This paper was presented in a forum without peer-review. DEIM Forum 2018 G6-1

Optimizing Categorization Accuracy in Crowdsourcing through Hierarchical Reorganization

Xiaoni DUAN† and Keishi TAJIMA†

† School of Informatics, Kyoto University
Yoshida-Honmachi, Sakyo Kyoto 606-8501, Japan
E-mail: †duan@dl.soc.i.kyoto-u.ac.jp, ††tajima@i.kyoto-u.ac.jp

Abstract In this paper, we propose a method of improving accuracy of multiclass classification tasks in crowdsourcing. In order to improve quality of outputs in crowdsourcing, it is important to assign workers to tasks they are good at. In multiclass classification tasks, which are common in crowdsourcing, sometimes different workers are good at distinguishing different sub-categories. It suggests that we can improve the accuracy of a multiclass classification task by reorganizing the task into a hierarchical classification task and assigning workers to appropriate sub-tasks in the hierarchy. We create three worker allocation algorithms to show that we can actually improve the accuracy of a classification task experimentally. We collecte performance data of workers in a flat multiclass classification task on Amazon Mechanical Turk and simulate the process of the reorganization and the worker allocation. We confirm that the output in the simulation achieves higher accuracy than that of the original flat classification task. We also find that the combination of one of the reorganized hierarchical schemes and one of our worker allocation algorithm achieves the highest accuracy.

Key words Crowdsourcing, work flow management, hierarchical, worker allocation

1. Introduction

Crowdsourcing is an efficient and economical way to solve large quantity of tasks that are easy for humans but difficult for computers. Classification tasks, such as categorizing images, are common in many crowdsourcing platforms.

Categorization into multiple classes are called multiclass classification. In multiclass classification tasks, flat classification organization, which shows all choices of categories to workers at once, is commonly used. Many crowdsourcing platforms, such as Amazon Mechanical Turk,(注1) provide models for creating such flat multiclass classification tasks. However, showing many choices at once may confuse workers especially in fuzzy classification [1].

Another approach to a multiclass classification is to reorganize it into a hierarchical classification [2]. For example, a flat classification task can be reorganized into a hierarchical classification task that consists of a root sub-task which classifies data into several top-level categories, and sub-tasks that further classify each top-level category into their sub-categories.

In crowdsourcing, it is important to assign workers to tasks that they are good at in order to improve the quality of the outputs. Workers' ability often varies even within one multiclass classification task. Some workers may be good at some sub-categories while other workers may be good at other sub-categories. Therefore, if we reorganize a flat classification task into a hierarchical classification task, it may be possible to improve the accuracy of the classification by allocating each worker to a sub-category that the worker is good at.

In order to validate this idea, we ran a simulation using real worker data. We first collected images of several Canis class animals including four breeds of wolf-like dogs (Siberian Husky, Alaskan Malamute, Samoyed, and German Shepherd) and three other Canis animals (wolf, coyote and dhole). We then published a flat image classification project on Amazon Mechanical Turk, where we asked workers to classify these images into these seven categories. We collected answers from workers, we calculated the accuracy of each worker, and we also calculated the accuracy that each worker would have achieved if we had reorganized the flat classification task into some hierarchical classification task and had assigned the worker to some sub-tasks in the hierarchy. We know the correct answer for each image so we can calculate the accuracy.

There are many ways to reorganize this classification into a hierarchical classification. For example, we can reorganize it into a hierarchical classification task consisting of the following three sub-tasks:

(1) a task of classifying images into two categories "dogs" and "wild species"

(2) a task of classifying images that have been classified into the dog category further into the four breeds (Siberian Husky, Alaskan Malamute, Samoyed, and German Shepherd)

(注1)：http://www.mturk.com

(3) a task of classifying images that have been classified into the wild species category further into its three sub-categories (wolf, coyote, and dhole).

Some workers may be good at distinguishing dogs and wild species but bad at distinguishing four breeds of dogs. Some workers may be good at distinguishing four breeds of dogs but bad at distinguishing wolfs, coyotes, and dholes.

We generated 63 candidate hierarchical classification schemes, and for each of them, we ran a simulation in order to calculate the accuracy that we could have achieved by reorganizing the classification into that hierarchical classification, and allocating the given workers to appropriate sub-tasks in it.

In each simulation, we first allocate workers to sub-tasks in the hierarchy. We designed three algorithms to assign workers to appropriate sub-tasks in accordance with their accuracy for each sub-task. The first algorithm focuses on the number of remaining jobs in each sub-tasks, the second one focuses on the ratio of remaining jobs in each sub-classes and the third one focuses on the diversity of working ability of each workers in doing different jobs. For all algorithms, we calculated the accuracy of each sub-task we could have obtained based on the accuracy achieved by allocated workers in the original flat classification task. By merging them, we finally calculated the overall accuracy of the classification of images into the seven categories.

We compared the results for 63 candidates with the accuracy of the original flat classification, and our comparison have shown that the accuracy of hierarchical classification tasks is higher than that of the original flat classification task. We observed that one hierarchical scheme with the ability-focusing worker algorithm achieved the highest accuracy.

