McGill University

Doctoral Thesis

Data visualization and crowdsourcing approaches for complex data analysis

---

Author:

Alexander Butyaev

Supervisor:

Dr. Jérôme Waldispühl

A thesis submitted to McGill University in partial fulfillment of the requirements of the degree of Doctor of Philosophy

School of Computer Science

McGill University, Montreal

August 3, 2019

© Alexander Butyaev 2019

Declaration of Authorship

I, Alexander BUTYAEV, declare that this thesis titled, "Data visualization and crowdsourcing approaches for complex data analysis" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

_____

Date: August 3, 2019

_____

"After a bunch of time, you realize that something is not quite right. However, it takes just a few seconds to realize that that bunch of time was necessary to make that painful realization."

Dr David Becerra

MCGILL UNIVERSITY

Abstract

School of Computer Science

McGill University, Montreal

Doctor of Philosophy

Data visualization and crowdsourcing approaches for complex data analysis

by Alexander BUTYAEV

State-of-the-art machine learning algorithms (e.g., convolution neural networks, long short term memory recurrent networks) have allowed human-driven analysis to become completely automated. However, there is an increasing need for human supervision and intuition when automating these analyses. The objective of this thesis is to analyze crowdsourcing and human computation in Human-Computer Interaction (HCI) systems, which engage human participants to efficiently solve problems that are hard for computers but intuitive for humans. This thesis addresses questions related to problems of data decomposition, visualization, and interpretation from the perspective of HCI. Solution assembly strategies, as well as user motivation, will also be explored.

First, to inform the design of HCI systems, data visualization techniques are explored. The goal of this project is to design HCI systems that are convenient for humans and allow data to be easily interpreted. Genomics was a natural choice for this study as the field explore complex data sets (e.g., Hi-C and Experimental ChIP-Sequencing data) that require advance visualization to allow a user to interpret the data. Here, we describe 3DGB, an interactive web-based 3D genome browser that is

capable of visualizing and exploring 3D genome structures. 3DGB provides the user with a simplified tool set that improves the interpretability for complex 3D genomic data (like Hi-C), regardless of a user's background knowledge in genomics.

Next, we address the problem of large-scale human collaboration by designing and implementing a multiplayer, real-time simulation, called the 'Market game'. This game-with-a-purpose creates a market simulation to investigate a competitive, yet collaborative, environment for players to solve classical graph theory problem. 'Market game' simulations have allowed us to develop multiple tools that promote and enhance collaboration within a large-scale HCI system.

Finally, an application of HCI techniques in cluster analysis is explored. Here, a mobile online human computing game, called 'Colony B', is used to collect human annotations of 2D clustering problems. Our work then characterizes the collection of human-perceived 2D clusters relative to standard automated clustering techniques. The results of this experiment demonstrate the specificity of aggregated human annotations and show that these clusters are approximately the same quality as those produced by clustering algorithms. Colony B is designed to accommodate high dimensionality problems and allows us to address the curse of dimensionality. Finally, we propose two crowdsourcing multidimensional clustering algorithms that partially or entirely rely on aggregated human answers and benchmark them against state-of-the-art fully-automated clustering algorithms.

UNIVERSITÉ MCGILL

Abrégé

École d'Informatique

Université McGill, Montréal

Docteur en Philosophie

Methodes pour la visualisation de données et le crowdsourcing de l'analyse de données complexes

par Alexander BUTYAEV

Des algorithmes d'apprentissage automatique de pointe (réseaux de neurones de convolution, réseaux récurrents à mémoire à court et long terme, par exemple) ont permis à l'analyse de données manuelle de devenir complètement automatisée. Cependant, on observe un besoin croissant de supervision humaine et d'intuition lors de l'automatisation de ces analyses. L'objectif de cette thèse est d'analyser le crowdsourcing et le calcul humain dans les systèmes d'interaction homme-machine (IHM), qui engagent les participants humains à résoudre efficacement des problèmes difficiles pour les ordinateurs mais intuitifs pour les humains. Cette thèse aborde des questions liées aux problèmes de décomposition, de visualisation et d'interprétation des données du point de vue de HCI. Les stratégies d'assemblage de solutions, ainsi que la motivation des utilisateurs, seront également explorées.

