Ph.D. DISSERTATION

Overcoming Noise and Interference of Labeled Data via Parametric Learning

파라미터 학습 통한 데이터 잡음 및 간섭극복 연구

BY

KIM JOONYOUNG

FEBRUARY 2021

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Ph.D. DISSERTATION

Overcoming Noise and Interference of Labeled Data via Parametric Learning

파라미터 학습 통한 데이터 잡음 및 간섭극복 연구

BY

KIM JOONYOUNG

FEBRUARY 2021

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Overcoming Noise and Interference of Labeled Data via Parametric Learning

파라미터 학습 통한 데이터 잡음 및 간섭극복 연구

지도교수 정 교 민

이 논문을 공학박사 학위논문으로 제출함

2021년 2월

서울대학교 대학원

전기 컴퓨터 공학부

김 준 영

김준영의 공학박사 학위 논문을 인준함

2021년 2월

위 원 장:	이 정 우	(인) [illegible]
부위원장:	정 교 민	(인) [illegible]
위 원:	한 보 형	(인)
위 원:	양 인 순	(인)
위 원:	임 성 수	(인) [illegible]

Abstract

Data-driven approaches based on neural networks have emerged as new paradigm to solve problems in computer vision and natural language processing fields. These approaches achieve better performance compared to existing human-design approaches (heuristic), however, these performance gains solely relies on a large amount of high quality labeled data. Accordingly, it is important to collect a large amount of data and improve the quality of data by analyzing degrading factors in order to well-train a model. In this dissertation, I propose iterative algorithms to relieve noise of labeled data in crowdsourcing system and meta architecture to alleviate interference among them in continual learning scenarios respectively.

Researchers generally collect data using *crowdsourcing* system which utilizes human evaluations [1]. However, human annotators' decisions may vary significantly due to misconceptions of task instructions, the lack of responsibility, and inherent noise [2, 3, 4]. To relieve the *noise* in responses from crowd annotators, I propose novel inference algorithms for discrete multiple choice and real-valued vector regression tasks. Web-based crowdsourcing platforms are widely used for collecting large amount of labeled data. Due to low-paid workers and inherent noise, the quality of acquired data could be easily degraded. The proposed algorithms can overcome the noise by estimating the true answer of each task and a reliability of each worker updating two types of messages iteratively. For performance guarantee, the performances of the algorithms are theoretically proved under probabilistic crowd model. Interestingly, their performance bounds depend on the number of queries per task and the average quality of workers. Under a certain condition, each average performance becomes close to an oracle estimator which knows the reliability of every worker (theoretical upper bound). Through extensive experiments with both real-world and synthetic datasets, the practical performance of algorithms are verified. In fact, they are superior to other state-of-the-art algorithms.

Second, when a model learns a sequence of tasks one by one (*continual learning*), previously learned knowledge may conflict with new knowledge. It is well-known phenomenon called "*Catastrophic Forgetting*" or "*Semantic Drift*" [5, 6, 7]. In this dissertation, we call the phenomena "*Interference*" since it occurs between two knowledge from labeled data separated in time. It is essential to control the amount of noise and interference for neural network to be well-trained.

In the second part of dissertation, to solve the *Interference* among labeled data from consecutive tasks in continual learning scenario, a homeostasis-inspired meta learning architecture (HM) is proposed. The HM automatically controls the intensity of regularization (IoR) by capturing important parameters from the previous tasks and the current learning direction. By adjusting IoR, a learner can balance the amount of interference and degrees of freedom for its current learning. Experimental results are provided on various types of continual learning tasks. Those results show that the proposed method notably outperforms the conventional methods in terms of average accuracy and amount of the interference. In experiments, I verify that HM is relatively stable and robust compared to the existing *Synaptic Plasticity* based methods[8, 9, 10, 11]. Interestingly, the IoR generated by HM appears to be proactively controlled within a certain range, which resembles a negative feedback mechanism of *homeostasis* in synapses.

keywords: Crowdsourcing, Continual Learning, Meta Learning

student number: 2012-20756

Contents

List of Tables

List of Figures

Chapter 1

INTRODUCTION

Crowdsourcing has become one of the cornerstones of research in the development of human computation-based intelligence systems. Recent web-based services such as Amazon Mechanical Turk have arisen and become popular, as they can provide ideal solutions, gathering enormous responses from widespread crowds in a short time with a relatively low budget [13, 14]. For example, ImageNet, a large-scale image database, was a successful project that exploited the idea of crowdsourcing to label 3.2 million images hierarchically [15].

Despite the innovative framework of crowdsourcing systems, responses from workers can be unreliable [16, 4, 2, 17], since workers hired by crowdsourcing systems are low-paid and have low responsibility. Therefore, extensive works have been proceeded to find reliable solutions that infer the true answers from noisy responses. One natural method for aggregating responses is majority voting. But due to its simplicity, Expectation-Maximization (EM)-based algorithms have become popular. Since EM-based algorithms can deal with inference problems with latent variables and unknown model parameters, researchers applied the EM algorithm to proper graphical models for crowdsourcing systems, and showed that their results generally outperform those of majority voting [18, 19, 20]. Recently, Karger et al. [21, 22] made a significant breakthrough by proposing a novel iterative algorithm based on the idea of low-rank matrix approximations and the message passing technique. They showed that the performance of their iterative algorithm is order-optimal, which outperforms majority voting and EM-based algorithms.

Major research studies in this field have concentrated on cases with binary answers, yes (+1) or no (-1) [22, 20]. One example of such a binary case would be when workers have to determine whether a given image is suitable for children. However, real crowdsourced data posted on Amazon Mechanical Turk usually consists of multiple-choice questions and vector regression problems, so more general inference techniques should be employed.

The first part of the dissertation focuses on a more general structure for crowdsourcing systems that can be applied to multiple-choice questions and vector regression problems. As for the multiple-choice question, the examples of vector regression tasks are as follow: (1) Determining the breed of a dog, (2) classifying galaxies accroding to their shapes. Also, those of vector regression tasks are: (1) Rating movies or items, (2) Finding the location of an object in an image, and (3) Estimating a human posture in an image.

Our algorithms iteratively compute relative reliability of each worker in a novel way, where relative reliability is exploited as a weight of the worker's responses. Our algorithm also gets reliable results rapidly with small error compared to majority voting or EM-based algorithms. One of our main contributions is the performance guarantee of our algorithm by proving that the error bound of our algorithm decays exponentially. An interesting aspect of the error bound is its dependency on the negative entropy of workers in a perspective on information theory. Naturally, it is reasonable to assume that the true answers can be revealed by how much information there is in the workers' responses. To verifying the performance of our algorithm, numerical experiments on various cases are conducted and its average performance is close to that of oracle estimator.

The second part of dissertation covers *spatial interference* in continual learning scenario. Human intelligence includes abilities to learn and memorize. Though there are thousands of valuable works on building intelligent machines that learn well, however, when it comes to the memorizing capability, we usually rely on the capacity of storage devices installed on the machines. In this regard, if a machine carries out a sequence of tasks under a memory-limited environment with a single neural network, i.e., a *continual learning* (CL) scenario, it lacks ability to memorize whole knowledge learned in the past and may incur unintended forgetting phenomenon, which we usually call *a catastrophic forgetting* [23, 24, 25]. Such phenomenon happens more under strict conditions, where the network topology cannot be expanded and actual episodic real data cannot be stored, which is called a strict CL.

ELASTIC WEIGHT CONSOLIDATION (EWC) [8], which is recognized as a pioneering research of the strict CL scenario, proposes a way of alleviating such forgetting inspired by a neuroscientific principle, a *Synaptic Plasticity* (SP). The principle teaches us that more frequently activated synapses tend to strengthen over time in human brains. Likewise, the EWC leverages a regularization method based on information obtained from previously-learned tasks when learning a new one. This can be done by finding important parameters of the network for each task, handing their list over to the next task, and using this information to conduct elastic parameter-wise regularization when dealing with the current task. Though the aforementioned approach is an innovative piece of work, however, it leaves the following fundamental questions:

Q1) *Does such regularization always bring positive effects to continual learning?*

Given limited amount of resources to a learner, the regularization done for less forgetting the knowledge acquired may disturb learning new knowledge. Thus, it is necessary for the learner to control an *intensity of regularization (IoR)*, to gain high overall accuracy when learning continually.

Q2) *How can the learner control the IoR for less forgetting?*

Intuitively, we can think of a simple example that can give us a clue to address the question. Suppose (x_1, y_1) and (x_2, y_2) are samples and l is the loss function for learning the samples. Following the principle on *transfer-interference trade-off* in [26], the IoR should be set small when

$$\text{(Weak interference)} \quad \frac{l(x_1, y_1)}{\partial\theta} \cdot \frac{l(x_2, y_2)}{\partial\theta} > 0,$$

where θ denotes the network parameters and $\cdot$ represents a dot-product, meaning the learning directions of the two samples are similar. Thus, with weak interference, choosing small IoR can help increasing the overall accuracy of the learning. On the other hand, with strong interference, the IoR should be set large for less forgetting the acquired knowledge when

$$\text{(Strong interference)} \quad \frac{l(x_1, y_1)}{\partial\theta} \cdot \frac{l(x_1, y_2)}{\partial\theta} < 0.$$

As such, to minimize the amount of forgetting knowledge and increase overall accuracy, the IoR requires to be controlled according to the above-mentioned principle. Throughout this part, we answer **Q1** and **Q2**, by proposing a novel trainable meta network termed *homeostatic meta-model* (HM), which effectively and automatically controls IoR in the main network (Section 4.2). The term homeostatic is brought from the neuroscientific mechanism, *Homeostatic Plasticity*, which prevents neural activities from being driven towards runaway or quiescence states [27, 28]. The proposed HM basically aims to balance the IoR to optimize the overall accuracy of the continual learning by capturing important parameters from the previous tasks and the current learning direction. Specifically, the model batch-wisely generates an optimal IoR, depending on the degree of interference between the previously-learned tasks and the currently drawn batch, and the Euclidean distance between the current-state parameters and the optimal parameters learned from the previous tasks.

To control the amount of correlation between the tasks when training the HM, I propose a Block-wise permutation(`BPERM`)-based continual learning tasks (Section 4.3). The proposed type of tasks is generalization of the existing `MNIST-PERM` type tasks, where the size of permuting blocks can be manually controlled. In consequence, we can possibly train the HM with meta samples depicting various kinds of relations between the tasks.

To show the effectiveness and feasibility of the proposed HM, I conduct experiments on the continual learning with a learner using various types of `BPERM` tasks (Section 4.4). In the preliminary experiment, I found that using a fixed value of IoR degenerates the overall performance of continual learning in terms of accuracy and forgetting. This finding can give a clear answer to **Q1**. Moreover, in the main experiment (continual learning with 10 tasks), I found that the proposed method with IoR generated by the HM outperforms the conventional methods, i.e., EWC, Online-EWC (OEWC [9]), Incremental Moment Matching (IMM [11]). Even in a long sequence of tasks (40 tasks), we showed that the proposed HM stably retains the previously-acquired knowledge compared to the other benchmark methods. Furthermore, I provide the trend of IoR over time during the continual learning. The IoR generated by HM interestingly appears to be proactively controlled within a certain range, which resembles a negative feedback mechanism of *homeostasis* in synapses.

Chapter 2

Reliable multiple-choice iterative algorithm for crowdsourcing systems

2.1 Setup

In this section, we define some variables and notations for problem formulation. Consider a set of m tasks, each of which can be a multiple-choice question that only has one correct answer. The number of choices for task i is denoted D_i. All tasks are distributed to several workers through a proper task allocation strategy.

Suppose that n workers participate to perform m tasks. We consider a probabilistic model to generate responses when workers face tasks. We assume that a worker j is parameterized by a latent variable $p_j \in [0, 1]$, which represents the probability of getting a correct answer. In other words, each worker gives the correct answer with a probability p_j and the wrong answer with probability $1 - p_j$ in the decision-making process. When a worker gives a wrong answer, we can assume that the worker has chosen one of distractors uniformly at random, so the probability of each wrong choice is $\dfrac{1 - p_j}{D_i - 1}$. It is reasonable that this latent variable p_j refers to the reliability of the worker, since it captures the ability or diligence of the worker.

In the response process, when a worker j solves an assigned task i, we define the submitted response $\vec{A}^{ij}$ in vector form. The response is represented as a D_i-dimensional binary unit vector $\vec{A}^{ij}$, having 1-of-D_i representation in which the element indicating the chosen answer is equal to 1 and all other elements are equal to 0. The values of A_d^{ij} therefore satisfy $A_d^{ij} \in \{0, 1\}$ and $\sum_d A_d^{ij} = 1$ where A_d^{ij} is the d^{th} component of the response $\vec{A}^{ij}$. For example, when there are three choices, the possible answer forms are $(1, 0, 0)$, $(0, 1, 0)$, and $(0, 0, 1)$. Our goal is to determine the correct answer for each task by querying and aggregating all the responses from the workers.

2.2 Algorithm

In this section, we propose our multiple-iterative algorithm with a minimum number of assignments. In advance, using random regular bipartite graph-generating model, we emulate a real crowdsourcing system scenario. Then, the message update rules of our iterative algorithm are explained. In addition, we propose the generalized iterative algorithm for general setting such as a adaptive strategy.

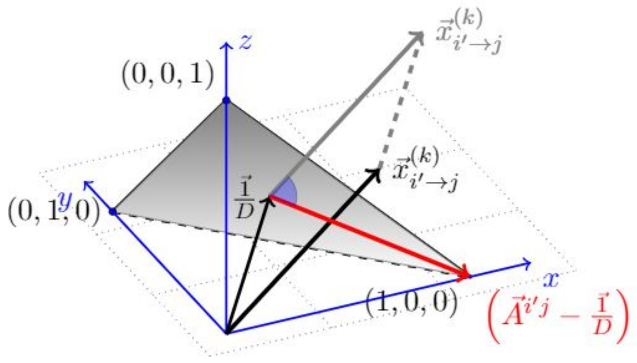
2.2.1 Task Allocation

To design a graph model for a crowdsourcing system, we use a bipartite graph which consists of two types of node sets. m tasks are defined as the set of nodes $[m]$ at the left side of the graph, and n workers are defined as the set of nodes $[n]$ at the right side respectively. Each edge represents an assignment between a task and a worker and this is determined according to the task assignment method. For simplicity, the i^{th} task and the j^{th} worker are denoted as i and j respectively. Given a bipartite graph $G = \{[m] \cup [n], E\}$ representing the allocation graph between tasks and workers, we connect the edge (i, j) if task i is assigned to worker j. We decide the task node degree l in proportion to the resources we can spend. In addition, the worker node degree r is determined by the work capacity that an individual worker can manage. Since we recruit workers through open-call, the (l, r) regular bipartite graph is adequate for our setting. To generate a (l, r) random regular bipartite graph such that $ml = nr$, we bring a simple random construction model known as the *pairing model*(This is also called a configuration model in [22]). In fact, any arbitrary bipartite graph instance can be used for task allocation. However, we will use the *pairing model* which generates a random bipartite graph with a local tree-like property. Using this property, we prove the tight error bounds of our algorithm in Section 3.4.4.

2.2.2 Multiple Iterative Algorithm

In this section, we describe the basic operations of our algorithm and the process of inferring true answers. For each edge (i, j), the response is denoted as $\vec{A}^{ij} \in U = \{\vec{e}_u | u \in [1 : D_i]\}$ which consists of D dimensional binary unit vectors all of whose components are 0 or 1. To extract the true answers from the unreliable responses of workers, we propose an iterative algorithm for multiple-choice questions.

Our algorithm generates two types of messages between task nodes and worker nodes. The first type is the task message $\vec{x}_{i\to j}$, which is denoted as a D_i dimensional vector. Each component of this vector corresponds to the likelihood meaning the possibility being a true answer. The second type is a worker message $y_{j\to i}$ which specifies the reliable worker j. Since these worker messages are strongly correlated with the reliability p_j, our algorithm can assess relative reliability. Hence, we will empirically verify the correlation between $\{y_{j\to i}\}$ and $\{p_j\}$ in section 2.5. The initial messages of our iterative algorithm are sampled independently from the *Gaussian* distribution with unit mean and variance, i.e., $y^{(0)}_{j\to i} \sim \mathcal{N}(1, 1)$. Unlike EM-based algorithms [29, 20], our approach is not sensitive to initial conditions as long as the consensus of the group of workers is positively biased. Now, we define the adjacent set of task i as ∂i and similarly the adjacent set of worker j is defined as ∂j. Then, at the k^{th} iteration, both



$\vec{x}^{(k)}_{i'\to j}$
z
$(0,0,1)$
$(0,1,0)$
y
$\frac{\vec{1}}{D}$
$\vec{x}^{(k)}_{i'\to j}$
x
$(1,0,0)$
$\left(\vec{A}^{i'j}-\frac{\vec{1}}{D}\right)$

Figure 2.1: Description of a task message $\vec{x}^{(k)}_{i'\to j}$ and a response vector $\vec{A}^{i'j}$, in the message vector space when $\vec{A}^{i'j}=(1,0,0)$ and $D_{i'}=3$.

messages are updated using the following rules:

$$\vec{x}^{(k)}_{i\to j}=\sum_{j'\in\partial i\backslash j}\vec{A}^{ij'}y^{(k-1)}_{j'\to i},\qquad \forall(i,j)\in E \tag{2.1}$$

$$y^{(k)}_{j\to i}=\sum_{i'\in\partial j\backslash i}\left(\vec{A}^{i'j}-\frac{\vec{1}}{D}\right)\cdot\vec{x}^{(k-1)}_{i'\to j},\qquad \forall(i,j)\in E \tag{2.2}$$

At the task message update process shown in (2.1), our algorithm gives weight to the answer according to the reliability of a worker. At the worker message update process shown in (2.2), it gives greater reliability to a worker who strongly follows consensus of other workers.

Figure 1 describes two vectors in the message vector space. As shown above, $(\vec{A}^{i'j}-\frac{\vec{1}}{D})$ represents the difference between response of worker j for task i' and the random answer $\frac{\vec{1}}{D}$. Also, $\vec{x}^{(k-1)}_{i'\to j}$ means the weighted sum of responses of other workers who have solved the task i'. Thus, the inner product of these two vectors in (2.2) can assess the similarity between the response of worker j for the task i' and sum of those of other workers who have solved the task i'. A larger positive similarity value of the two vectors means that worker j is more reliable. Meanwhile, the negative value specifies that the worker j does not follow the consensus of other workers and our algorithm regards the worker j as unreliable. Specially, when $\vec{x}^{(k-1)}_{i'\to j}$ and $(\vec{A}^{i'j} - \frac{\vec{1}}{D})$ are orthogonal for fixed task i', the inner product of two vector is close to zero. This means that $\vec{x}^{(k-1)}_{i'\to j}$ does not contribute to the message of the worker j. Then, $y^{(k)}_{j\to i}$ is defined as the sum of the inner product from each task message except for that of task i, representing the relative reliability of the worker j. Returning to (2.1), $\vec{x}^{(k)}_{i\to j}$ is determined by the weighted voting of workers who have solved task i, except for the message from the worker j. The worker j' contributes to the response $\vec{A}^{ij'}$ as much as the weight value $y^{(k-1)}_{j'\to i}$. Thus, $\vec{x}^{(k)}_{i\to j}$ is defined as the sum of $\vec{A}^{ij'} y^{(k-1)}_{j'\to i}$ which represents the estimated true answer for the task i. The following describes the pseudo code of our algorithm.

The maximum number of iterations k_{max} is analyzed in section 2.4.2. In practice, a dozen of iterations is sufficient for the convergence of our algorithm. After k_{max} iterations, our algorithm makes the final estimate vector $\vec{x}_i$ of a task i, and each component of the vector represents the possibility of being the true answer. Our algorithm infers the true answer by choosing u_i that has the maximum component among final likelihoods of $\vec{x}_i$. Then, our algorithm outputs the estimate of the true answer denoted as a unit vector, $\vec{e}_{u_i}$.

2.2.3 Task Allocation for General Setting

In the previous section, we proposed our iterative algorithm for a bipartite graph according to the *pairing model*. However, the number of workers allocated to each task can differ in cases that are more general. That must bring about the variation of the number of tasks that each worker solves. Hence, we consider a general bipartite graph with various node degrees. To apply our algorithm in this scenario, the update rules of both messages should be slightly changed in terms of the task node degree l_i and the worker node degree r_j. For a task message $\vec{x}^{(k)}_{i\to j}$, we divide each message value by the task node degree $(l_i - 1)$ so that tasks with different degrees receive the similar effect from worker nodes. In other words, dividing by $(l_i - 1)$ equalizes the task mes-

Algorithm 1 Multiple Iterative Algorithm

```
1: Input: $E$, $\{\vec{A}^{ij}\}_{(i,j)\in E}$, $k_{max}$
2: Output: Estimation $^\forall i \in [m]$ , $\hat{t}_i \in \{\vec{e}_{u_i} | u_i \in [1:D]\}$
3: For $^\forall (i,j) \in E$ do
4:     Initialize $y^{(0)}_{j\to i}$ with random $Z_{ij} \sim N(1,1)$;
5: For $k = 1, 2 \ldots, k_{max}$ do
6:     For $^\forall (i,j) \in E$ do $\vec{x}^{(k)}_{i\to j} \leftarrow \sum_{j'\in \partial i \setminus j} \vec{A}^{ij'} y^{(k-1)}_{j'\to i}$;
7:     For $^\forall (i,j) \in E$ do $y^{(k)}_{j\to i} \leftarrow \sum_{i'\in \partial j \setminus i} (\vec{A}^{i'j} - \frac{\vec{1}}{D}) \cdot \vec{x}^{(k-1)}_{i'\to j}$;
8: For $^\forall j \in [n]$ do $y_j \leftarrow \sum_{i\in \partial j} (\vec{A}^{ij} - \frac{\vec{1}}{D}) \cdot \vec{x}^{(k_{max}-1)}_{i\to j}$;
9: For $^\forall i \in [m]$ do $\vec{x}_i \leftarrow \sum_{j\in \partial i} \vec{A}^{ij} y^{(k_{max}-1)}_{j\to i}$;
10: Estimate vector $\hat{t}_i = \vec{e}_{u_i}$   where $u_i = \arg\max_d(\vec{x}_i)$
```

Algorithm 2 Generalized Multiple Iterative Algorithm

```
1: Input: $E$, $\{\vec{A}^{ij}\}_{(i,j)\in E}$, $k_{max}$
2: Output: Estimation $^\forall i \in [m]$ , $\hat{t}_i \in \{\vec{e}_{u_i} | u_i \in [1:D_i]\}$
3: For $^\forall (i,j) \in E$ do
4:     Initialize $y^{(0)}_{j\to i}$ with random $Z_{ij} \sim N(1,1)$;
5: For $k = 1, 2 \ldots, k_{max}$ do
6:     For $^\forall (i,j) \in E$ do $\vec{x}^{(k)}_{i\to j} \leftarrow \sum_{j'\in \partial i \setminus j} \left(\frac{1}{l_i - 1}\right) \vec{A}^{ij'} y^{(k-1)}_{j'\to i}$;
7:     For $^\forall (i,j) \in E$ do
8:         $y^{(k)}_{j\to i} \leftarrow \sum_{i'\in \partial j \setminus i} \left(\frac{1}{r_j - 1}\right) (\vec{A}^{i'j} - \frac{\vec{1}}{D_{i'}}) \cdot \vec{x}^{(k-1)}_{i'\to j}$;
9: For $^\forall j \in [n]$ do
10:        $y_j \leftarrow \sum_{i\in \partial j} \left(\frac{1}{r_j - 1}\right) (\vec{A}^{ij} - \frac{\vec{1}}{D_i}) \cdot \vec{x}^{(k_{max}-1)}_{i\to j}$;
11: For $^\forall i \in [m]$ do $\vec{x}_i \leftarrow \sum_{j\in \partial i} \left(\frac{1}{l_i - 1}\right) \vec{A}^{ij} y^{(k_{max}-1)}_{j\to i}$;
12: Estimate vector $\hat{t}_i = \vec{e}_{u_i}$   where $u_i = \arg\max_d(\vec{x}_i)$
```

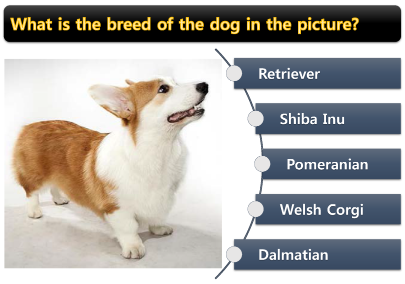
sage values. Likewise, a worker message $y_{j\to i}^{(k)}$ is divided by the worker node degree $(r_j - 1)$ for general setting.