In summary, this paper reports the result of our preliminary simulation-based experiment which shows that there is a possibility that we can improve the accuracy of classification tasks by reorganizing them into hierarchical tasks.

2. Related Work

According to [3], improving worker quality, shortening time interval of completing a batch and reducing cost are three main problems of crowdsoucing, nowadays. A lot of work is done to solve these three problems. Some studies have combined majority voting and worker characteristic estimation. Dawid and Skene [4] proposed a method of estimating the quality of workers by the EM algorithm, and Ipeirotis et al. [5] combined it with weighted majority voting for removing bias of workers. Sheng et al. [6] reported how much repeated labeling can improve the task quality. Concerning reducing cost, Gao [7] designed two models to estimate the profit of tasks and eliminate answers that have negative effects.

There are also some proposals about worker flow control. Ho [8] derived a optimized assignment algorithm and achieved higher accurate precision as well as reduced cost. Eickhoff C [9] proposed a game-used worker flow management method to attract reliable workers.

We also use characteristics of workers, but our new idea is that we decompose a given classification task into hierarchical sub-tasks so that we can assign each worker to the best sub-task. To the best of our knowledge, this is the first research where the accuracy of a flat classification task is compared with the accuracy of the same classification task reorganized into a hierarchical classification task.

3. Hierarchical Scheme Generation

In order to achieve high accuracy by the reorganization into a hierarchical classification, we need to choose the best hierarchical scheme. Even when the number of final categories are small, e.g., less than 10, we have many ways to reorganize it into a hierarchical scheme.

In this paper, we only consider two-level hierarchical classification schemes with two top-level categories. In other words, we only consider schemes consisting of the following three sub-tasks:

(1) a top-level sub-task of classifying data into two top-level categories A and B

(2) a sub-task of classifying data that has been classified into A in the top-level sub-task further into the final sub-categories within A

(3) another sub-task of classifying data that has been classified into B further into the final sub-categories within B. The example scheme shown in Section 1. is also of this type.

If there are n ($n \geqq 3$) categories in a flat classification, the number of such hierarchical schemes, denoted by N, is calculated by the formula below:

$$N = 2^n/2 - 1 \tag{1}$$

We subtract 1 from $2^n/2$ because there is a scheme that is equivalent to the flat classification scheme. Notice also that there are n schemes where only one category is classified into A (or B) and the other $n-1$ categories are classified into B (or A). In these n schemes, we do not need the second-level classification for A (or B). Those n schemes, therefore, consist of two sub-tasks, not three.

In our experiment, we used a task of classifying images into 7 categories as explained before. Therefore, there are $2^7/2 - 1 = 63$ ways to reorganize the task into such a two-level classification task consisting of three (or two) sub-tasks. We generated all of them and ran a simulation for each of them.

4. Worker Allocation

As we explained in the previous section, we reorganize a classification task into a hierarchical classification task consisting of three (or two) sub-tasks, and assign each worker to a sub-task which the worker is good at. It is possible to assign a worker to more than one sub-tasks, but in this paper, we only consider simple worker allocation schemes where each worker is allocated only to one sub-task.

Notice that we cannot simply allocate each worker to a sub-task

Table 1 An Example of Confusion Matrix

	Alaskan Malamute	wolf	...	Husky
Alaskan Malamute	5	0	...	2
wolf	0	3	...	1
⋮	⋮	⋮	⋮	⋮
Husky	2	1	...	3

where the worker has achieved one's best accuracy. If we do it, all workers may be assigned to the same easiest sub-task and we may have no worker in other difficult sub-tasks. In addition, we need to consider how many images we can expect each worker to process. Some workers will process lots of images while some workers will process only a few images. For each sub-task, we need to assign enough workers to process all images in that sub-task.

We design three algorithms that assigns enough workers to each sub-task while trying to assign workers to a sub-task which they are good at. After getting workers' answers in the original flat classification task, we first calculated the following three values for each worker.

(1) Accuracy in distinguishing A and B. When we calculate it, if a worker classify an image in one category into another category within the same top-level category (e.g., classifying an image of Husky into Samoyed category, or classifying a coyote into wolf category), it is regarded as a correct answer.

(2) Accuracy in classification within A. When calculating it, we only consider images whose correct category is within A.

(3) Accuracy in classification within B. When calculating it, we only consider images whose correct category is within B.

When calculating Accuracy (1), if the user classify an iimage into A or B which includes the correct category of the image, we regard it as the correct answer, and we calculate the ratio of the correct answers. When calculating Accuracy (2) and (3), we calculate them by imitating the situation in hierarchical classification, and by using the confusion matrix [10] of the worker in the past as shown in Table 1. In the example of confusion matrix in Table 1, suppose we have a scheme where Husky and malamute is in A and wolf is in B, and a worker have classified 6 images of Husky in the past, 3 of them into Husky, 2 into malamute, and 1 into wolf. If the worker is assigned to the sub-task A, she cannot classify given images into wolf, which is not listed in the choices. When calculating her accuracy within A, we assume that she would classify the image she misclassified into wolf into either Husky or malamute in the probability 3/5 and 2/5, respectively, based on her past performance.

