Effective Quality Assurance for Data Labels through Crowdsourcing and Domain Expert Collaboration

Wei Lee
National Cheng Kung Univ.
Taiwan
wlee@netdb.csie.ncku.edu.tw

Chien-Wei Chang
National Cheng Kung Univ.
Taiwan
cwchang@netdb.csie.ncku.edu.tw

Po-An Yang
National Cheng Kung Univ.
Taiwan
payang@netdb.csie.ncku.edu.tw

Chi-Hsuan Huang
National Cheng Kung Univ.
Taiwan
chihsuan@netdb.csie.ncku.edu.tw

Ming-Kuang Daniel Wu
Slice Technologies Inc.
San Mateo, California, USA
danielwu@slice.com

Chu-Cheng Hsieh
Slice Technologies Inc.
San Mateo, California, USA
chucheng@ucla.edu

Kun-Ta Chuang
National Cheng Kung Univ.
Taiwan
ktchuang@mail.ncku.edu.tw

ABSTRACT

Researchers and scientists have been using crowdsourcing platforms to collect labeled training data in recent years. The process is cost-effective and scalable, but research has shown that the quality of truth inference is unstable due to worker bias, work variance, and task difficulty. In this demonstration, we present a hybrid system, named IDLE (Integrated Data Labeling Engine), that brings together a well-trained troop of domain experts and the multitudes of a crowdsourcing platform to collect high-quality training data for industry-level classification engines. We show how to acquire high quality labeled data through quality control strategies that dynamically and cost-effectively leverage the strengths of both domain experts and crowdsourcing.

1 INTRODUCTION

Hand-annotated training data, such as ImageNet [1][2], have been the basis of many machine learning research. In recent years, crowdsourcing has become a common practice for generating training data [3], empowering researchers to outsource their tedious and labor-intensive labeling tasks to workers of crowdsourcing platforms. Crowdsourcing platforms provide large and inexpensive workforce for better cost control and scalability. However, the unstable quality of work produced by crowdsourcing platform workers is the main concern for crowdsourcing adopters.

Recent research by Zheng et al. [15] shows that the best truth inference algorithm is very domain-specific, and no single algorithm outperforms others in most scenarios. Sometimes an intuitive approach like an ExpectationâĂŞMaximization algorithm could be a practical solution. In the literature, research advances focus on handling task difficulty [7, 13], worker bias [9], and worker variance [10, 12]. Specifically, *task difficulty* describes the degree of ambiguity of a question for which an annotated answer is sought; whereas *worker bias* and *variance* model the quality of workers – describing how likely a worker gives a wrong answer, assuming all tasks have equal difficulty.

The ability to collect high-quality and stable training data (i.e. the inferred truth) is essential for powering many supervised algorithms. These algorithms are often the foundation for modern business solutions, such as search engine rankings [6], image recognition [2, 11, 14], news categorization [8], and so on. Even though research [15] has unveiled the challenges of crowdsourcing labeling, it is undeniable that cost-effectiveness and scalability make crowdsourcing an attractive approach to generate training data.

In this study, we present a practical end-to-end multilevel solution based on a hybrid strategy. On the first level, we collect cost-effective truth inference from crowdsourcing workers whose answers have potentially high bias and variance.

On another level, we train a group of domain experts who are expected to perform labeling tasks with low worker bias and variance due to the training and financial incentives they receive. Our trained experts are intimately familiar with our product category taxonomy as well as the guidelines for assigning the most appropriate product category label to any given product item. They are instructed to mark high-difficulty tasks as "*unsolvable*" to circumvent ambiguous cases.

We propose IDLE as a system to facilitate the automated collaboration between our well-trained domain experts and the crowdsourcing workers to deliver high-quality hand-annotated training data. The IDLE framework streamlines the workflow for generating high quality training data by automating data filtration (by crowdsourcing) and data relabeling (by in house domain experts). It also provides an integrated environment for managing training data generation tasks as well as for assessing quality of classification results generated by our product classification engine as described in details in section 2.3.

