Label TAG (Temporal AGgregation): Capturing Dynamic Learning Ability in Label Aggregation

Connor Douglas, Calvin Isley, Aditya Krishnamachar
{c.douglas, calvin.isley, akrishnamachar} @ wustl.edu

Abstract

Label aggregation is an integral part of crowdsourcing activities. Typical label based aggregation models, such as Expectation-Maximization (EM) or Singular Value Decomposition (SVD), attempt to capture a worker's true ability, assigning each worker a weight based on the proportion of labels they correctly assign. These models assume static worker abilities, and are effective predictors in such cases, however in some situations, this assumption does not hold. Indeed, worker abilities are often dynamic, with true worker accuracy changing over time. In these situations, algorithms that assume static worker ability do not create the most accurate model of worker abilities. We present a new model, Label TAG, for capturing crowdsourcing workers' dynamic learning ability in binary label settings. We show the theoretical mathematical validation of our model, as well as an empirical validation over synthetic datasets, and present our results. We find that TAG outperforms the classical EM model in most cases of dynamic ability and is robust against non-learning. We also discuss next steps for the model, including how to optimize the model in situations of unknown ability.

Introduction

Data is at the foundation of modern society, with machine learning algorithms acting as the backbone of nearly all technological processes today. Disparate industries are increasingly leveraging the predictive power of machine learning to augment processes and make them more efficient. Central to this change is the use of *supervised machine learning*, and in particular *classification algorithms*, which train models on known, labeled data and then attempt to classify labels on unseen instances in a similar way. By this process, these models are only as strong as the data they are trained on. The quality of data here is essential, yet establishing the ground truth of instances is a non-trivial problem. To address this, humans are used to establish the ground truth of instances. In order to achieve labelling en masse, we turn to crowdsourcing, a way of employing many online workers to perform simple labeling tasks.

Crowdsourcing has become a fairly standard, cost-effective method of labeling massive datasets, where people unknown to the requester manually label images, text, or other mediums with the desired (ideally, the correct) label. This process, outsourcing tasks to these workers, has drastically risen in popularity over time. It has become an integral part of a great variety of tasks that require large amounts of hand-labeled information.

Within this ever-growing field sits the central plank of label aggregation. The process of accumulating all the disparate labels, processing the labels, and understanding worker abilities all sit under the catch-all term that is label aggregation. Nearly all previous work in this area assumes that all worker abilities are constant and unchanging throughout the duration of their disparate labeling processes. Thus, even if a worker improved dramatically; say from an accuracy of 0.50 from $t \in [0, 25)$ to 1.00 from $t \in [25, 50)$, this methodology would assign them an average ability of 0.75 over all 50 timesteps.

This approach, assuming that workers have static and unchanging abilities throughout the entire cycle, in some sense glances over this crucial step in label aggregation. Typical aggregation methods assign weights to workers based upon their accuracy. But if their accuracy is assumed to be constant over the entire time period, this could lead to damaging estimations of labeling. Take our previous example of a worker improving dramatically across 50 time steps, for example. If they were among the highest worker accuracies across the entire labeling set, it is likely their labels would have an outsize weighting, given their higher accuracy. But for $t \in [0, 25)$, their labeling performs extremely poorly, at no better than a random guess! This method of worker accuracy labeling is thus in need of a robust upgrade.

Related Work

Previous work in this area has mainly been in the context of predicting or understanding more about worker accuracy and reliability. This, once again, is generally in the context of one average metric that captures this 'reliability', and not at different time steps.

Qiu *et al.* (2018), however, proposed a model that is substantially more dynamic - making use of gold standard evaluation and peer consistency evaluation methods to track and update worker performance. The researchers adjusted the proportion of these two methods based on an estimated distribution of overall worker behavior - essentially, where workers swung from acting reliable to performing malicious behavior. They place specific emphasis on this adversarial activity, as it has become more frequent and relevant in the context of label collection. They used Mechanical Turk to source worker data and methodology. The approach specifically focuses on "improving workers' quality control on the-fly" [2]. Their model dynamically adjusts to take into account the gold evaluations performed (in the midst of the other assigned tasks); and showed, using the Mechanical Turk platform for worker data, that this approach does better than other comparable methods to track worker reliability.