These three are expected accuracy of the worker in three sub-tasks. We create three lists of workers, each of which is sorted by one of these three kinds of accuracy in descending order. We assign workers to three sub-tasks by scanning these three lists from their tops in parallel. During that process, we also keep track of three values, j_{AB}, j_A, j_B, indicating how many jobs are remaining in three sub-tasks (the top-level sub-task distinguishing A and B, the sub-task for A, and the sub-task for B, respectively).

Algorithm 1 Remaining Jobs Focused Allocation Algorithm

Input: sorted worker lists: l_{AB}, l_A, l_B
Input: c, n, n_A, n_B, w_i
Output: workers allocated to sub-tasks: S_{AB}, S_A, S_B

$S_{AB} := \emptyset$; $S_A := \emptyset$; $S_B := \emptyset$
$j_{AB} := c$; $j_A := c * n_A/n$; $j_B := c * n_B/n$
while any of l_{AB}, l_A, l_B is not empty **do**
  $X :=$ the index of the largest one among j_{AB}, j_A, j_B
  (I.e., X is either AB, A, or B.)
  $i :=$ the worker at the top of l_X
  $S_X := S_X \cup \{i\}$
  remove i from l_{AB}, l_A, l_B
  $j_X = j_X - (2c * w_i/\Sigma_i w_i)$
return S_{AB}, S_A, S_B

Suppose we have c images. We may not know the ratio of images in each category in advance, so we simply assume that there are c/n images in each category, where n is the number of categories. We need c jobs to be processed by workers in the top-level sub-task, and we need $c * n_A/n$ jobs in the sub-task for A, where n_A is the number of categories within A. Similarly, we need $c * n_B/n$ jobs in the sub-task for B, where n_B is the number of categories within B and thus $n_A + n_B = n$. In total, we need $2c$ jobs to process.

Accordingly, suppose a worker i has done w_i jobs in the recent past on the crowdsourcing platform. We assume that the expected number of jobs each worker will do is proportional to w_i. Therefore, we expect worker i to do $2c * w_i/\Sigma_i w_i$ jobs.

4.1 Algorithm focusing on remaining jobs

This algorithm mainly focuses on the amount of remaining jobs. In this algorithm, we first sort worker according to three accuracy we discussed in Section 3. In this way we can get three worker lists l_{AB}, l_A, and l_B. In these three lists, workers are the same but sorted in different sequences. Then we choose the sub-task according to the largest value among j_{AB}, j_A, and j_B. We next allocate the top worker in the corresponding list to the task and remove the worker from all three lists. We also need to subtract the number of jobs we expect the worker to do, that is:

$$j_X = j_X - (2c * w_i/\Sigma_i w_i). \quad (2)$$

This process is repeated while there is a worker to assign. The detail of the algorithm is shown in Algorithm 1.

4.2 Algorithm focusing on ratio of remaining jobs

Now we explain the second algorithm for allocating workers to subtasks. Different from Algorithm 1 that allocates worker to the task that has the largest number of remaining jobs, in this algorithm, we allocate worker to the task that has the largest ratio of remaining jobs. We first initialize j_{AB}, j_A, and j_B to c, $c * n_A/n$, and $c * n_B/n$, respectively. We then choose a sub-task corresponding to the largest value among j_{AB}/n, j_A/n_A, j_B/n_B. That sub-task has the largest re-

Algorithm 2 Allocation Algorithm Focusing on Remaining Job Ratio

Input: sorted worker lists: l_{AB}, l_A, l_B
Input: c, n, n_A, n_B, w_i
Output: workers allocated to sub-tasks: S_{AB}, S_A, S_B

$S_{AB} := \emptyset$; $S_A := \emptyset$; $S_B := \emptyset$
$j_{AB} := c$; $j_A := c * n_A/n$; $j_B := c * n_B/n$
while any of l_{AB}, l_A, l_B is not empty **do**
    $X :=$ the index of the largest one among j_{AB}/n, j_A/n_A, j_B/n_B
    (I.e., X is either AB, A, or B.)
    $i :=$ the worker at the top of l_X
    $S_X := S_X \cup \{i\}$
    remove i from l_{AB}, l_A, l_B
    $j_X = j_X - (2c * w_i/\Sigma_i w_i)$
return S_{AB}, S_A, S_B

maining job ratio, so has the highest priority for worker allocation. We allocate the worker at the top of the list sorted by the accuracy in this sub-task. Let the worker is denoted by i. We then remove the worker i from all the three worker lists, and also subtract from j_X the expected number of jobs the worker i will do according to fomula 2. We repeat it until all the workers are assigned and removed from the lists. The algorithm is shown in Algorithm 2.