2 SYSTEM FRAMEWORK

Figure 1 shows the architecture of our system. There are 4 key components in our framework: (1) **Multilevel Worker Platform:** a system that assigns tasks to domain experts and various crowdsourcing platforms through **Adapters**; it also performs **Worker Quality Assessment** and **Answer Aggregation**. (2) **Sampling Strategy:** with a unified user interface, job requester

[1] http://www.image-net.org/

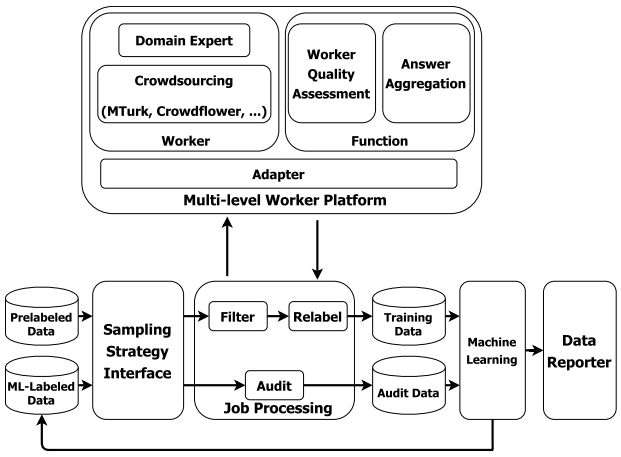

Figure 1: IDLE system framework

can choose among various sampling strategies. (3) **Job Processing:** job requester can launch jobs of various types. (4) **Data Reporter:** a dashboard showing the aggregated results from crowdsourcing and the improvement of the machine learning model.

2.1 Multilevel Worker Platform

With a unified interface, the job requester can submit a job through **Adapters** to various crowdsourcing platforms, such as MTurk[2] and Crowdflower[3]. Furthermore, the job requester can assign difficult labeling jobs to domain experts who sign into their **IDLE** account to label data. We also design a uniform **Function** interface for common features, such as **Worker Exclusion** and **Answer Aggregation**, across various crowdsourcing platforms.

Adapter: Using MTurk API[4] as a reference, we design the interface through which job requester can (1) launch a job, (2) stop a job, and (3) retrieve results. Adapters allow us to easily integrate with different crowdsourcing platforms without making significant changes to the user experience or the rest of the IDLE platform.

Answer Aggregation: Since answers returned by crowdsourcing workers are not always consistent and worker's quality varies (for example, master vs. non-master workers in MTurk), we have the challenge of inferring ground truth from the returned answers. To tackle the answer aggregation challenge, we implement three algorithms : **Majority Voting**[1], **Weighted Majority Voting**[4], and **Bayesian Voting**[5]. Through the provided interface, developers of IDLE platform can easily implement customized answer aggregation algorithms. Moreover, job requester can specify rules in the form of [*#answer*, *#yes*] for determining the final answer. The rule template is interpreted as seeking *#yes*/#answer level of consensus in total *#answer* number of answers. More elaborate answer aggregation strategies may be expressed through a sequence of rules. For instance, rules [3, 3] followed by [4, 3] together instruct the system to first seek unanimous consensus among 3 answers ([3, 3]). For questions whose answers fail to meet the first rule, the system needs to try again by soliciting an additional answer (*#answer*= 3 + 1) hoping to reach the specified 3/4 consensus.

[2] https://www.mturk.com/
[3] https://www.crowdflower.com
[4] https://aws.amazon.com/documentation/mturk/

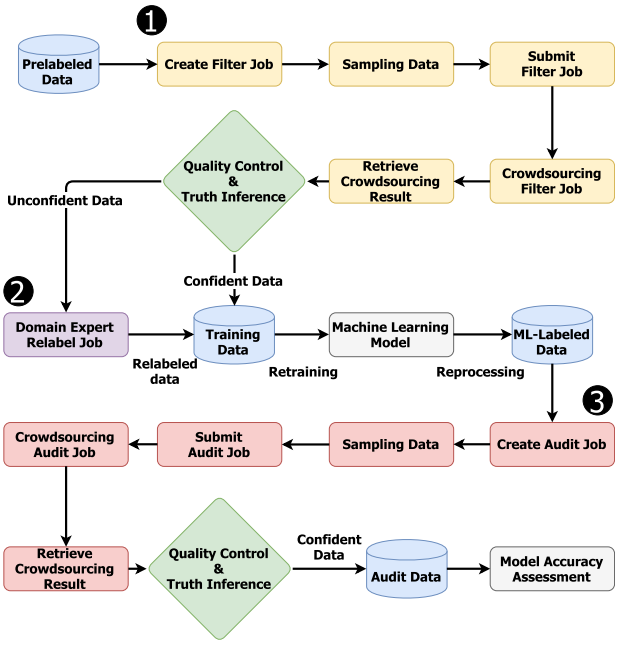

Figure 2: Control Flow

Worker Quality Assessment: Worker's quality varies widely on crowdsourcing platforms. The fact that this quality is unknown to us in advance makes it even more important to assess worker's quality. In IDLE, we randomly select questions from a curated pool of questions with ground truth answers (called *golden tasks*) to estimate worker's quality. We apply two strategies: (1) **Qualification Test:** BEFORE performing the job, workers must first pass the *golden tasks*; (2) **Hidden Test:** mixing the *golden tasks* with the regular job questions, and we assess worker's quality based on the *golden tasks* AFTER the job is completed. In our platform, job requester may use either one or both strategies to estimate worker's quality.