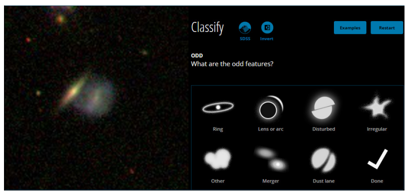
In addition to the generalization of the degree profile, we consider the various number of choices for each task (For example $\forall i \in [m]$, $D_i \in \{2, 3, 4\}$). In practice, the number of choice for each task can differ from one another and our Algorithm 2 can cope with this variation. The following describes the pseudo code of our generalized algorithm.

Adaptive task allocation method. One of significant points of our algorithm is that worker's relative reliability can be assessed in the course of its iterations. If we use this property, the performance of inferring the true answer can be improved further. Consider the adaptive strategy as an improvement method using the above property. First, a small portion of the tasks is used to infer the reliability of each worker using the iterative algorithm. Then, we select partial workers who have higher worker values to message and let them solve all of the remaining tasks. Although this method gives a larger burden to workers who are more reliable, the total number of edges is maintained. In section 2.5, the adaptive task allocation method will be explained in detail and we will verify some of the gains of this method through several experiments.

2.3 Applications

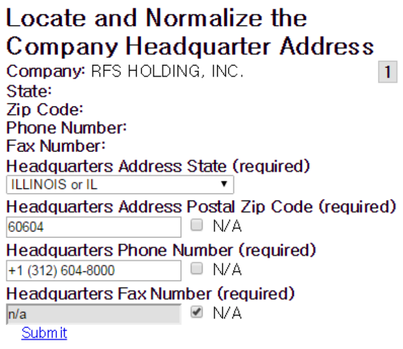
We described an algorithmic solution to crowdsourcing systems for multiple-choice questions in the previous section, and we now look into some applications that our algorithm can treat. As we can see in crowdsourcing systems like Amazon Mechanical Turk, tasks are distributed in the form of multiple-choice questions and short-answer questions like entering zip-code. Although previous algorithms like [22, 20] have shown remarkable results in binary cases, a merit of our algorithm is that outstanding results can even be achieved on multiple-choice and short-answer questions that real tasks usually contain. Furthermore, a remarkable characteristic of our model





(a) An independent multiple-choice question: Determining the breed of a dog.

(b) GalaxyZoo project: classifying galaxies according to their shapes.



Locate and Normalize the Company Headquarter Address

1

Company: RFS HOLDING, INC.
State:
Zip Code:
Phone Number:
Fax Number:

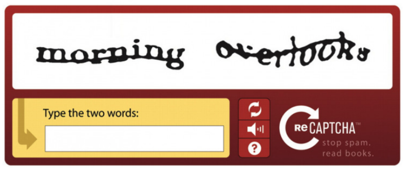
Headquarters Address State (required)
ILLINOIS or IL

Headquarters Address Postal Zip Code (required)
60604 ☐ N/A

Headquarters Phone Number (required)
+1 (312) 604-8000 ☐ N/A

Headquarters Fax Number (required)
n/a ☑ N/A

Submit



(c) A real task in Amazon Mechanical Turk: Filling up address information of a given company.

(d) reCAPTCHA: Typing words for spam protection and a book digitization project.

Figure 2.2: Examples of multiple-choice questions.

is that the number of choices can vary for each question. This flexibility makes our model more applicable for real crowdsourced data. In this section, we describe some applications in detail that can apply our algorithm.

Labeling or tagging images is a common usage of exploiting crowdsourcing systems, and shows successful results in practice [15]. One of such example is classifying species or breeds of dogs in the images illustrated in Figure 2.2(a). Such tasks are very tough for machines, and even humans who have no background knowledge of dogs. These tasks are suitable for crowdsourcing materials and have multiple choices that are directly applicable to our algorithm.

Another application of labeling tasks is Galaxy Zoo, one of the well known projects using the wisdom of crowds (cf. Figure 2.2(b)). Galaxy Zoo has distributed over 300,000 images of galaxies to crowds for classification by their shape. Any volunteer with no prior knowledge can visit the website, where they are presented with an image of a galaxy and instructions of labeling manner. Then they answer a series of questions about the visual form of the galaxy, like whether it has arms or a bulge. Each step consists of multiple-choice questions, and the number of choices varies for each question. Since our algorithm is flexible for the number of choices, the responses of Galaxy Zoo can be easily aggregated using our algorithm.

For short-answer questions, it is hard to aggregate workers' responses in general, because their responses can vary. Our algorithm can settle this problem with the idea of transforming short-answer questions into several multiple-choice questions. When the length of the response to a short-answer question is fixed, short-answer questions can be split into several smaller tasks by considering each character of a response. In other words, each character is treated as one *microtask* in short-answer questions. For example, consider the task of entering a fixed-length answer such as a zip code like 97232. It can be treated as five microtasks, and each of the characters has 10 possible answers, from 0 to 9. Note that in each microtask, we only consider the number of choices as much as the number of candidate answers. For example, if candidate answers for a microtask are “4”, “7”, and “9”, then we set the number of choices to three for this microtask. In addition, we can decide a set of candidate answers as all gathered responses simply, or only responses of top-K likelihood effectively.

Next, we consider when the length of the response varies. We can make another small task that determines the true length of the response and then we can discard the answers whose length is determined as a minor option. In summary, every short-answer question can be decomposed to several microtasks by considering each character of the answer and its length. Characters of the response and its length are transformed into small microtasks, and each microtask is considered a multiple-choice question. Thus, by applying our algorithm, responses to these short-answer questions can be easily aggregated. For a real task in Amazon Mechanical Turk, as illustrated in Figure 2.2(c), entering zip codes or phone numbers is an example of short-answer questions.

Another popular crowdsourcing application for short-answer questions is reCAPTCHA [30] illustrated in Figure 2.2(d). In its original version, CAPTCHA was first introduced to distinguish automatic bots by typing some characters correctly in a given image. It was extended to reCAPTCHA which digitalizes some hard phrases that Optical Character Recognition (OCR) techniques cannot recognize. In this case, the length of responses can vary, so a small task determining the length of response is necessary, as we mentioned. Although discarding the rest of the responses can be viewed as a waste, it is a tolerable loss, since the length of the responses is generally consistent. In addition, we need discuss the number of tasks r, each worker is given. In reCAPTCHA, we can only assign one entering task to each worker, while our algorithm needs sufficient number of tasks for each worker to ensure reliable inference. However, since we split each worker’s response into several microtasks, the task size problem is naturally solved.

Another special application of our algorithm is as an adaptive task allocation strategy, since it explicitly computes the relative reliability of the workers, even with no prior knowledge of the worker distribution. If we design a proper adaptive strategy for crowdsourcing systems, we can boost its performance from the perspective of quality control of workers. The best workers can be recruited and exploited to resolve more questions. It can be viewed as a method for finding experts from crowds or filtering out workers who just spam for rewards; therefore, we can exploit reliable workers efficiently under the same budget through an adaptive task allocation strategy. We will examine such an adaptive strategy in the experiment section.

2.4 Analysis of algorithms

In this section, we provide proof for the performance guarantee of Algorithm 1. In *Theorem 1*, we show that the error bound depends on task degree l and the quality of the workers. More precisely, we show that an upper bound on the probability of error decays exponentially. From this section, we assume that $D_i = D$ for all $i \in [n]$.

2.4.1 Quality of workers

Let $\vec{v}_j$ denote the confusion vector of each worker j. Each component of the vector means the probability that a worker chooses the corresponding choice for a response. For a fixed task i with true answer $\hat{t}_{u_i} \in U$, the confusion vector $\vec{v}_j$ of worker j is defined as follows:

$$v_{jd} = \begin{cases} p_j & \text{if } \hat{t}_{u_i} = \vec{e}_d \\ \frac{1-p_j}{D-1} & \text{otherwise} \end{cases}$$

From an information theoretical perspective, the quality of workers can be defined as negative entropy with an offset and using the above confusion vector, we can define the quality of workers as

$$q = \mathbb{E}\Big[H(p) - \bar{p}\log(\hat{D}) + \log(D)\Big], \tag{2.3}$$

$$\text{where} \quad H(p) = p\log p + \bar{p}\log\bar{p}, \quad \bar{p} = 1-p, \quad \hat{D} = D-1.$$

According to the quality of each worker, we can divide the workers into three types. At the extreme, workers with a quality close to zero make arbitrary responses. Since,

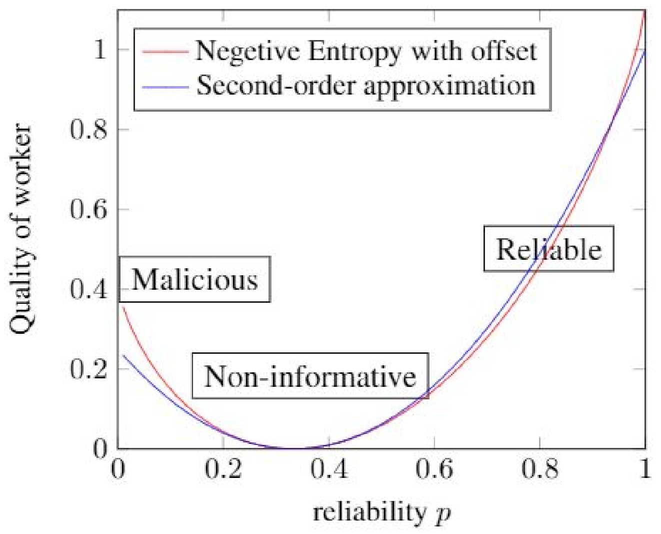


Figure 2.3: Comparison of the quality between negative entropy with offset and second-order polynomial approximation.

we cannot obtain any information from them, let us define them as “Non-informative workers.” At the other extreme, workers with the a quality close to one make almost true answers and we call them “Reliable workers.” Lastly, there are workers who make wrong answers on purpose and affect the crowdsourcing system badly; they can be regarded as “Malicious workers.” In our algorithm, since the worker message value y_j is related to the quality, workers with negative y_j, positive y_j and y_j close to zero correspond to “Reliable workers,” “Malicious workers,” and “Non-informative workers,” respectively.

Although the quality of workers theoretically follows negative entropy, we found that a second-order polynomial approximation is sufficient for our analysis as described in Figure 2.3. As the dimension of the tasks increase, the approximation deviates from the real quality. Nevertheless, second-order approximation fits well to the real quality in the acceptable dimension case that our algorithm targets.

$$q \simeq \tilde{q}_1 = \mathbb{E}\left[\left(\frac{D}{D-1}\right)^2\left(p_j - \frac{1}{D}\right)^2\right] \tag{2.4}$$

For simplicity, we will use this approximated quality in the following sections. There is one more necessary assumption about worker distribution that workers give the correct answers on average rather than random or adversarial answers, so that $E\left[p_j\right] > \dfrac{1}{D}$. Given only workers' responses, any inference algorithms analogize the true answers from the general or popular choices of crowds. Consider an extreme case in which everyone gives adversarial answers in a binary classification task; no algorithm can correctly infer the reliability of the crowd. Hence, the assumption $E\left[p_j\right] > \dfrac{1}{D}$ is a natural necessary.

2.4.2 Bound on the Average Error Probability

From now on, let $\hat{l} \equiv l-1$, $\hat{r} \equiv r-1$, and the average quality of workers is defined as $q = \mathbb{E}[(\frac{D}{D-1})^2(p_j - \frac{1}{D})^2]$. Also, σ_k^2 denotes the effective variance in the *sub-Gaussian* tail of the task message distribution after k iterations.

$$\sigma_k^2 \equiv \frac{2q}{\mu^2 T^{k-1}} + \left(\frac{D}{D-1}\right)^2 \left(3 + \frac{1}{8q\hat{r}}\right)\left[\frac{1-1/T^{k-1}}{1-1/T}\right], \tag{2.5}$$

$$\text{where} \quad T = \frac{(D-1)^2}{(D^2-D-1)} q^2 \hat{l}\hat{r}.$$

Theorem 1 *For fixed $l > 1$ and $r > 1$, assume that m tasks are assigned to n workers according to a random (l,r)-regular bipartite graph according to the pairing model. If the distribution of the reliability satisfies $\mu \equiv \mathbb{E}[\frac{D}{D-1}(p_j - \frac{1}{D})] > 0$ and $T > 1$, then for any $t \in \{e_i\}^m$, the estimate after k iterations of the iterative algorithm achieves*

$$\frac{1}{m}\sum_{i=1}^{m} \mathbb{P}(t_i \neq \hat{t}_i^{(k)}) \leqslant (D-1)e^{-lq/(2\sigma_k^2)} + \frac{3lr}{m}(\hat{l}\hat{r})^{2k-2}. \tag{2.6}$$

The second term of the equation is the upper bound of probability that the graph dose not have a local tree-like structure and it can be quite small as long as we treat a large number of tasks. Therefore, the dominant factor of the upper bound is the first exponential term. As shown in (2.5), $T = 1$ is the crucial condition and we can satisfy

$T > 1$ by using a sufficiently larger l or r. Then, with $T > 1$, σ_k^2 converges to a finite limit σ_∞^2, and we have

$$\sigma_\infty^2 = \left(3 + \frac{1}{8q\hat{r}}\right)\left(\frac{T}{T-1}\right). \tag{2.7}$$

Thus, the bounds of the first term of (2.6) does not depend on the number of tasks m or the number of iterations k. The following corollary describes an upper bound that only depends on l, q, σ_∞^2, and D.

Corollary 1 *Under the hypotheses of Theorem 1, there exists* $m_0 = 3lre^{lq/4\sigma_\infty^2}(\hat{l}\hat{r})^{2(k-1)}$ *and* $k_0 = 1 + (\log(q/\mu^2)/\log T)$ *such that*

$$\frac{1}{m}\sum_{i=1}^{m}\mathbb{P}(t_i \neq \hat{t}_i^{(k)}) \leqslant De^{-lq/(4\sigma_\infty^2)}, \tag{2.8}$$

for all $k \geqslant k_0$ *and for all* $m \geqslant m_0$.

First, we will show that $\sigma_k^2 \leqslant 2\sigma_\infty^2$ for $k \geqslant 1 + (\log(q/\mu^2)/\log T)$. Since $T > 1$, as per our assumption, $\sigma_k^2 = (2q/\mu^2 T^{k-1}) + (\frac{D}{D-1})^2(3 + 1/8q\hat{r})\frac{1-1/T^{k-1}}{T-1} \leqslant 2 + \sigma_\infty^2 \leqslant \sigma_\infty^2 + \sigma_\infty^2 \leqslant 2\sigma_\infty^2$. Therefore, the first term of (2.6) is bounded like $(D-1)e^{-lq/2\sigma_k^2} \leqslant (D-1)e^{-lq/4\sigma_\infty^2}$. Next, it is sufficient to set $m \geqslant 3lre^{lq/4\sigma_\infty^2}(\hat{l}\hat{r})^{2(k-1)}$ to ensure $\frac{3lr}{m}(\hat{l}\hat{r})^{2k-2} \leqslant e^{-lq/(4\sigma_\infty^2)}$.

From *corollary 1*, we obtained that the required number of iterations k_0, is small in that it is the only logarithmic in l,r,q,μ and D. On the other hand, although the required number of entire tasks m_0, is very large in *corollary 1*, the experimental result in section 2.5 shows that the performance of error exhibits exponential decay as stated in (2.8).

Now, if we assume that there are no limitation on worker degree r and $T \geqslant 2$, we can find $\sigma_\infty^2 \leqslant 2(3 + 1/8q\hat{r})$. Then, for all $r \geqslant 1 + 1/8q$, as similar with the [31], we get the following bound:

$$\frac{1}{m}\sum_{i=1}^{m}\mathbb{P}(t_i \neq \hat{t}_i^{(k)}) \leqslant De^{-lq/32}. \tag{2.9}$$

Also, we can check the following corollary in terms of the number of queries per task l to achieve a target accuracy. Hence, we get the following corollary.

Corollary 2 *Using the task assignment scheme according to pairing model with* $r \geqslant 1 + 1/8q$ *and the iterative algorithm, it is sufficient to query* $(32/q)log(D/\epsilon)$ *times per task to guarantee that the error bound is at most* ϵ *for any* $\epsilon \leqslant 1/2$ *and for all* $m \geqslant m_0$.

2.4.3 Proof of the Theorem 1

The proof is roughly composed of three parts. First, the second term at the right-hand side of (2.6) is proved using its local tree-like property. Second, the remaining term of the right-hand side of (2.6) is verified using *Chernoff bound* in the assumption that the estimates of the task message follow *sub-Gaussian* distribution. Lastly, we prove that the assumption of the second part is true within certain parameters. To apply *density evolution* with multi-dimensional vector form is difficult in that the cross term of each components are generated. Therefore our proof can be differentiated from binary setting in [31].

Without a loss of generality, it is possible to assume that the true answer of each task, for any $i \in [m]$, $t_i = \vec{e}_1$. Let $\hat{t}_i^{(k)}$ denote the estimated answer of task i defined in section 2.4.2. If we draw a task **I**, uniformly at random from the task set, the average probability of error can be denoted as

$$\frac{1}{m}\sum_{i\in[m]} \mathbb{P}(t_i \neq \hat{t}_i^{(k)}) = \mathbb{P}(t_{\mathbf{I}} \neq \hat{t}_{\mathbf{I}}^{(k)}), \tag{2.10}$$

Let $\mathrm{G}_{\mathbf{I},k}$ denote a subgraph of G that consists of all the nodes whose distance from the node '**I**' is at most k. After k iterations, the local graph with root '**I**' is $\mathrm{G}_{\mathbf{I},2k-1}$, since the update process operates twice for each iteration. To take advantage of *density evolution*, the full independence of each branch is needed. Thus, we bound the probability of error with two terms, one that represents the probability that subgraph $\mathrm{G}_{\mathbf{I},2k-1}$ is not a tree and the other, which represents the probability that $\mathrm{G}_{\mathbf{I},2k-1}$ is a tree with a wrong answer.

$$\begin{aligned}\mathbb{P}(t_{\mathbf{I}} \neq \hat{t}_{\mathbf{I}}^{(k)}) &\leqslant \mathbb{P}(\mathrm{G}_{\mathbf{I},2k-1} \text{ is not a tree }) \\ &+ \mathbb{P}(\mathrm{G}_{\mathbf{I},2k-1} \text{ is a tree and } t_{\mathbf{I}} \neq \hat{t}_{\mathbf{I}}^{(k)}). \qquad (2.11)\end{aligned}$$

The following lemma bounds the first term and proves that the probability that a local subgraph is not a tree vanishes as m grows. A proof of Lemma 1 is provided [31] (cf. *Karger, Oh and Shah* 2011, section 3.2).

