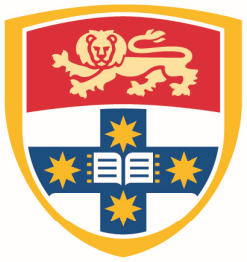
THE UNIVERSITY OF
SYDNEY

Augmenting the Performance of Image Similarity Search through Crowdsourcing



THE UNIVERSITY OF SYDNEY

This thesis is submitted in fulfilment of the requirements for the degree of Master of Philosophy in Information Technology

School of Information Technologies

The University of Sydney

Bahareh Rahmanian

August 2014

Acknowledgements

I couldn't complete this dissertation without Professor Joseph Davis's help, who not only served as my supervisor but also encouraged and challenged me throughout my academic program. Returning to university after many years and immigration to Australia was a big challenge for me and if it wasn't for his support and engagement I wouldn't be able to accomplish it.

I would like to extend sincere gratitude to all members of "Knowledge Discovery and Management Research Group" for exchanging research ideas. Especially I have to thank Dr. Simon Poon, my associate supervisor, for his helpful input and my dearest friend Andrea Stern for her kindness, love and support during my study.

Most importantly, I would like to thank my parents for their constant support and love. Every successful moment of my life is because of them. And finally my gracious thanks to my beloved husband, Sepehr, for all his support, patience and encouragement.

This thesis is dedicated to my loving parents and my husband.

For their love, endless support and encouragement.

Publications

1. Rahmanian, B., & Davis, J. G. (2014). User Interface Design for Crowdsourcing Systems. In *Proceedings of the AVI 2014 conference*. Como, Italy: ACM.

2. Rahmanian, B., & Davis, J. G. (2013). Crowdsourcing, Cognitive Load, and User Interface Design. In *24th Australasian Conference on Information Systems*. Melbourne, Australia.

Table of Contents

List of Figures

List of Tables

List of Acronyms

API	Application Program Interface
ASR	Automatic Speech Recognition
CAPTCHA	Completely Automated Public Turing test to tell Computers and Humans Apart
CBIR	Content-Based Image Retrieval
CG	Control Group
CHC	Crowdsourced Human-based Computation
CLT	Cognitive Load Theory
CS	Crowdsourcing (system)
DB	DataBase
DBMS	Database Management System
DoG	Difference of Gaussians
EMD	Earth Mover's Distance
ERR	Expected Reciprocal Rank
HCI	Human–Computer Interaction
HIT	Human Intelligence Task
HTML	HyperText Markup Language

IDF	Inverse Document Frequency
MD	Majority Decision
MTurk	Amazon Mechanical Turk
OCR	Optical Character Recognition
ORB	Oriented FAST and Rotated BRIEF
QBE	Query-By-Example
SIFT	Scale-Invariant Feature Transform
SFO	Scaled Footrule Aggregation (method)
SQL	Structured Query Language
SURF	Speeded-Up Robust Features
TET	Total Execution Time
TF	Term Frequency
tf-idf	term frequency-inverted document frequency
UI	User Interface

Abstract

The world has witnessed incredible advances in information and communication technology during the past decade. The availability of internet access and the evolution of the World Wide Web have provided an excellent platform for communication and have given rise to a new, efficient, on-demand and affordable workforce made up of humans which has contributed to the rise of crowdsourcing. Crowdsourcing is the concept of "outsourcing a task that is traditionally performed by an employee to a large group of people in the form of an open call" (Howe 2006). Many different platforms designed to perform several types of crowdsourcing (e.g. Amazon Mechanical Turk, InnoCentive, Threadless) and studies have shown that results produced by crowds in crowdsourcing platforms are generally accurate and reliable.

For several years, researchers studied computational algorithms and developed machine learning methods with the goal of increased automation and replaced humans with computers to increase the accuracy and performance of diverse systems. But despite the improvements in computational algorithms, computers still perform very poorly in some fields of research and image similarity search is one of them. Rapid advances in image capturing devices and the availability of online photo storage services have caused the development of very large image databases and these image collections are of limited

value without efficient image retrieval systems. An efficient image browsing, searching and retrieval system is required in various domains, including crime prevention, fashion and medicine. Many image retrieval systems have been developed based on two different approaches, text-based and content-based retrieval mechanisms. Using text-based search methods, text-based image retrieval systems provide a high performance image search system for fully annotated images. While collecting accurate annotations for large image databases is an expensive and time-consuming task, researchers started designing a new generation of image retrieval systems in the early 1980s. This new approach uses raw image data, indexes images based on their visual content and is called content-based image retrieval or CBIR. The fundamental difference between text-based image retrieval and CBIR is that, in the former, human interaction is necessary to provide meta-data (e.g. keyword, annotation) but, in the latter, the search is performed based on image content rather than meta-data. The lack of human interaction and the absence of a direct link between humans' high-level concepts and the low-level features in CBIR systems have resulted in very low performance image similarity search systems.

Crowdsourcing can provide a fast and efficient way to use the power of human computation to solve problems that are difficult for machines to perform. From several different microtasking crowdsourcing platforms available, we decided to perform our study using Amazon Mechanical Turk. In the context of our research we studied the effect of user interface design and its corresponding cognitive load on the performance of crowd-produced results. Our results highlighted the importance of a well-designed user interface on crowdsourcing performance.

Using crowdsourcing platforms such as Amazon Mechanical Turk, we can utilize humans to solve problems that are difficult for computers, such as image similarity search. However, in tasks like image similarity search, it is not possible to ask crowds to search within a database of millions of images; therefore, it is more efficient to design a hybrid human–machine system. Several researchers have studied the design of hybrid human–machine systems to cover the semantic gap of computational algorithms and human perceptions. In the context of our research, we studied the effect of involving the crowd on the performance of an image similarity search system and proposed a hybrid human–machine image similarity search system. Our proposed system uses machine power to perform heavy computations and to search for similar images within the image dataset and uses crowdsourcing to refine results. In another words our hybrid system is system composed of a CBIR retrieval algorithm to achieve recall and shallow filtering and a crowdsourced-based human input to achieve precision. We designed our CBIR system using SIFT, SURF, SURF128 and ORB feature detector/descriptors and compared the performance of the system using each feature detector/descriptor. Our experiment confirmed that crowdsourcing can dramatically improve the CBIR system performance.

Chapter 1

Introduction, Background and Motivation

1.1 Introduction

Humans are always searching for ways to automate and speedup their tasks. Invention of electronic computers and the Internet was a big step forward and since then computers are helping humans in solving complex mathematical problems, storing and retrieving large amounts of data and automating tasks. Researchers around the world are working on new algorithms and devices to replace humans with machines to increase speed and accuracy.

While machine are very good at computations and dealing with large amount of data, humans perform better in tasks that involves perceptual comparison and decision making. Despite magnificent advances in computational algorithms, there are still some tasks that computers have very low performance with high speed, but humans perform very well but with low speed and low efficiency. In these tasks designing a hybrid system that uses the computational power for increased speed and human power for increased accuracy is

a very good solution. Humans can be involved in computational algorithms by the help of crowdsourcing. Crowdsourcing provides a very efficient and fast way to recruit humans to provide answers to the problems that computers are unable to.

Search for similar images or Query-By-Example image search is one of the tasks that computer systems don't achieve a high level of performance. But because the concept of similarity goes beyond the mare matching of visual feature, comparing images and determining their similarity is a very easy task for humans. Through our research we proposed a hybrid Human—Machine image similarity search and compared its performance against pure computational image similarity search.

1.2 Crowdsourcing

In the past decade, the World Wide Web has evolved into a powerful medium for active collaboration among people located around the world. The evolution of the World Wide Web and its transition from Web 1.0 (read only web) to Web 2.0 (read-write web) have made it easier to involve users in making its contents and sharing knowledge. Nowadays, users are not only consumers of content on the Web but also providers of data and the source of a new kind of computation. Many successful examples exist of people coming together on the Web to combine their resources – whether it is knowledge, creativity, opinions, skills, etc. – including the world's largest knowledge base Wikipedia and the problem-solving platform InnoCentive. These phenomena are commonly referred to as "crowdsourcing": this term has been coined by Jeff Howe and describes a new distributed problem-solving and business model. Howe defined "crowdsourcing" as "an idea of outsourcing a task that is traditionally performed by an employee to a large group

of people in the form of an open call" (Howe 2006). Crowdsourcing has evolved over the years into a range of endeavours including open innovation, distributed human computation, prediction markets, crowdfunding and crowdservicing, to name a few (Davis 2011).

Crowdsourcing can provide good solutions to a wide range of applications. The power of humans can replace computational algorithms in fields in which computers perform poorly. Providing annotations for images (VonAhn & Dabbish 2004), an iPhone app providing answers blind people questions (Bigham et al. 2010) are some examples of crowdsourcing applications. Researchers have also used crowdsourcing to compute ground truth data. Their experiments outcome showed that results generated using this process are reliable and can be used as ground truth (Urbano et al. 2010).

1.2.1 Microtasking and MTurk

Microtasking is a type of crowdsourcing in which larger tasks are broken into smaller short-duration tasks. These small microtasks are performed by more than one crowdworker and the aggregated result is assumed to be the solution to the microtask. There are several platforms for microtasking (Microtask.com, CrowdFlower, Amazon Mechanical Turk) and we chose Amazon Mechanical Turk (MTurk) in our study (more discussion on choosing the platform is provided in Chapter 5).

In MTurk, requesters can post their tasks. Workers sign onto the system, search for their preferred tasks, accept and solve the tasks, and send the results back to MTurk.

Microtasks on MTurk are referred to as HITs (human intelligence tasks) and are grouped into HITGroups. Each requester can assign the same HITs to more than one worker.

MTurk provides some tools for requesters to implement microtask-based crowdsourcing. Requesters can chose between the web-based user interfaces (UIs) to create simple HITs and collect results, or create more complex HITs using the specialized UIs from Amazon's API for MTurk. MTurk APIs support a variety of programming languages.

1.2.1.1 User Interface Design in MTurk

Regardless of the approach used to create HITs (web UI or API), all tasks are shown in an iframe[1] inside workers' main web interface page. In order to view the HIT and complete the task, workers need to scroll within this iframe. This limited HIT design environment highlights the importance of a good UI design which has the potential to affect the quality of results provided by workers. A poorly designed UI can result in low quality results of the crowdsourcing task, or discourage the workers from accepting the task and increases the time needed to finish it.

As a part of our research, we conducted an experiment to study the effect of different UI designs with assumed different cognitive loads on the performance of the results produced by workers and also the time that it took for the task to be completed. The experiment and results are explained in Chapter 3.

[1] An iframe is a frame used to display a web page within another web page.

1.3 Image Retrieval

During the past decade, the world has witnessed a rapid increase in the size of digital image collections and making use of these collections is not possible unless they are organized and allow efficient browsing and retrieval. There are two dominant trends in the image retrieval field: the first one is text-based image retrieval and the second one is content-based image retrieval. In text-based (typically annotations) image retrieval techniques, the whole dataset of images is manually annotated by text and then image retrieval is performed using a text-based database management system (DBMS) (Chang & Fu 1979; Chang et al. 1997).

There are two challenges regarding text-based image retrieval systems. The first one is that providing annotations for images requires a considerable level of human labour and the second one is inaccuracy of the provided labels. To overcome these two challenges, another image retrieval system was introduced in the early 1980s and attracted a large community of researchers (Gupta & Jain 1997; Vailaya et al. 2001; Rahmani et al. 2008; Loy & Eklundh 2006). This approach is called content-based image retrieval or CBIR in which feature extraction, multidimensional indexing and retrieval system design are the three fundamental bases (Rui et al. 1999).

1.4 Limitations of Computational Algorithms

Developments in computer science and Artificial Intelligence (AI) have made it possible to replace humans with computers in fields where speed and efficiency is important (factories, repetitive tasks, etc.) and it is predicted that humans will lose their

jobs in many more fields (Aquino n.d.; Burn-Callander 2013). But despite these advances in computational algorithms and the design of powerful hardware/software to speed up calculations, there are still applications where computers perform very poorly but humans have a very high performance. One of these areas is image processing and specifically CBIR systems.

In CBIR systems, low-level features (colour, texture, shape, etc.) are extracted automatically using computer vision techniques. As previously noted, in text-based image retrieval systems, human interaction is one of the main parts of the system. In these systems, humans tend to interpret images and measure their similarity using high-level features, such as keywords and text descriptors. This human interaction makes the fundamental difference between text-based and content-based image retrieval systems. Experiments have shown that low-level contents in CBIR systems fail to describe the high-level semantic concepts in the user's mind (Zhou & Huang 2000) and this gap has caused very low performance of CBIR systems.

Crowdsourcing can provide a fast and efficient way to use the power of humans to decrease the semantic gap in CBIR systems. The resulting system will be a human–machine hybrid system.

1.5 Human–Machine Hybrid Systems

In the previous section, we pointed out the limitations of computational algorithms in CBIR systems. There are also other areas of research where computers have very low performance (e.g. handwriting and speech recognition). On the other hand, humans

perform very well in these areas and especially in image comparison, but asking the crowds to search within a large image dataset is a very expensive and time-consuming task. Researchers have tried to cover the limitations of computational algorithms by designing a hybrid human–machine system in which machines are used to do the initial heavy computation task, and people are used to verify the most likely results.

CrowdSearch (Yan et al. 2010) is an example of a machine and crowd combination and is a real-time image search on mobile phones which uses machine computation to search for similar images based on a given query image. Results of the computational algorithm are given to crowds to be validated and the most accurate search result is selected and returned to the user. This system not only puts heavy machine computations and human power together, but also provides a trade-off model of energy, delay, accuracy and cost. CrowdER (Wang et al. 2012) and CROWDSAFE (Shah et al. 2011) are two other examples of such hybrid systems.

1.6 Research Questions

As previously noted, humans outperform computational algorithms in some areas such as CBIR systems. Involving the crowd with the purpose of improving the system's performance has been tested in many research fields (e.g. image annotation (Russell et al. 2007), filling missing database data (Franklin et al. 2011)) but there is not enough study on the hybrid human–machine CBIR system. We believe that the power of crowds in the conceptual comparison of images can overcome the limitations of the computational image similarity system and result in a higher performance of the CBIR system.

Can the power of the crowd overcome the limitations of a CBIR system and does crowdsourcing improve the performance of a CBIR system using a hybrid human–machine system?

To test our hypothesis, we designed a CBIR system using four different feature detector/descriptors (SIFT, SURF, SURF128 and ORB) to search for similar images based on a query image. We combined this system with crowdsourcing using MTurk. We compared the performance of the system in each stage and also we compared the performance of each feature detector/descriptor. Our system and experiment are explained in Chapters 4 and 5.

As a part of our research, we studied different user interface (UI) designs and their effect on workers' performance and execution time of the crowdsourcing task. Our research questions about UI design in crowdsourcing platforms are:

Do user interface design and its corresponding cognitive load affect the performance of crowdworkers?

Does the user interface design affect the execution time of the crowdsourcing task?

We conducted experiments to test our hypotheses based on these suggestions. This study and the results are explained in Chapter 3.

1.7 Outline of Research

This thesis is organized as follows:

Chapter 1 Introduction, Background and Motivation: This chapter introduces crowdsourcing and our research problem. We summarised the motivation and the approach taken to answer the research question.

Chapter 2 Review of Existing Literature on Crowdsourcing: This chapter provides the relevant literature to build the theoretical foundation of this research.

Chapter 3 User Interface Design in MTurk: The methodology, experiment and results for user interface (UI) design are explained in this chapter.

Chapter 4 Content-Based Image Retrieval System: The architectural design of the CBIR system that we used for our experiment is explained in this chapter.

Chapter 5 Hybrid Human–Machine System: Chapter 5 describes the method and procedures that we used to design a human–machine hybrid system.

Chapter 6 Conclusions: This chapter presents the discussion and conclusion of the research.

Chapter 2

Review of Existing Literature on Crowdsourcing and CBIR

In this chapter, crowdsourcing systems and their properties are studied in detail. There are different types of crowdsourcing systems and platforms with different types of motivators. We have provided a detailed review of existing crowdsourcing systems. Another part of our research is using a Content Based Image Retrieval (CBIR) system. A brief review of CBIR systems is also provided in this chapter.

2.1 Crowdsourcing Definition

Crowdsourcing by definition is using the intelligence of people to complete tasks in an open call. The word "crowdsourcing" was first coined by Jeff Howe in *Wired* Magazine in 2006 and is a portmanteau word combining "crowd" and "outsourcing". Howe defined "crowdsourcing" as "an idea of outsourcing a task that is traditionally performed by an employee to a large group of people in the form of an open call"(Howe 2006). In other words, crowdsourcing is the act of obtaining needed services from a large group of the

online community. Wisdom of the crowd or collective intelligence, crowd creation or user-generated content, crowd voting and crowd funding are four categories of crowdsourcing applications defined by Howe and discussed in detail by (Yuen et al. 2011) and (Erickson 2012).

The widespread Internet accessibility has led to the growth of crowdsourcing systems and caused a surge of research activity in crowdsourcing. Many researchers have contributed to a growing literature of crowdsourcing which can describe applications, algorithms, performance and datasets (Yuen et al. 2011; Doan et al. 2011; Wightman 2010; Zhang et al. 2011).

While (Schneider et al. 2012) suggests that peer production, crowdsourcing, mass collaboration, mass persuasion, human computation, collective intelligence and crowd–computer interaction together bring droves of people to collaborate, (Quinn & Bederson 2011) tried to distinguish the differences and overlaps of human computation, crowdsourcing, social computing, collective intelligence and data mining concepts and drive a taxonomy of human computation. In their taxonomy, human computation overlaps with crowdsourcing in situations where humans and computers already have roles which can be replaced by each other (e.g. translation). Collective intelligence is a superset of social computing and crowdsourcing but the distinction between human computation and collective intelligence is where human computation jobs involve performing a task by an individual isolated human. The goal in human computation is to select computational tasks actively and assign them to the right workers to minimize cost and maximize quality (Law & von Ahn 2011).

2.2 Classifying Crowdsourcing Systems

There are many studies on the classification of crowdsourcing systems. Researchers have divided crowdsourcing systems into multiple classes based on the different behaviours of crowdsourcing tasks. (Wightman 2010) classified crowdsourcing systems on their competitiveness and their motivation behaviour into four categories. In "non-competitive direct motivation" tasks like image labelling, Wikipedia or news aggregation websites, computers can be used to coordinate humans, and humans are motivated by the task itself. In designing these kinds of tasks, the difficulty of the task and its accessibility for humans should be considered and methods might be needed to filter inaccurate information.

CAPTCHA is a completely automated public test to verify if a user is a human or a robot pretending to be human. Von Ahn designed reCAPTCHA as a web service so people not only prove themselves to be human but also digitize texts which OCR (Optical Character Recognition or image to speech) systems are unable to translate (VonAhn et al. 2004); (von Ahn et al. 2008). This crowdsourcing system is an example of a "non-competitive indirect motivation" task in which designers modify an existing task to achieve a CHC (Crowdsourced Human-based Computation) goal by providing incentives and might achieve an improved response rate if they use less advertising approaches.

The third category of crowdsourcing tasks is defined as those which are "competitive with indirect motivation". Amazon Mechanical Turk (MTurk) and InnoCentive are good examples of such systems in which users are motivated to participate due to the ease of

performing tasks in distractive environments and the ability to earn some money in their spare time.

The last category defined by Wightman is “competitive direct motivation” tasks. In systems like Yahoo! Answers, users are directly motivated to be competitive; thus, collusion control might be considered.

While Wightman categorized human-based computation tasks from the point of view of motivation and competitiveness, (Doan et al. 2011) provided a global picture of the crowdsourcing systems of the Web. They classified crowdsourcing (CS) systems by the nature of the collaboration (implicit and explicit systems); architecture of the system (stand-alone or piggyback); whether or not they recruited people; what users could do; and the type of target problem. They demonstrated that explicit stand-alone systems that recruit users can be used for evaluation (review, vote, tag, e.g. voting at Amazon); sharing items, textual knowledge and structured knowledge (e.g. YouTube, Flicker, Yahoo! Answers); networking (LinkedIn, Facebook); and building artifacts (Linux, Wikipedia, InnoCentive). They also implied that implicit stand-alone CS systems that recruit users can be built for tasks such as labelling images and rating movies (e.g. ESP, IMDB). Spell correction and product suggestion are examples of implicit piggyback CS systems.

(Geiger et al. 2011) adopted a different approach to the classification of crowdsourcing systems and suggested defining a taxonomy which can be applied to all forms of crowdsourcing systems. As a taxonomy definition needs the users and purpose to be defined, they assumed organizations to be the users who try to reach a certain goal by crowdsourcing and it does not matter if the decision is made in-house or not: for the

purpose, they tried to clarify the process of crowdsourcing and derived meta-characteristics which must apply to all kinds of crowdsourcing organizations. They categorized characteristics in four different stages: firstly, preselection of contributors; secondly, accessibility of peer contributions; thirdly, aggregation of contributions; and, lastly, remuneration for contributions. Figure 2-1 shows these classifications and their characteristics.

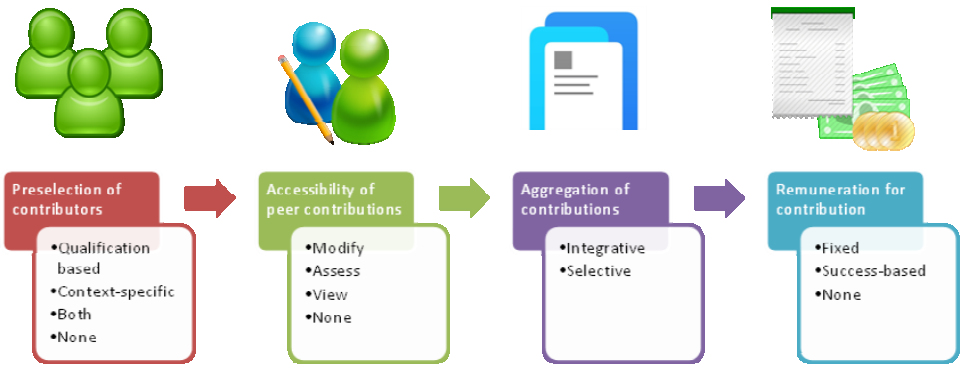


Figure 2-1. Characteristics of Crowdsourcing Process (Geiger et al. 2011, p19)

Clustering and 96 possible combinations of process characteristics in 46 examples resulted in 19 distinct types which can be classified in five distinct clusters: interactive sourcing without remuneration (Delicious, Wikipedia); selective sourcing without crowd assessment (Netflix Prize, InnoCentive, 99Designs); selective sourcing with crowd assessment (InnoCentive@Work, Atizo); interactive sourcing with success-based remuneration (Android Market, iStockPhoto); and interactive sourcing with fixed remuneration (MTurk).

By studying crowdsourcing systems, some patterns can be found. (Erickson et al. 2012; Erickson 2012) undertook some research on finding the patterns associated with the use of crowdsourcing as has been established by organizations. They found five reoccurring themes related to crowdsourcing with their characteristics summarised in Table 2-1.

Table 2-1. Emergent Themes Related to Crowdsourcing (Erickson et al. 2012 p94)

Engagement Themes	Characteristics
Common Tasks	• Routine time-consuming activities • Data collection • Knowledge sharing • Marketing • Ideation • Design • Development • Filtration • Evaluation • Complex problem solving
Crowd Knowledge	• General • Situational (e.g. time, place, event) • Product/Service • Specialized • Domain expertise
Crowd Location	• Internal • External
Organizational Challenges	• Accuracy • Availability • IP leakage/Loss of competitive advantage • Clear articulation of the task • Internal acceptance/buy-in • Motivation of the crowd • Loss of control
Value Capture	• Tangible • Intangible • Immediate • Delayed

When Erickson et al.'s five themes are applied to four major basic categories of crowdsourcing (productivity, innovation, knowledge capture and marketing/branding), the result is their suggested framework as shown in Table 2-2.

Table 2-2. Preliminary Framework for Crowdsourcing Uses and Key Characteristics (Erickson et al. 2012, p94)

	Productivity	Innovation	Knowledge Capture	Marketing/Branding
Organizational Motivation	• Reduction in costs • Replacing current resources	• Retaining/Gaining competitive advantage, increasing innovative potential • Supplementing current resources	• Advancing understanding or accuracy • Creating new knowledge resources	• Increasing profits and brand affinity • Supplementing current resources
Common Tasks	• Routine time-consuming activities difficult to automate	• Ideation • Evaluation • Filtration • Design • Development • Problem solving	• Data collection • Knowledge sharing	• Creative • Market insights
Crowd Knowledge	• General • Specialized	• Product/Service • Specialized • Domain expertise	• Product/Service • Situational • Domain expertise	• Product/Service • Specialized
Crowd Location	• External	• Internal • External	• Internal • External	• External
Organizational Challenges	• Accuracy/ Quality of work • Availability	• IP leakage/Loss of competitive advantage • Clear articulation of the task • Internal acceptance/buy-in	• Motivating the crowd to share	• Control of the crowd
Value Capture	• Tangible • Immediate	• Tangible • Delayed	• Tangible • Immediate and delayed	• Tangible • Immediate and delayed

This framework shows that, for example, routine time-consuming tasks which are difficult to automate can be crowdsourced to the external crowd; cost reduction and replacing current resources will motivate the organization to do crowdsourcing; crowd knowledge can be general or specialized; the organization faces the challenges of accuracy and quality of the work, and the availability of the crowd; and the captured value is tangible and immediate.

