Worker Skill Estimation in Team-Based Tasks

Habibur Rahman[‡], Saravanan Thirumuruganathan[‡], Senjuti Basu Roy[†]
Sihem Amer-Yahia[⋄], Gautam Das[‡].
[‡]UT Arlington, [†]UW Tacoma, [⋄] CNRS, LIG
{habibur.rahman,saravanan.thirumuruganathan}@mavs.uta.edu, senjutib@uw.edu,
sihem.amer-yahia@imag.fr,gdas@uta.edu

ABSTRACT

Many emerging applications such as collaborative editing, multi-player games, or fan-subbing require to form a team of experts to accomplish a task together. Existing research has investigated how to assign workers to such team-based tasks to ensure the best outcome assuming the skills of individual workers to be known. In this work, we investigate how to estimate individual worker's skill based on the outcome of the team-based tasks they have undertaken. We consider two popular skill aggregation functions and estimate the skill of the workers, where skill is either a deterministic value or a probability distribution. We propose efficient solutions for worker skill estimation using continuous and discrete optimization techniques. We present comprehensive experiments and validate the scalability and effectiveness of our proposed solutions using multiple real-world datasets.

1. INTRODUCTION

Automated team formation is widely studied in computer-assisted cooperative systems [23, 1, 2, 25, 19, 18]. This body of work assumes that a team of experts is to be formed to undertake a task that requires expertise in one or more domains. The formed team is assumed to have the expertise or skills required to meet the expected quality of the task (as well as other constraints such as coordination cost). Naturally, the formulation of this team formation problem assumes that the skills of individual workers are known a priori. We seek to investigate an orthogonal question: *Given a set of completed tasks undertaken by a team of workers, estimate the skills of the individual workers.* We refer to this as the skill estimation problem for team-based tasks.

A number of applications rely on team-based work. Examples are researchers co-authoring a paper, experts reviewing scientific papers, athletes playing team-based games, as well as some emerging crowdsourcing applications, such as Galaxy Zoo[1] or Foldit[2].

[1]http://www.galaxyzoo.org/
[2]http://fold.it/portal/

Proceedings of the VLDB Endowment, Vol. 8, No. 11
Copyright 2015 VLDB Endowment 2150-8097/15/07.

Skill Estimation Problem: Estimating the skills of individual workers for team-based tasks is acknowledged to be an important open problem [24] in this space. We borrow the settings of the team formation problem [23, 1, 2, 25, 19, 18]. Inputs to our skill estimation problem are a set of teams (each team is a set of workers), that has completed one (or more) tasks. Each task requires a skill that is also known a-priori. Each completed task gets evaluated quality-wise and a numeric score is assigned to it. A worker may participate in different tasks with different teams. A worker skill is a *deterministic value*, or a *probability distribution (pdf)* that we wish to estimate as accurately as possible from the quality feedback assigned to the tasks in which her team participated. Modeling skill as a pdf can capture the fact that some workers have large variance in their skill levels when performing tasks, whereas others have smaller variances. For example, two players may have the same average points per game, but one has greater variance over the other.

Skill Aggregation Functions: To be able to effectively estimate the skills of workers involved in team-based efforts, it is critical to formulate how a team's skill is computed by aggregating the skills of individual workers in the team. Prior work [24] indicates that there exists several skill aggregation functions: (1) `Sum` where the skill of a team is the sum of skills of individual workers. As an example, the number of blocks that a basketball team makes in a game is the *sum* of the defense skills of the defenders. (2) `Max` where the skill of a team corresponds to its most skilled worker. The quality of a research paper may be dominated by the expertise of the most skilled author. (3) A `complex` function defined over workers skills, as well as other aspects, such as collaboration effectiveness, is another alternative. Such a complex function might not assume independence between different workers' skills. We explore Sum and Max in depth and discuss extending our algorithms to handle complex aggregation functions in Section 6.

Task Quality: In general, measuring task quality depends on the application. We assume that we are provided with a quality evaluation (as a numeric score) for each task. The applications we describe above can indeed be evaluated in many ways: for example, the number of citations of a paper reflects its (quality) impact, a team has an offense or a defense score in a particular basketball game.

Team Skills and Task Quality: Typically, workers are evaluated based on their skills (where skill is deterministic or probabilistic) while tasks are evaluated based on quality. It is apparent that the skills of the workers in a team contribute to the quality outcome of the task they undertake together. We assume that there is a known, *one-to-one correspondence* between worker skills and task quality. In the basketball example, defense skills of workers provide defense score for the game, while offense skills give rise to an offense score.

Challenges: Even when the relationship between skill and quality is injective, i.e. one-to-one, there are number of challenges in solving the skill estimation problem. The key challenge comes from the fact that the quality evaluation reflects the aggregated skill of the entire team, while we seek to estimate the skill of individual workers. Proportionally allocating the final quality of a task among its constituent workers to estimate the skill of every worker, considering different tasks that she has undertaken, is non-trivial.

Our Approach: In this paper, we primarily focus on learning individual worker skills under `Sum` and `Max` aggregation functions. Our methods could be trivially extended to `Min` aggregation. Moreover, if dependency between the workers can be expressed in a linear fashion, our deterministic solutions can be extended to handle those scenarios as well. We defer more detailed discussion on this to Section 6. At a high level, our approach is based on computing the "distance" between the estimated skills of the individuals and the known quality of the completed tasks that they have undertaken, assuming a given skill aggregation function. We refer to this distance as error and quantify it using the ℓ_2 function, a common distance measure. Thereby, we formalize skill estimation as an optimization problem with the goal of minimizing error.

We start by considering deterministic skills. The `Sum` variant, `Sum-Skill-D`, is formalized as a continuous optimization problem while the `Max` variant, `Max-Skill-D`, is posed as a discrete optimization problems. We propose quadratic programming-based solutions for `Sum` and max-algebra[7, 3] based solutions for `Max`. We employ a similar optimization framework for skills described as a probability distribution function (pdf). Both variants, `Sum-Skill-P` for `Sum` and `Max-Skill-P` for `Max`, are designed to estimate the skill pdf of each worker such that the aggregated ℓ_2 error between the joint pdf, i.e., the team's skill, and that of the individual pdfs is minimized. For `Sum-Skill-P` (resp., `Max-Skill-P`), the pdf that represents the team skill is a joint pdf computed by taking the *sum convolution* (resp., *max convolution*) [4] of individual skill pdfs of the participants. The key challenge here is to be able to *deconvolve* the joint pdfs to estimate the individual pdfs accurately.

Finally, we present a comprehensive evaluation of our solutions using two real-world datasets and demonstrate that they indeed estimate the true skills of individual workers effectively and compare with several appropriate baseline algorithms. Additionally, we demonstrate that our solutions are scalable using large-scale synthetic data. In summary, we make the following contributions:

- **Formalism:** We formalize the problem of skill estimation for team-based tasks for different skill aggregations (Section 3).
- **Solutions:** We propose a comprehensive optimization-based formulation considering both deterministic and probabilistic interpretations of skills. We propose principled solutions and present theoretical analyses (Sections 4 & 5).
- **Discussion & Experimental results:** We conduct comprehensive experiments on multiple real-world datasets and a synthetic one that show that our algorithms are accurate and efficient (Section 7). We discuss the extensions of the problems in Section 6.

	u_1	u_2	u_3	quality ($\overrightarrow{Q}$)
$\mathbf{t_1}$	1	1	0	15
$\mathbf{t_2}$	0	1	1	10

Table 1: Task Assignment Matrix and Quality Evaluation Vector

2. APPLICATIONS OF TEAM-BASED TASKS

We now present a generic running example which will be used throughout the paper and motivate the two skill aggregation functions studied in the paper.

Example 1. *Running Example: Imagine a specific instance of the skill estimation problem where the following input is provided: A Boolean matrix, that represents which worker worked on which tasks (i.e., worker to task assignment matrix) and a vector that represents the evaluated quality of the two tasks (see Table-1). Our objective is to learn the skills of workers u_1, u_2, u_3.*

Sum Skill Aggregation - Team-based sports: Consider a team-based activity such as Basketball where each player contributes to a game in multiple ways: attack players who together contribute to *scores* and defense players who together contribute to *blocks*. Naturally, the *scores* and *blocks* of a team are the sum of its individual worker's scores and blocks. Given available history of past games and their respective outcomes (scores and blocks), we intend to learn the skill of individual players in *scores* and in *blocks*.

Maximum Skill Aggregation - Research paper co-authorship: For a team of researchers co-authoring a paper, the qualitative outcome of the work is often driven by the most skilled (or experienced) researcher. Similarly, the quality evaluation of a paper could be modeled simply as the number of citations it gets within a given time period. As a concrete scenario, the number of citations that a database paper gets is an indicator of its technical quality. In this example, we intend to learn each co-author's skill (expertise) in the database area.