4.3 Algorithm focusing on worker ability

We also designed an algorithm focusing on diversity of workers' ability in different sub-tasks. We first load three sorted worker lists l_{AB}, l_A, l_B. For each worker i, we obtain i's three ranks, $r_{AB_i}, r_{A_i}, r_{B_i}$ in lists l_{AB}, l_A, l_B. We use rank here instead of exact accuracy values in three lists because accuracy values in different sub-tasks are uncomparable, as well as in different schemes. Next, we calculate variance of $r_{AB_i}, r_{A_i}, r_{B_i}$ for each worker, and produce a list of workers, W, sorted according to the variance mentioned above by descending order. If the variance is high, it means that the worker can do some jobs well, but cannot do others as well as doing that job. Then starting from the top worker in W, for each worker i in W, we find the largest one among $r_{AB_i}, r_{A_i}, r_{B_i}$. If there are remaining jobs in its corresponding sub-task, we assign worker to that task and subtract w_i from the number of remaining jobs in the task. If there is no job to do in the that task, we assign the worker to the second best task for her. If there are remaining jobs only in the third task, we have to assign worker i to that task. The detail of this algorithm is shown in Algorithm 3.

5. Experiments

Based on the proposed method, we simulate the process of reorganization and worker allocation for all the 63 hierarchical schemes. Our aim is to find the best scheme and allocate appropriate worker to appropriate sub-task. After we run simulation for all 63 schemes, we compare the result and publish three real hierarchical tasks to validate the result we get from the simulation. We compare the result of these three hierarchical tasks with the result of flat classification.

Algorithm 3 Worker Ability Focused Allocation Algorithm

Input: sorted worker lists: l_{AB}, l_A, l_B
Input: c, n, n_A, n_B, w_i
Output: workers allocated to sub-tasks: S_{AB}, S_A, S_B

1: $S_{AB} := \emptyset$; $S_A := \emptyset$; $S_B := \emptyset$
2: $j_{AB} := c$; $j_A := c * n_A/n$; $j_B := c * n_B/n$
3: get worker rank $r_{AB_i}, r_{A_i}, r_{B_i}$ of i in l_{AB}, l_A, l_B
4: Sort worker list W by variance of $r_{AB_i}, r_{A_i}, r_{B_i}$ in descending order
5: **for** i in worker list W **do**
6: Sort $R_i = r_{AB_i}, r_{A_i}, r_{B_i}$
7: $X_i :=$ the index of task $R_i[0]$
8: (I.e., X_i is either AB, A, or B.)
9: **if** j_{X_i} is greater than 0 **then**
10: $S_X := S_X \cup \{i\}$
11: $j_X = j_X - (2c * w_i/\Sigma_i w_i)$
12: **else**
13: $X_i :=$ next item in R and back to line 9
return S_{AB}, S_A, S_B

5.1 Simulation

We collected worker performance data by publishing a flat categorization task about classifying seven kinds of Canis animals on Amazon Mechanical Turk. We collected over 100 photos of each category, and published 720 photos in total. One HIT (a unit of task on Amazon Mechanical Turk) is consisted of 20 identified images. We finally obtained 10,040 answers from 107 unique workers. When reorganizing flat classification into hierarchical classification and assigning workers, we removed 15 spammers whose accuracy of flat classification was lower than 0.3. Although in reality, it is impossible to block spammers, we can still block workers who attended the original flat classification and were considered as spammers.

As described in Section 3., there are 63 different hierarchical schemes in our experiment. We imitated the process of reorganizing this flat classification task into 63 different hierarchical schemes and ran all algorithms for all these schemes. Our aim was to find the best schemes, in other words, finding the scheme that achieves the highest overall accuracy. Final overall accuracy is shown in Figure 1. In both graphs in Figure1, green bars indicate overall accuracy of Algorithm 3 focusing on worker ability, red bars indicate overall accuracy of Algorithm 2, and yellow bars show the overall accuracy of flat classification. The accuracy of both algorithm and flat cassification after removing spammers are shown in the upper graph. In the lower graph, we compare the accuracy of Algorithm 3 with original worker performance data and data after removing spammers. From the result we can find the best scheme with Algorithm 3.

Figure 2 shows the overall accuracy of Algorithm 3. In the figure, green bars indicate the overall accuracy of hierarchical schemes after removing spammers, while red bars indicate the overall accuracy from the same algorithm with original data, data with 107 workers including 15 spammers. From the graph we can see, the accu-

