Information and Computation **150**, 22–56 (1999)
Article ID inco.1998.2753, available online at http://www.idealibrary.com on IDEAL®

Distributed Cooperative Bayesian Learning

Kenji Yamanishi

Theory NEC Laboratory, Real World Computing Partnership, c/o Systems Basics Research Laboratory, C & C Media Research Laboratories, NEC Corporation, 1-1, 4-chome, Miyazaki, Miyazaki-ku, Kawasaki, Kanagawa 216, Japan
E-mail: yamanisi@ccm.cl.nec.co.jp

This paper addresses the issue of designing an effective distributed learning system in which a number of agent learners estimate the parameter specifying the target probability density in parallel and the population learner (for short, the p-learner) combines their outputs to obtain a significantly better estimate. Such a system is important in speeding up learning. We propose as distributed learning systems two types of the *distributed cooperative Bayesian learning strategies* (DCB), in which each agent learner or the p-learner employs a probabilistic version of the Gibbs algorithm. We analyze DCBs by giving upper bounds on their average logarithmic losses for predicting probabilities of unseen data as functions of the sample size and the population size. We thereby demonstrate the effectiveness of DCBs by showing that for some probability models, they work approximately (or sometimes exactly) as well as the nondistributed optimal Bayesian strategy, achieving a significant speed-up of learning over it. We also consider the case where the hypothesis class of probability densities is hierarchically parameterized, and there is a feedback of information from the p-learner to agent learners. In this case we propose another type of DCB based on the Markov chain Monte Carlo method, which we abbreviate as HDCB, and characterize its average prediction loss in terms of the number of feedback iterations as well as the population size and the sample size. We thereby demonstrate that for the class of hierarchical Gaussian distributions HDCB works approximately as well as the nondistributed optimal Bayesian strategy, achieving a significant speed-up of learning over it.

* An extended abstract of this paper appeared in "Proceedings of the Tenth Annual Conference on Computational Learning Theory" [16].

0890-5401/99 $30.00

1. INTRODUCTION

1.1. Problem Statement

We consider the situation where each example is generated according to an unknown parametric probability density function, which we call the target density. We are concerned with the problem of learning the target density or equivalently, estimating the parameters specifying the target density, using a *distributed learning system*. A distributed learning system consists of a number of agent learners (for short, agents) and the population learner (for short, the p-learner). Each agent independently observes a sequence of examples and outputs an estimate of the parameter specifying the target density or statistics of the examples. The p-learner does not have direct access to the random examples, but only to a set of outputs of the agents. The p-learner combines the outputs of the agents in order to obtain a significantly better estimate of the parameter for the target density.

The main purpose of designing distributed learning systems is to speed up learning utilizing the parallelism, compared to the nondistributed learning system. Here the nondistributed learning system is the system that receives all examples given to the agents and outputs an estimate of the parameter for the target density. Distributed learning systems can also meet the situation where there is sufficient communication bandwidth available for the agents to send their outputs to the p-learner, but not enough time or bandwidth for the p-learner to receive all the examples themselves.

We measure the performance of a distributed learning system in terms of the *average logarithmic loss* for predicting the probability density of an unseen example where the average is taken for the data generation and the randomness that the system may induce. We wish to design a distributed learning system such that the average logarithmic loss is as small as possible and the p-learner eventually can predict future data approximately as well as the *nondistributed Bayesian learning strategy* (abbreviated as NDB), which can observe all examples at once and attains the least average logarithmic loss.

This paper proposes two models of distributed learning: the *plain model* and the *hierarchical model*. The plain model deals with the case where each agent observes a data sequence generated according to an identical target density, and the parameter value specifying the target density is randomly generated according to a fixed prior density. The output of the p-learner is an estimate of the parameter for the target density, which is itself an output of the distributed learning system in this model.

The hierarchical model deals with the case where the target densities according to which data sequences are generated may not be identical over all agents, and the parameter values specifying the target densities are randomly generated according to an identical prior density specified by a *hyperparameter*, which is itself distributed according to a certain prior density. That is, the hierarchical model is specifically applicable to the case where the distributed information sources form a hierarchically parametrized probability distribution (e.g., hierarchical Gaussian models including the variance component model [10], hierarchical Bernoulli models), which is so often observable in real situations. In this model the p-learner outputs

an estimate of the hyperparameter, while each agent outputs an estimate of the parameter for the target density. The joint vector composed of the outputs of the agents and the p-learner is an output of the distributed learning system in this model. In each of the two models we propose specific types of distributed learning systems and analyze their performance for particular classes of probability distributions (e.g., Gaussian distributions, Poisson distributions, multidimensional discrete distributions, hierarchical Gaussian distributions, hierarchical Bernoulli distributions) and prior densities.

1.2. Previous Work

The framework of distributed learning that we propose here is very much inspired by Kearns and Seung's seminal model of *population learning* [6] (see also the work by Nakamura *et al.* [8]). Our framework is similar to theirs in that each agent independently observes a data sequence and the p-learner has access only to the outputs of the agents. The differences between our framework and Kearns and Seung's are as follows:

Kearns and Seung's model may be characterized by the following features:

(1) The target to be learned is a deterministic rule taking values in $\{0, 1\}$.

(2) The prediction loss is measured in terms of the 0-1 loss, equivalently, the discrete loss.

(3) Each agent uses a *deterministic version* of the Gibbs algorithm; i.e., a deterministic hypothesis is randomly chosen according to the uniform distribution from the set of hypotheses consistent with examples.

(4) The p-learner uses the maximum likelihood learning strategy.

(5) The performance of a distributed learning system is evaluated within the PAC (probably approximately correct) learning model.

(6) There is no feedback of information from the p-learner to agents.

In contrast, our model may be characterized by the following features:

(1)′ The target to be learned is a probability density or a probability mass function. It is assumed that prior densities of the parameter specifying the target density exist in several levels and form a hierarchical structure.

(2)′ The prediction loss is measured in terms of the average logarithmic loss.

(3)′ Each agent uses a *probabilistic version* of the Gibbs algorithm; i.e., a parameter value of a hypothesis is randomly chosen according to the Bayes posterior distribution from the whole parameter space.

(4)′ The p-learner uses a simple algebraic operation or the Gibbs algorithm.

(5)′ The performance of a distributed learning system is evaluated in terms of the *additional loss*, defined as the difference between its average logarithmic loss and the average Bayes risk for NDB (the nondistributed Bayesian learning strategy).

(6)′ There is a feedback of information from the p-learner to agents (in the hierarchical model).

In summary, our framework may be considered as a *probabilistic version of Kearns and Seung's* and also includes an extension of theirs to the case where there are a hierarchical parameter structure and a feedback loop between each agent and the p-learner.

This work is also technically related to *hierarchical Bayesian inference* [1] and *Markov chain Monte Carlo* (*MCMC*) *method* [2–4, 10, 12]. We apply MCMC to the iterative learning process induced by the feedback of information from the p-learner to agents in the hierarchical model. MCMC has mainly been applied to efficient approximations of analytically intractable Bayesian inference [2, 10, 12, 15]. This work suggests a new application of MCMC to the design of distributed learning systems.

1.3. Overview of Results

In Section 2, we first give a mathematical formalization of the plain model. We introduce the average logarithmic loss for a distributed learning system as its performance measure and show that it is lower-bounded by the average Bayes risk for NDB. Hence, it turns out that the *additional loss*, which is defined as the difference between the average logarithmic loss for a distributed learning system and the average Bayes risk for NDB, is the key quantity to be analyzed.

In the plain model we define two types of *distributed cooperative learning strategies*, which we denote by DCB1 and DCB2, respectively. DCB1 is a distributed learning system in which each agent employs the probabilistic version of the Gibbs algorithm and the p-learner employs a simple algebraic operation such as an arithmetic mean operation. DCB1 is applicable to the situation where each agent has enough computation power to run the Gibbs algorithm while the p-learner does not necessarily have it. We investigate how well DCB1 works in the cases where the hypothesis class is a class of Gaussian distributions with a constant variance and a class of Poisson distributions. Here the *hypothesis class* is a class that each agent uses, which is assumed to be the same as the class of possible target densities in our Bayesian framework. Theorems 11 and 14 give upper bounds on the additional losses for DCB1 for the cases above as functions of (1) the number m of examples that each agent observes and (2) the population size s, in order to quantify the relation between the degree of speed-up of learning and the prediction accuracy. We thereby demonstrate that DCB1 works approximately as well as NDB, achieving a significant speed-up of learning over it.

DCB2 is a distributed learning system in which each agent outputs sufficient statistics for the target density from the examples and the p-learner employs the probabilistic version of the Gibbs algorithm. DCB2 is applicable to the situation where the p-learner has enough computation power to run the Gibbs algorithm, while each agent does not necessarily have it. We give a design and analysis of DCB2 for the case where the hypothesis class belongs to the exponential family. Specifically we demonstrate that DCB2 attains exactly the same average logarithmic loss as NDB, attaining a significant speed-up of learning over it, for the classes of Gaussian distributions with a constant mean, Gaussian distributions with a constant variance, Poisson distributions, and multidimensional discrete distributions.

In Section 3, we first give a mathematical formalization of the hierarchical model of distributed learning and then introduce the distributed cooperative Bayesian learning strategy of this model, which we denote by HDCB, as a strategy in which each agent and the p-learner employ the probabilistic version of the Gibbs algorithm iteratively through the feedback of information from the p-learner to agents. Theorem 23 gives a general upper bound on the additional loss for HDCB for the case where the parameter space is bounded. The bound is obtained as a function of (1) the number m of examples that each agent observes, (2) the population size s, and (3) the number N of feedback iterations that the p-learner and agents make. We see from the bound that the additional loss for HDCB converges to zero exponentially in N.

Further, Theorem 26 shows that for the class of Gaussian distributions with a hierarchical parameter structure, the variation distance between the average probability density for HDCB and that for NDB is upper-bounded by $O(\exp(-(c_1 N \ln m)/\ln(sN))) + O((N \ln m)/(\ln(sN)) \exp(-c_2 N))$, where $0 < c_1,\ c_2 < \infty$ are constants. Thus, for the total sample size $\ell = sm$, setting $s = O(\sqrt{\ell})$ and $N = O(\ln \ell)$, the variation distance between the average probability density for HDCB and that for NDB can be made $O((\ln \ell)/\ell)$ while HDCB achieves a speed-up in computation time from $\Omega(\ell)$ for NDB to $O(\sqrt{\ell} \ln \ell)$.

In Section 4, we discuss extensions of DCBs to the following two cases: one is the multi-hierarchically parameterized case, and the other is the general decision-theoretic case where a general real-valued function may be used as a hypothesis and a loss function other than the logarithmic loss may be used as a distortion measure for prediction. Specifically, in the latter case, we relate a generalized version of DCB1 to Vovk's aggregating strategies [13] and the notion of extended stochastic complexity [14].

2. PLAIN MODEL

2.1. Model

Let s be a positive integer. A *distributed learning system* $\mathscr{S}$ consists of s *agent learners* (for short, *agents*) and a single *population learner* (for short, a *p-learner*). We call the number s the *population size*.

Let $\mathscr{D}$ be a measurable space. Let $\mathscr{C} = \{ p(D \mid \theta) : \theta \in \Theta \subset \mathbf{R}^k \}$ be a given class of probability density functions over $\mathscr{D}$ specified by a k-dimensional real-valued parameter θ, belonging to a parameter space Θ. (Note: Throughout the paper we use the terminology "a probability density function" or simply "a probability density" assuming that $\mathscr{D}$ is continuous, but it should be replaced with a "probability mass function" when $\mathscr{D}$ is a discrete space.) We call $\mathscr{C}$ the *hypothesis class*. Now we make the following assumption on the data generation:

ASSUMPTION 1. (1) *Each agent independently observes a sequence of examples, each of which is independently generated according to an identical* target probabilty density function (*for short, a* target density) $p(D \mid \theta)$, *belonging to* $\mathscr{C}$.

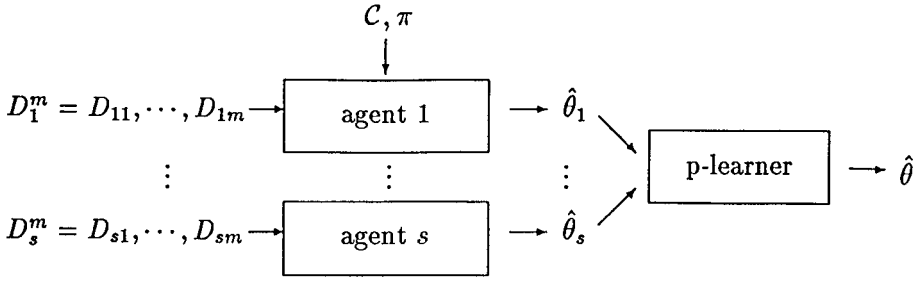

FIG. 1. Distributed learning system of Type 1: Plain model.

(2) *The value of the parameter θ specifying the target density is unknown and is generated according to a known prior probability density function $\pi(\theta)$ (for short,* a prior density) *over Θ.*

We call the model of learning in which Assumption 1 is satisfied the plain model.

We may consider two types of distributed learning systems, which we name *Type* 1 and *Type* 2, respectively. In a *distributed learning system $\mathscr{S}$ of Type* 1 in the plain model agents and the p-learner perform as follows: Let a positive integer m be given. For each i ($=1, ..., s$), the ith agent takes as input a sequence $D_i^m = D_{i1} \cdots D_{im}$ of m examples, $\mathscr{C}$, and π, then outputs an estimate of θ. Letting $\hat{\theta}_i$ be the output of the ith agent, the p-learner takes $\hat{\theta}_1, ..., \hat{\theta}_s$ as input, then outputs an estimate $\hat{\theta}$ as a function of $\hat{\theta}_1, ..., \hat{\theta}_s$, which is itself an output of $\mathscr{S}$. Note that the p-learner does not have direct access to the random examples, $\mathscr{C}$, or π, but only to a set of outputs of the agents. (See Fig. 1.)

In a *distributed learning system $\mathscr{S}$ of Type* 2 in the plain model agents and the p-learner perform as follows: Letting ϕ be a given function $\phi: \mathscr{Y}^* \to \mathbf{R}^*$ for some positive integer d, for each i ($=1, ..., s$), the ith agent takes a sequence $D_i^m = D_{i1} \cdots D_{im}$ as input, then outputs $\phi(D_i^m)$. The p-learner takes $\phi(D_1^m), ..., \phi(D_s^m)$, $\mathscr{C}$, and π as input, then outputs an estimate $\hat{\theta}$, which is itself an output of $\mathscr{S}$. (See Fig. 2.)