In the context of improving worker output, and finding ways by which to measure the increase in this output, we investigated an approach that attempts to constructively improve worker accuracy on a rolling basis (throughout the tasks they are asked to complete). The researchers, Drapeau *et al* (2016), presented an algorithm that required workers to either (1) justify a choice they made or (2) reconsider a decision after reading discussions advocating for the opposite choice, written by a fellow worker. This attempt to get workers to rethink or show critical understanding of their choices appeared to be successful. The authors showed that this argumentation improves worker accuracy by 20 percent (study performed on the Mechanical Turk crowdsourcing platform); and this workflow appears to outperform simple majority voting in every scenario. We note that this model definitively shows that workers are able to improve their performance over a period of time, when placed in the right environment or given the correct stimuli.

This approach is critical to our methodology, as our model also involves performing calculations at each time step for the worker data. Additionally, we could, for example, use the results of this paper to measure differences in workers' reliability over a long period of time. Assuming that the observed data lines up with our expected findings that labeling accuracies + reliability increases over time , we could then refine our proposed model (a temporal weighting of worker accuracy) to be consistent with these observed trends.

The majority of work in the field treats individual workers as independent and identically distributed random variables (i.i.d). However, a new model was proposed by *Jung et al.* (2014). They created a time-series label prediction model for crowdsourcing work, with the overall goal of improving the quality of labeling. It takes into account workers' so-called 'accuracy patterns' (a gradual increase in accuracy or a sharp dropoff in performance would be two examples).

In the same sub-area as our proposed work is research done by Hata *et al.* (2016). Their findings were actually counter to our modeling - observing that worker quality is stable over a long period of time. The tasks they tested, however, were quite simple ones: image descriptions, question-answer pairings, and binary verification. We understand that this is a significant body of work, but also note that the tasks are primarily simplistic ones.

Domnez *et al.* (2010) proposed a methodology for handling temporal differences in labeling accuracy and overall reliability for multiple workers. Their model has a Sequential Bayesian Estimation framework in order to perform two tasks. First, learn a worker's expected accuracy at a time step t; and second, to evaluate which workers to survey for a label. In brief, the model they present estimates the *expected* worker accuracy at each time step, performing sequential Bayes updates for every t. The researchers' analysis demonstrated that their method far outperforms a simple majority voting algorithm (majority voting takes into account all worker inputs (+1, -1) per timestep and takes the sum to predict the correct label). Two key assumptions that the authors made in this analysis was that (a) the rate of change of a worker's learning ability (or at least the bound of this rate of change) was known and (b) this rate of change was the *same* for every labeler. We take (a) to be true in our analysis, but we do not require that this be known when implementing our algorithm, noting that a person's learning ability can be estimated by linear, logarithmic, or similar functions. We also note that a person's learning ability is certain to be bounded by some number (they cannot learn at an infinite rate!), so we are confident that making the same assumption as Domnez, et al. will result in reliable and trustworthy results. This is explained in detail in the following analysis.

Model

The Label TAG model follows the expectation-maximization framework, iteratively generating results and assigning weights until convergence is reached with the results. The pseudocode for the general algorithm is shown below.

Algorithm 1 Label TAG

```
 1: procedure TAG(data, k)
 2:     results ← majorityVote(data)
 3:     oldResults ← results
 4:     convergence ← false
 5:     while ! convergence do
 6:         weights ← getWeights(data,results,k)
 7:         results ← getResults(data,weights)
 8:         if results = oldResults then
 9:             convergence ← true
10:         else
11:             oldResults ← results
12:     return results
```

Weight Assignment

We assign weights to every worker at each of timestep of their labeling process. We store these weights in a dictionary indexed by *worker* and *task*. For each worker, we begin by sorting all tasks completed by timestep. Then, we compare worker responses against results to generate a binary accuracy value for each timestep, thus creating an timestep-indexed array of accuracy values for the worker. We refer to this array as, A. Now, we iterate across timesteps and calculate the window indices at each timestep. The rule for window size at timestep t is: $i_{lower} = max(0, ceiling(t-k/2))$ and $i_{upper} = min(n, ceiling(t + k/2))$, where n is the total number of tasks a worker has completed. Then we create a window, $W = [i_{lower}, i_{upper})$.This gives us a window of size k centered around timestep t. It is important to note that the window is truncated near $t = 0$ and $t = n$

to maintain valid indexing in corner cases. See Figure 1 for a visualization of weight assignment. Now, we calculate an average accuracy from array A on this window W , such that $\varphi = mean(a[W])$. From this value φ, we compute a weight such that $weight = 2\varphi - 1$. By making this transformation, we can count adversarial accuracies (accuracies that are worse than a random guess, $\varphi < .5$) negatively. This weight is then stored in the weight dictionary $weights_{worker,task} = weight$.

