Combining Human and Machine Computing Elements for Analysis via Crowdsourcing

Julian Jarrett, Iman Saleh, M. Brian Blake
Department of Computer Science
University of Miami
Coral Gables, Florida, USA
{j.jarrett, iman, m.brian.blake}@miami.edu

Rohan Malcolm, Sean Thorpe
Computational Sciences Research Group
School of Computing and IT
University of Technology, Jamaica
Kingston, Jamaica
rmalcolm92@gmail.com, sthorpe@utech.edu.jm

Tyrone Grandison
Proficiency Labs
Ashland, Oregon, USA
tgrandison@proficiencylabs.com

Abstract— **Crowd computing leverages human input in order to execute tasks that are computationally expensive, due to complexity and/or scale. Combined with automation, crowd computing can help solve problems efficiently and effectively. In this work, we introduce an elasticity framework that adaptively optimizes the use of human and automated software resources in order to maximize overall performance. This framework includes a quantitative model that supports elasticity when performing complex tasks. Our model defines a task complexity index and an elasticity index that is used to aid in decision support for assigning tasks to respective computing elements. Experiments demonstrate that the framework can effectively optimize the use of human and machine computing elements simultaneously. Also, as a consequence, overall performance is significantly enhanced.**

Keywords: crowdsouring; experimentation; elastic systems

I. INTRODUCTION

Crowd computing leverages the input of a *crowd* of online users [1, 2] to collaboratively solve complex problems. Humans and machines are seen as programmable computational units capable of executing tasks; both require seamless interactions even though their requirements may differ [3]. In this paper, we aim at finding the optimal interplay between machine computing elements (MCEs) and human computing elements (HCEs). We propose an elastic computing framework, a system that leverages both MCEs and HCEs, through a uniformed interface, in order to optimally solve a complex task [3]. Our work investigates three main research questions:

- *R1 – When human and machine elements are capable of performing the same task, is there a general model that can define and evaluate their respective performance outcomes simultaneously?*
- *R2 - Can experimentation in a specific domain, such as face recognition, uncover the most appropriate, shared evaluative attributes that have cross-domain applicability?*
- *R3 - Can the specific performance variations in real-life experimentation enhance our overall understanding and ultimately lead to a more generalized elastic model?*

To address these questions, we devise an elastic model and supporting architecture that governs the provisioning of MCE's, HCE's or both (hereinafter referred to as Elastic Computing Elements (ECEs)) given a specific task. The algorithms will use the elasticity model to ascertain the complexity of the task and the current operating environment and then proactively enforce constraints on its successful completion. The main idea is to extract the elasticity attributes of a certain task and use these attributes to orchestrate the use of HCEs and MCEs. Our work differs from other related projects in the fact that we focus on situations where the task for humans and for machine elements are *exactly the same*. Related projects focus on a variety of tasks for humans and machine elements where humans and machines might address distinct tasks.

To evaluate the effectiveness of our proposed framework, we consider a face recognition task. Face recognition techniques have been extensively used in security and law enforcement applications such as post-event analysis, shop lifting, suspect tracking and forensic crime scene investigation [4]. Despite the rapid progression in the area and the development of face recognition techniques, successful face recognition still faces several challenges; these include the capability to discern the same person when faced with variation in illumination, pose, image quality and background clutter in another image of the same person. Other factors such as the availability of well-defined recognition or matching criteria and nature or type of user input also affect the face recognition process [4]. As such, for R2 and R3, the face recognition domain is effective to understand factors that affect human and software elements differently.

II. RELATED WORK

Crowdsourcing approaches have been studied in a variety of related projects. Crowdsourcing has been used to harness large bodies of human resources to generate data for use in several systems and to complete tasks that are costly or time consuming with traditional methods [5]. There also have been several ambitious attempts to integrate human computing elements (HCE) and machine computing elements (MCE) under a uniformed service model.

COLLABORATECOM 2014, October 22-25, Miami, United States
Copyright © 2014 ICST
DOI 10.4108/icst.collaboratecom.2014.257298

Active Crowd Translation (ACT) [6], a system that leverages crowdsourcing via Amazon Mechanical Turk (MTurk) was created to aid in machine translation for language pairs. The system is classified as an end-to-end human in the loop translation system framework that combines online, human non-expert and expert translators with an automatic machine translation system. As multiple persons translate a sentence, they are compared with each other and other external sources to improve the translation quality. The best translations are then re-introduced to the system to serve as parallel data for driving future translation suggestions.

CrowdDB [7] was developed to leverage human input to respond to database queries that cannot be otherwise handled by the database management system. These queries typically require subjective comparisons that only a human can do. In other cases, human input is used to compensate for missing or incomplete data. Human query operators are used to solicit, integrate and clean crowd sourced data. The research collects metrics such as performance and cost and shows how these metrics are affected by worker affinity, training, fatigue, motivation and location.

Yan et al. [5] took advantage of the emergence of smartphones and implemented an iOS mobile application; *mCrowd*, to distribute image related tasks. The application uses the Amazon MTurk and ChaCha crowdsourcing platforms to perform tasks such as querying images and capturing geo-tagged images amongst other image-related tasks.

Heer & Bostock [8] explored the possibility of using crowdsourcing to evaluate a large design space of visualizations. They used Amazon MTurk to distribute human intelligence tasks to evaluate how visual variables affect the impact effectiveness of data visualizations. Results demonstrated that crowdsourcing results in lower cost to conduct the experiment. Savings are attributed to lower compensation rates for participants, lower expense for recruitment and solicitation, automated administration via Amazon MTurk platform among others. The quality of the results was high and is attributed to qualifications enforced for the task. Crowdsourcing also proved to give access to a wider population as opposed to convenient sampling; this catalyzed an experiment to be complete in a day versus weeks in a traditional experiment subject to recruitment and scheduling. It was found that crowd sourced tasks also allows for easier modification and extension of the study. On the other hand, some tasks took longer to complete. Data showed that tasks completed by participants in a traditional and controlled lab setting took an average of 5 seconds versus 42 seconds for the same crowd sourced task.