Lemma 1 *From a random (l,r)-regular bipartite graph generated according to the pairing model,*

$$\mathbb{P}(\mathrm{G}_{\mathbf{I},2k-1} \textit{ is not a tree }) \leqslant \left(\hat{l}\hat{r}\right)^{(2k-2)} \frac{3lr}{m}.$$

From the result of Lemma 1, we can concentrate directly on the second term of (2.11) and define the pairwise difference of task messages as $\tilde{\mathbf{x}}_d^{(k)} = \mathbf{x}_1^{(k)} - \mathbf{x}_d^{(k)}$ for $^{\forall}d \in [2:D]$.

$$\begin{aligned}\mathbb{P}(t_{\mathbf{I}} \neq \hat{t}_{\mathbf{I}}^{(k)} | \mathrm{G}_{\mathbf{I},k} \text{ is a tree}) &\leqslant \mathbb{P}(\cup_{d=2}^{D}\{\tilde{x}_{\mathbf{I}}^{(k)} \leqslant 0\} | \mathrm{G}_{\mathbf{I},k} \text{ is a tree}) \\ &\leqslant \mathbb{P}(\cup_{d=2}^{D}\{\tilde{x}_{\mathbf{I}} \leqslant 0\}).\end{aligned}$$

To obtain a tight upper bound on $\mathbb{P}(\cup_{d=2}^{D}\{\hat{\mathbf{x}}_d^{(k)} \leqslant 0\})$ of our iterative algorithm, we assume that $\tilde{\mathbf{x}}_d^{(k)}$ follow *sub-Gaussian* distribution for any $d \in [2:D]$ and prove these in section 2.4.4. Then, *Chernoff bound* is applied to the independent message branches and this brings us the tight bound of our algorithm. A random variable $\mathbf{z}$ with mean m is said to be *sub-Gaussian* with parameter $\tilde{\sigma}$ if for any $\lambda \in \mathbb{R}$ the following inequality holds:

$$\mathbb{E}[e^{\lambda \mathbf{z}}] \leqslant e^{m\lambda + (1/2)\tilde{\sigma}^2\lambda^2}. \qquad (2.12)$$

We will first show that for $^{\forall}d \in [2:D]$, $\tilde{\mathbf{x}}_d^{(k)}$ is *sub-Gaussian* with mean m_k and parameter $\tilde{\sigma}_k^2$ for specific region of λ, precisely for $|\lambda| \leqslant 1/(2m_{k-1}\hat{r})$. Now we define

$$m_k \equiv \mu \hat{l} U^{k-1}, \quad ^{\forall}k \in \mathbb{N}$$

$$\tilde{\sigma}_k^2 \equiv 2\hat{l}S^{k-1} + [\mu^2\hat{l}^2\hat{r}(3q^2\hat{l}\hat{r} + \hat{l}/8)]U^{2k-4}\left[\frac{1-(1/T)^{k-1}}{1-(1/T)}\right],$$
$$\text{where } U = \frac{D-1}{D}q\hat{l}\hat{r}, \quad S = \frac{D^2-D-1}{D^2}\hat{l}\hat{r}$$
$$T = \frac{U^2}{S} = \frac{(D-1)^2}{D^2-D-1}q^2\hat{l}\hat{r}$$

then

$$\mathbb{E}[e^{\lambda \tilde{\mathbf{x}}_d^{(k)}}] \leqslant e^{m_k\lambda + (1/2)\tilde{\sigma}_k^2\lambda^2}. \tag{2.13}$$

The locally tree-like property of a sparse random graph provides the distributional independence among incoming messages, that is $\mathbb{E}[e^{\lambda \hat{\mathbf{x}}_d^{(k)}}] = \mathbb{E}[e^{\lambda \tilde{\mathbf{x}}_d^{(k)}}]^{(l/\hat{l})}$. Thus, $\hat{\mathbf{x}}_d^{(k)}$ satisfies $\mathbb{E}[e^{\lambda \hat{\mathbf{x}}_d^{(k)}}] \leqslant e^{(l/\hat{l})m_k\lambda + ((l/2\hat{l}))\tilde{\sigma}_k^2\lambda^2}$ for all $d \in [2 : D]$. Because of full independence of each branch, we can apply *Chernoff bound* with $\lambda = -m_k/(\tilde{\sigma_k^2})$, and then we obtain

$$\mathbb{P}(\hat{\mathbf{x}}_d^{(k)} \leqslant 0) \leqslant \mathbb{E}[e^{\lambda \hat{\mathbf{x}}_d^{(k)}}] \leqslant e^{-lm_k^2/(2\hat{l}\tilde{\sigma}_k^2)}. \tag{2.14}$$

$$\begin{aligned}\mathbb{P}(\cup_{d=2}^{D}\{\hat{\mathbf{x}}_d^{(k)} \leqslant 0\}) &\leqslant \sum_{d=2}^{D}\mathbb{P}(\hat{\mathbf{x}}_d^{(k)} \leqslant 0) \\ &\leqslant (D-1)e^{-lm_k^2/(2\hat{l}\tilde{\sigma}_k^2)}.\end{aligned} \tag{2.15}$$

Since $m_k m_{k-1}/(\tilde{\sigma}_k^2) \leqslant 1/(3\hat{r})$, we can easily check $|\lambda| \leqslant 1/(2m_{k-1}\hat{r})$. This finalizes the *Proof of the Theorem 1.*

2.4.4 Proof of Sub-Gaussianity

Now we prove that for all $d \in [2 : D]$, $\tilde{\mathbf{x}}_d^{(k)}$ is *sub-Gaussian* with some m_k and $\tilde{\sigma}_k^2$. Recurrence relation of the evolution of the MGFs(moment generating functions) on $\tilde{\mathbf{x}}_d$ and $\mathbf{y_p}$ are stated as

$$\begin{aligned}\mathbb{E}[e^{\lambda \tilde{\mathbf{x}}_d^{(k)}}] &= \Bigg(\mathbb{E}_{\mathbf{p}}\left[\mathbf{p}\mathbb{E}\left[e^{\lambda \mathbf{y}_p^{(k-1)}}|\mathbf{p}\right] + \frac{\bar{\mathbf{p}}}{D-1}\mathbb{E}\left[e^{-\lambda \mathbf{y}_p^{(k-1)}}|\mathbf{p}\right]\right. \\ &\quad \left. + \frac{\bar{\mathbf{p}}}{D-1}(D-2)\right]\Bigg)^{\hat{l}},\end{aligned} \tag{2.16}$$

$$\begin{aligned}\mathbb{E}[e^{\lambda \mathbf{y}_p^{(k)}}] &= \Bigg(p\mathbb{E}\Big[e^{\lambda(\frac{1}{D}\sum_{d=2}^{D}\tilde{\mathbf{x}}_d^{(k)})}\Big] \\ &\quad + \frac{\bar{p}}{D-1}\sum_{j=2}^{D}\mathbb{E}\Big[e^{\lambda(-\tilde{\mathbf{x}}_j^{(k)}+\frac{1}{D}\sum_{d=2}^{D}\tilde{\mathbf{x}}_d^{(k)})}\Big]\Bigg)^{\hat{r}}, \qquad (2.17)\\ &\text{where} \quad \bar{p} = 1-p \text{ and } \bar{\mathbf{p}} = 1-\mathbf{p}.\end{aligned}$$

Using above MGFs and *mathematical induction*, we can prove that $\tilde{\mathbf{x}}_d^{(k)}$ are *sub-Gaussian*, for all $d \in [2:D]$.

First, for $k = 1$, we prove that all of $\tilde{\mathbf{x}}_d^{(1)}$ are *sub-Gaussian* random variables with mean $m_1 = \mu\tilde{l}$ and variance $\tilde{\sigma}_1^2 = 2\tilde{l}$, where $\mu \equiv \mathbb{E}[\frac{D}{D-1}(p_j - \frac{1}{D})]$. Using *Gaussian* initialization of $\mathbf{y}_p \sim \mathcal{N}(1,1)$, we obtain $\mathbb{E}[e^{\lambda \mathbf{y}_p^{(0)}}] = e^{\lambda+(1/2)\lambda^2}$ regardless of p. Substituting this into equation (13), we have

$$\begin{aligned}\mathbb{E}[e^{\lambda\tilde{\mathbf{x}}_d^{(1)}}] &= \Bigg(\mathbb{E}_{\mathbf{p}}\Big[\mathbf{p}e^{\lambda+(1/2)\lambda^2} + \Big(\frac{1-\mathbf{p}}{D-1}\Big)e^{-\lambda+(1/2)\lambda^2} \\ &\quad + \Big(\frac{1-\mathbf{p}}{D-1}\Big)(D-2)\Big]\Bigg)^{\hat{l}} \\ &\leqslant \Big(\mathbb{E}[a]e^{\lambda} + \Big(\mathbb{E}[\bar{a}]e^{-\lambda}\Big)^{\hat{l}} e^{(1/2)\hat{l}\lambda^2} \\ &\leqslant e^{(\mu\lambda+\lambda^2)\hat{l}}, \qquad (2.18)\end{aligned}$$

$$\text{where} \quad a = \frac{Dp+D-2}{2(D-1)}, \quad \bar{a} = 1-a = \frac{D(1-p)}{2(D-1)}$$

where the first inequality follows from the fact that $2 \leqslant e^{\lambda} + e^{-\lambda}$ for any $\lambda \in \mathbb{R}$, and the second inequality follows from that

$$be^{z} + (1-b)e^{-z} \leqslant e^{(2b-1)z+(1/2)z^2}, \qquad (2.19)$$

for any $z \in \mathbb{R}$ and $b \in [0,1]$ (cf. *Alon and Spencer* 2008, Lemma A.1.5) [32].

From k^{th} inductive hypothesis, we have $\mathbb{E}[e^{\lambda\tilde{\mathbf{x}}_d^{(k)}}] \leqslant e^{m_k\lambda+(1/2)\tilde{\sigma}_k^2\lambda^2}$ for $|\lambda| \leqslant 1/(2m_{k-1}\hat{r})$. Now, we will show $\mathbb{E}[e^{\lambda\tilde{\mathbf{x}}_d^{(k+1)}}] \leqslant e^{m_{k+1}\lambda+(1/2)\tilde{\sigma}_{k+1}^2\lambda^2}$ for $|\lambda| \leqslant 1/(2m_k\hat{r})$. In advance, substituting (2.19) into (2.17), we have

Lemma 2 *For any* $|\lambda| \leqslant 1/(2m_k\hat{r})$ *and* $p \in [0,1]$, *we get*

$$\mathbb{E}[e^{\lambda \mathbf{y}_p^{(k)}}] \leqslant \Big[\Big(pe^{(1/2)m_k\lambda} + \bar{p}e^{-(1/2)m_k\lambda}\Big)\Big]^{\hat{r}}$$

$$\cdot e^{(\frac{D-2}{2D})\hat{r}m_k\lambda+(\frac{D^2-D-1}{D^2})\hat{r}\tilde{\sigma}_k^2\lambda^2}.$$

Similar to (2.18)'s process, from (2.16), we get

$$\mathbb{E}[e^{\lambda\tilde{\mathbf{x}}_d^{(k+1)}}] \leqslant \mathbb{E}_{\mathbf{p}}\Big(a\mathbb{E}\Big[e^{\lambda\mathbf{y}_p^{(k)}}\Big] + \bar{a}\mathbb{E}\Big[e^{-\lambda\mathbf{y}_p^{(k)}}\Big]\Big)^{\hat{l}}.$$

with $2 \leqslant e^{\lambda} + e^{-\lambda}$ for any $\lambda \in \mathbb{R}$.

Substituting the result of Lemma 2 into the above inequality provides

$$\begin{aligned}\mathbb{E}[e^{\lambda\tilde{\mathbf{x}}_d^{(k+1)}}] \leqslant \mathbb{E}_{\mathbf{p}}\Big[&a\Big(\mathbf{p}e^{(1/2)m_k\lambda} + \bar{\mathbf{p}}e^{-(1/2)m_k\lambda}\Big)^{\hat{r}} \\ &+\bar{a}\Big(\mathbf{p}e^{(1/2)m_k\lambda} + \bar{\mathbf{p}}e^{-(1/2)m_k\lambda}\Big)^{\hat{r}}\Big]^{\hat{l}} \\ &\cdot e^{(\frac{D-2}{2D})\hat{l}\hat{r}m_k\lambda+(\frac{D^2-D-1}{D^2})\hat{l}\hat{r}\tilde{\sigma}_k^2\lambda^2}.\end{aligned} \tag{2.20}$$

Now we are left to bound (2.20) using following Lemma 3.

Lemma 3 *For any $|z| \leqslant 1/(2\hat{r})$ and $p \in [0,1]$, we get*

$$\mathbb{E}_{\mathbf{p}}\Big[a\Big(\mathbf{p}e^{\frac{D-1}{D}z} + \bar{\mathbf{p}}e^{-\frac{1}{D}z}\Big) + \bar{a}\Big(\mathbf{p}e^{-\frac{D-1}{D}z} + \bar{\mathbf{p}}e^{\frac{1}{D}z}\Big)\Big]^{\hat{r}}$$

$$\leqslant e^{\frac{D-1}{D}q\hat{r}z+\left(\frac{3}{2}q\hat{r}+\frac{1}{8}\right)\hat{r}z^2}.$$

Applying this to (2.20) gives

$$\mathbb{E}[e^{\lambda\tilde{\mathbf{x}}_d^{(k+1)}}] \leqslant e^{\frac{D-1}{D}q\hat{l}\hat{r}m_k\lambda+\left[\left(\frac{3}{2}q\hat{r}+\frac{1}{8}\right)m_k^2+\left(\frac{D^2-D-1}{D^2}\right)\tilde{\sigma}_k^2\right]\hat{l}\hat{r}},$$

for $|\lambda| \leqslant 1/(2m_k\hat{r})$.

From the result of *mathematical induction*, we can obtain the recurrence relations of two parameters of the *sub-Gaussians*

$$m_{k+1} = \Big[\frac{D-1}{D}q\hat{l}\hat{r}\Big]m_k,$$

$$\tilde{\sigma}_{k+1}^2 = \Big[\Big(\frac{3}{2}q\hat{r} + \frac{1}{8}\Big)m_k^2 + \Big(\frac{D^2-D-1}{D^2}\Big)\tilde{\sigma}_k^2\Big]\hat{l}\hat{r},$$

with $\frac{D-1}{D}q\hat{l}\hat{r} \geqslant 1$, where m_k is increasing geometric series. Thus, the above recursions hold for $|\lambda| \leqslant 1/(2m_k\hat{r})$ and we get

$$m_k = \mu\hat{l}\Big[\frac{D-1}{D}q\hat{l}\hat{r}\Big]^{k-1},$$

for all $k \in \mathbb{N}$. Substituting m_k into $\tilde{\sigma}_k^2$, we obtain

$$\tilde{\sigma}_k^2 = a\tilde{\sigma}_{k-1}^2 + bc^{k-2}, \tag{2.21}$$

where

$$a = \frac{D^2 - D - 1}{D^2}\hat{l}\hat{r}, \quad b = \mu^2\hat{l}^3\hat{r}\big(\frac{3}{2}q\hat{r} + \frac{1}{8}\big) \quad c = \Big[\frac{D-1}{D}q\hat{l}\hat{r}\Big]^2$$

For $T \neq 1$, This type of recurrence relation can be represented as the following closed formula.

$$\tilde{\sigma}_k^2 = \tilde{\sigma}_1^2 a^{k-1} + bc^{k-2}\Big[\frac{1-(a/c)^{k-1}}{1-(a/c)}\Big]. \tag{2.22}$$

This finishes the proof of (2.13).

Proof of Lemma 2. In the $k{+}1^{th}$ inductive step of *mathematical induction*, we assume that $\mathbb{E}[e^{\lambda\tilde{\mathbf{x}}_d^{(k)}}] \leqslant e^{m_k\lambda+(1/2)\tilde{\sigma}_k^2\lambda^2}$ for any $d \in [2:D]$ with $|\lambda| \leqslant 1/(2m_{k-1}\hat{r})$. In other words, all of $\tilde{\mathbf{x}}_d^{(k)}$ follow *sub-Gaussian* distribution with parameters m_k and $\tilde{\sigma}_k^2$. From (2.17), each component at the right-hand side can be represented as the product of several combinations of $[e^{\lambda\tilde{\mathbf{x}}_d^{(k)}}]$ and the product of variables means a linear combination in the exponential field. Using *hölder's inequality*, we prove that the linear transformation of *sub-Gaussian* random variables follows also *sub-Gaussian* distribution with some parameters. Moreover, these parameters are determined by D, mean m_k and variance $\tilde{\sigma}_k^2$ of each *sub-Gaussian* $\tilde{\mathbf{x}}_d^{(k)}$. Applying *h"older's inequality* to (2.17), the first term at the right-hand side of (2.17) gives

$$\mathbb{E}\Big[e^{\lambda(\frac{1}{D}\sum_{d=2}^{D}\tilde{\mathbf{x}}_d^{(k)})}\Big] \leqslant \prod_{d=2}^{D}\Big[\mathbb{E}\Big(e^{\lambda(1/D)\tilde{\mathbf{x}}_d^{(k)}}\Big)^{D-1}\Big]^{\frac{1}{D-1}}$$

$$\leqslant e^{(\frac{D-1}{D})m_k\lambda+(\frac{D-1}{2D^2})\tilde{\sigma}_k^2\lambda^2}.$$

For the second term at the right-hand side of (2.17), we have

$$\mathbb{E}\Big[e^{\lambda(-\tilde{\mathbf{x}}_j^{(k)}+\frac{1}{D}\sum_{d=2}^{D}\tilde{\mathbf{x}}_d^{(k)})}\Big] \leqslant \mathbb{E}\Big[e^{-\lambda(\frac{D-1}{D})\tilde{\mathbf{x}}_j^{(k)}}\Big]$$

$$\cdot \prod_{d=2,d\neq j}^{D}\Big[\mathbb{E}\Big(e^{\lambda(1/D)\tilde{\mathbf{x}}_d^{(k)})}\Big)^{D-1}\Big]^{\frac{1}{D-1}}$$

$$\leqslant e^{(-\frac{1}{D})m_k\lambda+(\frac{D^2-D-1}{2D^2})\tilde{\sigma}_k^2\lambda^2}.$$

Getting these two results together finishes the proof of Lemma 2.

Proof of Lemma 3. From (2.19), we get

$$\Big(\mathbf{p}e^{\frac{D-1}{D}z}+\bar{\mathbf{p}}e^{-\frac{1}{D}z}\Big) \leqslant e^{(\mathbf{p}-\frac{1}{D})z+\frac{1}{8}z^2}.$$

Applying this result to the original formula, we have

$$\mathbb{E}_{\mathbf{p}}\Big[a\Big(\mathbf{p}e^{\frac{D-1}{D}z}+\bar{\mathbf{p}}e^{-\frac{1}{D}z}\Big)+\bar{a}\Big(\mathbf{p}e^{-\frac{D-1}{D}z}+\bar{\mathbf{p}}e^{\frac{1}{D}z}\Big)\Big]^{\hat{r}}$$

$$\leqslant \mathbb{E}\Big[e^{\frac{D}{D-1}(\mathbf{p}-\frac{1}{D})\hat{r}z+\frac{1}{2}(\mathbf{p}-\frac{1}{D})^2\hat{r}^2z^2}\Big]\cdot e^{\frac{1}{8}\hat{r}z^2}.$$

In this point, we bring the fact that $e^a \leqslant 1+a+0.63a^2$ for $|a| \leqslant 5/8$

$$\mathbb{E}\Big[e^{\frac{D}{D-1}(\mathbf{p}-\frac{1}{D})\hat{r}z+\frac{1}{2}(\mathbf{p}-\frac{1}{D})^2\hat{r}^2z^2}\Big]$$

$$\leqslant \mathbb{E}\Big[1+\Big(\frac{D-1}{D}\Big)q\hat{r}z+\frac{1}{2}\Big(\frac{D-1}{D}\Big)^2q\hat{r}^2z^2$$

$$+0.63\Big\{\Big(\frac{D-1}{D}\Big)q\hat{r}z+\frac{1}{2}\Big(\frac{D-1}{D}\Big)^2q\hat{r}^2z^2\Big\}^2\Big]$$

$$\leqslant 1+\Big(\frac{D-1}{D}\Big)q\hat{r}z+\frac{3}{2}\Big(\frac{D-1}{D}\Big)^2q\hat{r}^2z^2$$

$$\leqslant e^{\left(\frac{D-1}{D}\right)q\hat{r}z+\frac{3}{2}q\hat{r}^2z^2},$$

for $|z| \leqslant 1/(2\hat{r})$ and $D \leqslant 2$.

Phase Transition. As shown in (2.22), the performance of our algorithm is only bounded when the condition $T > 1$ is satisfied. Meanwhile, with $T < 1$, $\tilde{\sigma}_k^2$ which means the variance of the $\tilde{\mathbf{x}}_d^{(k)}$ diverges as the number of iteration k increases. In this case, our performance guarantee is no longer valid and the performance becomes worse compared to other algorithms such as EM and majority voting. Note that except for extreme case such as when using very low quality workers and the deficient assignments, $T > 1$ is easily satisfied and our performance guarantee is valid. In section 2.5, we will verify the existence of this critical point at $T = 1$ through several experiments with different conditions.

$$\frac{a}{c} = \frac{(D-1)^2}{(D^2 - D - 1)} q^2 \hat{l} \hat{r} = T.$$

2.5 Experiments

In this section, we verify the performance of the *multiple iterative algorithm* discussed in the previous sections with different sets of simulations. First, we check that the error of the iterative algorithm exhibits exponential decay as l increases or q increases. In addition, we show that our algorithm achieves a better performance than that of the majority voting and EM approach above a phase transition of $T = 1$. Next simulation investigates the linear relationship between y_j value and the ratio of $the\, number\, of\, correct\, answers$ to r_j for each worker. Then, we do experiments on the adaptive scenario by varying the proportion of pilot tasks and selected reliable workers. Finally, we do simulations on the experiments introduced above with a task set consisting of various D values.

Comparison with other algorithms. To show the competitiveness of our algorithm, we ran our *multiple iterative algorithm*, majority voting, and the EM approach for 2000 tasks and 2000 workers with fixed D = 2, 3, 4, and 5 (Figure 2.4 and Figure 2.5). The performance of the oracle estimator is also presented as a lower bound and the

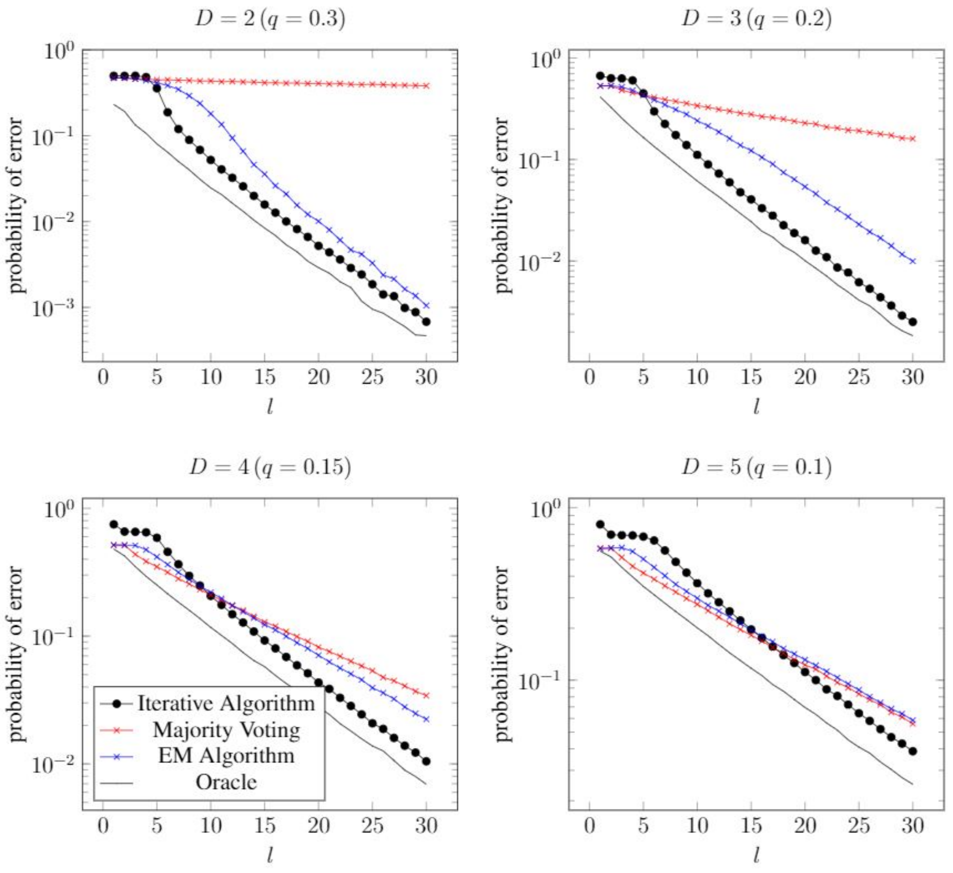


Figure 2.4: Comparisons of probabilities of error between different algorithms varying l values ($m = n = 2000$, $l = r$).