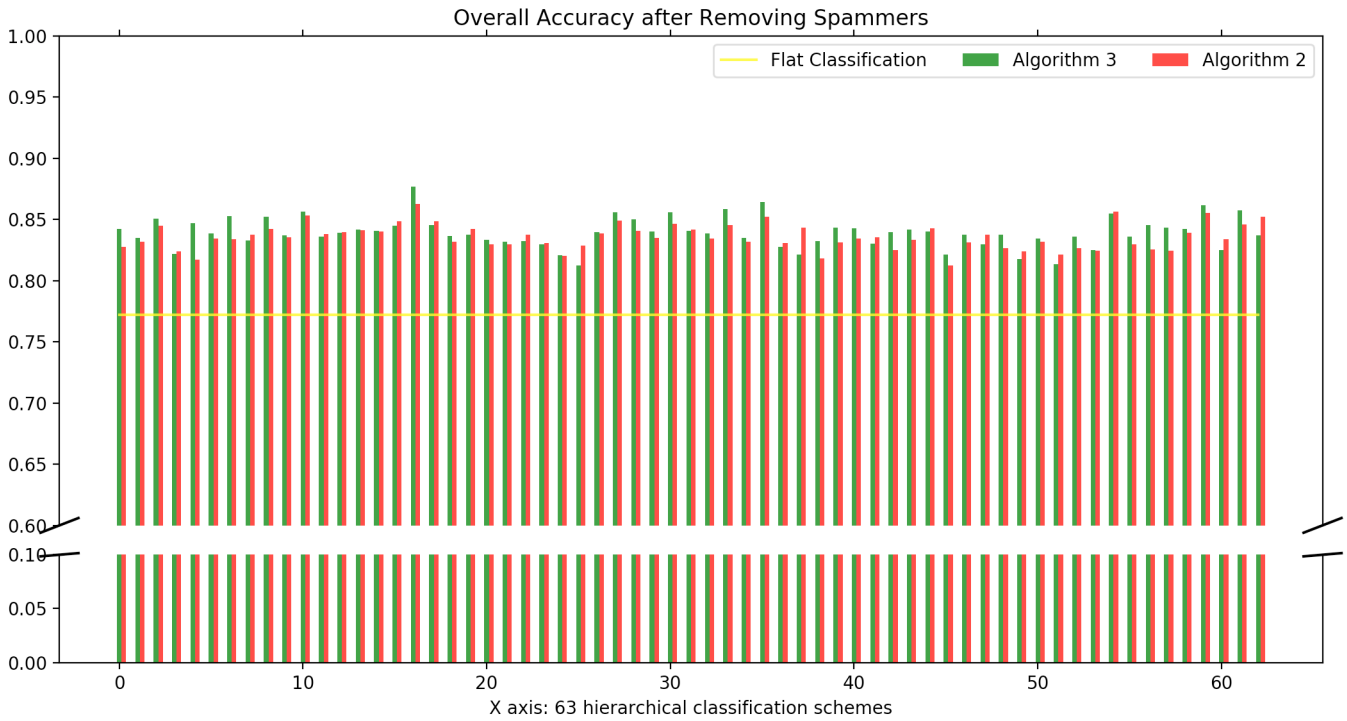

Figure 1 Overall accuracy of 2 worker allocation algorithms and flat classification

Table 2 Best and Worst Three Hierarchical Schemes in Algorithm 3 with former data set

rank	accuracy	*A*	*B*
1	0.877	(Samoyed, German shepherd)	others
2	0.864	(Samoyed, German shepherd, dhole)	others
3	0.861	(Samoyed, coyote, dhole)	others
⋮	⋮	⋮	⋮
61	0.817	(Malamute, coyote, dhole)	others
62	0.813	(Malamute, German shepherd, dhole)	others
63	0.812	(Malamute, German shepherd)	others

Table 3 Best and Worst Three Hierarchical Schemes in Algorithm 3 with new data set

rank	accuracy	*A*	*B*
1	0.866	(Samoyed, coyote,wolf)	others
2	0.862	(Alaskan malamute, Husky, dhole)	others
3	0.862	(Malamute, German Shepherd, Husky)	others
⋮	⋮	⋮	⋮
61	0.807	(German Shepherd)	others
62	0.800	(Alaskan malamute, wolf, dhole)	others
63	0.776	(Alaskan malamute, coyote)	others

racy was improved a lot after removing spammers. Therefore, it is vital to eliminate spammers when reorganizing flat classification tasks. We can also find that before removing spammers, the former schemes achieve higher accuracy than the other schemes. In our experiment, these schemes are 1-to-6 schemes, which means there is only one sub-category in class *A*, and six sub-categories in class *B*. This is because it is easier to distinguish one kind of animals even for spammers, so in these schemes, the top-layer help us removing spammers from attending the important sub-task, sub-task *B*.

Table 2 shows the best three and worst three schemes we obtained by Algorithm 3. In the table, categories in sub-class *A* and *B* are shown. We found the best scheme, rank 1 in Table 2. The highest accuracy of this scheme is 0.877.

Since we want to publish several real hierarchical classification tasks in Amazon Mechanical Turk, we also collected another data set using only 200 photos of 7 categories in flat classification. We collected another 80 photos, so we have 800 photos in total in our new data set. The remaining 600 photos are used for hierarchical classification task. In the new data set, we collected information of 153 workers, removed 11 spammers and ran the three algorithms with this data set. The detailed result was slightly different from the result we got from the former data set, but we still obtained the best scheme with Algorithm 3 and the accuracy of the simulation of hierarchical classification was better than that of the flat classification. We listed the best three and worst three schemes of this data set in Table 3.