Tout d'abord, pour se familiariser avec la conception des systèmes HCI, nous avons exploré les techniques de visualisation des données. Le but de ce projet est de concevoir des systèmes HCI qui conviennent aux humains et permettent d'interpréter facilement les données. La génomique était un choix naturel pour cette étude, car

ce domaine couvre des ensembles de données complexes (par exemple, des données de séquençage PIP et Hi-C expérimentales) qui nécessitent une visualisation avancée pour permettre aux utilisateurs d'interpréter les données. Nous décrivons ici 3DGB, un navigateur Web interactif pour le génome 3D, capable de visualiser et d'explorer les structures du génome 3D. 3DGB fournit aux utilisateurs un ensemble d'outils simplifiés qui améliorent l'interprétation des données génomiques 3D complexes (telles que Hi-C), quelles que soient les connaissances de base de l'utilisateur en génomique.

Nous abordons ensuite le problème de la collaboration humaine à grande échelle en concevant et en mettant en œuvre une simulation multijoueur en temps réel, appelée le «jeu du marché». Ce jeu de calcul (GWAP) crée une simulation de marché pour étudier un environnement concurrentiel, mais collaboratif, permettant aux joueurs de résoudre un problème classique de la théorie des graphes. Les simulations de «jeux de marché» nous ont permis de développer de multiples outils qui favorisent et améliorent la collaboration au sein d'un système HCI à grande échelle.

Enfin, nous finissons par une application des techniques HCI en analyse par grappe. Ici, un jeu mobile en ligne d'informatique humaine, appelé «Colony B», est utilisé pour collecter des annotations, écrites par l'homme, de problèmes de classification 2D. Notre travail caractérise ensuite la collection de clusters 2D perçus par l'homme par rapport aux techniques de clustering automatisées standards. Les résultats de cette expérience démontrent la spécificité des annotations humaines agrégées et montrent que ces grappes ont approximativement la même qualité que celles produites par les algorithmes de groupement. Colony B est conçu pour prendre en charge les problèmes de haute dimensionnalité et nous permet de faire face à la malédiction de la dimensionnalité. En conclusion, nous proposons deux algorithmes de clustering multidimensionnels en crowdsourcing qui reposent partiellement ou entièrement sur des réponses humaines agrégées et les comparons à des algorithmes de clustering de pointe entièrement automatisés.

Acknowledgements

If I have been asked to think of a single word to describe the last 5 years, it would be a very long word with no spaces in between and likely make a little sense for a reader. However, the closest synonym would be the 'gratitude' because no thesis would exist without the constant support from my family and friends, who have become my family for this period of my life.

I can't express my thankfulness to my parents who, despite the distance, were always near encouraging me to dream on. It is impossible to imagine this journey without my wife Lily, the greatest source of motivation and encouragement. We will conquer the world one day... or at least travel.

I am also grateful to all the current and former members of the Waldispuhl and Blanchette research groups. I would love to particularly thank David and Chris for all the advise, support, knowledge, experience, for hours of fun even during the darkest Montreal winter days. I owe my sincere gratitude to all the members of the labs Olivier, Vladimir, Mohammed I and II, Ayrin, Rola, Antoine, Dilmi, Chrisostomos, Carlos, Roman, James, Faizy, Zhang, Navin, Pouriya, Maia, Samy, Julie, Eliot, Vincent, and many more that I forgot but appreciate nonetheless.

I would like to thank my volleyball team for a challenge, an unforgettable experience, and great advice to jump on any opportunity, literally and figuratively.

I am also very thankful to my supervisor Professor Jérôme Waldispühl for excellent research opportunities, his optimism, invaluable advice, and, of course, for dealing with all bad and good news. I would also like to thank Professor Mathieu Blanchette for his help, advice, and productive meetings.

Finally, I am greatly thankful to my PhD oral defence committee: Prof. Yue Li, Prof. Reihaneh Rabbany, Prof. David Meger, Prof. Abdoulaye Baniré Diallo and Prof. Edith Law, for finding time to read my thesis, and helping me better understand my research.

My thesis discusses the crowdsourcing topic however should have been titled as a crowdsourcing work itself. We all did that.

Thank you.