2.2 Sampling Strategy Interface

There are many statistical sampling techniques. In IDLE, we design the general interface for developers to implement the required sampling strategies. The goal is for job requester to obtain sampling data from a diverse data set. We incorporate two hierarchical sampling strategies for IDLE in this version: (1) data clustering followed by stratified sampling; (2) topic modeling followed by stratified sampling.

2.3 Job Processing

As illustrated in Figure 2, there are three types of jobs in IDLE: Filter jobs, Relabel jobs, and Audit jobs.

Filter Job: A small set of data are sampled from prelabeled data and sent to crowdsourcing platform for confirming their labels. Questions of a filter job are presented either as yes/no questions (e.g. Does the given label match this datum?) or multiple choice questions (e.g. Which of the following labels best matches this datum?). *Golden tasks* questions used for excluding poor-quality workers are also included in the filter job. After the workers submit their answers, the results are collected through the answer aggregation techniques described above in section 2.1. The results that are identified with high confidence level by our answer aggregation algorithm become new training data for the machine

learning model. The remaining (filtered-out) data are treated as mislabeled data and become input data for relabel jobs which are handled by domain experts as described in section 1. We expect data that are trivial for crowdsourcing workers can quickly pass through and data that are difficult to label are filtered out, hence, the name 'Filter' job. The cost of domain experts is much higher than that of crowdsourcing workers, which is why it is more cost-effective to have crowdsourcing workforce perform filter jobs on large number of trivial questions first and leave a small number of more challenging relabel jobs to domain experts.

Relabel Job: As mentioned above, mislabeled data are automatically collected and made available in IDLE framework to domain experts for relabeling. These domain experts are trained to assign correct labels to the provided data. Thus, data relabeled by domain experts do not require quality control or truth inference measures before they become training data for the machine learning model. With that said, there might be some data that even domain experts cannot label, thus are regarded as rejected data and recorded for further analysis.

Audit Job: After the filter job and the relabel job are done, all the sampled data are either identified as new training data for the machine learning model or as rejected data for analysis. After retraining the machine learning model in our product classification engine with the new training data, the model reprocesses data and updates the product category labels. Up to this point, all the efforts for enhancing the performance of the machine learning model are completed. We then assess the accuracy of this retrained model with an audit job. Similar to a filter job, a small set of data are sampled and sent to crowdsourcing platform for identifying correctly labeled data. We then apply our answer aggregation algorithm to identify data with high confidence level and calculate model accuracy while the mislabeled ones are simply discarded.

2.4 Data Reporter

To maximize the effectiveness of crowdsourcing and minimize the costs, there are certain questions that analysts would be curious about: for example, what is the ratio of filter job questions that need to be handled by a relabel job? The data reporter is a data visualization dashboard for administrators and analysts to evaluate the effectiveness of crowdsourcing and the performance of the machine learning algorithms. There are two parts of data reporter: (1) **Crowdsourcing Report** (2) **Machine Learning Model Report**.

Crowdsourcing Report: The purpose of crowdsourcing report is to evaluate the effectiveness and the efficiency of crowdsourcing. Therefore, it is designed to provide insights, such as the answer distribution and processing time. The crowdsourcing report includes the stats and results of crowdsourcing jobs. For filter jobs and audit jobs, the stats would include the ratio of YES vs. NO besides job completion time. For relabel jobs, the report would display the ratio of relabeled rate and job completion time. To estimate the overall performance of crowdsourcing for each job, the dashboard would also show the ratio of mislabeled data vs. data with high confidence level in addition to the total processing time.

Machine Learning report: The machine learning report is used to track the rate of improvement for the machine learning model. Thus, the report shows not only the history of accuracy for the model but also the ratio of data processed through crowdsourcing.