In their work, Dustdar and Truong [3] assert that elasticity in hybrid systems should have proactive approach for provisioning both MCE and HCE components. They propose that computing elements should be provisioned based on performance metrics including but not limited to financial costs, time, quality of results and compliance. This applies well to systems they describe where quality evaluations are done by human experts in the loop due to the complexity of a task. HCE's serve the role of reviewing intermediate results rendered by their MCE counterparts.

Several fundamental issues were identified in virtualizing humans and machine computing elements in an elastic process for large-scale complex applications [3]. They experimented with pattern recognition in satellite imagery. Their study showed that software does not detect specific patterns in many cases and thus human analysis is used. Human analysis was done by leveraging the input of crowds of novices, volunteers, professionals and experts in the area. Their study also addressed some of the challenges arising from managing the interactions of both machine-based computing elements (MCEs) and human-based computing elements (HCEs) given that their requirements and implementations are different. With the crowds, every HCE can work autonomously or in groups and constraints may or may not be imposed on quality of work. Some processes may also require HCE experts in the loop to verify and validate quality of results produced by MCEs before authorizing the completion of tasks. Systems applying these configurations for elasticity, need to consider integrating metrics for performance including time, costs, compliance, precision and quality of results and how these hybrid systems of both human and machine allow for the proactive scaling.

To consider virtualization and the provisioning of MCEs and HCEs in an elastic manner across clouds, the Vienna Elastic Computing Model, *VieCom*, was proposed in [3, 9]. VieCom supports the automated scaling in and out of MCE and HCE components based on dynamic contexts that impacts quality and costs. Service-oriented computing concepts and techniques are used to provision MCEs and HCEs under different clouds, elastic services may be invoked based on well-defined service interfaces [3]. VieCom seeks to address concern of traditional crowdscouring where there's a delay between posting a task and matching it with a person with the right expertise. Instead, VieCom proactively uses human computing capabilities to perform tasks in a time effective fashion. It considers HCEs as programmable units allowing them to play passive roles in task assignment by simply posting their own capabilities and possibilities and wait for incoming tasks. Tasks may also be given to a group that may be seen as one computational unit [9].

Tai, Leitner and Dustdar [10] defined units to measure and manage system resources in a system using an elasticity model. These units include usage and attributes of dynamic system resources, which extends to MCE's and HCE's. They propose units to measure impact of different system configurations to determine performance for higher qualities of service. Cost units were also presented considering the impact and the different system configurations. Finally, dynamicity units were defined; these acquire and release resources in a timely and a Just-In-Time fashion.

Dynamicity also considers impact and cost units to configure the runtime in constantly changing environment in a fully or semi-automated manner which may include human intervention.

Kamar, Hacker and Horvitz [11] investigated how Bayesian predictive models operated in architecture for crowdsourcing that combines the efforts of both machine vision and humans to solve consensus tasks. Consensus tasks are focused on uncovering correct answers to some phenomena or "hidden state of the world", not known to the owner of the task but possibly a larger population capable of making predictions. These predictions are collected through multiple assessments from the crowd [11]. For their consensus task, the authors attempted to classify celestial bodies using a system dubbed "Galaxy Zoo". Probabilistic models were used to predict human behavior and augment human and machine effort. The models were also used as a tool for crowd worker recruitment to optimize crowd utility based on inferences drawn on predictions from models [11].

Authors in [12] propose a metric model called an Elasticity Profile (EP) that defines metrics for MCE's and HCE's to adjust the delivery of service of a SaaS cloud platform. The adjustment of service delivery is based on quality and cost metrics for provisioning HCEs and/or MCEs in a dynamically changing runtime environment to improve performance, reliability and throughput based on consumer input. The EP has three (3) dimensions namely resources, quality and costs and have different criteria for both MCE's and HCE's. Metrics defined in the EP are used to further design and determine activities and behavior of the elasticity runtime. The EP is also used to define system trade-offs between resources, cost and quality of the system.

Kulkarni et al. [13] combined peer and machine grading to open-ended assessments while simultaneously preserving the quality of peer assessment and alleviating the burden of grading the assessments. They propose an algorithm for grading dubbed the "identify-verify pattern" that regulates the number of peers that evaluate an answer based on algorithm confidence in predictions and peer assessment. The first phase of the pattern predicts the grade of a student within some level of confidence; later used to quantify the number of peer raters needed. Using a rubric, a set of peers evaluates the answers and establishes key features followed by another set of peers corroborating the correct application of feature labels.

Our work proposes a generic service-oriented framework seeking to leverage machine and human computing elements and optimize their performance in completing assigned tasks across different problem domains. Elasticity properties of tasks are identified and used to determine task assignment to HCE and MCE components to maximize output of the final solution. Other works such as mCrowd, Active Crowd Translation and Crowd DB [5, 6, 7] have focused on a pure human computing model via crowdsourcing. Hybrid computing model with humans and machines were incorporated in VieCom [5, 9]; however machine and humans complete distinct tasks or subtasks of a larger task. We focus our model on optimizing performance for the same task as in [13] to be performed by both humans and machine components in a sequenced workflow capable of provisioning the more appropriate type of component for a specific task based on previous performance history.

III. Elascticity framework and architecture

To address our first research question, we propose a service-oriented elasticity framework that leverages machine and human resources to efficiently and effectively perform a task. The framework is depicted in Figure 1.

Our proposed elasticity framework (Figure 1) consists of three major components; (1) an *Elasticity Manager* that handles the selection process between HCE and MCE, (2) a *Resource Manager* that deals with the actual completion of the task by either MCE or HCE, and (3) a *Solution Formulation* component that determines the optimal output based on MCE and HCE feedback. It also determines if there enough varying of the data to warrant the application learning from the test case.