EM algorithm is implemented with Dawid and Skene's method [29]. In Figure 2.4, we can check that the probability of error decays exponentially as l increases, and is lower than that of the majority voting and EM approach above the phase transition $T = 1$. In addition, in Figure 2.5, we find the probabilities of error decays as q increases.

We expect a phase transition at $T = \frac{(D-1)^2}{(D^2-D-1)} q^2 \hat{l}\hat{r} = 1$ or $l = 1 + \sqrt{\frac{(D^2-D-1)}{(D-1)^2}}\frac{1}{q}$ when $l = r$ according to our theorem. With this, we can expect transitions to happen around $l = 4.33$ for $D = 2(q = 0.3)$, $l = 6.59$ for $D = 3(q = 0.2)$, $l = 8.37$ for $D = 4(q = 0.15)$, and $l = 11.89$ for $D = 5(q = 0.1)$. From the experiments in Figure 2.4, we see that iterative algorithm starts to perform better than majority vot-

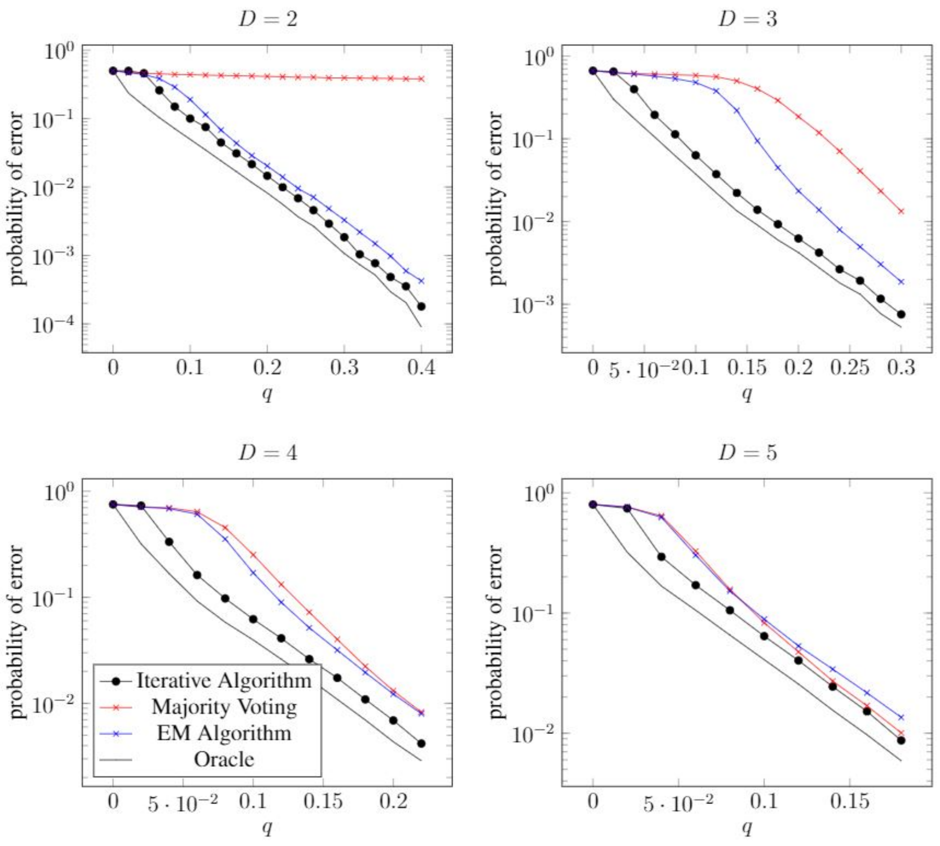


Figure 2.5: Comparisons of probabilities of error between different algorithms varying q values ($m = n = 2000$, $l = r = 25$).

ing around $l = 5,\ 6,\ 10,\ 18$ for each D. Note that these values are very similar with the theoretical values. It follows from the fact the error of our method increases with k when $T < 1$ as stated in Section 2.4. As can clearly be seen from the simulation results, we can check that the l values required for achieving $T > 1$ are not large. For example, if we consider dealing with short-answer questions like reCAPTCHA which is introduced in Section 2.3, carrying off the required $r(= l)$ is accomplished easily since each alphabet is considered as a separate question.

Adaptive Scenario. The inference of workers' relative reliability in the course of iterations is one of the algorithm's most important aspects. Now, we define $\hat{p_j}$ for each worker j as following:

$$\hat{p_j} = \frac{the\, number\, of\, correct\, answers}{r_j}.$$

After k_{max} iterations, we can find reliable workers by the value of worker message y_j since this value is proportional to $\hat{p_j}$, which is influenced by p_j. Relative reliability y_j is calculated by the following equation in Algorithm 1.

$$y_j \leftarrow \sum_{i \in \partial j} \left(\vec{A}^{ij} - \frac{\vec{1}}{D} \right) \cdot \vec{x}_{i \to j}^{(k_{max}-1)}$$

Figure 2.6 shows that there are strong correlations between y_j and $\hat{p_j}$. In one simulation, the correlation coefficients[1] between y_j and $\hat{p_j}$ are measured as 0.993, 0.989, 0.968, 0.938 for each $D = 2,\ 3,\ 4$, and 5, which are significantly large values. We can also check that the line passes approximately the point of $(\frac{1}{D}, 0)$, which represents a non-informative worker's reliability, as expected in Section 2.4.

One of the utilizations of this correlation property is the *adaptive scenario*, which extracts more reliable workers from the crowds after the inference of pilot tasks, and lets them solve the remaining tasks. We can improve the performance of our algorithm further with the scenario. The strategy consists of two steps in detail. In the first

[1]Pearson product-moment correlation coefficient(PPMCC) is used for evaluation.

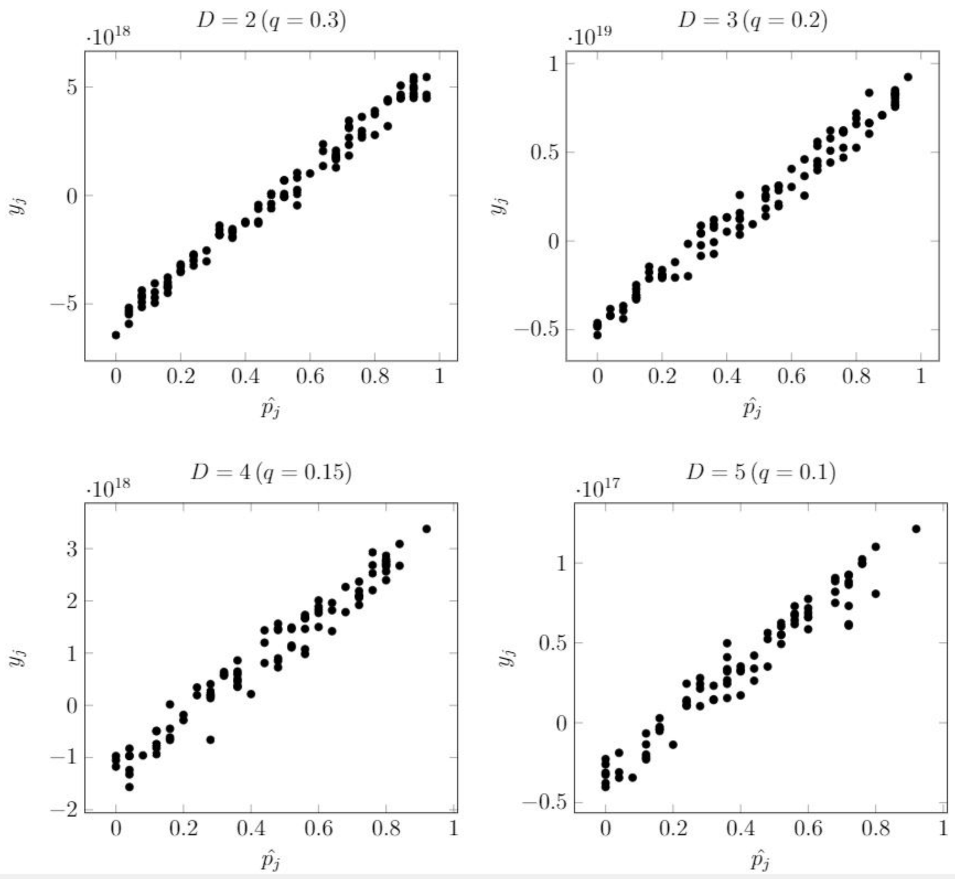


Figure 2.6: Relationship between y_j and $\hat{p_j}$ ($m = n = 2000$, $k = 10$).

step, we use $m' = \alpha m$ pilot tasks to infer the relative reliability of workers using the iterative algorithm.

$$m' = \alpha m, n' = n$$
$$l' = l, r' = \alpha r$$

In the second step, we select βn workers who have higher $|y_j|$ values after the first step, and each worker solves $\frac{m-m'}{\beta m} r$ tasks out of the remaining $m - m'$ tasks. We sort them out with higher $|y_j|$ values since we can gain less information from workers who have lower $|y_j|$ values, which means that their reliability is closer to $1/D$ than those of the others (Figure 2.6 and Figure 2.3).

$$m'' = m - m', n'' = \beta n$$
$$l'' = l, r'' = \frac{m - m'}{\beta m} r$$

To show the performance improvements when using the adaptive scenario, we perform experiments with several (m', β) sets. Figure 2.7 shows that the probability of error is smaller than for the non-adaptive scenario when proper m' and β are used. Specifically, as β decreases, the error tends to decrease since fewer, but more reliable, workers then solve the rest of the questions. However, we have to consider each worker's inherent capacity[2] when choosing an appropriate β. With limited capacity, we cannot use an unreasonably low β, since it places too high a burden on each worker. In addition, we have to take enough m' pilot tasks to guarantee the accuracy of the relative reliability, which are inferred in the first step.

Simulations on a set of various D values. To show the performance of the *generalized multiple iterative algorithm*, we do simulations on a task set consisting of various D values with Algorithm 2 . In detail, we repeat the same experiments with a question

[2]The number of possible questions that each worker can manage to solve in one transaction.

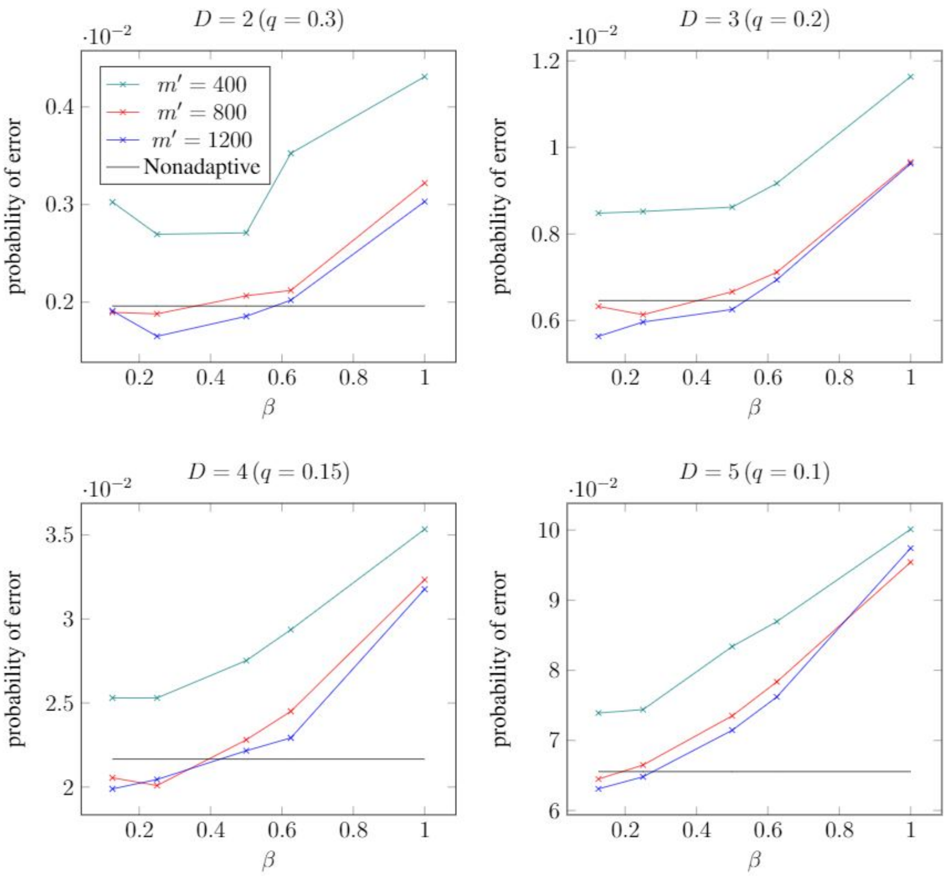


Figure 2.7: Adaptive Scenario ($m = n = 2000$, $l = 25$).

set composed in $1 : 1 : 1$ ratios of tasks which D are 2, 3, 4 respectively. Then, we have to investigate for the general case that q is calculated with the following equation.

$$q = \mathbb{E}[q_j] = \mathbb{E}\left[\left(\frac{D_i}{D_i - 1}\right)^2 \left(p_{ij} - \frac{1}{D_i}\right)^2\right]$$

We define q_j as an individual quality of the worker j. To perform simulations and to analyze the results, we have to make an assumption that a worker with individual quality q_j solves question with a reliability p_{ij} for each D_i. We can check that the same tendencies found in previous simulations also appear in Figure 2.8. There is also the strong correlation between y_j and $\hat{p_j}$ as 0.960. This result is notable in that in the real world, there are many more cases where questions have varying number of choices than fixed ones.

2.6 Related Literature

A common, intuitive strategy for aggregating responses is majority voting, which is widely used in real life due to its simplicity. However, in crowdsourcing systems, this simple inference technique has several limitations, since it assumes all workers have an equal level of expertise, and it gives the same weight to all responses. In general, there are unreliable workers such as novices or free money collectors, and even adversarial workers can be shown, so majority voting has obvious weak points when workers are unreliable [2].

There have been various approaches to trying to improve the reliability of results from unreliable responses. Two key ideas are introducing latent variables and estimating results by an iterative algorithm known as the EM algorithm. Dawid and Skene [29] exploited these ideas when they developed a simple probabilistic model using confusion matrices for each labeler as latent variables. They proposed an iterative algorithm based on EM to infer ground truth from unreliable responses.

Since the EM algorithm has an effective procedure to evaluate missing or hidden

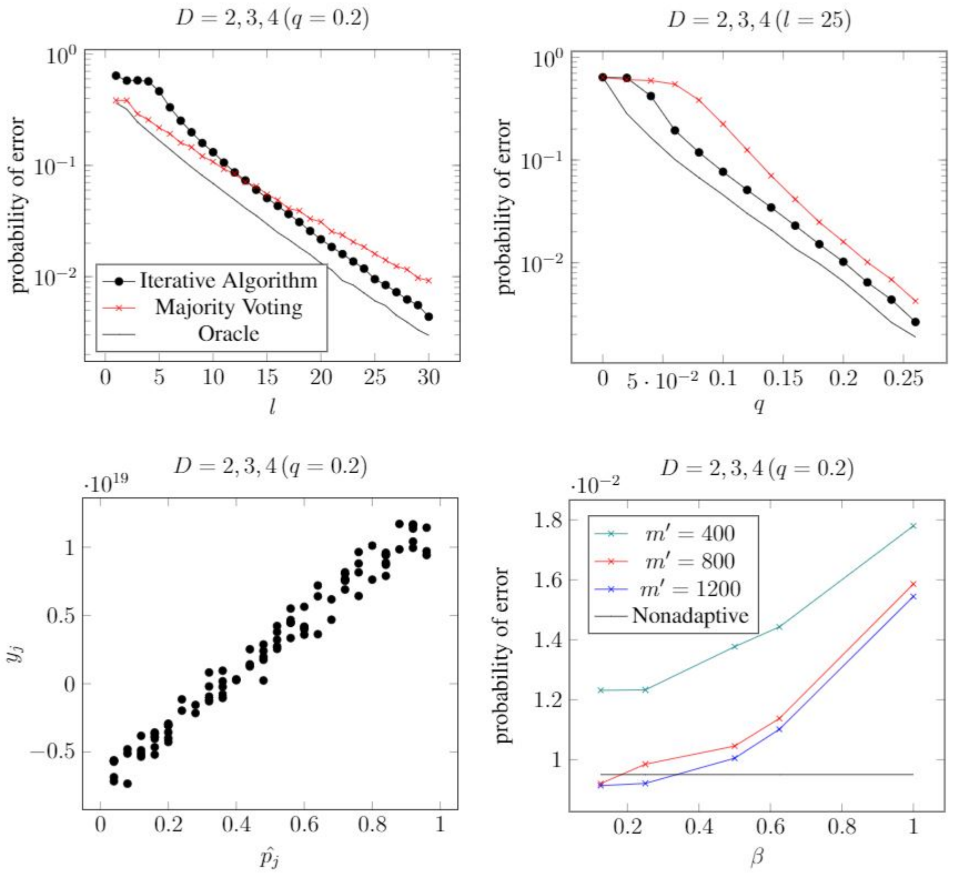


Figure 2.8: Simulations on a set of various D values ($m = n = 2000\,(D = 2 : 666\,/\,D = 3 : 667\,/\,D = 4 : 668)$).

data and performs quite well, this model has been generalized and extended by several researchers. The GLAD model [20] combines the implicit characteristics of tasks and workers. Responses from workers are determined by several factors, such as the difficulty of the task, the expertise of the labeler, and the true label. The EM-based model can operate flexibly on various cases by introducing extra latent variables, which can be represented as the natural properties of tasks and workers [19]. Another variant proposed by Raykar et al. [18] considers a proper classifier for crowdsourcing systems, and aims to learn the classifier and the ground truth together.

Despite its popularity, there are some arguments in existing EM algorithms. The main thing is lack of intensive analysis about performance guarantees since their performance is only empirically evaluated in most cases. Another point is that inference techniques based on EM algorithms are not scalable. If the data size increases, EM-based algorithms become inefficient and degenerate, because their time and space requirements grow exponentially. Moreover, designing model-based EM algorithms with greater complexity leads to the introduction of an increased number of latent variables and model parameters. Apart from the computational complexity problem, the performance of EM-based algorithms could degenerate due to the initialization problem, even though it is designed to be a more complex model.
Alternative approaches have been suggested by Karger et al. [21] in the context of spectral methods that use low-rank matrix approximations. They treated the data matrix A which involves workers' responses perturbed by a random noise. The true answers can be approximated by a rank-1 matrix, of which the singular vector reflects the correct answer of the tasks. When the spectral radius of the signal matrix outweighs the spectral radius of the random noise matrix, the correct answers can be extracted by the singular vector of the data matrix A. Using the power iteration method, the top singular vector can be obtained more efficiently compared to the computation complexity of EM-based algorithms.

They also proposed a novel iterative learning algorithm [22] that learns the likelihood of candidate answers and the reliability of workers. It is inspired by the standard *Belief Propagation* (BP) algorithm, which approximates the maximal marginal distribution of variables. This message passing algorithm achieves almost the same results as the previous spectral method, but they provide novel analysis techniques such as *Density Evolution* in coding theory to improve the error bound more tightly, which decays exponentially. Although they did not assume any prior knowledge, Liu et al. [33] shows that choosing a suitable prior can improve the performance via a Bayesian approach.

Recently, Karger et al. [34] focused on multi-class labeling based on their existing novel algorithms, but their strategy for multi-class labeling is well suited to the linearly ordered choices, not independent multiple choices. By converting each multiple-choice question into a bunch of binary-choice questions, they could exploit the existing algorithms to determine true answers of multiple-choice questions. Although this strategy can be extended to independent multiple choices, it overexploits redundancy since each task should be split and queried in multiple times to obtain reliable results. Furthermore, in real crowdsourcing systems, it is natural that workers solve intact multiple-choice questions rather than split binary-choice questions. Therefore it has difficulty in combining into real crowdsourcing systems.

On top of the problem inferring the true answers, proper adaptive strategies are developed to utilize reliable workers when they are reusable. [35, 36, 37, 3] showed that the performance can be significantly improved through exploration/exploitation approaches.

Chapter 3

Reliable Aggregation Method for Vector Regression in Crowdsourcing

3.1 Preliminaries

In this section, we describe a problem setup with variables and notations. First, we assume that there are m tasks in total and each task i is assigned to distinct l_i workers. Similarly, there are n workers in total and each worker j solves different r_j tasks. Here and after, we use $[N]$ to denote the set of first N integers. If we regard tasks and workers as set of vertices and connect the edge $(i, j) \in E$ when the task i is assigned to the worker j, our system can be described as a bipartite graph $G = \{[m], [n], E\}$ in Figure 3.1.

Our crowdsourcing system considers a specific type of task whose answer space spans a finite continuous domain. If a task asks D number of real values, a response $\tilde{\boldsymbol{A}}$ is a D-dimensional vector. On one task node i, given all of responses $\big\{\tilde{\boldsymbol{A}}_{ij}|(i, j) \in E\big\}$, we transform them to $\boldsymbol{A}$ subject to $\|\boldsymbol{A}_{ij}\|_1 = 1$ by the min-max normalization since each task can have a different domain length.