These two aggregation functions represent a wide range of team formation application scenarios [27, 13]. Further discussions on other skill aggregation is deferred to Section 6.

3. DATA MODEL AND FORMALISM

3.1 Data Model

Workers: We have a set $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$ of n available workers. For the NBA application, workers are players, whereas in the co-authorship application, workers are the authors.

Domains & Skills: We are given a set of skill domains $D = \{d_1, d_2, \ldots, d_m\}$. Skill domains are associated with both tasks and workers. We assume that each domain is independent and focus on estimating workers' skill per domain. For the NBA application, domains could be defense or attack and each player has a value for each domain (0 denotes no skill in a domain).

Our problem now simplifies to estimating a single skill per worker and invoking the estimation for each domain independently. We represent a worker skill as s^u. The skills of all n workers are presented by a vector $\overrightarrow{S}$. s^u is either a *deterministic value, or a probability distribution.* The latter scenario assumes that some workers have large variance in their skill levels when performing tasks, whereas others have smaller variances. For example, in NBA two players may have the same average points per game, but one has greater variance over the other. Thus, s^u is a random variable and is represented by a probability distribution function (pdf). To simplify exposition, we assume that the skill pdf of a worker u is discretized across w possible range of values (i.e.,buckets), where $\sum_w Pr(s^u = w) = 1$. Using Example 1, if the skill pdf of worker u_1 (in the range of $[0-15]$) is discretized using 3 equi-width buckets, the buckets may represent skill $[0-5], [5-10], [10-15]$.

Team Based Tasks: We assume a set $\mathcal{T}$ of l completed team-based tasks. Each task involves a team of workers. For the NBA application, each game is a task, whereas for co-authorship, each research paper is a task.

Teams or Groups: A team or group $\mathcal{G} \subseteq \mathcal{U}$ comprises of a set of workers from $\mathcal{U}$ who participate in a task together. A team $\mathcal{G}$ undertaking a task t is referred to as $\mathcal{G}^t$.

Task Assignment Matrix: For each completed task t, we know the workers in $\mathcal{G}^t$ who undertook t. This gives rise to the task assignment matrix $\mathcal{A}_{l\times n}$ (n workers and l tasks). Each cell $a_{ij} \in \{0,1\}$ contains a binary value, where a 1 indicates that the i^{th} task was undertaken by the j^{th} worker, and 0 otherwise.

Task Quality Evaluation Vector: Each completed task t is assigned a continuous quality score. For all l tasks, we obtain a vector of length l, $\overrightarrow{Q}$. For example, the quality of a team $\mathcal{G}$ in a game t could be measured as the total *scores* in that game. The number of citations could automatically reflect the quality of a published research paper.

3.2 Formalism

Next, we formalize different skill aggregation functions for team-based applications.

Additive (Sum) Skill Aggregation Model: In this model [24], the performance of a team for a task t is computed as the sum of skills of the workers who undertook task t together. Formally, the skill of a team $\mathcal{G}^t$ for task t, can be computed as:

$$q^t = \sum_{u\in\mathcal{G}^t} s^u \tag{1}$$

Team-based sports are popular examples of the additive skill aggregation model, where more workers add more value to the task. In the running example, the skill of team of workers u_1 and u_2 working on task t_1 is $(s^{u_1} + s^{u_2}) = 15$.

Maximum (Max) Skill Aggregation Model: In this model [24], the team skill is dominated by the skill of the most skilled worker in the team. Formally, we have:

$$q^t = \max_{u\in\mathcal{G}^t} s^u \tag{2}$$

This model fits closely with creative tasks[27]. For example, a research work may require forming a team, where the quality is primarily dominated by the highest skilled researcher. In the running example, the skill of team (u_1, u_2) working on task t_1 can be computed as $\max(s^{u_1}, s^{u_2}) = 15$.

3.2.1 Problem Definition

Worker Skill Estimation: For each worker u, s^u needs to be computed considering all the tasks undertaken by u. As a simple example, for the *additive aggregation model*, this gives rise to a system of linear equations, satisfying $\mathcal{A}\times\overrightarrow{S} = \overrightarrow{Q}$, where the objective is to estimate $\overrightarrow{S}$.

In many scenarios, there may not be any feasible solution to a given problem instance. Consider our running example again under `Max` skill and assume that worker u_1 also participated in task t_2. Now we can assign either skill value of 15 or 10 to her; either way, this does not produce a feasible solution (because $\mathcal{A}\times max(\overrightarrow{S})$ and $\overrightarrow{Q}$ are not same). Therefore, we must estimate skill accuracy by measuring some error.

We relax our formulation into an inequality - i.e $\mathcal{A}\times\overrightarrow{S} \preceq \overrightarrow{Q}$. Our objective is to estimate an optimal value for $\overrightarrow{S}$ that satisfies all the inequalities, and has a small *reconstruction error*. For every task t, the *reconstruction error* is the difference between the estimated skills of $\mathcal{G}^t$ and the given quality of t. The overall reconstruction error across all l tasks is denoted by $\mathcal{E}(\overrightarrow{Q}, \mathcal{A}\otimes\overrightarrow{S})$. Our optimization, therefore, is to

$$\text{Minimize } \overrightarrow{Q} - \mathcal{A}\otimes\overrightarrow{S} \tag{3}$$

The operator $\otimes$ is $\times$ for additive skill model and max for maximum skill aggregation model.

Reconstruction Error: Our problem is most aptly represented with *one sided error* [6] which assumes that the actual quality value of task t (i.e., q^t) is never smaller than that of the estimated skills of the team that undertook t, for a given skill aggregation model. While this conservative approach may underestimate the true skill of a worker, it in turn provides better assignment of workers to future tasks, where the assigned workers will necessarily surpass the minimum skill requirement of the tasks. On the other hand, *two sided error* may overestimate worker's skill, which may lead to poor task assignment, because the true skill of a worker is actually smaller than what is estimated. Formally, one sided error could be specified as $\overrightarrow{Q} - \mathcal{A}\times\overrightarrow{S}$ and we require this expression to be non-negative. Considering one sided error, operator $\preceq$ only represents $\leq$ between the left and the right hand side of the above equation.

Error Functions: Recall that we compute the reconstruction error $\mathcal{E}(\overrightarrow{Q}, \mathcal{A}\otimes\overrightarrow{S})$ between two vectors, $\overrightarrow{Q}$ and $\mathcal{A}\otimes\overrightarrow{S}$ by measuring their distance or norm. The distance between two vectors $\overrightarrow{V}$ and $\overrightarrow{V'}$ could naturally be computed using several norms. We focus on ℓ_2 and note that our solution framework requires simple adaptation for ℓ_1 and L_∞.

ℓ_2 norm: $||\overrightarrow{V} - \overrightarrow{V'}||_2 = \sqrt{\Sigma_k(v_k - v'_k)^2}$. As an example, for our running example, if $\mathcal{A}\otimes\overrightarrow{S}$ is a vector $(9,10)^T$ for two tasks and $\overrightarrow{Q}$ is $(15,10)^T$, the reconstruction error is $\mathcal{E}(\overrightarrow{Q}, \mathcal{A}\otimes\overrightarrow{S}) = \sqrt{(9-15)^2 + (10-10)^2} = 6$

Selecting Optimal $\overrightarrow{S}$: In an over-determined system [5], where there are more tasks than workers, there may not be any feasible solution. Our objective in this case is to identify a solution that has the smallest ℓ_2 reconstruction error. For an under-determined system [5], there are more workers than tasks. In this scenario, there may be many feasible solutions and the objective is to select one of them that minimizes some prior function. The most common approach is to use `MaxEnt` or principle of Maximum Entropy [15] for such scenarios. We explore the former in depth (which is realistic for our applications) and defer the latter to future work.

Optimization Problems: Formally, given a task assignment matrix $\mathcal{A}$ and task quality estimate $\overrightarrow{Q}$, where,

$\mathcal{A} \times \overrightarrow{S} \leq \overrightarrow{Q}$, estimate $\overrightarrow{S}$ (where s^u is the skill of worker u in this vector) that minimize:

Problem 1. `Sum-Skill-D` $\mathcal{E}(\overrightarrow{Q}, \mathcal{A} \times \overrightarrow{S})$, *where s^u is deterministic.*

Problem 2. `Sum-Skill-P` $\mathcal{E}(\overrightarrow{Q}, \mathcal{A} \times \overrightarrow{S})$, *where s^u is a discrete pdf.*

Problem 3. `Max-Skill-D` $\mathcal{E}(\overrightarrow{Q}, \mathcal{A} \times max(\overrightarrow{S}))$, *where s^u is deterministic.*

Problem 4. `Max-Skill-P` $\mathcal{E}(\overrightarrow{Q}, \mathcal{A} \times max(\overrightarrow{S}))$, *where s^u is a discrete pdf.*

4. SUM-SKILL

We consider the first skill aggregation function - `Sum` - where the team skill corresponds to the *sum* of its members.