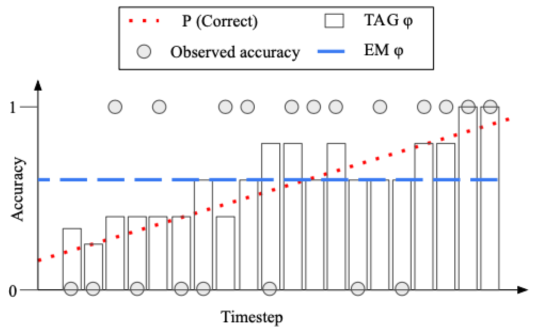

Figure 1: An example of weights by TAG for each timestep versus EM assigned weight across a linear learning function with $k = 5$.

Result Aggregation

Result aggregation in this method is nearly identical to traditional EM approaches. For each *task*, determine all workers who labeled this item. The equation below signifies how to assign labels given worker responses, *R*, and weights,*W'*.

$$label_{task} = sign(\sum_{workers} R_{worker,task} W'_{worker,task})$$

Note, we use *W'* locally here for brevity of notation, it is not to be confused with *W* used to denote a window elsewhere in the paper.

Theoretical Validation

Computational Expected Error on Label TAG Algorithm

In calculating our expected error, we generally define the window as follows:

$$W = \{n - \frac{k}{2}, n - \frac{k}{2} + 1, ..., n + \frac{k}{2} - 1, n + \frac{k}{2}\}$$

rounding up to ensure integer values and ensure constant window size *k*. In edge cases, we truncate the window such that indices are within the bounds $[0, n)$ as described above. Having defined our window W, we calculate our expected mean absolute error for a window size *k*, as follows:

$$MAE_k = \frac{1}{n} \sum_{t=0}^{n-1} \sum_{i=0}^{k} |\frac{i}{k} - f(t)| P(X = i)$$

Where $f(t)$ is the worker's learning function, returning their true accuracy at time t, $\frac{i}{k}$ is a workers potential accuracy assigned at timestep *t*, $P(X = i)$ is the probability of them receiving that accuracy at that timestep.

We define $P(X = i)$ with a Poisson Binomial distribution, with the PDF:

$$P(X = i) = \sum_{N \in S} \prod_{l \in N} f(l) \prod_{j \in N^c} 1 - f(j)$$

With S as the set of all possible subsets of W of size i, and N^c is the complement on N, or S/N. For example, if $W = \{1, 2, 3\}$ and $i = 2$, $S = \{\{1, 2\}, \{1, 3\}, \{2, 3\}\}$. If $S = \{1, 2\}$, $S^c = \{3\}$. In words, $P(X = i)$ is the probability that at a given timestep, the worker will label i labels correctly.

At every timestep, a worker has $k + 1$ possible skill levels and their true skill level, $f(t)$. The probability they are of skill level $\frac{i}{k}$ is calculated by $P(X = i)$, and thus we find the total expected error for a timestep *t* by subtracting potential skill from true skill, times the probability of having said potential skill level. We do this for every time step, and average the total to find mean absolute error.

Note this computation gets very expensive quite quickly. For a window size > 20, for example, $|S| > 10^5$ when $i = 10$. While more economical computation methods exist [1] for finding $P(X = i)$, we find significant decreases in individual worker accuracy prediction error with window sizes < 15, and thus we do not need to explore higher k values.

Bounds on EM Error

The distribution of potential worker accuracy values assigned by EM can effectively be calculated by using a window size of *n* in the Poisson Binomial distribution. Unfortunately, as previously stated, the Poisson Binomial distribution becomes intractable at large sets of *W*. So, we employ a trick to calculate the lower bound of error on a worker's learning function *f(t)* for a weight assigned by EM. We do this in order to compare the expected error of TAG at different window sizes to the baseline EM.

We motivate the calculation of this lower bound on the error of EM by knowing that in the best case, EM will choose a value *b** to represent a worker's accuracy that minimizes the sum of absolute error to all points in *f(t)*. The true distribution of possible weights cannot better minimize MAE than picking worker accuracy value *b** with probability *p(b*)=1*.