Erickson et al. grouped crowdsourcing tasks into categories of productivity, innovation, knowledge capture and marketing/branding, while (Yuen et al. 2011) classified CS applications into four different groups: voting systems, information sharing, game systems and creative systems which overlap with the former categorization in the two groups of information sharing and creative systems.

2.2.1 Microtasking Crowdsourcing Platforms

Microtasking is the act of breaking large and complex tasks into smaller tasks and asking multiple crowdworkers to perform them. The aggregated result from crowdworkers is the result of the microtask. Most microtasks just take minutes to complete but there are also more complex tasks. Many platforms have been designed for microtasking crowdsourcing and Amazon Mechanical Turk (MTurk) and CrowdFlower are the two largest platforms.

2.2.1.1 Amazon Mechanical Turk

Amazon Mechanical Turk[2] (MTurk) is one of the best platforms designed to facilitate the crowdsourcing of microtasks. MTurk is a marketplace in which requesters can put their task and workers can sign into the system and do the tasks. HITs (Human Intelligence Task) are grouped into HITGroups and each requester can assign each HIT to more than one worker to perform the task. Most of the rewards on MTurk are typically between USD0.01 to USD0.10 which can be paid if the worker completes the task satisfactorily. Each task typically takes no longer than a minute but, in the extreme, some

[2] http://www.mturk.com

tasks may require an hour to complete. Some of the HITs are just one single task but some can be a collection of tasks, for example, comparison between 50 images. MTurk provides two interfaces for its systems: one is the main interface of the website that workers use to search for HITs and the other is the user interface provided for HITs. MTurk also provides its own API by which requesters can automate their workflow if publishing HITs and collecting results. Different types of crowdsourcing tasks can be performed using MTurk and studies such as (Franklin et al. 2011; Bernstein et al. 2010; Pai & Davis 2012; Williams et al. 2011; Yan et al. 2010) are some examples.

2.2.1.1.1 MTurk User Demographics

Many studies have been conducted on the demographics of workers on MTurk (Silberman, Irani, Tomlinson, et al. 2010; Ross et al. 2010; Silberman, Irani & Ross 2010). Comparing (Gosling et al. 2000) study with that of (Buhrmester et al. 2011) showed that MTurk participants came from over 50 different countries and gender splits were similar in the standard Internet (57% female) and MTurk (55% female) samples. A greater percentage of MTurk participants were non-white (36%) and almost equally non-American (31%) compared with the Internet sample (23% and 30%, respectively). MTurk participants were older than the Internet participants. In short, MTurk participants were more demographically diverse than standard Internet samples and significantly more diverse than typical American college samples.

Surveys by (Ross et al. 2010) and (Ipeirotis 2010) have shown that in the time period from 2008 to 2010, workers became more international. In 2009, (Ross et al. 2010) found that 57% were from the US and 32% were from India compared to Ipeirotis's (2010)

findings of 46.8% from the US, 34% from India and 19.20% from miscellaneous countries. They also found that a growing population of young educated male Indian workers earn less than USD10,000/year and also that 31% of Indian workers and 13% of US workers always or sometimes rely on MTurk as their primary source of income.

An experiment by Downs et al.'s (2010) showed that young men (under 25 years old) tend to game the system more than older men and also more than women of all ages. Professionals, students and non-workers seem to take the task more seriously.

2.2.1.1.2 Cognitive Load and User Interface Design in MTurk

Cognitive load refers to the amount of mental resources required to process a given task; the higher amount of information needed to process a task, the more cognitive load the task has. Humans' mental resources are limited and when the amount of information and instruction for a task exceeds this limit, learning will be inhibited and performance will decrease. (Sweller 1988) described a model of cognitive load and distinguished three distinct memory types: sensory memory, working memory and long-term memory.

Recent studies have focused on cognitive load and suggested that the limited working memory is the critical bottleneck in human information processing. Through these studies, three types of cognitive load are distinguished by Cognitive Load Theory (CLT): intrinsic, extraneous and germane. Intrinsic cognitive load is related to the level of expertise of a learner and is defined by the intrinsic complexity of information that is to be learned (Sweller et al. 1998; Bannert 2002). Extraneous cognitive load is defined by any cognitive load associated with the way the task can be carried out and caused by activities that are irrelevant to the task (Ayres & Sweller 2005). As found by Paas and

Van Merrienboer (1994), the variations of worked example types support the construction of schema but, at the same time, increase cognitive load. This type of cognitive load is introduced as germane cognitive load.

CLT provides a basis by which to predict user performance when using different user interface (UI) designs: it also gives guidelines to minimize cognitive load in the design of user interfaces. In typical educational systems, monitoring and lowering cognitive load lead to increased student learning ratios. CLT research has also addressed techniques for decreasing extraneous cognitive load for these systems (Reis et al. 2012) and has tried to design new interfaces that effectively minimize students' cognitive load. Applying these findings to an educational system's UI will help students focus on the learning task and learn efficiently. Studies have also found that using principles of user-centred design and CLT leads to minimized extraneous cognitive load of the task (e.g. user input planning, minimizing interruptions by eliminating unnecessary features, and applying split-attention effect, redundancy effect and modality effect learning techniques) (Erry et al. 2006; Feinberg & Murphy 2000; Oviatt 2006).

Educational systems are not the only systems in which UI design directly affects user performance. While, in crowdsourcing systems, humans (workers) play the main role in solving problems, their performance directly affects the overall quality of the crowdsourcing tasks. One of the disadvantages of MTurk as a crowdsourcing platform is its limitations in the HIT interface environment. As previously mentioned, MTurk provides two separate web-based interfaces for requesters and workers. The third interface, which is the subject of this study, is the HITs' interface. This is the interface

that workers are facing to perform the crowdsourcing task, and is shown in an iframe inside MTurk's web page. This limited visual space of the HITs makes designing the HIT UI an important consideration which can directly affect crowdsourcing task performance. The importance of a well-designed UI might not be much highlighted in other crowdsourcing platforms.

It is very common in crowdsourcing tasks to ask more than one worker to perform the same task. The final solution is then created by aggregating responses. If the quality of responses produced by each worker is low, requesters have to send more tasks and collect more results to have to a proper solution to the crowdsourcing task. In this situation, the crowdsourcing task will cost the requester more money. By increasing workers' performance and avoiding low quality results, the overall cost of the crowdsourcing task will be decreased.

2.2.1.2 CrowdFlower

CrowdFlower, which is very similar to MTurk, is another platform through which to do crowdsourcing. Like MTurk, CrowdFlower has a requester UI and its own API so requesters can interact easily with it. In addition, CrowdFlower offers a higher degree of quality control called "gold-standard data". Gold-standard data are pre-completed tasks provided by the requester to determine workers' accuracy and trustworthiness. CrowdFlower also claims to have multiple labour channel partners such as MTurk and TrialPay.

2.2.1.3 microWorkers.com

Another platform for microtasking or micro jobs is microWorkers.com and is an international platform for connecting workers and employers from all around the world. Unlike MTurk, there isn't any restriction on the country of residence and any one can sign into the systems and use it for free. Tasks on microWorkers.com are simple and easy tasks such as sign-ups, social bookmarking tasks, forum participation, website visits, rating videos or articles, voting up contest entries, adding comments, suggesting leads, creating backlinks, writing reviews or articles, downloading applications.

2.2.2 Crowdsourcing Examples

Crowdsourcing has a wide range of applications and can be a problem-solving method in a wide variety of domains. Crowdsourcing applications vary from simple microtask annotation tasks (von Ahn & Dabbish 2004; Rashtchian et al. 2010) and multimedia retrieval (Snoek et al. 2010) to complex text editing jobs (Bernstein et al. 2010) or even collaborative coding (Goldman et al. 2011).

There are some applications that computers are unable to perform well and, in these cases, crowdsourcing can provide a very high performance solution. An example of such cases is providing text annotations for images to help improve an image search system. Von Ahn and Dabbish (VonAhn & Dabbish 2004) designed a system called ESP Game to ask for image labels through a computer game. Using crowdsourcing to evaluate colours (Xue et al. 2012) and providing cheap speech data for speech recognizers through mobile phones (Ledlie et al. 2010) are other examples of crowdsourcing systems that are used to

collect information for computational processing. Studies have shown that results generated using this process are reliable and can even be used as ground truth (Urbano et al. 2010).

An Amazon Mechanical Turk (MTurk) study on using crowdsourcing to assess visualization (Heer & Bostock 2010) shows that by using qualification tests before assigning actual tasks to workers, the quality of results provided by crowdworkers significantly increases.

Other than just performing operations, humans can handle the control flow of the algorithm. CrowdForge (Kittur et al. 2011); Turkomatic (Kulkarni et al. 2011; Kulkarni et al. 2012); CDAS (Liu et al. 2012); and TurKit (Little et al. 2010b; Little et al. 2009) are examples of frameworks in which the crowd takes control of the workflow and decides to solve a problem by decomposing it into smaller parts and then combines results to make the final result.

In the following section, we study some major examples of crowdsourcing systems.

2.2.2.1 Games with a Purpose

The ESP Game presented by von Ahn and Dabbish (VonAhn & Dabbish 2004) is a computer game to provide image labels for images to improve image search performance. It uses the concept of games with a purpose (VonAhn 2006) and organizes the power of the crowd implicitly in a funny way to label images by non-expert users.

reCAPTCHA (VonAhn et al. 2008) is another example of implicit piggyback systems which is widely being used to verify humans from robots. reCAPTCHA's main

application is to digitize texts which are not translatable by OCR systems. By using this tool, not only do humans verify themselves as humans but they also help to solve problems which are difficult for computers to solve. The mechanism is very simple: it shows two words to users for verification, one of them is already known for the system and the other one is unknown. By comparing data provided by the user for the known word, the user is verified and the data that the user provided for the unknown word are collected and by performing the aggregation method, the correct digitized translation for the unknown text will be provided. By major websites running this tool across the world, they have tried to digitize the whole *New York Times* newspaper archive.

Asirra (Elson et al. 2007) is another CAPTCHA that can be used as a service to identify humans from robots. With Asirra, users have to pick cats out of 12 pictures of cats and dogs and Elson et al.'s study shows that in 99.6% of the time, the task can be done in less than 30 seconds. The image dataset is provided by Petfinder.com and Asirra shows a link for "adopt me" under each photograph to help Petfinder.com find homes for homeless animals.

2.2.2.2 Design of Gold Standards or AI Training Sets

To see if crowdsourcing can create a repeatable and reliable search system evaluation campaign (Blanco et al. 2011; Nowak & Rüger 2010), experiments have shown that it is possible to make a crowdsourced "gold-standard" which is repeatable and will not change from time to time. With regard to their experiments, crowdsourced judgments are different from those of experts due to the object retrieval task and the time pressure on workers but the rank ordering of systems does not change. They found that three

judgments seem to be sufficient and increasing the number of judges in a crowdsourcing system has little effect.

(McDuff et al. 2011) tried to create a large bough dataset for studying natural and spontaneous facial responses using crowdsourced data. They designed a framework and collected over 3000 trackable videos on 54 days from locations across the world. Their method used popular media to motivate participants rather than payment or recruitment, and the dataset they provided has a more dynamic range of position, scale, pose, movement and illumination of participants in comparison with traditional MMI, CK+ and Forbes datasets.

To compare the quality of non-expert annotators with the existing gold standard for annotation, (Snow et al. 2008) proposed experiments in five tasks: affect recognition, word similarity, recognizing textual entailment, temporal event recognition and word sense disambiguation. They found that for many tasks only a small number of non-expert annotations per item are equal to the performance of an expert annotator. In greater detail, they declared that for the face recognition task, an average of four non-expert labels per item is enough to emulate expert-level label quality.

2.2.2.3 Mobile Crowdsourcing

Widespread daily access to Smartphones with Internet connectivity offers a great platform for crowdsourcing applications. In addition, audio-visual sensors, geo-location and other sensors on Smartphones provide an efficient way of collecting data in crowdsourcing systems. Crowdsourcing applications on Smartphones can be in the form

of web-based applications or platform-dependent mobile applications. CrowdTranslator (Ledlie et al. 2010), ParkJam (Kopecký & Domingue 2012) and Waze[3] are three examples of crowdsourcing mobile applications: the first one is a web-based application and the others are mobile applications.

In developing regions with limited access to mobile Internet, crowdsourcing mobile applications can be designed using SMS or GSM data. mClerk (Gupta et al. 2012) is an example of a successful mobile crowdsourcing system which is designed in developing regions in India. In these regions, low-income workers do not have access to technology or accessing the Internet is very expensive, and also their lack of English language prevents them from contributing in web-based systems. mClerk uses an SMS system not only for text-based tasks but also for sending small bitmap images. By the means of mClerk, (Gupta et al. 2012) provided a system for digitizing local-language documents. They automatically segment the scanned forms, send it as an image to workers and collect the text in English form. Correctness is checked by duplicating the task to multiple workers, and then the corrected English text is converted to the local language. Their experiments discovered that the ideal users for this kind of crowdsourcing are the ones who have occupations that allow them to have free time as well as social interactions.

Another example of crowdsourcing mobile applications that does not use the Internet for communication is txteagle (Eagle 2009). txteagle is being launched and operated in Kenya and Rwanda as a successful system and is capable of crowdsourced translation, transcription and survey tasks through GSM services in cooperation with mobile phone

[3] Waze.com

providers in those countries. Through experiments, it has been shown that groups of people, who were mostly taxi drivers, security guards and students, successfully completed translation tasks with 75% accuracy, and students completed twice as many tasks as taxi drivers and security guards.

2.2.2.4 Hybrid Human–Machine Systems

Computers perform very well in repetitive tasks or heavy mathematical computations but in tasks such as image similarity checks or text editing, computers have very low performance. One solution to improve the performance in these tasks is to take advantage of humans or crowdsourcing. We call these systems "hybrid human–machine systems" in which computers do the heavy computational tasks and crowds validate the results.

Some examples of hybrid human–machine systems are called crowd query processing systems (Franklin et al. 2011; Marcus, Wu, Karger, et al. 2011; Parameswaran & Polyzotis 2011; Marcus, Wu, Madden, et al. 2011). CrowdDB (Franklin et al. 2011) is a new prototype to design new database systems. Franklin et al. designed this new prototype to overcome some limitations of traditional database systems. They used the power of the crowd to fill incomplete data and also to provide new data. In this system, queries are very similar to the traditional SQL (Structured Query Language), but crowd features added to this system make it more efficient in comparison with the traditional database (DB). In their proposed prototype, if any piece of information is missing from the database or if there is a need for conceptual comparison (e.g. image comparison), CrowdDB will produce proper HITs and the required user interface (UI) and publish them on MTurk.

CrowdSearcher (Bozzon et al. 2012) is an example of crowdsourcing query processing which was inspired by CrowdDB. Bozzon et al. aimed to fill the gap between computerized search systems which operate on worldwide information and social systems which are capable of interaction with real people and can capture their opinions. They proposed their new prototype to not only use the crowd to fill data as in the former prototypes, but also to add human suggestions and insights in order to improve the answers for more complex queries. CrowdSearcher uses social platforms such as Facebook, Twitter and LinkedIn to provide search-related tasks.

Another prototype developed with the help of MTurk is Soylent (Bernstein et al. 2010), a word-processing interface of which the main purpose is to integrate human expertise with writing tools. Shorten, Crowdproof and The Human Macro are Soylent's three main components which use the power of paid workers through MTurk to help with proofreading, document shortening, editing and commenting tasks. Bernstein et al. have shown that the combination of Soylent and Microsoft Word's grammar check can correct 82% of grammar errors, and also that Soylent shortened text to 85% of its original length.

In order to help blind people solve their visual problems, many expensive talking devices have been designed which use OCR to convert images to speech. However, unfortunately OCR systems are unable to identify the text in many real-world situations, such as handwritten texts or even the street name on a street sign. VizWiz (Bigham et al. 2010) is an iPhone application designed to help blind people address their visual problems. VizWiz uses its own abstraction layer on top of MTurk, which is called "quikTurk", to provide a pool of ready workers to answer visual questions and reducing

the response time on average to 30 seconds. By using VizWiz, blind people can take a photo of what they want to visualize, record a question and send to people to answer in a very short time and at a low cost compared with other commercial systems.

CrowdSearch (Yan et al. 2010) is another example of the combination of machines and crowds and is a real-time image search on mobile phones which uses machine computation to search for similar images based on a given query image. Results of the computational algorithm are given to crowds to be validated and the most accurate search result is selected and returned to the user. This system not only puts heavy machine computations and human power together, but also provides a trade-off model of energy, delay, accuracy and cost. CrowdER (Wang et al. 2012) and CROWDSAFE (Shah et al. 2011) are two other examples of such hybrid systems.

2.3 Challenges in Crowdsourcing

2.3.1 How to Recruit Crowdworkers

The designer of a crowdsourcing system may face several challenges regarding humans. (Doan et al. 2011) implies that one of the challenges of CS systems is recruiting users. The first way that he suggests is to require users to make a contribution. This means that in a company, the manager can require employees to help build a company-wide system. Stewart et al.'s (2009) additional research on crowdsourcing for enterprises suggested that incentives in company-wide CS systems are different from public domain CS. This research showed that optimizing the portal for participants makes enterprise crowdsourcing successful. In another study on crowdsourcing inside the enterprise,

Stewart et al. (Stewart et al. 2010) proposed a SCOUT ((S)uper Contributor, (C)ontributor and (OUT)lier) model for describing user participation and showed that that it is possible to achieve a more equitable distribution of 33-66-1 instead of the general 90-9-1 rule. The cheapest solution for recruiting users in CS systems is to ask for volunteers. Wikipedia, YouTube and Geo-Wiki (Fritz et al. 2009) are three examples of knowledge-sharing CS systems in which users contribute voluntarily.

2.3.2 Incentives

Another challenge in crowdsourcing systems is how to motivate users to contribute in the system. Several research studies have been conducted to find out the true incentives of crowds in crowdsourcing systems to improve the performance of systems. Cuel and Zamarian (Tokarchuk et al. 2012) surveyed past research and categorized motivations into eight classes: reciprocity and expectancy; reputation; competition; altruism; self-esteem and learning; fun and personal enjoyment; implicit promise of future monetary rewards; and money. They designed a framework for studying motivations on the crowdsourcing platform which is based on goals, the task, the social structure and the nature of good variables.

In order to design an efficient crowdsourcing mechanism, (Archak 2010) studied incentives and strategic choices of participants on TopCoder.com and found that project quality is affected by specific traits of individuals along with project payment and the number of project requirements. His results also showed that high rated contestants sign up earlier to the contest to deter the entry of opponents and by this strategy, they can gain a surplus amount.

In microtasking crowdsourcing platforms (MTurk and CrowdFlower), earning money is the main incentive and research has shown that there are a number of crowdworkers who rely on MTurk (Ipeirotis 2010) for their income. To find out how workers search for HITs, Chilton et al.'s (2011) studies have shown that workers tend to sort HITs by newest HITs and most HITs' available options provided by MTurk, and focus mostly on the first two pages of results ignoring the position of HITs.

2.3.3 Creativity

To increase the creativity in innovative crowdsourcing tasks, (Dontcheva et al. 2011) have shown that the visual design of the task has a direct impact on creativity: positive background images lead to more significantly original ideas than having no image or a negative image. Based on studies on work environments, they have four recommendations. They suggest that building a community to increase collaboration between workers will tend to increase creativity. Also as Franklin et al. (2011) and Kittur et al. (2008) have shown, providing a good and proper interface for users has a direct effect on the results. (Kittur et al. 2008) suggest that, to have expert-level results from crowdsourcing systems, it is essential to have explicitly verifiable questions, which require more effort than random or malicious completion, along with multiple ways of detecting suspicious responses.

2.3.4 Quality Control

In crowdsourcing systems in which the main incentive is monetary reward (e.g. MTurk), it is possible that workers try to finish as many tasks as possible without

focusing on the quality of the results they provide and do not fully engage in tasks. In these situations, one solution is to collect responses from multiple workers and aggregate results. Another solution is to design a strategy or method to screen participants and to remove those who are gaming the system. (Downs et al. 2010) explored a screening task in which, to continue the tasks, participants were asked to answer some demographic questions (age, gender, current occupation) for demographic analysis, and two qualification questions about the task. They also explored the use of a time stamp to identify participants who were just clicking rather than answering conscientiously.

Historically, transcription has been an expensive and slow process done by experts which can be replaced by inexpensive and fast crowdsourcing methods. (Williams et al. 2011) proposed three techniques to improve the quality of the crowdsourced transcription method. First, they suggested collecting transcriptions one at a time until k matches are obtained, then treating automatic speech recognition (ASR) output as the first crowdworker and, finally, using regression to estimate the probability of the correctness of the crowdsourced transcription.

(Hirth et al. 2011) proposed majority decision (MD) and control group (CG) as two mechanisms for cheat detection in crowdsourcing platforms. Their studies on different types of crowdsourcing tasks suggest that, for routine and low-paid crowdsourcing tasks, the MD approach should be preferred and, for complex and more creative, the CG approach will provide better results.

In order to increase the quality of crowdsourcing tasks, one approach is to reduce the number of malicious workers by discouraging them from accepting the task. Experiments

by (Eickhoff & De Vries 2008) have shown that, despite the classical design of a practice which suggests reduced context change to keep users focused for efficient work, in crowdsourced tasks greater variability and context changes discourage malicious workers. They have also shown that the previous acceptance rate of workers is not a predictor of their reliability. In another study, majority voting was compared with the method in which votes are weighted by worker quality. Their results showed that removing spammers who are workers with poor precision increased the accuracy of relevant judgments (Vuurens et al. 2011).

As previously mentioned, earning money seems to be the primary incentive in MTurk; therefore, some may consider increasing the reward to get higher quality results (Kazai 2011). It has been shown that increasing the monetary reward will decrease the response time to HITs (Franklin et al. 2011) or increase the demand for the task (Faridani et al. 2011), but may not necessarily increase the quality of results in some applications (Franklin et al. 2011; Buhrmester et al. 2011; Mason & Watts 2009) and, in some others, it may decrease the demand for the task as high reward means more complex and more involved tasks (Faridani et al. 2011).

In crowdsourcing algorithms in which humans are asked to select the best item among others that match specific criteria or, in other words, to select the item that is believed to be the maximum, it is important that the algorithm can perform a desired balance between quality, cost and execution time and can handle user mistakes or variability. (Venetis et al. 2012) studied different strategies for tuning parameters of two parameterized max algorithms: Bubble Max and Tournament Max, with their goal being to find parameters

that optimize the performance of a given family of algorithms. Results showed that increasing the budget tended to lead to higher quality in all strategies but Tournament Max algorithms performed better than Bubble Max for the same budget. A study on two popular aggregation rules (plurality and majority) has shown that the plurality rule performs better in all cases (Venetis et al. 2012).

2.3.5 Latency in Crowdsourcing Systems

One major challenge in crowdsourcing systems is latency. The time interval between sending jobs to crowdsourcing platforms (e.g. MTurk) and receiving responses from workers can be broken down into two components. The first component, T1, is the time between sending the HITs to the MTurk crowdsourcing platform until some workers find your HITs, feel motivated to solve them and start solving the problems. When workers start completing the HIT, it takes T2 time (the second component) for them to complete the job and send the results. The sum of T1 and T2 time is referred to as the total execution time (TET) of the crowdsourcing job. The whole process of sending HITs and receiving responses from workers may take minutes to days depending on the HITs' design and the specified rewards. To make crowdsourcing applications near real time, (Bigham et al. 2010) designed a mechanism called "quikTurkit" which recruits workers and keeps them busy with other available HITs until the required HIT arrives. The workers accept the actual HIT as it arrives and send back the responses. quikTurkit also uses search engine optimization techniques. As an example, quikTurkit posts more HITs than what is actually required and sends the alternate HITs by different titles or rewards. Using these mechanisms, quikTurkit tries to keep the posted HITs on the first page of the

search results. By applying quikTurkit, they were able to receive their responses almost in real time and at a low cost (Bigham et al. 2010).

In related research which sought to reduce the latency of crowdsourcing systems, (Bernstein et al. 2012; Bernstein 2011) proposed two techniques to get responses in just two seconds. First, they defined a retainer model in which crowds are paid to wait until the actual task arrives. Unlike quikTurkit which keeps workers busy, users are free to do other HITs while waiting. When the actual task arrives, they are alerted and notified by sound. Rapid refinement is their second technique which seeks early agreement on multiple responses to decrease the overall amount of time needed to produce the desired result.

Even though previous research has attempted to design mechanisms to speed up recruitment and the HIT selection process of crowdsourcing tasks, the impact of the HIT UI design on the TET of crowdsourcing tasks has not received adequate research attention.

2.3.6 Can You Crowdsource Your Task?

In crowdsourcing a task, we are facing two challenging questions, "*What"* to crowdsource and "*How"* to crowdsource. We have to find out whether or not the task is really crowdsourceable which means can we get higher quality results by crowdsourcing a specific task or not? Some tasks can be completely done by crowdsourcing (e.g. image annotation) with a high quality result and low cost, and some tasks can tend to better results if we let the computer do the heavy lifting tasks and use the power of the crowd

where computers have less power (e.g. CrowdDB). If crowdsourcing is a good solution to our problem, the next step is to select a good category and design in order to have high performance. (Little et al. 2010a) studied two types of crowdsourcing tasks: parallel and iterative. In parallel tasks, all workers are working alone and are not aware of others but in iterative tasks each worker sees responses from previous workers. They discovered that in writing and brainstorming tasks, the iterative process increases quality but in brainstorming and transcription tasks, parallel crowdsourcing produces the best results. This shows that, depending on the type of task you want to crowdsource, you need to use the best type of process to achieve higher quality results.