4.1 Sum-Skill-D

For the deterministic case, the skill of each worker u (i.e. s^u) corresponds to an unknown variable. Given a task t, the skill of a team is the sum of its individual workers who undertook it - $\sum_{u \in t} s^u$. This expression is upper-bounded by the qualitative skill assigned to the task. Formally, each task t completed by team $\mathcal{G}^t$ corresponds to an inequality

$$\sum_{u \in \mathcal{G}^t} s^u \leq q^t \tag{4}$$

`Sum` naturally lends itself to formulating skill estimation as a system of linear inequalities as follows:

$$\mathcal{A} \times \overrightarrow{S} \leq \overrightarrow{Q} \tag{5}$$

Running Example: The example from Section 3.1 can be formalized as a set of constraints, such as:

$$s^{u_1} + s^{u_2} \leq 15; \qquad s^{u_2} + s^{u_3} \leq 10; \qquad lb \leq s^u \leq ub$$

where lb and ub are problem specific lower and upper bounds for the skill of workers. The above inequalities are trivially satisfiable by setting all entries of $\overrightarrow{S}$ to 0. In order to obtain realistic values, we need to design an optimization formulation based on how close the current assignment is to the qualitative assignment provided by the domain expert, i.e., by minimizing the reconstruction error.

ℓ_2 Reconstruction Error: We use ℓ_2 measure that computes the Euclidean distance between $\overrightarrow{Q}$ and $\mathcal{A} \times \overrightarrow{S}$. If the linear inequalities are indeed equalities, this would reduce to linear least squares [16]. Due to inequalities and additional constraints this becomes a constrained least square problem. Specifically, our formulation has a quadratic objective and linear constraints - which we model as a quadratic programming problem with linear constraints. This variant is a known instance of convex optimization [29].

The corresponding optimization problem is formalized as

$$\begin{aligned} \text{minimize} \quad & \sqrt{\sum_t {e_t}^2} \\ \text{subject to} \quad & q^t - (\overrightarrow{A_t} \times s^u) \geq e_t, \forall u \in \mathcal{G}^t \\ & lb \leq s^u \leq ub \end{aligned} \tag{6}$$

where lb and ub are problem specific lower and upper bounds for the skill of workers. Specifically, our problem corresponds to a *box-constrained* least squares as the solution vector must fall between known lower- and upper-bounds. The solution to this problem can be categorized into active-set or interior-point methods. The active-set based methods construct a feasible region, compute the corresponding active-set, and use the variables in the active constraints to form an alternate formulation of least square optimization with equality constraints [29]. The interior-point methods encode the convex set (of solutions) as a barrier function. Quasi-Newton methods are then used to optimize this function. Using our running example, the estimated skill vector of workers are $\langle 9.0014, 5.994, 4.0031 \rangle$ with ℓ_2 error of 0.

Complexity: Both the active-set and interior-point methods are very efficient and run in polynomial time [5]. The worst case complexity for computing constrained least squares (by using generalized singular value decomposition) is $O(n^2 l + n^3)$ [5]. In practice, iterative algorithms often return the solution within a small number of iterations [5].

4.2 Sum-Skill-P

We now describe the skill estimation problem considering skill of a worker u, i.e., s^u as a probability distribution function (or simply a pdf) that is unknown to us. It can be any arbitrary distribution, which is discretized over a set of w possible range of values (each range is a bucket) that the pdf can take. While such discretization may introduce error in the overall calculation, there are no efficient alternatives that can handle any arbitrary pdf.

If there is only a single task in the task assignment matrix $\mathcal{A}$, we intend to produce the skill pdfs of the workers such that the *joint pdf of quality of the assigned team is as close as possible to the obtained quality.* However, the task assignment matrix contains many tasks with (possibly) different quality and a worker has typically undertaken many tasks. Thus, we need to estimate the skill pdfs of the workers such that the ℓ_2 error across all the tasks is minimized. The quality of each completed task (q^t) by a team is known and performed by taking the `Sum` of individual worker's skill pdfs. As we describe below, this step is akin to taking the `Sum`*-convolution of the individual skill pdfs of the workers to compute the joint skill pdf of the team.* However, we do not know these individual skill pdfs - rather, only the quality of each team as a whole (i.e., the q^t's) are available at our disposal. The challenge is to be able to estimate the individual skill pdfs from these q^t's (in other words, deconvolve the q^ts to generate the individual skill pdfs) such that the error is minimized. Moreover, the one-sided error constraints must also be respected. More specifically, we need to perform the following three necessary transformations for that.

(1) **Computing skill pdf of a team:** When each worker's skill in a team $\mathcal{G}^t$ who undertakes task t is a pdf, the quality of the team $\mathcal{G}^t$ is also a pdf. We assume the independence of workers' skill, i.e., the skill of a worker does not improve/degrade due to the presence of certain fellow workers. For `Sum` skill, the joint pdf of the team's skill (or quality) (e.g., multiparty online games) could be computed using *Sum-Convolution* of the individual skill pdfs. The definition of *Sum-Convolution* of two pdfs is adapted from prior work [4] and is given below. A simple example of the joint skill distribution using `Sum`-aggregation (i.e., Sum Convolution) is presented in Figure 1(a) (further described in (2)), considering two skill pdfs after appropriate discretization. In general, Sum-Convolution of an arbitrary number of M pdfs can be computed by performing a sequence of $M - 1$ Sum-convolutions, first convolving the first and the second

pdfs, then convolving the resultant pdf with the third pdf, and so on.

Definition 1 (Sum-Convolution of Distributions). *Assume that $f(x)$, $g(x)$ are the pdfs of two independent random variables X and Y respectively. The pdf of the random variable $X+Y$ (the sum of the two random variables) is the convolution of the two pdfs: $*(\{f,g\})(x) = \int_0^x f(z)g(x-z)\,dz$.*

(2) **Representing q^t as a pdf:** The quality of task t, q^t, a deterministic value, should also be represented as a pdf. Consider Example 1 (Section 2) and notice that $q^{t_1} = 15$. If u_1 and u_2 have worked in task t_1, without any prior information, the skill of both workers s^{u_1} and s^{u_2} can range between $[0, 15]$. While the task quality can be between $[0-30]$, we know that it has a skill of 15. Therefore, the resultant pdf is between $[0, 30]$, yet only the skill value of 15 has a probability of 1, and all other skill values have probability of 0. We translate the deterministic quality of each completed task to a pdf and discretize involving w equi-width buckets: for t_1, these buckets are $[0-10], [10-20], [20-30]$, where only the second bucket is associated with a probability of 1 (as it contains 15).

(3) **One sided error:** Unlike the deterministic case, where one can easily specify one sided error constraints, such as, $s^{u_1} + s^{u_2} \leq 15$, there is no obvious easy way to specify such *hard constraints*, when each s^u is a pdf. Therefore, we ensure that the *probability that the joint pdf $s^{u_1} + s^{u_2}$ is larger than* 15 is smaller than a predefined threshold, λ. i.e., $Pr(s^{u_1} + s^{u_2} > 15) \leq \lambda$. By controlling the value of λ, we can tune these constraints in a flexible manner. Since the skill pdf of each worker is discretized over a set of w different ranges, we can still use ℓ_2 distance to compute the difference or error between the joint pdf (represented using Sum-Convolution) and the pdf that represents the obtained quality [9].The corresponding optimization problem is,

$$\text{minimize} \quad \sqrt{\sum_t (q^t - (\overrightarrow{A_t} \times s^u))^2} \tag{7}$$

The constraints are to be set up such that the pdfs of the workers satisfy the *probability axioms* as well as the one-sided error constraints, such as.

$$Pr(\sum_{u \in t} s^u > q^t) \leq \lambda \qquad \forall t$$
$$\sum_w Pr(s^u = w) = 1 \qquad \forall u.$$

If skill pdf a worker is represented involving w buckets, then, each skill pdf is associated with w unknown variables. Table 2 shows the various pdfs that are associated with Example 1, if each of the pdfs are discretized using 3 ($w = 3$) buckets. Without loss of generality, imagine variables (unknowns) p_i^u represent the probability of the i-th skill bucket of s^u (for skill of u_1, $p_1^{u1}, p_2^{u1}, p_3^{u1}$ are $[0-5]$, $[5-10]$, $[10-15]$, respectively). Also, let p_i^t (known) represent the probability of the i-th skill bucket of q^t (for q^{t1}, $[0-10]$, $[10-20]$, $[20-30]$ are represented using $p_1^{t1}, p_2^{t1}, p_3^{t1}$ respectively, where, $p_2^{t1} = 1$).