In Table 4 and Table 5 we list the overall confusion matrix of two data sets. Table 4 is the confusion matrix of the former data and Table 5 is the confusion matrix of the later data. We use the ratio of photos classified into different categories instead of the number of photos in both tables. Rows in the confusion matrix show the ratio of photos in certain category classified into each category. For example, in the row Alaskan Malamute in Table 5, the value 0.611 means that 61.1% Alaskan malamute photos are classified as Alaskan malamute, while 0.22% photos of Alaskan malamute are classified as coyote. We also analyzed the reason why scheme 63

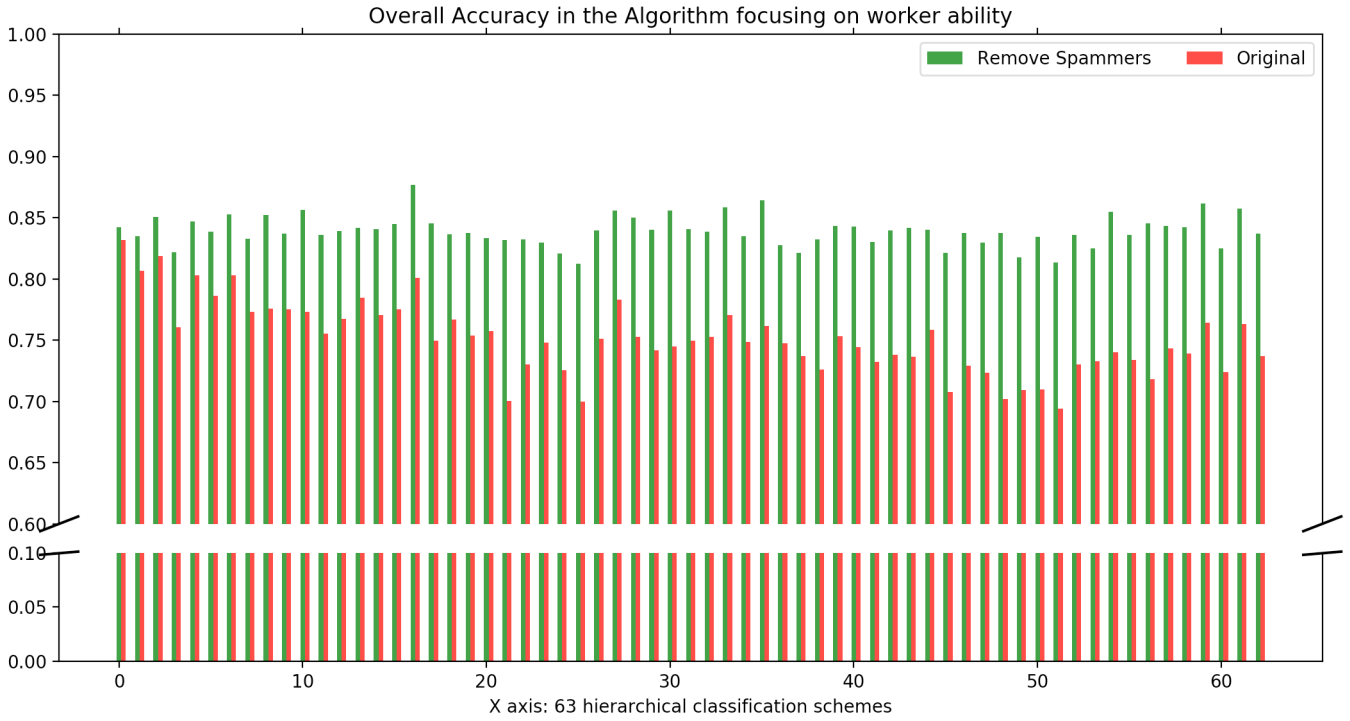

Figure 2 Overall accuracy of worker ability focusing algorithm with data after removing spammers and original data.

Table 4 Confusion Matrix of former data

	Alaskan Malamute	Coyote	Dhole	Wolf	German Shepherd	Samoyed	Husky
Alaskan Malamute	0.536	0.016	0.007	0.038	0.034	0.056	0.322
Coyote	0.038	0.600	0.092	0.194	0.032	0.017	0.026
Dhole	0.030	0.151	0.713	0.038	0.020	0.023	0.026
Wolf	0.053	0.094	0.023	0.659	0.046	0.036	0.088
German Shepherd	0.032	0.022	0.017	0.028	0.842	0.022	0.038
Samoyed	0.048	0.013	0.012	0.017	0.030	0.843	0.037
Husky	0.204	0.016	0.010	0.047	0.028	0.035	0.659