2.4 Content Based Image Retrieval

In past decade there has been an enormous advance in digital imaging devices. In 2014 it is expected that around 63% of the world population will be using smartphones (EMarketer 2014). All smartphones have built-in cameras and billions of digital photographs are produced every second by these smartphone, satellite devices, surveillance cameras and personal digital cameras. This mass production of digital photographs resulted in creation of very large image datasets. This image datasets can't be efficiently used unless there is a good system to search within images and retrieve requested images.

Two major trends in image search are Text-Based Image Retrieval systems and Content-Based Image Retrieval systems. Text-Base Image Retrieval systems act in very similar way to text search systems; they search on image annotations and other text properties of images. In these systems, the whole image database needs to be annotated

prior to image retrieval. There are many techniques to provide annotations for images. LableMe (Russell et al. 2007) is one of the proposed systems that uses crowdsourcing to provide label for images. A major challenge in Text-Based Image Retrieval is that human provided annotations are subject to their perceptions of the image, there is a sematic gap between the provided annotations and actual image content.

Another method for image retrieval is Content-based Image Retrieval (CBIR) systems. CBIR systems which are also known as Query-By-Image-Content (QBIC) use computer vision algorithms to describe image contents. In CBIR systems, images are represented by features. These features are specific visual properties of images that describe the image. There are different types of features being studied and used by researchers in CBIR context (colour, shape, texture as global features and SIFT, SURF, ORB as local features). Many techniques are proposed to extract, classify and search within image features for object detection or image similarity search (Rubner et al. 2000; Jing & Baluja 2008; Tran n.d.; Grauman 2010). In our research we decided to use SIFT, SURF and ORB features and using a techniques similar to text search on image features to search for similar images. More details are provided in Chapter 4.

2.5 Summary

Crowdsourcing provides a new way of problem solving in which humans can assist computers. Humans can play a huge role in providing missing information, voting and comparisons. As examples, the combination of humans and computers can provide a more complete database system, help digitize texts or provide a more efficient image search system.

The main challenges that should be considered in designing a crowdsourcing (CS) system are: providing a good incentive to motivate people to participate in a CS system, controlling the quality of provided information, decreasing the cost if the CS system has a monetary reward as incentive, and decreasing the response time of the system.

Chapter 3

User Interface Design in MTurk

This chapter describes our experiments on different user interface (UI) designs of an image similarity ranking on Amazon Mechanical Turk (MTurk) as a crowdsourcing platform. In all crowdsourcing tasks, increasing the performance of workers and decreasing the execution time of a task is a goal. We investigated the effect of a well-designed user interface (UI) on the task performance and execution time. The goal of this research was to evaluate different UI designs for our crowdsourcing tasks and select the one which leads to high performance results in shortest time. We used the findings of this experiment in our main experiment which explained in Chapter 5.

3.1 Overview

As discussed earlier, Amazon Mechanical Turk (AMT or MTurk) is one of the platforms that implement microtask-based crowdsourcing. Using MTurk, requesters can contract and interact with an on-demand, global workforce through a web-based user interface. Monetary reward is the main incentive and workers try to earn as much money as they can in short periods of time. Job requesters have the option to accept and pay

workers for results or to reject workers' results without paying them. In the case of tasks which solicit people's opinions, it is not possible to check all responses from workers and reject all low performance results. This highlights the importance of having high quality responses from workers.

Another important factor for requesters in crowdsourcing tasks is time. To make crowdsourcing tasks closer to real time, there need to be mechanisms to help workers find the tasks easily, complete them and return the results as quickly as possible. If workers do not feel motivated to do the task due to the amount of reward, the task design, task completion time or crowdsourcing system, latency will be increased.

While much of the previous research on MTurk has tended to focus on factors that affect the motivation and creativity of workers and on cheating detection methods, there have not been many studies that deal with the impact of the visual design of the tasks' interface on workers' performance and the crowdsourcing system's latency. The usability of the software and the user interface (UI) that are part of the MTurk platform can potentially affect worker satisfaction levels and the costs incurred by the requesters. While many researchers have studied the usability of systems in software design (Juristo et al. 2007; Seuken et al. 2010; Liu & Ma 2010), and the effects of cognitive load and its integration with human–computer interaction (HCI) concepts on user interface (UI) design (Huang et al. 2009; Antle & Wise 2013), there are very few studies that have addressed the effects of user interface (UI) design on crowdsourcing using the MTurk platform. (Khanna et al. 2010) studied and designed a simplified UI with simplified instructions and localized language to reduce barriers on task execution. Their studied UI

design helped low-income workers in India to participated in crowdsourcing with MTurk and earn money.

It is our contention that the design of the interfaces through which the workers perform the human computation and related tasks has a significant effect on the performance and the time taken to complete the tasks. Drawing on cognitive load theory and usability design principles, we report on the design and preliminary results of two experiments that tested the effects of different user interface (UI) designs on performance and system latency in the context of crowdsourcing.

3.2 Research Questions and Hypotheses

As noted before, increasing performance and decreasing latency are two main goals in all crowdsourcing tasks. Requesters want their crowdsourcing tasks to be completed in minimum time with maximum quality results. The HITs' UI design is an important aspect of crowdsourcing tasks. We address the following research questions:

Do cognitive load theory (CLT) design principles help in designing improved interfaces for crowdsourcing tasks?

Does the design of UIs impact on workforce performance and productivity?

One of the CLT design suggestions is eliminating unnecessary distracting features in a UI. If there are too many unnecessary features in a UI, more of the working memory will be wasted dealing with these features. It has been studied by (Oviatt 2006) that if unnecessary features are eliminated, the user's cognitive load will be minimized and will result in a higher learning ratio in educational software. For the context of this research,

we will examine the same design principle and its effect on the performance of results produced by workers in our crowdsourcing task.

Another part of our study is to investigate the effect of HIT UI design on total execution time of the crowdsourcing task. We will try to answer whether the UI design has any effect on crowdsourcing system latency.

The specific hypotheses are stated below:

H1: Lowering extraneous cognitive load by eliminating unnecessary features from HIT UI design will result in higher quality responses from workers.

H2: In the same task with similar reward, the complexity of the HIT's UI has a negative effect on the total execution time (TET).

3.3 Research Methodology and Design

We describe two experiments that were performed to test the hypotheses. In Experiment1, we tested H1 and studied the impact of different UI designs on workers' performance which directly affects the cost of the crowdsourcing task. The task we chose is an image ranking task. It involves ranking 10 images based on their similarity to a given query image. The task involves visual information processing for which the quality of the user interface (UI) is particularly critical. For this experiment, we designed three different UIs based on ranking, direct sorting (drag and drop) and rating.

The first UI design is called Rank UI design. In this type of design, workers are asked to compare 10 images with the query image and rank them based on their similarity to the

query image. Workers are asked to assign a number between 1 and 10 to each image indicating the position of the image in a ranked list.

The second UI design is called Sort UI. In this type of UI, users have to click and move each image to visually create a ranked list of images. In this UI design, moving each image causes the whole list to be moved.

In the third UI design, which is called Rate UI, workers are asked to give a score between 1 and 5 based on the similarity of each image to the given query image. If the image is very similar to the query image, they can assign the image score 5, and if it is not similar, they can assign it score 1.

We ran Experiment1 for six image categories from the Corel-Princeton Image Similarity Benchmark (section 3.3.1) (airplane, car, flower, fruit, horse and model) and for each category we created 50 HITs for each of the three methods (Rank, Sort and Rate). All three sets of HITs sent to MTurk in a 5minutes period of time with Rank-Rate-Sort order. In total, 350 HITs were created for each UI design for $17.50. The total number of HITs created for this experiment was 1050 HITs at a cost of $52.50.

To test the second hypothesis, we designed our second experiment (Experiment2). In this, the crowdsourcing task is to define a category for a number of images. We designed two UI designs for HITs, Type1 and Type2. In this experiment, we study the effect of HIT UI design on the TET of the crowdsourcing task and system latency.

3.3.1 Datasets

Experiment1 involves assessing the performance of the workers in the ranking tasks for which a gold-standard is needed. In this research we studied image similarity search and for this purpose we decided to use the Corel-Princeton Image Similarity Benchmark Dataset [4] for this reason. In this dataset, for each query image, similar images and their (gold standard) similarity score are provided. For our experiment, we selected six query images and randomly selected 10 similar images based on each query image. Aggregated rankings provided by workers were compared against the gold-standard ranking.

The task for the second experiment deals with image categorization for which we used a categorized image dataset. Caltech-256[5] data set was selected for this experiment. This dataset consists of more than 30,000 images categorized into 256 folders. Each folder has a category name. We ran Experiment2 twice for each UI design. On the first run, 12 images were selected from five different categories. We created 20 HITs for each UI. For the second run, we selected with 21 images from eight different categories and 20 HITs were created for each UI.

3.4 Experiment1: Image Ranking

For Experiment1, we created and posted several HITs using the three UIs (Rank, Sort and Rate) that we designed. The HIT structure for this experiment was:

- $0.05 reward for all three types of HITs

[4] http://www.cs.princeton.edu/cass/benchmark/
[5] http://authors.library.caltech.edu/7694/

- instructions for users to do the task
- added time stamps to the design of the HIT to detect workers who just clicked and did not do the task carefully
- added text box to collect user comments.

Figure 3–1 describes the system that was designed to create the HITs, collect and store the results obtained from the workers. MTurk makes it possible for workers to view the HIT in preview mode before accepting it. However, in our experiment, we only showed a simple preview description and not the full HIT. When workers accepted a HIT, the corresponding page was created on a remote host and shown to workers.

Users sent their results back to MTurk using the "Submit" button that we provided on each page and then we collected results using our program and prepared them for analysis as shown in Figure 3-1.

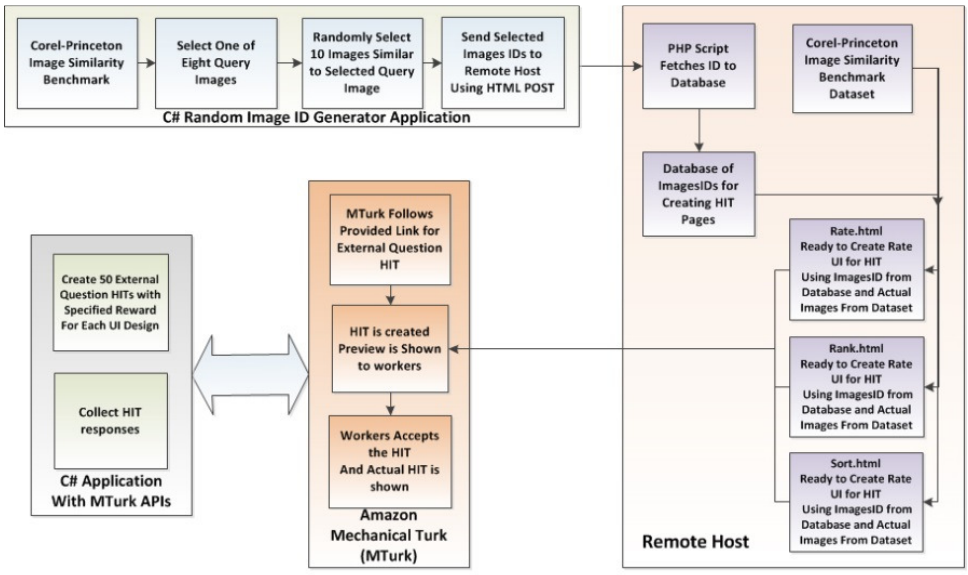


Figure 3-1. Experiment1

3.4.1 Rank UI design

For the Rank user interface design, we provided workers with 10 images and asked them to assign a number between 1 and 10 to each image according to their similarity to the given query image. Value 10 means that the given image is the most similar image to the query and value 1 means that the image is the least similar image to the query image. Workers had to select a number for each image and they could not use each value more than once. In this task design, users had to compare 10 images with the query image and rank each image not only based on its similarity to the query image but also based on the degree of similarity to other images of the query image.

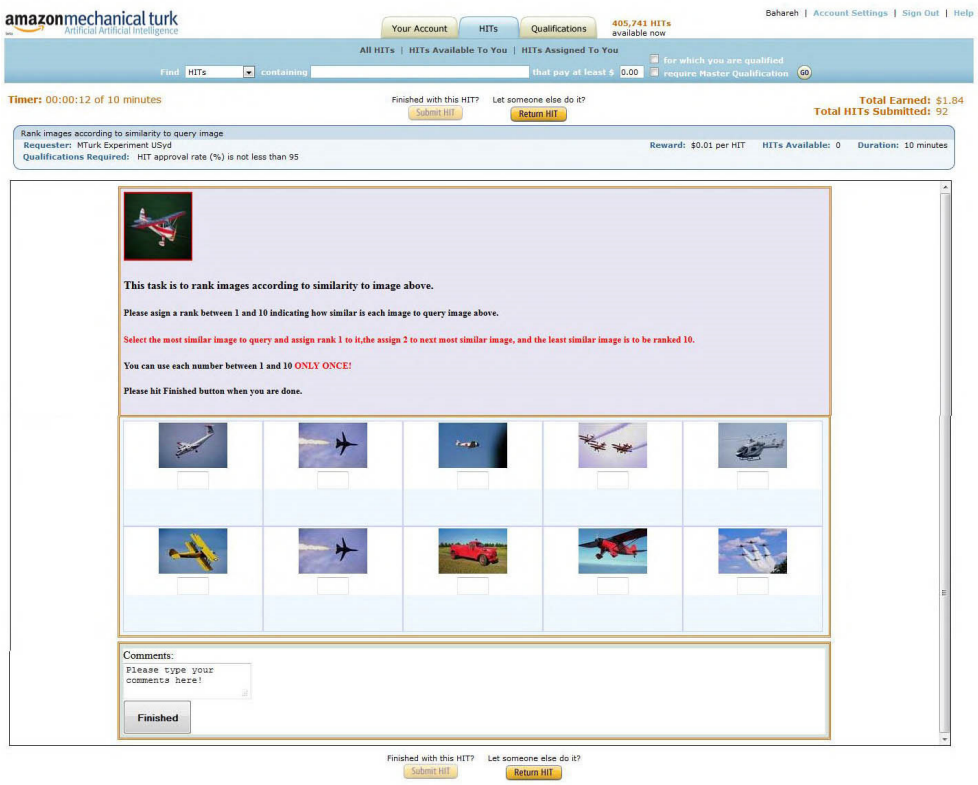


Figure 3-2. Rank UI design

3.4.2 Sort UI design

In the Sort user interface design for ranking images, we used JQuery UI functions to create a draggable list of images and asked workers to sort images by their similarity to the given query image using the drag-n-drop functionality of the HTML page.

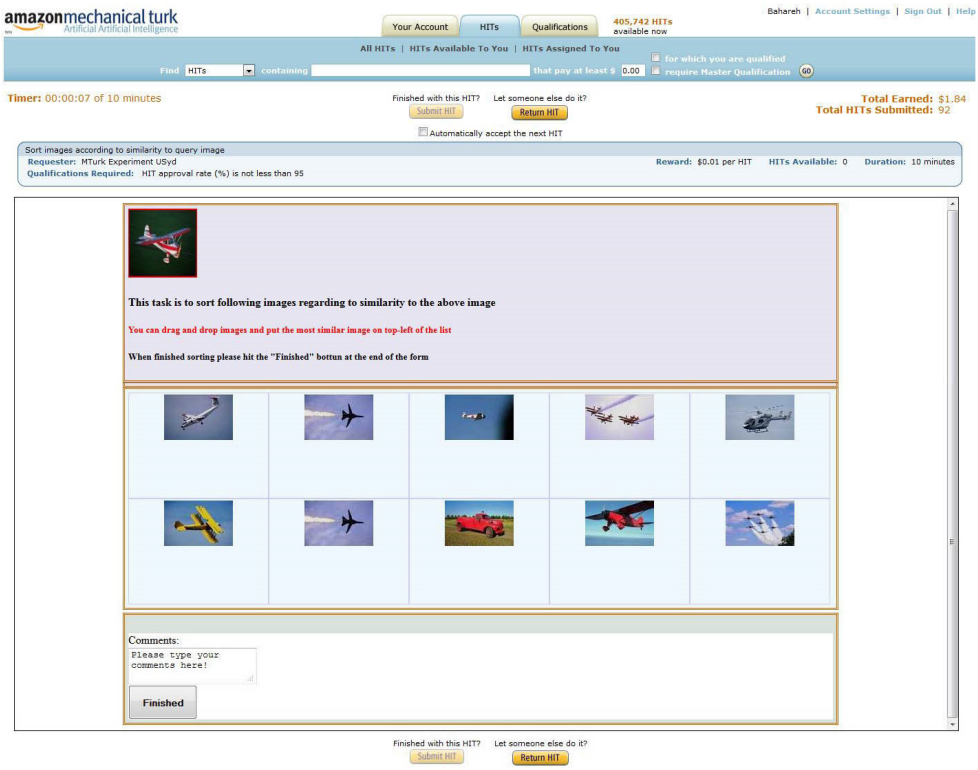


Figure 3-3. Sort UI design

3.4.3 Rate UI design

In the Rate user interface design, once again we provided 10 randomly selected images from the Corel-Princeton dataset. In this task, we asked workers to rate the similarity of each image to the given query image. They were asked to assign a number between 1 and 5 according to the similarity of each image to the given query image, 5 for high similarity and 1 for low similarity. In this task, workers had to provide a rate for all images and they were allowed to use each number more than once.

Unlike the Rank method in which workers had to compare all images to provide a rank for them, in this task, they were able to focus on each image and rate its similarity to the query image.

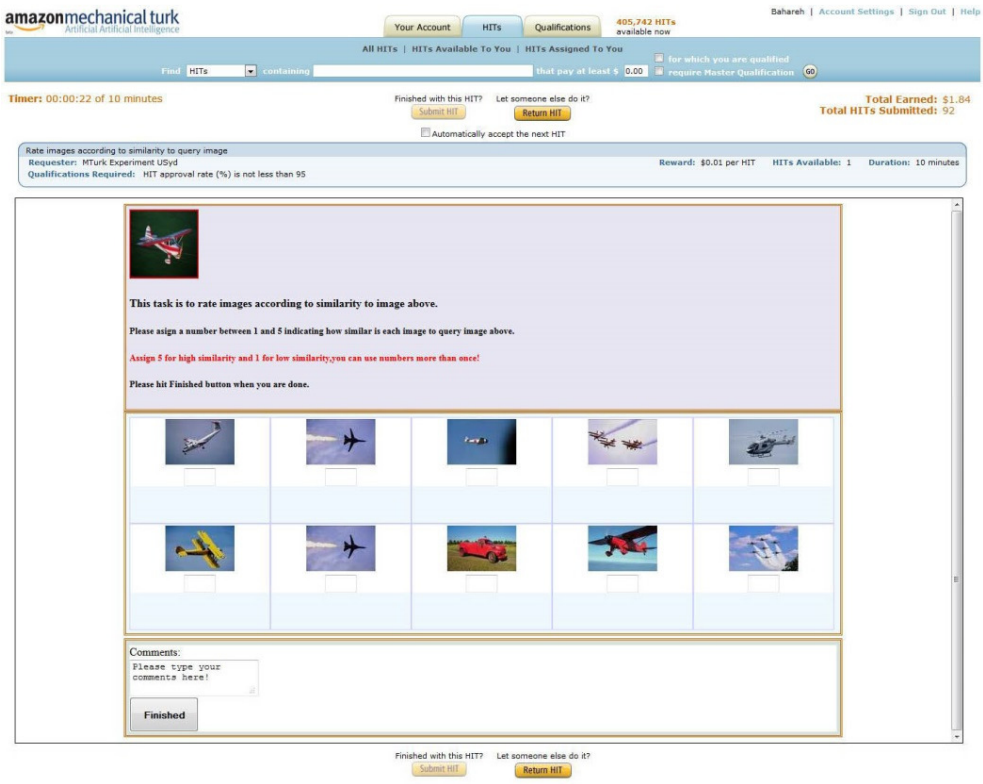


Figure 3-4. Rate UI design

3.5 Aggregating Crowdsourced Responses

The power of crowdsourcing systems is based on the collective intelligence of the involved crowd. The main concept of microtasking crowdsourcing is collecting several crowdworkers' opinions about a specific task and then, by aggregating the responses, we can achieve a result with a high level of accuracy (Urbano et al. 2010; Wang et al. 2012;

Yan et al. 2010; Rashtchian et al. 2010). This highlights the importance of aggregating the crowdsourced responses. The aggregation method used in each crowdsourcing task varies depending on the type of crowdsourced responses.

In our experiment, we collected two types of responses from workers. For Rank and Sort UI designs, crowdworkers' responses were ranked lists and for the Rate UI design, crowdworkers provided ratings. We used different techniques to aggregate each type of response.

3.5.1 Rank Aggregation

The problem of combining ranking results from various sources arises in many areas. One of the best examples is building meta-search engines for the Web and aggregating viewers' ranking for a specific product (e.g. movies, books). This problem is defined as finding a ranking for a group of input rankings that best represents that group of inputs: this problem has also been a point of interest in the computer science community (Liu et al. 2007; Dwork et al. 2001).

3.5.1.1 Type of Ranked Lists

We can define ranking or ordered lists as:

With respect to universe U, c is an ordering subset $S \subseteq U$ i.e $\tau = [x_1 \geq x_2 \geq \cdots \geq x_d]$ and each $x_i \in U$ and $\geq$ is an ordering relation. Based on this definition, there are three types of ranked lists:

full lists: τ contains all elements in U. Our case is an example of *full lists*, in which crowdworkers are required to provide a ranking for all the given images.

partial lists: In some situations, it is not possible to provide full lists. For instance, results of different search engines on a specific query might not contain similar elements. In other words $|\tau| < |U|$ which means the list τ ranks only some of the elements of U.

top k lists: Top k lists are a special type of partial list in which τ ranks only a subset of S. For instance, if S is a set of all the pages indexed by a search engine and τ represents only the top 100 results, the pages that are not present in τ can be assumed to be ranked below 100. In this condition, we call τ a *top k list* and k is the size of the list.

3.5.1.2 Distance Measure

There are several distance metrics used in the information retrieval literature ranging from the classic Kendall tau and Spearman's footrule to new ones such as generalized distance (Kumar & Vassilvitskii 2010), Expected Reciprocal Rank (ERR) (Chapelle et al. 2009) and (Carterette 2009). The most popular distance measures for computing the distance between two full lists are Spearman's footrule distance and Kendall tau distance (Diaconis 1988) which are explained below.

***Kendall tau distance*:** This distance measure, introduced by Maurice Kendall (Kendall 1938), is a metric that counts the numbers of pairwise disagreements between two lists: the larger the distance, the more dissimilar are the two lists. *Kendall tau distance* is also

called “Bubble Sort” distance as it is equivalent to the number of swaps that the bubble sort algorithm makes to place one list in the same order as the other list. *Kendall tau distance* for two full lists $\sigma$and τ is:

$$K(\sigma,\tau) = |\{(i,j)|i<j\,,\sigma(i)<\sigma(j), but\ \tau(i)>\tau(j)\}|$$

Dividing $K(\sigma,\tau)$ by $n(n-1)/2$ (n is the size of the list) results in a normalized version of *Kendall tau distance*.

***Spearman's footrule distance*:** For all $i \in S$, *Spearman's footrule distance* is the sum of absolute difference between the rank of i in each list:

$$F(\sigma,\tau) = \sum_{i=1}^{|S|} |\sigma(i)-\tau(i)|$$

$F(\sigma,\tau)$ can be normalized if divided by $|S|^2/2$. The footrule distance between two lists can be computed in linear time.

(Diaconis & Graham 1977) showed that the relation between Spearman's footrule and the Kendall tau is:

$$K(\sigma,\tau) \le F(\sigma,\tau) \le 2K(\sigma,\tau)$$

3.5.2 Rank Aggregation Methods

With the distance metrics, finding an aggregated ranked list for a group of ranked lists is the problem of finding the ranked list that has the shortest distance with all of the group members. Rank aggregation methods have been studied in the context of several research studies (Schalekamp & van Zuylen 1998; Dwork et al. 2001; Fagin et al. 2003; Fagin et

al. 2004). The methods can be positional algorithms (i.e. Borda rule, footrule), comparison sort algorithms (QuickSort, MergeSort) or a combination of positional and comparison algorithms (Copeland (Copeland 1951), MC4 (Dwork et al. 2001)). In our study, we used Dwork et al.'s (2001) scaled footrule aggregation (SFO) method.

3.5.2.1 Footrule and Scaled Footrule

Spearman's footrule distance between two ranked lists σ, τ is defined as:

$$F(\sigma, \tau) = \sum_{i=1}^{|S|} |\sigma(i) - \tau(i)|$$

(Dwork et al. 2001) proposed that: "For full lists $\sigma_1, \sigma_2, \dots, \sigma_k$ if the median positions of candidates in the lists form a permutation, then this permutation is a footrule optimal aggregation and for full lists can be computed in polynomial time".

For S, the union of ranked lists with n elements, the weighted complete bipartite graph (C, P, W) can be defined as:

$C = \{1, \dots, n\}$ is the set of elements to be ranked (in our research, the images showed to crowdworkers)

$P = \{1, \dots, n\}$ is n available positions.

The weight $W(c, p)$ is the total footrule distance (from σ_i) that places element c at position p.

$$W(c, p) = \sum_{i=1}^{k} |\sigma_i(c) - p|$$

$W(c,p)$ is the scaled footrule distance if:

$$W(c,p) = \sum_{i=1}^{k} |\sigma_i(c)/\sigma_i - p/n|$$

Scaled footrule aggregation (SFO) is obtained by solving the minimum cost maximum matching problem on (C, P, W).