The challenge in solving the optimization problem is estimating these variables (in other words, deconvolve the q^ts to generate the individual skill pdfs) such that they minimize the ℓ_2 error. For our running example, just considering the second bucket of q^{t_1} ($[10-20]$, where the probability mass is 1), we need to set up the variables such that the probability that the sum of $s^{u_1} + s^{u_2}$ to be in $[10-20]$ is as high as possible. This can be done by computing the probability, when $s^{u_1} = [5-10] \&\ s^{u_2} = [5-10]$, or $s^{u_1} = [0-5] \&\ s^{u_2} = [10-15]$, or $s^{u_1} = [10-15] \&\ s^{u_2} = [0-5]$. Therefore, the corresponding formulation is to solve and minimize for

$$(p_2^{t1} - \{p_2^{u1} * p_2^{u2} + p_1^{u1} * p_3^{u2} + p_3^{u1} * p_1^{u2}\})^2$$

pdf	range	known/unknown
s^{u_1}	$[0-5], [5-10], [10-15]$	all unknown
s^{u_2}	$[0-5], [5-10], [10-15]$	all unknown
s^{u_3}	$[0-5], [5-10], [10-15]$	all unknown
q^{t_1}	$[0-10], [10-20], [20-30]$	all known
q^{t_2}	$[0-10], [10-20], [20-30]$	all known

Table 2: Discretized pdfs using 3-equi-width histograms for Example 1

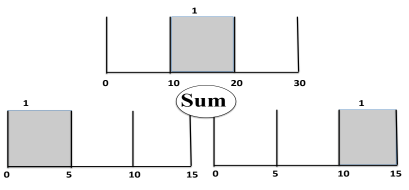
It is easy to notice that even for the toy example that involves only two workers per task, such a formulation gives rise to a quartic polynomial (degree of 4). For the general case, the degree of the resultant polynomial could be of the order n. Solving such functions optimally is thus prohibitively expensive. We resort to a hill climbing based efficient heuristic solution, as a viable alternative.

Heuristic Algorithm: We design a hill climbing based heuristic algorithm that uses random restarts. We start with a pdf for each s^u (uniform in our case in lack of any prior knowledge) and the overall objective function value (i.e., ℓ_2 error) is computed. In a single iteration, this algorithm selects one of the workers u at random and updates its pdf by a small value δ. Notice that, since the pdf of each worker is discretized using w buckets, this step corresponds to randomly choosing one of the w buckets and increasing (or decreasing) the probability of it by δ, while readjusting the other buckets uniformly to keep the probability mass to 1. As an example, for u_1, if $w = 3, \delta = 0.2$, and the first skill bucket (p_1^{u1}) of the initial uniform distribution ($p_1^{u1} = 0.33, p_2^{u1} = .33, p_2^{u1} = .34$) of s^{u_1} is being increased, then the adjusted pdf of s^{u_1} will be, $p_1^{u1} = 0.53, p_2^{u1} = .23, p_2^{u1} = .24$. With the modified pdf of u, it recomputes the objective function value and takes this change, if the error is further reduced. This process continues until no change can be found to improve the error. The solution is then said to be "locally optimal". With random restart, the algorithm performs hill-climbing iteratively, each time with a random initial condition and the best solution is kept at the end as the final solution. The various restarts increase the likelihood of finding "global optima". Our algorithm discovers the global optima on Example 1 and produces the following 3 distributions, $s^{u_1} = [0, 0, 1], s^{u_2} = [1, 0, 0], s^{u_3} = [0, 1, 0]$.

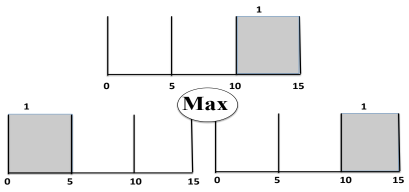
Complexity: The exact asymptotic form is hard to derive, as that depends on how fast it reaches the local optima in a given iteration. Our experimental results indicate that the solution converges within a few minutes most of the time even for a large scale dataset.

5. MAX-SKILL

We now describe our solution for the maximum skill aggregation function `Maximum` (or simply `Max`). Under `Max`, the skill of a team corresponds to that of its most skilled



(a)



(b)

Figure 1: An example joint pdf after, (a) `Sum` aggregation using Sum-Convolution (b) `Maximum` aggregation using Max-Convolutions. In fact, our objective is to learn the individual skill pdfs, given q^t.

member and the problem of estimation individual skills becomes a discrete optimization problem. As before, we first describe the deterministic solution, and then illustrate the probabilistic case.

5.1 Max-Skill-D

Before we describe our solution, we explain the correspondence between our problem and a mathematical algebraic theory *Max-Plus Algebra* [7, 3], which has been developed to solve a number of discrete optimization problems.

5.1.1 Overview of Max-Algebra

Traditionally, Max (or Max-Plus) Algebra provides techniques for solving non-linear problems that could be specified in the form of linear problems, when arithmetic *addition is replaced by a maximum operation*, and arithmetic multiplication is replaced by addition [7, 3]. Further, the inverse of a number is equivalent to its negation and ∞ is denoted by ϵ. Using mathematical notations, the key max-algebraic equations are given below:

$$a \oplus b = \max(a, b) \qquad a \otimes b = a + b \qquad a^{-1} = -a$$

$$a \oplus \epsilon = a \qquad a \otimes \epsilon = \epsilon$$

Almost all linear algebraic operations could be derived in the context of max algebra. Specifically, the matrix-vector multiplication required for solving a system of (in)equalities can be specified as:

$$A \otimes b = \sum_{k}^{\otimes} a_{i,k} \otimes b_k = \max_{k}(a_{i,k} + b_k) \tag{8}$$

Intuitively, the system of inequalities $A \otimes x \leq b$ can be interpreted as

$$(a_{11} \otimes x_1) \oplus (a_{12} \otimes x_2) \oplus \ldots \oplus (a_{1n} \otimes x_n) \leq b_1$$
$$\ldots$$
$$(a_{l1} \otimes x_1) \oplus (a_{l2} \otimes x_2) \oplus \ldots \oplus (a_{ln} \otimes x_n) \leq b_l$$

Using the standard linear algebraic notation, this is equivalent to solving the system of linear inequalities:

$$\max\{(a_{11} + x_1), (a_{12} + x_2), \ldots, (a_{1n} + x_n)\} \leq b_1$$
$$\ldots$$
$$\max\{(a_{l1} + x_1), (a_{l2} + x_2), \ldots, (a_{ln} + x_n)\} \leq b_l$$

If this system of inequalities has a solution, then we can see that it must satisfy the following set of inequalities:

$$x_1 \leq min\{(b_1 - a_{11}), (b_2 - a_{21}), \ldots, (b_l - a_{l1})\}$$
$$\ldots$$
$$x_n \leq min\{(b_l - a_{1n}), (b_2 - a_{2n}), \ldots, (b_l - a_{ln})\}$$

The candidate solution that we derive this way is called the *principal solution* [7, 3], using Equation 9.

$$\overline{x_i} = \left(\max A_{i,j} \otimes (b_i)^{-1}\right)^{-1} = \min\{b_i \otimes a_{ij}^{-1}\} \tag{9}$$

5.1.2 Proposed Solution

Considering `Max`, we however have slightly different formulations than in Max-Algebra. Given a task t which is undertaken by group $\mathcal{G}^t$, we intend to estimate the skills of the workers to minimize, $q^t - \max(\overrightarrow{A_t} \times s^u) \geq e_t, \forall u \in \mathcal{G}^t$. For our running example, this gives rise to the following set of constraints:

$$\max(s^{u_1}, s^{u_2}) \leq 15; \quad \max(s^{u_2} + s^{u_3}) \leq 10; \quad lb \leq s^u \leq ub$$

where lb and ub are problem specific lower and upper bounds for the skill of workers. Even though, the operator inside $\max(\overrightarrow{A_t} \times s^u)$ is a multiplication (instead of an addition in the traditional max-algebra), the techniques proposed in Max-Algebra leaves enough intuition behind to design a solution for our problem considering one sided error. Algorithm 1 presents the pseudo-code.

Algorithm 1 Algorithm for `Max-Skill-D`

```
1: Input: $\mathcal{A}$, $\overrightarrow{Q}$
2: Replace 1 and 0 in $\mathcal{A}$ to 0 and $\epsilon$ respectively
3: Construct system of linear inequalities $\mathcal{A} \otimes \overrightarrow{S} \leq \overrightarrow{Q}$
4: Compute principal solution vector $\overrightarrow{S}$ using Equation 9
5: return $\overrightarrow{S}$
```

In particular, consider our running example and notice that we can adapt the principal solution technique to form the following inequalities, based on the aforementioned constraints.

$$s^{u_1} \leq \min(15) = 15 \qquad s^{u_2} \leq \min(15, 10) = 10$$
$$s^{u_3} \leq \min(10) = 10$$

`Max` becomes computationally much harder when two sided error is considered. Exploration of two sided error for max skill aggregation is deferred to future work.

LEMMA 1. *The principal solution vector is a valid solution to the system of inequalities for the max skill aggregation problem* `Max`.