As shown in the figure, the different components are comprised of the following services and their interactions:

- *Elasticity Service*: This service orchestrates the use of MCEs and HCEs. It uses a metric model, an *Elasticity Index* (EI) to decide on the best combination of HCEs and MCEs that optimizes performance. The EI is domain-specific and captures the probability of successful task completion based on task modeling, as will be detailed later.
- *MCE Service:* This service manages the automated execution of a task.
- *HCE Service*: This service manages the execution of a task by a crowd of people. Different models of collaboration can be applied by customizing the HCE Service. For example, tasks can be done either by volunteers, experts, or in return of compensation. The service can also acquire resources by employing online crowdsourcing systems such as Amazon MTurk.
- *Consolidation Service*: This service amalgamates the output of the HCE and MCE services. The service can decide, for example, on ignoring MCE or HCE output if certain quality threshold is not met. The service sends feedback to enhance future decisions.
- *Learning Service*: This service builds intelligence into the HCEs and MCEs orchestration process by building a knowledge base, denoted as *Training Examples* in the figure. This knowledge base is enriched by feedback from the consolidation service and is used to enhance future system performance based on historical data.

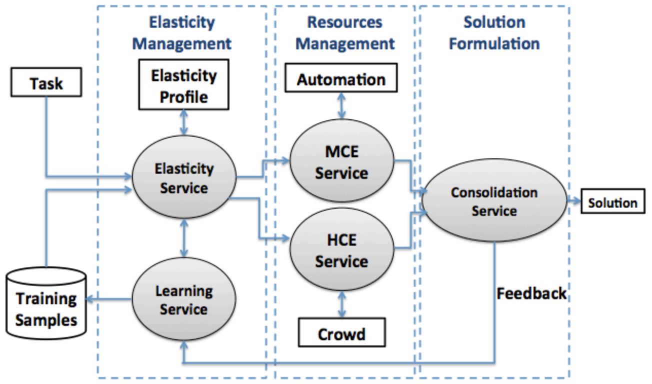


Figure 1. The Elasticity Service-Oriented Framework.

A. *Modelingd Elasticity*

For our elasticity service (component of elasticity manager) to determine when it is most appropriate to assign a task to machine versus human computing elements, we propose two weighted metric models, the Task Complexity Index (TCI) and the Elasticity Index (EI).

B. *Task Complexity Index*

The Task Complexity Index (TCI) is defined as a quantitative model that may be used to describe difficulty in completing a task. The TCI is comprised of weighted mean of metrics that are used to describe a task on a user-defined nominal scale. Formally, we define TCI as:

$$TCI = \frac{W_1 \times M_1 + \ldots + W_x \times M_x}{x} \tag{1}$$

where

- $x \in Z^+$ and $x \geq 1$.
- $W_i \in Z^+$ and $W_i \geq 0$, $i \in 0, 1, \ldots, x$.
- $M_i \in [1, N]$, where
  - ⋆ $N \in Z^+$ and $N \geq 1$.
  - ⋆ $i \in 0, 1, \ldots, x$.

where $W_1 \ldots W_x$ are weights attributed to corresponding metrics $M_1 \ldots M_x$ with x *(subscript)* denoting the maximum index of the weight metric pairs from a given set. $M \in \{1, 2, \ldots N\}$ is a value on a nominal scale that captures a qualitative aspect of the task where N is the maximum value of the same nominal scale. All values are constrained to Z^+, the set of positive integers. Domain experts define the metrics M_i and their weights W_i can be manually set by human or adjusted over time through the learning service of the system.

C. *Elasticity Index*

The EI is defined as a weighted sum model that considers the attributes A_i namely TCI, costs, time and other runtime attributes defined by a domain expert that may affect the quality of results when completing tasks. EI is defined as:

$$EI = W_1 \times A_1 + \cdots + W_i \times A_i \tag{2}$$

where

- $x \in Z^+$ and $x \geq 1$.
- $A_i \in R^+$ and $A_i \geq 0$, $i \in 0, 1, \ldots, x$.

where $W_1 \ldots W_x$ are weights attributed to corresponding attributes values $A_1 \ldots A_x$ with x *(subscript)* denoting the maximum index of the weight metric pairs from a given set. Z^+ denotes the set of positive integers and R^+ the set of positive real numbers.

For example, in case of face recognition, the TCI's attributes capture the pictures' characteristics that affect face recognition. These characteristics include the pictures' size, color scheme, quality…etc. The EI (which considers TCI and other runtime attributes like cost, time, etc) is then used to predict the precision of face recognition by both HCEs and MCEs. The purpose of our preliminary experimentation in this paper is to determine the nature of weights and how they vary in a real-life implementation.

Listing 1 outlines the typical workflow of the Elasticity and Consolidation services. The Elasticity Service is given a queue of tasks and calculates the optimal HCE and MCE subtasks for each task. The decompose module is used to determine if the task should be assigned exclusively to MCE's, HCE's or a combination of both based on the task's EI. The service assigns tasks to the HCE and MCE services to produce partial solutions that are amalgamated in the optimize module of the Consolidation Service to create an optimal solution.

```
Elasticity Service
Start
    Foreach Task T in the queue
        EI = Calculate_EI(T)
        //Based on EI, decompose T into substasks
        //to be executed by HCEs and MCEs
        {$T_{HCE}$,$T_{MCE}$}= Decompose (T, EI)
        Output {$T_{HCE}$,$T_{MCE}$}
    EndFor
Stop

Consolidation Service
Start
    Foreach SubTask T{$T_{HCE}$, $T_{MCE}$} in the queue
        //Based on EI, calculate the optimal result
        {$S_{HCE}$, $S_{MCE}$}= Optimize ($T_{HCE}$, $T_{MCE}$, EI)
        //Get Task solution from HCE and MCE Services
        $S_{ECE}$ = getSolution()
        Output $S_{ECE}$
    EndFor
```

```
Stop
```

Listing 1. Pseudo-Code of Core Services in the Proposed Elasticity Framework.