In our research, we developed our MATLAB® code based on the proposed method to aggregate rankings for responses collected from crowdworkers. We also used Markus Buehren's[6] algorithm for optimal assignment to solve the minimum cost maximum matching problem.

3.5.3 Rate Aggregation

The problem of aggregating ratings has been studied in measuring the quality of a product based on ratings from several authors and review websites. In this field, researchers try to address issues with regard to different rating scales (1-5 stars, 0-10 stars, etc.) and propose efficient rating methods (McGlohon et al. 2010).

Calculating the average rating for each product is one of the methods for aggregating reviews. Despite Hu et al.'s (2006) study that showed that the average rating is not always the best way of measuring the quality of a product, it is used widely for products on the Web. In our study, we used the average rating to aggregate ratings provided by crowdworkers through our Rate UI design. Once the average rating was calculated, we

[6] http://www.mathworks.com.au/matlabcentral/fileexchange/authors/26973

sorted the rating and created a new ranked list for image similarities. We called this new rating "crowd-provided rank" and compared it with gold-standard ranking.

3.6 Measuring the System Performance

Our system's performance was calculated by comparing the ranking provided by aggregating crowdworkers' ranking/rating with the gold-standard ranking of the Corel-Princeton dataset. Rank correlation is a statistic that can measure the relationship between rankings: the rank correlation coefficient measures the degree of similarity between two rankings and can be used to define the significance of the relation between them. The Spearman ρ and Kendall τ are two popular rank correlation coefficients.

Our preliminary studies on using Spearman ρ or Kendall τ for performance measurement showed that they provide the same results but using Spearman ρ produced were more highlighted results. In our research, we used the Spearman ρ for the rank correlation coefficient between the aggregated ranking by crowdworkers for each UI design and the gold-standard data. While ρ is a nonparametric value, we also calculated Spearman's footrule distance to compare rankings.

3.6.1 Spearman ρ

The Spearman rank-order correlation is a nonparametric version of the Pearson product–moment correlation, and the Spearman correlation coefficient ρ measures the level of dependence between two variables. The Spearman ρ can only be calculated for ranked lists and if the data are not ranked, we have to first rank the data and then calculate

the Spearman ρ. In some cases, there are two identical values in the list (ties). In such situations, we have to take the average of the ranks that they would have if they were not identical.

The Spearman ρ for ranks without ties is calculated:

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

In this formula, d_i is the difference in paired ranked lists and n is the size of the list.

For lists with ties, the Spearman ρ is:

$$\rho = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}}$$

3.7 Analysis and Results

Since workers' responses for Rank and Sort methods were ranked lists, we aggregated them using the scaled footrule aggregation (SFO) method (Dwork et al. 2001). For the Rate method, we aggregated rates for each image by computing the weighted average on rates given by workers, and then sorted the list according to this new calculated rate and created a ranked list. To see how close the aggregated ranked lists provided by these three methods were to the gold-standard rank, we calculated the distance between aggregated results and the gold-standard ranking using the Spearman ρ correlation metric.

The ranking UI design with the higher rank correlation coefficient with the gold-standard ranking had better performance. Results showed that the Spearman rank

correlation coefficient (rho) of the results produced using the Rate UI design was higher than the other two methods. This implies that the ranked list produced by users from the Rate user interface was more similar to the gold-standard ranked list created by professionals and therefore that using the Rank UI design leads to relatively higher performance results (Table 3-1).

Table 3-1. Spearman ρ for different UI designs

Distance between gold-standard ranking and Experiment1 results							
		Airplane Dataset	Car Dataset	Flower Dataset	Fruit Dataset	Horse Dataset	Model Dataset
Rank UI	Spearman ρ rank correlation	0.54	0.51	0.66	0.58	0.8	0.12
Sort UI	Spearman ρ rank correlation	0.79	0.84	0.86	0.59	0.77	0.23
Rate UI	Spearman ρ rank correlation	0.80	0.88	0.91	0.73	0.91	0.32

3.8 Experiment2: Image Categorization

The second experiment's goal is to study the impact of UI design on the TET with MTurk. We designed two different UIs for defining categories of images. Similar to Experiment1, HITs were sent to MTurk with $0.05 rewards. Caltech-256 dataset was used for selecting categories and corresponding images. The system we used to create HITs and collect and store results is described in Figure 3-5.

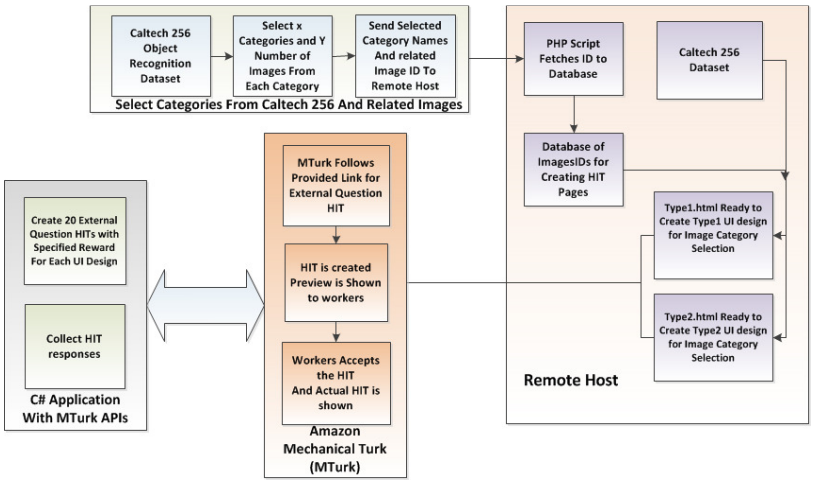


Figure 3-5. Experiment2

3.8.1 Type1 UI design

In the Type1 UI design, we put radio buttons for categories under each image and asked workers to select one category for each image. (Figure 3-6)

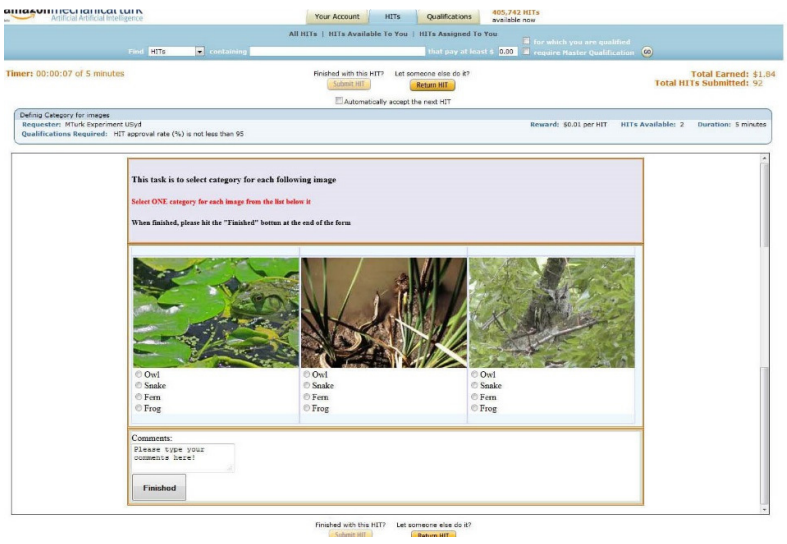


Figure 3-6. Type1 UI Design

3.8.2 Type2 UI Design

Studying a more complex user interface (UI) is the goal of the Type2 UI. For this purpose, we first put all images at the top of the HIT web page and at the bottom of the page, we asked users to select image IDs which belonged to a specific category. Due to limited space on MTurk's main HIT page, users had to scroll up and down to select image IDs for each category. (Figure 3-7)

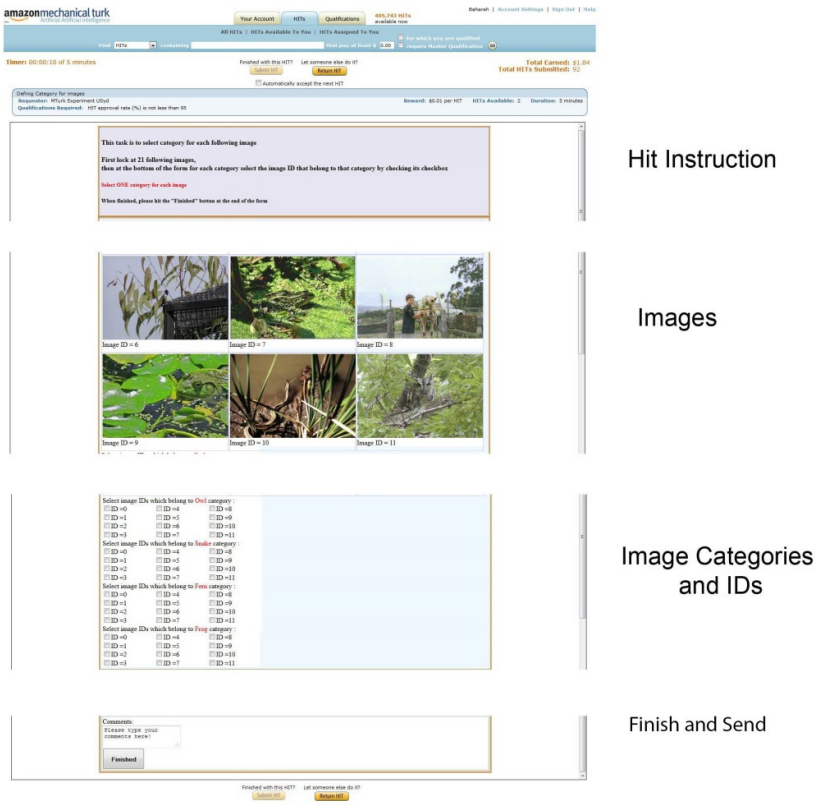


Figure 3-7. Type2 UI Design

3.8.3 Analysis and Results

For the first run we selected 12 images from 5 different categories and 20 HITs. For the Type1 UI design, it took around seven hours to have 20 completed HITs and only three workers rejected the HIT, but for the Type2 UI design, it took more than 16 hours to have 20 completed HITs and 12 workers rejected the task (Table 3-2).

Table 3-2. Experiment2 First Run

Experiment2 First Run Results; 12 Images From 5 Categories		
	Type1 UI	Type2 UI
Total HITS	20	20
Total Cost	$1	$1
Average Task Time (s)	76	139
TET	7 Hours, 20 Mins	16 Hours, 40 Mins
Number of Rejected HITs	3	12

Users provided nearly 100% correct answers in both UI types. However, the average completion time for Type1 is less than the time for Type2. In Type 2, more workers rejected HITs, meaning that workers were not motivated in doing the HIT on Type2. This increased the TET of the task and higher TET results in increased latency of the crowdsourcing task.

For the second run, we selected 21 images from eight different categories and 20 HITs were created for each UI. This time, it took around 24 hours to have 20 completed HITs for Type1, but for Type2 after 38 hours we received only 11 completed HITs. Hence, had to terminate the task. Checking the number of workers who did not complete the HIT

showed that in Type1, seven workers accepted the task but did not complete it and returned the HIT; and for Type2, 62 workers rejected the task (Table 3-3).

Table 3-3. Experiment2 Second Run

Experiment2 Second Run Results; 21 Images From 8 Categories		
	Type1 UI	Type2 UI
Total Completed HITS	20	11
Total Cost	$1	$1
Average Task Time (s)	137	244
TET	23 Hours, 50 Mins	38 Hours
Number of Rejected HITs	7	62

These results show the importance of designing a task UI which creates more interest among the workers. If workers are not interested in the HIT UI design, they will reject the HIT. As a result, the latency of crowdsourcing will increase.

3.9 Discussion

Results of our Experiment1 highlighted the importance of lowering the extraneous cognitive load of UI design and its effect on the performance of results produced by workers. While all three parts of the experiment cost the same, using the interface which results in higher performance responses from workers will make the crowdsourcing task more affordable.

Taking a closer look at these three UI designs, we can say that in the Rank UI design, users have to compare the whole 10 images with each other and the query image to find a rank for each image. While the number of images to compare is more than Miller's magic

number 7±2 (Miller 1956), we believe it imposes a higher cognitive load on the task resulting in lower performance and poorer results.

In the Sort UI design, users have to move images to produce a ranked list and moving one single image makes the whole list move. These movements of the images on the page distract the user from the original task and place more cognitive load on the task. The number of user clicks is also higher in this UI design which is not recommended by CLT. Unnecessary distracting features and a high number of clicks place more cognitive load on the task and tend to lead to poor results from users.

We agree that the reason that workers perform better with the Rate UI design is that they can focus on each image by itself and assign a more accurate similarity score. This reduces the number of comparisons from 10 to two resulting in a lower cognitive load. These results suggest that if the task has a higher intrinsic cognitive load, poorly designed UI design with a high extraneous cognitive load can have a negative effect on workers' performance.

In Experiment2, we studied the impact of UI design on the TET of a crowdsourcing task. In this experiment, increased cognitive load and higher complexity of the UI design did not affect workers' performance but it contributed to reduced levels of willingness to accept and finish the task. If workers do not want to accept and finish the task, requesters will not receive their desired number of responses. This means higher system latency and also contributes to the increased probability of incomplete crowdsourcing tasks.

Results of our experiments highlight the demand for more research on UI design of MTurk HITs from the aspects of cognitive load and usability. We examined the impact of

UI design on two visual crowdsourcing tasks. The cognitive load aspect of HIT design in textual tasks can also be studied. In our future work, we will use the findings of this study to design crowdsourcing tasks.

3.10 Conclusion

In light of the limitations of the generic user interface (UI) in MTurk, it is important to design HIT UIs that reduce poor quality results and increase worker productivity. This has the potential to reduce the execution time of the crowdsourcing task. In this thesis, we studied the impacts of user interface (UI) design of HITs in the MTurk crowdsourcing platform on workers' performance and total execution time (TET). Our experiments show that designing a HIT UI with the goal of reducing the cognitive load will help workers focus on the task and achieve better performance. In some crowdsourcing tasks (like our image ranking task), it is not possible to differentiate false results and reject workers' responses, so requesters have to pay all workers. We showed that, in such tasks, it is possible to have higher quality results by eliminating the factors that lead to workers' poor performance, with the same cost. This means that we have a more cost-effective crowdsourcing task.

We also investigated the effect of UI design on the demand for the task. Our results showed that MTurk workers prefer to accept tasks with less complex UI designs. If the user interface (UI) is perceived to be complex from the workers' point of view, they are less likely to accept the crowdsourcing tasks. As a result, it takes more time to complete the task and the crowdsourcing system's latency will increase. To have a crowdsourcing

system closer to real time, we suggest spending more time on designing the HIT UI to create a UI with less complexity.

Results of our experiments showed that by spending more time on HIT UI design, requesters can achieve high quality results in a shorter time. The results can help to develop guidelines for making crowdsourcing tasks more efficient with less latency. We used the findings of this research in our next experiment in Chapter 5.

Chapter 4

Content-Based Image Retrieval System

In this chapter, we provide the architectural designs for content-based image retrieval (CBIR) system using SIFT, SURF, SURF 128 and ORB features. We used this system to search for similar images based on a given query image. The search results will be crowdsourced. The crowdsourcing part of our system which used to study the effect on the performance of the hybrid Human--MachineCBIR system is explained in Chapter 5.

4.1 Overview

Advances in image acquisition techniques have resulted in the creation of large image databases. In this scenario, it is necessary to develop a system to manage these databases, and the need to provide a high performance image search system is highlighted. Content-based image retrieval (CBIR) is the use of computer vision applications to search for images in these large databases and, through using these systems, the contents of the images are analysed and indexed. This content can be global such as colour, shape or texture or it can be information about specified local areas of the image.

SIFT, SURF and ORB are three types of local feature detector/descriptors that are widely used in CBIR systems. SIFT claims to have a very high performance but while SIFT features are 128-dimension vectors, it has a very high computation cost. SURF and ORB features are 64-dimension vectors and have a lower level of computation cost and they claim to have the same performance as SIFT.

As a part of our research, we designed four CBIR systems using each of the SIFT, SURF, SURF 128 (128-dimension version of SURF) and ORB feature extractors. We compared the performance of the system using these feature extractors and sent the results of this system to the second part of our research, which is a crowdsourcing system.

In this chapter, the CBIR system design, experiments and results are explained.

4.2 CBIR Architecture and Implementation

Our computer image similarity search system has two subsystems. The first subsystem is a MATLAB (FeatureExtractor) code that we used to extract image features and create a feature database. This part was done once only for each feature type (SIFT, SURF, SURF 128 and ORB) and the created database was used in the C# application (CBIR) to take the query image and search for similarities. In the next sections, these subsystems are explained in detail.

4.2.1 Feature Extraction

Feature extraction is the first step in CBIR systems. In this step, the visual contents of all images in the database are detected, extracted and described by multidimensional

feature vectors and can be global or local. A global descriptor uses visual features of the whole image, whereas a local descriptor divides the image into parts or regions and describes the visual features of the regions of the image.

Colour, texture and shape are the three most widely used features. Colour descriptors are three-dimensional (3-D) values and are proven to be a very discriminating feature for object recognition. Texture features are not as well-defined as colour features and describe the direction and granularity of the structuring elements of a region. Texture features can describe the content of many real-world images such as fruit skin, clouds, trees and fabrics. While colour and shape features can be used in image retrieval of any type of image, shape features are mainly used for domain-specific images such as human-made objects (Rui et al. 1999; Long et al. 2003).

4.2.1.1 SIFT

In 1999, Lowe proposed a new local image feature detector/descriptor method called Scale-Invariant Image Transform (SIFT) (Lowe 1999). SIFT transforms the image to a large collection of 128-dimension feature vectors which are invariant to image translation, scaling and rotation, and partially invariant to illumination changes and affine or 3-D projection. SIFT computes a histogram of local oriented gradients around the interest point and stores the bins in a 128-dimension vector (eight orientation bins for each of the 4*4 location bins) Figure 4-1 .

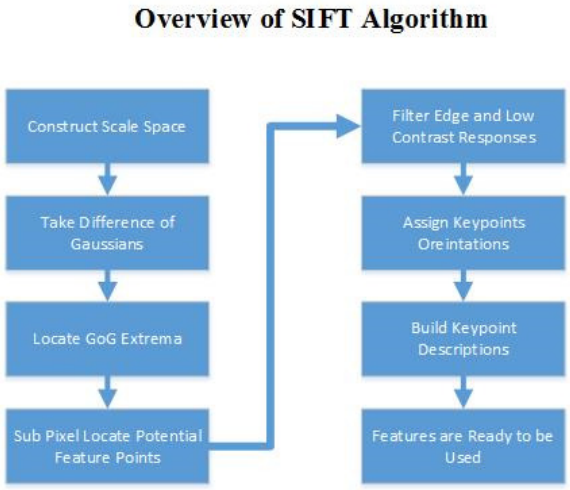


Figure 4-1. SIFT feature detector/descriptor

In our research we used Vedaldi[7] technical implementation of SIFT. They define "SIFT descriptor as a 3-D spatial histogram of the image gradients that characterize the appearance of a keypoint. The gradient at each pixel is regarded as a sample of a 3-D elementary feature vector, formed by the pixel location and the gradient orientation. Samples are weighted by the gradient norm and accumulated in a 3-D histogram *h* which (until normalization and clamping) forms the SIFT descriptor of the region. An additional Gaussian weighting function is applied to give less importance to gradients further away from the keypoint centre. Orientations are quantized into eight bins and the spatial coordinates into four each."

The 3-D histogram (consisting of 8×4×4=128 bins) is stacked as a single 128-dimensional vector, where the fastest varying dimension is the orientation and the slowest is the *y* spatial coordinate (Figure 4-2).

[7] http://www.vlfeat.org/api/sift.html

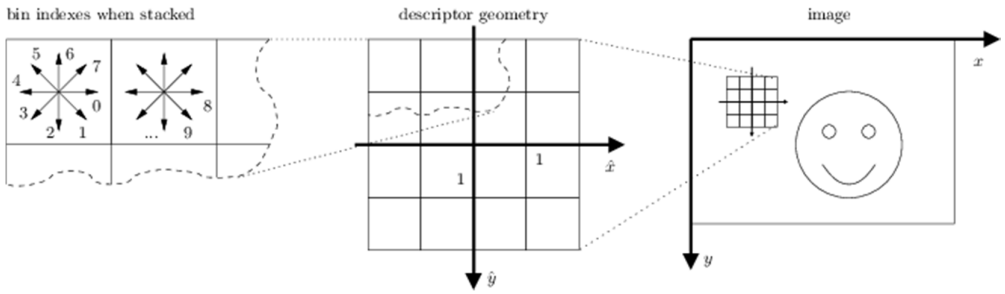


Figure 4-2. SIFT descriptor

4.2.1.2 SURF

Due to the large vector size of SIFT features, (Ke & Sukthankar 2004) tried to apply PCA on the gradient image and reduce the vector size to 36. The proposed PCA-SIFT is fast for matching but less distinctive. Another variant of SIFT is called GLOH (Mikolajczyk & Schmid 2005) which is proven to be more distinctive, but as it has the same number of dimensions as SIFT, it is also computationally very expensive.

Another feature detector/descriptor is SURF (Speeded-Up Robust Features) (Bay et al. 2006) and SURF features are 64-dimension or 128-dimension vectors. Similar to SIFT, SURF is also invariant to scale and rotation and is claimed to be distinctive and robust and can be computed much faster than other methods.

4.2.1.3 ORB

Building on top of the FAST keypoint detector (Rosten & Drummond 2006) and BRIEF descriptor, (Rublee et al. 2011) proposed new feature detector/descriptor called ORB (Oriented FAST and Rotated BRIEF). Based on the characteristics of FAST and

BRIEF, the ORB descriptor has good performance and low cost and it outperforms SIFT and SURF in speed.

In our research, we used SIFT, SURF and ORB features and we compared the performance of our image similarity search system using each feature detector/descriptor method.

4.2.1.4 Feature Extractor Application Implementation

Based on the properties of the feature detector/descriptor algorithm, each feature extractor function has some parameters which make it possible to extract a variable number of features from images. These functions are able to extract as low as five and as many as 7000 features from each image. An increased number of extracted features from each image will result in a larger size dataset and may improve the performance of the image search system. However, at the same time, it will increase the computation cost of the system and make the whole process very slow. We designed an object detection system to study the effect of an increased number of features on the system performance and to decide on an optimal number of extracted features. We called this program "FeatureCount" and conducted experiments on the number of features to extract from images using this program. Based on the results of our experiments, we decided to use default parameter settings of all functions.

To create an indexed database of features, we developed a MATLAB program called "FeatureExtractor". This program has four main functions to extract SIFT, SURF, SURF128 and ORB features. SIFT features are extracted using the VLFeat library

(Vedaldi & Fulkerson 2010): SURF and ORB features are extracted using MATLAB's built-in functions. In all four functions, extracted features are stored in the memory for further computations.

4.2.2 Indexing Features

After feature extraction, the next step in CBIR systems is measuring the similarity of images using the database of extracted features. Several methods have been proposed (Rubner et al. 2000; Grauman & Darrell 2005; Grauman 2007; Grauman 2010) but (Sivic & Zisserman 2003) is one of the simplest and most efficient methods. They proposed a text retrieval approach in which a vocabulary tree of features is constructed using *k*-means clustering. Visual features are then indexed after calculating tf-idf (term frequency-inverted document frequency) and, at the retrieval stage, images are ranked based on their tf-idf score. According to definitions[8] the "tf-idf is a weight often used in information retrieval and text mining. Tf-idf weight can be used as a statistical measure for evaluating the importance of a word to a document in a collection or corpus. The weight increases by increased number of times a word repeats in the document but is offset by the frequency of the word in the corpus."

The tf-idf weight is a product of term frequency (TF) and inverse document frequency (IDF). TF computes the normalized term frequency (the number of times a word appears in a document) divided by the total number of words in that document and IDF computed as the logarithm of the number of the documents in the corpus divided by the number of documents where the specific term appears.