PROOF. (Sketch): Once we express the maximum skill aggregation using the system of inequalities, the proof directly follows from [7, 3]. □

LEMMA 2. *The principal solution vector minimizes the reconstruction error for ℓ_2 for one sided error.*

PROOF. (sketch): The principal solution vector outputs a feasible region for each variable (each variable corresponds to a worker's skill in the given domain). A detailed proof in [8] shows that how the proposed solution minimizes ℓ_∞ norm. Extending it to ℓ_2 is trivial. □

Time Complexity: The principal solution could be computed by a single pass over the matrix $\mathcal{A}$ and $\overrightarrow{Q}$. The running time is dominated by the dimension of matrix $\mathcal{A}$ which is $l \times n$. Therefore, the time complexity is $O(nl)$.

5.2 Max-Skill-P

Next, we describe the probabilistic skill estimation under `Max`. Akin to Section 4.2, we first translate the quality of each completed task, i.e., q^t as a pdf and intend to estimate the skill pdf of each worker u, represented as s^u, while satisfying the one-sided error constraints using ℓ_2. However, unlike `Sum`, the skill aggregation function is now *maximum* (i.e., `Max`), reflected by taking the *maximum* skill among the participating workers.

To compute the joint probability distribution of two independent random variables under `Max`, one has to compute the *Max-Convolution* [4] of two random variables, that we formally define below. The joint distribution of M random variables under `Max` could be computed by performing a sequence of $M-1$ pairwise *Max-Convolution.* Consider Figure 1(b), where the joint pdf of two random variables under `Max` skill aggregation is presented.

DEFINITION 2 (MAX-CONVOLUTION OF DISTRIBUTIONS). *Assume that $f(x)$, $g(x)$ are the pdfs of the two independent random variables X, Y respectively. The pdf of the random variable $Max(X,Y)$ (the maximum of the two random variables) is the max convolution of the two pdfs:*$_{max} * (\{f,g\})(x) = f(x)\int_0^x g(z)\,dz + g(x)\int_0^x f(z)\,dz$.

The main challenge is to estimate the individual skill pdfs of the workers as accurately as possible, such that, the distance between the obtained skills (based on Max-Convolution) and the assigned quality is as small as possible. The optimization problem exploits the same framework and is now restated as,

$$\text{minimize} \quad \sqrt{\sum_t (q^t - (\overrightarrow{A_t} \times max(s^u)))^2}$$

The constraints are to be set up such that the pdfs of the workers satisfy the *probability axioms*, as well as one sided constraints, akin to Section 4.2. Similar to the `Sum` counterpart, we discretize the skill pdf of each worker using w buckets and and assign a variable per worker per bucket that we intend to estimate.

Similar to the sum problem, the optimization problem gives rise to solving a polynomial of degree of n, if a task has n number of workers. Unfortunately, solving the problem is prohibitively expensive. Therefore, we design a hill climbing based efficient heuristic algorithm.

Heuristic Algorithm: Algorithm for `Max-Skill-P` runs is a greedy fashion and performs hill climbing with random restarts. It is very similar with the algorithm for `Sum-Skill-P` in flavor, except the fact that now it has to perform Max-Convolution to compute the joint pdf of skill of the workers. We omit the details for brevity. Our algorithm discovers the global optima on Example 1 and produces the following 3 distributions, $s^{u_1} = [0,0,1], s^{u_2} = [1,0,0], s^{u_3} = [0,1,0]$.

Complexity: The running time complexity is similar to that of `Sum-Skill-P`.

6. DISCUSSION

Min Skill Aggregation: It is easy to see that the techniques developed in Section 5 can easily be extended to handle `Min` skill, should the application fit that aggregation model. A well studied body of research, `Min-Plus` (or Tropical) algebra [7] has been developed to study the algebraic operations with the `Min` operator. The fundamental operations can be specified by the identities $a \oplus b = a + b$ and $a \otimes b = min\{a, b\}$. Most results from Max-Plus algebra are directly applicable just by switching the `Max` operator with `Min`. Specifically, we can rewrite Equation 9 specifying the principal solution as,

$$\overline{x_i} = \left(MinA_{i,j} \otimes (b_i)^{-1}\right)^{-1} = Max\{b_i \otimes a_{ij}^{-1}\} \qquad (10)$$

The only change required in Algorithm 1 is to replace Equation 9 with 10.

Similarly, the probabilistic skill learning algorithms could be adapted by deconvolving `Min`-functions [4].

Complex Skill Aggregations: In our paper, we have initiated the first ever formal treatment to estimate the skills of the workers for team based tasks. Our chosen functions share some attractive properties: (a) They are representative of the skill aggregation functions commonly used in a number of real-world team based tasks [27, 13]. (b) They are supported by a well developed body of research (such as linear, max-plus algebra, or min-algebra) that allows the development of efficient polynomial time algorithms.

Our proposed optimization framework could be used to represent any arbitrary skill aggregation function that takes the skills of a set of workers and outputs a scalar score for the team. Specifically, the optimization objective for a complex skill aggregation function f can be formulated as:

$$\begin{aligned} &\text{minimize} && \mathcal{E}(\overrightarrow{Q}, f(\mathcal{A}, \overrightarrow{S})) \\ &\text{subject to} && q^t - f(\overrightarrow{A_t}, s^u) \geq 0, \forall u \in \mathcal{G}^t \\ &&& lb \leq s^u \leq ub \end{aligned} \qquad (11)$$

Under certain constraints (such as convexity), such a formulation might even be solved efficiently. Investigation into their (in)tractability and design of efficient solutions remain open problems at this point.

Independence and Collaboration: The independence assumption is indeed true in several applications, such as online multi-party games and sentence translation by fans (fan-subbing). It also allows us designing efficient solutions.

Our deterministic approaches could be extended to incorporate collaborations, as long as, the aggregation functions are linearly expressible. Similarly, our probabilistic algorithms could also be adapted by representing worker-dependence with higher order histograms [26] to compute their joint distributions.

In team productivity literature [27, 13], it is known that some individuals act as "multipliers" or "enablers". Similarly, affinity between team members also plays a role. We describe two popular models next.
(1) Additive Factor Model [27, 13]: The skill of a worker is composed of two factors; a baseline skill that the worker exhibits in all tasks, and a constant factor that depends on the team and the task. Thus the same worker could have different levels of performance for each task. If the additive factor is computed through a known function, then it could be added into the model as a constant factor with a unit weight. The problem then becomes finding the baseline skills of the workers, and our algorithms are applicable. *(2) Pairwise Affinity Model* [27, 13]: Alternatively, the skill of a worker in a task could be computed as the sum of the individual worker and the pairwise affinity that she has with other members of the team. Our model could be extended to handle this scenario through "linearization", where, we add a new variable for each pairwise interaction. Under this, our proposed deterministic algorithms extend. *(3) Non-Linear Affinity Models:* Notice that both the models described above could be described by a linear function. It is possible to design non-linear affinity functions (such as based on a clique). These scenarios fall under the broader category of complex aggregation functions.

7. EXPERIMENTAL EVALUATION

Our development and test environment uses Python 2.7 on a Linux Ubuntu 14.04 machine, with Intel Core i5 2.3 GHz processor and a 6-GB RAM. We use an existing convex optimization package [3] for solving `Sum-Skill`. All numbers are presented as the average of three runs.

7.1 Dataset Descriptions

1) NBA: We collected 317,371 tuples of NBA scores from 1991-2004 regular season. We pre-process this dataset and generate the worker to task assignment matrix $\mathcal{A}$ by matching players with games. We consider two independent skill dimensions, i) Number of points, ii) Number of Assists where the team skill is the computed by the additive skill model. Our final dataset contains 21000 matches and 1200 players.

Ground truth in NBA dataset: The ground truth consists of the number of points and the number of assists of a player played in a particular game. If a player has played several games (which is really the case always), this gives rise to a distribution, as opposed to a single skill value per worker.

2) DBLP: We use a subset of DBLP[4] considering the papers that are published from year 2000. We primarily consider authors who publish in database conferences (SIGMOD, VLDB, CIKM, ICDE, EDBT), in the area of query processing. Each publication is a completed task that is undertaken by a set of authors; we consider the number of citations as its quality. This dataset consists of 20123 publications and 22700 unique authors.

Ground truth in DBLP dataset: Unlike the NBA dataset, there is no ground truth available per worker (i.e., no truth is known about an author's expertise). Therefore, we neither have a skill pdf nor a single skill value per worker. As we describe in Section 7.3, we take up a cross-validation type of approach for evaluation here.

3) Synthetic Dataset: We generate a synthetic dataset for evaluating the scalability of our proposed solutions for the deterministic and probabilistic algorithms. The total number of workers varies between 5000 and 20000 and the total number of tasks varies between 5000 and 250000. The quality vector is an uniform distribution $[0-1]$.