For a simple example, in an image object localization regression task, a response is a bounding box to capture the target object. Considering the x axis only for brevity, the box coordinate is $\tilde{\boldsymbol{A}} = [x_{tl}, x_{br}]$, where x_{tl} and x_{br} stand for the top-left and bottom-right coordinates. Then it can be transformed as

$$\boldsymbol{A} = \big(x_{tl}, x_{br} - x_{tl}, x_{max} - x_{br}\big)/x_{max},$$

where x_{max} represents the width of the image. Since images have different size of width and height, all responses are transformed to have the same domain length.

In summary, when the worker j solves the task i, the response is denoted as $\tilde{\boldsymbol{A}}_{ij} \in \mathbb{R}^D$ and transformed to $\boldsymbol{A}_{ij} \in \mathbb{R}^{D+1}$ with respect to $\|\boldsymbol{A}_{ij}\|_1 = 1$. For convenience, δ_i and δ_j denotes the group of workers who give responses to the task i and the group of tasks which are assigned to worker j respectively.

Majority Voting ($\mathcal{MV}$). The simplest method in response aggregation is majority voting, well-known sub-optimal estimator, which computes the centroid of responses. However, its performance can be easily degraded whether there exist a few adversarial workers or spammers who give wrong answers intentionally or random answers respectively.

Majority voting method gives the identical weight to every worker who annotates the task for fixed task i.

$$\hat{\boldsymbol{t}}_i^{(\mathcal{MV})} = \sum_{j \in \delta_i} \frac{1}{l_i} \boldsymbol{A}_{ij}. \tag{3.1}$$

3.2 Inference Algorithm

In this section, we propose a message-passing algorithm for vector regression tasks. Our iterative algorithm alternatively estimates two types of messages: (1) task messages $\boldsymbol{x}_{i\to j}$, and worker messages $\boldsymbol{y}_{j\to i}$. This updating process estimates the ground truth of each task and the reliability of each worker respectively. From now on, $\hat{l_i}$ and $\hat{r_j}$ denote $(l_i - 1)$ and $(r_j - 1)$ respectively for brevity.

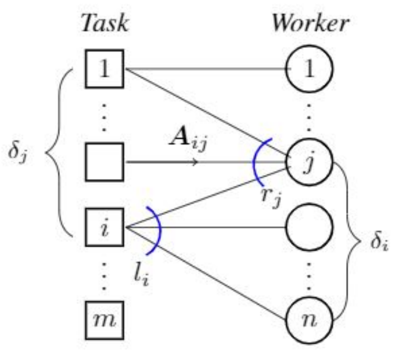
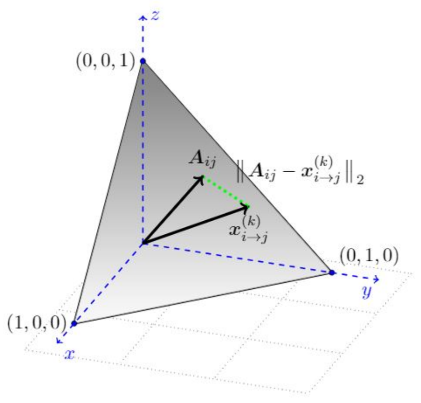


Figure 3.1: System model for task-worker assignments.



z
$(0,0,1)$
$\boldsymbol{A}_{ij}$
$\|\boldsymbol{A}_{ij} - \boldsymbol{x}^{(k)}_{i\to j}\|_2$
$\boldsymbol{x}^{(k)}_{i\to j}$
$(0,1,0)$
y
$(1,0,0)$
x

Figure 3.2: Distance between answer $\boldsymbol{A}_{ij}$ and x message $\boldsymbol{x}_{i\to j}$ in the standard 2-simplex space when $D_i = 2$.

3.2.1 Task Message

We first describe a task message that estimates the current candidate of a ground truth. It simply computes the centroid of weighted responses from the workers assigned to the task. Thus, it can be viewed as a simple estimator of weighted voting in that those weights are computed according to how workers are reliable. Note that a task message $\boldsymbol{x}_{i\to j}$ averages weighted responses from workers assigned to a task i except for the response from worker j. This helps to block any correlation between the task message and the responses from worker j.

$$\boldsymbol{x}^{(k)}_{i\to j} = \sum_{j'\in\delta_i\backslash j} \left(\frac{y^{(k-1)}_{j'\to i}}{y^{(k-1)}_{\delta_i\backslash j}} \right) \boldsymbol{A}_{ij'}, \tag{3.2}$$

where $y^{(k-1)}_{\delta_i\backslash j} = \sum_{j'\in\delta_i\backslash j} y^{(k-1)}_{j'\to i}$.

3.2.2 Worker Message

The next step is to compute worker messages $y_{j\to i}$ which represents the importance of response $\boldsymbol{A}_{ij}$. These worker messages are used as weights in the weighted voting pro-

Algorithm 1 Inference Algorithm

Input: $G = \{[m], [n], E\}$, $\{\boldsymbol{A}_{ij}\}_{(i,j)\in E}$, k_{max}

Output: Estimated truths $\hat{t}_i$, $^\forall i \in [m]$

```
 1: Initialization
 2: for $^\forall(i,j) \in E$ do
 3:     $y^{(0)}_{j\to i} \leftarrow \mathcal{N}(0,1)$
 4: Iteration Step
 5: for $k = 1$ to $k_{max}$ do
 6:     for $^\forall(i,j) \in E$ do
 7:         Update task message, $\boldsymbol{x}^{(k)}_{i\to j}$ using Eq. 3.2
 8:     for $^\forall(i,j) \in E$ do
 9:         Update worker message, $y^{(k)}_{j\to i}$ using Eq. 3.3
10: end for
11: Final Estimation
12:     for $^\forall j \in [n]$ do
13:         $y_j \leftarrow \left(\frac{1}{r_j}\sum_{i\in\delta_j}(\|\boldsymbol{A}_{ij} - \boldsymbol{x}^{(k_{max})}_{i\to j}\|_2)^2\right)^{-1}$
14:     for $^\forall i \in [m]$ do
15:         $\boldsymbol{x}_i \leftarrow \sum_{j\in\delta_i}\left(\frac{y_j}{y_{\delta_i}}\right)\boldsymbol{A}_{ij}$
16: return $\hat{t}^{(\mathcal{ALG})}_i \leftarrow \boldsymbol{x}_i$, $^\forall i \in [m]$
```

cess in task messages update. Since it is desirable to give a higher weight to more reliable workers, each worker's reliability should be evaluated as the similarity between his response and the task message which indicates the consensus of other workers' responses. In our algorithms, it takes advantage of the reciprocal of the summation of the euclidean distance between the response and the task message as a similarity measure. In analysis section, our analysis verify that this measure is proper to estimate weights of workers' responses. Note that a worker message $y_{j\to i}$ represents the average of similarities between worker j's responses and the average response of other workers' responses in the same task.

$$y^{(k)}_{j\to i} = \left(\frac{1}{\hat{r_j}} \sum_{i'\in\delta_j\backslash i} \left(\|\boldsymbol{A}_{i'j} - \boldsymbol{x}^{(k)}_{i'\to j}\|_2)^2\right)\right)^{-1}. \tag{3.3}$$

In the worker message update (3.3), we adopt the reciprocal of ℓ_2 norm in the vector space as a similarity measure. However, our algorithm can be generalized with any metric induced by other norm and similarity function which is continuous and monotonically decreasing.

3.3 Experimental Results

In our experiments, we have evaluated the performance of our algorithm with two popular benchmarks, MSCOCO [38] and the Leeds Sports Pose Extended Training (LSPET) datasets. We compare our algorithm with baselines algorithms which are majority voting ($\mathcal{MV}$) and weighted voting ($\mathcal{WV}$) whose weights are externally given by web-based crowdsourcing platform. We also implemented several state-of-the art which are inner-product method ($\mathcal{IP}$) [12], Welinder's EM model [39], DALE model [40], and outlier rejection methods which are *Mean shift* and *Top-K* selection.

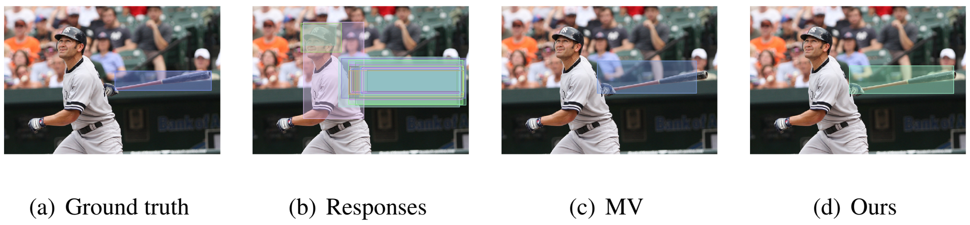


Figure 3.3: Drawing a bounding box task on the ‘bat’. (a) the ground truth (b) bounding boxes drawn by 25 workers. (c) Estimated answer of majority voting. (d) Estimated answer of our algorithm.

3.3.1 Real crowdsourcing data

We crowdsourced two types of tasks in CrowdFlower. One is for image object localization in which the task is to draw a bounding box on the specified object as tightly as possible. The other one is for human pose estimation, where the task is to construct a skeleton-like structure of a human in a given image.

Bounding box on MSCOCO dataset. In this task, we randomly chose 2,000 arbitrary images from MSCOCO dataset, and each image was distributed to 25 distinct workers, so there were 50,000 tasks to be solved in total. Total 618 workers were employed, and each worker solved 10 (min) to 100 (max) tasks. We exclude some invalid responses (no box, box over out of bounds [0, image size]). Note that a general bipartite graph is created with different node degrees l_i and r_j, which is not a regular bipartite graph. We measured algorithms' performances by the average error in the ℓ_2 norm and the Intersection over Union (IoU), which is another standard measure for object localization computed by a ratio of intersection area to union area of two bounding boxes. In this experiment, DALE model does not converge due to its complex graphical model raising an out of memory error.

To measure the performance of DALE model in smaller data, we collected a dedicated dataset of 100 images each of which was assigned to 20 distinct workers. Results are listed in Table 3.1 with two evaluation metric Euclidean distance(ℓ_2) and Intersec-

Dataset	MSCOCO		LSPET	
Type	Box(ℓ_2)	Box(IoU)	Joints	Angles
$\mathcal{WV}$	0.22227	0.89593	0.15877	0.10524
$\mathcal{MV}$	0.22090	0.89666	0.15858	0.10462
$\mathcal{IP}$	0.22026	0.89712	0.15483	0.10462
Welinder	0.21886	0.89821	N/A	N/A
DALE	0.21834	0.89914	N/A	N/A
Top-K	0.18869	0.91250	0.12222	0.10051
MeanShift	0.18034	0.92150	0.11812	0.09962
Ours	**0.14837**	**0.93445**	**0.09308**	**0.09941**

Table 3.1: An error table of experimental results on real crowdsourced data where the tasks are (1st column) an object detection on MSCOCO dataset, (2nd column) same task with Intersection of Union measure (3rd column) a human joints estimation and (4th column) an angle segmentation by neck and adjacent human joints on LSPET dataset. For Top-K selection, we choose K as a half of the task degree l.

tion over Union(IoU). Our algorithm significantly outperforms others and, even with small number of iterations, can reduce errors rapidly. Empirically, our algorithm converges in less than 20 iterations as plotted in Figure 3.5.

Varying degree on MSCOCO dataset. Here we show how the performances of different algorithms vary with task degree l. We made a number of task-worker bipartite graphs by randomly dropping some edges to make degree l for each task. As expected, the average error of each algorithm decreases as the task degree l increases. Even when the degree value falls until 5, ours can still keep the large gap among other algorithms. In other words, our algorithm needs less budget to get same error rate. The results are listed in Figure 3.4.

Robustness. Since it is well known that message-passing algorithms suffers from the

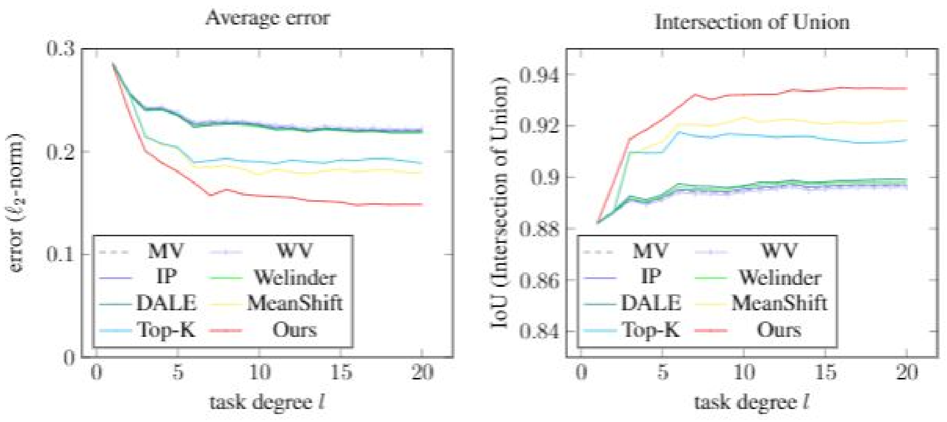


Figure 3.4: Comparisons of error and IoU between different algorithms with varying task degree l.

hyper	α	β	μ	σ
$\mathcal{D}$	$\mathcal{U}(0, 0.5)$	$\mathcal{U}(0, 0.5)$	$\mathcal{U}(0.25, 1.25)$	$\mathcal{U}(0.5, 2)$

Table 3.2: Hyperparameters for initialization α, β of *beta* distribution and μ, σ of *Gaussian* distribution.

initialization issue in general, we tested robustness of our algorithm by initializing workers' weights to be sampled from proper distributions with moderate hyperparameters. Here we used *Beta* distribution with (α, β), and *Gaussian* distribution with (μ, σ^2) sampled from uniform distribution $\mathcal{U}$. The result is shown by error bar plots in Figure 3.5 which represents the deviation reduces rapidly. This result shows that our algorithm is robust to the initialization of workers' weights.

When the number of edges are not sufficient to estimate worker message, our algorithm can diverge as iteration progresses since worker message is computed by the reciprocal of the summation between the response and the task message. It can be resolved by adding a very small positive constant ϵ on the summation before computing

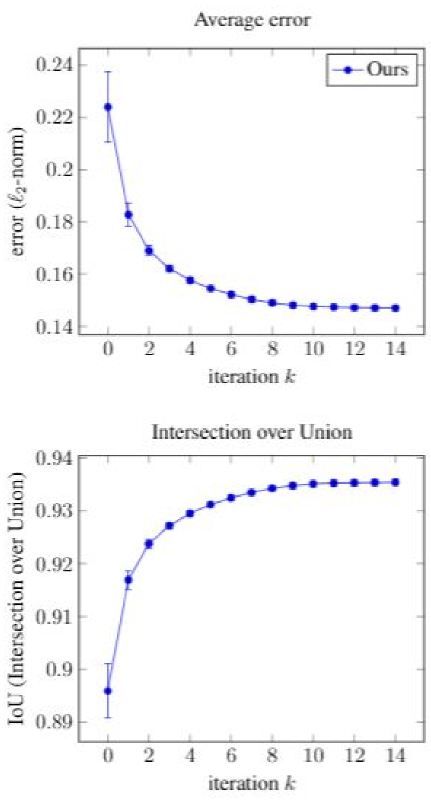


Figure 3.5: Error bar plots of our algorithm for the initialization issue on 2k-edge bounding box task.

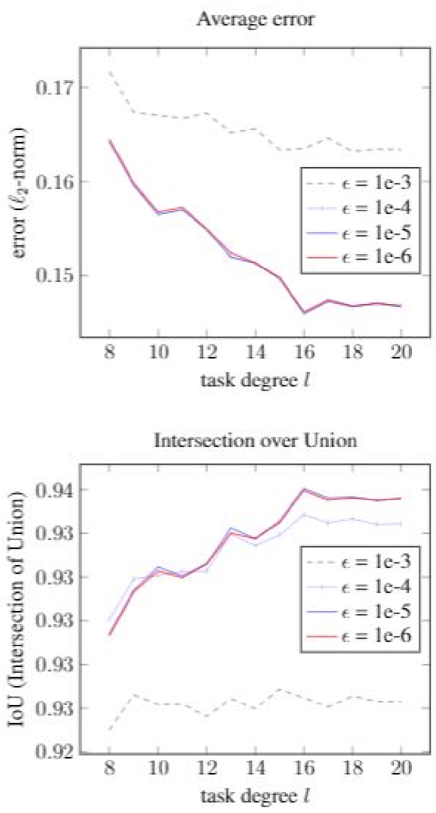


Figure 3.6: The influence of ϵ on error and IoU when computing y-messages with varying task degree l.

the reciprocal.

$$y_{j\to i}^{(k)} = \left(\frac{1}{\hat{r}} \sum_{i' \in \delta_{j \setminus i}} (\|\boldsymbol{A}_{ij} - \boldsymbol{x}_{i\to j}^{(k)}\|_2)^2) + \epsilon\right)^{-1}. \tag{3.4}$$

We investigate the influence of ϵ in Figure 3.6. This result shows our algorithm works well when $\epsilon \le 10^{-5}$.

Human pose estimation. We collected the human pose estimation data of 1,000 images chosen from LSPET dataset using CrowdFlower platform. Each image was distributed to ten distinct workers who were asked to mark dots on the 14 human joints

Discretization	10	10^3	10^5	Ours
Accuracy	0.2395	0.1493	0.1492	0.1483

Table 3.3: Comparison with discretization method [12]

(head, neck, left/right shoulders, elbows, wrists, hips, knees, and ankles). In this experiment, we aggregated their answers to estimate the point of each human joint. Moreover, we estimated angles from the neck and adjacent joints (head, shoulders, hips) as another task which is also important in pose estimation. Estimating angles can be viewed as dividing angle task whose domain is $[0, 2\pi]$. As shown in Table 3.1, our algorithm outperforms others on both joint and angle estimation tasks.

Iteration	$\mathcal{MV}$(0)	1	2	4	8
Accuracy	0.221	0.182	0.169	0.158	0.149

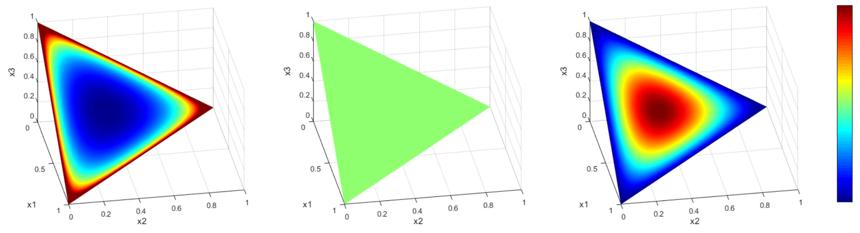
Table 3.4: The ℓ_2 error performance of our algorithm based on iteration number in a 2k-edge bounding box task.

3.4 Performance Analysis

In this section, we analyze the average performance of our algorithm using a probabilistic crowd model. Although our algorithm works in a general bipartite graph, here we assume a regular bipartite graph (i.e. $^\forall i, j, l_i = l$ and $r_j = r$ solely for a mathematical proof which were commonly used in previous works [22, 12]. This is an ordinary setting when we intend to assign our budget to each task equally. If the crowdsourcing system creates every batch arbitrarily, our setting can be represented as a random (l, r)-regular bipartite graph. To generate a random regular bipartite graph, we bring a simple construction model known as the *configuration model* [41, 42].

3.4.1 Dirichlet crowd model

Here we propose Dirichlet crowd model rather than multivariate Gaussian one. The former is more appropriate to capture all types of worker's behaviors in a simplex domain, while the latter is unable to represent *Spammer*'s behavior which give an arbitrary answer. In addition, the latter can occasionally generate invalid responses since it spans infinite continuous domain and does not have an order of selection among its dimensions.



(a) Adversarial (b) Spammer (c) Hammer

Figure 3.7: Three types of crowds in standard 2-simplex space.

For given ground truths on the standard D-simplex, we can consider each response as a sample from a Dirichlet distribution governed by corresponding task and worker. Since responses are commonly drawn around the ground truth (if worker is reliable enough), we can assume the mean of this Dirichlet distribution as the task's ground truth t_i. The expected error between the ground truth and the response then comes from its variance, which is determined by the location of ground truth $\boldsymbol{t_i}$ and the worker's reliability w_j. The expected error in responses is determined by the worker reliability which measures the ability, diligence, and precision of the worker. We parametrize this as a positive continuous value $w_j \in [0, \infty]$ where a higher w_j means that worker j gives a closer answer to ground truth.

Now we define the average quality of workers as

$$q^{-1} = \mathbb{E}_{\mathbb{W}}\Big[\frac{1}{\mathrm{w}+1}\Big]. \tag{3.5}$$

A higher value of q indicates that the group of workers are more reliable, whereas a value q close to one means most workers behave maliciously. We will show that the error bound of our algorithm depends on this parameter q. The expected error of our algorithm is defined as

$$\mathrm{E}_{\mathcal{ALG}} := \lim_{k_{\max}\to\infty} \frac{1}{m} \sum_{i\in[m]} \mathbb{E}\Big[\Big(\|\boldsymbol{t}_i - \hat{\boldsymbol{t}}_i^{(\mathcal{ALG})}\|_2\Big)^2\Big]. \tag{3.6}$$

where $\boldsymbol{t}_i$ is the ground truth of task i.

3.4.2 Error Bound

Theorem 2 *For fixed $l > 1$, $r > 1$ and dimension $D \geqslant 1$, assume that m tasks are assigned to n workers according to a random (l, r)-regular bipartite graph. If the average quality satisfies $q > (1 + (D+1)/\hat{l}\hat{r})$, then when $k \to \infty$ the average error of the our algorithm achieves*

$$\mathrm{E}_{\mathcal{ALG}} \leqslant \left(\frac{(1+1/\hat{l}\hat{r})^2}{(\sqrt{2}+1)q\hat{r}}\right) \cdot \frac{1}{\hat{l}m} \sum_{i\in[m]} T_i. \tag{3.7}$$

First of all, the above mild condition $q > (1 + (D+1)/\hat{l}\hat{r})$ is easy to satisfy. If this condition satisfies, the average error of ours is upper-bounded. Also, higher lr makes the quality condition weaker. The sum of T_i is the small constant which reflects the prior of ground truths. Thus, we can control the error performance by adjusting the average quality of workers and the number of queries. As q and lr increase, the upper bound of our algorithm becomes lower.