The difference between Type 1 and Type 2 is that in Type 1, the agents take the most intelligent action and the p-learner just combines the outputs of the agents without knowing $\mathscr{C}$ or π, while in Type 2, the agents just send statistics to the p-learner and the p-learner takes the most intelligent action, knowing $\mathscr{C}$ and π. This difference just originated from that of the situations. Type 1 meets the situation where each agent has enough computation power to learn θ from the training examples (specifically, to make a random number generation in order to run the

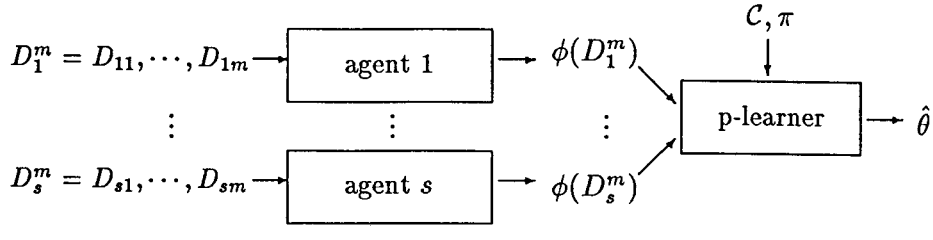

FIG. 2. Distributed learning system of Type 2: Plain model.

Gibbs algorithm), while the p-learner does not necessarily have it. The p-learner in Type 1 might be thought of as a nonexpert general manager who makes a decision on the basis of his subordinates' reports only. On the other hand, Type 2 meets the situation where each agent does not necessarily have enough computation power to learn θ from examples, while only the p-learner has it.

Further, Type 1 allows us to design algorithms for agents and the p-learner independently so that we can exchange algorithms for the agents and the p-learner with some others, independently, in order to make the distributed learning system stronger.

The amount of information sent from each agent to the p-learner for Type 2 is larger than that for Type 1 when the dimension of ϕ is larger than that of θ. Hence Type 1 might be more appropriate than Type 2 for the case where there is not sufficient large communication bandwidth between each agent and the p-learner.

Remark. Although we assume, for the sake of analytical simplicity, that the sample size m is uniform over all agents, the model can be immediately extended into a general case where the sample size is not uniform.

Let $D^{sm} = D_1^m \cdots D_s^m$ be the *training sequence*. For a distributed learning system $\mathscr{S}$, let $q(\theta \mid D^{sm})$ be the probability density according to which the output $\hat{\theta}$ of the p-learner is generated. Then we can think of a mixture density $\int p(D \mid \theta)\, q(\theta \mid D^{sm})\, d\theta$ as an average probability density of D induced by D^{sm} for $\mathscr{S}$. We measure the performance of $\mathscr{S}$ in terms of its *average logarithmic loss* $L(\mathscr{S})$ for predicting the probability density of an unseen example using the average probability density for $\mathscr{S}$, where $L(\mathscr{S})$ is defined by

$$L(\mathscr{S}) \stackrel{\text{def}}{=} E_\theta E_{D^{sm}\mid\theta} E_{D\mid\theta}\left[-\ln \int p(D \mid \theta)\, q(\theta \mid D^{sm})\, d\theta \right],$$

where E_θ, $E_{D^{sm}\mid\theta}$, and $E_{D\mid\theta}$ denote the expectations taken with respect to $\pi(\theta)$, $p(D^{sm} \mid \theta)$, and $p(D \mid \theta)$, respectively, and ln denotes the natural logarithm. Note that the range of $L(\mathscr{S})$ is $(-\infty, \infty)$.

Let $p^*(\theta \mid D^{sm})$ be the *Bayes posterior density* over Θ from D^{sm} defined as

$$p^*(\theta \mid D^{sm}) = \frac{\pi(\theta) \prod_{i=1}^s \prod_{j=1}^m p(D_{ij} \mid \theta)}{\int \pi(\theta) \prod_{i=1}^s \prod_{j=1}^m p(D_{ij} \mid \theta)\, d\theta}.$$

We define the *average Bayes risk* L^* for the sample size sm by

$$L^* \stackrel{\text{def}}{=} E_\theta E_{D^{sm}\mid\theta} E_{D\mid\theta}\left[-\ln \int p(D \mid \theta)\, p^*(\theta \mid D^{sm})\, d\theta \right],$$

which is obtained by plugging $q(\theta \mid D^{sm}) = p^*(\theta \mid D^{sm})$ to the formula of $L(\mathscr{S})$. We define the *nondistributed Bayesian learning strategy* (in the plain model), which we abbreviate as *NDB*, as a strategy which takes as input D^{sm} at once (not in parallel) then chooses an estimate $\hat{\theta}$ randomly according to $p^*(\theta \mid D^{sm})$ and outputs it. We can think of L^* as the average logarithmic loss for NDB for the sample size sm. Below we show a general relationship between $L(\mathscr{S})$ and L^*.

LEMMA 2. *For a distributed learning system* $\mathscr{S}$ (*both of Type* 1 *and Type* 2), *let* $q(\theta \mid D^{sm})$ *be the probability density according to which the output* $\hat{\theta}$ *of the p-learner in* $\mathscr{S}$ *is generated, and let*

$$m^*(D \mid D^{sm}) \stackrel{\text{def}}{=} \int p(D \mid \theta)\, p^*(\theta \mid D^{sm})\, d\theta,$$

$$m_{\mathscr{S}}(D \mid D^{sm}) \stackrel{\text{def}}{=} \int p(D \mid \theta)\, q(\theta \mid D^{sm})\, d\theta,$$

$$D(m^* \| m_{\mathscr{S}}) \stackrel{\text{def}}{=} \int m^*(D \mid D^{sm}) \ln \frac{m^*(D \mid D^{sm})}{m_{\mathscr{S}}(D \mid D^{sm})}\, dD,$$

where $m^*(D \mid D^{sm})$ *is the average probability density function of* D *induced by the training sequence* D^{sm} *for NDB,* $m_{\mathscr{S}}(D \mid D^{sm})$ *is the one induced for the distributed learning system* $\mathscr{S}$, *and* $D(m^* \| m_{\mathscr{S}})$ *is the Kullback–Leibler divergence between* m^* *and* $m_{\mathscr{S}}$ *for fixed* D^{sm}. *Then for any distributed learning system* $\mathscr{S}$, *the following equation holds,*

$$L(\mathscr{S}) = L^* + E_{D^{sm}}[D(m^* \| m_{\mathscr{S}})], \tag{1}$$

where $E_{D^{sm}}$ *denotes the expectation taken with respect to* $p(D^{sm}) = \int \pi(\theta)\, p(D^{sm} \mid \theta)\, d\theta$.

Proof of Lemma 2. Observe first that for any f independent of θ,

$$\begin{aligned} E_\theta E_{D^{sm}|\theta} E_{D|\theta}[f] &= \int d\theta \pi(\theta) \int dD^{sm} p(D^{sm} \mid \theta) \int dD p(D \mid \theta)\, f \\ &= \int dD^{sm} \int dD \int d\theta p(D^{sm})\, p(D \mid \theta) \left(\frac{\pi(\theta)\, p(D^{sm} \mid \theta)}{p(D^{sm})} \right) f \\ &= \int dD^{dm}\, p(D^{sm}) \int dD \int d\theta p(D \mid \theta)\, p^*(\theta \mid D^{sm})\, f \\ &= E_{D^{sm}} E_{D \mid D^{sm}}[f], \end{aligned}$$

where $E_{D \mid D^{sm}}$ and $E_{D^{sm}}$ denote the expectations taken with respect to $m^*(D \mid D^{sm}) = \int p(D \mid \theta)\, p^*(\theta \mid D^{sm})\, d\theta$ and $p(D^{sm}) = \int \pi(\theta)\, p(D^{sm} \mid \theta)\, d\theta$, respectively. Since m^* and $m_{\mathscr{S}}$ are both independent of θ by their definitions, the above relation immediately yields

$$\begin{aligned} L(\mathscr{S}) &= E_\theta E_{D^{sm}|\theta} E_{D|\theta}[-\ln m^*(D \mid D^{sm})] + E_\theta E_{D^{sm}|\theta} E_{D|\theta} \left[\ln \frac{m^*(D \mid D^{sm})}{m_{\mathscr{S}}(D \mid D^{sm})} \right] \\ &= L^* + E_\theta E_{D^{sm}|\theta} E_{D|\theta} \left[\ln \frac{m^*(D \mid D^{sm})}{m_{\mathscr{S}}(D \mid D^{sm})} \right] \\ &= L^* + E_{D^{sm}} E_{D \mid D^{sm}} \left[\ln \frac{m^*(D \mid D^{sm})}{m_{\mathscr{S}}(D \mid D^{sm})} \right] \\ &= L^* + E_{D^{sm}}[D(m^* \| m_{\mathscr{S}})]. \end{aligned}$$

This completes the proof of Lemma 2. ∎

Since $D(m^* \| m_{\mathscr{S}}) \geqslant 0$ holds in general, we immediately see from (1) that

$$L(\mathscr{S}) \geqslant L^*,$$

where the equality holds if and only if

$$q(\theta \mid D^{sm}) = p^*(\theta \mid D^{sm}), \tag{2}$$

for every θ and D^{sm}. A distributed learning system that realizes (2) is *ideal* in the sense that it can attain the same average logarithmic loss as NDB. However, (2) does not necessarily hold for most distributed learning systems with $s(\geqslant 2)$ agents. Then the effectiveness of a distributed learning system is measured in terms of how large its average logarithmic loss is deviated from L^*. Hence we may measure the performance of a distributed learning system $\mathscr{S}$ in terms of $L(\mathscr{S}) - L^*$, which we call the *additional average logarithmic loss* (for short, the *additional loss*) for $\mathscr{S}$. We wish to design a distributed learning system such that the additional loss is as small as possible, while attaining a significant speed-up of learning over NDB.

Remark. Let L^{**} be the average logarithmic loss associated with the target density $p(D \mid \theta)$, i.e., $L^{**} \stackrel{\text{def}}{=} E_\theta E_{D^{sm} \mid \theta} E_{D \mid \theta}[-\ln p(D \mid \theta)]$. Then the Bayes risk for NDB is not smaller than L^{**}, i.e., $L^* \geqslant L^{**}$. However, only the strategy that knows θ can attain L^{**}. Note that NDB achieves the least average logarithmic loss over all learning strategies that are *not given any information about the parameter θ for the target density other than the prior density π, in advance.*

2.2. Algorithms

The performance of a distributed learning system depends on what types of algorithms are used for the agents and the p-learner. We first introduce a specific form of distributed learning systems of Type 1.

DEFINITION 3. We define the *distributed cooperative Bayesian learning strategy of Type* 1 (in the plain model), which we abbreviate as DCB1, as the distributed learning system of Type 1 satisfying the following:

(1) Each agent employs the *Gibbs algorithm*, i.e., the ith agent takes as input $D_i^m = D_{i1} \cdots D_{im}$, $\mathscr{C}$, and π, then outputs the parameter value $\hat{\theta}_i$ chosen randomly according to the Bayes posterior density $p(\theta \mid D_i^m)$, which is calculated as

$$p(\theta \mid D_i^m) = \frac{\pi(\theta) \prod_{j=1}^m p(D_{ij} \mid \theta)}{\int \pi(\theta') \prod_{j=1}^m p(D_{ij} \mid \theta')\, d\theta'} \qquad (i = 1, ..., s).$$

(2) The p-learner receives $\hat{\theta}_1, ..., \hat{\theta}_s$ as input, then outputs the parameter value of the form

$$f(\hat{\theta}_1, ..., \hat{\theta}_s), \tag{3}$$

where f is a given function from Θ^s to Θ, which does not depend on any D^{sm}.

Remarks. (1) In designing DCB1, it is assumed that each agent has enough computation power to make random samplings according to a given probability distribution. On the other hand, the p-learner is only required to perform a simple algebraic operation (3), which should be designed as simple as possible in order to make the computational complexity for the p-learner as small as possible. As will be seen in Examples 4 and 5, this paper specifically considers the case where f in (3) is the *arithmetic mean operation.*

(2) As will be seen in Example 4 (Gaussian distributions) and Example 5 (Poisson distributions), there exist some cases where the running times required by the agent and the p-learner to run the Gibbs algorithm are linear in the sample size m and the population size s, respectively. For such cases the computation time for each agent is $O(m)$, while that for NDB is $\Omega(sm)$. Further assume that the computation time for the p-learner in DCB1 is $O(s)$ (this assumption is satisfied for the case where f in (3) is the arithmetic mean operation). Then the overall computational time for DCB1 is $O(s+m)$. For the total sample size $\ell = sm$, the overall computation time for DCB1 is $O(\sqrt{\ell})$ if we set $s = \Theta(\sqrt{\ell})$ and $m = \Theta(\sqrt{\ell})$ (i.e., s and m are set to be of the same order so that $(s+\ell/s)$ is made smallest). Then we say that *DCB*1 *achieves a speed-up in computation time from* $\Omega(\ell)$ *for NDB to* $O(\sqrt{\ell})$.

EXAMPLE 4. Let $\mathscr{D} = \mathbf{R}$ and $\mathscr{C} = \{N(\theta, \sigma^2 : D)\colon \theta \in (-\infty, \infty),\ \sigma \text{ is known}\}$. Here $N(\theta, \sigma^2 : D)$ denotes the Gaussian distribution with mean θ and variance $\sigma^2 : (1/\sqrt{2\pi}\,\sigma)\exp(-(D-\theta)^2/2\sigma^2)$. (*Note*: We may write $N(\theta, \sigma^2)$ instead of $N(\theta, \sigma^2 : D)$ when D is trivial from the context.) Let the prior density of θ be $N(\theta_0, \sigma_0^2 : \theta)$ where $\theta_0 \in (-\infty, \infty)$ and $0 < \sigma_0 < \infty$ are known. Then the ith agent outputs $\hat{\theta}_i$ chosen randomly according to the following probability density:

$$\begin{aligned}\hat{\theta}_i &\sim p(\theta \mid D_i^m) \\ &= N\left(\frac{\sigma_0^2 \sum_{j=1}^m D_{ij} + \sigma^2\theta_0}{m\sigma_0^2 + \sigma^2}, \frac{\sigma^2\sigma_0^2}{m\sigma_0^2 + \sigma^2}\right) \qquad (i = 1, ..., s). \end{aligned} \tag{4}$$

Here $p(\theta \mid D_i^m)$ is easily derived by noting that the joint density of θ and D_i^m is proportional to $\exp(-(\theta-\theta_0)^2/2\sigma_0^2)\exp(-\sum_{j=1}^m (D_{ij}-\theta)^2/2\sigma^2)$.