7.2 Implemented Algorithms

Since no prior work has studied the skill estimation problem for team based tasks, we ourselves design multiple baseline solutions for comparative evaluation.

7.2.1 Sum Skill

Deterministic Algorithms:

(1) `BL-Sum-Regression-D`: We treat the problem as a multivariate regression problem, with the presence or absence of each individual in a team as the independent variables. The quality of a completed task is the dependent variable. The objective is to learn the co-efficients (skills) of the workers, such that the ℓ_2 error between the estimated quality and actual quality is minimized. This is equivalent to solving a least squares regression problem that minimizes $\|\mathcal{A} \times \overrightarrow{S} - \overrightarrow{Q}\|^2$. Notice that this baseline does not necessarily satisfy the one sided error constraints.
(2) `BL-Sum-Uniform-Avg-D`: For each task with quality value q^t, we uniformly distribute the skill value among its constituent workers. Thus, any worker u, who undertake t receives a skill score $s^u = \frac{q^t}{|\mathcal{U}'|}$, where $|\mathcal{U}'|$ is the set of workers who undertake t. The final skill of u is calculated by taking the average of her received scores across all the tasks. This baseline does not optimize the error value or satisfy the one sided error constraints.
(3) `BL-Sum-Uniform-Min-D`: This baseline is similar to the the previous one, except that we choose more conservative assignment of skill for each worker such that the one sided error constraints are fully satisfied. First, we uniformly distribute the obtained quality of each task among its constituent workers. The final skill of u is calculated by taking the *minimum* of her received scores across all the tasks she has undertaken.
(4) `Sum-Skill-D` : Our proposed solution in Section 4.1 is compared with the baseline solutions, whenever appropriate.

Probabilistic Algorithms: To the best of our knowledge, there does not exist a regression based model which treats the independent variables in a probabilistic manner.
(1) `BL-Sum-Uniform-P`: This algorithm is similar to `BL-Sum-Uniform-Avg-D`, but produces a pdf per worker at the end. For a task with quality q^t, we uniformly distribute q^t among its constituent workers and generate a discrete pdf per worker, after considering all the tasks she participated.
(2) `Sum-Skill-P` : Our proposed solution in Section 4.2 is compared with the baseline solution, whenever appropriate.

7.2.2 Max-Skill

Deterministic Algorithms:

(1) `BL-Max-D`: In this baseline solution, we assume that the quality of a task reflects the skill of only one of the constituent workers. For each task, in step 1, we first choose a worker uniformly at random and assign her skill s^u as the quality of the task q^t. Each of the remaining workers u' who

[3]http://cvxopt.org/userguide/solvers.html
[4]http://dblp.uni-trier.de/xml/

undertook the same task receives a $s^{u'}$ smaller than s^u, using a uniform random distribution. In step 2, finally these obtained skills are averaged per worker to compute the final skill value.
(2) `Max-Skill-D` : Our proposed solution in Section 5.1 is compared with the baseline solution, whenever appropriate.

Probabilistic Algorithms:
(1) `BL-MAX-P`: This algorithm is designed in the same spirit as that of `BL-MAX-D`. The step-1 of this algorithm is similar to its deterministic counterpart; In step 2, instead of averaging the skill of each worker, we generate a pdf.
(2) `Max-Skill-P` : Our proposed solution in Section 5.2 is compared with the baseline solution, whenever appropriate.

7.3 Experimental Analyses Setup

7.3.1 *Measures & Parameter Setup: Quality*

Deterministic Scenario: We adopt a classical cross-validation based set up [12] to evaluate our deterministic algorithms. We divide the dataset in train and test and perform 3-fold cross validation. In particular, each record in a test set is a task that is undertaken by a set of workers. For each such worker, we estimate their respective skill considering only the train dataset (those who do not appear in train get a skill of 0). For each task in test set, the ground truth (i.e., the true quality) is the associated quality value. We compare the estimated quality of a task with that of the ground truth for that task.

We compare and contrast different algorithms by presenting *average absolute error* and *normalized relative error*. Relative error is computed as $\frac{\sqrt{\sum_{\forall t} e_t \times e_t}}{\sqrt{\sum_{\forall t} q^t \times q^t}}$. We present the percentage of tasks in our test set which overestimates the task quality (compared to the ground truth), thus violating the one-sided error constraints (Section 3.2.1).

Probabilistic Scenario: For the probabilistic variant of the NBA dataset, we compute the ℓ_2 error between the *estimated skill pdfs and that of the ground truth distributions*. We measure *relative error* over the test dataset (assuming the evaluated quality of each task in the test set as ground truth). We need to transform the estimated pdf of each worker to a single value by computing *expected skill*, i.e., $ExpectedSkill(u) = \sum_w (Pr(s^u = w) \times w)$ and measuring the ℓ_2 error between the actual quality and the expected quality of the tasks.

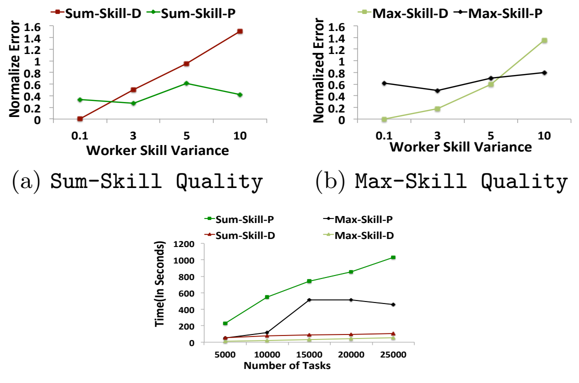
We discretize the pdfs using w equi-width histograms (buckets), where each bucket is a skill range. To compute the expected skill of a worker, we consider the upper limit of the skill range per bucket: for example, if the pdf of a worker is $Pr([0-5]) = .7, Pr([5-10]) = .3$, then the expected skill is $.7 \times 5 + .3 \times 10 = .65$.

Hill climbing algorithms have many parameters, w - # buckets to approximate the equi-width pdfs, δ - the amount by which we modify the pdfs in each step, and α - # failed iteration for the convergence of the hill climbing algorithm. Our default set up is $w = 10$, $\delta = 0.05$, $\lambda = 0.01$ (one sided error threshold), $\alpha = 1000$, # random restarts=5.

7.4 Summary of Results

Quality Experiments: Our first set of results (considering relative error) strongly corroborate our hypothesis on the underlying skill aggregation model for a given application - i.e., NBA dataset follows `Sum-Skill`, whereas, DBLP dataset follows `Max-Skill`. After this result, we present the rest of the experiments by considering the most appropriate dataset for it (i.e.,NBA for `Sum`, DBLP for `Max`). Our deterministic algorithms demonstrate that we consistently outperform all the baseline algorithms (including the regression based one) in *minimizing the error value*, as well as *consistently obeying the one sided error constraints*. Same observation holds for the probabilistic algorithms, where `Sum-Skill-P` and `Max-Skill-P` significantly outperform their respective baseline counterparts.

Scalability Experiments: For the deterministic scenario, our results indicate that our proposed solutions are scalable. Even when the task assignment matrix is very large, they only take a few minutes to complete. `Max-Skill-D` is more scalable than `Sum-Skill-D`. This is consistent with our theoretical analyses in Section 5.1. For the probabilistic scenario, the heuristic algorithms are more efficient than the optimal variants and often converge within few minutes and scale well. For lack of space, we only present a subset of results. The presented results are representative.



(a) `Sum-Skill Quality` (b) `Max-Skill Quality`

(c) `Scalability`

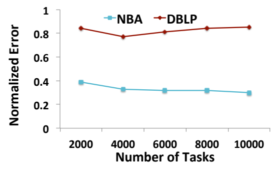
Figure 2: **Quality and Scalability trade-off: Deterministic/Probabilistic Models**

Deterministic/Probabilistic Skill Models: To highlight the tradeoffs between deterministic and probabilistic models, we conduct three experiments. We generate a synthetic dataset where each worker is associated with a skill pdf where we vary the variance for a fixed task assignment matrix. When the variance is low, then the workers are more consistent in their performance across different tasks. When the worker has high variance, her performance might have high deviation from the expected value. The results in Figure 2(a) and Figure 2(b) show that the deterministic variants are preferable for lower variance of worker's skill, while the probabilistic variants are preferable for higher variance. In our third experiment, we calculate the runtime of our deterministic and probabilistic algorithms with varying task size. Unsurprisingly, Figure 2(c) shows that deterministic algorithms are more scalable than probabilistic algorithms. This is quite expected as probabilistic algorithm takes longer time to converge due to greater number of unknowns than its deterministic counterparts.