Proof sketch. Here, we briefly describe the proof precedure of Theorem 1. At first, We consider any worker distribution with the average quality q. Under this worker distribution, our strategy is to inspect the average behavior of worker messages, $\mathbb{E}\big[y_{j\to i}^{(\infty)}\big]$ as $k_{max} \to \infty$.

$$\left\{\mathbb{W}|q^{-1} = \mathbb{E}_{\mathbb{W}}\left[\frac{1}{\mathrm{w}+1}\right]\right\}.$$

In task message update step, our algorithm normalizes the worker message values in order to make each task message on the simplex. Since we assume the sparse bipartite graph, the task-worker graph G is locally tree-like. Therefore, the messages and responses are uncorrelated with high probability.

Lemma 1 *For each task i, worker j, If the average quality satisfies $q > (1 + (D + 1)/\hat{l}\hat{r})$, then when $k \to \infty$ the expected worker message of our algorithm converges such that*

$$\mathbb{E}\left[y_{j\to i}^{(\infty)}\right] \cong \frac{1}{\frac{1}{(w_j+1)} + \frac{1}{\hat{l}\hat{r}q}} \cdot T_i. \tag{3.8}$$

This result implies our algorithm gradually gives worker j a weight proportional to $(w_j + 1)$ like the oracle estimator as degree l or the worker quality q grows. In the other extreme case, the our algorithm gives the constant weight to each worker like the majority voting (i.e., $(w_j + 1) \gg lq$). In the next part, we will prove that the oracle estimator assigns a worker j the optimal weight in proportion to $(w_j + 1)$. Thus, the average error increases as the KL divergence between the vector of weights and that of oracle estimator's weight increases. Since the weights of workers who annotate the task i are normalized by their sum, we can determine those weights precisely in terms of their ratios mentioned in *Lemma 1*. Then, we can verify that Algorithm 1 is gradually getting closer to oracle estimator as $k \to \infty$. In initialization, since Algorithm 1 gives same weight to each worker, it behaves as the majority voting. After a few iterations, each weight vector follows (3.8) similar to oracle estimator which is the theoretical lower bound in ℓ_2 norm. Detailed proof of *Lemma 1* will be omitted here but the whole process of the proof is provided in 3.4.4

Oracle Estimator ($\mathcal{OC}$). Under a probabilistic worker model, we can construct theoretically the optimal inference algorithm as obtained by an oracle who knows every worker's reliability. Given workers' reliabilities, the oracle can estimate the ground truths which minimize the expected error. Therefore its performance can be considered as a lower bound of the expected error rate.

$$\mathrm{E}_{\mathcal{OC}} = \left(\frac{1}{\mathbb{E}_{\mathbb{W}}\left[\mathrm{w}+1\right]} \right) \cdot \frac{1}{lm} \sum_{i \in [m]} T_i. \tag{3.9}$$

Then, we look into the expected error of majority voting method. It gives the identical weight for each worker. Thus, the expected error of majority voting is simply represented as

$$\mathrm{E}_{\mathcal{MV}} = \left(\frac{1}{qlm} \right) \sum_{i \in [m]} T_i. \tag{3.10}$$

From above results, we always have

$$\mathrm{E}_{\mathcal{OC}} \leqslant \mathrm{E}_{\mathcal{MV}} \quad (\because \textit{Jensen's inequality}), \tag{3.11}$$

and the equality holds when the distribution of worker reliability $\mathbb{W}$ follows a degenerate distribution.

Then we compare the upper bound of our algorithm with the performance of the oracle estimator. To show concisely, the mean and variance of the reliability distribution are denoted as μ_w and σ_w^2 respectively.

Corollary 1 *Under the hypotheses of Theorem 1, if the distribution of the reliability satisfies*
$\mathbb{P}\Big((\mathrm{w}+1) \geqslant 2\mu_w\Big) \leqslant \frac{(\sqrt{2}+1)\hat{l}\hat{r}}{l(1+1/(\hat{l}\hat{r}))^2}$ *and symmetrical, then the upper bound of* $\mathrm{E}_{\mathcal{ALG}}$ *can be approximated as*

$$\mathrm{U}_{\mathcal{ALG}} \rightarrow \mathrm{E}_{\mathcal{OC}}. \tag{3.12}$$

This result shows that if the reliability distribution satisfy above condition, the upper bound of our algorithm is close to the oracle estimator.

3.4.3 Optimality of Oracle Estimator

In this part, we prove why the oracle estimator is optimal and it gives worker j the weight in proportion to $(w_j + 1)$. The result of (9) implies that the expected error of oracle estimator is weighted average of $(w_j+1)^{-1}$ excluding the effect of the ground truth. For fixed task i and given l number of worker batch δ_i, the problem of finding the oracle estimator's optimal weights is formulated as the following convex optimization problem. The object function $G(\boldsymbol{v})$ represents the sum of expected errors from workers in the batch δ_i.

$$\begin{aligned}\underset{v}{\text{minimize}} \quad & G(\boldsymbol{v}) = \sum_{j=1}^{l} \frac{v_j^2}{w_j+1} \\ \text{subject to} \quad & \sum_{j=1}^{l} v_j = \boldsymbol{v}^T\mathbf{1} = 1. \quad v_j \geqslant 0, \ ^\forall j\end{aligned}$$

This problem is equivalent to minimize a l-dimensional ellipsoid's radius with a constraint on the subspace $\boldsymbol{v}^T\mathbf{1}=1$. The optimal point $\boldsymbol{v}^*$ is on the point of contact between the l-dimensional ellipsoid and the subspace. Thus, we can obtain $\nabla G(\boldsymbol{v}^*) = b\mathbf{1}$ for some constant b and $(\boldsymbol{v}^*)^T\mathbf{1} = 1$. In addition, the problem is also convex problem with strong duality (zero dual gap). Therefore, points are optimal if and only if they satisfy the KKT condition:

$$\boldsymbol{v}_j^* \geqslant 0, \ ^\forall j, \ \boldsymbol{v}^T\mathbf{1} = 1,$$

$$\psi_j \geqslant 0, \ \psi_j v_j = 0 \ ^\forall j,$$

$$\nabla H(\boldsymbol{v}) = 0.$$

Once you solve the above equations, we obtain the optimal weight which the oracle estimator gives to worker j. In fact, the optimal weight of the worker j is in proportion to (w_j+1).

$$\boldsymbol{v}_j^* = \frac{(w_j+1)}{\sum_{j=1}^{l}(w_j+1)}. \quad \square$$

3.4.4 Performance Proofs

Proof sketch. Here, we briefly describe the proof precedure of Theorem 1. At first, We consider any worker distribution with the average quality q. Under this worker distribution, our strategy is to inspect the average behavior of worker messages, $\mathbb{E}\big[y_{j\to i}^{(\infty)}\big]$ as $k_{max} \to \infty$.

$$\left\{\mathbb{W}|q^{-1} = \mathbb{E}_{\mathbb{W}}\left[\frac{1}{\mathrm{w}+1}\right]\right\}.$$

In task message update step, our algorithm normalizes the worker message values in order to make each task message on the simplex. Since we assume the sparse bipartite graph, the task-worker graph G is locally tree-like. Therefore, the messages and responses are uncorrelated with high probability.

Lemma 2 *For each task i, worker j, If the average quality satisfies* $q > (1 + (D + 1)/\hat{l}\hat{r})$*, then when* $k \to \infty$ *the expected worker message of our algorithm converges such that*

$$\mathbb{E}\left[y_{j\to i}^{(\infty)}\right] \cong \frac{1}{\frac{1}{(w_j+1)} + \frac{1}{\hat{l}\hat{r}q}} \cdot T_i. \tag{3.13}$$

This result implies our algorithm gradually gives worker j a weight proportional to $(w_j + 1)$ like the oracle estimator as degree l or the worker quality q grows. In the other extreme case, the our algorithm gives the constant weight to each worker like the majority voting (i.e., $(w_j + 1) \gg lq$). In the next part, we will prove that the oracle estimator assigns a worker j the optimal weight in proportion to $(w_j + 1)$. Thus, the average error increases as the KL divergence between the vector of weights and that of oracle estimator's weight increases. Since the weights of workers who annotate the task i are normalized by their sum, we can determine those weights precisely in terms of their ratios mentioned in *Lemma 2*. Then, we can verify that Algorithm 1 is gradually getting closer to oracle estimator as $k \to \infty$. In initialization, since Algorithm 1 gives same weight to each worker, it behaves as the majority voting. After a few iterations, each weight vector follows (3.13) similar to oracle estimator which is the theoretical lower bound in ℓ_2 norm. Detailed proof of *Lemma 1* will be omitted here but the whole process of the proof is provided in the supplemental material.

Proof of Lemma 1

The average value of worker message converges proportionally to $\big((w_j+1)^{-1} + (lq)^{-1}\big)^{-1}$ as $k \to \infty$ in *Lemma 1*. For the proof, we describe a distance of responses in advance. Since we adopt the euclidean distance as a metric in Section 2, the distance between the single response $\boldsymbol{A}_{ij}$ and the task message $\boldsymbol{x}_{i\to j}^{(k)}$ is denoted as

$$d_{j\to i}^{(k)} = (\|\boldsymbol{A}_{ij} - \boldsymbol{x}_{i\to j}^{(k)}\|_2)^2.$$

Then, the expected distance is represented as

$$\mathbb{E}\big[d_{j\to i}^{(k)}\big] = \Bigg(\frac{1}{(w_j+1)} + \underbrace{\sum_{j'\in\delta_i\backslash j} \mathbb{E}\Bigg[\Bigg(\frac{y_{j'\to i}^{(k-1)}}{y_{\delta_i\backslash j'}^{(k-1)}}\Bigg)^2\Bigg]\frac{1}{(w_{j'}+1)}}_{p_{ij}^k}\Bigg)T_i.$$

Since our algorithm initializes each worker message value equally and we assume average quality satisfies the condition, $q > (1 + D/\hat{l}\hat{r})$, p_{ij}^k value decreases as $k \to \infty$. In particular,

$$\mathbb{E}\Bigg[\frac{1}{\hat{r}}\sum_{i'\in\delta_j\backslash i} p_{ij}^k\Bigg] \leqslant \frac{1}{\hat{l}\hat{r}q}$$

On the other hand, the estimation process of worker messages is described as

$$y_{j\to i}^{(k)} = \frac{1}{\frac{1}{\hat{r}}\sum_{i'\in\delta_j\backslash i}\big(d_{j\to i'}^{(k)}\big)}. \tag{3.14}$$

In section C, we will prove that the oracle estimator gives each worker j the optimal weight in proportion to (w_j+1). The expected distance and (3.14) implies that our algorithm gives each worker j similar weight but not equal. Thus, we can obtain the expected worker message after k iteration.

$$\mathbb{E}[y_{j\to i}^{(k)}] \cong \frac{1}{\frac{1}{\hat{r}}\sum_{i'\in\delta_j\backslash i}\mathbb{E}[d_{j\to i'}^{(k)}]}. \tag{3.15}$$

Therefore, as $k \to \infty$, the expected worker message satisfies

$$\mathbb{E}\Big[y_{j\to i}^{(\infty)}\Big] \cong \frac{1}{\frac{1}{(w_j+1)} + \frac{1}{\hat{l}\hat{r}q}} \cdot T_i. \tag{3.16}$$

Upper bound

Here the detailed proof of *Theorem 1* is provided using the result o f *Lemma 1*. In Section 2, we adopt a random (l, r)-regular bipartite graph as a task assignment. If this sparse task-worker graph is assumed ($|m| \gg r$, $|n| \gg l$), we can claim that the graph has locally-tree like structure. This property implies that the response A_{ij} is uncorrelated with the task message $x^{(k)}_{i\to j}$. Then, for each task i, the expected error of our algorithm $\mathrm{E}_{\mathrm{iter}}$ is represented as

$$\mathrm{E}_{\mathcal{ALG}} = \frac{1}{m}\sum_{i\in[m]} \underbrace{\sum_{j\in\delta_i} \mathbb{E}\left[\left(\frac{y^{\infty}_{j\to i}}{y^{\infty}_{\delta_i}}\right)^2\right]\cdot\frac{1}{w_j+1}}_{s_i}\cdot T_i.$$

In this point, we can obtain the expected worker message using *Lemma 1*,

$$\mathbb{E}\left[y^{(\infty)}_{j\to i}\right] \cong \frac{(w_j+1)}{1+\frac{(w_j+1)}{lq}}\cdot T_i. \tag{3.17}$$

According to the definition of the worker message, the weight is described as

$$\frac{y^{\infty}_{j\to i}}{y^{\infty}_{\delta_i}} = \frac{y^{\infty}_{j\to i}}{\sum_{j\in\delta_i} y^{\infty}_{j\to i}}. \tag{3.18}$$

Using (3.16), (3.18) and arithmetic-geometric mean inequality with $w_j > 0$, the s_i is bounded as

$$s_i \leqslant \frac{(1+1/\hat{l}\hat{r})^2}{(\sqrt{2}+1)q\hat{l}\hat{r}}. \tag{3.19}$$

The second inequality comes from the definition of average quality. Thus, we obtain the error bound U_{iter}

$$\mathrm{E}_{\mathcal{ALG}} \leqslant \mathrm{U}_{\mathcal{ALG}} = \left(\frac{(1+1/\hat{l}\hat{r})^2}{(\sqrt{2}+1)q\hat{r}}\right)\cdot\frac{1}{\hat{l}m}\sum_{i\in[m]} T_i. \quad \square$$

Proof of Corollary 1

Then, we compare the average error of our algorithm with that of the oracle estimator. From (7) and (9), the gap between $\mathrm{E}_{\mathcal{ALG}}$ and $\mathrm{E}_{\mathcal{OC}}$ is as follows.

$$\varDelta = \underbrace{\left\{\frac{l(1+1/\hat{l}\hat{r})^2}{(\sqrt{2}+1)q\hat{l}\hat{r}} - \frac{1}{\mathbb{E}_{\mathbb{W}}\left[\mathrm{w}+1\right]}\right\}}_{\varDelta_w}\cdot\frac{1}{lm}\sum_{i\in[m]} T_i. \tag{3.20}$$

To show concisely, the mean and variance of the reliability distribution are denoted as μ_w and σ_w^2 respectively.
Using (3.20) with the definition of Δ_w, we are left to find the condition that satisfies $\Delta_w = 0$. If the distribution of worker reliability $\mathbb{W}$ is degenerate distribution with zero variance, the level of each worker is equal. Thus, it is not meaningful to distinguish the quality of the worker and for some $l > 1$and the error bound of *Theorem 1* is loose. As mentioned in (11), equal weighting is the best in degenerate distribution. However, when the distribution of worker reliability $\mathbb{W}$ follows certain probabilistic distribution rather than degenerate, we can approximate the quality as

$$q^{-1} = \mathbb{E}_{\mathbb{W}}\left[\frac{1}{\mathrm{w}+1}\right] \cong \frac{1}{\mathbb{E}_{\mathbb{W}}\big[\mathrm{w}+1\big]} \frac{\mathbb{E}_{\mathbb{W}}\big[(w+1)^2\big]}{\mathbb{E}_{\mathbb{W}}\big[w+1\big]^2} \tag{3.21}$$

using third order *Taylor* expansion around $\mathbb{E}_{\mathbb{W}}[w+1]$. It is known that this approximation is quite accurate when the distribution of worker reliability is symmetric. Substituting (3.21) into (3.20) provides $\Delta_w = 0$ is equivalent to

$$\frac{\sigma_w^2}{\mu_w^2} = \left(\frac{(\sqrt{2}+1)\hat{l}\hat{r}}{l(1+1/\hat{l}\hat{r})^2} - 1\right).$$

Then, applying *Chebyshev*'s inequality, it immediately follows

$$\mathbb{P}\Big((\mathrm{w}+1) \geqslant 2\mu_w\Big) \leqslant \frac{(\sqrt{2}+1)\hat{l}\hat{r}}{l(1+1/\hat{l}\hat{r})^2}. \quad \square$$

Next we compare the upper bound of our algorithm to the error performance of majority voting.

Corollary 2 *Under the hypotheses of Theorem 1,*

$$\mathrm{E}_{\mathcal{ALG}} \leqslant \mathrm{U}_{\mathcal{ALG}} \leqslant \mathrm{E}_{\mathcal{MV}} \tag{3.22}$$

Proof. From the results of (9) and (13), we always have

$$1 > \frac{(1+1/\hat{l}\hat{r})^2}{(\sqrt{2}+1)\hat{l}\hat{r}},$$

Source	Binary	Multi-class	Regression
Dawid and Skene[29]	✓	✓	
Whitehill et al. [20]	✓		
Welinder et al. [39]	✓		✓
Raykar et al. [18]	✓	✓	✓
Karger et al. [22]	✓		
Liu et al. [33]	✓		
Dalvi et al. [43]	✓		
Salek et al. [40]			✓
Karger et al. [34]	✓	✓	
Zhang et al. [44]	✓	✓	
Lee et al. [12]	✓	✓	

Table 3.5: Comparisons of the types of tasks covered by well-known crowdsourcing algorithms

given the fixed average quality q if the size of batch satisfies $\hat{l}\hat{r} > (\sqrt{2}+1)$ and $l > 1, r > 1$. Even if we assume the sparse random bipartite graph (i.e., $|m| \gg l$, $|n| \gg r$), only a few number of task allocation is sufficient to outperform majority voting. However, the performance gap is further increases as batch size increases. $\square$

3.5 Related Literature

Related work. For aggregation methods, majority voting is a widely used for its simplicity and intuitiveness. [45] shows majority voting can effectively reduce the error in the attribute-based setting. However, it regards every worker as equally reliable and gives an identical weight to all responses. Therefore, the performance of majority voting suffers even with a small number of erroneous responses [2]. To overcome this limitation, there have been several approaches for improving the inference performance from unreliable responses. [29, 39, 20] adopt Expectation and Maximization (EM) to evaluate the implicit characteristics of tasks and workers. Also, [44] improves this EM approach using a spectral method with performance guarantees. However, in practice, there is a difficulty in parameter estimation since these EM approaches are aimed at estimating a huge confusion matrix from relatively few responses.

[22, 33] proposed Belief Propagation(BP)-based iterative algorithms and proved that their error performances are bounded by worker quality and the number of queries in binary-choice tasks. Furthermore, there are several researches for crowdsourcing systems with multiple-choice tasks. [34] focused on multi-class labeling using a spectral method with low rank approximation, [46] proposed an aggregating method with minimax conditional entropy and [47] suggested an aggregation method using a decoding algorithm of coding theory. In addition, [12] exploits a inner product method($\mathcal{IP}$) for evaluating similarity measures between an answer from a worker and the group consensus.

There have been studies to target vector regression tasks: [48] and the DALE model in [40], which focus on finding the location of a bounding box in an image. The former suggests a simple serial task assignment method for a quality-controlled crowdsourcing system with no theoretical guarantee. The latter proposes a probabilistic graphical model for image object localization and inference method with expectation propagation. However, the worker model assumption in these papers has two limitations; it strictly divides the workers' expertise level and ignores the order of selection when a crowd divides a length into multiple segments. Also, the latter graphical model has too many parameters to learn from relatively small number of responses.

On the other hand, there are *outlier rejection* methods that can be used to filter unreliable responses without a graphical model. For non-parametric setting, *mean shift* and *top-k selection* are typically used as classical methods. *mean shift* is the technique for locating the maxima of a density function and *top-k selection* picks k most reliable responses based on distances between the mean vector and each response itself. For parametric setting, *RANSAC*(random sample consensus) is widely used. it is an iterative method to estimate parameters of a mathematical model from a set of responses that contains outliers, when they are to be accorded no influence on the values of the estimates.

While most of the papers mentioned above assume random regular task assignments, [43, 49] proposed inference methods in irregular task assignments. Also, [34, 12, 50] suggested the adaptive task assignment which gives more tasks to more reliable workers in order to infer more accurate answers given a limited budget.

Chapter 4

Homeostasis-Inspired Meta Continual Learning

4.1 Preliminaries

4.1.1 Continual Learning

As mentioned before, studies on the continual learning consider a scenario of which a single machine carries out a series of tasks with a limited amount of resources, such as computational power, memory space, etc. Note that depending on the details, it is also variously termed as *lifelong learning* [51], endless *online learning*, the special case of *transfer learning* [52], and methods of *Overcoming Catastrophic Forgetting* [8, 11, 6, 53]. In the literature, there are several different approaches that alleviate the forgetting phenomena in the continual learning, such as structural regularization [8, 10, 9, 53], adaptation with episodic memory [54, 55, 56, 57], expansion of network topology [58, 7], and dedication of partial network to a specific task [59, 60, 6]. Among them, we are especially interested in the *structural regularization* based methods. In brief, the studies on such methods focus on finding *important network parameters* in a trained network that affects a lot on learning accuracy with a small difference on their values. Having such information, we can regularize the network parameters when handling a series of tasks. Notable studies on the structural regularized based methods include the following: [8] proposes the square of the gradient as diagonal Fisher information using Laplace approximation; [9] develops this method to accumulate previous information in an online manner; *Synaptic Intelligence* [10] saves the full trajectory of gradient steps rather than a few steps; *Structured Laplace*[53] computes sequentially block diagonal components of Hessian in the order through the layers; and [11] suggests several transfer methods to make the search space of posterior weight smooth.

Synaptic Plasticity. In order to help understanding the proposed concept in this article, we make a quick revisit to the previous works EWC [8] and Online-EWC [9], in advance. Conventionally, in the related works on the continual learning framework, it is considered that a set of I tasks

$$\mathcal{T} = \{T_1, \cdots, T_I\},$$

where its i-th element $T_i = \{(x_i^s, y_i^s, t_i^s)_{s=1}^{S_i}\}$ is S_i-triplet of an input $x_i^s \in \mathcal{X}$, a target $y_i^s \in \mathcal{Y}$ and a task descriptor $z_i^s \in \mathcal{Z}$ for $s \in \{1, \ldots, S_i\}$, is given to a learner to carry out a series of tasks in increasing order of task index i. Note that $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$ respectively denote the set of inputs, targets and task descriptors.