To choose the optimal accuracy value *b** that minimizes MAE, we perform a simple optimization process on a related equation. We find a *b* value that minimizes MAE as a function of *b* by minimizing the Sum of Squared Errors (SSE) with respect to *b*. This function is differential and obtains a minimum at the same value as MAE. The SSE is represented below:

$$SSE = \sum_{t=0}^{n-1} (f(t) - b)^2 \tag{1}$$

To find the accuracy assignment that minimizes this equation, *b**, we differentiate the function with respect to *b* and set this to equal zero.

$$\frac{dSSE}{db} = \sum_{t=0}^{n-1} -2(f(t) - b) = 2nb - 2\sum_{t=0}^{n-1} f(t) \tag{2}$$

$$0 = 2nb^* - 2\sum_{t=0}^{n-1} f(t) \tag{3}$$

$$b^* = \frac{\sum_{t=0}^{n-1} f(t)}{n} = \bar{f} \tag{4}$$

We see that $b*$ is equal to the mean value of the learning function on the timestep domain $t \in [0, n-1]$. We verify this is a global minimum because the function is convex and concave up as the second derivative is positive on the whole domain. This value, $b*$, minimizes MAE, so EM cannot have an expected MAE value lower than that computed from $b*$. By plugging in $b*$ into MAE, we achieve the closed form lower bound on expected MAE for EM.

$$MAE = \sum_{t=0}^{n-1} |f(t) - b^*| = \sum_{t=0}^{n-1} |f(t) - \bar{f}| \tag{5}$$

Evaluation of Window Sizes on Learning Functions

To demonstrate, consider the following example of assement of error in calculating an individual workers weight. Let worker i has the following learning function:

$$f(t) = \frac{1}{2} + \frac{1}{2}\frac{t}{100}$$

Assume this worker labels 100 tasks, or in other words, reaches their skill cap and stops labeling as soon as they do. We compute the MAE for increasing values of k, as seen in the table below:

k	MAE_k
1	.42
2	.30
3	.21
5	.17
10	.11
12	.11
15	.10

From (4), we observe b^* in this case to be .75, which we can then plug into (5) to find the lower bound on EM's MAE to be .25. With *k* values ≥ 3 in this table, we show TAG must have a lower expected MAE than EM. This approach could be run exhaustively on any learning function to see when, and at what *k* values, TAG could better match a worker's true accuracy in expectation than EM. For the sake of brevity, however, we employ this example to show a concrete instance of TAG theoretically outperforming EM in capturing worker ability and use it to justify proceeding to experimental validation.

Dataset Generation

Due to the novelty of temporal label aggregation, we were unable to find a dataset on which to empirically validate our model. Thus, we synthetically generated data sets which modeled user improvement over time.

We generated *simple* data as proof of concept and eventually dynamic, realistic data for our experimental validation. For simple datasets, we selected a number of tasks n_s, a number of workers l_s, and a learning function $f(t)$. Every worker labels every task, and reaches the peak of their skill-cap at their last timestep. While these datasets do not best reflect real worker behavior, they represent a hypothetical near-best-case scenario, useful for demonstrating the value of TAG.

In our more robust *dynamic* data sets, we defined some number of tasks N_d which we wanted to be labeled with workers reaching their skill caps at M_d, and took the following steps to best simulate real worker behavior, based on prior research and traditional crowdsourcing practices:

- Tasks are assigned to workers randomly, and the number of tasks an individual worker completes is random.
- Workers will self select out of a task if their accuracy is too low (high rejection rate) [4]. To achieve this, we limited the range of worker skill discounts to $d \in [.7, .9]$.
- The number of workers is not constant, but the number of labels per task is.
- Workers share a common learning function with unique discount factors.
- Workers may continue to work after reaching their skill cap.

Following these guidelines, we generated datasets with varying values of N_d and M_d. We then created a large number of such datasets for each learning function, and selected values of n_d and m_d to enhance accuracy in our assessment of algorithm accuracy.