Let the p-learner use the arithmetic mean operation as f in (3); i.e., the p-learner outputs $\hat{\theta}$ calculated as

$$\hat{\theta} = \frac{1}{s}\sum_{i=1}^{s} \hat{\theta}_i, \tag{5}$$

then $\hat{\theta}$ is generated according to the probability density

$$\begin{aligned}\hat{\theta} &\sim q(\theta \mid D^{sm}) \\ &= N\left(\frac{\sigma_0^2(\sum_{i=1}^s \sum_{j=1}^m D_{ij}) + s\sigma^2\theta_0}{s(m\sigma_0^2 + \sigma^2)}, \frac{\sigma^2\sigma_0^2}{s(m\sigma_0^2 + \sigma^2)}\right),\end{aligned}$$

which can be verified by transforming the random variables $(\theta_1, ..., \theta_s)$ with joint density $\prod_{i=1}^{m} p(\theta_i \mid D_i^m)$ into $(\hat{\theta}, \xi_1, ..., \xi_{s-1})$, where

$$\begin{aligned}
\hat{\theta} &= (1/s) \sum_{i=1}^{s} \theta_i, \\
\xi_1 &= (\theta_1 - \theta_2)/\sqrt{2}, \\
\xi_2 &= (\theta_1 + \theta_2 - 2\theta_3)/\sqrt{6}, \\
&\vdots \\
\xi_{s-1} &= (\theta_1 + \theta_2 + \cdots + \theta_{s-1} - (s-1)\theta_s)/\sqrt{s(s-1)}
\end{aligned}$$

and examining the joint density of $\hat{\theta}, \xi_1, ..., \xi_{s-1}$. (Note: $\hat{\theta}, \xi_1, ..., \xi_{s-1}$ are independent.)

On the other hand, by noting that the joint density of θ and D^{sm} is proportional to $\exp(-(\theta-\theta_0)^2/2\sigma_0^2) \exp(-\sum_{i=1}^{s} \sum_{j=1}^{m} (D_{ij}-\theta)^2/2\sigma^2)$, it is verified that the Bayes posterior density $p^*(\theta \mid D^{sm})$ is given by

$$p^*(\theta \mid D^{sm}) = N\left(\frac{\sigma_0^2(\sum_{i=1}^{s} \sum_{j=1}^{m} D_{ij}) + \sigma^2\theta_0}{sm\sigma_0^2 + \sigma^2}, \frac{\sigma^2\sigma_0^2}{sm\sigma_0^2 + \sigma^2}\right). \tag{6}$$

In this case $q(\theta \mid D^{sm})$ and $p^*(\theta \mid D^{sm})$ differ from each other when $s \geqslant 2$ and $\sigma_0 < \infty$. We bound the additional loss induced by this difference subsequently. It is straightforward to extend the above analysis to the case where θ is multidimensional.

Note here that the running time required by the ith agent to compute $\hat{\theta}_i$ with density (4) and that for NDB to compute $\hat{\theta}$ with density (6) are both linear in the sample size, since the computation time required for a random sampling according to a Gaussian distribution specified by a given parameter value is $O(1)$ in s and m, supposing that a program for a random number generation according to a Gaussian distribution is given to all the agents. There actually exists an efficient algorithm for generating random numbers according to a given Gaussian distribution (see Chap. 4 in [7]).

EXAMPLE 5. Let $\mathscr{D} = \mathbf{Z}^+ \cup \{0\}$ and let $\mathscr{C} = \{e^{-\theta}\theta^D/D!: \theta > 0\}$ be a class of Poisson distributions. Let the prior distribution of θ be the gamma distribution with density $G(\alpha, \beta) = \theta^{\alpha-1} e^{-\theta/\beta}/\beta^\alpha \Gamma(\alpha)$, where $\alpha > 0$ and $\beta > 0$ are given, and Γ denotes the gamma function defined as $\Gamma(x) = \int_0^\infty e^{-t} t^{x-1}\, dt$ $(x > 0)$. Then each agent outputs $\hat{\theta}_i$ chosen randomly according to the following probability density:

$$\begin{aligned}
\hat{\theta}_i &\sim p(\theta \mid D_i^m) \\
&= G\left(\alpha + \sum_{j=1}^{m} D_{ij}, \left(m + \frac{1}{\beta}\right)^{-1}\right) \qquad (i = 1, ..., s).
\end{aligned} \tag{7}$$

Let the p-learner output $\hat{\theta}$ using the arithmetic mean operation as in (5), then the probability density of $\hat{\theta}$ is given by

$$\begin{aligned}\hat{\theta} &\sim q(\theta \mid D^{sm}) \\ &= G\left(s\alpha + \sum_{i=1}^{s}\sum_{j=1}^{m} D_{ij}, \left(sm + \frac{s}{\beta}\right)^{-1}\right),\end{aligned}$$

which can be verified by transforming the random variables $(\theta_1, ..., \theta_s)$ with joint density $\prod_{i=1}^{m} p(\theta_i \mid D_i^m)$ into $(\hat{\theta}, \xi_1, ..., \xi_{s-1})$, where $\hat{\theta} = (1/s)\sum_{i=1}^{s} \theta_i$, $\xi_1 = \theta_2/\theta_1$, ..., $\xi_{s-1} = \theta_s/\theta_1$ and examining the joint density of $\hat{\theta}, \xi_1, ..., \xi_{s-1}$.

On the other hand, the Bayes posterior density $p^*(\theta \mid D^{sm})$ is given by

$$p^*(\theta \mid D^{sm}) = G\left(\alpha + \sum_{i=1}^{s}\sum_{j=1}^{m} D_{ij}, \left(sm + \frac{1}{\beta}\right)^{-1}\right). \tag{8}$$

Here the derivations of $p(\theta \mid D_i^m)$ and $p^*(\theta \mid D^{sm})$ are done similarly to those for Example 4.

In this case $q(\theta \mid D^{sm})$ and $p^*(\theta \mid D^{sm})$ differ from each other when $s \neq 1$. Note here that the running time required by the ith agent to compute $\hat{\theta}_i$ with density (7) and that for NDB to compute $\hat{\theta}$ with density (8) are both linear in the sample size, since the computation time required for a random sampling according to a fixed gamma distribution is $O(1)$, supposing that a program for a random number generation according to a gamma distribution is given to all of the agents. There actually exists an efficient algorithm for generating random numbers according to a given gamma distribution (see Chap. 4 in [7]).

Next we consider the design of distributed learning systems of Type 2 for the case where the p-learner is allowed to take as input $\mathscr{C}$ and π as well as $\hat{\theta}_1, ..., \hat{\theta}_s$, while each agent employs a simple algebraic operation.

We say that $\mathscr{C} = \{p(D \mid \theta) : \theta \in \Theta\}$ belongs to the *exponential family* if and only if for some positive integer d, for some scalar-valued functions g, h, κ_l, and ψ_l $(l = 1, ..., d)$, any probability density $p(D \mid \theta)$ in $\mathscr{C}$ is decomposed as

$$p(D \mid \theta) = g(\theta)\, h(D) \exp\left(-\sum_{l=1}^{d} \kappa_l(\theta)\, \psi_l(D)\right).$$

Supposing that the data generation is independent, we see that for any given $D^{\ell} = D_1 \cdots D_{\ell}$, the joint density of D^{ℓ} is given by

$$p(D^{\ell} \mid \theta) = (g(\theta))^{\ell}\left(\prod_{j=1}^{\ell} h(D_j)\right) \exp\left(-\sum_{l=1}^{d} \kappa_l(\theta) \sum_{j=1}^{\ell} \psi_l(D_j)\right),$$

where a d-dimensional statistics: $\phi(D^{\ell}) = (\sum_{j=1}^{\ell} \psi_1(D_j), ..., \sum_{j=1}^{\ell} \psi_d(D_j))$ is called a *sufficient statistics* for θ (from D^{ℓ}). Here is a general form of distributed cooperative Bayesian learning strategy of Type 2 for classes belonging to the exponential family.

DEFINITION 6. Suppose that $\mathscr{C} = \{p(D|\theta): \theta \in \Theta\}$ belongs to the exponential family and that for some positive integer d, for some scalar-valued functions g, h, κ_l, and ψ_l ($l = 1, ..., d$), each probability density $p(D|\theta)$ in $\mathscr{C}$ is decomposed as $p(D|\theta) = g(\theta)\, h(D) \exp(-\sum_{l=1}^{d} \kappa_l(\theta)\, \psi_l(D))$. We define the *distributed cooperative Bayesian learning strategy of Type* 2 (in the plain model), which we abbreviate as DCB2, as the distributed learning system of Type 2 satisfying the following:

(1) The ith agent takes as input a sequence $D_i^m = D_{i1} \cdots D_{im}$ of examples, then outputs a d-dimensional vector:

$$\phi(D_i^m) = (\phi^{(1)}(D_i^m), ..., \phi^{(d)}(D_i^m)) = \left(\sum_{j=1}^{m} \psi_1(D_{ij}), ..., \sum_{j=1}^{m} \psi_d(D_{ij}) \right). \tag{9}$$

(2) The p-learner receives as input $\phi(D_1^m), ..., \phi(D_s^m)$, $\mathscr{C}$, and π, then outputs the parameter value $\hat{\theta}$ chosen randomly according to $p(\theta|\phi(D_1^m), ..., \phi(D_s^m))$ calculated as

$$p(\theta|\phi(D_1^m), ..., \phi(D_s^m)) = \frac{\pi(\theta)(g(\theta))^{sm} \exp(-\sum_{l=1}^{d} \kappa_l(\theta)\, x_l)}{\int \pi(\theta')(g(\theta'))^{sm} \exp(-\sum_{l=1}^{d} \kappa_l(\theta')\, x_l)\, d\theta'}, \tag{10}$$

where

$$x_l = \sum_{i=1}^{s} \phi^{(l)}(D_i^m) \qquad (l = 1, ..., d).$$

Remark. In designing DCB2, it is assumed that the p-learner has enough computation power to make random samplings according to a given probability distribution. On the other hand, each agent is only required to perform a simple algebraic operation to compute $\phi(D_i^m)$.

EXAMPLE 7. For $\mathscr{C}$ and π in Example 4, we see that $\mathscr{C}$ belongs to the exponential family with $d = 1$ and $\psi_1(D) = D$. We may design DCB2 in which $\phi(D_i^m)$ as in (9) and $p(\theta|\phi(D_1^m), ..., \phi(D_s^m))$ as in (10) are given by

$$\phi(D_i^m) = \sum_{j=1}^{m} D_{ij}, \tag{11}$$

$$p(\theta|\phi(D_1^m), ..., \phi(D_s^m)) = N\left(\frac{sm\sigma_0^2 x + \sigma^2 \theta_0}{sm\sigma_0^2 + \sigma^2}, \frac{\sigma^2 \sigma_0^2}{sm\sigma_0^2 + \sigma^2} \right), \tag{12}$$

where $x = \sum_{i=1}^{s} \phi(D_i^m)$. Note that $p(\theta|\phi(D_1^m), ..., \phi(D_s^m))$ matches $p^*(\theta|D^{sm})$ as in (6).

The running time for the ith agent to compute (11) is $O(m)$ and that for the p-learner to compute $\hat{\theta}$ with density (12) is $O(s)$, since the computation time required for a random sampling according to a Gaussian distribution specified by a fixed parameter value is $O(1)$ in s and m, supposing that a program for a random number generation according to a Gaussian distribution is given to the p-learner.

EXAMPLE 8. Let $\mathscr{D} = \mathbf{R}$ and $\mathscr{C} = \{N(\mu, \theta^2 : D)\colon \theta \in (0, \infty),\ \mu \text{ is known}\}$. We see that $\mathscr{C}$ belongs to the exponential family with $d = 2$. Let the prior distribution of θ be the inverse gamma distribution with density $IG(\alpha, \beta) = (\beta^\alpha e^{-\beta/\theta})/(\Gamma(\alpha)\,\theta^{\alpha+1})$, where α and β are given positive constants. We may design DCB2 in which $\phi(D_i^m)$ as in (9) and $p(\theta \mid \phi(D_1^m), ..., \phi(D_s^m))$ as in (10) are calculated as

$$\phi(D_i^m) = (\phi^{(1)}(D_i^m), \phi^{(2)}(D_i^m)) = \left(\sum_{j=1}^{m} D_{ij}, \sum_{j=1}^{m} D_{ij}^2 - \frac{1}{m}\left(\sum_{j=1}^{m} D_{ij} \right)^2 \right), \tag{13}$$

$$p(\theta \mid \phi(D_1^m), ..., \phi(D_s^m)) = IG\left(sm + \alpha, \frac{1}{2}\left(sm\left(\frac{x}{sm} - \mu \right)^2 + y + 2\beta \right) \right), \tag{14}$$

where $x = \sum_{i=1}^{s} \phi^{(1)}(D_i^m)$ and $y = \sum_{i=1}^{s} \phi^{(2)}(D_i^m)$.

The running time for the ith agent to compute (13) is $O(m)$ and that for the p-learner to compute $\hat{\theta}$ with density (14) is $O(s)$, since the computation time required for a random sampling according to a fixed inverse gamma distribution is $O(1)$ in s and m, supposing that a program for a random number generation according to an inverse gamma distribution is given to the p-learner. There actually exists an efficient algorithm for generating random numbers according to a given inverse gamma distribution (see Chap. 4 in [7]).

EXAMPLE 9. For $\mathscr{C}$ and π in Example 5, we see that $\mathscr{C}$ belongs to the exponential family with $d = 1$. We may design DCB2 in which $\phi(D_i^m)$ as in (9) and $p(\theta \mid \phi(D_1^m), ..., \phi(D_s^m))$ as in (10) are given by

$$\phi(D_i^m) = \sum_{j=1}^{m} D_{ij}, \tag{15}$$

$$p(\theta \mid \phi(D_1^m), ..., \phi(D_s^m)) = G(\alpha + x, (sm + 1/\beta)^{-1}), \tag{16}$$

where $x = \sum_{i=1}^{s} \phi(D_i^m)$.

The running time for the ith agent to compute (15) is $O(m)$ and that for the p-learner to compute $\hat{\theta}$ with density (16) is $O(s)$, since the computation time required for a random sampling according to a fixed gamma distribution is $O(1)$ in s and m, supposing that a program for a random number generation according to a gamma distribution is given to the p-learner.

EXAMPLE 10. Let $\mathscr{D} = \{0, 1, ..., k\}$ and $\mathscr{C} = \{p(l \mid \theta) = \theta_l\ \ (l = 0, 1, ..., k)\colon\ \theta = (\theta_0, \theta_1, ..., \theta_k) \in [0, 1]^k,\ \sum_{l=0}^{k} \theta_l = 1\}$, which is the class of k-dimensional discrete distributions. We see that $\mathscr{C}$ belongs to the exponential family with $d = k$. Let the prior distribution of θ be the Dirichlet distribution with density $D(\alpha_0, ..., \alpha_k) = (\Gamma(\alpha_0 + \cdots + \alpha_k)/\Gamma(\alpha_0) \cdots \Gamma(\alpha_k))\ \theta_0^{\alpha_0 - 1} \cdots \theta_k^{\alpha_k - 1}$, where $\alpha_0, ..., \alpha_k$ are given positive constants. We may design DCB2 in which $\phi(D_i^m)$ as in (9) and $p(\theta \mid \phi(D_1^m), ..., \phi(D_s^m))$ as in (10) are calculated as

$$\phi(D_i^m) = (\phi^{(1)}(D_i^m), ..., \phi^{(k)}(D_i^m)) = (m_i(1), ..., m_i(k)), \tag{17}$$

$$p(\theta \mid \phi(D_1^m), ..., \phi(D_s^m)) = D(\alpha_0 + x_0, ..., \alpha_k + x_k), \tag{18}$$

where $m_i(l)$ is the number of occurrences of $D=l$ in D_i^m, $x_l=\sum_{i=1}^{s}\phi^{(l)}(D_i^m)$ $(i=1, ..., s,\ l=1, ..., k)$, and $x_0=sm-\sum_{l=1}^{k}x_l$.