[8] www.tf.idf.com

TF: Term frequency, measures the frequency of a term in a document. Due to different size of document, it is possible that a term appears many more times in long documents than in shorter ones. For normalization purpose, the term frequency is often divided by the document length (the total number of terms in the document) as a way of normalization:

$$tf(t) = (\frac{Number\ of\ times\ term\ t\ appears\ in\ a\ document}{Total\ number\ of\ terms\ in\ the\ document})$$

IDF: Calculates the importance of a term. It is known that certain terms with little importance, such as "is", "of" and "that", may appear many. In order to weight down the frequent terms while scaling up the rare ones, we compute the following:

$$idf(t) = \log_e(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ with\ the\ term\ t\ in\ it})$$

The tf-idf weight is the product of TF and IDF:

$$tf - idf = tf \times idf$$

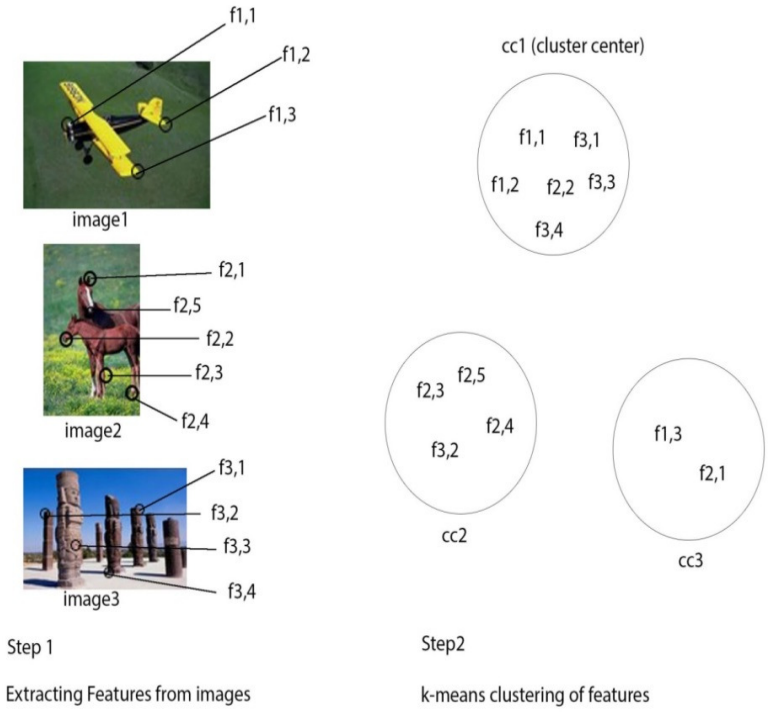
We use the same concept to create our feature dataset. In image similarity, terms are cluster centres, documents are images and word corpus is the whole image set. Using such assumptions, the *tf* for each cluster centre is computed as:

$$tf(c) = (\frac{Number\ of\ times\ cluster\ c\ appears\ in\ image\ i}{Total\ number\ of\ clusters\ image\ i\ has\ features\ in})$$

The *idf* can be computed as:

$$idf(c) = \log_e\left(\frac{Total\ Number\ of\ images}{Number\ of\ images\ with\ features\ in\ cluster\ c}\right)$$

Similar to text td-idf, if a cluster is repeated in all images, it has the lowest importance. This is explained in Figure 4-3.



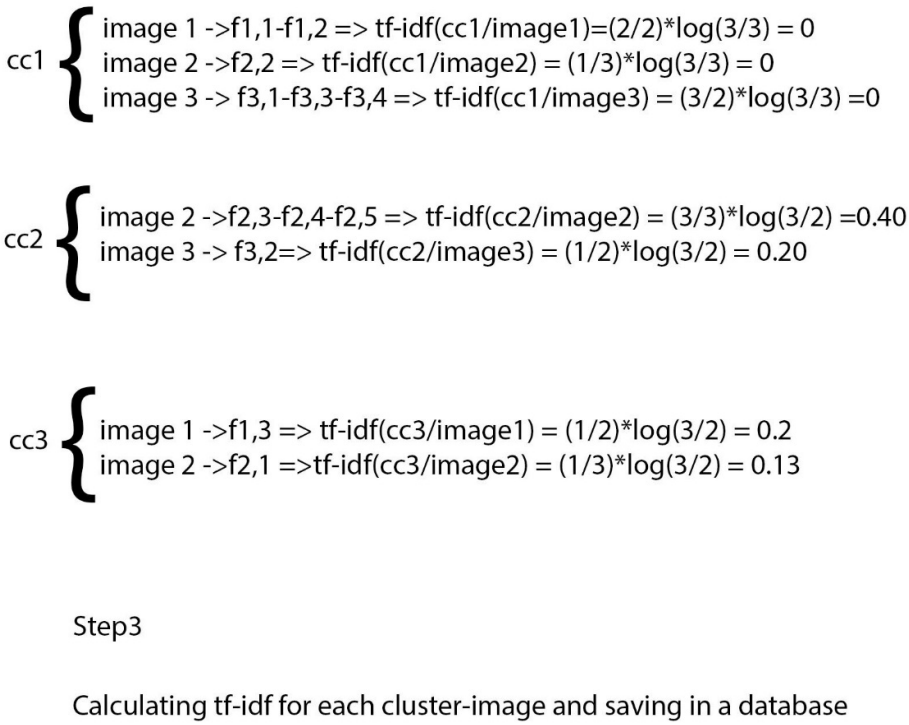


Figure 4-3. tf-idf for clusters

To make the whole search process faster, we developed another MATLAB application, calculated tf-idf for all cluster images and saved them in a MS SQL Server database. In our search application, we simply pulled the data from the memory and used them. We created four separate databases for SIFT, SURF, SURF128 and ORB features.

4.2.3 Search for an Image

Once features are extracted from an image, they can be used in a manner similar to keywords in text retrieval (Baeza-Yates & Ribeiro-Neto 1999). Each feature extracted from the query image is compared with all cluster centres to find the cluster to which they

belong. The search engine will then compute the tf-idf score for relevant cluster images. The list of candidate images is returned ranked in order of their tf-idf score.

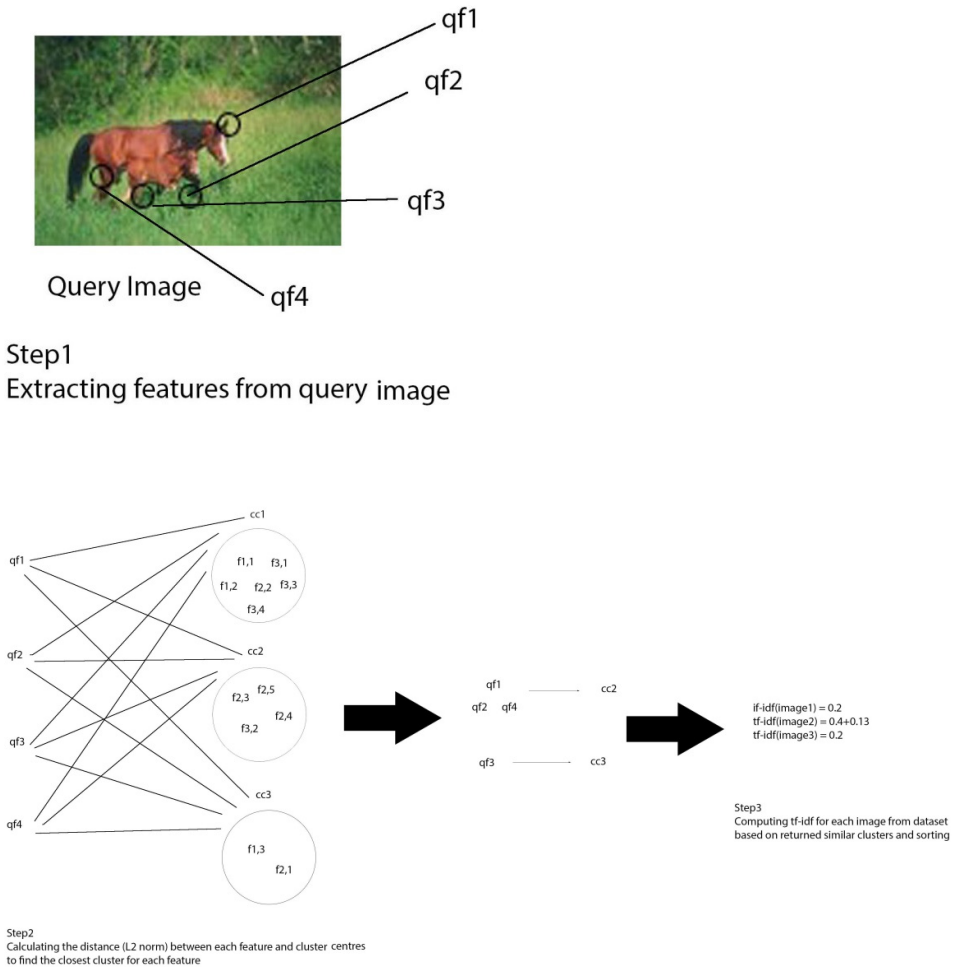


Figure 4-4. Searching for similar images

4.2.4 Selecting Top 10 Images

Studies have shown that around 80% of Web searchers view no more than 10 to 20 results (Jansen & Spink 2003; Spink et al. 2002; Jansen et al. 2000; Jensen 2011). In

our crowdsourcing subsystem, we put the results of the CBIR system in a web page and asked crowdworkers to rate them: based on our previous studies, putting a higher number of images in the crowdsourcing task increases the cognitive load associated with the task and, as a result, the system execution time increases and performance decreases. Based on these studies, we selected the top 10 images from the top 50 similar images returned by the CBIR system and prepared them as input for the crowdsourcing subsystem.

4.2.5 Measuring the System Performance

The CBIR system output is a list of images similar to the query image. These images are ordered based on their level of similarity to the given query image. To measure the performance of the CBIR system, we had to compare this ranked list with the gold standard provided in the Corel-Princeton image dataset. We used the same method as the one we used in Chapter 3 which is calculating the Spearman ρ and Spearman Distance to evaluate the system performance.

4.2.6 Dataset

In order to assess the performance of our image similarity search system, we needed an image dataset which contains gold-standard data. Corel-Princeton[9] is an image similarity benchmark dataset which has been created at Princeton University using Corel image sets. Our study's dataset contains eight query images (airplane, beach, car, flower, horse, fruit, model, columns) and for each query image there is a set of 48-59 images grouped in a folder named by the category of query image (airplane, car …). All images in each

[9] http://www.cs.princeton.edu/cass/benchmark/

folder are similar to the query image related to that folder. These images are ranked based on a similarity measure. Ground truth is created based on a human subject study of 121 people. In our studies, we used six query images and their corresponding similar images to construct our feature dataset and to search for similarities.

4.3 Analysis and Results

We developed our proposed CBIR system using C# programming language and we used Microsoft SQL Server 2012 as the database to save image features. We evaluated this CBIR system using the Corel-Princeton dataset. We used six out of eight sets of the Corel-Princeton dataset (airplane, car, flower, horse, fruit and model). In each round of the experiment, the query image from each set was selected and fed into the CBIR system to search for similar images. The top 10 similar images were then selected and removed from the image set to create a new set. For the newly created image set, we again repeated the feature database creation, as explained in Section 4.2.2.1. We repeated this process until the remaining images from each set were less than 10. This process was repeated for each set of six image sets and, as a result, we increased the number of "query-list of similar images" from six to 20.

This experiment was repeated extracting SIFT, SURF, SURF128 and ORB features and the resulting ranked list of images were saved for analysis. Figure 4-5 to Figure 4-8 are some samples of the query image and the ranked list of images returned using SIFT and SURF and the gold-standard rank.

Figure 4-5. Sample Query Image

Figure 4-6. Top 10 similar images to the query image using SIFT feature (ordered from left to right)

Figure 4-7. Top 10 similar images to the query image using SURF feature (ordered from left to right)

Figure 4-8. Gold-standard

We calculated the Spearman ρ and Spearman Distance for each set of "query-ranked list of similar images" with the gold standard (Table 4-1,Table 4-2 and Figure 4-9).

Table 4-1. Spearman ρ between ranks using SIFT, SURF, SURF128, ORB with the Goldstandard

Spearman ρ				
ImageSet	SIFT	SURF	SURF128	ORB
airplane1	-0.1273	-0.41818	-0.2848	0.3697
airplane2	0.09091	-0.33333	-0.1636	0.06667
airplane3	-0.3576	0.10303	0.4303	-0.6364
car1	-0.1879	-0.29697	0.01818	-0.1152
car2	0.11515	0.06667	0.68485	0.09091
car3	-0.5394	-0.47879	0.24848	-0.0545
car4	-0.3939	0.24848	0.04911	0.01818
flower1	0.55152	0.27273	-0.1273	-0.4061
flower2	0.53939	-0.10303	0.06667	0.01818
flower3	-0.1636	-0.11515	-0.4303	0.30909
fruit1	-0.3576	-0.21212	0.2	-0.103
fruit2	-0.3455	0.22424	0.66061	0.58788
fruit3	0.50303	-0.18788	-0.1273	-0.0909
horse1	0.40606	0.52727	0.45455	-0.3212
horse2	0.00606	0.68485	-0.15151	-0.2485
horse3	-0.0909	0.6	0.45455	0.10303
model1	0.27273	-0.23636	-0.1515	0.47879
model2	0.52727	0.24848	0.17576	-0.1273
model3	-0.297	-0.0667	0.72121	0.28485
model4	-0.1273	-0.30909	0.06667	0.04242
Average	**0.001206**	**0.0109075**	**0.133065**	**0.01333**

Table 4-2. Spearman Distance between ranks using SIFT, SURF, SURF128, ORG and Goldstandard

Spearman Distance				
ImageSet	SIFT	SURF 64	SURF128	ORB
airplane1	38	42	34	24
airplane2	32	42	36	30
airplane3	42	34	24	46
car1	32	38	36	40
car2	30	34	16	32
car3	46	40	28	32
car4	40	24	31	32
flower1	22	24	38	46
flower2	20	32	34	30
flower3	55	36	44	26
fruit1	42	38	28	40
fruit2	40	26	26	20
fruit3	20	38	36	32
horse1	24	22	22	42
horse2	28	20	20	38
horse3	36	20	22	28
model1	30	36	36	20
model2	22	28	28	36
model3	40	55	14	28
model4	40	42	34	32
Average	**33.95**	**33.55**	**29.35**	**32.7**

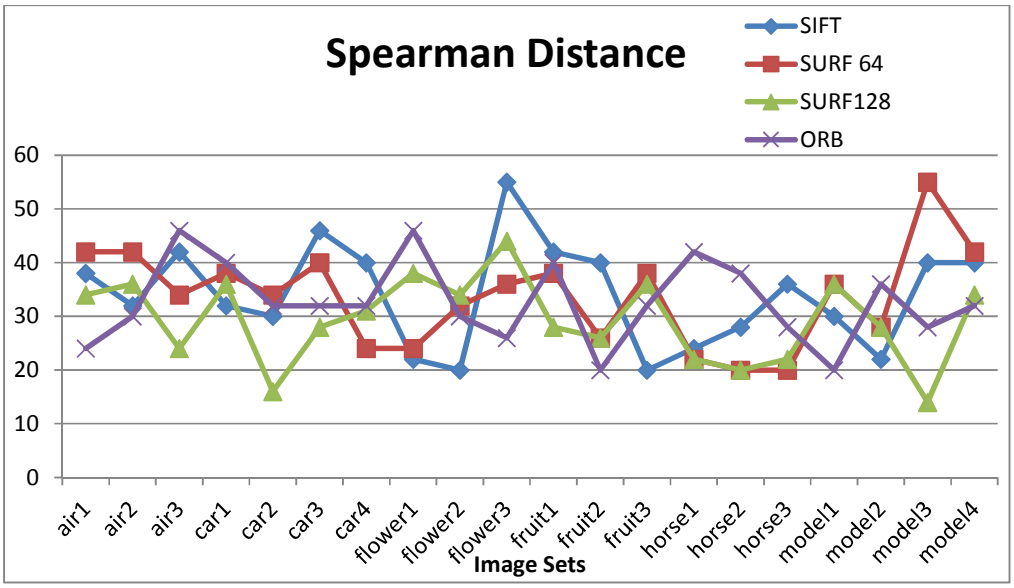


Figure 4-9. Spearman Distance for different feature extractors

4.4 Summary

In this chapter, the architectural design of our CBIR system was explained. We used this system to create a ranked list of similar images to the query images from the Corel-Princeton dataset. We examined different feature detector/descriptors to create these ranked lists. The ranked list of images from this part of our experiment was used in the second part of our experiment to be re-ranked by crowdworkers. In the next chapter, the crowdsourcing system and changes in the performance of the system are explained in detail.

Chapter 5

Hybrid Human–Machine CBIR System

In the previous chapter, the architecture of the computational CBIR system was explained. The results from the CBIR system were used as input to the crowdsourcing subsystem of the hybrid CBIR system. This chapter presents our experiment and the hybrid human–machine CBIR system design and the results of our study on the effect of using the power of the crowd on an image similarity search.

5.1 Overview

During the past decades, the evolution of information technology has resulted in the design of very powerful computers and algorithms that can do repetitive and complex tasks in a fraction of a second. Nowadays, computers are involved in every aspect of human life to help us to perform tasks more rapidly and more accurately. Despite these advances, there are still tasks which computers perform very poorly, but which humans perform with a high accuracy level. One of these tasks is image similarity check. Most image similarity algorithms have a very low performance level in relation to the high complexity level of the task. By taking a quick look at some image search engines (i.e.

Google, Bing), we can see that they have poor query-by-example (QBE) image search results (Figure 5-1 and Figure 5-2). On the other hand, humans perform very well in comparing images and measuring the similarity between two images. In such cases, crowdsourcing can be a good solution to improve the accuracy of a system. However, the problem with using only crowdsourcing in image similarity is that with large image databases, it is not possible to ask humans to search for similar images based on a given query image. It will be a time-consuming and expensive task.

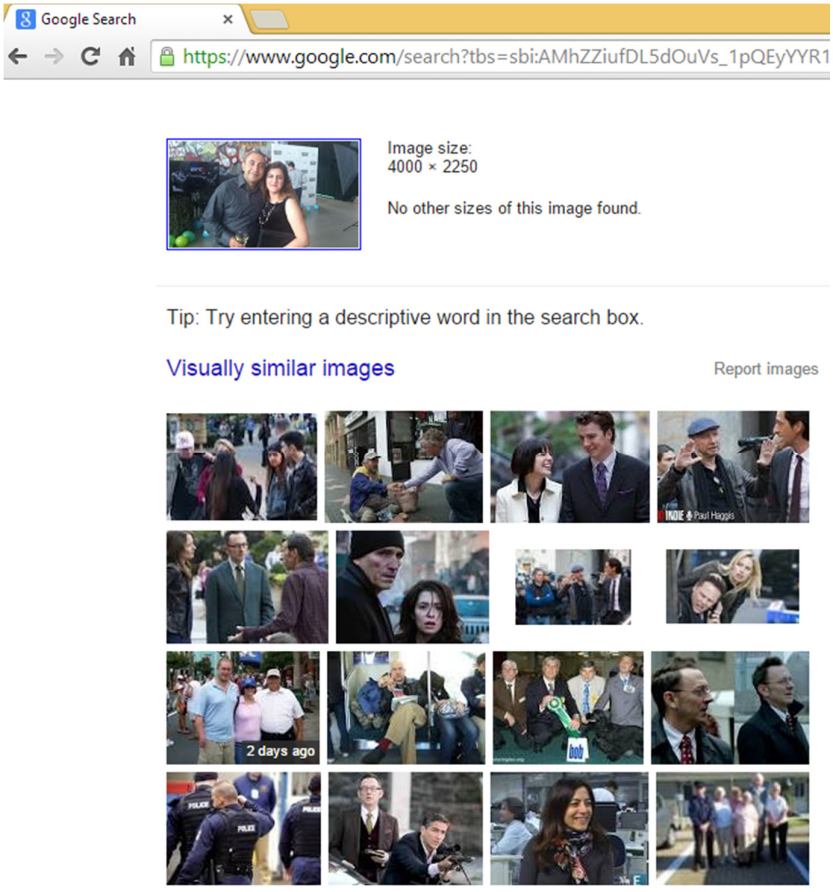


Figure 5-1-Example of Google Query-by-Image search returning irrelevant results

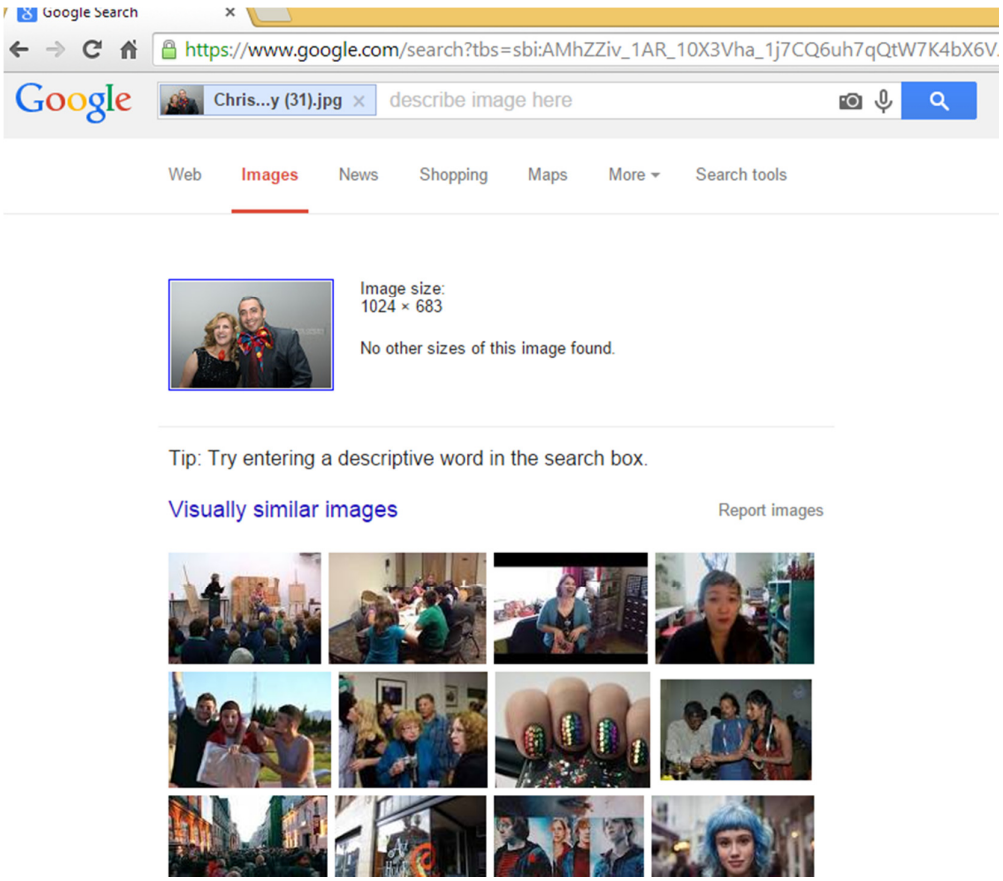


Figure 5-2- Another sample of Google QBE returning irrelevant results

Several studies have tried to combine the power of the human computation with the computational power of the machine and design a hybrid human–machine system to improve the performance (Wang et al. 2012; Yan et al. 2010; Franklin et al. 2011). These studies have confirmed that by involving the crowd in tasks that computers do not perform well, the overall performance of the system can be increased.

5.2 Research Questions and Hypotheses

As previously mentioned, computational algorithms are very limited in describing the conceptual information of images. This limitation has caused very low performance in CBIR systems. Furthermore, the human is highly capable of comparing images and easily defines the degree of similarity between two images. Therefore, our research question is:

> Can the power of the crowd cover the limitations of a CBIR system and does crowdsourcing improve the performance of a CBIR system using a hybrid human–machine system?

The specific hypothesis is:

H3: Involving crowds in a CBIR system and designing a human–machine hybrid image similarity search system will improve the overall system's performance.

5.3 Research Design and Methodology

To test our hypothesis, we designed a simple CBIR system using four different feature detectors/descriptors as the first subsystem of our hybrid system (the system architecture was explained in Chapter 4). For the second subsystem, we crowdsourced the output results of an image similarity search of the CBIR system. The performance of the system was computed in each subsystem and compared with each other. In contrast, our goal is to compare the performance of a pure CBIR image similarity search system using four feature detector/descriptors (SIFT, SURF, SURF128 and ORB) with the performance of hybrid Human–Machine image similarity search system.

Table 5-1. Research Design

Feature Detector/Descriptor	Pure Computation CBIR System Performance	Hybrid Human–Machine System Performance
SIFT	20 ImageDatsets	Same 20 ImageDatasets 10 Unique Mturk Workers
SURF	20 ImageDatsets	Same 20 ImageDatasets 10 Unique Mturk Workers
SURF128	20 ImageDatsets	Same 20 ImageDatasets 10 Unique Mturk Workers
ORB	20 ImageDatsets	Same 20 ImageDatasets 10 Unique Mturk Workers

In the following section, the crowdsourcing subsystem, our experiment and the results are described.

5.3.1 Selecting Crowdsourcing Platform

As noted in Chapter 2, there are different platforms that can be used to perform microtasking crowdsourcing tasks. CrowdFlower and Amazon Mechanical Turk (MTurk) are two very famous examples. We compared these two platforms to choose the best one to match our requirements:

CrowdFlower is a crowdsourcing service founded in 2009 by Lukas Biewald and Chris Van Pelt. In this microtasking platform, it is possible to provide gold-standard data along with the requested task to measure the performance of each worker. CrowdFlower puts the benchmark data within the actual task, compares the results provided with the benchmark and calculates a quality factor for each worker. Based on the quality factor, crowdsourcers can accept or reject responses of a specific worker.

CrowdFlower provides a very limited tool to define the crowdsourcing task. This limited tool prevents the design of tasks with different user interfaces (UIs). For text-

based crowdsourcing tasks, CrowdFlower is a very efficient tool but for tasks involving images or other objects, CrowdFlower is not a good platform.

MTurk is another microtasking crowdsourcing platform: it was founded by Amazon and is a marketplace in which requesters can put their tasks and workers can sign into the system and do the tasks. Tasks on MTurk are called HITs (Human Intelligence Tasks) and grouped into HITGroups. Requesters can assign each HIT to more than one worker to be done. Tasks in MTurk typically take no longer than a minute but, at the extreme end of the scale, some tasks may require an hour to complete. Some of the HITs are just one single task but some can be a collection of, for example, comparison between 50 images.

MTurk provides two different tools for designing crowdsourcing tasks. The first one is its online task designer which is limited but suitable for simple text-based tasks or comparison tasks. MTurk also provides different programming APIs for developers to design customised tasks programmatically. Using these APIs and a special task type called “ExternalQuestions”, it is possible to design highly customised crowdsourcing tasks.