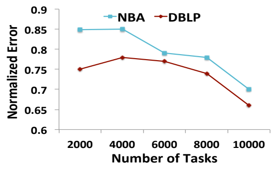
7.5 Qualitative Experiments

7.5.1 *Hypothesis Validation*

In this set of experiments, we vary the number of tasks and measure the average relative ℓ_2 error of both `Sum-Skill` and `Max-Skill` based skill estimation algorithms using both DBLP and NBA dataset. We present these results using `Sum-Skill-D` and `Max-Skill-D` algorithms. Figures 3(a) and 3(b) present the results which strongly corroborate our hypothesis: i.e., NBA dataset follows `Sum-Skill`, ensuring lower error compared to DBLP dataset. On the contrary, `Max-Skill` is better estimated using DBLP dataset (with smaller error compared to that of NBA). Clearly, in case of `Sum-Skill` error remains low in NBA dataset, as the training size gets higher where as DBLP dataset did not show any of this pattern. These results show that the formalized skill aggregation functions are indeed appropriate to capture real world applications.



(a) `Sum-Skill`



(b) `Max-Skill`

Figure 3: **Experiments for hypothesis validation:** Average relative ℓ_2 error for `Sum-Skill-D` and `Max-Skill-D` considering NBA and DBLP datasets, with varying # tasks. For `Sum-Skill-D` , NBA has significantly lower error, and for `Max-Skill-D` DBLP dataset outputs smaller relative error.

7.5.2 *Sum-Skill*

Here we vary the training dataset size to evaluate our proposed algorithms `Sum-Skill-D` and `Sum-Skill-P`.

Normalized Error - `Sum-Skill-D` : Figure 4(a) presents the results and clearly demonstrates that our proposed algorithm consistently outperforms all the baseline solutions, including the regression based baseline. The error decreases with increasing number of tasks, which corroborates that with increasing training set size, the skill estimation becomes more accurate.

One-sided Error Constraints - `Sum-Skill-D` : Figure 4(c) clearly demonstrates that our proposed algorithm consistently outperforms the three other baselines in one sided error constraints. `BL-Sum-Uniform-Avg-D` heavily overestimates workers' skill. `BL-Sum-Uniform-Min-D` is the best baseline, as it is designed to obey such constraints. Same observation holds for `Sum-Skill-P` and `BL-Sum-Uniform-P`.

Normalized Error - `Sum-Skill-P` : We present the normalized error of our proposed solution with that of the baseline algorithm, `BL-Sum-Uniform-P`. Figure 4(b) corroborates that `Sum-Skill-P` is significantly more accurate.

Normalized Error -`Sum-Skill-D` vs `Sum-Skill-P` : Figure 4(d) presents the comparison between them. Although `Sum-Skill-D` performs better, `Sum-Skill-P` provides more granular information on worker's skill.

7.5.3 *Max-Skill*

We use the DLBP dataset to perform further experiments on `Max-Skill`. We vary the number of tasks and measure the average absolute ℓ_2 error.

Figure 5(a) shows that `Max-Skill-D` outperforms `BL-Max-D` algorithm. With increasing training set, the algorithm learns better, hence the error of `Max-Skill-D` decreases. Figure 5(b) shows that `Max-Skill-P` outperforms the probabilistic baseline `BL-Max-P` consistently. However, error varies with the task size using `Max-Skill-P` , because the algorithm cannot always find the global optima. Figure 5(c) shows the comparison between `Max-Skill-D` and `Max-Skill-P`. We observe that `Max-Skill-D` performs better for larger training data. However, `Max-Skill-P` provides the distribution of worker skill which can be of importance for a particular application.

7.6 Scalability Experiments

7.6.1 *Deterministic Estimation: Sum and Max Skill*

For the deterministic scenario, we vary number of workers, tasks, workers/task, and domains. Our run-time experiments have the following default settings: #workers=5000, #tasks=10000, #workers/task=10, #domains= 1.

Varying Number of Tasks: We vary the number of tasks in this experiment. Figure 6(a) shows that both `Sum-Skill-D` and `Max-Skill-D` scale well with increasing number of tasks. Quite unsurprisingly, the latter outperforms the former algorithm scalability-wise. This result is expected and is consistent with our theoretical analyses of the algorithms.

Varying Number of Workers: Our observation here is akin to the previous experiment. Both algorithms scale well. Figure 6(b) shows the result.

Varying Number of (Workers/Task): The objective of this experiment is to observe the influence of the number of workers per task in the running time analyses. From figure 6(c), it is apparent that while `Max-Skill-D` scales very well due to its linear time complexity. However, `Sum-Skill-D` also performs reasonably with larger data size.

Varying Number of Domains: In this experiment, we vary number of domains and measure the scalability of the proposed algorithms. Figure 6(d) presents the results and demonstrates that our proposed solutions are scalable.

7.6.2 *Probabilistic Estimation: Sum and Max Skill*

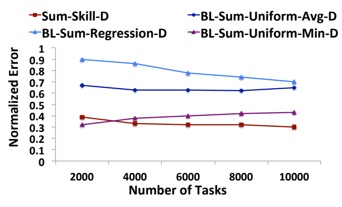
For the probabilistic skill, for brevity, we only present a subset of results. As stated before, we vary the parameters that influence the efficiency of the hill climbing algorithms.

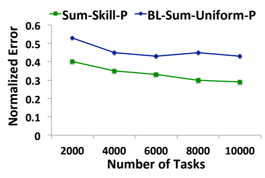
Varying w - `Max`: As we increase the number of skill buckets, the number of unknown to solve increases, hence it takes longer to converge. Figure 7(a) demonstrates that `Max-Skill-P` scales well with varying w for five different size datasets. `Sum-Skill-P` takes about 60% more time than `Max-Skill-P` as the joint pdf has larger ranges (e.g., two pdfs in the range of $[0-15]$ gives a joint pdf between $[0-30]$ for sum, whereas, it is still $[0-15]$ for max) for sum, thereby requiring more computations. Although efficiency decreases with increasing w, but we get the distribution of worker skill with more granularity.

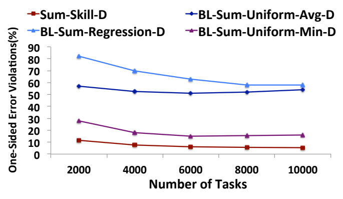
Varying δ - `Sum`: With higher step size (δ) in the hill climbing, the algorithm converges faster but with lower accuracy. `Sum-Skill-P` results are presented in Figure 7(b). `Max-Skill-P` has similar trend, but takes about 60% less time. We omit the chart for brevity.

Varying # failed iterations α - `Sum`: This parameter α dictates after how many failed iterations a random restart takes place inside the hill climbing algorithms. Figure 7(c) shows that our solution scales well with increasing α. As usual, `Max-Skill-P` takes about 60% less time always.

Parameter Tuning: This is evident from Figure 7(a), 7(b) and 7(c) that with the increase of w and α and decrease of δ, latency will be higher. However, high value of w and α


(a) Deterministic


(b) Probabilistic


(c) % unsuccessful estimation

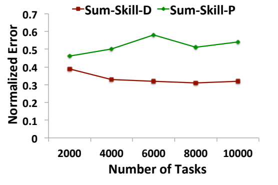
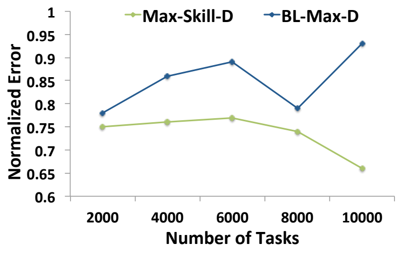
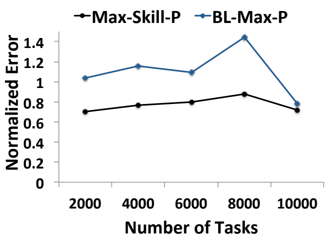

(d) Deterministic vs Probabilistic

Figure 4: **Experiments to validate `Sum-Skill` aggregation**: These results clearly demonstrate the our solutions consistently outperform the baseline for both the measures we present in the Y-axis. The underlying dataset that is used is NBA.


(a) `Max-Skill-D`


(b) `Max-Skill-P`

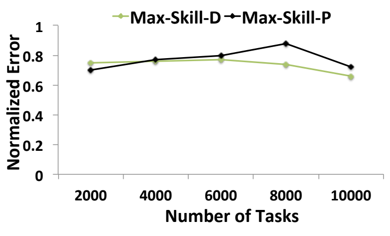

(c) Deterministic vs Probabilistic

Figure 5: **Experiments to validate `Max-Skill` aggregation** : In these experiments, we compute the average error by varying the number of tasks. Clearly, our proposed solutions `Max-Skill-D` outperforms the baseline algorithm `BL-Max-D` and `Max-Skill-P` outperforms `BL-Max-P`. The underlying dataset is DBLP.

and low value of δ ensure better results in terms of quality. Additionally, with higher # random restart, the solutions likely get better qualitatively, although the running time also increases (1.1 minute per restart on an average). With smaller λ (one sided error threshold), we get solutions that better satisfy the one-sided error constraint, but that improvement comes with an increasing computation time. Clearly, we need to consider trade-offs while choosing these parameter values. Empirically, with $w = 10$, $\delta = 0.05$, $\lambda = 0.01$, $\alpha = 1000$, # random restarts=5, we get the best trade-off between quality and scalability.