The running time for the ith agent to compute (17) is $O(km)$ and that for the p-learner to compute $\hat{\theta}$ with density (18) is $O(ks)$, since the computation time required for a random sampling according to a fixed Dirichlet distribution is $O(1)$ in s and m, supposing that a program for a random number generation according to a Dirichlet distribution is given to the p-learner. There actually exists an efficient algorithm for generating random numbers according to a given Dirichlet distribution (see Chap. 4 in [7]).

2.3. *Analysis*

This section first gives analyses of DCB1 for the particular classes introduced in the previous section. Theorem 11 gives an upper bound on the additional loss for DCB1 in Example 4.

THEOREM 11. *For the class of Gaussian distributions with a constant variance as in Example* 4, *for sufficiently large* $\ell=sm$, *the additional loss for DCB*1 *is given as*

$$L(\mathrm{DCB1})-L^*=\frac{\sigma^2\sigma_0^2}{2(m\sigma_0^2+\sigma^2)^2}\left(\frac{s-1}{s}\right)^2(1+o(1)), \tag{19}$$

where $o(1)$ *tends to zero as* $\ell=sm$ *goes to infinity.*

COROLLARY 12. *For the total sample size* $\ell=sm$, *setting* $s=\Theta(\sqrt{\ell})$, *the additional loss for DCB*1 *in Example* 4 *can be made* $O(1/\ell)$ *while DCB*1 *achieves a speed-up in computation time from* $\Omega(\ell)$ *for NDB to* $O(\sqrt{\ell})$.

Remarks. (1) When $\ell=sm$ is fixed, substituting $m=\ell/s$ into (19), we see that the main term of the additional loss for DCB1 is $\Theta(s^2)$, which is an increasing function of s. When m is fixed, we see that as s increases, the main term of the additional loss for DCB1 would approach $\sigma^2\sigma_0^2/2(m\sigma_0^2+\sigma^2)^2$, which can be thought of as an inevitable loss due to the parallelism.

(2) As seen from (19), for any $\varepsilon>0$, if we set $m=\Theta(1/\sqrt{\varepsilon})$ and $s=\Theta(1/\sqrt{\varepsilon})$, then the additional loss for DCB1 can be made at most ε, while DCB1 achieves a speed-up in computation time from $\Omega(1/\varepsilon)$ for NDB to $O(1/\sqrt{\varepsilon})$.

(3) Consider the case where π is the *uniform prior* for θ, which can be thought of as the Gaussian prior $N(\theta_0, \sigma_0^2)$ with $\sigma_0=\infty$. This prior density makes $q(\theta\,|\,D^{sm})$ completely coincide with $p^*(\theta\,|\,D^{sm})$. Thus setting $s=\Theta(\sqrt{\ell})$, DCB1 using the uniform prior attains the same average logarithmic loss as NDB, while DCB1 achieves a speed-up in computation time from $\Omega(\ell)$ for NDB to $O(\sqrt{\ell})$.

Proof of Theorem 11. We start with a lemma concerning the Kullback–Leibler divergence between two Gaussian distributions.

LEMMA 13. *For $p = N(\mu_1, \sigma_1^2)$ and $q = N(\mu_2, \sigma_2^2)$, the Kullback–Leibler divergence $D(p \| q)$ is given by*

$$D(p \| q) = \ln \frac{\sigma_2}{\sigma_1} + \frac{1}{2\sigma_2^2}(\sigma_1^2 + (\mu_1 - \mu_2)^2) - \frac{1}{2}.$$

(The proof of Lemma 13 is omitted.)

Next note that for $p(D \mid \theta) = N(\theta, \sigma_1^2 : D)$ and $\pi(\theta) = N(\mu, \sigma_2^2 : \theta)$ for constants μ, σ_1 and σ_2, the mixture density $m(D) = \int p(D \mid \theta)\, \pi(\theta)\, d\theta$ is proportional to $\int \exp(-(D-\theta)^2/2\sigma_1^2) \exp(-(\theta-\mu)^2/2\sigma_2^2)\, d\theta$, which is also proven to be proportional to $\exp(-(D-\mu)^2/2(\sigma_1^2+\sigma_2^2))$. This implies that $m(D) = N(\mu, \sigma_1^2 + \sigma_2^2 : D)$. Applying this fact into the cases of $\pi(\theta) = q(\theta \mid D^{sm})$ and $\pi(\theta) = p^*(\theta \mid D^{sm})$ obtained in Example 4, we see that $m^*(D \mid D^{sm})$ and $m_{\mathscr{S}}(D \mid D^{sm})$ are calculated as follows:

$$m^*(D \mid D^{sm}) = N\left(\frac{\sigma_0^2 \sum_{i,j} D_{ij} + \sigma^2\theta_0}{sm\sigma_0^2 + \sigma^2}, \sigma^2 + \frac{\sigma^2\sigma_0^2}{sm\sigma_0^2 + \sigma^2}\right),$$

$$m_{\mathscr{S}}(D \mid D^{sm}) = N\left(\frac{\sigma_0^2 \sum_{i,j} D_{ij} + s\sigma^2\theta_0}{s(m\sigma_0^2 + \sigma^2)}, \sigma^2 + \frac{\sigma^2\sigma_0^2}{s(m\sigma_0^2 + \sigma^2)}\right).$$

Using Lemma 13 we see that $D(m^* \| m_{\mathscr{S}})$ is expanded as follows:

$$D(m^* \| m_{\mathscr{S}}) = \frac{1}{2} \ln \frac{\sigma^2 + \sigma^2\sigma_0^2/s(m\sigma_0^2 + \sigma^2)}{\sigma^2 + \sigma^2\sigma_0^2/(sm\sigma_0^2 + \sigma^2)} + \frac{\sigma^2 + \sigma^2\sigma_0^2/(sm\sigma_0^2 + \sigma^2)}{2(\sigma^2 + \sigma^2\sigma_0^2/s(m\sigma_0^2 + \sigma^2))}$$
$$+ \frac{\sigma^4\sigma_0^4(\sum_{i,j} D_{ij} - sm\theta_0)^2\, ((s-1)/s)^2}{2(\sigma^2 + (\sigma^2\sigma_0^2/s(m\sigma_0^2 + \sigma^2)))(m\sigma_0^2 + \sigma^2)^2\, (sm\sigma_0^2 + \sigma^2)^2} - \frac{1}{2}. \tag{20}$$

Letting $x = 1/(sm + s(\sigma^2/\sigma_0^2)) = O(1/sm)$ and $y = 1/(sm + (\sigma^2/\sigma_0^2)) = O(1/sm)$, the sum of the first and second terms in the right-hand side of (20) is evaluated for sufficiently large $\ell = sm$ as follows:

$$\frac{1}{2} \ln \frac{1+x}{1+y} + \frac{1+y}{2(1+x)} = \frac{1}{2}\left(x - \frac{x^2}{2} - y + y^2 - O(1/(sm)^3)\right)$$
$$+ \frac{1}{2}(1+y)(1 - x + x^2 + O(1/(sm)^3))$$
$$= \frac{1}{2} + \frac{1}{4}\left(\frac{s-1}{s}\right)^2 \frac{\sigma^4\sigma_0^4}{(sm\sigma_0^2 + \sigma^2)^2\, (m\sigma^2 + \sigma^2)^2} + O(1/(sm)^3).$$

The third term in the right-hand side of (20) is $\sigma^2/(2(m\sigma_0^2 + \sigma^2)^2)((s-1)/s)^2 \times ((1/sm)\sum_{i,j} D_{ij} - \theta_0)^2\, (1 + o(1))$. Hence, we have

$$D(m^* \| m_{\mathscr{S}}) = \frac{\sigma^2}{2(m\sigma_0^2 + \sigma^2)^2}\left(\frac{s-1}{s}\right)^2 \left(\frac{1}{sm} \sum_{i,j} D_{ij} - \theta_0\right)^2 (1 + o(1)), \tag{21}$$

where $o(1)$ tends to zero uniformly with respect to D^{sm} as $\ell = sm$ goes to infinity. Note here that it is immediately proven that

$$E_{D^{sm}}\left[\left(\frac{1}{sm}\sum_{i,j} D_{ij} - \theta_0\right)^2\right] = \sigma_0^2 + \frac{\sigma^2}{sm}. \tag{22}$$

Taking an expectation of (21) with respect to D^{sm} and then using (22) yield

$$E_{D^{sm}}[D(m^* \| m_{\mathscr{S}})] = \frac{\sigma^2\sigma_0^2}{2(m\sigma_0^2+\sigma^2)^2}\left(\frac{s-1}{s}\right)^2 (1+o(1)).$$

Combining this fact with Lemma 2 yields (19). This completes the proof of Theorem 11. ∎

Theorem 14 gives an upper bound on the additional loss for DCB1 in Example 5.

THEOREM 14. *For the class of Poisson distributions as in Example* 5, *for sufficiently large* $\ell = sm$, *the additional loss for DCB*1 *is upper-bounded as*

$$L(\mathrm{DCB1}) - L^* \leqslant \frac{2\alpha}{m}\left(\frac{s-1}{s}\right)(1+o(1)), \tag{23}$$

where $o(1)$ *tends to zero as* $\ell = sm$ *goes to infinity.*

COROLLARY 15. *For the total sample size* $\ell = sm$, *setting* $s = \Theta(\sqrt{\ell})$, *the additional loss for DCB*1 *in Example* 5 *can be made* $O(1/\sqrt{l})$ *while DCB*1 *achieves a speed-up in computation time from* $\Omega(\ell)$ *for NDB to* $O(\sqrt{\ell})$.

Remarks. (1) As seen from (23), the main term of the additional loss for DCB1 is $\Theta(s)$ for fixed $\ell = sm$, which is an increasing function of s. Further we see that as s increases *for fixed* m, the main term of the right-hand side of (23) would approach $2\alpha/m$, which can be thought of as an inevitable loss due to the parallelism.

(2) As seen from (23), for any $\varepsilon > 0$, if we set $m = \Theta(1/\varepsilon)$ and $s = \Theta(1/\varepsilon)$, then the additional loss for DCB1 can be made at most ε, while DCB1 achieves a speed-up in computation time from $\Omega(1/\varepsilon^2)$ for NDB to $O(1/\varepsilon)$.

Proof of Theorem 14. First note that for $p(D\,|\,\theta) = e^{-\theta}\theta^D/D!$ and $\pi(\theta) = G(\alpha, \beta)$ for constants $\alpha, \beta > 0$, the mixture density $m(D) = \int p(D\,|\,\theta)\,\pi(\theta)\,d\theta$ is $(1+1/\beta)^{(\alpha+D)}\,\Gamma(\alpha+D)/(D!\,\Gamma(\alpha))$. Applying this fact into the cases of $\pi(\theta) = q(\theta\,|\,D^{sm})$ and $\pi(\theta) = p^*(\theta\,|\,D^{sm})$ obtained in Example 5 and writing $\sum_{i,j} D_{i,j}$ as τ for the sake of notational simplicity, $m^*(D\,|\,D^{sm})$ and $m_{\mathscr{S}}(D\,|\,D^{sm})$ are calculated as follows:

$$m^*(D\,|\,D^{sm}) = \frac{(sm+1/\beta)^{\alpha+\tau}\,\Gamma(\alpha+D+\tau)}{(1+sm+1/\beta)^{D+\alpha+\tau}\,D!\,\Gamma(\alpha+\tau)},$$

$$m_{\mathscr{S}}(D\,|\,D^{sm}) = \frac{(sm+s/\beta)^{s\alpha+\tau}\,\Gamma(s\alpha+D+\tau)}{(1+sm+s/\beta)^{D+s\alpha+\tau}\,D!\,\Gamma(s\alpha+\tau)}.$$

Using these formulas we immediately obtain

$$\ln \frac{m^*(D \mid D^{sm})}{m_{\mathscr{S}}(D \mid D^{sm})} = (D + s\alpha + \tau) \ln(1 + sm + s/\beta) - (D + \alpha + \tau) \ln(1 + sm + 1/\beta)$$

$$+ (\alpha + \tau) \ln(sm + 1/\beta) - (s\alpha + \tau) \ln(sm + s/\beta)$$

$$+ \ln \frac{\Gamma(\alpha + D + \tau)\, \Gamma(s\alpha + \tau)}{\Gamma(s\alpha + D + \tau)\, \Gamma(\alpha + \tau)}$$

$$= D \ln \frac{1 + sm + s/\beta}{1 + sm + 1/\beta} \tag{24}$$

$$+ s\alpha \ln \frac{1 + sm + s/\beta}{sm + s/\beta} + \alpha \ln \frac{sm + 1/\beta}{1 + sm + 1/\beta} \tag{25}$$

$$+ \tau \ln \frac{(1 + sm + s/\beta)(sm + 1/\beta)}{(sm + s/\beta)(1 + sm + 1/\beta)} \tag{26}$$

$$+ \ln \frac{\Gamma(\alpha + D + \tau)\, \Gamma(s\alpha + \tau)}{\Gamma(s\alpha + D + \tau)\, \Gamma(\alpha + \tau)}. \tag{27}$$

As for the expectation of D, we have

$$E_{D^{sm}} E_{D \mid D^{sm}}[D] = E_{D^{sm}} \left[\sum_D D \int p(D \mid \theta)\; p^*(\theta \mid D^{sm})\, d\theta \right]$$

$$= E_{D^{sm}} \left[\int \theta p^*(\theta \mid D^{sm})\, d\theta \right] \tag{28}$$

$$= E_{D^{sm}} \left[\left(\alpha + \sum_{i,j} D_{i,j} \right) (sm + 1/\beta)^{-1} \right] \tag{29}$$

$$= \alpha(1 + \beta sm)(sm + 1/\beta)^{-1} \tag{30}$$

$$= \alpha\beta(1 + o(1)), \tag{31}$$

where $o(1)$ tends to zero as $\ell = sm$ goes to infinity. In deriving (28) we have used the fact that the mean of the Poisson distribution with density $e^{-\theta}\theta^D/D!$ is θ. In deriving (29) and (30) we have used the fact that the mean of the gamma distribution with density $G(\alpha, \beta)$ is $\alpha\beta$. Thus, taking expectations of (24) with respect to D and D^{sm} and then plugging (31) into the expected form of (24) yields

$$E_{D^{sm}} E_{D \mid D^{sm}} \left[D \ln \frac{1 + sm + s/\beta}{1 + sm + 1/\beta} \right] = \frac{\alpha}{m} \left(\frac{s - 1}{s} \right) (1 + o(1)), \tag{32}$$

where we have used the fact that $\ln((1 + sm + s/\beta)/(1 + sm + 1/\beta)) = (s - 1) \times (1 + o(1))/\beta sm$.