Now suppose the learner is training tasks from T_1 to T_i, where $i \leq I$. Then, we can obtain the following negative log posterior of the network parameters θ in the approximated Bayesian perspective:

$$-\log p(T_i|\theta) + \frac{\alpha}{2} \sum_{j=1}^{i-1} ||\theta - \theta_j^*||_{F_j}^2, \tag{4.1}$$

where $\theta, \theta_j^*, \alpha$ and F_j represent the current-state network parameters of the learner, learned parameters after training task j, the IoR and Fisher information of task j, respectively. the norm $||\theta||_x^2 = \theta^T diag(x)\theta$ describes Mahalanobis norm [61] and the exact posteriors are replaced as

$$\log p(\theta|T_j) \approx \mathcal{N}\big(\theta; \theta_j^*, F_j^{-1}\big), \quad {}^{\forall}j \in \{1 : i-1\} \tag{4.2}$$

using Laplace approximation [62]. Unfortunately, in the original EWC, there exists a problem that the second term, i.e., the regularization term, tends to increase linearly with respect to the increment of the number of tasks. To solve the problem, in [9], the authors propose Online-EWC, which re-centers the likelihood of the previous tasks with the optimal parameters of the $(i-1)$-th task, and as a consequence, we can have

$$-\log p(T_i|\theta) + \frac{\alpha}{2}||\theta - \theta^*_{i-1}||^2_{F_{1:i-1}}, \tag{4.3}$$

where $F_{1:i-1} := \sum_{j=1}^{i-1} F_j$ for brevity. This approach is quite effective for reducing the required memory space for handing over information from the previous tasks to the present and computational burden.

By replacing the negative log-likelihood with the loss function of the task T_i, $l_i(\theta) := -\log(p(\theta|T_i)$, then we have the EWC loss as

$$l_{i,EWC}(\theta) = l_i(\theta) + \frac{\alpha}{2}||\theta - \theta^*_{i-1}||^2_{F_{1:i-1}}. \tag{4.4}$$

4.1.2 Meta Learning

Meta-learning, also known as *learning to learn*, aims to design models that can learn new skills or adapt to new environments rapidly, allowing the algorithm to learn to exploit structure in the problems of interest in an automatic way [63]. In the literature on the meta learning frameworks, there are many interesting works, such as [63] that proposes general framework for meta learning by mimicking the learning process with Recurrent Neural Network (RNN) and [64] that suggests Model Agnostic Meta Learning (MAML) which uses the original Stochastic Gradient Descent (SGD) as meta learner, but the initialization is learned via meta learning. For an advance version of MAML, [65] learns not only an initialization but also updating directions and the learning rates via meta learning.

Meta Learning for Continual Learning. Recently, the philosophy of meta learning is applied to the continual learning framework to enhance its performance. [66] proposes a simple optimizer which outputs the component-wise modified gradient for alleviating forgetting. On the other hand, [26] proposes a meta continual learning algorithm inspired by Reptile algorithm [67] which considers approximated second order correlation between parameters. However, the methods proposed in the preceding work lack scalability or incur high computing complexity. Unlike the previous studies, our approach is to design a relatively simple network for controlling the IoR (meta knowledge).

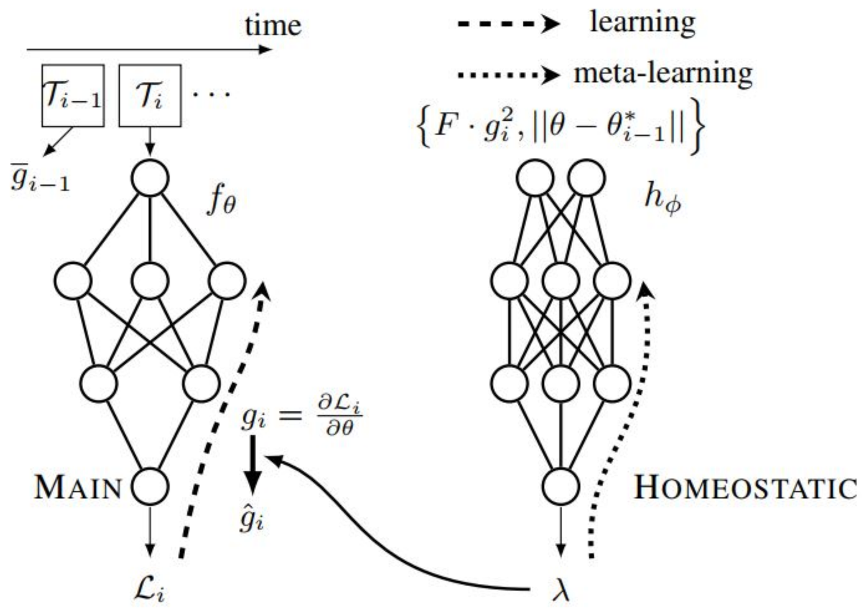
Towards applying the meta learning to the continual learning, we must distinguish the tasks for meta training and testing. Therefore, we hereafter consider a pair of task sets,

$$\mathcal{T}^0 = \{T_1^0, \cdots, T_{I_0}^0\} \text{ and } \mathcal{T}^1 = \{T_1^1, \cdots, T_{I_1}^1\},$$

respectively denoting meta training and testing set, where I_0 and I_1 is positive integers such that $I_0 > I_1$. As done in the previous studies the continual learning scenario ([8, 57]), $\mathcal{T}^0$ allows the learner to access all samples for the purposes of training its model which measures the importance of network parameters for new gradient update, while $\mathcal{T}^1$ is the actual sequence of tasks used for meta testing. Notice that the tasks in the meta test set $\mathcal{T}^1$ are only accessible during the meta testing process and cannot included in the meta training set for unbiased training.

4.2 Homeostatic Meta-Model

In this section, we propose a novel trainable network HM, as shown in Figure 1, which is well suited to the CL scenario to learn better and forget less. The role of the HM denoted by h_ϕ is to generate IoR, denoted as α. Recall that the fundamental meaning of regularization is to utilize side information for solving ill-posed problems, and correspondingly, the semantics of IoR indicates how large the optimization result is affected by the side information. Mathematically, the generation of the current- state



$$\text{VANILLA SGD} : \theta \leftarrow \theta - \gamma g_i$$

$$\text{HOMEOSTATIC SYNAPSE} : \theta \leftarrow \theta - \gamma\Big(g_i + h_\phi F \cdot (\theta_i - \theta^*_{i-1})\Big)$$

Figure 4.1: Illustrating a main network (learner) and the homeostatic network (meta-learned). Continual learning is performed across the sequence of tasks $\{\mathcal{T}_k\}$ with this model. Original gradient update g_k is replaced by $\hat{g}_k$ according to the F and λ from a homeostatic model.

IoR can be expressed as

$$\alpha = h_\phi\big(\underbrace{F_{1:i-1} \cdot g^2}_{(a)},\ \underbrace{||\theta - \theta^*_{i-1}||_2}_{(b)}\big) \tag{4.5}$$

when the learner is training T_i. The proposed model h_ϕ is a neural network, which is lighter than the main network and takes two inputs: (a) a inner product between the element-wise square of current-state gradients of given batch from current task i, i.e., g^2 and accumulated Fisher information $F_{1:i}$, and (b) Euclidean distance between the current-state parameters θ and θ^*_{i-1} which is the learned parameters from the previous task. Note that (a) is to give the HM the information on the similarity between the current learning direction and the previously-accumulated importance of parameters, and (b) is to inform how far the current parameters deviates from the previously-learned parameters. In this way, our HM guarantees scalability, in other words, the computational burden of the homeostatic network does not depend on the number of parameters in main network. To put it precisely with mathematical expressions, we can derive the following loss function of the proposed structure

$$l_{i,main}(\theta) = l_i(\theta) + \frac{h_\phi}{2}||\theta - \theta^*_i||^2_{F_{1:i-1}}. \tag{4.6}$$

In the training phase, since the HM is allowed to access all the samples in the task set ($\mathcal{T}^0$), it arbitrarily draws $\mathcal{B}_{i-1}$ and $\mathcal{B}_i$ from $\mathcal{T}^0_{i-1}$ and $\mathcal{T}^0_i$ respectively, then takes their union $\mathcal{B}_{i-1} \cup \mathcal{B}_i$ as an input of the meta model h_ϕ. The gradient of the model f_θ taking the union batch, $g_\cup$ is denoted by

$$g_\cup = \nabla_\theta l\big(f_\theta(\mathcal{B}_{i-1} \cup \mathcal{B}_i)\big). \tag{4.7}$$

Note that the gradient can play a role as a supervision for training h_ϕ with the absolute loss function,

$$l_{HM}(\hat{g}, g_\cup) = |\hat{g} - g_\cup|, \tag{4.8}$$

where

$$\hat{g} = g + h_\phi\big(F_{1:i-1} \cdot g^2, ||\theta - \theta^*_{i-1}||_2\big)F_{1:i-1} \cdot (\theta - \theta^*_{i-1}). \tag{4.9}$$

Algorithm 1 Training Homeostatic meta-model h_ϕ

```
procedure Train ($f_\theta, h_\phi, \mathcal{T}^0$)
for all $\{T^0_{i-1}, T^0_i\}$ in $T^0$ do
    $\theta^*_{i-1} \leftarrow$ base train $f_\theta$ on $T^0_{i-1}$ using SGD
    $\theta \leftarrow \theta^*_{i-1}$
    $F_{i-1} \leftarrow \mathbb{E}\big[\big(\nabla_\theta \mathcal{L}\big(f_\theta(T^0_{i-1})\big)\big)^2\big]$
    $F_{1:i-1} \leftarrow F_{1:i-2} + F_{i-1}$
    for batches $\mathcal{B}_{i-1}, \mathcal{B}_i$ drawn respectively from $T^0_{i-1}, \mathcal{T}^0_i$ do
        $g \leftarrow \nabla_\theta \mathcal{L}\big(f_\theta(\mathcal{B}_i)\big)$
        $g_\cup \leftarrow \nabla_\theta l\big(f_\theta(\mathcal{B}_{i-1} \cup \mathcal{B}_i)\big)$
        Update $\hat{g}$ using Eq (9) with $h_\phi$
        $\theta \leftarrow \theta - \gamma \hat{g}$
        $\phi \leftarrow \phi - \eta \nabla_\phi l_{HM}(h_\phi, g, g_\cup, F_{1:i-1})$
    end for
end for
end procedure
```

To stabilize the direction of the gradient $g_\cup$, we adopt larger batch and lower learning rate when training h_ϕ compared to main network training.

For testing the trained meta model, we generate IoRs by $h_{\phi^*}((a), (b))$, where h_{ϕ^*} denotes the optimized meta-trained HM using Algorithm 1, and (a) and (b) are respectively the inputs for the meta model as discussed above. Note that we freeze ϕ^* during the meta testing, while training the main network sequentially with tasks in $\mathcal{T}^1$. Here we describe the algorithm to test our model in detail.

4.3 Preliminary Experiments and Findings

`MNIST-PERM` [68, 69] is a widely-used set of tasks in the continual learning literature, which is made by simply permuting the input pixels of `MNIST` dataset. Its permutation

Algorithm 1 Testing meta-trained Homeostatic meta-model h_{ϕ^*}

```
procedure Test ($f_\theta, h_{\phi^*}, \{\mathcal{T}_1^1, \mathcal{T}_2^1, \mathcal{T}_3^1, ...\}$)
for all $\mathcal{T}_i^1$ do
  if $i = 1$ then
    $\theta_1^* \leftarrow$ train $f_\theta$ for one epoch on $\mathcal{T}_1^1$ using SGD
  else
    $\theta \leftarrow \theta_{i-1}^*$
    $F_{i-1} \leftarrow \mathbb{E}\big[\big(\nabla_\theta \mathcal{L}\big(f_\theta(\mathcal{T}_{i-1}^1)\big)\big)^2\big]$
    $F_{1:i-1} \leftarrow F_{1:i-2} + F_{i-1}$
    for each batch $\mathcal{B}_i$ from $\mathcal{T}_i^0$ do
      $g \leftarrow \nabla_\theta l\big(f_\theta(\mathcal{B}_i)\big)$
      Update $\theta$ using (9) with $h_{\phi^*}$
    end for
    $\theta_i^* \leftarrow \theta$
  end if
end for
end procedure
```

map are uniformly drawn and applied to generate a task. Since the number of pixels of a sample is generally a few hundreds or more, the Kendall tau correlation measure [70] between two randomly chosen permutation patterns is near zero, which means the correlation between them is very low. However, we can easily predict that the chance of drawing two permutation patterns with high correlation is larger in block-wise permutation tasks as shown in Figure 2 (a).

4.3.1 Block-wise Permutation

We call such block-wise permuted tasks as `BPERM`. The `BPERM` can be seen as a generalization of a pixel-wise permutation-based tasks, such as `MNIST-PERM` with block size of a single pixel. As shown if Figure 2 (c), the `MNIST-BPERM` looks like a Jigsaw puzzle, since it is made by dividing the input pixels into square blocks and permuting the blocks while the pixels inside the blocks are stay fixed.

To give more insights about the property of BPERM, we conducted a simple experiment with MNIST and CIFAR10 datasets. For each random seed, we train a learner with a normal task and BPERM task consecutively, then evaluate the accuracy of the first task.

Figure 2(b) shows the trend on the amount of information about the previous task that a learner forget, depending on the size of the blocks. Interestingly, the moderate-sized blocks cause more forgetting, which implies that the BPERM task gives larger interference than pixel-wise permutation task.

4.3.2 Performance Metrics

To assess how a learner's performance evolves as it learns a sequence of tasks, we define average accuracy (A), average forgetting (F) (*backward transfer*) [55]) and step forgetting (SF). Let $R_{i,j}$, for $j \leq i$, be the accuracy of the model on task T_j^1 after training the task T_i^1. Then we define a metric, *average accuracy*, defined as $A_k = \frac{1}{k} \sum_{j=1}^{k} R_{k,j}$, which describes the average of the model accuracy on each task after training T_k^1. To measure the amount of knowledge forgot after the initial acquisition, we consider *average forgetting*, defined as $F_k = \frac{1}{(k-1)} \sum_{j=1}^{k-1} (R_{j,j} - R_{k,j})$. In addition, we also consider *step-forgetting*, defined as $SF_k = \frac{1}{(k-1)} \sum_{j=1}^{k-1} (R_{k-1,j} - R_{k,j})$, which indicates the amount of information loss owing to the acquisition of new knowledge by training T_k^1.

4.4 Experiment

The experimental results for assessing the performance of the HM-based continual learning is provided in this section. In general, the performance of continual learning largely depends on how the sequence of tasks are chosen. To take care of this issue, we conducted the experiments with several different random seeds and averaged the performance measures to avoid bias that comes from a naive choice of task sequence.

4.4.1 Datasets

We consider three reasonable variants for each `MNIST` and `CIFAR10` datasets:
(1) `MNIST-PERM` and `CIFAR-PERM` permute input pixels of the `MNIST` and `CIFAR10` through randomly drawn permutation maps.
(2) `MNIST-ROTA` and `CIFAR-ROTA` rotate them by a randomly drawn integer angles in $[0, \pi]$.
(3) `MNIST-BPERM` and `CIFAR-BPERM` block-wisely permute them through randomly drawn coarser permutation maps.
Note that, for `CIFAR10`, each transformation is applied to all RGB dimensions.

	MNIST			CIFAR		
	PERM	ROTA	BPERM	PERM	ROTA	BPERM
number of tasks	$10, 40$	10		$10, 40$	10	
input size		28×28			32×32	
training smaples		10			10	
test images		10			10	
setup range	28×28	$[0, \pi]$	block 7×7	32×32	$[0, \pi]$	block 7×7

4.4.2 Methods

The proposed *homeostatic meta-model* (HM) described in Section 4.2 is compared with 6 alternative baseline methods listed as follows. - Single : a single learner based on SGD for a sequence of tasks.
- Independent : a dedicated (independent) learner based on SGD for each task.
- EWC [8] : regularized with the dedicated Fisher information F_i for each task i.
- OEWC [9] : regularized with the accumulated Fisher information $F_{1:i-1}$ as Eq. (4).
- IMM [11] : *Incremental Moment Matching* with a weight transfer method.
- Multi-task : allowed to access all the tasks (violation of strict CL scenario).

We adopt fully-connected networks with two hidden layers of 50 ReLU units for the main networks in each considering method and four additional hidden layers of 30 ReLU units are used in the HM. All the networks in baselines and the proposed HM is trained using SGD with mini-batch size of 100 samples.

4.4.3 Overall Performance

Figure 4.3 and 4.5 show that the proposed HM outperforms the other alternatives when training a sequence of tasks, in terms of average accuracy and forgetting, with equal or less number of model parameters (*resource-efficient*). Specifically compared to EWC, the performance variance of the HM is relatively small, which implies that the proposed method is more robust than EWC (*robustness*).

To capture the performance of the HM with given a longer sequence of tasks, we conducted a experiment with 40 consecutive `MNIST-PERM` tasks. The experimental results are illustrated in Figure 4.4. The proposed HM still outperforms the alternatives considerably. Figure 4.4 (a) represents the evolution of the accuracy of the main network on the 2nd task. Likewise, each (b), (c) and (d) represents the accuracy of the main network on 8th, 18th and 24th task, respectively.

Figure 4.6 provides the trend of IoR and step-forgetting during the continual learning. The IoR generated by HM interestingly appears to be proactively controlled within a certain range. In most cases, IoR is small at each task boundary and becomes larger as learning progresses, until the next task boundary. In addition, the proposed HM has smaller step-forgetting value compared to the alternatives, which implies the amount of information lost is the smallest among all the compared methods when the main model acquires new knowledge.

4.5 Related Literature

Adaptation with Episodic Memory. Other group of works assume a learner has fixed-size memory for episodic memory. In the beginning, [54] proposes a class incremental problem with a memory management method, and [55] solves a continual learning task as an optimization problem with episodic memory constraints. Then [56] suggested that a random access method for episodic memory with large learning rate would be sufficient for the model adaptation.

Expansion of Network Topology. If a learner can expand the size of the network, it can allocate a little portion of the whole network dedicated to each task for alleviating forgetting. [58] proposes the simple method of network expansion with an adaptor which summarizes previous information into the low dimensional vector. But their work has limitations in scalability. To overcome this limitation [7] suggests a gradual method of determining the number of neurons that expand according to certain thresholds.

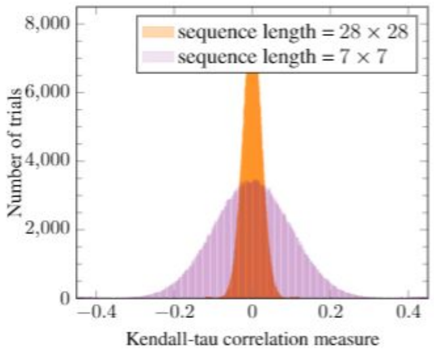
Dedicating a partial network to a specific task. In order to share a single network for multiple tasks, several studies softly divide the given network into some parts. Each part is dedicated to each task for avoiding interference. [59] defines a modular network to make a different pathway for each task and [60] proposes a network pruning method to help neurons to be uncorrelated for packing information in fixed network topology. Also, [6] proposes a hard attention method with annealing method for training, in order to dedicate the part of neural network for each task.

4.6 Discussion

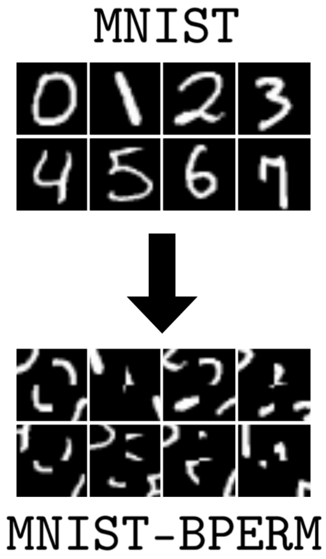
Crossing over the Meta Knowledges. In the meta-training process, the meta learner learns *Meta Knowledge* which means the method of not forgetting the previous task's information. We need to scrutinize how *Meta Knowledge* from different kinds of tasks works for each continual learning task. In particular, we consulted experiments by crossing over task A's *Meta Knowledge* to task B's continual learning task and task B's *Meta Knowledge* to task A's continual learning task. However, due to the discrepancy in parameter scale between two tasks, crossover meta learner did not perform well. Thus it was not covered in this paper. Choosing the pair with high similarity between two types of tasks, then it will be possible to apply *Meta Knowledge* gained in one field to another.

Layer-wise Meta Learner. According to the types of tasks, each layer of a neural network shows a large gap in the amount of parameter change. To overcome this discrepancy, we need to consider to build multiple meta-learners so that each meta-learner is dedicated to each layer, given the number of parameters. However, when we use deep neural networks, building a meta learner for each layer can be infeasible in terms of resources, thus it is necessary to group them into group-wise configurations.

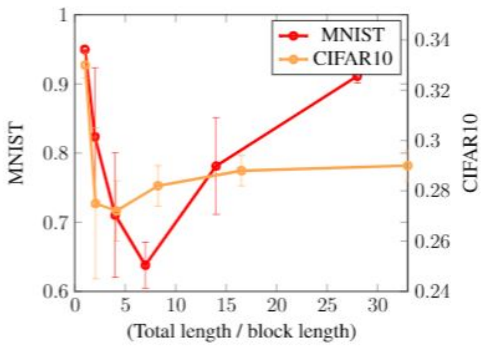
Various architectures for meta learner. In this paper, we use only Fully-connected Neural Networks for meta learners. However, it is better to add modules such as CNN or attention to save memory for the parameters of the network and to effectively learn the correlation between the parameters. If the resources required for meta learners become smaller, it becomes possible to compute the full correlation matrix of parameters as in [53].



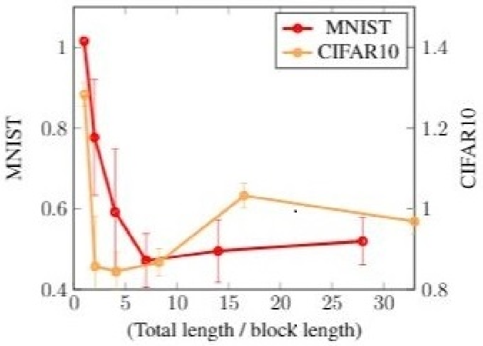
(a)



(b)



(c)



(d)

Figure 4.2: (a) A histogram of Kendall-tau correlation measure between randomly drawn two permutations $(500,000$ trials). (b) An example of `MNIST-BPERM`. An illustration of the average accuracy of Task 1 (normal task) after training Task 2 (`BPERM`) `MNIST` and `CIFAR10` datasets (with (c) FNN and (d) CNN).