Learning Functions

The literature on how humans learn over time is relatively sparse. Quantifying knowledge is no simple task, though luckily in binary label aggregation, workers have two options: right or wrong. Existing literature has provided reason to believe temporal information proves beneficial in crowdwork problems, specifically noting that learning ability can be approximated and accurately bounded [2]. Furthermore, in some cases, workers have displayed dynamic abilities suggesting a window based solution may lead to enhanced prediction accuracy [3]. In some examples, we define a m as the "mastery point," where workers will stop improving and continue to work at exactly their skill cap. In cases where m is undefined, $m = n$, the number of tasks a worker completes in total. We define a learning function $f(t)$ for a worker as follows:

$$f(t) = \frac{1}{2} + \frac{discount * g(t)}{2}$$

where $g(t)$ is a growth function returning a value $x \in [0, 1]$, increasing to 1 as $t \to m$ and exactly 1 when $t >= m$.

For baseline comparisons to EM, we tested TAG on a dataset with no worker skill improvement. For our primary tests, we assumed worker skill grew linearly, though we also tested TAG on logarithmic, asymptotic, and sigmoidal skill growth. Figure 2 displays a example with a linear learning function and a skill cap.

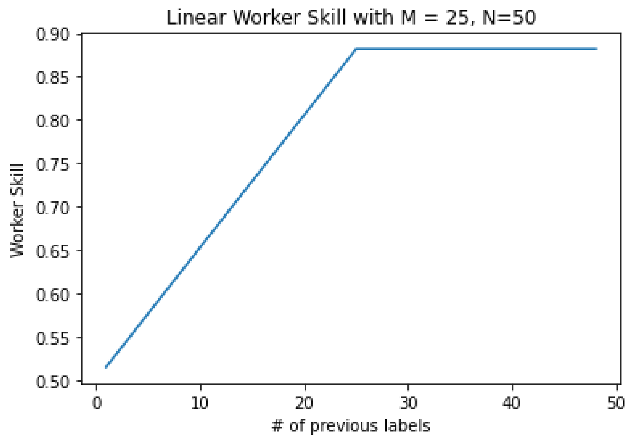

Figure 2: An individual worker's skill over time, mastering the task at $t = 25$, completing an additional 23 tasks.

Experimental Results

In this section, we compare the accuracy of TAG to EM on generated data. In our simple datasets, we use 5 workers per task. As a reminder, dynamic datasets have variable numbers of workers. This value was chosen as with fewer workers per task, it becomes more important to accurately gauge the ability of each worker. We omit SVD and majority vote results as these aggregation methods tend to perform poorly without large numbers of workers per task and therefore would not make for a fair comparison. EM is the most similar aggregation algorithm and as such is used as the baseline. Confidence intervals are plotted at the 90% level for both TAG and EM in all plots.

Simple Data

Here, we analyze our performance on a simple dataset, following a sigmoidal learning function. The function used here is shown below. In simple datasets, we exclude a mastery point m.

$$f(t) = \frac{1}{2} + \frac{1}{2(1 + e^{-1(t-\frac{n}{2})})}$$

with n as the total number of labels per worker. This function represents a worker who has some stagnant ability, goes through some period of rapid learning, and settles in to some higher, stagnant ability. While not representative of most learning functions, this represents a near best case scenario for TAG, with a high degree of variance in worker ability over time. We still make some assumptions of worker ability, assuming their skill starts at .5, as opposed to starting at 0. While TAG would outperform EM significantly more in the latter situation, it is unrealistic to have a worker go from entirely adversarial to entirely cooperative, while being able to discern a correct label the whole time.

In figure 3, we see the performance of TAG and EM on a dataset with 50 tasks per worker. We note TAG slightly outperforms EM at intermediate window sizes, which is in line with our expectations, where low k values result in large MAE values from a learning function, and where k values near the number of tasks per worker results in an algorithm that closely resembles EM.

We contrast the performance at 50 tasks per worker with a slower, longer learning period in figure 4. In this situation, a worker labels 250 tasks, while experiencing a similar learning function as before. Here, we see that TAG *significantly* outperforms EM at intermediate window sizes, with over a 10 point jump in accuracy. It is clear that TAG performs better on longer learning periods, which is validated through MAE calculations.