As for (25), we have

$$s\alpha \ln \frac{1+sm+s/\beta}{sm+s/\beta} + \alpha \ln \frac{sm+1/\beta}{1+sm+1/\beta} = \frac{\alpha}{m}\left(\frac{s-1}{s}\right)(1+o(1)). \tag{33}$$

Further notice that both (26) and (27) are not larger than zero when $s \geqslant 1$. Thus, taking a sum of (32) and (33), we see that the expected value of $\ln(m^*(D \mid D^{sm})/m_{\mathscr{S}}(D \mid D^{sm}))$ is upper-bounded by $(2\alpha(s-1)/sm)(1+o(1))$. Hence we obtain (23) using Lemma 2. This completes the proof of Theorem 14. ▮

Next we give an analysis of DCB2 for general classes belonging to the exponential family.

THEOREM 16. *Suppose that* $\mathscr{C} = \{p(D \mid \theta): \theta \in \Theta\}$ *belongs to the exponential family and that for some positive integer* d, *for some scalar-valued functions* g, h, κ_l, *and* ψ_l $(l = 1, ..., d)$, *each density* $p(D \mid \theta)$ *in* $\mathscr{C}$ *is decomposed as* $p(D \mid \theta) = g(\theta)\, h(D) \times \exp(-\sum_{l=1}^{d} \kappa_l(\theta)\, \psi_l(D))$. *Suppose that the computation time for a random sampling of* θ *according to* (10) *is* $O(ds)$. *Then for the total sample size* $\ell = sm$, *the additional loss for DCB2 with* $s = \Theta(\sqrt{\ell})$ *can be made zero while DCB2 achieves a speed-up in computation time from* $\Omega(\ell)$ *for NDB to* $O(d\sqrt{\ell})$.

COROLLARY 17. *For each of Examples* 7, 8, *and* 9, *for the total sample size* $\ell = sm$, *the additional loss for DCB2 with* $s = \Theta(\sqrt{\ell})$ *can be made zero while DCB2 achieves a speed-up in computation time from* $\Omega(\ell)$ *for NDB to* $O(\sqrt{\ell})$. *For the class of k-dimensional discrete distributions as in Example* 10, *the additional loss for DCB2 with* $s = \Theta(\sqrt{\ell})$ *can be made zero, while DCB2 achieves a speed-up in computation time from* $\Omega(\ell)$ *for NDB to* $O(k\sqrt{\ell})$.

Proof of Theorem 16. For $\mathscr{C}$ satisfying the assumption as in Theorem 16, the Bayes posterior density of θ for NDB is given by

$$p^*(\theta \mid D^{sm}) = \frac{\pi(\theta) \prod_{i=1}^{s} \prod_{j=1}^{m} p(D_{ij} \mid \theta)}{\int \pi(\theta) \prod_{i=1}^{s} \prod_{j=1}^{m} p(D_{ij} \mid \theta)\, d\theta}$$

$$= \frac{\pi(\theta)(g(\theta))^{sm} \exp(-\sum_{l=1}^{d} \kappa_l(\theta) \sum_{i=1}^{s} \psi_l(D_i^m)))}{\int \pi(\theta)(g(\theta))^{sm} \exp(-\sum_{l=1}^{d} \kappa_l(\theta) \sum_{i=1}^{s} \psi_l(D_i^m))\, d\theta},$$

which coincides with $p(\theta \mid \phi(D_1^m), ..., \phi(D_s^m))$ as in (10). Hence the average logarithmic loss for DCB2 coincides with that for NDB.

Since the running time for each agent to compute (9) is $O(dm)$ and that for the p-learner to compute $\hat{\theta}$ chosen randomly according to (21) is $O(ds)$ under the assumption as in Theorem 16, the total computation time for DCB2 is $O(d(m+s)) = O(d\sqrt{\ell})$ letting $s = \Theta(\sqrt{\ell})$, while that for NDB is $\Omega(\ell)$. This completes the proof of Theorem 16. ▮

3. HIERARCHICAL MODEL

3.1. Model

In this section we consider the model in which probability densities according to which examples are generated are not identical over all agents, but there exists a hyperparameter that relates all the densities to one another. A distributed learning system $\mathscr{S}$ in this model consists of s agents and a single p-learner as in the plain model, where the role of the agent is to estimate the parameter for the target density while that of the p-learner is to estimate the hyperparameter. The main difference between this model and the plain model is a probabilistic assumption on the generation of examples and parameters. Below we give a mathematical form of this assumption. Let $\mathscr{C} = \{ p(D \mid \theta): \theta \in \Theta \subset \mathbf{R}^k \}$ be a given hypothesis class of probability densities specified by a k-dimensional real-valued parameter vector θ, belonging to a parameter space Θ.

ASSUMPTION 18. (1) *Each agent independently observes a sequence of examples each of which is independently generated according to the* target density $p(D \mid \theta_i)$ $(i = 1, ..., s)$, *belonging to* $\mathscr{C}$.

(2) *The values of the parameters* θ_i $(i = 1, ..., s)$ *are unknown, and each of them is independently generated according to a known prior density* $\pi(\theta \mid \mu)$ *over* Θ *where* μ *is an unknown hyperparameter.*

(3) *The value of the hyperparameter* μ *is generated according to a known prior density* $\pi(\mu)$.

We call the model in which Assumption 18 is satisfied the *hierarchical model*. We set $\omega = (\theta_1, ..., \theta_s, \mu)$, which is a $(ks+1)$-dimensional real-valued vector. Let Θ and $\mathscr{M}$ be the ranges of θ and μ, respectively. We denote the range of ω by $\Omega = \Theta^s \times \mathscr{M}$.

Let positive integers m, N, and an initial value $\hat{\mu}^{(0)}$ be given. A *distributed learning system* $\mathscr{S}$ in the hierarchical model makes the following iteration process: At the lth iteration step $(l = 1, 2, ..., N)$, for each i $(= 1, ..., s)$, the ith agent takes as input a sequence $D_i^m = D_{i1} \cdots D_{im}$ of m examples, $\hat{\mu}^{(l-1)}$, $\mathscr{C}$, and $\pi(\theta \mid \mu)$, then outputs an estimate $\hat{\theta}_i^{(l)}$ of θ_i. The p-learner takes as input $\hat{\theta}_1^{(l)} \cdots, \hat{\theta}_s^{(l)}$, $\pi(\theta \mid \mu)$, and $\pi(\mu)$, then outputs an estimate $\hat{\mu}^{(l)}$ of μ. The output $\hat{\mu}^{(l)}$ of the p-learner is sent back to all of the agents. This process goes on iteratively with respect to l. The vector of parameter values, $\hat{\omega} = (\hat{\theta}_1^{(N)}, ..., \hat{\theta}_s^{(N)}, \hat{\mu}^{(N)})$ obtained after N-times repetitions of the iteration process is an output of $\mathscr{S}$. (See Fig. 3.)

Let $D^{sm} = D_1^m \cdots D_s^m$ be the *training sequence*. Let $q(\omega \mid D^{sm})$ be the probability density according to which the output $\hat{\omega}$ of $\mathscr{S}$ is generated. Let $\mathbf{D} = (D_1 \cdots D_s)$

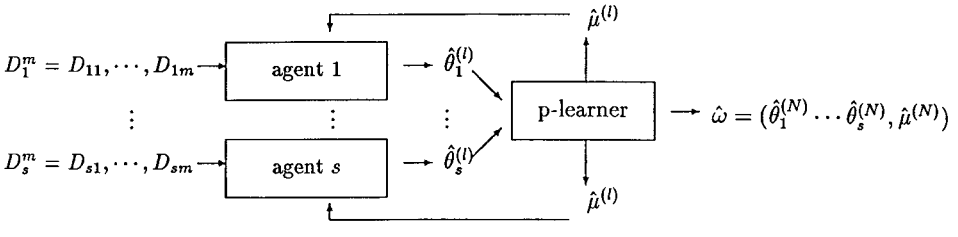

FIG. 3. Distributed learning system: Hierarchical model.

where D_i denotes a random variable representing an example that the ith agent observes. We write the probability density of $\mathbf{D}$ as

$$p(\mathbf{D}\,|\,\omega) = p(D_1 \cdots D_s\,|\,\omega) = \prod_{i=1}^{s} p(D_i\,|\,\theta_i).$$

We define $m_{\mathscr{S}}(\mathbf{D}\,|\,D^{sm})$ as

$$m_{\mathscr{S}}(\mathbf{D}\,|\,D^{sm}) \stackrel{\text{def}}{=} \int p(\mathbf{D}\,|\,\omega)\, q(\omega\,|\,D^{sm})\, d\omega, \tag{34}$$

which can be thought of as an average probability density function of D induced by D^{sm} for $\mathscr{S}$. We measure the performance of a distributed learning system $\mathscr{S}$ by the *average logarithmic loss* $L(\mathscr{S})$ for predicting the probability density of an unseen example using the average probability density $m_{\mathscr{S}}(\mathbf{D}\,|\,D^{sm})$, where $L(\mathscr{S})$ is defined as

$$L(\mathscr{S}) \stackrel{\text{def}}{=} E_{\omega} E_{D^{sm}|\omega} E_{\mathbf{D}|\omega}[-\ln m_{\mathscr{S}}(\mathbf{D}\,|\,D^{sm})],$$

where E_{ω}, $E_{D^{sm}|\omega}$, and $E_{\mathbf{D}|\omega}$ denote the expectations taken with respect to $\pi(\omega)$ $(=\pi(\mu)\,\pi(\theta\,|\,\mu))$, $p(D^{sm}\,|\,\omega)(=\prod_{i=1}^{s}\prod_{j=1}^{m} p(D_{ij}\,|\,\theta_i))$, and $p(\mathbf{D}\,|\,\omega)$, respectively.

Similar to the plain model, we define the *Bayes posterior density* $p^*(\omega\,|\,D^{sm})$ of ω from D^{sm} as

$$p^*(\omega\,|\,D^{sm}) = \frac{\pi(\mu)\prod_{i=1}^{s}(\pi(\theta_i\,|\,\mu)\prod_{j=1}^{m} p(D_{ij}\,|\,\theta_i))}{\int \pi(\mu')\prod_{i=1}^{s}(\pi(\theta'_i\,|\,\mu')\prod_{j=1}^{m} p(D_{ij}\,|\,\theta'_i))\, d\mu'\, d\theta'_1 \cdots d\theta'_s}, \tag{35}$$

and the mixture density $m^*(\mathbf{D}\,|\,D^{sm})$ as

$$m^*(\mathbf{D}\,|\,D^{sm}) \stackrel{\text{def}}{=} \int p(\mathbf{D}\,|\,\omega)\, p^*(\omega\,|\,D^{sm})\, d\omega, \tag{36}$$

then also define the *average Bayes risk* L^* by

$$L^* \stackrel{\text{def}}{=} E_{\omega} E_{D^{sm}|\omega} E_{\mathbf{D}|\omega}[-\ln m^*(\mathbf{D}\,|\,D^{sm})].$$

We define the *nondistributed Bayesian learning strategy* (in the hierarchical model), which we denote by *NDB* as with the plain model, as a strategy which takes as input D^{sm} at once (not in parallel) then outputs $\hat{\omega}$ chosen randomly according to $p^*(\omega\,|\,D^{sm})$. We can think of $m^*(D\,|\,D^{sm})$ as the average probability density of D induced by the training sequence for NDB, and can think of L^* as the average logarithmic loss for NDB. We define the *additional average logarithmic loss* (for short, *additional loss*) for $\mathscr{S}$ by $L(\mathscr{S}) - L^*$. It can be proven as with Lemma 2 that the additional loss is not less than zero. It is zero if and only if $q(\omega\,|\,D^{sm}) = p^*(\omega\,|\,D^{sm})$ for all ω and D^{sm}.

In this section, we introduce another performance measure $d(\mathscr{S})$ for the sake of analytical simplicity. For a distributed learning system $\mathscr{S}$, for any D^{sm}, we define $d(\mathscr{S})$ by

$$d(\mathscr{S}) \stackrel{\text{def}}{=} \int |m_{\mathscr{S}}(\mathbf{D} \mid D^{sm}) - m^*(\mathbf{D} \mid D^{sm})| \, d\mathbf{D}, \tag{37}$$

where the notations of $m_{\mathscr{S}}(\mathbf{D} \mid D^{sm})$ and $m^*(\mathbf{D} \mid D^{sm})$ follow (34) and (36), respectively. Note that $d(\mathscr{S})$ depends on the training sequence D^{sm}. Here $d(\mathscr{S})$ is the *variation distance* between m^* and $m_{\mathscr{S}}$, which measures how large the average performance of $\mathscr{S}$ for predicting the probability density of an unseen example is deviated from that of NDB. Recall now from Lemma 2 that the additional loss is given by $E_{D^{sm}}[D(m^* \| m_{\mathscr{S}})]$, which is an expected Kullback–Leibler divergence between m^* and $m_{\mathscr{S}}$. Thus $d(\mathscr{S})$ can be thought of as an analogue of the additional loss $L(\mathscr{S}) - L^*$. We wish to design a distributed learning system $\mathscr{S}$ such that its additional loss or $d(\mathscr{S})$ is as small as possible, while attaining a significant speed-up of learning over NDB.

3.2. Algorithms

We introduce the distributed cooperative Bayesian learning strategy in the hierarchical model as follows:

DEFINITION 19. Let a positive integer N and an initial hyperparameter value $\hat{\mu}^{(0)}$ be given. We define the *distributed cooperative Bayesian strategy* (in the hierarchical model), which we abbreviate as *HDCB*, as the distributed learning system that makes the following iteration process N times: At the lth iteration ($l = 1, 2, ..., N$),

(1) Each agent employs the *Gibbs algorithm*, i.e., the ith agent takes as input $D_i^m = D_{i1} \cdots D_{im}$, $\mathscr{C}$, $\pi(\theta \mid \mu)$, and the latest estimate $\hat{\mu}^{(l-1)}$ of μ, then outputs the parameter value $\hat{\theta}_i^{(l)}$ chosen randomly according to the Bayes posterior density $p(\theta_i \mid D_i^m, \hat{\mu}^{(l-1)})$, which is calculated as

$$p(\theta_i \mid D_i^m, \hat{\mu}^{(l-1)}) = \frac{\pi(\theta_i \mid \hat{\mu}^{(l-1)}) \prod_{j=1}^m p(D_{ij} \mid \theta_i)}{\int \pi(\theta'_i \mid \hat{\mu}^{(l-1)}) \prod_{j=1}^m p(D_{ij} \mid \theta'_i) \, d\theta'_i} \qquad (i = 1, ..., s).$$

(2) The p-learner also employs the *Gibbs algorithm*; i.e., it takes as input the latest estimates $\hat{\theta}_1^{(l)}, ..., \hat{\theta}_s^{(l)}$, $\pi(\theta \mid \mu)$, and $\pi(\mu)$, then outputs the parameter value $\hat{\mu}^{(l)}$ chosen randomly according to the Bayes posterior density $p(\mu \mid \hat{\theta}_1^{(l)}, ..., \hat{\theta}_s^{(l)})$, which is calculated as

$$p(\mu \mid \hat{\theta}_1^{(l)}, ..., \hat{\theta}_s^{(l)}) = \frac{\pi(\mu) \prod_{i=1}^s \pi(\hat{\theta}_i^{(l)} \mid \mu)}{\int \pi(\mu') \prod_{i=1}^s \pi(\hat{\theta}_i^{(l)} \mid \mu') \, d\mu'}.$$

The output of the system is $\hat{\omega} = (\hat{\theta}_1^{(N)}, ..., \hat{\theta}_s^{(N)}, \hat{\mu}^{(N)})$.