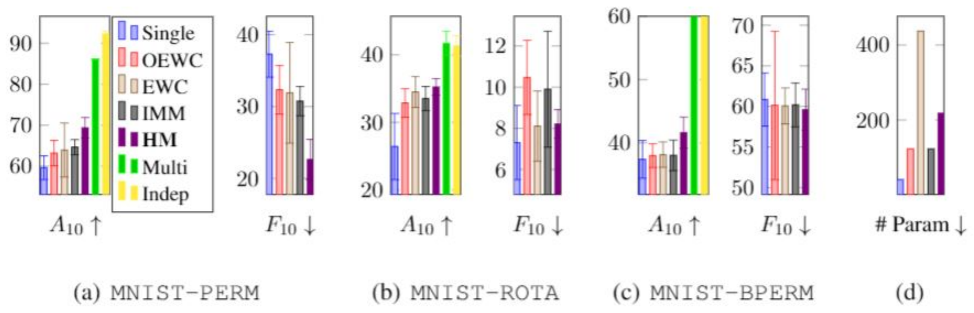
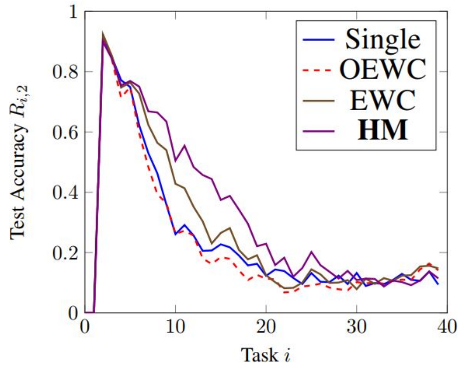
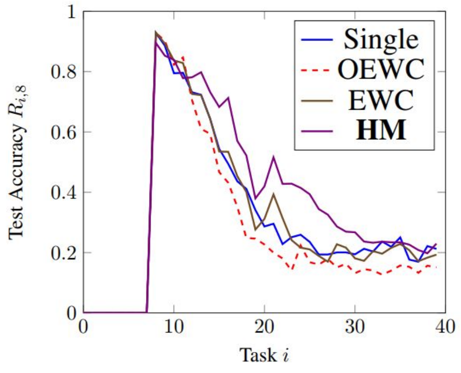


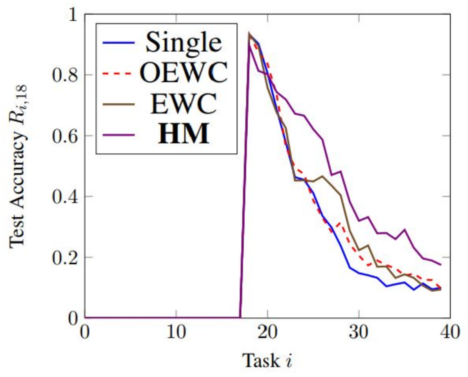
Figure 4.3: Comparisons on average accuracy (A_{10}) and forgetting (F_{10}), in percentage, of the considered learning methods including the HM, averaged over 10 random seeds on 10 tasks ((a) `MNIST-PERM`, (b) `MNIST-ROTA`, (c) `MNIST-BPERM` with block length = 4) (d) illustrates the number of model parameters employed in each method.) Note $\uparrow$ indicates that larger number is a better model, and $\downarrow$ means the opposite.



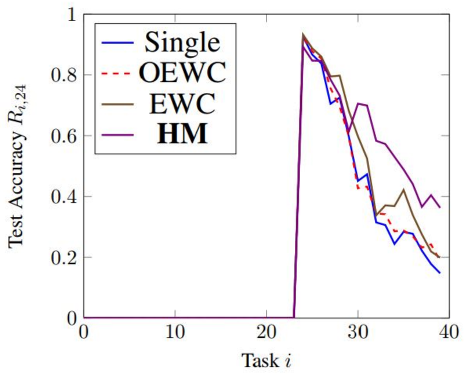
(a) $j = 2$



(b) $j = 8$



(c) $j = 18$



(d) $j = 24$

Figure 4.4: Evolution of test accuracy ($R_{i,j}$) of the main network with a long MNIST-PERM task sequence ($i \in [1, 40]$).

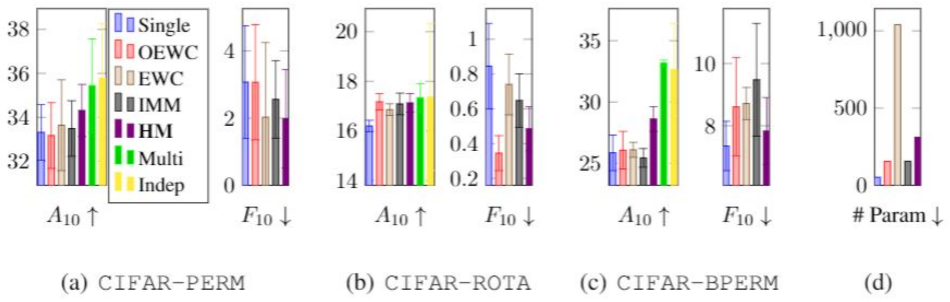
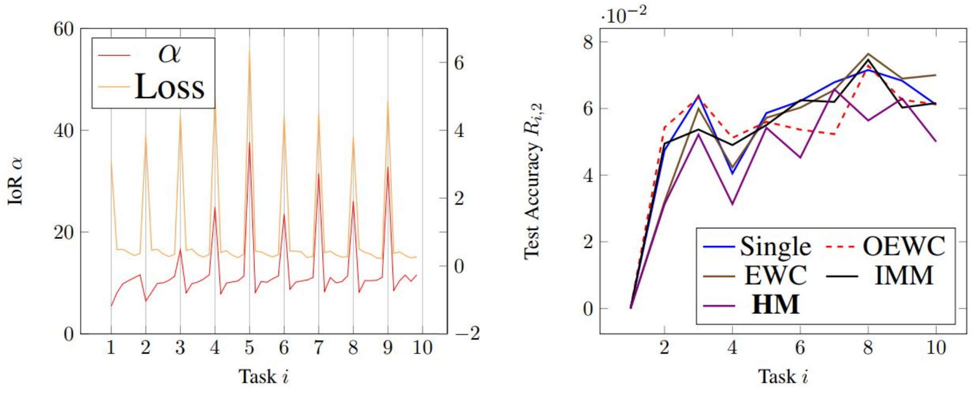
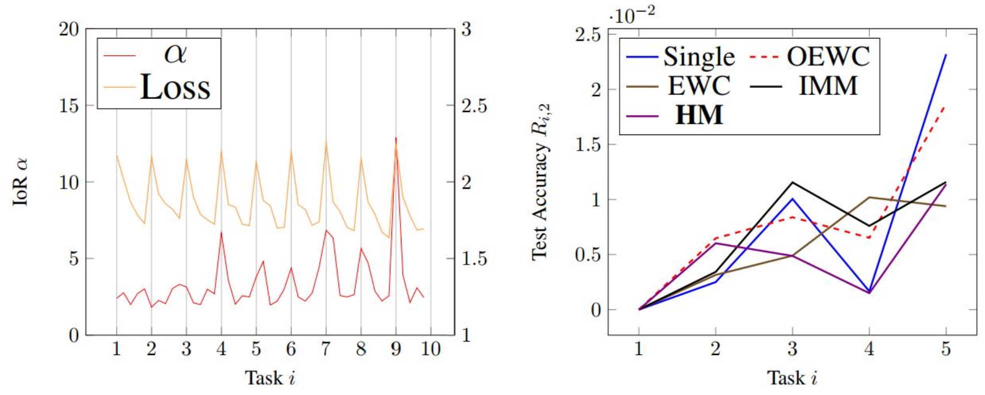


Figure 4.5: Comparisons on average accuracy (A_5) and forgetting (F_5), in percentage, of the considered learning methods including the HM, averaged over 4 random seeds on 5 tasks ((a) `CIFAR-PERM`, (b) `CIFAR-ROTA`, (c) `CIFAR-BPERM` with block length = 8, (d) figure illustrates the number of model parameters employed in each method.) Note $\uparrow$ indicates that larger number is a better model, and $\downarrow$ means the opposite.



(a) MNIST-PERM



(b) CIFAR-PERM

Figure 4.6: A trend of IoR generated by HM during continual learning and performance comparison of step-forgetting with (a) MNIST-PERM and (b) CIFAR-PERM

Chapter 5

Conclusion

To train a model that shows high performance on various tasks, it is essential to control the noise and interference among training data. To alleviate the noise in crowd-sourced data, the iterative algorithms for both multiple-choice and vector regression problems are proposed. They give a general solution for real crowdsourcing systems to infer the correct answers from unreliable responses. From the performance analysis of the algorithms, I have proved that an upper bound on the probability of error decays exponentially according to task assignments.

As for the interference in continual learning scenario, a *robust*, *resource-efficient*, and neuromophic architecture is proposed. It can control of IoR to alleviate the amount of interference between the previously-acquired knowledge and the current learning one. Moreover, by considering a generalized continual task type, i.e., `BPERM`, for the continual learning, the effectiveness of the proposed method is highlighted. The experimental results demonstrate the proposed HM outperforms the state-of-the-art methods with less number of model parameters in terms of the average accuracy and amount of the interference.

I hope that this effort to reduce the noise and interference will be a stepping stone to improve the relation among labeled data for better training regime.

Bibliography

[1] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[2] Victor S Sheng, Foster Provost, and Panagiotis G Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 614–622. ACM, 2008.

[3] Yaling Zheng, Stephen Scott, and Kun Deng. Active learning from multiple noisy labelers with varied costs. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 639–648. IEEE, 2010.

[4] Gabriella Kazai, Jaap Kamps, and Natasa Milic-Frayling. An analysis of human factors and label accuracy in crowdsourcing relevance judgments. *Information retrieval*, 16(2):138–178, 2013.

[5] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[6] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pages 4555–4564, 2018.

[7] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017.

[8] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[9] Jonathan Schwarz, Jelena Luketina, Wojciech M Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. *arXiv preprint arXiv:1805.06370*, 2018.

[10] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3987–3995. JMLR. org, 2017.

[11] Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. In *Advances in neural information processing systems*, pages 4652–4662, 2017.

[12] Donghyeon Lee, Joonyoung Kim, Hyunmin Lee, and Kyomin Jung. Reliable multiple-choice iterative algorithm for crowdsourcing systems. In *Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 205–216. ACM, 2015.

[13] Aniket Kittur, Ed H Chi, and Bongwon Suh. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 453–456. ACM, 2008.

[14] Chris J Lintott, Kevin Schawinski, Anže Slosar, Kate Land, Steven Bamford, Daniel Thomas, M Jordan Raddick, Robert C Nichol, Alex Szalay, Dan Andreescu, et al. Galaxy zoo: morphologies derived from visual inspection of galaxies from the sloan digital sky survey. *Monthly Notices of the Royal Astronomical Society*, 389(3):1179–1189, 2008.

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[16] Panagiotis G Ipeirotis, Foster Provost, and Jing Wang. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 64–67. ACM, 2010.

[17] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 254–263. Association for Computational Linguistics, 2008.

[18] Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. *The Journal of Machine Learning Research*, 11:1297–1322, 2010.

[19] Peter Welinder, Steve Branson, Pietro Perona, and Serge J Belongie. The multidimensional wisdom of crowds. In *Advances in neural information processing systems*, pages 2424–2432, 2010.

[20] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in neural information processing systems*, pages 2035–2043, 2009.

[21] David R Karger, Sewoong Oh, and Devavrat Shah. Budget-optimal crowdsourcing using low-rank matrix approximations. In *Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on*, pages 284–291. IEEE, 2011.

[22] David R Karger, Sewoong Oh, and Devavrat Shah. Iterative learning for reliable crowdsourcing systems. In *Advances in neural information processing systems*, pages 1953–1961, 2011.

[23] Robert Hecht-Nielsen. Theory of the backpropagation neural network. In *Neural networks for perception*, pages 65–93. Elsevier, 1992.

[24] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

[25] Roger Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97(2):285, 1990.

[26] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018.

[27] Gina Turrigiano. Homeostatic synaptic plasticity: local and global mechanisms for stabilizing neuronal function. *Cold Spring Harbor perspectives in biology*, 4(1):a005736, 2012.

[28] Gina G Turrigiano and Sacha B Nelson. Homeostatic plasticity in the developing nervous system. *Nature reviews neuroscience*, 5(2):97–107, 2004.

[29] Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Applied statistics*, pages 20–28, 1979.

[30] Luis Von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. recaptcha: Human-based character recognition via web security measures. *Science*, 321(5895):1465–1468, 2008.

[31] David R Karger, Sewoong Oh, and Devavrat Shah. Budget-Optimal Task Allocation for Reliable Crowdsourcing Systems. *Operations Research*, 62(1):1–24, February 2014.

[32] Noga Alon and Joel H. Spencer. *The probabilistic method*. John Wiley & Sons, Hoboken, NJ, 2008.

[33] Qiang Liu, Jian Peng, and Alex Ihler. Variational inference for crowdsourcing. In *Advances in Neural Information Processing Systems*, pages 692–700, 2012.

[34] David R Karger, Sewoong Oh, and Devavrat Shah. Efficient crowdsourcing for multi-class labeling. In *Proceedings of the ACM SIGMETRICS/international conference on Measurement and modeling of computer systems*, pages 81–92. ACM, 2013.

[35] Pinar Donmez, Jaime G Carbonell, and Jeff Schneider. Efficiently learning the accuracy of labeling sources for selective sampling. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 259–268. ACM, 2009.

[36] S Ertekin, H Hirsh, and C Rudin. Approximating the wisdom of the crowd. In *Proceedings of the Workshop on Computational Social Science and the Wisdom of Crowds*, 2011.

[37] Chien-Ju Ho, Shahin Jabbari, and Jennifer W Vaughan. Adaptive task assignment for crowdsourced classification. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 534–542, 2013.

[38] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014.

[39] Peter Welinder and Pietro Perona. Online crowdsourcing: rating annotators and obtaining cost-effective labels. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 25–32. IEEE, 2010.

[40] Mahyar Salek and Yoram Bachrach. Hotspotting-a probabilistic graphical model for image object localization through crowdsourcing. In *AAAI*, 2013.

[41] Béla Bollobás. Random graphs. In *Modern graph theory*, pages 215–252. Springer, 1998.

[42] Tom Richardson and Ruediger Urbanke. *Modern coding theory*. Cambridge university press, 2008.

[43] Nilesh Dalvi, Anirban Dasgupta, Ravi Kumar, and Vibhor Rastogi. Aggregating crowdsourced binary ratings. In *Proceedings of the 22nd international conference on World Wide Web*, pages 285–294. ACM, 2013.

[44] Yuchen Zhang, Xi Chen, Dengyong Zhou, and Michael I Jordan. Spectral methods meet em: A provably optimal algorithm for crowdsourcing. In *Advances in neural information processing systems*, pages 1260–1268, 2014.

[45] Asif R Khan and Hector Garcia-Molina. Attribute-based crowd entity resolution. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 549–558. ACM, 2016.

[46] Dengyong Zhou, Qiang Liu, John C Platt, and Christopher Meek. Aggregating ordinal labels from crowds by minimax conditional entropy. In *ICML*, volume 14, pages 262–270, 2014.

[47] A. Vempaty, L.R. Varshney, and P.K. Varshney. Reliable crowdsourcing for multi-class labeling using coding theory. *Selected Topics in Signal Processing, IEEE Journal of*, 8(4):667–679, Aug 2014.

[48] Hao Su, Jia Deng, and Li Fei-Fei. Crowdsourcing annotations for visual object detection. In *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*, volume 1, 2012.

[49] Yao Ma, Alex Olshevsky, Venkatesh Saligrama, and Csaba Szepesvari. Crowdsourcing with sparsely interacting workers. *arXiv preprint arXiv:1706.06660*, 2017.

[50] Chenxi Qiu, Anna C Squicciarini, Barbara Carminati, James Caverlee, and Dev Rishi Khare. Crowdselect: increasing accuracy of crowdsourcing tasks through behavior prediction and user selection. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 539–548. ACM, 2016.

[51] Sebastian Thrun. Lifelong learning algorithms. In *Learning to learn*, pages 181–209. Springer, 1998.

[52] Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 17–36, 2012.

[53] Hippolyt Ritter, Aleksandar Botev, and David Barber. Online structured laplace approximations for overcoming catastrophic forgetting. In *Advances in Neural Information Processing Systems*, pages 3738–3748, 2018.

[54] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[55] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pages 6467–6476, 2017.

[56] Pablo Sprechmann, Siddhant M Jayakumar, Jack W Rae, Alexander Pritzel, Adria Puigdomenech Badia, Benigno Uria, Oriol Vinyals, Demis Hassabis, Razvan Pascanu, and Charles Blundell. Memory-based parameter adaptation. *arXiv preprint arXiv:1802.10542*, 2018.

[57] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018.

[58] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[59] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.

[60] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018.

[61] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. National Institute of Science of India, 1936.

[62] David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[63] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in neural information processing systems*, pages 3981–3989, 2016.

[64] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org, 2017.

[65] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*, 2017.

[66] Risto Vuorio, Dong-Yeon Cho, Daejoong Kim, and Jiwon Kim. Meta continual learning. *arXiv preprint arXiv:1806.06928*, 2018.

[67] Alex Nichol and John Schulman. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999*, 2:2, 2018.

[68] Yoshua Bengio, Mehdi Mirza, Ian Goodfellow, Aaron Courville, and Xia Da. An empirical investigation of catastrophic forgeting in gradient-based neural networks. 2013.

[69] Rupesh K Srivastava, Jonathan Masci, Sohrob Kazerounian, Faustino Gomez, and Jürgen Schmidhuber. Compete to compute. In *Advances in neural information processing systems*, pages 2310–2318, 2013.

[70] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

[71] Paschalis Gkoupidenis, Dimitrios A Koutsouras, and George G Malliaras. Neuromorphic device architectures with global connectivity through electrolyte gating. *Nature communications*, 8(1):1–8, 2017.

초록

인공신경망 모델에 다량의 데이터를 학습시키는 방식은 컴퓨터 비전 및 자연어 처리 분야의 문제들을 해결하는데 새로운 패러다임으로 자리매김하였다. 기존 사람의 직관으로 모델을 설정하는 방식과 비교하여 높은 성능을 달성할 수 있었으나, 학습데이터의 양과 품질에 따라서 그 성능이 크게 좌우된다. 이렇게 인공 신경망을 효과적으로 훈련하려면 많은 양의 데이터를 모으는 것과 데이터의 품질을 저하시키는 요인을 파악하는 것이 중요하다. 본 연구에서는 라벨링된 데이터의 품질을 결정하는 주요 요인으로 알려져 있는 잡음(Noise)과 간섭(Interference)을 극복할 수 있는 기법을 제시한다.

연구자들은 일반적으로 웹기반의 크라우드 소싱시스템을 사용하여 다양한 사람들로부터 답변을 수집하여 데이터그룹을 구성한다[1]. 그러나 사람들의 답변으로 얻는 데이터는 작업 지침에 대한 오해, 책임 부족 및 고유한 오류로 인해서 데이터 입력(Input)과 출력(Target)사이에 잡음이 포함된다 [2, 3, 4]. 본 연구에서는 이렇게 크라우드 소싱을 통해 라벨링된 데이터에 존재하는 잡음을 극복하기 위한 추론 알고리즘을 제안한다.

두번째로, 모델의 학습성능을 저하시키는 요인인 데이터간의 간섭을 다룬다. 잡음이 제거되어 정제된 입력과 출력을 라벨링된 데이터 샘플이라고 하면, 학습시에 샘플들 사이의 관계를 생각할 수 있다. 사람 수준의 인공지능에 도달하기 위해서는 하나의 모델이 하나의 문제만을 해결하는 것이 아니라 시간상 순차적으로 직면하는 여러 문제를 동시에 해결할 수 있어야 한다[55, 8]. 이러한 상황에서, 샘플들 사이에 간섭이 발생할 수 있고, 학계에서는 연속학습(Continual Learning)에서의

"*Catastrophic Forgetting*"또는 "*Semantic Drift*"으로 정의하고 있다[5, 6, 7]. 본 연구에서는 이러한 간섭을 효과적으로 극복하기 위한 방법에 대한 연구를 다룬다.

앞서 언급한 데이터 잡음을 극복하기 위해서 첫 번째 장에서는 크라우드 소싱 시스템의 이산 객관식 및 실수 벡터 회귀 작업에 대한 새로운 추론 알고리즘을 각각 제안한다. 제안 된 알고리즘은 크라우드 소싱 모델을 그래프 모델(Graphical Model)로서 상정하고, 테스크와 답변을 주는 사람들간의 두 가지 유형의 메시지를 반복적으로 주고 받음으로써 각 작업의 정답과 각 작업자의 신뢰성을 추정 할 수 있다. 또한 이들의 평균 성능은 확률적 군중 모델을 이용하여 분석하고 입증한다. 이러한 성능에러 한계는 작업당 할당되는 사람들의 수와 작업자의 평균 신뢰성의해 결정된다. 사람들의 평균 신뢰도가 일정 수준을 넘어서면, 제안된 알고리즘의 평균 성능은 모든 작업자의 신뢰성을 알고있는 오라클 추정기 (이론적인 한계)에 수렴한다. 실제 데이터 세트와 합성 데이터 세트 모두에 대한 광범위한 실험을 통해, 제안된 알고리즘의 실제 성능이 이전의 state-of-the-art 알고리즘들 보다 우수하다는 것을 입증한다.

논문의 두 번째 장에서는 연속학습상황에서 데이터샘플사이에 발생하는 간섭을 해결하기 위해, 항상성기반의 메타 학습 구조 (Homeostatic Meta Model)를 제안한다. 구체적으로, 이전 테스크 중요한 학습 변수를 찾고 정규화에 선별적으로 적용하는 방법을 사용하는데, 제안된 모델은 이러한 정규화의 강도를 자동으로 제어한다. 이러한 기법은 새로운 학습을 진행할 때 이전에 획득한 지식을 최소한으로 잃어버리도록 인공신경망의 학습을 유도한다. 다양한 유형의 연속 학습 과제에서 제안된 방법을 검증하는데, 실험적으로 제안된 방법이 학습의 간섭완화 측면에서 기존 방법보다 우수하다는 점을 보인다. 또한 기존 시냅스 가소성 기반 방법들에[8, 9, 10, 11] 비해 상대적으로 변화에 강인하다. 제안된 모델에 의해 생성된 정규화의 강도 값은 시냅스에서 항상성 [71]의 음의 피드백 메커니즘과 유사하게, 특정 범위 내에서 능동적으로 제어된다.

주요어: Crowdsourcing, Continual Learning, Meta Learning

학번: 2012-20756