D. Task Abstract Data Type

The two metric models, TCI and EI are embedded in a task defined by the Task abstract data type (ADT) as shown in Listing 2. The Task ADT consists of four floating-point properties, Precision_Result, TCI, EI and EI_Tolerance. The Precision_Result is the current Task performance index of a particular computing element provisioned by the elasticity framework. TCI describes the complexity of the task and EI describes the holistic evaluation of resources required to complete task. EI_Tolerance describes a percentage value that is set to increase the system's EI_Threshold value; this is in order to allow the provisioning of more computing elements considering EI components when uncertainty is high. The properties of the Task ADT will be evaluated in algorithms we propose in the elasticity process to determine appropriate provisioning computing resources, human or machine.

```
Define Task As Structure
        Precision_Result As Double
        TCI As Double
        EI As Double
        EI_Tolerance As Double
End Definition
```

Listing 2. Task Abstract Data Type.

E. Maximum Performance Index Algorithm

Using the TCI, we have designed the Maximum Performance Index Algorithm (MPIA) outlined in Listing 3. The MPIA only considers the Task's TCI, and a MCE_TCI_Threshold (the correlated TCI for previous performance history of available MCE's of the elasticity framework), to provision computing components that would produce optimal solution completing a task. The MPIA's sole objective is to render maximum precision results for task completion and does not consider costs, time and other factors that may affect the elasticity process.

If Task TCI is greater than or equal to MCE_TCI_Threshold, HCE's are assigned the task; else if the Task TCI is less than the MCE_TCI_Threshold, MCE's are assigned the task. However if the MCE results consist of high levels of uncertainty, HCE's will be assigned the task for further evaluation to lower uncertainty and improve precision in results to produce results via ECE.

```
Algorithm_Maximum_Performance_Index
Start
  Set System MCE_TCI_Threshold
  Set Acceptable_Uncertainty
  Input Task
  Task.TCI = Task.Calculate_TCI()
  If Task.TCI>= MCE_TCI_Threshold
        Task.Precision_Result = processTaskHCE(Task)
  Else
     Task.Precision_Result = processTaskMCE(Task)
     If Task.Precision_Result<Acceptable_Uncertainty
        Task.Precision_Results = processTaskHCE(Task)
     End If
  End If
Stop
```

Listing 3. Maximum Performance Index Algorithm.

F. Weighted Metric Performance Index Algorithm

Using the EI model, we have designed the Weighted Metric Performance Index Algorithm (WMPIA) as shown in Listing 4. The WMPIA is more holistic and pragmatic; it considers a Task's EI and the system's EI_Threshold to provision available computing components to complete the task. The system EI_Threshold ensures that tasks are completed within constraints of the resources of the elasticity framework. Given our earlier definition of EI, the WMPIA considers multiple factors that may affect the elasticity process, including financial costs, available time for task completion, TCI, expertise and other factors that may affect the process depending on the particular task environment.

If Task EI is less than or equal to System_EI_Threshold, HCE's are assigned the task; else if the Task EI is greater than the System_EI_Threshold, MCE's are assigned the task. However if the MCE results consist of high levels of uncertainty, HCE's will be assigned the task for further evaluation to lower uncertainty and improve precision in results via ECE if and only if the Task EI is lower than the System_EI_Threshold after being increased by a percentage defined by the Task EI Tolerance.

```
Algorithm_Weighted_Metric_Performance_Index
Start
    Set System_EI_Threshold
    Set Acceptable_Uncertainty
    Set Task.EI_Tolerance
    Input Task
    Task.EI = Task.Calculate_EI()
    If Task.EI<= System_EI_Threshold
       Task.Precision_Result= processTaskHCE(Task)
    Else
       Task.Precision_Result = processTaskMCE(Task)
       If Task.Precision_Result<_
          Acceptable_Uncertainty AND_
          (Task.EI<  (System_EI_Threshold_
         * (1+Task.EI_Tolerance)))
             Task.Precision_Result = _
             processTaskHCE(Task)
       End If
    End If
Stop
```

Listing 4. Weighted Metric Performance Index Algorithm.

IV. Case Study: Mobile Face Recognition

In our work [14], we applied our elasticity service-oriented framework to the problem of face recognition. The physical layout for this case study encapsulating our proposed elasticity framework is illustrated in Figure 2. Reference images of popular individuals were stored in an image dataset. Testing images of the same individuals were passed through the elasticity system for identification. Face recognition tasks were initially sent to the MCE component for processing as shown in Figures 3, 4 and 5, and then forwarded to HCE components via mobile crowdsourcing application (Figure 6) to execute the same task. Tasks assigned to HCE's were forwarded with their preliminary results provided from MCE recognition process. We found that HCE components had an average certainty of 69.13% with maximum certainty of 100% a minimum of 16.67% and a range of 83.33%.