Algorithm 1 Adaptive Task Assignment

Require: Label confidence scores of prelabeled data $PC = (pc_1, pc_2, \cdots, pc_n)$, task confidence threshold θ, worker set $\mathcal{W}$, budget B

Ensure: $(|W_1|, \ldots, |W_n|)$

$(W_1, \ldots, W_n) = (\emptyset, \emptyset, \ldots, \emptyset)$

$PC_{order} = Order(PC)$ ▷ reverse sort PC

for all $pc_i \in PC_{order}$ **do**

  $(W_i^*, C_i^*) = \underset{Conf(pc_i, W_i) \geq \theta, W_i \in \mathcal{W}}{\arg\min} Cost(W_i)$

  $B = B - C_i^*$

  if $B \leq 0$ **then**

    break;

return $(|W_1|, \ldots, |W_n|)$

3 ADAPTIVE TASK ASSIGNMENT (ATA) ALGORITHM

To best utilize available resources (such as a given budget), we further study the mechanism to adaptively assign the number of workers for each crowdsourcing task. In practice, the equal assignment of workers per task is not the most effective approach to achieve satisfactory quality when dispatching large-scale crowdsourcing tasks. Advances in machine learning research provide powerful labeling capabilities in predicting the label for each task along with a confidence score. Keeping the confidence granularity and the importance of a task in mind, we adjust the number of workers for each task in pursuit of overall optimization. Therefore, we propose **Adaptive Task Assignment Algorithm** to optimize the crowdsourcing resource utilization.

In Algorithm 1, we outline the steps to determine the number of workers for each task. Each crowdsourcing task consists of one single product item with category label predicted by our product classification engine. Initially, we are given prelabeled data and *label confidence score* pc_i of each task (provided by our supervised learning algorithms). *Label confidence score* ranges from 0 to 1. Next, we assign workers to crowdsourcing tasks that are reverse ordered by their label confidence scores. Here we introduce a threshold called θ, to ensure that the task confidence score of each task (calculated by the $Conf(pc, W)$ function described below) exceeds θ eventually. In addition, we use $Cost$ function to represent the cost for a worker set in search for the optimal worker set W_i^*, where the total cost C_i^* of the worker set is minimal for a task's confidence score to exceed θ. When the sum of cost C_i^* is equal to budget B, the algorithm terminates and returns the optimal number of workers $|W_i^*|$ to assign to each task.

The task confidence is calculated based on the given *label confidence* of prelabeled data and the quality of the assigned worker.

$$Conf(pc, W) = \max_{a \in \{Yes, No\}} Conf_a(pc, W)$$

We use label confidence score of prelabeled data pc_i and worker's quality q^w to calculate the Bayesian probability [5]. Worker's quality $q^w \in [0, 1]$ is the probability that the worker answers the correct label. As an example, assuming the supervised learning algorithm provides the prelabeled data with label confidence score of 0.45; the confidence threshold $\theta = 0.75$, and we have four workers with quality scores 0.5, 0.8, 0.6 and 0.4 respectively (assessed through golden tasks described above): Initially, we

pick the first two workers in the first iteration, and their answers are *No* (rejecting the label assigned by machine learning) and *Yes* (confirming the label assigned by machine learning). We can calculate the task confidence score as the following:

$$Conf_{Yes} \propto 0.45 \cdot (1 - 0.5) \cdot 0.8 = 0.18$$
$$Conf_{No} \propto (1 - 0.45) \cdot 0.5 \cdot (1 - 0.8) = 0.055,$$

which lead to

$$Conf_{Yes} = \frac{0.18}{0.18 + 0.055} = 0.76$$
$$Conf_{No} = \frac{0.055}{0.18 + 0.055} = 0.24$$

Since $Conf(pc_i, \{w_1, w_2\}) = 0.76$ exceeds $\theta = 0.75$, it is not necessary to assign additional workers to this task. In other words, the ATA algorithm can confidently confirm the machine-assigned label (*i.e,* concluding the answer *Yes*) based on answers from merely two crowdsourcing workers.

4 DEMONSTRATION

Our system implementation of IDLE is based on Flask (a Python web framework) and ReactJS. We use distributed task queue Celery to handle asynchronous tasks, such as data sampling and crowdsourcing result retrieval. At the time of writing, IDLE is undergoing beta testing at Slice Technologies, and we are actively improving the system. The screenshots for the following demonstrations can be found in the GitHub repo[5] of IDLE:

Data Ingestion: Job requester first selects prelabeled data to ingest into IDLE by picking a category from a data set, uploading a file, or querying the database with SQL commands. Afterwards, job requester chooses a sampling strategy and sample count for filter job creation.