Contents

List of Figures

List of Tables

List of Abbreviations

3DBG	**3** D**ata**B**ase** G**enome**
3DGB	**3D** G**enome** B**rowser**
4DN	**4D** N**ucleome**
acsCLIQUE	**A**verage **C**luster **S**ize **CLIQUE**
AGP	**A**merican **G**ut **P**roject
AKDE	**A**daptive **K**ernel **D**ensity **E**stimation
AMT	**A**mazon **M**echanical **T**urk
AP	**A**ffinity **P**ropagation
asCLIQUE	**A**ll **S**olutions **CLIQUE**
AUC	**A**rea **U**nder the **C**urve
BIRCH	**B**alanced **I**terative **R**educing and **C**lustering using **H**ierarchies
CAPTCHA	**C**ompletely **A**utomated **P**ublic **T**uring test to tell **C**omputers and **H**umans **A**part
ChIA-PET	**Ch**romatin **I**nteraction **A**nalysis by **P**aired-**E**nd **T**ag Sequencing
CHiCP	**C**apture **Hi-C P**lotter
ChIPseq	**Ch**romatin **I**mmuno**p**recipitation **seq**uencing
CLARANS	**C**lustering **LAR**ge **A**pplication upo**n** randomized **s**earch
CLARA	**C**lustering **LAR**ge **A**pplication
CLIQUE	**CL**ustering **I**n **QUE**ue
ClOCworks	**C**lustering **O**f **C**rowd-sourced net**works**
confCLIQUE	**Conf**idence **CLIQUE**
consCLIQUE	**Cons**ervation **CLIQUE**
CURE	**C**lustering **U**sing **RE**presentatives
DBSCAN	**D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise

ENCODE	**ENC**yclopedia **O**f **D**NA **E**lements
GMM	**G**aussian **M**ixture **M**odel
GWAS	**G**enome-**W**ide **A**ssociation **S**tudy
HCI	**H**uman **C**omputer **I**nteraction
Hi-C	**Hi**gh-throughput **3C**apture
MDL	**M**inimal **D**escription **L**ength
OCR	**O**ptical **C**haracter **R**ecognition
OPTICS	**O**rdering **P**oints **T**o **I**dentify the **C**lustering **S**tructure
OSM	**O**pen**S**treet**M**aps
OS	**O**peration **S**ystem
PCA	**P**rincipal **C**omponent **A**nalysis
PCoA	**P**rincipal **Co**ordinate **A**nalysis
SBM	**S**tochastic **B**lock **M**odel
SNP	**S**ingle **N**ucleotide **P**olymorphism
SVM	**S**upport **V**ector **M**achines
TAD	**T**opologically **A**ssociating **D**omain
TFBS	**T**ranscription **F**actors **B**inding **S**ites
UCSC	**U**niversity of **C**alifornia **S**anta **C**ruz
VRD	**V**oice **R**ecognition **D**ataset
VR	**V**irtual **R**eality
WashU	**Wash**ington **U**niversity in St. Louis

To all those who helped me here and there. . . or just here. . . or just helped me. . . or at least wanted to do so. . .

Chapter 1

Introduction to Human Computer Interface

Alexander Butyaev [1], Jérôme Waldispühl [1]

[1] School of Computer Science, McGill University, Montréal, Québec, Canada

1.1 Introduction

Over 50% of the human brain neural tissue and more than 2/3 of the electrical activity is dedicated to sight and visual perception of information [1]. Thus, the visual representation of information is essential for human comprehension.

Multiple models were proposed to study impact of visual representation. Card et al. [2] proposed the Human Processor Model (HPM) that considers a person as an active processor of data due to his or her ability to perceive, process, organize, store, and also produce information. The HPM model introduces simplistic relations between long-term and working memory; auditory and visual image storage, where perceptual memories will be stored, while encoded associated information will be moved to the working memory section.

Rolls et al. [3] presented the neuronal mechanism that creates new associations between visual stimuli and taste. While much work was dedicated to exploring the reward and punishment system explaining an association between human behavior vs. taste and/or smell [4, 5], Eng et al. [6] showed that visual working memory is triggered by visual stimuli and degrades with its complexity. This study by Eng et al. highlights the importance of visual representation of data in a simplistic way for activating and engaging human working memory.

Although the effect of simplistic visual representation was scientifically observed in 2005 [6], the idea of simplicity was proposed long before. This idea was first presented as a concept of Graphical User Interfaces (GUI) [7] and, afterward, it was adopted as a key concept in the field of Human-Computer Interaction (HCI) [8].

HCI is a field of computational/psychological research that focuses on human computer communication (i.e., the interface) [9]. Wickents et al. [10] described multiple HCI principles that enhance the usability of a system. These principles cover visual interactivity, legibility, and simplicity with respect to the system and propose minimizing the use of human working memory by introducing external, sometimes redundant, info-related visuals.