Remarks. (1) Let the outputs of the agents and the p-learner at the lth iteration be $\hat{\theta}_i^{(l)}$ $(i=1, ..., s)$ and $\hat{\mu}^{(l)}$, respectively. We set $\hat{\omega}^{(l)}=(\hat{\theta}_1^{(l)}, ..., \hat{\theta}_s^{(l)}, \hat{\mu}^{(l)})$ $(l=1, ..., N)$. Then the process

$$\hat{\omega}^{(1)} \to \hat{\omega}^{(2)} \to \cdots \to \hat{\omega}^{(N)} \tag{38}$$

forms a Markov chain. That is, letting $K(\omega', \omega)$ be the transition probability density from ω' to ω $(\omega=(\theta_1, ..., \theta_s, \mu), \omega'=(\theta'_1, ..., \theta'_s, \mu') \in \Omega)$ and $p^*(\omega \mid D^{sm})$ be the Bayes posterior density as in (35), it is easily checked that $K(\omega', \omega)=(\prod_{i=1}^s p(\theta_i \mid D_i^m, \mu))\, p(\mu \mid \theta'_1, ..., \theta'_s)$ and that

$$p^*(\omega \mid D^{sm}) = \int K(\omega', \omega)\, p^*(\omega' \mid D^{sm})\, d\omega',$$

which implies that $p^*(\omega \mid D^{sm})$ is a stationary distribution for this Markov chain. The iteration process (38) is known as a *Markov chain Monte Carlo* process (see, [2–4, 10, 12]) and is algorithmically equivalent to the *Gibbs sampler* [3].

(2) As will be seen in Example 21, there exist some cases where the computation times required by the agent and the p-learner to run the Gibbs algorithm are linear in the sample size m and the population size s, respectively. For such cases, since each agent in HDCB employs the Gibbs algorithm for m examples N times and the p-learner employs it for s examples N times, the computation time for HDCB is $O(N(s+m))$, while that for NDB is $\Omega(sm)$. For the total sample size $\ell=sm$, the computational time for HDCB is $O(N\sqrt{\ell})$ if we set $s=\Theta(\sqrt{\ell})$ and $m=\Theta(\sqrt{\ell})$ so that $(N(s+\ell/s))$ is made smallest. Then we say that *HDCB achieves a speed-up in computation time from $\Omega(\ell)$ for NDB to $O(N\sqrt{\ell})$.*

(3) HDCB may have another computational merit in comparison with NDB. As will be seen in Examples 20 and 21, there exist some cases where the joint Bayes posterior density $p(\omega \mid D^{sm})=p(\theta_1, ..., \theta_s, \mu \mid D^{sm})$ for NDB is analytically hard to calculate, while the conditional densities $p(\theta_1, ..., \theta_s \mid D^m, \mu)$ and $p(\mu \mid \theta_1, ..., \theta_s)$ can be straightforwardly calculated.

(4) HDCB can be thought of as a hierarchical variant of DCB1. We may also consider a hierarchical variant of DCB2 in which each agent outputs a number of statistics for the target densities and the p-learner chooses $\hat{\omega}$ randomly according to the joint Bayesian posterior density. However, the computation argument in remark (3) implies that such a hierarchical variant of DCB2 might be inefficient in computing the joint Bayes posterior density, in general, for the hierarchical probability model. (This holds even for the simple case as in Example 21.) Hence this section focuses on the hierarchical variant of DCB1.

EXAMPLE 20. Let $\mathscr{Y}=\{0, 1\}$ and let $\mathscr{C}=\{p(1 \mid \theta)=\theta,\ p(0 \mid \theta)=1-\theta : \theta \in [c, 1-c]\}$ be a class of Bernoulli distributions for which the parameter value is restricted to $[c, 1-c]$ for a given $c \in (0, 1/2)$. Let the prior density of θ_i specified by a hyperparameter μ be $\pi(\theta_i \mid \mu)=D_c(\mu, n-\mu : \theta_i)$ $(i=1, ..., s)$, where n is a given positive real number, and $D_c(\alpha, \beta : \theta) \stackrel{\text{def}}{=} \theta^\alpha(1-\theta)^\beta / \int_c^{1-c} \theta'^\alpha(1-\theta')^\beta\, d\theta'$, where

$\alpha > 0$ and $\beta > 0$ are given. Let the prior distribution of μ be the uniform distribution over $[0, n]$. For each i, let m_i be the number of examples such that $D_{ij} = 1$ in the sequence $D_i^m = D_{i1} \cdots D_{im}$. Then, noting that joint density of $\theta_1, \ldots \theta_s, \mu$, and D^{sm} is proportional to $\prod_{i=1}^{s} \theta_i^{m_i+\mu}(1-\theta_i)^{m+n-m_i-\mu}$, we have

$$p(\theta_i \mid D_i^m, \mu) = D_c(m_i + \mu, m + n - m_i - \mu : \theta_i) \quad (i = 1, \ldots, s),$$

$$p(\mu \mid \theta_1, \ldots, \theta_s) = \left(\prod_{i=1}^{s} \theta_i \right)^{\mu} \left(\prod_{i=1}^{s} (1-\theta_i) \right)^{n-\mu} \ln \left(\prod_{i=1}^{s} \frac{\theta_i}{1-\theta_i} \right) \times \left(\left(\prod_{i=1}^{s} \theta_i \right)^{n} - \left(\prod_{i=1}^{s} (1-\theta_i) \right)^{n} \right)^{-1}.$$

We can implement the iteration process in HDCB using these conditional densities.

EXAMPLE 21. Let $\mathscr{D} = \mathbf{R}$ and let $\mathscr{C} = \{ N(\theta, \sigma^2 : D) : \theta \in (-\infty, \infty),\ \sigma^2 \text{ is known} \}$ be a class of Gaussian distributions with a constant variance. Let the prior density of θ_i specified by a hyperparameter μ be $\pi(\theta_i \mid \mu) = N(\mu, \sigma_\theta^2 : \theta_i)$ $(i = 1, \ldots, s)$, where $0 < \sigma_\theta^2 < \infty$ is given. Let the prior density of μ be $\pi(\mu) = N(\mu_0, \sigma_0^2 : \mu)$ where μ_0 and σ_0^2 are given. Then, noting that joint density of $\theta_1, \ldots \theta_s, \mu$, and D^{sm} is proportional to $\exp(-(\mu - \mu_0)^2/2\sigma_0^2) \prod_{i=1}^{s} \exp(-(\theta_i - \mu)^2/2\sigma_\theta^2) \prod_{j=1}^{m} \exp(-(D_{ij} - \theta_i)^2/2\sigma^2)$, we have

$$p(\theta_i \mid D_i^m, \mu) = N\left(\frac{\sigma_\theta^2 \sum_{j=1}^{m} D_{ij} + \sigma^2 \mu}{m\sigma_\theta^2 + \sigma^2}, \frac{\sigma_\theta^2 \sigma^2}{m\sigma_\theta^2 + \sigma^2} \right) \qquad (i = 1, \ldots, s), \tag{39}$$

$$p(\mu \mid \theta_1, \ldots, \theta_s) = N\left(\frac{\sigma_\theta^2 \mu_0 + \sigma_0^2 \sum_{i=1}^{s} \theta_i}{\sigma_\theta^2 + s\sigma_0^2}, \frac{\sigma_\theta^2 \sigma_0^2}{\sigma_\theta^2 + s\sigma_0^2} \right). \tag{40}$$

We can implement the iteration process in HDCB using these conditional densities.

Notice here that the running time required by the ith agent to compute $\hat{\theta}_i$ with density (39) is linear in the sample size m and that the running time for the p-learner to compute $\hat{\mu}$ with density (40) is linear in the population size s, since the computation time required for a random sampling according to a fixed Gaussian distribution is $O(1)$ in s and m, supposing that a program for a random number generation according to a Gaussian distribution is given to the p-learner and agents.

3.3. *Analysis*

First we consider the case where the parameter space is bounded. We start with the following assumption and notation:

ASSUMPTION 22. *The hypothesis class* $\mathscr{C} = \{ p(D \mid \theta) : \theta \in \Theta \}$ *satisfies the condition*: *There exist some positive numbers* $\underline{c}$ *and* $\bar{c}$ *such that for all* D, *for all* θ, $0 < \underline{c} \leqslant p(D \mid \theta) \leqslant \bar{c} < \infty$.

Note that Assumption 22 is satisfied for Example 20 but is not for Example 21.

Below $K(\cdot, \cdot)$ denotes a Markov chain transition kernel on a state space Ω; i.e., for $\omega, \omega' \in \Omega$, $K(\omega, \omega')$ denotes the transition probability density from ω to ω'. For $\omega \in \Omega$, $S \subset \Omega$, we define $K(\omega, S)$ by $K(\omega, S) = \int_{\omega' \in S} K(\omega, \omega')\, d\omega'$. For any positive integer l, we define $K^{(l)}(\omega, S)$ inductively by

$$K^{(l)}(\omega, S) = \int K^{(l-1)}(\omega, \omega')\, K(\omega', S)\, d\omega' \qquad (l = 2, 3, \ldots),$$

$$K^{(1)}(\omega, S) = K(\omega, S).$$

Theorem 23 gives a general upper bound on the average logarithmic loss for HDCB in the case where the parameter space Ω is bounded.

THEOREM 23. *Suppose that the parameter space $\Omega = \{\omega = (\theta_1, \ldots, \theta_s, \mu)\}$ is bounded. Also suppose that the prior density π over Ω is everywhere positive and continuous. Then, for some $0 < C < \infty$, for any positive integer l, for each training sequence D^{sm}, we have the following upper bound on d*(HDCB):

$$d(\mathrm{HDCB}) \leqslant C\rho_l^{\lfloor N/l \rfloor}. \tag{41}$$

Here $\lfloor x \rfloor$ denotes the largest integer that does not exceed x, and $0 < \rho_l < 1$ is a function of D^{sm}, defined as

$$\rho_l = 1 - \int \inf_{\theta'_1, \ldots, \theta'_s} K^{(l)}(\omega', \omega)\, d\mu\, d\theta_1 \cdots d\theta_s, \tag{42}$$

where $K(\omega', \omega) = (\prod_{i=1}^{s} p(\theta_i \mid D_i^m, \mu)) \cdot p(\mu \mid \theta'_1, \ldots, \theta'_s)$. Under Assumption 22 for $\mathscr{C}$ in addition to the above conditions, for some $0 < C' < \infty$, $1 < \kappa < \infty$, for any positive integer l, we have the following upper bound on the additional loss for HDCB,

$$L(\mathrm{HDCB}) - L^* \leqslant C'\kappa^s E_{D^{sm}}[\rho_l^{\lfloor N/l \rfloor}], \tag{43}$$

where ρ_l is as defined in (42).

Remarks. (1) Theorem 23 shows that the additional loss for HDCB converges to zero exponentially in the iteration number N. This implies that unlike DCB1s in Examples 4 and 5, HDCB can make its additional loss arbitrarily small by increasing the number N of feedback iterations. Theorem 23 also characterizes the rate of convergence of the additional loss in terms of the quantity ρ_l, which may depend on $s, m, l, \mathscr{C}, \pi(\theta \mid \mu)$, and $\pi(\mu)$.

(2) If the Markov chain that HDCB induces is such that there is a constant $0 < \rho_l < 1$ as in (42) independent of s and m for smaller l, then it is *good* in the sense that both d(HDCB) and $L(\mathrm{HDCB}) - L^*$ become small more quickly as N increases.

(3) Suppose that the computational times for the agent and the p-learner are linear in the sample size and the population size, respectively. Then if there exist

some positive integer l_0 and some $0<\rho<1$ such that $\sup_{l\geqslant l_0}\sup_{D^{sm}}\rho_l\leqslant\rho$, then for any $\varepsilon>0$, by setting $s=\Theta(1/\sqrt{\varepsilon})$, $m=\Theta(1/\sqrt{\varepsilon})$, and $N=\lceil l_0(\ln(1/\varepsilon))/(\ln(1/\rho))\rceil$, both the additional loss for HDCB and the distance $d(\text{HDCB})$ can be made at most ε, while HDCB achieves a speed-up in computation time from $\Omega(\ell)$ for NDB to $O(l_0((1/\sqrt{\varepsilon})\ln(1/\varepsilon))/(\ln(1/\rho)))$. Here $\lceil x\rceil$ denotes the smallest integer that is not smaller than x. This fact implies that for the total sample size $\ell=sm$, setting $s=\Theta(\sqrt{\ell})$, $m=\Theta(\sqrt{\ell})$, and $N=\Theta(l_0(\ln\ell)/(\ln(1/\rho)))$, the additional loss for HDCB can be made $O(1/\ell)$ while HDCB achieves a speed-up in computation time from $\Omega(\ell)$ for NDB to $O(l_0(\sqrt{\ell}\ln\ell)/(\ln(1/\rho)))$.

We prepare Lemma 24 and Proposition 25 in order to prove Theorem23. Lemma 24 relates the additional loss and $d(\mathscr{S})$ to the rate of convergence of the Markov chain Monte Carlo process that $\mathscr{S}$ induces.

LEMMA 24. *For any distributed learning system $\mathscr{S}$ in the hierarchical model, let q be the probability density over Ω according to which the output of $\mathscr{S}$ is generated, and let p^* be the Bayes posterior density* (35) *over Ω. Then for each D^{sm}, we have*

$$d(\mathscr{S})\leqslant d(q,\,p^*), \tag{44}$$

where $d(q,\,p^*)\stackrel{\text{def}}{=}\int|q(\omega\,|\,D^{sm})-p^*(\omega\,|\,D^{sm})|\,d\omega$. *In addition, under Assumption* 22 *for $\mathscr{C}$, for any distributed learning system $\mathscr{S}$ in the hierarchical model, for some $1<\kappa<\infty$, we have*

$$L(\mathscr{S})-L^*\leqslant\kappa^s E_{D^{sm}}[\,d(q,\,p^*)\,]. \tag{45}$$

(The proof of Lemma 24 is in the Appendix.)

Equations (44) and (45) show that the distance between the distributions of the parameter ω upper-bounds the distances of the induced distributions of **D**.

Recall that the iteration process in HDCB induces a Markov chain with a stationary density $p^*(\omega\,|\,D^{sm})$. Proposition 25 gives an upper bound on the rate of convergence for the chain.