Crowdsourcing Job Configuration: After sampled prelabeled data are ingested, job requester configures parameters of a crowdsourcing task, *e.g,* reward per assignment and number of assignments per HIT (Human Intelligence Task, a term to denote a single crowdsourcing task on Amazon Mechanical Turk platform). To estimate worker's quality, the system can also be configured to automatically include golden tasks (quality control questions) in the job.

Crowdsourcing Job Creation: Configurations of a crowdsourcing job are reviewed and confirmed prior to job creation. After publishing a job to crowdsourcing platform, IDLE automatically performs answer aggregation.

Stats Reporting: Job status and other job-related information are displayed on the main IDLE dashboard. History of the machine learning model's performance is also available for evaluation purposes.

5 CONCLUSIONS

In this study, we present IDLE, an integrated data labeling platform consisting of two main features: quality assessment and answer aggregation. The platform incorporates the adaptive task assignment algorithm, an algorithm that enables us to provide a cost-effective process for training data generation. This streamlined process alleviates the impact of highly difficult tasks as well as of crowdsourcing's worker bias and worker variance. As a result, IDLE system empowers researchers to effectively and efficiently collect high-quality training data through collaboration between in-house domain experts and external crowdsourcing workers in an automated and integrated manner. IDLE provides us an integrated platform for generating large amount of training data with higher quality, faster speed, and optimal cost.

[5] https://github.com/slice-ncku/IDLE

REFERENCES

[1] Caleb Chen Cao, Jieying She, Yongxin Tong, and Lei Chen. 2012. Whom to Ask?: Jury Selection for Decision Making Tasks on Micro-blog Services. *Proc. VLDB Endow.* (2012), 1495–1506.

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

[3] Anhai Doan, Raghu Ramakrishnan, and Alon Y. Halevy. 2011. Crowdsourcing Systems on the World-Wide Web. *Commun. ACM* (2011), 86–96.

[4] Chien-Ju Ho, Shahin Jabbari, and Jennifer Wortman Vaughan. 2013. Adaptive Task Assignment for Crowdsourced Classification. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28 (ICML'13)*. JMLR.org, I–534–I–542.

[5] Panagiotis G. Ipeirotis, Foster Provost, and Jing Wang. 2010. Quality Management on Amazon Mechanical Turk. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*. ACM, 64–67.

[6] Gabriella Kazai. 2011. In search of quality in crowdsourcing for search engine evaluation. In *European Conference on Information Retrieval*. Springer, 165–176.

[7] Fenglong Ma, Yaliang Li, Qi Li, Minghui Qiu, Jing Gao, Shi Zhi, Lu Su, Bo Zhao, Heng Ji, and Jiawei Han. 2015. Faitcrowd: Fine grained truth discovery for crowdsourced data aggregation. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 745–754.

[8] Richard MC McCreadie, Craig Macdonald, and Iadh Ounis. 2010. Crowdsourcing a news query classification dataset. In *Proceedings of the ACM SIGIR 2010 workshop on crowdsourcing for search evaluation (CSE 2010)*. 31–38.

[9] Robin Wentao Ouyang, Lance Kaplan, Paul Martin, Alice Toniolo, Mani Srivastava, and Timothy J. Norman. 2015. Debiasing Crowdsourced Quantitative Characteristics in Local Businesses and Services. In *Proceedings of the 14th International Conference on Information Processing in Sensor Networks (IPSN '15)*. ACM, 190–201.

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

[11] Padhraic Smyth, Usama Fayyad, Michael Burl, Pietro Perona, and Pierre Baldi. 1994. Inferring Ground Truth from Subjective Labelling of Venus Images. In *Proceedings of the 7th International Conference on Neural Information Processing Systems (NIPS'94)*. MIT Press, 1085–1092.

[12] Peter Welinder, Steve Branson, Serge Belongie, and Pietro Perona. 2010. The Multidimensional Wisdom of Crowds. In *NIPS (NIPS'10)*. Curran Associates Inc., 2424–2432.

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

[14] Tingxin Yan, Vikas Kumar, and Deepak Ganesan. 2010. CrowdSearch: Exploiting Crowds for Accurate Real-time Image Search on Mobile Phones. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys '10)*. ACM, 77–90.

[15] Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. 2017. Truth Inference in Crowdsourcing: Is the Problem Solved? *Proc. VLDB Endow.* (2017), 541–552.