MTurk is limited in defining quality control procedures. The only method provided directly with MTurk is qualification tests. MTurk makes it possible to select from predefined qualification controls or to design a specific qualification task and based on the score of workers, they can gain access to the actual task. Our studies on defining qualification tasks have shown that custom-designed qualification tasks will result in less willingness of workers to accept the task. In other words, workers prefer to perform the tasks without qualification controls and if a task has a qualification test fewer workers

will accept it. One solution to this problem is putting quality control constraints within the task (such as time) and rejecting the worker's response depending on the constraint. This method is explained in Section 5.3.3.

Regarding the nature of our crowdsourcing task which deals with images and also our plan to design an automated hybrid system, we found MTurk to be a more flexible platform for our task.

5.3.2 What is the Right Amount of Reward for the Task?

The amount of reward in microtasking crowdsourcing plays an important role. While (Harris 2011)suggest that an increased reward encourages the quality of results in a resumé review task, other studies have shown that an increased reward results in decreased response time and increased demand for the task but not necessarily increases the quality in Amazon Mechanical Turk (MTurk) (Franklin et al. 2011; Faridani et al. 2011; Buhrmester et al. 2011; Mason & Watts 2009).

Rewards on MTurk vary from $0.01 to $50 and, based on the difficulty of the task and the average time it takes to be completed by workers, the reward is calculated with the US (United States) minimum wage of $8/hour.

To decide on the right amount of reward for our task that would attract workers and, at the same time, not have a negative impact on the quality of responses, we examined different rewards. Our crowdsourcing task was very simple and did not take more than two minutes on average so we decided to study $0.01, $0.05 and $0.10 rewards. The results of this preliminary study showed that $0.01 did not attract enough workers in our

defined time frame (two hours). As $0.05 and $0.10 rewards were not very different in terms of the number of workers in the time frame and the quality of workers' responses, we decided to use $0.05 as the reward for our task.

5.3.3 HIT Quality Control

Unlike CrowdFlower, MTurk does not have a built-in mechanism to provide benchmarked data for the tasks to test the quality of crowdworkers' responses. Instead, MTurk has a tool called "Qualification Type". There are a number of predefined Qualification Types such as Masters, Categorization Masters, Photo Moderation Masters, etc. If requesters put one of these qualification requirements in the HIT design, only workers with these qualifications will be able to perform the task. In our task, we set Photo Moderation Masters as the qualification requirements of our HIT.

MTurk also gives requesters the ability to design custom qualification tests for their HIT. In this case, workers had to pass the qualification tests in order to be able to perform the task. Our preliminary studies showed that assigning custom-designed qualification tests for HITs will result in a reduced number of workers who accept the task.

Another option to control the quality of crowdworkers' responses is to define some factors to detect and reject low quality results. In the case of our experiment, we decided to put a hidden timer in the task to compute the time each worker spent to finish the task. The time to complete our task correctly was from 90-180 seconds and we rejected the tasks with time of less than 30 seconds as low quality results.

We also used another mechanism to prevent workers from skipping the task or entering incorrect responses. We designed the task user interface (UI) in such a way that workers had to complete the task and provide ratings for all images before submitting the response. In addition to this, workers were not allowed to enter numbers out of our specific range for rating images. (The user interface (UI) design of the task in explained in Section 5.3.5.)

5.3.4 How Many Crowdworkers for Each HIT?

The power of microtasking crowdsourcing is that more than one worker performs the task and studies have shown that these aggregated results are reliable and can even be used as ground truth (Urbano et al. 2010). The number of crowdworkers performing a single HIT varies depending on the type of the task and can be at least two. Studies by Blanco et al. (2011) and Snow et al. (2008) have shown that three and four judgments are sufficient to build the gold standard using crowdsourcing.

We examined the performance of aggregated results using different numbers of workers. We examined five, 10 and 20 workers for our HIT. Our studies showed that asking five workers to perform the task will result in low performance aggregated results. While the performance of aggregated results from 10 workers was very close to the performance using 20 workers, the price and execution time of the task using 20 workers for each HIT were higher. Based on the findings of this study, we decided to ask 10 workers to perform each HIT.

In MTurk, to assign the HIT to more than one crowdworker, we can set the number of assignments to our desired number. MTurk then will create these assignments and allow crowdworkers to accept them and perform the job. Using this process, MTurk prevents crowdworkers from performing the HIT more than once and we can make sure that the results are unique and there are no duplicates. Crowdworkers are allowed to accept other HITs that we publish. In our experiment, we submitted 80 HITs, each with 10 assignments (total of 800 HITs): we had 796 accepted HITs with 248 unique workers and approximately 75% of workers performed more than one HIT during our experiment.

5.3.5 User Interface Design

Our studies in Chapter 3 showed that the user interface (UI) design of crowdsourcing tasks affects the performance of crowdworkers' responses. For this part of our experiment, we used the Rate UI design, which had the higher performance of the three UI designs, for our HITs. In a very similar way to our experiment in Chapter 3, we put the query image and the images returned from the CBIR system in their original ranking into an HTML page and asked crowdworkers to rate the degree of similarity of each image with the given query image.

As previously noted, we put some controls in the UI design to prevent cheaters or workers who tried to skip performing the task. One of the mechanisms we used to ensure that workers did not skip the task is that they were allowed to submit the task only when they provided ratings for every one of the 10 images on the page. The advantage of using this procedure is that we could make sure that workers provided a rating for each image; however, this rating might not necessarily be a good rating. Another control we put inside

the HIT page was that workers were allowed to enter a scale in our HIT provided it was in the specific range and any number out of the range was not accepted by the system. By using this control, we could eliminate an out-of-range rating; however, again we could not make sure that the provided rating was an accurate and careful rating.

5.3.5.1 Rating Scale

There are several rating scales used in recommendation or review systems. Some systems use 1-10 scales and many other systems use 1-5 rating scales (IMDB[10], NetFlix[11], Movielens (Miller et al. 2003)): in our experiment, we decided to use a 1-5 rating scale. We asked crowdworkers to rate an image 5 if the image was very similar to the query image and 1 if the image was not very similar to the query image. Figure 5-3 presents a screenshot of a HIT.

[10] http://www.imdb.com/
[11] https://www.netflix.com/

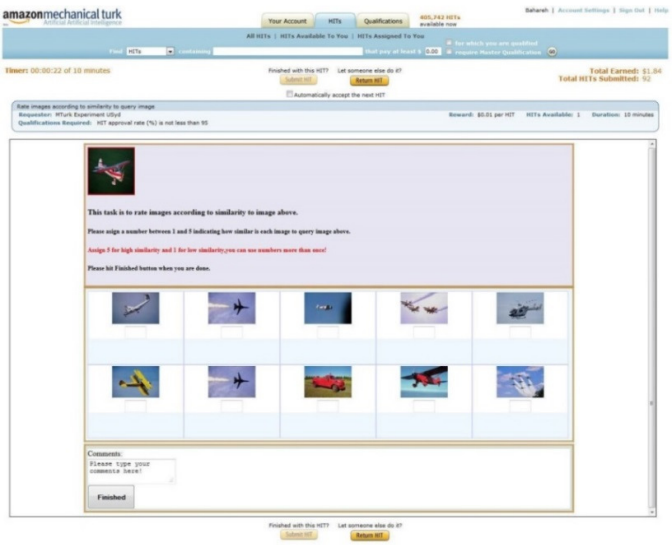


Figure 5-3. HIT screenshot

5.3.6 Aggregating Crowdsourced Responses

Similar to our user interface (UI) study (Chapter 3), we computed the average of the ratings to aggregate the crowd's responses.

In our research, we used the Spearman ρ for the rank correlation coefficient between the aggregated ranking by crowdworkers for each UI design and the gold-standard data. While ρ is a nonparametric value, we also calculated Spearman's footrule distance to compare rankings.

5.4 Experimental Results

We used the system we designed in Chapter 3 to create HITs and collect results. This system consists of three applications (Figure 5-4). The first part is the CBIR system that we designed in Chapter 4. The second part of this system comprises PHP scripts and a

MySQL database used to receive information from the CBIR system (QueryImage-RankedList pairs) to create HIT HTML pages. The final part of the system is another C# application developed to create the HITs, send them to MTurk and programmatically collect the crowd's responses. We used MTurk C# API for this part of the system.

The difference between this experiment and the Chapter 3 experiment is that in this experiment, we did not use random images in our HITs. The HIT pages were created using the QueryImage-RankedList pairs of the CBIR system as explained in Chapter 4. As was also explained in Chapter 4, we used six sets out of eight sets of the Corel-Princeton dataset with their corresponding query image. Each query image was given to the CBIR system as input and the top 10 images from the results were selected and removed from the image set to create a new set. This process was repeated until the remaining images from each set were less than 10. By using this procedure for each image set result, we created 20 QueryImage-RankedList pairs. Each QueryImage-RankedList pair was used to create an HTML page with the Rate UI designed HIT. In total, 20 HITs were created using our C# application each with 10 assignments. This process was repeated for each feature detector/descriptor (SIFT, SURF, SURF128 and ORB) and in total we published 4*20 HITs and 800 assignments.

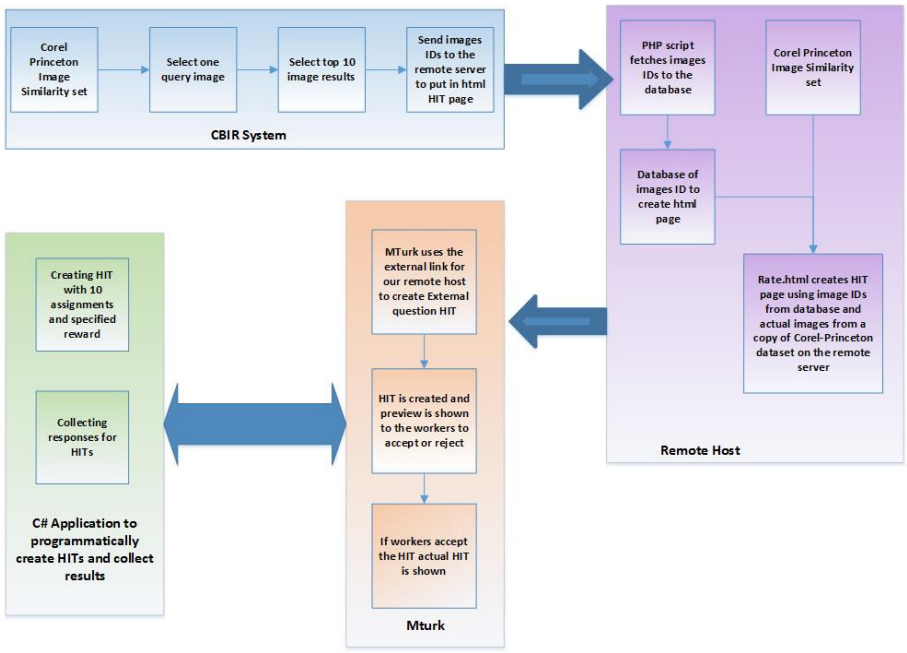


Figure 5-4. System overview

We designed our HITs with these controls:

- $0.05 reward
- instructions for users to do the task
- each worker was permitted to perform the task only one time
- added time stamps to the design of the HIT to identify and remove the workers who just clicked and did not perform the task carefully
- for each HIT, we asked 10 workers to perform the task (10 assignments).

Table 5-2 shows the Spearman ρ for the different feature detectors/descriptors in machine only (CBIR) and machine+crowd. Figure 5-5 to Figure 5-8 compare the Spearman ρ in the pure computational results and the results after involving crowds.

Table 5-2. Spearman ρ between machine only and machine+crowds ranking for using different feature types

Spearmanρ								
	SIFT		SURF		Surf128		ORB	
	Machine only	Machine +Crowd	Machine only	Machine +Crowd	Machine only	Machine +Crowd	Machine only	Machine +Crowd
airplane1	-0.1273	0.85107	-0.41818	0.8997	-0.2848	0.83538	0.3697	0.98359
airplane2	0.09091	0.94833	-0.33333	0.65654	-0.1636	0.84148	0.06667	0.68485
airplane3	-0.3576	-0.28747	0.10303	0.95099	0.4303	0.95441	-0.6364	0.80489
car1	-0.1879	0.93734	-0.29697	0.82675	0.01818	0.79	-0.1152	0.93
car2	0.11515	0.81961	0.06667	0.95417	0.68485	0.87175	0.09091	0.8997
car3	-0.5394	0.35976	-0.47879	0.56364	0.24848	0.79028	-0.0545	0.74164
car4	-0.3939	0.93582	0.24848	0.89091	0.04911	0.85595	0.01818	0.82471
flower1	0.55152	0.92075	0.27273	0.81818	-0.1273	0.88689	-0.4061	0.88681
flower2	0.53939	0.83891	-0.10303	0.82572	0.06667	0.79028	0.01818	0.8651
flower3	-0.1636	0.83538	-0.11515	0.93294	-0.4303	0.92402	0.30909	0.76693
fruit1	-0.3576	0.67273	-0.21212	0.67684	0.2	0.54776	-0.103	0.49083
fruit2	-0.3455	0.30909	0.22424	0.99392	0.52727	0.66061	0.58788	0.6383
fruit3	0.50303	0.13583	-0.18788	0.87879	-0.1273	0.88626	-0.0909	0.44928
horse1	0.40606	0.91515	0.52727	0.94833	0.45455	0.98481	-0.3212	0.91186
horse2	0.00606	0.95099	0.68485	0.97242	-0.15151	0.96363	-0.2485	0.9301
horse3	-0.0909	0.7805	0.6	0.89857	0.45455	0.93872	0.10303	0.91465
model1	0.27273	0.36426	-0.23636	0.75988	-0.1515	0.75758	0.47879	0.73172
model2	0.52727	-0.0614	0.24848	0.82675	0.17576	0.89362	-0.1273	0.92575
model3	-0.297	0.81495	-0.0667	0.64026	0.72121	-0.5627	0.28485	0.29879
model4	-0.1273	0.86323	-0.30909	0.3988	0.06667	0.07976	0.04242	0.8693
Average	**0.001206**	**0.645242**	**0.010908**	**0.815705**	**0.1333065**	**0.734525**	**0.01333**	**0.77744**

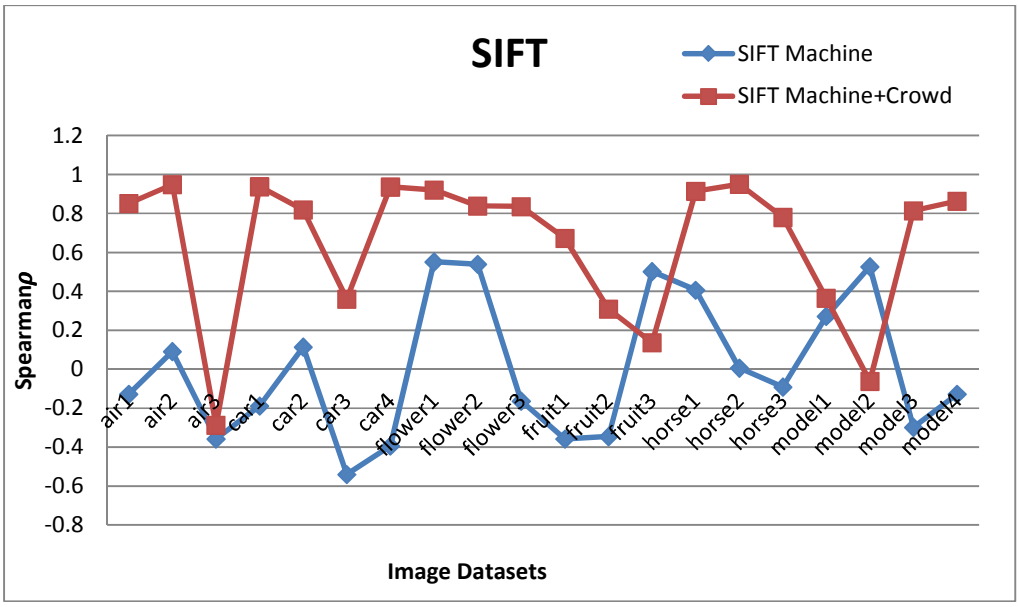


Figure 5-5. Spearman ρ for SIFT

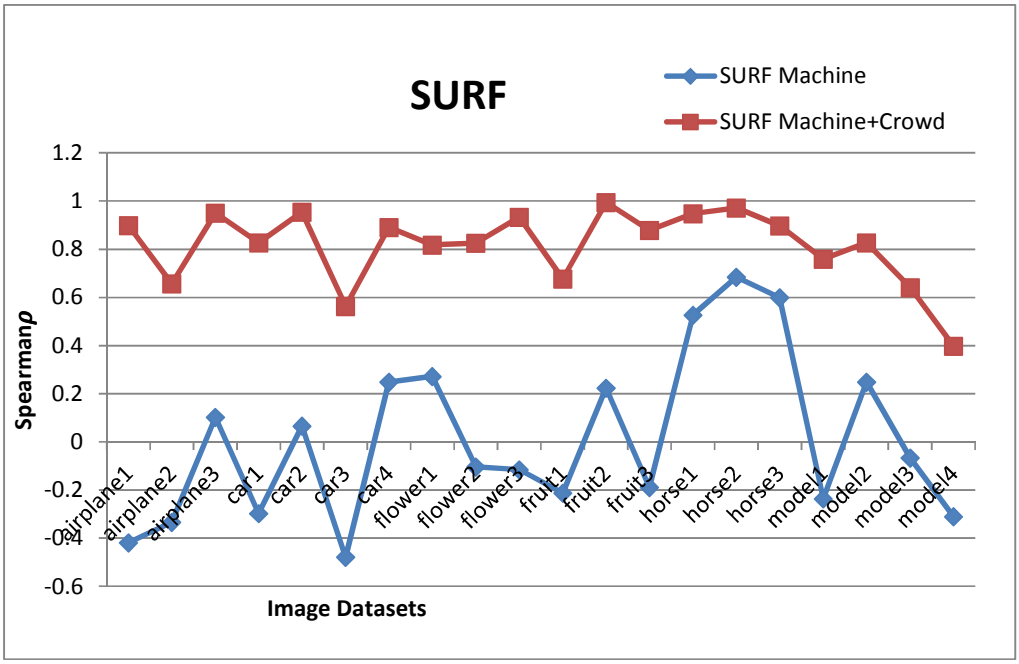


Figure 5-6. Spearman ρ for SURF

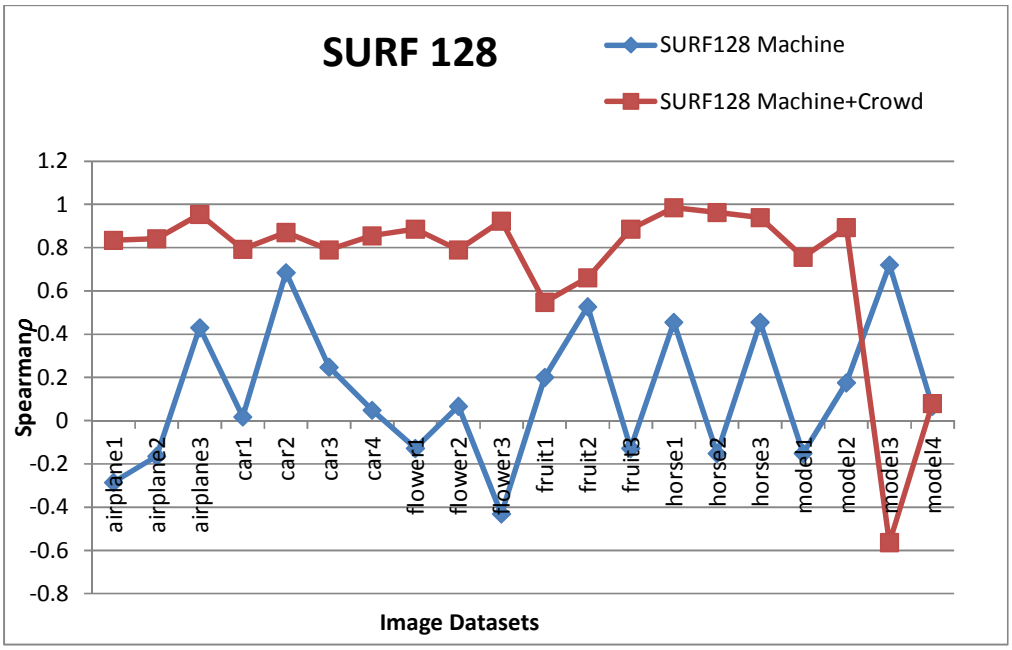


Figure 5-7. Spearman ρ for SURF128

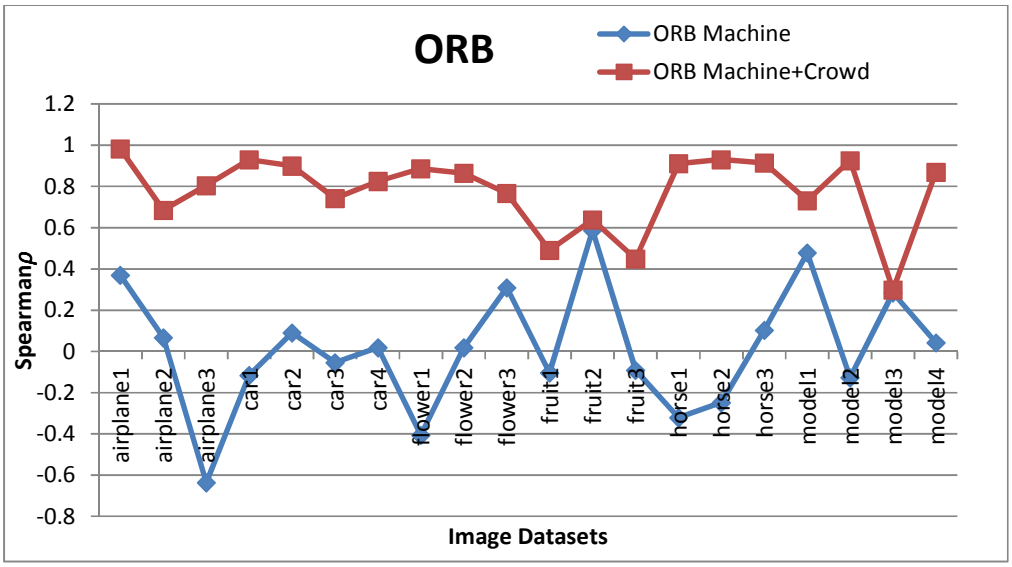


Figure 5-8. Spearman ρ for ORB

Results show that the Spearman ρ is increased by involving crowds in the image similarity search for all four features in most of the datasets. The increased rank correlation coefficient shows that the ranked list provided after crowdsourcing is more similar to the gold standard and can be translated into a higher performance system.

Since Spearman ρ is not a scalar value, we decided to compute the Spearman Distance. The following tables and figures show the Spearman Distance between machine-only results and the gold standard, compared with the Spearman Distance between machine+crowd results and the gold standard.

In Figure 5-9 to Figure 5-12 we can see how much crowdsourcing has decreased the Spearman Distance with the gold standard and improved the search performance.