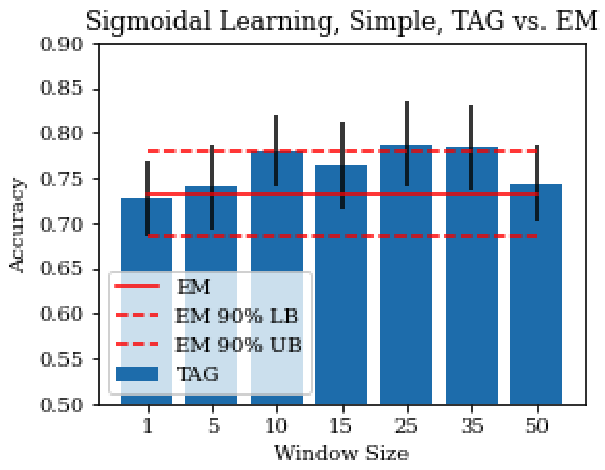

Figure 3: Sigmoidal learning, 50 tasks per worker

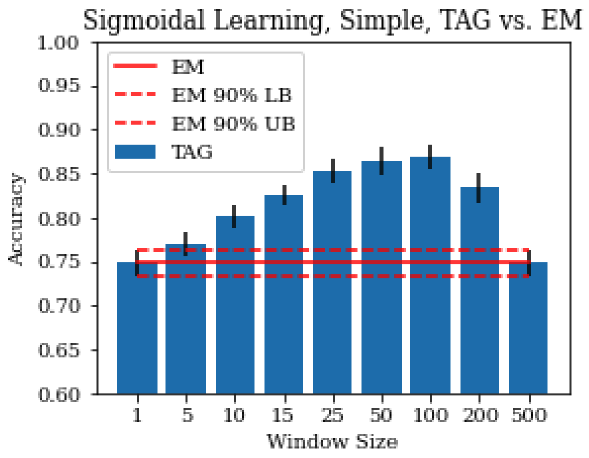

Figure 4: Sigmoidal learning, 250 tasks per worker

Dynamic Data

We move on to datasets generated using more sophisticated methods, which promise to better capture the dynam-

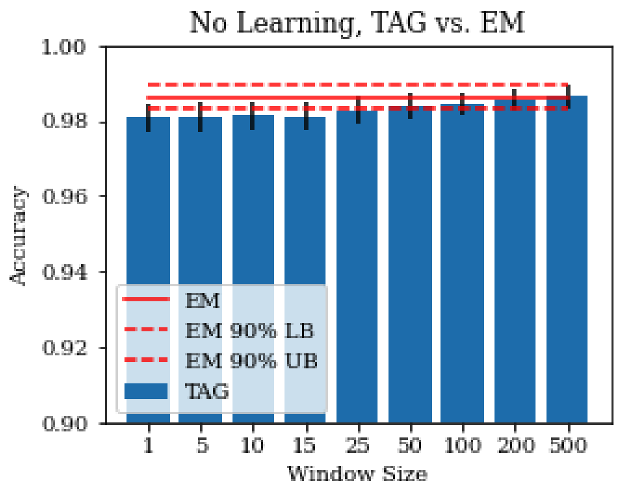

Figure 5: No learning, dynamic dataset generation.

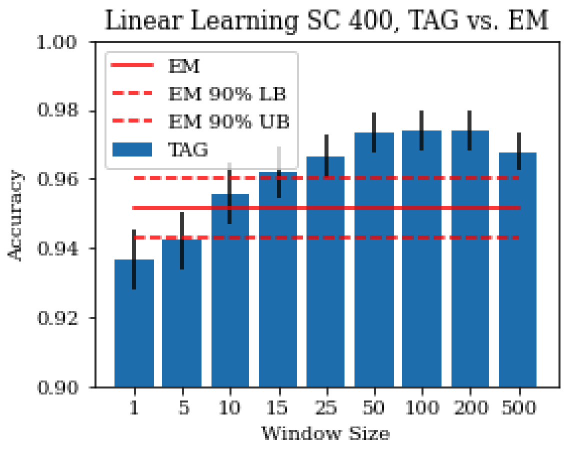

Figure 6: Linear learning, dynamic dataset generation.

ics of real-world label aggregation. These datasets leverage all methods outlined in the data generation section to create datasets that best mimic real crowdsourcing data.

The first set we try against has workers experience no learning, where we would expect our model to underperform against EM. Per our expectations, in figure 5 we see that TAG does underperform when compared to EM, but that this difference is only slight. Indeed, at intermediate window sizes of 100 and 200, TAG achieves near parity with EM. This is promising as it signifies that even in the worst case for TAG, where worker abilities truly are static, there is only a marginal performance hit.

