf{1}(i\in I)}{(1-\beta)n}$ □

Proof for Theorem 1 According to Lma. 1 and the recurrent relation on $\alpha_{0,k}$ in Eq. 21, the mass 1 will concentrate on $\alpha_{0,k}^i$ for $i \in I$ in $O(\ln k)$ steps. Therefore, starting from episode k, we have $\alpha_{0,k}^j < \varepsilon = O(e^{-k})$ for $j \notin I$. Therefore, we have for each iteration t from episode k, the gradient variation satisfies the following inequality

$$\|\nabla f(\theta_t) - \sum_i \alpha_i^k V_i^k\|^2 \leq \|\frac{2M^2}{(1-\beta)nS}\| + \beta\varepsilon \tag{22}$$

Therefore, by standard claims from convex optimization theory [4, Thm. 6.3], we have proved the theorem.

B.3 Proof for Proposition 1

Proof. The idea of this proof is: First, we introduce the EXP3 algorithm, which is an efficient for solving the adversarial n-armed bandit problem. Then, we show a proper initialization of one recurrent unit can reduce h_ψ to the situation of executing an EXP3 algorithm. Finally, we apply the $\sqrt{tn\ln n}$ pseudo-regret bound of EXP3 algorithm to the learning process of h_ψ and therefore prove this theorem. We provide technical details on each part as follows.

Algorithm 1 (EXP3 [1])**.**

Parameter: hyperparameters $(\eta_t)_{t\in\mathbb{N}}$ s.t. $\eta_t = \sqrt{\frac{2\ln K}{nK}}$

Let p_0 be the multinomial distribution over $\{1,\dots,n\}$ with the parameter $\alpha_0 := (1/n,\dots,1/n)$.

For each iteration $t = 1,2,\dots,n$

1. *Draw an arm I_t from the multinomial distribution p_t*
2. *For each arm $i = 1,\dots,n$ compute the estimated reward $\tilde{r}_{i,t} = \frac{r_{i,t}}{\alpha_t^i}$ and update the estimated cumulative loss $\tilde{R}_{i,t} = \tilde{R}_{i,t-1} + \tilde{r}_{i,t}$*

3. Compute the new multinomial distribution $p_{t+1} := mult(\alpha_{t+1})$*, where*

$$\alpha_{t+1}^i = softmax(\tilde{R}_{i,t}) := \frac{\exp(\eta_t \tilde{R}_{i,t})}{\sum_{k=1}^n \exp(\eta_t \tilde{R}_{k,t})} \tag{23}$$

Next, we would like to construct an initialization of h_ψ so that $\alpha_{t+1} = h_\psi(\alpha_t, s_{t+1})$ simulates a step of EXP3. We construct the recurrent unit in the following way.

$$\tilde{R}_{i,t} = g(Us_t, Q\alpha_t) + W\tilde{R}_{i,t-1}) \tag{24}$$

$$\alpha_{t+1} = \text{softmax}(V\tilde{R}_{i,t}/\eta_t) \tag{25}$$

where $V = \mathbf{I}_{d\times d}, W = \gamma \mathbf{I}_{d\times d}$, U is a projection layer s.t. $Us_t = \hat{f}(\theta_t)$ and g is a multilayer perception which approximates the term $\tilde{r}_{i,t}$ when given the reward $r_{i,t} = f(\theta_t) - f(\theta_{t+1})$ by the underlying learning process. The existence of such an architecture g is a classical result called universal approximation theorem [7]. Therefore, it can be easily checked that the construction above is equivalent with EXP3 algorithm. By transplanting the $\sqrt{tn\ln n}$ pseudo-regret bound previously established on EXP3 algorithm [3, Thm. 3.1], we have proved this proposition. □

B.4 Proof for Corollary A

Proof. With the pseudo-regret bound derived in previous proposition, we have the following inequality according to the definition of pseudo-regret of the adversarial n-armed bandit in Def. 4.3.

$$\mathbb{E}_{I_t} \sum_t r_{I_t,t} - \max_i \sum_t r_{i,t} \geq -\sqrt{tn\ln n} \tag{26}$$

Expanding the term $r_{i,t}$ and the expectation by definition, we have

$$\sum_{k=1}^t \frac{\nabla f^T(\theta_t) \sum_i \alpha_t^i V_i^t}{M^2} \geq \max_i \sum_{k=1}^t \frac{\nabla f^T(\theta_t) V_i^t}{M^2} - \sqrt{tn\ln n} \tag{27}$$

According to our security assumption that not all workers are controlled by the adversary, we can lower bound the first term on the right side by $1 - \frac{\delta^2}{2}$ where δ is the variation between true gradient and estimated gradient by an always benign worker, say 0, that is, $\mathbb{E}\nabla f^T(\theta_t) V_0^t \geq (1 - \frac{\delta^2}{2}) M^2$. Therefore, by Cauchy-Schwatz inequality, we have

$$\sum_{k=1}^t \|\nabla f(\theta_t) - \sum_i \alpha_i^k V_i^k\|_2^2 \tag{28}$$

$$\leq 2M^2 t - 2\sum_{k=1}^t \nabla f^T(\theta_t) \sum_i (\alpha_t^i V_i^t) \tag{29}$$

$$\leq 2M^2 t - 2(1 - \frac{\delta^2}{2}) M^2 t + \sqrt{tn\ln n} \tag{30}$$

$$\leq 2M^2 (\frac{t}{S} + \sqrt{tn\ln n}) \tag{31}$$

where the last line comes from the batch setting i.e.