The thesis touches on multiple aspects of the HCI field. First, we describe the major work in the area of Crowdsourcing and Human computing techniques, a well-established strategy to engage broad audience of regular web and mobile consumers for solving hard for a computer problem [11]. We review various applications of the HCI systems with an emphasize on crowd effort to contribute to specific computational tasks. The major research in HCI utilizes human computing and crowdsourcing techniques on a semantic understanding problem, which still can be ambiguous for a computer program and often requires specific criterion for items comparison (e.g., data labelling and annotation, object grouping, ranking, etc). In particular, this thesis investigates capabilities of a human worker in simple two-dimensional as well as high dimensional data clustering. Thus, we review major approaches in the topic of data clustering with the known and commonly used algorithms.

Moreover, the recent development in bio-technologies (e.g., 3C-techniques) enabled researchers to generate large amounts of complex data [12]. Although, the generation process of data becomes less an issue, its interpretation presents a challenge for researchers. Complexity of genomic data and numerous supplementary parameters perplex the visualization model that should also follow the simplicity rule for improved manual data analysis [13]. The thesis touches on genomic data visualization topic. Thus, at last, we review most relevant work in the area.

1.2 Human Computing and Crowdsourcing

Despite the recent advances in algorithms, technologies, and growth of computational resources, human intelligence is still required for many tasks that are still difficult for computers [11]. Luis von Ahn et al. proposed ESP Game that uses computational power of human to label images [14]. It identified the area where a human cannot be replaced yet and initiated rapid development of various systems that employ collective intelligence.

1.2.1 Shared Knowledge Systems

Shared Knowledge systems are among the most broadly used crowdsourcing platforms [15, 16, 17]. Wikipedia [18] is an online encyclopedia that utilizes an open collaboration model to create one of the largest multilingual knowledge databases. Following the peer production model used in Wikipedia, Haklay et al. [19] proposed the OpenStreetMap (OSM), the first community-generated street maps. A similar approach was used by TripAdvisor [20] to address the availability of travel-related consumer knowledge. In contrast, StackOverflow is more specialized crowdsourcing system that targets mainly applied computer science community [21]. It is a programming "Question-Answer" web service that accumulates advice, solutions, and best practices for coding challenges.

It is also interesting that shared knowledge concept was used in fiction writing to create a shared creative space for wiki-novel writing [22]. However, later, the experiment was announced to be unsuccessful due to a lack of control.

1.2.2 Crowdsourcing Marketplaces

Amazon Mechanical Turk (AMT) https://www.mturk.com/ is one of the first web services (founded in 2005) that started to provide a great base of crowd workers for solving microtasks regardless of its origin [23, 24]. A lot of research work was conducted using AMT. Heer et al. explored human graphical perception in assessing visualization design [25]. Alonso et al. showed how crowdsourcing could be useful as a relevance evaluation system [26]. Ott et al. address the problem of deceptive opinion (spam) [27]. Shank suggested multiple applications of crowdsourcing in sociology research [28].

Natural language annotation is also a very popular research topic on the AMT. Snow et al. [29] explored the affect recognition, word similarity, and word sense disambiguation using the crowdsourcing effort, and showed that AMT users produce labels comparable with expert gold standard annotations. Marge et al. [30] examined

AMT in the context of human speech recognition tasks. Mohammad et al. built a dictionary of phrases and words with emotional meaning [31]. In general, all the authors found that collective intelligence gave quite accurate results.

There is a number of alternatives for AMT: Crowdcrafting (available until April,1 2019) [32], Microworkers [33], CrowdFlower [34] (renamed to Figure Eight inc.), RapidWorkers [35], MiniJobz [36], etc. While they introduce generic crowdsourcing platform, Microworkers, MiniJobz, and RapidWorkers allocate primarily marketing related micro-tasks. Peer et al. compared crowd involvement the platforms and identified AMT and CrowdFlower to be clear leaders among others [37].

Most of the works placed on these systems suggest monetary compensation as a main human motivation driver. It caused multiple concerns in the scientific society such as gaming a system [38], demographics [39], cost-vs-quality question [40]. Ipeirotis et al. [41] analyzed the current (2010) state of the AMT marketplace including datasets and payments, and provided guidelines for researchers to employ human workers efficiently.

1.2.3 DIY Crowdsourcing

All crowdsourcing marketplaces provide researchers with API to design an experiment. However, only a few researchers have access to resources as well as technical background to launch a crowdsourcing system for their projects [42]. Multiple DIY (Do-It-Yourself) crowdsourcing services were proposed that provide easy-to-launch solutions. Crowdcrafting (available until April 1, 2019) [43] built on top of the crowdsourcing framework PyBossa (https://pybossa.com) offers easy to use platform with pre-built solutions for Geo-coding, Transcribing documents, as well as Sound, Image, and Video pattern recognition. Crowdcrafting was used to collect worldwide forest-related data (ForestWatch https://www.globalforestwatch.org) and to investigate the efficiency of new mosquito trapping technologies for malaria control (Rural GeoLocator [44]).