Moving on to datasets drawn from workers who experience learning, we witness that TAG again outperforms EM. In figure 6, data is generated with workers reaching a skill cap at $t = 400$ and continuing to work past then dynamically, as laid out previously. We again witness significant outperformance of EM at intermediate window sizes of $k = 50, 100,$ and 200. This result is incredibly promising in that it evidences the ability of TAG to be robust in more realistic settings of learning.

Discussion

With Label TAG, we set out to find an algorithm that can more incisively capture worker ability in cases where this shifts over time. Through theoretical validation, we show how to calculate in what situations TAG will better be able to capture true worker ability in expectation compared to EM, and we provide an example to illustrate our case. Through experimental validation, we show TAG is robust in cases of non-learning and outperforms baselines in situations where learning occurs. We also find that TAG performs better in situations of longer, slower learning.

With these findings, we conclude that implementing TAG can squeeze extra performance out of label aggregation in many situations. Specifically, TAG will likely outperform when aggregation processes have a low number of workers per task and when there is reason to believe worker ability is dynamic over time. These results may prove especially helpful when adversarial gaming occurs. If, say, a worker at any point alternates from benevolent and accurate to adversarial (intentionally labelling incorrectly), we have evidence to suggest that TAG could pick up on this change to generate considerably higher overall accuracy.

TAG is a parameterized algorithm, however, unlike other models, which can have implications on its performance. From our figures, we observe the performance of TAG to be roughly parabolic in terms of k. The iterative nature of an expectation-maximization based algorithm makes it impossible to precisely solve an optimization problem in terms of k to maximize accuracy. In cases of a known learning function, our theoretical model of capturing worker accuracy can be tried at different k values to find an optimal k. In cases of unknown learning functions, however, we qualitatively put forth rules for choosing an optimal k value. We find intermediate k values tend to work best, at roughly or just below $\frac{1}{2}$ of the total length of worker period n. These values seem to best solve the tradeoff of capturing trends in learning over time and making a tighter distribution of possible outcomes at a given timestep. These values also produce the most robust results when workers do not, in fact, experience learning. To precisely choose a k value requires an optimization over various parameters that define the dataset, including the learning function, the number of total tasks, the number of workers, and the skill cap.

Having made these qualitative claims, this is an area that would benefit greatly from future research. Studies on true learning functions for various tasks will enable the use of our computational tool to minimize MAE. This work would also benefit from research on algorithms to search for the best k in the high-dimensional space that characterizes a given aggregation problem. Ultimately, however, TAG is shown to be a robust, performant, and useful algorithm in many cases, even when using a heuristic for k. Through this work, we propose an effective algorithm that models aggregation in situations of dynamic worker abilities over time and highlight the potential for future work in this space.

References

[1] Chen, Sean Liu, Jun. (1997). Statistical Applications of the Poisson-Binomial and conditional Bernoulli distributions. Statistica Sinica. 7. 875-892.

[2] Donmez, Pinar Carbonell, Jaime Schneider, Jeff. (2010). A Probabilistic Framework to Learn from Multiple Annotators with Time-Varying Accuracy. 826-837. 10.1137/1.9781611972801.72.

[3] Drapeau, R., L. Chilton, J. Bragg, and D. Weld. “MicroTalk: Using Argumentation to Improve Crowdsourcing Accuracy”. Proceedings of the AAAI Conference on Human Computation and Crowdsourcing, vol. 4, no. 1, Sept. 2016, pp. 32-41, https://ojs.aaai.org/index.php/HCOMP/article/view/13270.

[4] Hata, K. Krishna, R. Fei-Fei, L. Bernstein, M. (2016). A Glimpse Far into the Future: Understanding Long-term Crowd Worker Accuracy. 10.1145/2998181.2998248.

[5] Jung, H., Y. Park, and M. Lease. “Predicting Next Label Quality: A Time-Series Model of Crowdwork”. Proceedings of the AAAI Conference on Human Computation and Crowdsourcing, vol. 2, no. 1, Sept. 2014, pp. 87-95, https://ojs.aaai.org/index.php/HCOMP/article/view/13165.

[6] Qiu, C., Squicciarini, A., Khare, D.R., Carminati, B., and Caverlee, J (2018). CrowdEval: A Cost-Efficient Strategy to Evaluate Crowdsourced Worker’s Reliability. Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS ’18). International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp. 1486–1494.