PROPOSITION 25 [9]. *Let K be a Markov chain transition kernel on a bounded state space Ω and let π^* be the stationary density of the Markov chain. Suppose that there are some probability measure Q over Ω, some positive integer l, and some $\varepsilon>0$ such that for all $\omega\in\Omega$, for all $S\subset\Omega$,*

$$K^{(l)}(\omega,\,S)\geqslant\varepsilon Q(S). \tag{46}$$

Then for any initial probability density π_0, the probability density π_N of the Markov chain after N steps satisfies

$$d(\pi_N,\,\pi^*)\leqslant 2(1-\varepsilon)^{\lfloor N/l\rfloor}. \tag{47}$$

Proof of Theorem 23. The Markov chain kernel induced by the iteration process in HDCB is written as

$$K(\omega', \omega) = p(\mu \mid \theta'_1, ..., \theta'_s) \prod_{i=1}^{s} p(\theta_i \mid D_i^m, \mu),$$

where $\omega = (\theta_1, ..., \theta_s, \mu)$ and $\omega' = (\theta'_1, ..., \theta'_s, \mu')$. Observe that for any positive integer l, for all $\omega, \omega'' \in \Omega$, we have

$$K^{(l)}(\omega'', \omega) \geqslant \varepsilon \cdot \frac{\inf_{\theta'_1 \cdots \theta'_s} K^{(l)}(\omega', \omega)}{\varepsilon},$$

where $\varepsilon \stackrel{\text{def}}{=} \int \inf_{\theta'_1 \cdots \theta'_s} K^{(l)}(\omega', \omega)\, d\omega$. Notice here that $K^{(l)}(\omega', \omega)$ depends on $\theta'_1, ..., \theta'_s$ but not on μ'. Hence, setting $Q(S) \stackrel{\text{def}}{=} \int_S (\inf_{\theta'_1 \cdots \theta'_s} K^{(l)}(\omega', \omega)/\varepsilon)\, d\omega$ for any $S \subset \Omega$, we see that Q forms a probability measure over Ω, and thus (46) is satisfied. By Proposition 25, setting $\rho_l = 1 - \varepsilon$ in (47) gives

$$d(q, p^*) \leqslant 2\rho_l^{\lfloor N/l \rfloor}. \tag{48}$$

Combining (48) with (44) and (45) in Lemma 24 yields (41) and (43), respectively. This completes the proof of Theorem 23. ▮

Theorem 23 can be applied to Example 20, but cannot be applied to the case where the parameter space is unbounded or the density value of each example may not be uniformly lower-bounded away from zero, as in Example 21. Below we give another type of analysis for HDCB with respect to the measure $d(\mathscr{S})$ as in (37).

THEOREM 26. *For HDCB for the class of hierarchical Gaussian distributions as in Example* 21, *for each training sequence* D^{sm}, *for some* $0 < c_1, c_2 < \infty$ *independent of* s, m, *and* N, *we have*

$$d(\text{HDCB}) \leqslant O(e^{-c_1 N(\ln m)/\ln(sN)}) + O\left(\frac{N \ln m}{\ln(sN)}\, e^{-c_2 N}\right). \tag{49}$$

COROLLARY 27. *Let* $c = \max\{c_1/2, c_2\}$. *For sufficiently large total sample size* $\ell = sm$, *setting* $s = \Theta(\sqrt{\ell})$ *and* $N = \lceil (\ln \ell)/c \rceil = \Theta(\ln \ell)$, *the additional loss for HDCB in Example* 21 *can be made* $O((\ln \ell)/\ell)$ *while HDCB achieves a speed-up in computation time from* $\Omega(\ell)$ *for NDB to* $O(\sqrt{\ell} \ln \ell)$.

Remarks. (1) By Theorem 26, $d(\text{HDCB})$ can be made arbitrarily small by increasing N even when m is fixed, while the additional losses for DCB1s in Examples 4 and 5 cannot be made arbitrarily small for fixed m (see Theorems 11 and 14). This implies that the feedback of information from the p-learner to agents is so effective that HDCB works approximately as well as NDB as N becomes sufficiently large even for fixed m.

(2) Let $c = \max\{c_1/2, c_2\}$. For any $\varepsilon > 0$, if we set $N = \lceil (1/c) \ln(1/\varepsilon) \rceil = \Theta(\ln(1/\varepsilon))$, $m = \Theta(1/\sqrt{\varepsilon})$, and $s = \Theta(1/\sqrt{\varepsilon})$, then $d(\mathrm{HDCB})$ can be made at most $\varepsilon \ln(1/\varepsilon)$ while HDCB achieves a speed-up in computation time from $\Omega(\ell)$ for NDB to $O(\ln(1/\varepsilon)/\sqrt{\varepsilon})$.

The outline of the proof of Theorem 26 basically follows Rosenthal's proof [10] of the rate of convergence for the *variance component model*. Notice, however, that the model we deal with can be thought of as a simplified variant of the variance composed model and thus our analysis is somewhat specific. This makes our bound tighter than Rosenthal's.

In order to prove Theorem 26, we give Proposition 28 and Lemmas 29 and 30. Note that for the case where the parameter space is bounded, we can apply Proposition 25 to obtain bounds (41) and (43). For the case where the parameter space is not bounded, however, it is not always possible to require that (46) hold for an arbitrary initial value $\omega \in \Omega$. Proposition 28, which was first proven by Rosenthal, enables us to perform an analysis similar to that of Proposition 25 by requiring that an equation similar to (46) hold for an arbitrary initial value in some *subset* of Ω rather than in the whole state space Ω.

PROPOSITION 28 [10]. *Let K be a Markov chain transition kernel on a state space Ω and let π^* be the stationary density of the Markov chain. Suppose that there are measurable subsets $R_1, R_2 \subset \Theta$, some probability measure Q over Ω, some positive integer l_0, and some $\varepsilon_1, \varepsilon_2 > 0$ such that for all $\omega \in R_1$,*

$$K^{(l_0)}(\omega, R_2) \geqslant \varepsilon_1, \tag{50}$$

and for all $\omega \in R_2$, for all $S \subset \Omega$,

$$K(\omega, S) \geqslant \varepsilon_2 Q(S). \tag{51}$$

Then for any initial probability density π_0 supported entirely in R_1, the probability density π_N of the Markov chain after N steps satisfies

$$d(\pi_N, \pi^*)/2 \leqslant (1 - \varepsilon_1 \varepsilon_2)^{\lfloor N/(l_0+1) \rfloor} + A + 2 \lfloor N/(l_0+1) \rfloor B, \tag{52}$$

where

$$A = 1 - \pi^*(R_1),$$

$$B = 1 - \inf_{\omega \in R_1} K^{(l_0+1)}(\omega, R_1).$$

In the right-hand side of (52), the first term $(1 - \varepsilon_1 \varepsilon_2)^{\lfloor N/(l_0+1) \rfloor}$ is derived by applying the technique in Proposition 25 to the subset R_2 in Ω, under an assumption that the Markov chain will jump from R_1 into R_2 with probability at least ε_1. The second term A will become small if the stationary distribution of the Markov

chain has a large probability mass on R_1. The third term $2\lfloor N/(l_0+1)\rfloor B$ will become small if the Markov chain does not easily jump outside from R_1. Lemma 29 gives a concrete method for constructing R_1, R_2, Q, l_0 to satisfy the conditions in Proposition 28 for the model in Example 21.

LEMMA 29. *Let* $\bar{D} = (1/sm)\sum_{i=1}^{s}\sum_{j=1}^{m} D_{ij}$. *For the model in Example* 21, *define* $R_1, R_2 \subset \Omega$ *by*

$$R_1 \stackrel{\text{def}}{=} \left\{\omega \in \Omega: \left|\bar{D} - \frac{1}{s}\sum_{i=1}^{s}\theta_i\right| \leqslant N^{1/2}\right\}, \tag{53}$$

$$R_2 \stackrel{\text{def}}{=} \left\{\omega \in \Omega: \left|\bar{D} - \frac{1}{s}\sum_{i=1}^{s}\theta_i\right| \leqslant \frac{2}{\sqrt{s}}\right\}. \tag{54}$$

Also, let Q *be the probability measure which first chooses* μ *randomly according to the uniform distribution over the set*

$$I = [\bar{D} - 2/\sqrt{s}, \bar{D} + 2/\sqrt{s}],$$

then chooses θ_i *independently randomly according to*

$$N\left(\frac{\sigma_\theta^2\sum_{j=1}^{m} D_{ij} + \sigma^2\mu}{m\sigma_\theta^2 + \sigma^2}, \frac{\sigma_\theta^2\sigma^2}{m\sigma_\theta^2 + \sigma^2}\right) \qquad (i = 1, \ldots, s).$$

Then for sufficiently large s, *for the Markov chain that HDCB in Example* 21 *induces, all the conditions* (50) *and* (51) *in Proposition* 28 *are satisfied by* R_1, R_2, *and* Q *as above, some positive integers* $\varepsilon_1, \varepsilon_2$ *independent of* s, m, N, *and* $l_0 = O((\ln(sN))/(\ln m))$.

(The proof of Lemma 29 is in the Appendix.)

Lemma 30 shows concrete evaluation of A and B as in Proposition 28, which can be directly proven by considering the tail of the stationary distribution.

LEMMA 30. *For* $R_1, R_2, Q, \varepsilon_1, \varepsilon_2, l_0$ *as in Lemma* 29, A *and* B *as in* (52) *in Proposition* 28 *are both upper-bounded by expressions of the form* $d_1 e^{-d_2 N}$, *with* $d_1, d_2 > 0$ *independent of* s, m, *and* N.

Proof of Theorem 26. From Proposition 28 and Lemmas 29 and 30, we see that for HDCB in Example 21 with N feedback iterations, for each training sequence D^{sm}, for some $0 < c_1, c_2 < \infty$ independent of s, m, and N, the following inequality holds:

$$d(q, p^*) \leqslant O(e^{-c_1 N(\ln m)/\ln(sN)}) + O\left(\frac{N\ln m}{\ln(sN)} e^{-c_2 N}\right). \tag{55}$$

Plugging (55) into (44) in Lemma 24 yields (49). This completes the proof of Theorem 26. ∎

4. EXTENSIONS OF DCB

4.1. Multihierarchical Model

First we consider an extension of HDCB into the hierarchical model with multi-hyperlevels. Recall that the parameters $\theta = (\theta_1 \cdots \theta_s)$ and μ form a hierarchical structure. We denote this structure by

$$\theta \mid \mu.$$

We can consider a general version of this structure, having k stages as follows:

$$\xi_1 \mid \xi_2 \mid \cdots \mid \xi_k.$$

Then the hierarchical structure implies the following probability relations among parameters:

$$p(\xi_i \mid D^{sm}, \xi_j, (j \neq i)) = \begin{cases} p(\xi_1 \mid D^{sm}, \xi_2), \\ p(\xi_i \mid \xi_{j-1}, \xi_{j+1}) & (i = 2, ..., k-1). \\ p(\xi_k \mid \xi_{k-1}). \end{cases}$$

We may obtain a variant of HDCB which uses these conditional distributions hierarchically to form an iteration process called the *Gibbs sampler* [3]. At each iteration step the parameter values are sampled in the order: $\xi_1 \to \xi_2 \to \cdots \to \xi_k$. After a given number of iterations the variant of HDCB outputs an estimate of $(\xi_1, ..., \xi_k)$.

4.2. General Decision-Theoretic Model

Next we consider an extension of DCB1 to a general decision-theoretic scenario. In Sections 2 and 3, we have developed a theory in the case where the hypothesis class is a class of probability density functions and the prediction loss is measured in terms of the logarithmic loss. We can extend our theory into the general case where the hypothesis class is a class of parametric real-valued functions of the form of $\mathscr{C} = \{f_\theta(x): \theta \in \Theta \subset \mathbf{R}^k\}$ and the prediction loss is measured in terms of a general loss function, which we write as $L(y, f_\theta(x))$ for $D = (x, y)$. For example, the square loss is written as $L(y, f_\theta(x)) = (y - f_\theta(x))^2$.

Below let us focus on the plain model. Let the range of the loss function L be $[0, 1]$. We assume that the parameter θ is generated according to the prior density $\pi(\theta)$. For a given training sequence $D^{sm} = D_1^m \cdots D_s^m$ with $D_i^m = D_{i1} \cdots D_{im}$, $D_{ij} = (x_{ij}, y_{ij})$ $(i = 1, ..., s, j = 1, ..., m)$, let us define the probability density $p(\theta \mid D_i^m)$ by

$$p(\theta \mid D_i^m) = \frac{\pi(\theta)\, e^{-\lambda \sum_{j=1}^m L(y_{ij}, f_\theta(x_{ij}))}}{\int \pi(\theta')\, e^{-\lambda \sum_{j=1}^m L(y_{ij}, f_{\theta'}(x_{ij}))}\, d\theta'} \qquad (i = 1, ..., s), \tag{56}$$

where λ is a positive real number depending on L. Here we obtain a variant of DCB1 in which each agent outputs $\hat{\theta}_i$ chosen randomly according to $p(\theta \mid D_i^m)$, and the p-learner calculates $\hat{\theta} = f(\hat{\theta}_1, ..., \hat{\theta}_s)$ using some appropriate deterministic function f.

In a decision-theoretic scenario, we might be concerned with *predicting* unseen data y for given input x rather than estimating the parameter θ. Below we describe how to predict future data using the version of DCB1.

For a given positive integer h, for each i ($=1, ..., s$), let each agent make random sampling of θ according to (56) h times and let $\hat{\theta}_i^{(t)}$ ($t=1, ..., h$) be the outputs. Let $\hat{\theta}^{(t)}$ be the output of the p-learner corresponding to $\hat{\theta}_1^{(t)}, ..., \hat{\theta}_s^{(t)}$ ($t=1, ..., h$), i.e., $\hat{\theta}^{(t)} = f(\hat{\theta}_1^{(t)}, ..., \hat{\theta}_s^{(t)})$. For $D=(x, y)$, define $\Delta(y)$ by

$$\Delta(y) = -\frac{1}{\lambda} \ln \left(\frac{1}{h} \sum_{t=1}^{h} e^{-\lambda L(y, f_{\hat{\theta}^{(t)}}(x))} \right). \tag{57}$$

On receiving x, DCB1 predicts y with $\hat{y}$ such that

$$L(0, \hat{y}) \leqslant \Delta(0) \qquad \text{and} \qquad L(1, \hat{y}) \leqslant \Delta(1). \tag{58}$$

It is known (see, e.g., [5, 13]) that under some smoothness conditions for a loss function, there exists a range of λ such that $\hat{y}$ as in (58) exists and the prediction loss is upper-bounded by $\Delta(y)$ uniformly with respect to a correct value $y \in [0, 1]$. For example, $0 < \lambda \leqslant 2$ for the square loss (see [5, 13]).

Let $q(\theta \mid D^{sm})$ denote the probability density according to which a final output $\hat{\theta}$ is generated. Define $\bar{\Delta}(y)$ by

$$\bar{\Delta}(y) \stackrel{\text{def}}{=} -\frac{1}{\lambda} \ln \left(\int q(\theta \mid D^{sm})\, e^{-\lambda L(y, f_\theta(x))}\, d\theta \right) \qquad (y=0, 1), \tag{59}$$

then $\Delta(y)$ can be thought of as a Monte Carlo approximation of $\bar{\Delta}(y)$. Hence the prediction loss for DCB1 is also upper-bounded by $\bar{\Delta}(y)$ with high probability, when h is sufficiently large.