Curio [42] (http://crowdcurio.com) is an alternative DIY crowdsourcing platform. In contrast with Crowdcrafting that provides tools more suitable for simple annotation and classification tasks, Curio allows more flexibility enabling researchers to investigate questions related to problem decomposition, incentive design, and quality control. Besides, it employs mixed-expertise crowdsourcing paradigm that involves both expert and amateur workers. This model was successfully used to collect data for climate change studies [45].

It is interesting that Curio uses 'Fostering Curiosity Through Science' as a slogan for a platform. Indeed, Law et al. [46] studied retention of AMT human workers in the experiment with the added curiosity element. Authors demonstrated that induced curiosity increases the average number of tasks completed by participants without a loss in accuracy. The study proposed curiosity as a competent driver for crowdsourcing system and, thus, it was adopted by Curio.

1.2.4 Implicit Crowdsourcing

Another way of crowd engagement is to transform already existing (but unexploited) routine into the computational system. Von Ahn et al. created an iteration of the security system CAPTCHA [47] - the reCAPTCHA - the human computation system to digitize the old newspapers or books that are not clear enough for optical character recognition process (OCR) [48]. The system presents to a user a pair of words or short phrases that are corrupted and often cannot be confidently interpreted by a computer program. While one part of the text is known in advance and serves the purpose of a not-a-robot verification test (Turing test) [47], another part is used to collect human generated transcriptions. Similar to the visual data interpretation, reCAPTCHA also allows a user to work with sounds.

Von Ahn also proposed the idea of a gamified human computational system Duolingo that offers knowledge instead of monetary reward for implicit translating the web content of the internet to other languages [49].

1.2.5 Disaster Response Systems

Often, a simple request or call for help is a very effective strategy to employ participants. Crowdsourcing systems for disaster response are among the first tools that provide needed information for emergency services. Zook et al. [50] showed the use of collective effort to accumulate geographic crisis information mapped to the OSM after the Haitian Earthquake (January 12, 2010). A similar approach was applied during the earthquake in Northern Japan followed by a tsunami and Fukushima nuclear disaster (March 11, 2011) [51]. Disaster management tools were also used to monitor the spread of wildfire in Colorado, US (2012, 2013) [52].

1.2.6 Contest-Based Crowdsourcing

Lakhani et al. [53] found that a contest is a great way to engage (often skilled) people into solving a particular problem. The author designs the TopCoder system, the crowdsourcing platform for software development. Authors also illustrated the outstanding results achieved by competitors in the two-week long competition of solving the problem of complex sequence alignment [54]. Kaggle (https://www.kaggle.com) is another crowdsourcing platform utilizing prize contests. In contrast to the TopCoder, it primarily targets data science problems [55, 56, 57, 58].

1.2.7 Games and Gamified Systems

The gamification has a very important niche in crowdsourcing and human computing field. Entertainment was shown to be a stronger motivator for a user than payment [59]. While the literature suggests multiple taxonomy of the human computing games (e.g., casual/hard games [60]), we describe previous work based on the class of targeted problem (data classification, multiple sequence alignment, and combinatorial problems). It is also interesting that the problems of data classification and multiple sequence alignment are peculiar to casual games, while serious games often inherit more complex combinatorial problem.

Data Classification and Annotation

Human computing games are broadly used by researchers to obtain manual data classification and annotation. Von Ahn and his colleagues were pioneers in creating a human computing system with an emphasis on the gamification process [61]. The ESP game [14] searches for "an agreement on an image" and helps to identify the object on the image, whereas the Peekaboom [62], in addition to the presence of the object, collects its position on the image. Phetch game [63] is used to generate meaningful image descriptions using a single "describer" and a group of "seekers" to find an image that matches an image description given by a "describer". Law et al. proposed TagATune, an online audio game that collects human-provided annotations for music and sounds using the concept of the ESP game directly [64]. The Listen Game [65] also provides audio annotations where a player has to choose the best and the worst tags from specified ahead list, while competing against multiple people at the same time.

Human computing games also improve geotagging systems. The EyeSpy game [66] provides textual and pictorial map annotations. The Geo-Zombie game [67] addresses a problem of accessibility of urban areas, encouraging people to provide information about barriers on the city streets while running away from a zombie.