$$V_0^t := \frac{1}{S} \sum_{x\in B} \tilde{g}_i(x)$$

which reduces the variation by the following inequality.

$$\mathbb{E}\|\frac{1}{S}\sum_{i=1}^S \tilde{g}_i(x) - \nabla f(x)\|^2 = \frac{1}{S}\mathbb{E}\|\tilde{g}_i(x) - \nabla f(x)\|^2 \leq \frac{2M^2}{S} \tag{32}$$

□

Proof for Theorem 2. With the variance bound in Corollary A, it is straightforward to derive the convergence rate above with similar techniques in the proof of [4, Theorem 6.3].

B.5 Complexity Analysis

In the following analysis, we denote the number of workers as n, the number of Byzantine workers as m and the dimension of parameter as d.

- Classical: The classical GAR aggregates the gradients by (weighted) average, which consists of n operations of vector summation. Its computational cost is $O(nd)$ and the space complexity is $O(nd)$ for storing the gradients.
- GAA: In our original implementation, the main cost of GAA is to compute $\alpha_{t+1} \leftarrow h_\psi(\alpha_t, s_t)$, where the most expensive operation happens in the linear layer involving a matrix-matrix multiplication of size $(3n+2, n)$ and $(3n+2, d)$. Thus the computational complexity is $O(n^3 d)$. The space complexity involves the storage of the gradients and the parameters of h_ψ, which is $O(n^2 + nd)$. For the stateless GAA, computing α_{t+1} only requires a matrix-vector computation, which invoves a matrix of size (n, n) and a vector of size n. Combined with the cost of a linear aggregation of the gradients, the computational cost of our stateless GAA is thus $O(n^2 + nd)$. To store the parameters of h_ψ and the gradients, the space complexity of the stateless GAA is also $O(n^2 + nd)$.
- Brute-Force: Executing the Brute-Force GAR requires searching for an optimal $n - m$ subset of the submitted gradients that have the minimal maximum pairwise distance. For each subset, computing the maximum pairwise distance costs $O((n-m)d)$. Thus the total computational complexity is $O(\binom{n}{m}(n-m)d)$. For comparing the pairwise distance, Brute-Force requires storing the intermediate results, which costs additional storage of $O(\binom{n}{m})$. Thus the space complexity of Brute-Force GAR is $O(\binom{n}{m} + nd)$.
- Krum: The main cost of Krum lies in finding the $n-m-2$ closest vectors to each gradient, which costs $O(n^2 d)$ for computing the pairwise distance and $O(n^2 \log n)$ for sorting out the $n-m-2$ smallest distance. In total, the time complexity of Krum is $O(n^2 d + n^2 \log n)$, which

conforms to the statement in Krum's original paper [2]. The space complexity involves storing the pairwise distance and the gradients, which costs $O(n^2+nd)$ in total.

- GeoMed: For GeoMed, we analyze the approximative version proposed in [6], which finds the gradient that has the smallest sum of distance with other gradients. The main computational cost lies in the computing the pairwise distance, which costs (n^2d). Similar to the above analysis on Krum, the space complexity of GeoMed is $O(n^2+nd)$.
- Bulyan: The main computational cost of Bulyan lies in running the Krum over the set of the submitted gradients without replacement for $n-2m$ times. Following the analysis in [6], the pairwise distance can be reused without recomputation, which therefore costs $O(n^2d)$. To compute the aggregation, Bulyan needs to compute the median of the $n-2m$ vectors in a coordinate-wise manner, which costs nd. Therefore, the computational cost of Bulyan in total is $O(n^2d)$. Similar to Krum, the space complexity mainly involves the storage of the gradients and the pairwise distance, which costs $O(n^2+nd)$.

In summary, GAA brings computation overheads on a similar scale compared with previous defenses, which roughly corresponds to the theoretical complexity listed in Table 1. Furthermore, GAA is observed to be a little faster than previous methods. We attribute the main reason to its high compatibility with the underlying classifier, which allows it to take full advantage of GPU acceleration in a straightforward way. It may also probably because previous defenses did not provide or discuss details of their implementations and thus our implementation may not be optimal.

References

[1] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *FOCS*, 1995.

[2] Peva Blanchard, Rachid Guerraoui, Julien Stainer, et al. Machine learning with adversaries: Byzantine tolerant gradient descent. In *NeurIPS*, 2017.

[3] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 2012.

[4] Sébastien Bubeck et al. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 2015.

[5] Kai Lai Chung. *A course in probability theory*. Academic press, 2001.

[6] Rachid Guerraoui, Sébastien Rouault, et al. The hidden vulnerability of distributed learning in byzantium. In *ICML*, 2018.

[7] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 1989.

[8] Bernt Oksendal. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013.