When $q(\theta \mid D^{sm})$ equals

$$p^*(\theta \mid D^{sm}) = \frac{\pi(\theta) \prod_{i=1}^{s} \prod_{j=1}^{m} e^{-\lambda L(y_{ij}, f_\theta(x_{ij}))}}{\int \pi(\theta') \prod_{i=1}^{s} \prod_{j=1}^{m} e^{-\lambda L(y_{ij}, f_{\theta'}(x_{ij}))}\, d\theta'},$$

the prediction strategy based on (59) becomes equivalent to Vovk's *aggregating strategy* [13, 5], and the prediction loss (59) is related to the notion of *extended stochastic complexity* (*ESC*) [14]. We may expect that the prediction strategy based on (59) works approximately as well as the aggregating strategy. It remains for future study to derive any concrete bounds on the additional loss for the decision-theoretic version of DCB1. A decision-theoretic extension of DCB2 has also been considered. See [17] for the details.

5. CONCLUDING REMARKS

We have developed the plain model and the hierarchical model of distributed learning, as probabilistic versions of Kearns and Seung's model of population learning. Within these models we have proposed three types of DCBs (distributed cooperative Bayesian learning strategies)—DCB1, DCB2, and HDCB, and analyzed their performance in terms of their additional losses (and the variation distance) as functions of the sample size, the population size, and the iteration number. For some concrete hypothesis classes we have demonstrated that DCBs work approximately (or sometimes exactly) as well as the nondistributed Bayesian learning strategy, achieving a significant speed-up of learning over it.

The following issues remain for future study:

(1) *Any other effective strategy of the p-learner in DCB*1? In the plain model we have analyzed only the cases where the p-learner in DCB1 employs the arithmetic mean operation (13). It is an interesting question when any other form of (3) gives a more effective strategy.

(2) *Relating the analysis of HDCB to rapidly mixing Markov chains.* As seen in Section 3.3, the additional loss for HDCB is related to the rate of convergence of the Markov chain Monte Carlo process that HDCB induces. On the other hand, in theoretical computer science, the theory of *rapidly mixing Markov chains* [11] has been developed in order to investigate the rate of convergence of general Markov chains. It would be interesting to explore another type of analysis of HDCB on the basis of the theory of rapidly mixing Markov chains.

(3) *Evaluating HDCB for Example* 21 *in terms of any other losses.* In Theorem 26 we have analyzed HDCB for Example 21 with respect to the variation distance for fixed D^{sm}. It remains for future study to analyze it with respect to the variation distance averaged over all D^{sm} or the additional logarithmic loss.

APPENDIX

Proof of Lemma 24. Equation (44) can be straightforwardly proven as follows: For any D^{sm},

$$
\begin{aligned}
d(\mathscr{S}) &= \int |m_{\mathscr{S}}(\mathbf{D} \mid D^{sm}) - m^*(\mathbf{D} \mid D^{sm})| \, d\mathbf{D} \\
&= \int \left| \int p(\mathbf{D} \mid \omega)(q(\omega \mid D^{sm}) - p^*(\omega \mid D^{sm})) \, d\omega \right| d\mathbf{D} \\
&\leqslant \int \left(\int p(\mathbf{D} \mid \omega) \, |q(\omega \mid D^{sm}) - p^*(\omega \mid D^{sm})| \, d\omega \right) d\mathbf{D} \\
&= \int |q(\omega \mid D^{sm}) - p^*(\omega \mid D^{sm})| \, d\omega \int p(\mathbf{D} \mid \omega) \, d\mathbf{D} \\
&= d(q, p^*),
\end{aligned}
$$

where $d(q, p^*) = \int |q(\omega \mid D^{sm}) - p^*(\omega \mid D^{sm})| \, d\omega$.

Next, in order to prove (45), we start by observing that the logarithmic loss induced for any distributed learning system in the hierarchical model is written as follows:

$$-\ln \int p(\mathbf{D}\,|\,\omega)\, q(\omega\,|\,D^{sm})\, d\omega = -\ln \int p(\mathbf{D}\,|\,\omega)\, p^*(\omega\,|\,D^{sm})\, d\omega + \ln \frac{\int p(\mathbf{D}\,|\,\omega)\, p^*(\omega\,|\,D^{sm})\, d\omega}{\int p(\mathbf{D}\,|\,\omega)\, q(\omega\,|\,D^{sm})\, d\omega}. \tag{60}$$

We can further upper-bound $\ln((\int p(\mathbf{D}\,|\,\omega)\, p^*(\omega\,|\,D^{sm})\, d\omega)/(\int p(\mathbf{D}\,|\,\omega)\, q(\omega\,|\,D^{sm})\, d\omega))$ as

$$\ln \frac{\int p(\mathbf{D}\,|\,\omega)\, p^*(\omega\,|\,D^{sm})\, d\omega}{\int p(\mathbf{D}\,|\,\omega)\, q(\omega\,|\,D^{sm})\, d\omega} \leqslant \frac{\int p(\mathbf{D}\,|\,\omega)\, p^*(\omega\,|\,D^{sm})\, d\omega}{\int p(\mathbf{D}\,|\,\omega)\, q(\omega\,|\,D^{sm})\, d\omega} - 1 \tag{61}$$

$$= \frac{\int p(\mathbf{D}\,|\,\omega)(p^*(\omega\,|\,D^{sm}) - q(\omega\,|\,D^{sm}))\, d\omega}{\int p(\mathbf{D}\,|\,\omega)\, q(\omega\,|\,D^{sm})\, d\omega}$$

$$\leqslant \frac{(\sup_{\mathbf{D},\,\omega} p(\mathbf{D}\,|\,\omega)) \int |q(\omega\,|\,D^{sm}) - p^*(\omega\,|\,D^{sm})|\, d\omega}{\inf_{\mathbf{D},\,\omega} p(\mathbf{D}\,|\,\omega)}$$

$$= (\bar{c}/\underline{c})^s\, d(q,\, p^*) \tag{62}$$

$$= \kappa^s d(q,\, p^*), \tag{63}$$

where we set $\kappa = (\bar{c}/\underline{c})$. We have used the fact that $\ln x \leqslant x - 1$ for any $x > 0$ to derive (61) and have used the facts that $\sup_{\mathbf{D},\,\omega} p(\mathbf{D}\,|\,\omega) = (\sup_{D,\,\theta} p(D\,|\,\theta))^* = \bar{c}^s$ and that $\inf_{\mathbf{D},\,\omega} p(\mathbf{D}\,|\,\omega) = (\inf_{D,\,\theta} p(D\,|\,\theta))^s = \underline{c}^s$ (by Assumption 22) to derive (62). Combining (63) with (60) we see

$$-\ln \int p(\mathbf{D}\,|\,\omega)\, q(\omega\,|\,D^{sm})\, d\omega \leqslant -\ln \int p(\mathbf{D}\,|\,\omega)\, p^*(\omega\,|\,D^{sm})\, d\omega + \kappa^s d(q,\, p^*).$$

Taking an expectation of the both sides of the above inequality with respect to the joint distribution of ω, D^{sm}, and $\mathbf{D}$ gives (45). This completes the proof of Lemma 24. ∎

Proof of Lemma 29. We start with the following claim.

Claim. Let the initial value $\hat{\omega}^{(0)}$ be in R_1 as in (53). *Then at each iteration of the Markov chain* (38) *for HDCB, $|\bar{D} - (1/s)\sum_{i=1}^{s} \theta_i^{(l)}|$ gets multiplied by $1/(1 + (\sigma_\theta^2/\sigma^2)m)$ until it reaches the value $1/\sqrt{s} + O(1/s)$ with at least uniform probability independent of s, m, and N.*

Proof of Claim. Let $\hat{\omega}^{(l)} = (\hat{\theta}_1^{(l)}, ..., \hat{\theta}_s^{(l)}, \hat{\mu}^{(l)})$ be the sampled value at the lth iteration in HDCB. Then we can immediately see that the mean of $(1/s)\sum_{i=1}^{s} \hat{\theta}_i^{(l+1)}$ is $(\sigma_\theta^2 m\bar{D} + \sigma^2\hat{\mu}^{(l)})/(\sigma_\theta^2 m + \sigma^2)$. Using Chebyshev's inequality we can prove that

$|\bar{D} - (1/s)\sum_{i=1}^{s} \hat{\theta}_i^{(l+1)}|$ is within $1/2\sqrt{s}$ of $(1/(1+(\sigma_\theta^2/\sigma^2)\,m))\,|\hat{\mu}^{(l)} - \bar{D}|$, with at least uniform probability independent of s, m, and N. Since the mean of $\hat{\mu}^{(l)}$ is

$$\frac{\sigma_\theta^2\mu_0 + \sigma_0^2 \sum_{i=1}^{s} \hat{\theta}_i^{(l)}}{\sigma_\theta^2 + s\sigma_0^2} = \frac{1}{s} \sum_{i=1}^{s} \hat{\theta}_i^{(l)} + O\left(\frac{1}{s}\right),$$

we can immediately see from the density (40) of $\hat{\mu}^{(l)}$ that $|\hat{\mu}^{(l)} - \bar{D}|$ is within $1/2\sqrt{s} + O(1/s)$ of $|(1/s)\sum_{i=1}^{s} \hat{\theta}_i^{(l)} - \bar{D}|$, with uniform probability. Thus we see that $|\bar{D} - (1/s)\sum_{i=1}^{s} \hat{\theta}_i^{(l+1)}|$ is within $1/\sqrt{s} + O(1/sm)$ of $|\bar{D} - (1/s)\sum_{i=1}^{s} \hat{\theta}_i^{(l)}|$ times $1/(1+(\sigma_\theta^2/\sigma^2)m)$ with uniform probability, say ε_1.

Choose R_1 and R_2 as in (53) and (54), respectively. From the claim above it can be readily proven that for any $\omega \in R_1$, for $l_0 = \lceil (\ln 4NS)/(2\ln(1+(\sigma_\theta^2/\sigma^2)m)) \rceil = O((\ln(sN))/(\ln m))$, with probability at least ε_1 independent of s, m, and N, we have $|\bar{D} - (1/s)\sum_{i=1}^{s} \hat{\theta}_i^{(l_0)}| \leqslant (1/(1+(\sigma_\theta^2/\sigma^2)m))^{l_0} N^{1/2} + 1/\sqrt{s} + O(1/sm) \leqslant 2/\sqrt{s}$ for sufficiently large s. This implies that (50) holds for ε_1, R_1, R_2, and l_0 as above.

Next let us define ε' by

$$\varepsilon' \stackrel{\text{def}}{=} \min_{\mu, x} \left\{ \left(\frac{4}{\sqrt{s}}\right) N\left(\frac{\sigma_\theta^2\mu_0 + \sigma_0^2 sx}{\sigma_\theta^2 + s\sigma_0^2}, \frac{\sigma_\theta^2\sigma_0^2}{\sigma_\theta^2 + s\sigma_0^2} : \mu\right) : |\mu - \bar{D}| \leqslant \frac{2}{\sqrt{s}}, |x - \bar{D}| \leqslant \frac{2}{\sqrt{s}} \right\}.$$

Then we see that for the probability measure Q as in Lemma 29, for all $\omega \in R_2$, for all $S \subset \Omega$, we have $K(\omega, S) \geqslant \varepsilon' Q(S)$. Since for any fixed $\bar{D}$, there exists $\varepsilon_2 > 0$ independent of s, m, and N such that $\varepsilon' > \varepsilon_2$, we see that (51) holds for ε_2, R_2, and Q as above. This completes the proof of Lemma 29. ∎

ACKNOWLEDGMENT

The author sincerely appreciates the anonymous reviewers for their helpful comments.

Received July 8, 1997; final manuscript received June 10, 1998

REFERENCES

1. Berger, J. O. (1985), "Statistical Decision Theory and Bayesian Analysis," Springer-Verlag, Berlin/New York.
2. Gelfand, A. E., and Smith, A. F. M. (1990), Sampling-based approach to calculating marginal densities, *J. Amer. Statist. Assoc.* **85**, 398–409.
3. Geman, S., and Geman, D. (1984), Stochastic relaxation, Gibbs distributions, and the Bayes restoration of images, *IEEE Trans. Pattern Anal. Mach. Intell.* **6**, 721–741.
4. Hastings, W. K. (1970), Monte Carlo sampling method using Markov chains and their applications, *Biometrika* **57**, 97–109.
5. Haussler, D., Kivinen, J., and Warmuth, M. (1995), Tight worst-case loss bounds for predicting with expert advice, *in* "Computational Learning Theory: Second European Conference, EuroCOLT'95," pp. 69–83, Springer-Verlag, Berlin/New York.

6. Kearns, M., and Seung, H. S. (1993), Learning from a population of hypotheses, "Proceedings of the Sixth Annual ACM Conference of Computational Learning Theory," pp. 101–110, Assoc. Comput. Mach. Press, New York.
7. Knuth, D. E. (1981), "The Art of Computer Programming, Vol. 2: Seminumerical Algorithms," 2nd ed., Addison–Wesley, Reading, MA.
8. Nakamura, A., Abe, N., and Takeuchi, J. (1994), Efficient distribution-free population learning of simple concepts, "ALT'96 Algorithmic Learning Theory," pp. 500–515, Springer-Verlag, Berlin/New York.
9. Nummelin, E. (1984), "General Irreducible Markov Chains and Non-negative Operators," Cambridge Univ. Press, Cambridge, UK.
10. Rosenthal, J. (1993), "Rates of Convergence for Gibbs Sampling for Variance Component Models," Technical Report 9322, Univ. of Toronto, Dept. of Statistics.
11. Sinclair, A. (1993), "Algorithms for Random Generation & Counting: A Markov Chain Approach," Birkhauser, Basel.
12. Tanner, M. A., and Wong, H. W. (1987), The calculation of posterior distributions by data augmentation, *J. Amer. Statist. Assoc.* **82**, 528–550.
13. Vovk, V. G. (1990), Aggregating strategies, "Proceedings of the Third Annual Workshop on Computational Learning Theory," pp. 371–386, Morgan Kaufmann, San Mateo, CA.
14. Yamanishi, K. (1998), A decision-theoretic extension of stochastic complexity and its approximation to learning, *IEEE Trans. Inform. Theory* **44**, No. 4, 1424–1439.
15. Yamanishi, K. (1996), A randomized approximation of the MDL for stochastic models with hidden variables, "Proceedings of the Ninth Annual Conference on Computational Learning Theory," pp. 99–109, Assoc. Comput. Mach. Press, New York.
16. Yamanishi, K. (1997), Distributed cooperative Bayesian learning strategies, *in* "Proceedings of the Tenth Annual Conference on Computational Learning Theory," pp. 250–262, Assoc. Comput. Mach. Press, New York.
17. Yamanishi, K. (1998), Minimax relative loss analysis for sequential prediction algorithms using parametric hypotheses, "Proceedings of the Eleventh Annual Conference on Computational Learning Theory," pp. 32–43.