Bioinformatics also contains multiple topics where human intelligence was proven to be more efficient than a fully automated approach [60]. Computational Biology research in antibody clustering, imputing gene expression data process demonstrated significant performance gains when using crowdsourced approaches comparing to established algorithms [68].

Mavandadi et al. [69] created a web-based crowdsourcing game to collect healthy red blood cells or kill malaria-infected ones in the blood sample images. A similar idea was proposed by Luengo-Oroz et al. [70] to utilize human intelligence for malaria parasites quantification in the blood sample. Both studies reported highly accurate results in their applications and, therefore, can be used in training computer vision algorithms to enhance further automated image annotation tasks.

Multiple Sequence Alignment

Besides from data labeling tasks, some games make use of human ability to detect abstract object similarity. Kawrykow et al. [71] presented human computing game Phylo, which approaches the problem of multiple sequence alignments by presenting ambiguous segments of precomputed alignments to the player as a Tetris-style puzzle. Players are asked then to align the blocks (encoded nucleotides) in order to achieve or overcome the score predefined by the heuristic algorithm. Results from post processing and final assembling stages showed that over 70% of original alignments were improved. Kwak et al. [72] designed the Open-Phylo system, an open-access Phylo-powered platform that allows any researcher to utilize gamers for solving the multiple sequence alignment problem on custom data. Both Phylo and Open-Phylo were proposed to be used within an educational process [73].

The games described above encapsulate some specific high-level problem into the gamified interface so that even a player without any training or knowledge of the problem can contribute.

Combinatorial problems

Problem of data clustering, ranking, or assignment of a finite set of objects to satisfy specific conditions are known as combinatorial problems. Multiple research studies describe human computing approach to address combinatorial problems in genomics and proteomics.

The Foldit game [74] addresses the problem of protein structure prediction. It shows a draft structure of a protein to the players and asks them to manually tweak the structure or execute small automated moves on it while minimizing free energy score. Just in three weeks, Foldit players were able to produce superior results in modeling specific protein structure, whereas all the previous approaches had never succeeded [75]. Later, the Foldit was transformed into a scientific tool for protein structure manipulation [76]. Inspired by Foldit model, the EteRNA game [77] integrates

game players into the loop of laboratory trials for RNA secondary structure prediction problem. The EteRNA players design RNA molecule (propose a set of rules) which folds into the specific shape of interest. Best eight designs are then evaluated in the laboratory with detailed feedback to the players in order to improve the design even further.

The listed in the section games are considered as serious games that suggest individual real-world problem accompanied by an intensive multilevel training session.

1.2.8 Data Clustering using Human Computing Technique

Recent advances in Machine Learning and Artificial Intelligence significantly improved automated image classification process achieving near-human performance [78]. Tasks that previously were considered as human-only domain - nowadays can be fulfilled by a computer program and less frequently require human intervention.

In contrast, automated algorithms need human assistance to handle more context-related or semantic data analysis. For example, entity resolution is a process of grouping synonymous objects (often words) together: choosing among {John F. Kennedy International Airport, JFK Airport, and Kennedy Memorial Airport } is relatively easy for a human [79]. Similarly, precise formulation of ranking and clustering problems are known to be computationally intractable [80]. Many researchers applied crowdsourcing technique to cluster complex textual data [81, 82, 83, 84, 85]. The majority of the work is focused on showing to the participants a fixed portion of data. This strategy leads to incoherent results, which sometimes referred to as a redundancy problem. In contrast, the Alloy system [86] provides a global context through sampling and searching the entire dataset (by request). Alloy allows the researcher to employ crowd workers i) to identify text keywords for categorization, ii) to merge existing text categories, and iii) to verify outlier text categories. The authors demonstrated that Alloy outperformed most of the automated and available crowdsourcing methods. Their

analysis also illustrated the benefit of providing a user with a broader context of the dataset.

While the topic of the text clustering gained a lot of interest from scientific society, an application of human computation to abstract data clustering problem is poorly described in the literature and yet to be studied.

Since this thesis touches on the abstract data clustering problem, in the following section we provide a reader with a general background in this field.

1.3 Data Clustering

Data clustering process is a complex data analysis task which produces partitions of a given dataset into groups. Each object is more similar to the object from the same group than from the other. Yet, this definition is very broad. Similarity metrics can be defined in several different ways, which redefines the whole problem of clustering into application based sub-problems.