8. RELATED WORK

While no prior work has solved skill estimation problem for team based tasks, we present existing work that are tangentially related.

Team Formation: A tangential problem is the team formation problem [23, 30]. Formation of team considering a social network is first studied in [23, 1, 2]. The objective of these body of work is to form a team of experts to solve a particular task, which assumes that the skill of the experts are known and given as inputs. On the contrary, we intend to estimate the skill of the workers by investigating the quality of the completed tasks they have undertaken. No prior work, however, studies any formalism or solution to estimate worker's skill for team based tasks.

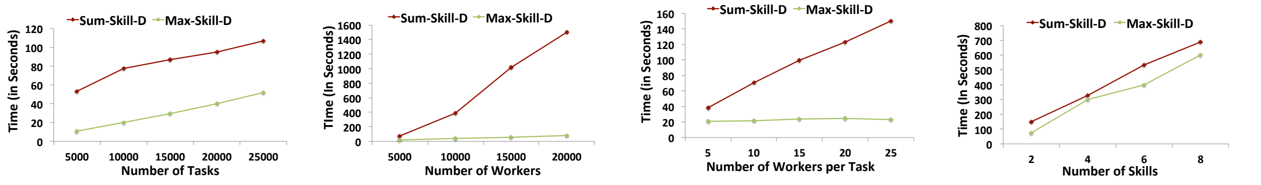
Skill Estimation in Micro-task based Crowdsourcing: Crowdsourcing has gained significant traction in the research community for solving problems, such as, image tagging, annotating labels, or looking up addresses of people [21]. These applications are primarily designed on micro-tasks, where the task is completed by an individual at its entirety (e.g., a worker tags an image all by herself). While skill estimation or evaluating the quality of the workers in crowdsourcing has gained recent research attention [14, 22, 17], the focus is entirely on micro-task based applications. We however consider team based tasks [20, 28].

Disaggregation Methods: Disaggregation methods are studied to dis-aggregate weather data to find hourly rainfall, temperature, or wind speed from daily maxima or minima [10, 11]. These methods do not lend any extension to solve our problem either, as their dis-aggregation happens "locally", i.e., per day, as opposed to our problem, where a worker can undertake many tasks and the skill estimation must consider all of them to minimize the error.

Regression Based Models: The `Sum`-skill problem bears resemblance with the *least square regression [12]* that we consider as a baseline, without having to satisfy the one-sided error constraints. Quantile regression [12] models the relationship between *the independent variables and the conditional quantile of the dependent variable.* Maximum quantile regression will learn the relationship between the maximum value of the dependent variable given the independent variables. Unlike that, our `Max` aggregation problem intends to learn the dependent variable which is the maximum of the independent variables. These are fundamentally different.

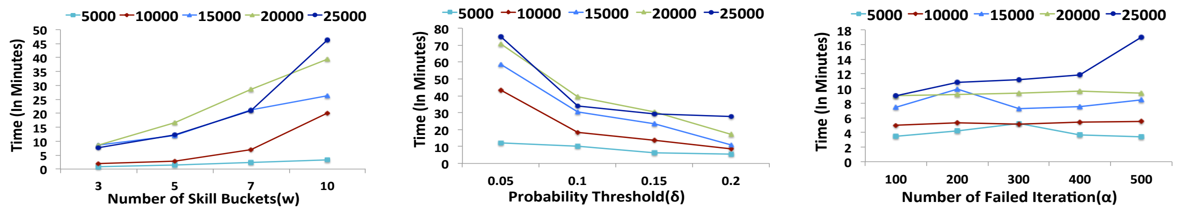
9. CONCLUSION

We initiate the study of estimating skill of the individual workers in team based tasks for various applications. We formalize it as an *optimization problem considering multiple skill aggregation functions, where skill of a worker is either deterministic or a probability distribution.* We propose principled solutions to all the studied variants of the problem and provide in-depth analyses. We run a comprehensive set of experiments considering two real world datasets that demonstrate the effectiveness of our proposed skill estimation algo-



(a) Varying # tasks (b) Varying # workers (c) Varying workers-to-tasks ratio (d) Varying # skill domains

Figure 6: **Experiments to validate the scalability of the deterministic skill estimation algorithms:** The following default settings is considered: # workers=5000, # tasks=10000, # workers/task=10, # domains= 1. These results clearly demonstrate that our proposed solutions are scalable.



(a) `Max-Skill-P` : Varying w (b) `Sum-Skill-P` : Varying δ (c) `Sum-Skill-P` : Varying α

Figure 7: **Experiments to validate the scalability of the probabilistic skill estimation algorithms**: default settings: # domains=1, $n = 1000, w = 3, \delta = 0.2$, # failed iterations=200, # random restarts=5; despite having to solve a polynomial of degree 20, our solutions scale well and terminate within a few minutes. We only present a subset of results for brevity.

rithms. We also conduct large scale synthetic experiments to validate the scalability of our proposed solutions.

Acknowledgment

The work of Habibur Rahman, Saravanan Thirumuruganathan and Gautam Das was partially supported by National Science Foundation under grants 0915834, 1018865, Army Research Office under grant W911NF-15-1-0020 and a grant from Microsoft Research.

References

[1] A. Anagnostopoulos et al. Power in unity: forming teams in large-scale community systems. In *CIKM*, 2010.

[2] A. Anagnostopoulos et al. Online team formation in social networks. In *WWW*, 2012.

[3] M. H. Andersen. Max-plus algebra: Properties and applications. 2011.

[4] B. Arai et al. Anytime measures for top-k algorithms on exact and fuzzy data sets. *The VLDB Journal*, 2009.

[5] A. Björck. *Numerical methods for least squares problems*. Siam, 1996.

[6] H. Buhrman et al. One-sided versus two-sided error in probabilistic computation. In *STACS 99*, 1999.

[7] P. Butkovic. *Max-linear systems: theory and algorithms*. Springer, 2010.

[8] P. Butkovič. One-sided max-linear systems and max-algebraic subspaces. In *Max-linear Systems: Theory and Algorithms*, pages 53–70. Springer, 2010.

[9] S.-H. Cha. Comprehensive survey on distance/similarity measures between probability density functions. *City*, 2007.

[10] B. Debele et al. Accuracy evaluation of weather data generation and disaggregation methods at finer timescales. *Advances in Water Resources*, 2007.

[11] T. A. Garrett. Aggregated versus disaggregated data in regression analysis: implications for inference. *Economics Letters*, 2003.

[12] J. Han et al. *Data Mining: Concepts and Techniques*. 2000.

[13] S. A. Haslam. *Psychology in organizations*. Sage, 2004.

[14] P. G. Ipeirotis et al. Quality management on amazon mechanical turk. In *KDD workshop*, 2010.

[15] E. T. Jaynes. On the rationale of maximum-entropy methods. *Proceedings of the IEEE*, 70(9):939–952, 1982.

[16] R. I. Jennrich. Asymptotic properties of non-linear least squares estimators. *The Annals of Math. Statistics*, 1969.

[17] M. Joglekar et al. Evaluating the crowd with confidence. In *SIGKDD*, 2013.

[18] M. Kargar et al. Finding affordable and collaborative teams from a network of experts.

[19] M. Kargar et al. Efficient bi-objective team formation in social networks. In *ECML PKDD*. 2012.

[20] A. Kittur. Crowdsourcing, collaboration and creativity. *ACM Crossroads*, 2010.

[21] A. Kittur et al. Crowdsourcing for usability: Using micro-task markets for rapid, remote, and low-cost user measurements. *CHI*, 2008.

[22] M. Kokkodis et al. Have you done anything like that?: Predicting performance using inter-category reputation. In *WSDM*, 2013.

[23] T. Lappas et al. Finding a team of experts in social networks. In *SIGKDD*, 2009.

[24] S. Leonardi. `http://research.microsoft.com/en-us/events/bda2013/team-formation.pdf`, 2013.

[25] A. Majumder et al. Capacitated team formation problem on social networks. In *SIGKDD*, 2012.

[26] V. Poosala et al. Selectivity estimation without the attribute value independence assumption. In *VLDB*, 1997.

[27] R. Reagans and E. W. Zuckerman. Networks, diversity, and productivity: The social capital of corporate r&d teams. *Organization science*, 12(4):502–517, 2001.

[28] S. B. Roy et al. Task assignment optimization in knowledge-intensive crowdsourcing. *VLDB Journal*, pages 1–25, 2015.

[29] P. B. Stark et al. Bounded-variable least-squares: an algorithm and applications. *Computational Statistics*, 1995.

[30] A. Zzkarian and A. Kusiak. Forming teams: an analytical approach. *IIE transactions*, 31(1):85–97, 1999.