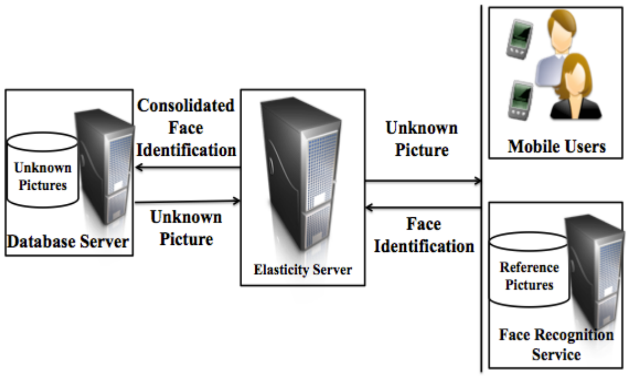


Figure 2. Face Recognition System Architecture Based on the Proposed Elastic Framework.

```
Procedure WorkFlowECE
Start
        Foreach ReferencePicture to Add to System
                Input ReferencePicture in ReferenceSet
        EndFor

        Foreach TestingPicture to Add to System
                Input TestingPicture in TestingSet
        EndFor

        Foreach TestingPicture in TestingSet
                ListSuggestions=Call_
                MCEMatch(accepts_
                TestingPicture, ReferenceSet)
        EndFor

        Send ListSuggestions to HCE via CrowdSourcing

        //WorkFlow executed on Mobile Device to
        //consolidate MCE with HCE to give ECE
        Foreach MCESuggestion in ListSuggestions
                Submit HCE Feedback for _
                MCESuggestion to Learning Service
        EndFor
        //WorkFlow on Mobile Device Ends

        //Begin MCE vs ECE analysis in Learning
        //service
        EI-ECE = Analyze HCE Feedback for
        MCESuggestions   //EI=ECE is EI-MCE //+ EI-
        HCE
        EI-MCE = Analyze ListSuggestions for Positive
        Identification
        Results = Compare EI-ECE vs EI-MCE
        Show Results
Stop


Function MCEMatch Returns ListSuggestions Accepts_
 TestingPicture, ReferenceSet
Start
        Foreach ReferencePicture in ReferenceSet

                ResultSimilarityMatch=Compare _
                TestingPicture to ReferencePicture using_
                MCE Face Recognition with bit _
                Threshold value 50

                If ResultSimilarityMatch > 60%
                        Add to ListSuggestions
                End If
        EndFor
        return ListSuggestions
Stop
```

Listing 5. Showing Algorithm Psuedo-BASIC Workflow of Mobile Face Recognition System.

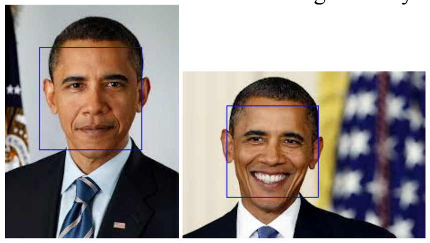


Figure 3. MCE component detecting faces in testing (left) and reference (right) pictures.

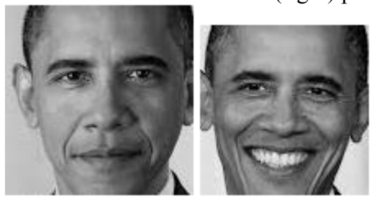
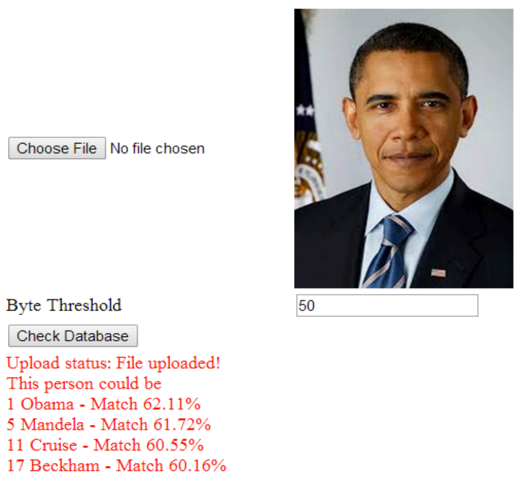


Figure 4. MCE component cropping faces and gray-scaling pictures to minimize impact of colors and focus on facial features.



Upload status: File uploaded!
This person could be
1 Obama - Match 62.11%
5 Mandela - Match 61.72%
11 Cruise - Match 60.55%
17 Beckham - Match 60.16%

Figure 5. MCE Component giving predictions for testing image using a bit matrix value comparison with byte threshold of value 50.

V. Evaluation

A. Experimentation

We conducted an experiment to test our research questions (R1 and R2) using face recognition as a domain specific task. To address R1, we conducted the experiment to ascertain performance index to build an elasticity profile for ECEs. For R2, we analyzed the variability in performance of both MCEs and HCEs based on task complexity. Our analysis helps define the attributes that are used to build the EP for face recognition. These attributes capture the different metrics that affect the ability of human and machine in identifying faces in a set of random pictures.

B. Data Set

The pictures of 23 popular individuals were selected to construct a reference set consisting of actors, singers, athletes and politicians. Another 23 pictures with random variations in 6 identifiable metrics (face angle, eyes, mouth image angle, face magnification, image quality) of the same individuals in the reference set were used as the testing set. The pictures were collected online by using Google Image search.

C. Workflow

In our experiment, we study the face recognition precision achieved with MCE, i.e. using the face recognition software. We then apply our elasticity approach that combines both software and human and we collect the corresponding face recognition precision using ECE.

The testing set was processed through the face recognition service for pre-verification. For each picture, the software component suggests a set of potential matches constrained to a matching threshold of greater than or equal to 60%. The custom Android application depicted in Figure 6 is used to collect the crowd's feedback on the same testing set. The matches generated by the face recognition service are sent via web service to the application. The application prompts the crowd user to indicate their celebrity category of expertise, and then s/he is presented with a series of pictures from the testing set and suggestions from the face recognition service as illustrated in the figure. The crowd used in our experiment is comprised of 30 randomly volunteering respondents 18 years and older, across the United States, Canada, France, the Middle East and Jamaica. The crowd was recruited via emails and social media connections.

We calculated the number of positive matches (P) of all suggestions (S) to determine the certainty level of the system. We then calculated the TCI for each test case, by qualitatively identifying and evaluating 6 metrics that potentially affect the MCE's performance in the face recognition and identification process. The metrics were evaluated on a nominal scale of 1 to 5, where 1 is least difficulty and 5 being maximum difficulty and itemized below:

- **(M_1)** Face Angle – Face in picture is 0° to an angle of 90°
- **(M_2)** Eyes – Eyes in picture are fully open to Shut
- **(M_3)** Mouth – Mouth is closed to fully open
- **(M_4)** Image Angle – Image is taken at an angle of 0° to 90°
- **(M_5)** Face Magnification – Face is close to far away
- **(M_6)** Image Quality – Quality of Image (lighting, pixels, etc.) High to Poor

For this experiment, all TCI weights $W_1 \ldots W_X$ were set to 1 for equal consideration in the task complexity. After obtaining the TCI for each test case, XY-Plots of TCI values were graphed as independent variables against the performance indices of MCE, HCE and ECE components using the statistical analysis tool R.