The field of data analysis splits the problem into multiple branches:

- Centroid-based clustering;
- Connectivity-based clustering;
- Density-based clustering;
- Model-based clustering;
- Graph-based clustering

Each branch has its own criterion for defining similarity metrics.

1.3.1 Centroid-based clustering

Centroid-based clustering contains the most commonly used set of clustering algorithms due to its simplicity and, therefore, speed. This type of algorithms represents each cluster through a single point (centroid), that is treated as a center of a

cluster. This point could be a point from a dataset (K-medoids [87], CLARA [88], CLARANS [89], K-Harmonic Means [90]) or an average computed over the points in a cluster (K-Means [91]). These algorithms are naturally capable of parallelism and relatively fast ($O(nkd)$). Yet they are easily drawn to the local optimum and can be affected by outliers.

Mean shift [92] is another centroid-based clustering algorithm that uses kernel density estimation function to identify centroids (modes) and then iteratively explores the proximity of each point to shift the mean of the points until its convergence. While this algorithm is slow($O(n^2)$), it can automatically determine the number of clusters and is also parallelizable.

1.3.2 Connectivity-based clustering

The main assumption of a connectivity-based clustering method is that the objects' similarity is proportional to the distance between the objects. Therefore, to build a model, it uses distance-based connectivity, which often represents a hierarchy between the objects. The algorithms usually employ either agglomerative (every data point is defined as a cluster and then merges existing clusters based on the provided linkage) or divisive strategy (partitions are made based in the context of the complete dataset). Though, the main difference among the connectivity-based clustering algorithms then is a linkage function used that serves as a criterion for merging or splitting previously observed clusters. The most discussed in the literature are single [93], complete [94, 95], average [96], and Ward [97, 98] linkage. The algorithm has a high time $O(n^3)$ and memory $O(n^2)$ complexity, which makes it often not feasible to use for a large dataset. BIRCH [99] and CURE [100] algorithms address this problem by increasing memory usage or performing clustering on a partition of a dataset.

1.3.3 Density-based clustering

More sophisticated branch of clustering is density-based clustering, which utilizes various heuristics to identify high-density regions in a dataset. For instance, DBSCAN [101] explores the ϵ-neighbourhood of a point to detect the specific number of points to determine if a point belongs to a cluster. OPTICS [102] is an extension of DBSCAN which records the number of points in a neighborhood and later adapts to varying density of the clusters.

The main advantage of these algorithms is its ability to handle arbitrarily shaped data as well as their speed ($O(n * log(n))$ for low dimensional dataset). Nevertheless, the results of these algorithms extremely depend on the choice of parameters. This problem is addressed by several authors [101, 103].

1.3.4 Model-based clustering

Model-based clustering assumes that a dataset is generated by some model, and therefore it attempts to recover the original model. It uses the expectation maximization algorithm [104], that iteratively approximates parameters of a statistical model. For example, the Gaussian Mixture Model [105] assumes data to be generated by a finite mixture-of-Gaussians. Although it is one of the fastest algorithms to identify the mixture models, there is a chance it converges to an arbitrary bad local maxim even in perfect conditions [106]. Therefore, it is recommended to run the algorithms multiple times.

1.3.5 Graph-/Network-based clustering

Graph-based clustering algorithms utilize graph theory to perform the partitioning of a dataset that is presented as a graph with vertices and edges denoting data points and the relationship between them, respectively. This branch of clustering methods includes algorithms which operate on minimum spanning trees (Kruskal's [107], Prim's [108]), or produces minimum *s*-*t* cuts (CLICK [109]).

Clustering on graph also can be interpreted as a community search problem [110] (in this context, it is common to see 'graph', 'vertex', and 'edge' terms to be replaced with 'network', 'node', and 'link'). To solve this problem, the algorithms optimize one of the criteria: i) Clustering coefficient/Transitivity that computes the fraction of all possible triplets of nodes (triangles) in a graph; ii) Modularity [111] that computers a difference between the number of edges that fall within clusters and an expected fraction of edges to be contained in a cluster given a random distribution of edges [112]

The major limitation of the structure evaluation method using Clustering coefficient is that it is heavily affected by the sparsity of a network, whereas the modularity optimization problem was shown to be NP-hard problem [113]. However, there are multiple heuristics proposed by the community: Louvain algorithm [114], Clauset, Newman, and Moore (CNM) algorithm [115], Walktrap [116], and Wakita and Tsurumi (WT) algorithm [117].