Table 5 Confusion Matrix of latter data

	Alaskan Malamute	Coyote	Dhole	Wolf	German Shepherd	Samoyed	Husky
Alaskan Malamute	0.611	0.002	0.009	0.016	0.024	0.038	0.3
Coyote	0.018	0.618	0.134	0.182	0.016	0.018	0.015
Dhole	0.010	0.209	0.698	0.030	0.012	0.018	0.015
Wolf	0.053	0.118	0.026	0.674	0.022	0.029	0.077
German Shepherd	0.009	0.009	0.022	0.008	0.917	0.018	0.017
Samoyed	0.104	0.006	0.045	0.020	0.015	0.755	0.055
Husky	0.218	0.010	0.008	0.040	0.028	0.023	0.674

in Table 3 is the worst one. We traced the process of worker allocation and observed workers allocated to each category. We found that the accuracy of classifying in sub-task A in this scheme was much lower than any other schemes. However, when we examined the worker allocated to sub-task A, we surprisingly found that workers in sub-task A were actually workers in good quality. Then we found that workers assigned to top-layer sub-task could not clearly classifying photos belong to sub-task A correctly. This may be because the accuracy of classifying Alaskan malamute and coyote are lower than other categories and it is easy to mistake them as categories in sub-task B, so some photos are wrongly classified into class B. That is to say, there is no way to correctly classify photos which is misclassified into another sub-class. Therefore, in order to find a better scheme, the accuracy of classification in top-layer task is of high importance.

From Table 4 and Table 5 we can see Alaskan malamute and husky, coyote and wolf are two pairs of categories that easily mistaken by workers. From Table 2 and Table 3 we can see that the best and worst schemes are different in two data sets even though we use the same algorithm. In Table 2 we find in top scheme Alaskan Mala-

mute and Husky, coyote and wolf are in the same sub-task, while in Table 3 we find that these two pairs are separated in two sub-tasks. We suspect that in the former data set, the workers that can best distinguish Alaskan Malamute and Husky and the workers that can best distinguish coyote and wolf are largely overlapping, and as a result, the mission of distinguishing such categories are in the top-level subtask, and those overlapping workers who are good at classifying them are assigned to this top-level subtask. On the other hand, in the later data set, these two sets of workers are not overlapping much, and as a result Alaskan Malamute and Husky were allocated in the sub-category A (or B), the workers good at this pair was assigned to the sub-task for A (or B), while coyote and wolf were assigned to the subtask B (or A), and the workers good at this pair was assigned to the subtask B (or A). In order to prove the speculation, we list worker by the accuracy of each worker distinguishing husky and Alaskan malamute and accuracy of distinguishing coyote and wolf in descending order. We examine top 30 workers in each data set. We find 13 workers are overlapping in the former data set and 9 workers overlapping in the later data set. That means the ratio of overlapping top worker in the former data set is larger than that in the later data set. This observation meets our speculation and it is reasonable to explain why we obtain the result schemes shown in Table 2 and Table 3.

Following the result that Alaskan malamute and Husky, coyote and wolf are two pairs that are most easily mistaken by workers, we decided to remove the other three categories and run simulation only with four categories, Alaskan malamute, Husky, wolf and coyote. We expected the scheme where Alaskan malamute and Husky are in the same sub-category and Coyote and wolf are in another category to be the best one. However, we cannot get the highest scheme by Algorithm 3 with which we got the highest accuracy with 7 categories. Whereas, we get the best accuracy with four categories in the former data set with Algorithm 1, algorithm focusing on maximum remaining jobs and in the later data set, and we get the best accuracy through Algorithm 2, the algorithm focusing on the maximum ratio of remaining jobs. The best scheme of the former data set is that husky and wolf in one category and Alaskan malamute and coyote in the other category and the accuracy this scheme is 0.7289. In the latter data set, the best scheme is in accord to our speculation, Alaskan malamute and Husky in the same sub-category and Coyote and wolf in another category and the accuracy is 0.8631. We can see that the accuracy of classifying four categories is lower than that of classifying seven categories. This is because we choose four categories that are harder to distinguish. Worker ability of distinguishing these four categories is not as diverse as that of classifying seven categories, so that Algorithm 3 does not work for data only with four categories.

5.2 Experiment on Amazon Mechanical Turk

We publish three real hierarchical classification tasks on Amazon Mechanical Turk. In Table 6, we list schemes that we published and the result of each scheme. We also publish a flat classification task on Amazon Mechanical Turk in order to evaluate our method. In all three hierarchical schemes and the flat classification scheme, five workers are assigned to answer one single question. We choose the best scheme, the worst scheme according to Table 3 and the best 1-to-6 scheme. We publish three hierarchical tasks with different numbers of categories in a certain sub-task. We have 600 photos remained to be used. We randomly put 200 photos in each hierarchical project and assign workers according to worker lists we get with Algorithm 3. All the workers are involved in three hierarchical tasks. Then we can compare the result we get from different schemes. These three hierarchical classification projects are processed at the same time and they are under processing on Amazon Mechanical Turk.

We used two methods to get correct answer in the flat classification task and we compare them with the ground truth. The first one is a method that commonly used, majority voting. However, even though we assigned five workers to one question, we cannot get majority answer for some questions. We also use a weighted voting method, EM Algorithm [4], to generate correct answer. In this algorithm, we first initiate answers by majority voting and then evaluate worker quality by these answers. Confusion matrix for each worker is generated and used then to evaluate correct answer. Different workers may have different weight in the weight voting. This process is repeated until the likelihood of expected answers converge. However, from Table 6 we can see, the overall accuracy of weighted voting method is worse than the simple majority voting method. This may because we only collect answers from 9 workers in this flat classification task, so low quality workers may answer more question than sincere workers. As a result, low quality workers reach higher weight in voting than high quality workers. In hierarchical tasks, we merely use majority voting to get correct answer. The accuracy is calculated by comparing the result of voting with the ground truth.