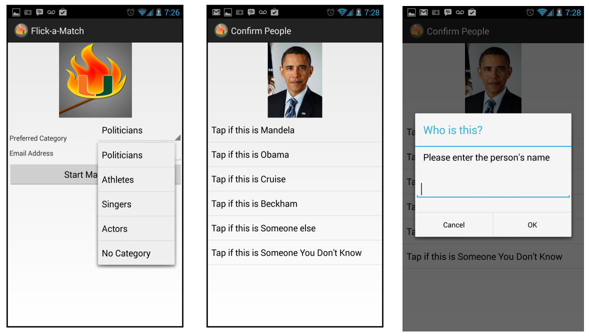


Figure 6. MCE components sent to Android Mobile crowdsourcing application for HCE's to provide feedback based on MCE predictions and their own knowledge.

VI. Results and Analysis

The face recognition precisions are depicted in Figure 9. As shown in the figure, out of the 23 test cases pictures, the recognition precision of 22 is increased by applying our ECE approach. The combined effort of MCE and HCE increased the probability of positively identifying an individual in the pictures in the testing set by a minimum of 16.67% and a mean of 55%. For test cases (1, 2, 6, 8, 16, 18, 20, 21 & 22), the pictures where MCE made suggestions consisting of a postive identification, ECE effort increased the precision by an average of 67.6%. In test case 8, MCE positively identifed the individual in the picture providing one (1) positive match. ECE responses reduced this probability by 53.3% as not all respondents positively identified the individual in the picture despite that the MCE component provided the correct suggestion; this is assumed to be related to the human respondents' prior knowledge of the individal and exposure to affairs that would enable them to identify the individual. We found that seven (7) of thirty (30) respondents said that their expertise was in identifying politicians, however only three (3) of the seven (7) positively identified the individual in test case 8; hence the majority of respondents had no prior knowledge of the individual.

We also observed a minimum increase of 16.67% and an average increase of 51.3% in positive identification of individuals when MCE and HCE efforts were combined; this includes situations were the MCE component failed to provide suggestions. As seen in test case 17, MCE failed to identify or make suggestions for the test portrait of athlete Usain Bolt however ECE was able to identify the athlete. We believe that facial expressions and difference in the angles of the face in the pictures impacted the performance of the MCE to positively identify the athelete (see Figures 7a and 7b). Poor performance of MCE may be attributed to face recognition approaches and techniques employed by the face recognition service. Humans on the other hand positively identified the athelete with an accuracy of 70% irrespective of facial expressions or other variances of the athelete's portrait.

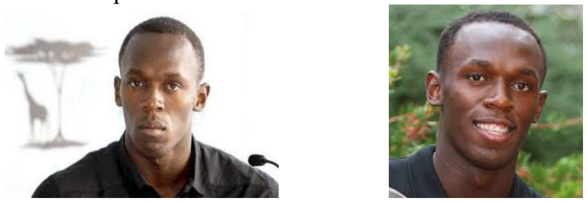

(a) (b)

Figure 7. Reference portrait of Usain Bolt Figure 7(a) used by MCE as a reference to identify the athlete in Test Case 17. Figure 7(b) is the unknown portrait to be identified by MCE and ECE processes.

We find the median and the mode of MCE effort at 0 and a mean positive match of 14.13%, as 14 of the 23 cases did not produce any suggestions positively identifying the individual. When using ECE, the median has significantly increased to 73.33% with a mode of 73.33% and a mean positive match of 69.13% (Figure 8).

Results clearly show that employing our elasticity approach, the probabilities of positive identification increase significantly.

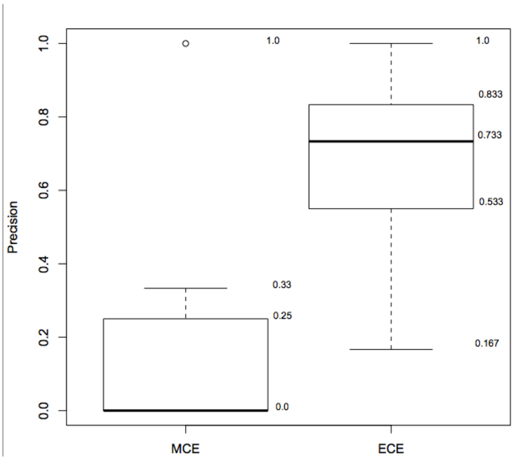


Figure 8. MCE vs. ECE box-plots showing dispersion of probabilities to positively identifying an individual in 23 test cases.

Computing elements' (MCE, HCE and ECE) performance indexes for successful face recognition were measured and compared. As Illustrated in Figure 9, almost all test cases with exception of test case 8, the MCE had the highest levels of uncertainty of the three types of computing elements. Test case 8 had 100% accuracy from MCE with a reduced accuracy of ~53% accuracy from HCE resulting in lower combined ECE performance; further analysis shows respondents in particular countries didn't know the public figure in the test case. It can also be seen in 16 of the 23 test cases, MCE completely failed and as such recorded no performance index. When HCE's were assigned the task, HCE's increased chances of successful face recognition by an average of 55%. When both MCE and HCE effort are combined to give ECE performance, ECE increases probability of successful face recognition by an average of 69%.