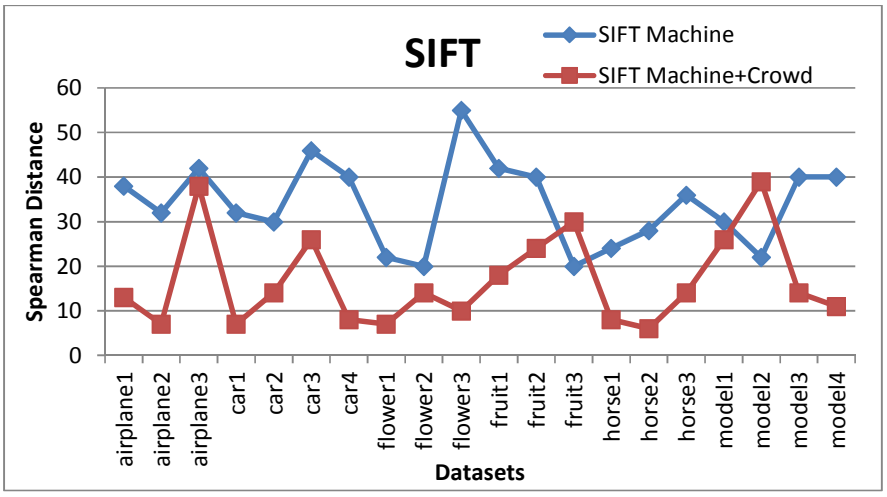


Figure 5-9. SIFT Spearman Distance

SIFT-Spearman Distance		
Dataset	Machine	Machine+crowd
airplane1	38	13
airplane2	32	7
airplane3	42	38
car1	32	7
car2	30	14
car3	46	26
car4	40	8
flower1	22	7
flower2	20	14
flower3	55	10
fruit1	42	18
fruit2	40	24
fruit3	20	30
horse1	24	8
horse2	28	6
horse3	36	14
model1	30	26
model2	22	39
model3	40	14
model4	40	11

t-Test: Paired Two Sample for Means

	Variable 1	Variable 2
Mean	33.95	16.7
Variance	91.20789474	106.5368
Observations	20	20
Pearson Correlation	-0.033797454	
Hypothesized Mean Difference	0	
df	19	
t Stat	5.395792804	
P(T<=t) one-tail	0.0000165492	
t Critical one-tail	1.729132812	
P(T<=t) two-tail	0.0000330984	
t Critical two-tail	2.093024054	

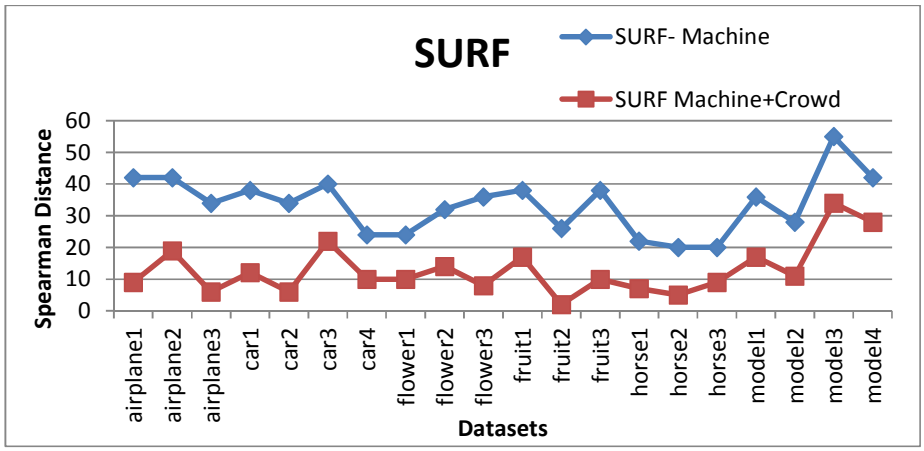


Figure 5-10. SURF Spearman Distance

SURF-Spearman Distance		
Dataset	Machine	Machine+Crowd
airplane1	42	9
airplane2	42	19
airplane3	34	6
car1	38	12
car2	34	6
car3	40	22
car4	24	10
flower1	24	10
flower2	32	14
flower3	36	8
fruit1	38	17
fruit2	26	2
fruit3	38	10
horse1	22	7
horse2	20	5
horse3	20	9
model1	36	17
model2	28	11
model3	55	34
model4	42	28

t-Test: Paired Two Sample for Means		
	Variable 1	Variable 2
Mean	33.55	12.8
Variance	82.15526316	64.37895
Observations	20	20
Pearson Correlation	0.741934157	
Hypothesized Mean Difference	0	
df	19	
t Stat	14.93261411	
P(T<=t) one-tail	0.0000000000030	
t Critical one-tail	1.729132812	
P(T<=t) two-tail	0.0000000000060	
t Critical two-tail	2.093024054	

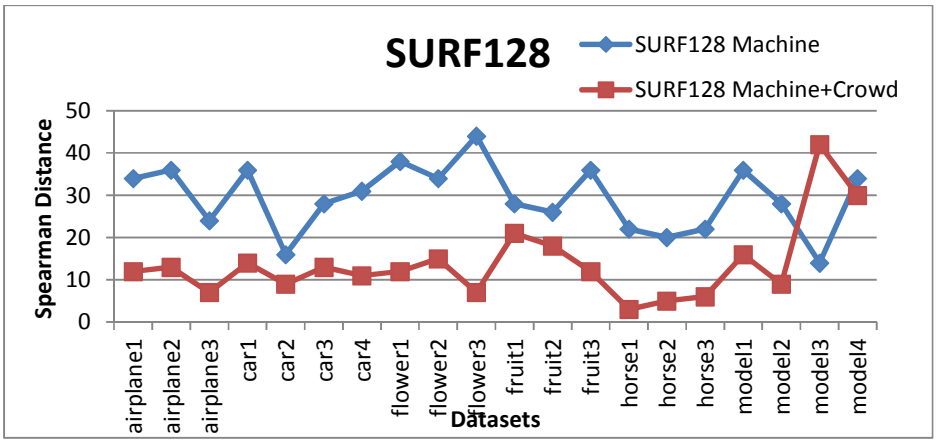


Figure 5-11. SURF128 Spearman Distance

SURF128-Spearman Distance		
Dataset	Machine	Machine+Crowd
airplane1	34	12
airplane2	36	13
airplane3	24	7
car1	36	14
car2	16	9
car3	28	13
car4	31	11
flower1	38	12
flower2	34	15
flower3	44	7
fruit1	28	21
fruit2	26	18
fruit3	36	12
horse1	22	3
horse2	20	5
horse3	22	6
model1	36	16
model2	28	9
model3	14	42
model4	34	30

t-Test: Paired Two Sample for Means		
	Variable 1	Variable 2
Mean	29.35	14.7
Variance	62.55526316	81.90526316
Observations	20	20
Pearson Correlation	-0.261689655	
Hypothesized Mean Difference	0	
df	19	
t Stat	4.857446224	
P(T<=t) one-tail	0.0000546961	
t Critical one-tail	1.729132812	
P(T<=t) two-tail	0.0001093922	
t Critical two-tail	2.093024054	

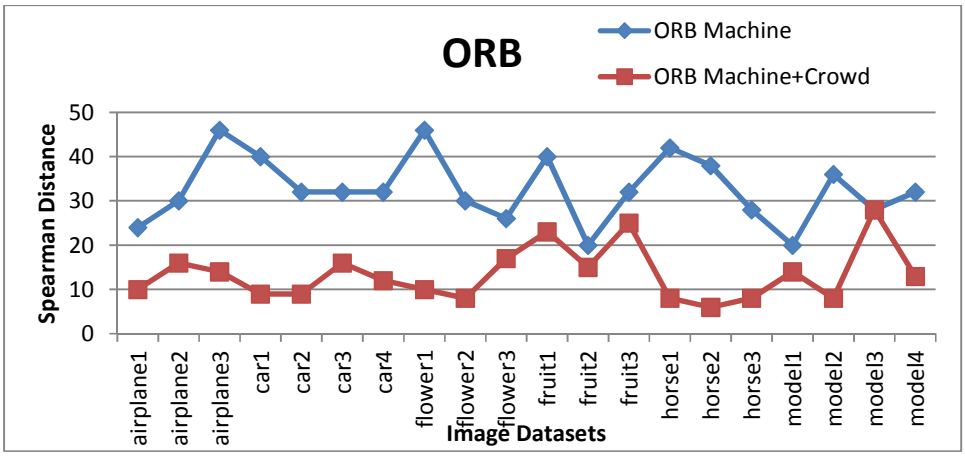


Figure 5-12. ORB Spearman Distance

ORB-Spearman Distance		
Dataset	Machine	Machine+Crowd
airplane1	24	10
airplane2	30	16
airplane3	46	14
car1	40	9
car2	32	9
car3	32	16
car4	32	12
flower1	46	10
flower2	30	8
flower3	26	17
fruit1	40	23
fruit2	20	15
fruit3	32	25
horse1	42	8
horse2	38	6
horse3	28	8
model1	20	14
model2	36	8
model3	28	28
model4	32	13

t-Test: Paired Two Sample for Means

	Variable 1	Variable 2
Mean	32.7	13.45
Variance	57.37894737	37.10263
Observations	20	20
Pearson Correlation	-0.19425956	
Hypothesized Mean Difference	0	
df	19	
t Stat	8.119837252	
P(T<=t) one-tail	0.0000000671	
t Critical one-tail	1.729132812	
P(T<=t) two-tail	0.0000001343	
t Critical two-tail	2.093024054	

We also computed the average Spearman Distance for all image sets for each feature detector/descriptor. Running t-Test confirms that results are statistically significant and we can't reject the null hypothesis of Machine+Crowd having higher performance.

Figure 5-13. Average Spearman Distance

Average Spearman Distance		
	Machine Only	Machine+Crowd
SIFT	33.95	17
SURF	33.10526	12
SURF128	29.10526	13.89474
ORB	32.73684	13.47368

t-Test: Paired Two Sample for Means		
	Variable 1	Variable 2
Mean	32.22434	14.09211
Variance	4.581746	4.418053
Observations	4	4
Pearson Correlation	0.257361	
Hypothesized Mean Difference	0	
df	3	
t Stat	14.02696	
P(T<=t) one-tail	0.000392	
t Critical one-tail	2.353363	
P(T<=t) two-tail	0.000785	
t Critical two-tail	3.182446	

5.5 Discussion

The results of our experiment highlight the effects of combining the power of crowds with computational algorithms to improve the performance of the overall system. Crowdsourcing is a very cost-effective way with reasonable speed and accuracy of involving humanss in some areas of studies where computers have low performance. Based on the findings of our studies, image similarity search or CBIR systems have good potential to achieve performance improvement if combined with crowds. Our results showed that the performance of the CBIR system significantly increased after re-ranking the image list with crowdsourcing. Further experiments on a diverse range of image datasets and crowdsourcing platforms will help in designing a universal hybrid platform.

The detailed analysis of results showed that the Spearman Distance was decreased in almost all of the datasets for every feature detector/extractor except in three datasets (SIFT fruit3 and model2, SURF128 model3).

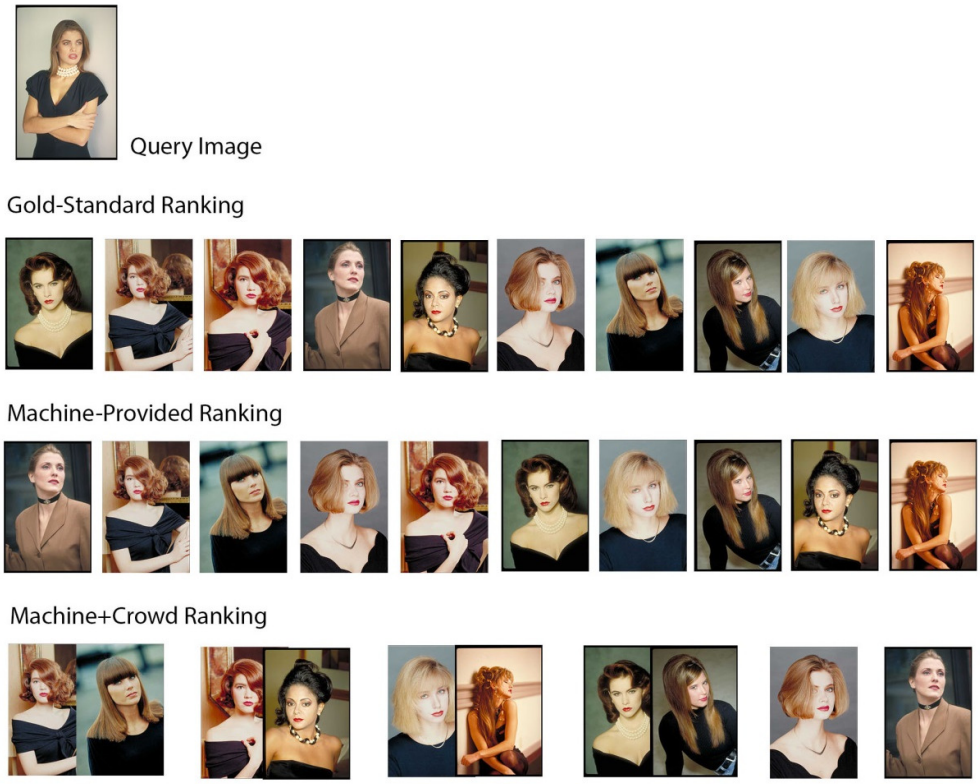


Figure 5-14. SIFT model2 dataset–Sample of decreased Spearman Distance in hybrid system

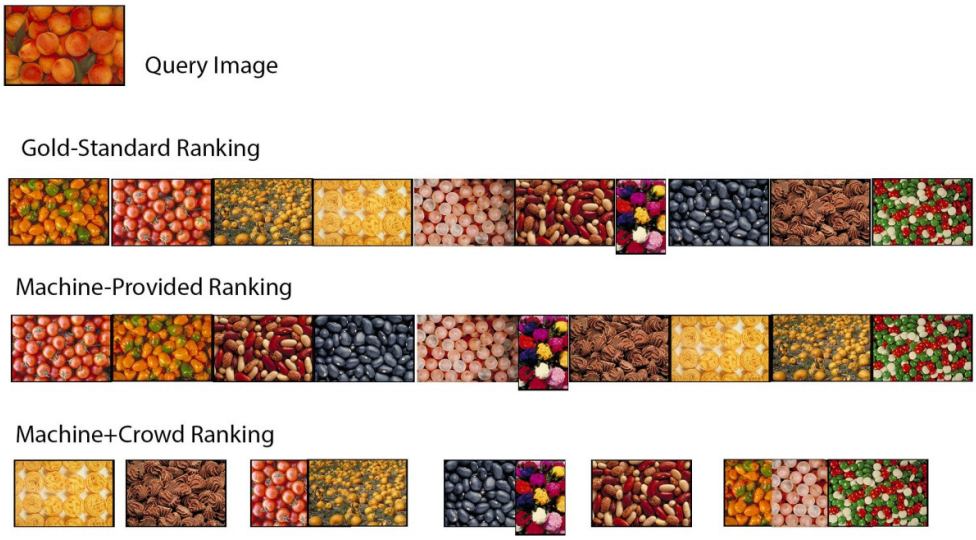


Figure 5-15. SURF128 fruit3 dataset– Sample of decreased Spearman Distance in hybrid system

Two of the datasets in which Spearman Distance is decreased in hybrid system contained images of humans. Studies on cross-cultural facial recognition suggest that cultural differences affect the accuracy of judgments in facial emotions and expressions (McAndrew 1986; Prado et al. 2013). In a universal crowdsourcing platform such as MTurk, crowdworkers can be from a wide range of different cultures and we believe this cultural diversity affected the performance of responses in the dataset that involved facial comparison.

Taking a closer look at the SURF128 fruit3 dataset, we can see that the query image for this dataset is not very clear and result images are not very similar to the query image. We believe that this low level of similarity between query image and results lead to decreased Spearman Distance in hybrid system.

Also results show that for SIFT air3 dataset, Spearman Distance in hybrid system is not much higher than CBIR alone system (Figure 5-16). In depth study of this dataset suggests that for this dataset, similar to SURF128 fruit3 dataset, result images have little similarity to the query image and caused lower Spearman Distance in hybrid system.

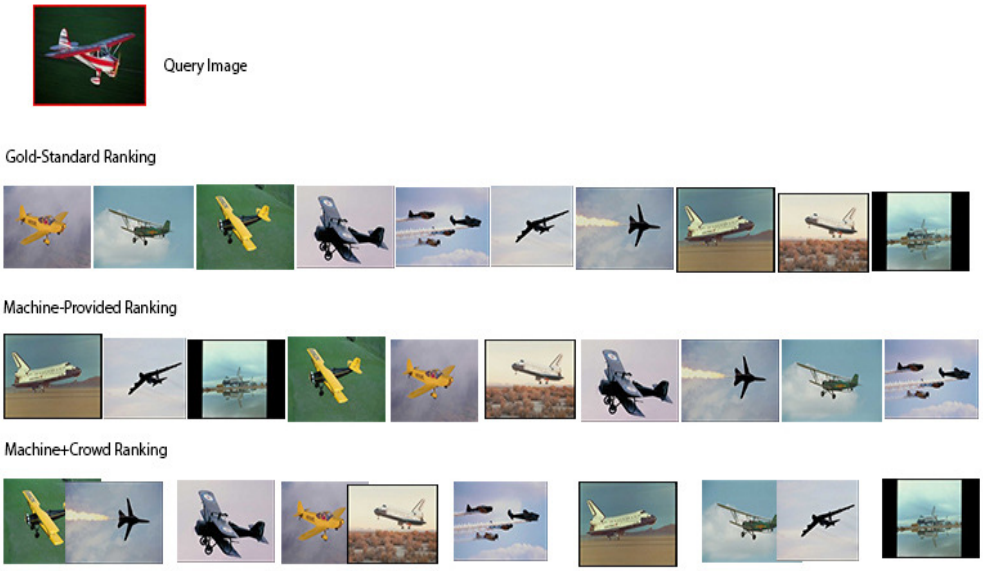


Figure 5-16. SIFT air3 dataset-Hybrid system's performance is low but still higher than CBIR performance

We suggest that further investigation on the performance of crowds on different types of image datasets can help to build the foundation of hybrid human–machine CBIR systems.

5.6 Conclusion

The evolution in digital imaging and the interest in digital images have increased enormously over the past few years and have resulted in the creation of large image databases and highlighted the need for powerful and efficient image retrieval systems.

The limitations of text-based image retrieval systems have led to new-generation content-based image retrieval (CBIR) systems in which shape, colour, texture or other features of images are described and used for similarity search. Despite the advances in feature detection/extraction methods, CBIR systems have a very low performance.

Conversely, humans compare images for similarity with a top-down overall view. Having a conceptual view of images, humans can select a similar image from a collection more effectively and accurately. Crowdsourcing can be a good solution to reduce the gap but the problem with a database of images containing millions of images is that asking the crowd to search for similar images is an expensive and time-consuming task.

Our proposed solution is a hybrid human–machine CBIR system that takes advantage of computational algorithms to search within large image databases and of the power of humans to improve the performance of returned results. We designed a system based on the proposed solution and tested our hypothesis. The results confirmed that involving the crowd in an image similarity search increased the overall performance of the system. Our finding confirms that in some class of problems that machine can't provide high performance results (such as image similarity search), designing a hybrid system which takes advantage of crowdsourcing can lead to higher performance and more accurate results.

Chapter 6

Conclusions

The purpose of this research was to study the effects of combining the power of humans or crowds with computational algorithms on the performance of the resulting hybrid human–machine system. We designed our content-based image retrieval (CBIR) system and combined it with Amazon Mechanical Turk (MTurk) as the microtasking crowdsourcing platform. We also examined different user interface (UI) designs in our crowdsourcing task and studied their effects on the performance of crowd-produced results.

This chapter presents an overview of our findings, the implications and limitations of our research, and suggestions for future work.

6.1 Review of Findings

6.1.1 Hybrid Human–Machine CBIR System

The evolution of the World Wide Web and the establishment of Web 2.0 as a read-write web have provided a framework for user interactions and Internet users are today

not only information consumers but also data providers. The widespread availability of Internet access and the new interactive web framework have given rise to a new workforce which Howe coined as "crowdsourcing" (Howe 2006) in 2006. Since then, many successful projects, platforms and applications have been implemented based on using crowdsourcing (Wikipedia[12], Threadless[13], Amazon Mechanical Turk (MTurk)[14], CrowdDB (Franklin et al. 2011)) and studies have shown that results provided by crowdsourcing are reliable and can even be used as the gold standard (Blanco et al. 2011; Nowak & Rüger 2010).

Even though there have been extensive developments in Artificial Intelligence and computational algorithms, there are still some tasks in which humans outperform computers. Content-based image retrieval (CBIR) systems are an example of such systems. In CBIR systems, images are represented as specific features and similarity comparison is performed by comparing these features. Despite advanced methods for comparing image contents and judging them on the degree of similarity, classic CBIR systems lack performance.

On the contrary, humans are very fast and accurate at comparing images and at defining the degree of similarity; therefore, crowdsourcing can provide a solution to improve the performance of CBIR systems. To search for similar images to a given query image within an image dataset, one solution is to crowdsource the whole image dataset and ask the crowds to find similar images. The problem with this solution is that with a large image dataset consisting of thousands of images, it is very time-consuming and

[12] www.wikipedia.org
[13] www.threadless.com
[14] www.mturk.com

expensive to crowdsource the whole dataset and search for similar images. Another solution is to use a hybrid human–machine system in which computational algorithms search the large dataset of images for similarities and crowds refine the results. To test our specific hypothesis, we designed a hybrid human–machine CBIR system and conducted an experiment using this system. We designed four query-by-example (QBE) CBIR systems using four feature detectors/descriptors (SIFT, SURF, SURF128 and ORB) to search for similar images based on a given query image. Using the Corel–Princeton Image Similarity Benchmark, we compared the performance of the system using each feature detector/descriptor against the gold standard provided by the dataset. Our experiment showed that the systems using SIFT, SURF, SURF128 and ORB have similar performance in a query-by-example image search.

For the crowdsourcing subsystem, we used Amazon Mechanical Turk (MTurk) and designed a custom HIT using MTurk's provided APIs. In the next step, we sent the results of our QBE system to be re-ranked by crowdworkers. We conducted experiment and measured the distance between the new ranked list and the gold standard (provided by Corel-Princeton) and our results showed that the performance of the system increased significantly. We conclude that a hybrid human–machine CBIR system can take advantage of a computational algorithm to increase speed and reduce the cost of the system and of crowdsourcing to increase the accuracy and performance.

6.1.2 User Interface Design

We noted that we used MTurk for our crowdsourcing task and designed our own HITs using HTML (HyperText Markup Language) pages. MTurk presents all HITs in a limited

small area inside its main interface (iframe) and this limited visual space highlights the importance of a well-designed user interface (UI). Previous research has focused on the cognitive load aspects of UI design in software design (Juristo et al. 2007; Seuken et al. 2010; Liu & Ma 2010), and on the effects of cognitive load and its integration with human–computer interaction (HCI) concepts on UI design (Huang et al. 2009; Antle & Wise 2013). There are not enough studies on the effects of UI design on crowdsourcing using the MTurk platform (Khanna et al. 2010).

We hypothesized that a poor UI design in MTurk crowdsourcing tasks reduces the performance of crowd-produced results. In addition, a poorly designed UI reduces crowdworkers' willingness to accept the task and results in higher execution time for the crowdsourced task. To test our hypothesis, we designed three UIs with different levels of cognitive load (Rank, Sort and Rate) for an image-ranking task and carried out experiments. Our results showed that crowd responses for the tasks using Rate UI design, which we believed has a lower cognitive load, have a higher performance than for the other two UI designs.

In another experiment, we designed two UIs with different complexity levels for an image classification task to study the effect on the task execution time. Our results confirmed that crowdworkers do not select HITs with complex UI designs and therefore the total execution time of the crowdsourced task increases.

6.2 Research Implications

Computers are very good at complex calculations and storing data. They are also good at storing and retrieving data: a computer never forgets facts or exaggerates. However, computers are not good at everything and when adaptation to change, observation and learning from experiments, or making judgments is needed, humans seem to perform better. Crowdsourcing provides a fast, easily accessible and efficient way of benefiting from the power of humans. Designing a hybrid human–machine system can take advantage of computers' calculation power and humans' judgment to provide higher performance results.

The outcomes of our research confirmed that a hybrid human–machine CBIR system is more powerful than humans or machines alone and that a number of applications can benefit from the performance of our proposed CBIR system. Crime investigators, gallery and museum owners, or biologists can use such a hybrid system to filter pictures and videos quickly and efficiently. However, the copyright and confidentiality of material being published in crowdsourcing platforms are very important factors and should be considered carefully by designers of these systems.

There are a number of research studies on hybrid human–machine systems and some systems have been designed (e.g. CrowdDB for query processing (Franklin et al. 2011), Soylent for text editing (Bernstein et al. 2010), VizWiz image search for blind people (Bigham et al. 2010)), but there are still some fields of study in which humans defeat computers in performance and where a hybrid system can act more efficiently. Translating from one language to another is one of the tasks that computers can perform

very fast but with very low performance. Using a crowdsourcing method, results from a machine translator can be verified by humans and improve the accuracy of translated text. Urban traffic monitoring, Web crawling and city map correction are other examples where a hybrid human–machine system can provide a better solution than traditional computing.

6.3 Research Limitations

This study faced a number of limitations and we suggest future work to resolve them and extend the findings.

As described in Chapter 4, we designed our CBIR systems using a single local feature detector/descriptor for each of SIFT, SURF, SURF128 and ORB to simplify the system implementation. Using more than one feature detector/descriptor in image search can improve the search performance and we suggest implementing a more complex and powerful CBIR system to improve the overall performance of hybrid human–machine CBIR system. In addition, we used a text search method with Euclidian distance for clustering, indexing and image retrieval. This method can be replaced with more sophisticated image search methods such as Earth Mover's Distance (EMD) (Rubner et al. 2000).

Considering our research goal which was to study the query-by-example (QBE) image similarity search, we used Corel-Princeton as our image dataset. The most important part of our study is measuring the performance of the QBE system by comparing it with a gold standard and the Corel-Princeton image dataset is the only dataset which provides

this gold standard. However, this dataset is limited in the number of images and also it is no longer available. We constructed the dataset with all the available images that we could purchase online and this reduced the size of our dataset. For future studies, we recommend designing a new dataset with the gold standard.

To perform crowdsourcing, we used Amazon Mechanical Turk (MTurk) for the microtasking platform. MTurk is not very flexible when it comes to selecting and controlling crowdworkers. The only option to filter crowdworkers is through Qualification Types and qualification tests which is not very effective and also can discourage workers from accepting the task. In addition, unlike CrowdFlower, MTurk does not provide any quality measure tool for crowd-provided responses and our only option for quality control was built-in mechanisms that we added to our design (time stamp, controlled data entry). Further studies can be conducted on different platforms and on the quality control of crowd-provided responses.

6.4 Suggestions for Future Work

This research has answered some questions regarding the effectiveness of associating crowds with computational algorithms and has provided a baseline for future studies.

In this research, we studied a few different UI designs for an image ranking and classification task and their effects on crowdworkers' performance in MTurk. While we analysed only the cognitive load aspects of UI designs, an extended research similar to usability design in software can reveal facts to improve the performance of crowd-generated responses. Furthermore, the effects of UI design on other types of

crowdsourcing tasks (e.g. text manipulation, image annotation, etc.) can be studied in depth. In addition, we analysed the cognitive load of each UI based on cognitive load theory principles and practical measurement of the actual cognitive load of UIs can confirm or reject our assumptions.

We conducted all of our experiments on MTurk as the microtasking platform. Other platforms such as CrowdFlower and Microtask.com have different characteristics and specifications to MTurk and comprehensive research on these platforms for different crowdsourcing tasks can build a framework for choosing a platform based on the crowdsourcing task.

References

Antle, A. & Wise, A., 2013. Getting down to details: Using theories of cognition and learning to inform tangible user interface design. *Interacting with Computers*, 25(1).