From Table 6 we can see, the top one scheme in the simulation achieves higher accuracy than the flat classification result by majority voting. However, the accuracy of the other two schemes are worse than that. From this observation we can confirm that the accuracy of classification task can be improved by reorganizing it into a hierarchical classification task. Furthermore, requesters should be meticulous to choose hierarchical schemes in that worker performance is strongly depended on the chosen hierarchical scheme. If the requester choose a good hierarchical scheme, the accuracy of the task can be improved, whereas, if he choose a bad one, he may get a bad result.

6. Conclusions

In this paper, we propose several methods to reorganize flat classification into corresponding hierarchical classification. We get this idea because we assume that different workers have various worker

Table 6 Result of hierarchical classification tasks and flat classification task on AMT

Scheme	Total Accuracy	Accuracy AB	Accuracy A	Accuracy B
A: Samoyed, Coyote,Wolf B: others	0.875	0.970	0.853	0.850
A: Alaskan malamute, Coyote B: others	0.590	0.610	0.444	0.644
A: wolf B: others	0.765	0.900	0.880	0.749
Flat classification (majority voting)		0.833		
Flat classification (EM Algorithm)		0.767		

ability in doing different tasks. We generate all the possible schemes and design three worker allocation algorithms. The first and second algorithms focus on balancing remaining jobs and allocate worker good at doing this job. The third algorithm focuses on worker ability. In this algorithm we sort worker by variance of working on different jobs and allocate in this sequence. We test our three algorithms with two data sets. Worker information is different in two data sets. Our results show that we can get the scheme that achieves the highest accuracy with the third algorithm testing with both data sets. We provide the confusion matrixes to evaluate answers collected from workers. We observe differences in two data sets and do further experiments to analysis the difference. We also publish three hierarchical tasks on the Amazon Mechanical Turk and compare the result with the result of flat classification. We find our best scheme can achieve higher accuracy than the flat classification. In conclusion, based on our simulation and analysis, we confirm that reorganizing flat classification into hierarchical classification can improve the accuracy that photos being classified into correct categories finally.

7. Future Work

Although the accuracy of classification task can be improved by using our method, we still find some limitations. In our future work, we'll think of some methods to solve these problems.

In our research, in order to qualify worker ability, a qualification task is required. We cannot modify the quantity of data that should be used in the qualification task. In our experiment, 200 photos in 800 photos are used. We cannot figure out whether we should use $1/4$ of the whole data or 200 photos in the qualification task if there is a large number of questions in the original classification task. If large quantity of data is required in the qualification task, it will be costly. We have to design an experiment to set up a threshold for the quantity of data for qualification task, and with this threshold, the cost of qualification task can be limited.

We proposed several greedy algorithms to allocate workers and we get some good schemes with these algorithms. However, these algorithms are not optimized. We desired to design an optimized method to allocate workers, so that we'll get the hierarchical scheme with the highest accuracy. We want to design an dynamic worker allocation algorithm for hierarchical reorganization.

8. Acknowledgement

This work was supported by JST CREST Grant Number JPMJCR16E3, Japan.

References

[1] Del Amo, A and Montero, J and Cutello, V, "On the principles of fuzzy classification," Fuzzy Information Processing Society, 1999. NAFIPS. 18th International Conference of the North American, pp. 675–679, 1999.

[2] Silla Jr, Carlos N and Freitas, Alex A, "A survey of hierarchical classification across different application domains," Data Mining and Knowledge Discovery, Vol. 22, No. 1-2, pp. 31–72, 2011.

[3] Li, Guoliang and Wang, Jiannan and Zheng, Yudian and Franklin, Michael J, "Crowdsourced data management: A survey," IEEE Transactions on Knowledge and Data Engineering, Vol. 28, No. 9, pp. 2296–2319, 2016.

[4] Dawid, Alexander Philip and Skene, Allan M, "Maximum likelihood estimation of observer error-rates using the EM algorithm," Applied statistics (JSTOR), pp. 20–28, 1979.

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

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

[7] Gao J, Liu X, Ooi B C, et al, " An online cost sensitive decision-making method in crowdsourcing systems," Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data (ACM), pp.217–228, 2013.

[8] Ho C J, Jabbari S, Vaughan J W, "Adaptive task assignment for crowdsourced classification," International Conference on Machine Learning, pp. 534–542, 2013.

[9] Eickhoff C, Harris C G, de Vries A P, et al, "Quality through flow and immersion: gamifying crowdsourced relevance assessments," Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval (ACM), pp. 871–880, 2012.

[10] Powers, David Martin, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," Bioinfo Publications, 2011.

[11] Xiaoni Duan, Keishi Tajima, "Improving Classification Accuracy in Crowdsourcing through Hierarchical Reorganization," 2017 IEEE International Conference on Big Data (BIGDATA), pp. 4290–4292, 2017.