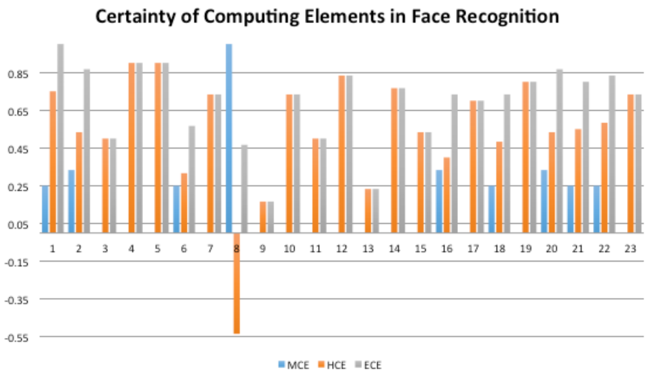


Figure 9. Performance Index bearing certainty for Computing Elements in elasticity framework for face recognition.

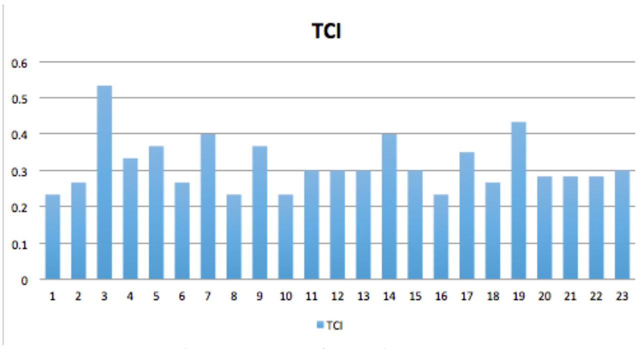


Figure 10. TCI for each test case.

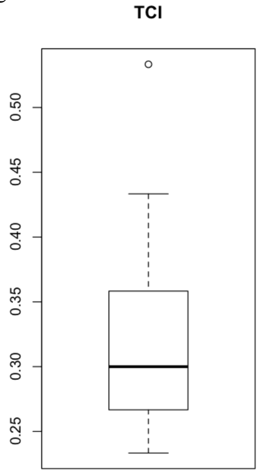


Figure 11. TCI Box Plot for each test case dataset.

The TCI for each test case was calculated and compared as shown in Figures 10 and 11. The testing data set had an average TCI of 0.32, a range of 0.3, a minimum TCI of 0.233 and a maximum TCI of 0.533. Using XY-Plots, we determined correlations between the independent variable TCI and the corresponding performance index for the computing element for each test case.

Figures 12, 13 14, and 15 are XY-Plots showing task TCI against certainties MCE, HCE, ECE, MCE/HCE/ECE respectively. Figure 12 shows an exponential decrease in the performance index of MCE components as the TCI for the task increased. It also portrays the MCE's inability to function once TCI surpasses a value of ~0.3. The point bearing a Y value of 1 indicating perfect performance is seen as an outlier as this result was not replicated in any other test case. Figure 13 shows a graceful increase of the performance index of HCE components as TCI increased; performance remained relatively consistent between ranges of 0.4 to 0.6. An outlier bearing a Y value of -0.5 resulted in test case 8 (Figure 13). For this test case, MCE performance was perfect with a certainty of 1, however uncertainty was increased when combined with HCE performance. Figure 14 also shows relatively consistent behavior of ECE as TCI increased with performance indexes ranging from 0.6 to 0.8.

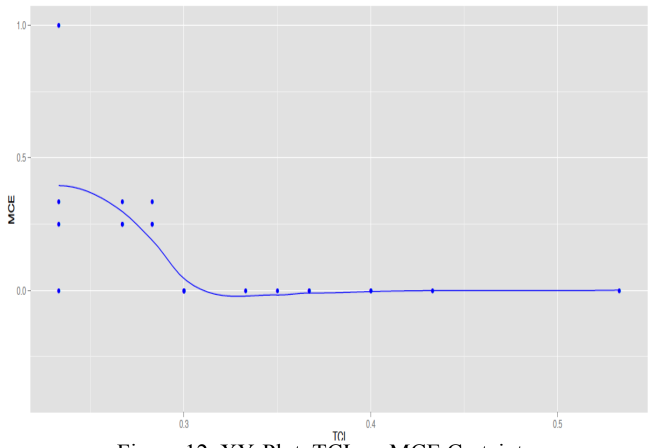


Figure 12. XY-Plot, TCI vs. MCE Certainty.

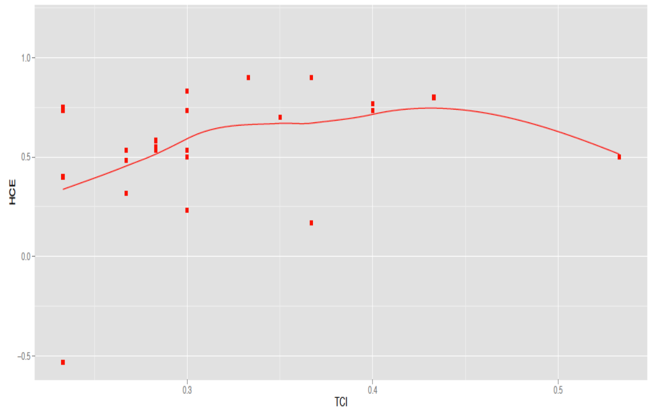


Figure 13. XY-Plot, TCI vs. HCE Certainty.

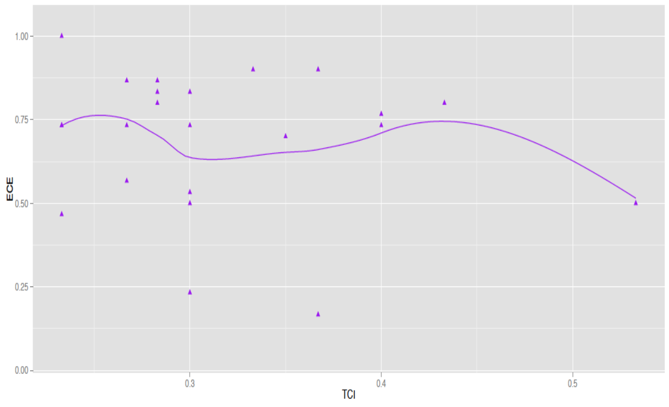


Figure 14. XY-Plot, TCI vs. ECE Certainty.