Aquino, J., Nine jobs that humans may lose to robots. Available at: http://www.nbcnews.com/id/42183592/ns/business-careers/t/nine-jobs-humans-may-lose-robots/#.UykYefmSyiw [Accessed March 19, 2014].

Archak, N., 2010. Money, glory and cheap talk. In *Proceedings of the 19th international conference on World wide web - WWW '10*. ACM Press, p. 21.

Ayres, P. & Sweller, J., 2005. The Split-Attention Principle in Multimedia Learning. In *The Cambridge handbook of multimedia learning*. Cambridge University Press, pp. 135–146.

Bannert, M., 2002. Managing cognitive load—recent trends in cognitive load theory. *Learning and Instruction*, 12(1), pp.139–146.

Bay, H., Tuytelaars, T. & Gool, L. Van, 2006. SURF : Speeded Up Robust Features. In *Proceedings of the ninth European Conference on Computer Vision*. pp. 404–417.

Bernstein, M. et al., 2012. Analytic Methods For Optimizing Realtime Crowdsourcing. In *Preceeding of Collective Intelligence conference -CI 2012*.

Bernstein, M., 2011. Crowds in Two Seconds: Enabling Realtime Crowd-Powered Interfaces. In *Proceedings of the 24th annual ACM symposium on User interface software and technology -UIST '11*. ACM Press, pp. 33–42.

Bernstein, M. et al., 2010. Soylent : A Word Processor with a Crowd Inside. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. pp. 313–322.

Bigham, J.P. et al., 2010. VizWiz: Nearly Real-time Answers to Visual Questions. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology - UIST '10*. ACM Press, p. 333.

Blanco, R. et al., 2011. Repeatable and reliable search system evaluation using crowdsourcing. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information - SIGIR '11*. ACM Press, p. 923.

Bozzon, A., Brambilla, M. & Ceri, S., 2012. Answering search queries with CrowdSearcher. In *Proceedings of the 21st international conference on World Wide Web - WWW '12*. ACM Press, p. 1009.

Buhrmester, M., Kwang, T. & Gosling, S.D., 2011. Amazon's Mechanical Turk: A New Source of Inexpensive, Yet High-Quality, Data? *Perspectives on Psychological Science*, 6(1), pp.3–5.

Burn-Callander, R., 2013. Artificial intelligence "will take the place of humans within five years." Available at: http://www.telegraph.co.uk/finance/businessclub/technology/10274420/Artificial-intelligence-will-take-the-place-of-humans-within-five-years.html [Accessed March 19, 2014].

Carterette, B., 2009. On rank correlation and the distance between rankings. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '09*. Boston, Massachusetts, USA: ACM Press, p. 436.

Chang, N.S. & Fu, K.S., 1979. A Relational Database System for Images. *Technical report TR-EE 79-28*.

Chang, S.-F. et al., 1997. Visual information retrieval from large distributed online repositories. *Communications of the ACM*, 40(12), pp.63–71.

Chapelle, O. et al., 2009. Expected reciprocal rank for graded relevance. In *Proceeding of the 18th ACM conference on Information and knowledge management - CIKM '09*. ACM Press, p. 621.

Chilton, L.B., Horton, J.J. & Miller, R., 2011. Task Search in a Human Computation Market Categories and Subject Descriptors. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. pp. 1–9.

Copeland, A., 1951. A "reasonable" social welfare function. *Seminar on Mathematics in Social Sciences*.

Davis, J., 2011. From Crowdsourcing to Crowdservicing. *IEEE Internet Computing*, 15(3), pp.92–94.

Diaconis, P., 1988. *Group representations in probability and statistics* S. S. Gupta, ed., Hayward, CA: Institute of Mathematical Statistics.

Diaconis, P. & Graham, R.L., 1977. Spearman's Footrule as a Measure of Disarray. *Journal of Royal Statistical Society*, pp.262–268.

Doan, A., Ramakrishnan, R. & Halevy, A.Y., 2011. Crowdsourcing systems on the World-Wide Web. *Communications of the ACM*, 54(4), p.86.

Dontcheva, M., Gerber, E. & Lewis, S., 2011. Crowdsourcing and Creativity. In *Proceedings of CHI Conference on Human Factors in Computing Systems*. ACM Press, pp. 7–10.

Downs, J.S. et al., 2010. Are your participants gaming the system? In *Proceedings of the 28th international conference on Human factors in computing systems - CHI '10*. ACM Press, p. 2399.

Dwork, C. et al., 2001. Rank aggregation methods for the Web. In *Proceedings of the tenth international conference on World Wide Web - WWW '01*. ACM Press, pp. 613–622.

Eagle, N., 2009. txteagle: Mobile Crowdsourcing N. Aykin, ed. *Internationalization, Design and Global Development Lecture Notes in Computer Science*, 5623.

Eickhoff, C. & Vries, A.P. De, 2008. How Crowdsourcable is Your Task ? In *Workshop on Crowdsourcing for Search and Data Mining*. pp. 11–14.

Elson, J. et al., 2007. Asirra : A CAPTCHA that Exploits Interest-Aligned Manual Image Categorization. In *Proceedings of 14th ACM Conference on Computer and Communications Security (CCS)*.

EMarketer, 2014. Smartphone Users Worldwide Will Total 1.75 Billion in 2014. Available at: http://www.emarketer.com/Article/Smartphone-Users-Worldwide-Will-Total-175-Billion-2014/1010536 [Accessed July 11, 2014].

Erickson, L.B., 2012. Leveraging the crowd as a source of innovation. In *Proceedings of the 50th annual conference on Computers and People Research - SIGMIS-CPR '12*. ACM Press, p. 91.

Erickson, L.B., Petrick, I. & Trauth, E.M., 2012. Organizational uses of the crowd. In *Proceedings of the 50th annual conference on Computers and People Research - SIGMIS-CPR '12*. ACM Press, p. 155.

Erry, C., Ginns, P. & Pitts, C., 2006. Cognitive load theory and user interface design: Making software easy to learn and use. Available at: http://www.ptg-global.com/PDFArticles/Cognitive load theory and user interface design Part 1 v1.0.pdf [Accessed March 8, 2013].

Fagin, R., Kumar, R. & Mahdian, M., 2004. Comparing and Aggregating Rankings with Ties. In *Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. pp. 47–58.

Fagin, R., Kumar, R. & Sivakumar, D., 2003. Efficient similarity search and classification via rank aggregation. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*. pp. 301–312.

Faridani, S., Hartmann, B. & Ipeirotis, P.G., 2011. What ' s the Right Price? Pricing Tasks for Finishing on Time. In *Human Computation: Papers from the 2011 AAAI Workshop*. pp. 26–31.

Feinberg, S. & Murphy, M., 2000. Applying Cognitive Load Theory to the Design of Web-Based Instruction. In *Proceedings of 2000 Joint IEEE International and 18th Annual Conference on Computer Documentation (IPCC/SIGDOC 2000)*. IEEE, pp. 353–360.

Franklin, M.J. et al., 2011. CrowdDB: Answering Queries with Crowdsourcing. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data SIGMOD '11*. ACM Press, pp. 61–72.

Fritz, S. et al., 2009. Geo-Wiki.Org: The Use of Crowdsourcing to Improve Global Land Cover. *Remote Sensing*, 1(3), pp.345–354.

Geiger, D. et al., 2011. Managing the Crowd : Towards a Taxonomy of Crowdsourcing Processes. In *Proceedings of 17th Americas Conference on Information Systems*. Detroit, Michigan, pp. 1–11.

Goldman, M., Little, G. & Miller, R.C., 2011. Real-time collaborative coding in a web IDE. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. New York, New York, USA: ACM Press, p. 155. Available at: http://dl.acm.org/citation.cfm?doid=2047196.2047215.

Gosling, S.D. et al., 2000. Should we trust web-based studies? A comparative analysis of six preconceptions about internet questionnaires. *The American psychologist*, 59(2), pp.93–104.

Grauman, K., 2010. Efficiently Searching for Similar Images. *Communication of the ACM*, 53(6), pp.84–94.

Grauman, K., 2007. The Pyramid Match Kernel : Efficient Learning with Sets of Features. *Journal of Machine Learning Research*, 8, pp.725–760.

Grauman, K. & Darrell, T., 2005. The Pyramid Match Kernel : Discriminative Classification with Sets of Image Features. In *Proceeding of the IEEE International Conference on Computer Vision*. Beijing, China, pp. 1458 – 1465.

Gupta, A. et al., 2012. mClerk: Enabling Mobile Crowdsourcing in Developing Regions. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*. ACM Press, p. 1843.

Gupta, A. & Jain, R., 1997. Visual Information Retrieval. *Communications of the ACM*, 40(5), pp.70–79.

Harris, C.G., 2011. You ' re Hired ! An Examination of Crowdsourcing Incentive Models in Human Resource Tasks. In *Proceedings of WSDM 2011 Workshop on Crowdsourcing for Search and Data Mining*. pp. 15–18.

Heer, J. & Bostock, M., 2010. Crowdsourcing Graphical Perception: Using Mechanical Turk to Assess Visualization Design. In *Proceedings of the 28th Annual Chi Conference on Human Factors in Computing Systems*. ACM Press, pp. 203–212.

Hirth, M., Hoßfeld, T. & Tran-Gia, P., 2011. Cost-Optimal Validation Mechanisms and Cheat-Detection for Crowdsourcing Platforms. In *Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. Ieee, pp. 316–321.

Howe, J., 2006. Crowdsourcing: A Definition. Available at: http://www.crowdsourcing.typepad.com/cs/2006/06/crowdsourcing_a.html [Accessed August 2, 2013].

Hu, N., Pavlou, P.A. & Zhang, J., 2006. Can Online Reviews Reveal a Product ' s True Quality ? Empirical Findings and Analytical Modeling of Online Word-of-Mouth Communication. In *Proceedings of the 7th ACM conference on Electronic commerce Pages*. pp. 324–330.

Huang, W.C. (Darren), Trotman, A. & Geva, S., 2009. A Virtual Evaluation Forum for Cross Language Link Discovery. In *Proceedings of the SIGIR 2009 Workshop on the Future of IR Evaluation*. pp. 19–20.

Ipeirotis, P.G., 2010. Demographics of Mechanical Turk. *NYU Center for Digital Economy Research Working Paper CeDER*.

Jansen, B.J. & Spink, A., 2003. An Analysis of Web Documents Retrieved and Viewed. In *Proceedings of 4th International Conference on Internet Computing*. pp. 65–69.

Jansen, B.J., Spink, A. & Saracevic, T., 2000. Real life, real users, and real needs: a study and analysis of user queries on the web. *Information Processing and Management*, 36, pp.207–227.

Jensen, T., 2011. 2nd Page Rankings: Your the #1 Looser. Available at: http://www.gravitateonline.com/google-search/2nd-place-1st-place-loser-seriously [Accessed April 11, 2014].

Jing, Y. & Baluja, S., 2008. Pagerank for product image search. In *Proceeding of the 17th international conference on World Wide Web - WWW '08*. New York, New York, USA: ACM Press, p. 307. Available at: http://dl.acm.org/citation.cfm?id=1367497.1367540 [Accessed August 23, 2012].

Juristo, N., Moreno, A.M. & Sanchez-Segura, M.-I., 2007. Analysing the impact of usability on software design. *Journal of Systems and Software*, 80(9), pp.1506–1516.

Kazai, G., 2011. In search of quality in crowdsourcing for search engine evaluation. In *Proceedings of the 33rd European conference on Advances in information retrieval*. pp. 165–176.

Ke, Y. & Sukthankar, R., 2004. PCA-SIFT: a more distinctive representation for local image descriptors. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Ieee, pp. 506–513.

Kendall, M., 1938. A New Measure of Rank Correlation. *Biometrika*, 30, pp.81–89.

Khanna, S. et al., 2010. Evaluating and improving the usability of Mechanical Turk for low-income workers in India. In *Proceedings of the First ACM Symposium on Computing for Development - ACM DEV '10*. ACM Press, p. 12.

Kittur, A. et al., 2011. CrowdForge: Crowdsourcing Complex Work. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM Press, p. 43.

Kittur, A., Chi, E.H. & Suh, B., 2008. Crowdsourcing user studies with Mechanical Turk. In *Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems*. ACM Press, p. 453.

Kopeck´, J. & Domingue, J., 2012. ParkJam : crowdsourcing parking availability information with linked data. In *9th Extended Semantic Web Conference (ESWC 2012)*.

Kulkarni, A., Can, M. & Hartmann, B., 2012. Collaboratively crowdsourcing workflows with turkomatic. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*. ACM Press, p. 1003.

Kulkarni, A., Can, M. & Hartmann, B., 2011. Turkomatic : Automatic Recursive Task and Workflow Design for Mechanical Turk. In *Extended Abstracts on Human Factors in Computing Systems*. pp. 2053–2058.

Kumar, R. & Vassilvitskii, S., 2010. Generalized Distances between Rankings. In *Proceedings of the 19th international conference on World wide web*. ACM, pp. 571–579.

Law, E. & von Ahn, L., 2011. Human Computation. In *Synthesis Lectures on Artificial Intelligence and Machine Learning*. pp. 1–121.

Ledlie, J. et al., 2010. Crowd translator: On Building Localized Speech Recognizers through Micropayments. *ACM SIGOPS Operating Systems Review*, 43(4), p.84.

Little, G. et al., 2010a. Exploring iterative and parallel human computation processes. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*. ACM Press, p. 68.

Little, G. et al., 2010b. TurKit : Human Computation Algorithms on Mechanical Turk. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. pp. 57–66.

Little, G. et al., 2009. TurKit : Tools for Iterative Tasks on Mechanical Turk. In *IEEE Symposium on Visual Languages and Human-Centric Computing*. pp. 29–30.

Liu, H. & Ma, F., 2010. Research on Visual Elements of Web UI Design. In *Proceedings of IEEE 11th International Conference on Computer-Aided Industrial Design & Conceptual Design*. Yiwu, China, pp. 428–430.

Liu, X. et al., 2012. CDAS: a crowdsourcing data analytics system. *Proceedings of the VLDB Endowment*, 5(10), pp.1040–1051.

Liu, Y.-T. et al., 2007. Supervised rank aggregation. *Proceedings of the 16th international conference on World Wide Web - WWW '07*, p.481. Available at: http://portal.acm.org/citation.cfm?doid=1242572.1242638.

Long, F., Zhang, H. & Feng, D.D., 2003. Fundamentals of Content-Based Image Retrieval. In *Multimedia Information Retrieval and Management*. Springer Berlin Heidelberg, pp. 1–26.

Lowe, D.G., 1999. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Ieee, pp. 1150–1157.

Loy, G. & Eklundh, J., 2006. *A Review of Benchmarking Content Based Image Retrieval*,

Marcus, A., Wu, E., Karger, D., et al., 2011. Crowdsourced Databases: Query Processing with People. In *Proceedings of the 5th Biennial Conference on Innovative Data Systems Research*. CIDR, pp. 211–214.

Marcus, A., Wu, E., Madden, S., et al., 2011. Human-powered sorts and joins. In *Proceedings of the VLDB Endowment*. pp. 13–24.

Mason, W. & Watts, D.J., 2009. Financial Incentives and the " Performance of Crowds ." In *Proceedings of the ACM SIGKDD Workshop on Human Computation*. ACM Press, pp. 77–85.

McAndrew, F.T., 1986. A Cross-Cultural Study of Recognition Thresholds for Facial Expressions of Emotion. *Journal of Cross-Cultural Psychology*, 17(2), pp.211–224.

McDuff, D., Kaliouby, R. el & Picard, R., 2011. Crowdsourced data collection of facial responses. In *Proceedings of the 13th international conference on multimodal interfaces*. ACM Press, p. 11.

McGlohon, M., Glance, N. & Reiter, Z., 2010. Star Quality: Aggregating Reviews to Rank Products and Merchants. In *Proceedings of Fourth International Conference on Weblogs and Social Media (ICWSM), AAAI (2010)*. AAAI, pp. 114–121.

Mikolajczyk, K. & Schmid, C., 2005. A Performance evaluation of local descriptors. *IEEE transactions on pattern analysis and machine intelligence*, 27(10), pp.1615–30.

Miller, B.N. et al., 2003. MovieLens Unplugged : Experiences with an Occasionally Connected Recommender System. In *Proceedings of the 8th international conference on Intelligent user interfaces*. ACM, pp. 263–266.

Miller, G. a, 1956. The magical number seven, plus or minus two: some limits on our capacity for processing information. 1956. *Psychological review*, 101(2), pp.343–52.

Nowak, S. & Rüger, S., 2010. How reliable are annotations via crowdsourcing. In *Proceedings of the international conference on Multimedia information retrieval*. ACM Press, p. 557.

Oviatt, S., 2006. Human-Centered Design Meets Cognitive Load Theory: Designing Interfaces that Help People Think. In *Proceedings of the 14th annual ACM international conference on Multimedia*. ACM Press, pp. 871–880.

Paas, F.G.W.C. & Merrienboer, J.J.G. Van, 1994. Variability of Worked Examples and Transfer of Geometrical Problem-Solving Skills : A Cognitive-Load Approach c o. *Journal of Educational Psychology*, 86(1), pp.122–133.

Pai, D. & Davis, J., 2012. Wally – Crowd powered image matching on tablets. In *Proceedings of the First International Workshop on Crowdsourcing and Data Mining*. pp. 10–14.

Parameswaran, A. & Polyzotis, N., 2011. Answering Queries using Humans , Algorithms and Databases. In *In Fifth Biennial Conference on Innovative Data Systems Research*. pp. 106–166.

Prado, C. et al., 2013. Facial emotion recognition: a cross-cultural comparison of Chinese, Chinese living in Australia, and Anglo-Australians. *Motivation and Emotion*.

Quinn, A.J. & Bederson, B.B., 2011. Human Computation : A Survey and Taxonomy of a Growing Field. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. pp. 1403–1412.

Rahmani, R. et al., 2008. Localized Content Based Image Retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11), pp.1902–1912.

Rashtchian, C. et al., 2010. Collecting image annotations using Amazon's Mechanical Turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*. pp. 139–147.

Reis, H.M. et al., 2012. Towards Reducing Cognitive Load and Enhancing Usability through a Reduced Graphical User Interface for a Dynamic Geometry System: An Experimental Study. In *Proceedings of IEEE International Symposium on Multimedia*. IEEE, pp. 445–450.

Ricardo Baeza-Yates, B.R.-N., 1999. *Modern Information Retrieval*, ACM Press.

Ross, J. et al., 2010. Who are the Crowdworkers ? Shifting Demographics in Mechanical Turk. In *Extended Abstracts on Human Factors in Computing Systems.* pp. 2863–2872.

Rosten, E. & Drummond, T., 2006. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*. pp. 430–443.

Rublee, E. et al., 2011. ORB: An efficient alternative to SIFT or SURF. *2011 International Conference on Computer Vision*, pp.2564–2571. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6126544.

Rubner, Y., Tomasi, C. & Guibas, L.J., 2000. The Earth Mover ' s Distance as a Metric for Image Retrieval. *International Journal of Computer Vision*, 40(2), pp.99–121.

Rui, Y., Huang, T.S. & Chang, S.-F., 1999. Image Retrieval: Current Techniques, Promising Directions, and Open Issues. *Journal of Visual Communication and Image Representation*, 10(1), pp.39–62.

Russell, B.C. et al., 2007. LabelMe: A Database and Web-Based Tool for Image Annotation. *International Journal of Computer Vision*, 77(1-3), pp.157–173.

Schalekamp, F. & Zuylen, A. van, 1998. Rank Aggregation : Together We ' re Strong. In *Proceedings of 11th ALENEX*. pp. 38–51.

Schneider, D. et al., 2012. CSCWD : Five Characters in Search of Crowds. In *Proceedings of the 2012 IEEE 16th International Conference on Computer Supported Cooperative Work in Design*. pp. 634–641.

Seuken, S. et al., 2010. Hidden Markets : UI Design for a P2P Backup Application. In *CHI2010 : Market Models for Q&A Services*. Atlanta, Georgia, USA, pp. 315–324.

Shah, S. et al., 2011. CROWDSAFE: Crowd Sourcing of Crime Incidents and Safe Routing on Mobile Devices (Demo Paper). In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM Press, p. 521.

Silberman, M.S., Irani, L., Tomlinson, B., et al., 2010. Sellers' problems in human computation markets. In *International Conference on Knowledge Discovery and Data Mining (2010)*. pp. 18–21.

Silberman, M.S., Irani, L. & Ross, J., 2010. Ethics and tactics of professional crowdwork. *XRDS: Crossroads, The ACM Magazine for Students - Comp-YOU-Ter*, pp.39–43.

Sivic, J. & Zisserman, A., 2003. Video Google: a text retrieval approach to object matching in videos. In *Proceedings Ninth IEEE International Conference on Computer Vision*. IEEE, pp. 1470–1477.

Snoek, C.G.M. et al., 2010. Crowdsourcing rock n' roll multimedia retrieval. In *Proceedings of the international conference on Multimedia*. ACM Press, pp. 1535–1538.

Snow, R. et al., 2008. Cheap and fast-but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pp. 254–263.

Spink, A. et al., 2002. From E-sex to E-commerce: Web Search Changes. *IEEE Computer*, 35, pp.107–111.

Stewart, O., Huerta, J.M. & Sader, M., 2009. Designing crowdsourcing community for the enterprise. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*. ACM Press, p. 50.

Stewart, O., Lubensky, D. & Huerta, J.M., 2010. Crowdsourcing participation inequality:A SCOUT Model for the Enterprise Domain. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*. ACM Press, p. 30.

Sweller, J., 1988. Cognitive Load During Problem Solving: Effects on Learning. *Cognitive Science*, 12, pp.257–259.

Sweller, J., Merrienboer, J.J.G. Van & Paas, F.G.W.C., 1998. Cognitive Architecture and Instructional Design. *Educational Psychology Review*, 10(3), pp.251–296.

Tokarchuk, O., Cuel, R. & Zamarian, M., 2012. A framework to analyze crowd labor and design a proper set of incentives for humans in the loop. *IEEE Internet Computing*, (Special Issue), pp.45–51.

Tran, P., Pyramid Matching Using SURF and SIFT Descriptors for SVM Classification 2 Related Work 1 Introduction 3 Current Experimental Results.

Urbano, J. et al., 2010. Crowdsourcing Preference Judgments for Evaluation of Music Similarity Tasks. In *Proceedings of the SIGIR 2010 Workshop on Crowdsourcing for Search Evaluation*. Geneva,Switzerland, pp. 9–16.

Vailaya, a et al., 2001. Image classification for content-based indexing. *IEEE transactions on image processing*, 10(1), pp.117–30.

Vedaldi, A. & Fulkerson, B., 2010. VLFeat - An open and portable library of computer vision algorithms. In *Proceedings of the international conference on Multimedia*. Firenze, Italy, pp. 1469–1472.

Venetis, P. et al., 2012. Max algorithms in crowdsourcing environments. In *Proceedings of the 21st international conference on World Wide Web*. New York, New York, USA: ACM Press, p. 989.

VonAhn, L., 2006. Games with a Purpose. *Computer-Invisible Computing*, 39(6), pp.92–94.

VonAhn, L. et al., 2008. reCAPTCHA: human-based character recognition via Web security measures. *Science (New York, N.Y.)*, 321(5895), pp.1465–8.

VonAhn, L., Blum, M. & John, L., 2004. Telling humans and computers apart automatically. *Communication of ACM*, 47(2).

VonAhn, L. & Dabbish, L., 2004. Labeling images with a computer game. In *Proceedings of the 2004 conference on Human factors in computing systems - CHI '04*. ACM Press, pp. 319–326.

Vuurens, J., Vries, A.P. De & Eickhoff, C., 2011. How Much Spam Can You Take ? An Analysis of Crowdsourcing Results to Increase Accuracy. In *Proceedings of the SIGIR 2011 Workshop on Crowdsourcing for Information Retrieval*. pp. 66–75.

Wang, J. et al., 2012. CrowdER : Crowdsourcing Entity Resolution. In *Proceedings of the VLDB Endowment*. Istanbul, Turkey, pp. 1483–1494.

Wightman, D., 2010. Crowdsourcing Human-Based Computation. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries - NordiCHI '10*. ACM Press, pp. 551–560.

Williams, J.D. et al., 2011. Crowd-sourcing for difficult transcription of speech. In *IEEE Workshop on Automatic Speech Recognition and Understanding*. Hawaii, USA, pp. 535–540.

Xue, S. et al., 2012. Crowd sourcing memory colors for image enhancement. In *ACM SIGGRAPH 2012 Talks*. ACM Press;NG, p. 1.

Yan, T., Kumar, V. & Ganesan, D., 2010. CrowdSearch: Exploiting Crowds for Accurate Real-time Image Search on Mobile Phones. In *Proceedings of the 8th international conference on Mobile systems, applications, and services - MobiSys '10*. ACM Press, pp. 77–90.

Yuen, M.-C., King, I. & Leung, K.-S., 2011. A Survey of Crowdsourcing Systems. In *IEEE Third International Conference on Privacy Security Risk and Trust and 2011 IEEE Third International Conference on Social Computing*. IEEE, pp. 766–773.

Zhang, H. et al., 2011. Crowdsourcing General Computation. In *ACM CHI 2011 Workshop on Crowdsourcing and Human Computation*. pp. 1–5.

Zhou, X.S. & Huang, T.S., 2000. CBIR : From Low-Level Features to High-Level Semantics. In *Proceeding of SPIE Image and Video Communication and Processing*. San Jose, CA, pp. 24–28.