Superimposing the three graphs above (Figures 12, 13 and 14) into Figure 15, we find HCE and ECE performance converging; this accounts for high failure (16 of 23 test cases) of MCE component where MCE had no direct contributions to ECE results. Consequently the inability of the MCE component to perform after TCI is ~0.3 or higher records the lowest performance of the three computing elements in Figure 15.

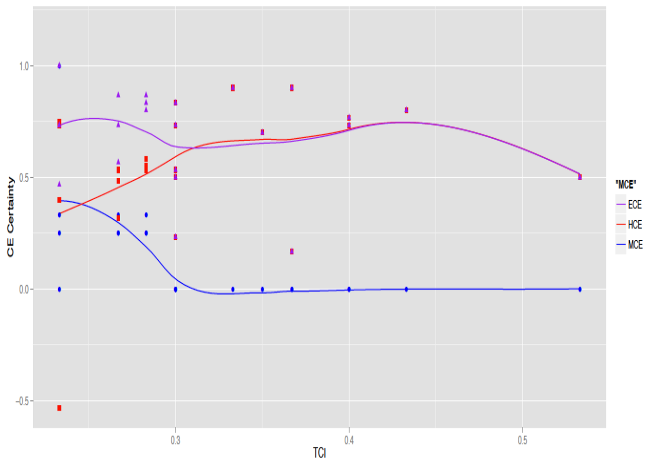


Figure 15. XY-Plot, TCI vs. MCE / HCE /ECE Certainty.

VII. Conclusion and Future Work

In this paper, we propose an elasticity model and framework to leverage the power of the crowd in solving complex problem. We apply our proposed elasticity framework to a face recognition problem. Our experimentations in that domain demonstrate that applying elasticity to the task of face recognition significantly increases the probability of positively identifying an individual in a picture. Virtualizing and provisioning humans as computing elements through crowdsourcing and integrating them with automated approaches produced positive results in test cases where the software alone failed to identify faces.

We have also illustrated the relationship of a task's TCI to the performance index of a computing element to complete the said task. Our results show that ECE in most cases, render the most optimal results. Based on this discovery, we propose the Maximum Performance Index Algorithm, which should in most cases if not all scenarios, produce the maximum performance index from an elasticity framework irrespective of the TCI for a task. From these results and observation, we also propose the Weighted Metric Performance Index Algorithm, which should provide the most optimal results working within the available resource constraints of an elasticity framework.

For future experiments, we intend to implement the MPIA and the WMPIA in our elasticity framework case study for face recognition to validate their design. Also we intend to prove that the TCI proposed in this solution can be qualitatively viewed in varying forms of tasks, this will be done by undertaking case studies in other forms of tasks in which its properties can be quantified.

References

[1] W. Tan, M. B. Blake, I. Saleh and S. Dustdar. (2013). Social-Network-Sourced Big Data Analytics. *Internet Computing, IEEE*, *17*(5), 62-69.

[2] T. Kraska. (2013). Finding the needle in the big data systems haystack. *Internet Computing, IEEE*, *17*(1), 84-86.

[3] S. Dustdar, and H. L. Truong. (2012). Virtualizing Software and Humans for Elastic Processes in Multiple Clouds-a Service Management Perspective. *International Journal of Next-Generation Computing*, *3*(2).

[4] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. (2003). Face recognition: A literature survey. *Acm Computing Surveys (CSUR)*, *35*(4), 399-458.

[5] T. Yan, M. Marzilli, R. Holmes, D. Ganesan, and M. Corner. (2009, November). mCrowd: a platform for mobile crowdsourcing. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems* (pp. 347-348). ACM.

[6] V. Ambati, S. Vogel, & J. G. Carbonell. (2010, May). Active Learning and Crowd-Sourcing for Machine Translation. In *LREC* (Vol. 1, p. 2).

[7] M. J. Franklin, D. Kossmann, T. Kraska, S. Ramesh, and R. Xin. (2011, June). CrowdDB: answering queries with crowdsourcing. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data* (pp. 61-72). ACM.

[8] J. Heer and M. Bostock. (2010, April). Crowdsourcing graphical perception: using mechanical turk to assess visualization design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 203-212). ACM.

[9] M. Z. Candra, R. Zabolotnyi, H. L. Truong and S. Dustdar. (2014). Virtualizing Software and Human for Elastic Hybrid Services. In *Advanced Web Services* (pp. 431-453). Springer New York.

[10] S. Tai, P. Leitner and S. Dustdar. (2012). Design by Units: Abstractions for Human and Compute Resources for Elastic Systems. *IEEE Internet Computing*, *16*(4).

[11] E. Kamar, S. Hacker and E. Horvitz. (2012, June). Combining human and machine intelligence in large-scale crowdsourcing. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1* (pp. 467-474). International Foundation for Autonomous Agents and Multiagent Systems.

[12] M. Z. Candra, H. L. Truong, and S. Dustdar. (2013, June). Modeling Elasticity Trade-Offs in Adaptive Mixed Systems. In *Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2013 IEEE 22nd International Workshop on* (pp. 21-26). IEEE..

[13] C. E.Kulkarni, R. Socher, M. S. Bernstein, and S. R. Klemmer. (2014, March). Scaling short-answer grading by combining peer assessment with algorithmic scoring. In *Proceedings of the first ACM conference on Learning@ scale conference* (pp. 99-108). ACM.

[14] J. Jarrett, I. Saleh, M. B. Blake, S. Thorpe, T. Grandison, R. Malcolm. "Mobile Services for Enhancing Human Crowdsourcing with Computing Elements". IEEE 3rd International Conference on Mobile Services, June 27 - July 2, 2014, Anchorage, Alaska, USA.