The Louvain algorithm detects communities in the network by maximizing recursively the modularity gain while reattaching nodes to neighboring communities. In contrast, CNM and WT merge entire communities using the expected improvement of modularity. Besides, WT uses community size balancing technique which gives a boost in speed but might have a negative effect on the modularity score of the final partition [114]. Walktrap captures the community structure in a network using random walks to measure the similarity between nodes.

1.3.6 Cluster validation

To compare the result of different clustering algorithms, it is common to use a cluster validation index. This is a function that provides a numerical evaluation of the clustering result.

Silhouette [118], Dunn [119], and SDbw [120] indices are among the most commonly used ones. Silhouette defines the total score through the average inter- and intra-cluster distances. The Dunn index compares a diameter of each cluster with

a minimal inter-cluster distance. In contrast, the SDbw index computes a scattering (sum of standard deviations of each cluster over the standard deviation of the dataset) and density at the midpoint of the cluster centers with respect to the cluster center densities.

A cluster validation index evaluates a result of a clustering algorithm, however, it is case-specific and should be thoughtfully chosen depending on the clusters property to be highlighted [121].

1.3.7 Clustering in high dimensionality space

While the clustering algorithms discussed above are applicable to work with multi-dimensional data, it is likely that their performance degrades with increasing dimensionality [122]. This effect would most likely be caused by the curse of dimensionality [123], where the distance function becomes less efficient at higher dimensionalities. Multiple strategies have been proposed in the literature to manage the high dimensional dataset [124, 125, 126], including feature selection, dimensionality reduction techniques, and subspace clustering algorithms.

Feature selection [127, 128] is a simple computational technique to filter out non-informative features (or dimensions) from a dataset using either internal properties of the dataset (e.g., variance, mutual information, a correlation between features and a target) or predictive models (Naive Bayes, SVM, Random Forest). Correlative approaches have shown to be very efficient and robust to overfitting and yet, these approaches tend to select redundant features. Predictive models, on the other hand, generally detect the relations between one or more variables and are sensitive to overfitting. These models are extensively discussed in the literature for both supervised [129, 130, 131, 132] and unsupervised learning problems [133, 134, 135, 136].

Another commonly used technique is a feature transformation (or extraction) [124], which is often referred to as a dimensionality reduction [124]. This techniques generate a completely new set of features, which can be either linear (PCA [137], PCoA [138])

or non-linear (Isomap [139], t-SNE [140], Autoencoder [141]) combinations of the original features. While these techniques are expected to provide features relevant to the original dataset, their results often lack interpretability [140].

While feature selection and extraction have proved to be efficient techniques, the alternative approach of biclustering [142] was designed to primarily work with high dimensional datasets. This algorithm simultaneously groups together samples and features to produce a submatrix of the original data with elements found to be useful. Biclustering has broadly been used in bioinformatics to explore gene expression data [143, 144, 145, 146] and also extended to the other fields, including databases, natural language processing and data mining. Subspace clustering [147, 126] is an implementation of the biclustering algorithm that employs either Bottom-Up or Top-Down subspace search techniques [148]. Bottom-Up compiles high dimensional clusters from previously obtained clusters in lower dimensional subspaces and assumes that if subspace $\mathcal{S}$ contains a cluster C then any lower dimensional subspace $\mathcal{P} \subset \mathcal{S}$ also includes C [149, 150, 151, 152]. In contrast, the Top-Down approach iteratively finds an approximation of the clusters in a complete feature space and then refines dimensional weight to be used for the next iteration [147, 153]. In the literature Bottom-Up and Top-Down subspace clustering methods are often referred to as 'subspace clustering' and 'projected clustering' algorithms, respectively [148].

1.4 Visualization of genomic data

The text found in this section is taken from:

> Jérôme Waldispühl, Eric Zhang, Alexander Butyaev, Elena Nazarova, and Yan Cyr. "Storage, visualization, and navigation of 3D genomics data." *Methods* 142 (2018): 74-80.

1.4.1 Contact maps and 2D representations

The release of the first draft of the human genome [154, 155] triggered enormous development in the area of genomics. Recent advances in biochemistry and biotechnology (e.g., high throughput technologies [12]) allowed the researchers to generate unseen previously amount of data. Nevertheless, while data-driven research is very prominent, data storage, searching, analysis, interpretation, and visualization are still an open-ended problem [156, 157].

The most elementary level of genomics data available is inferred inter- and intra-chromosomal contacts from 3C or ChiP-based experiments. The UCSC genome browser integrates this information in a data type called longTabix that is shown as arcs connecting two indexes in a linear representation of the genomes [158]. A similar approach has been implemented in HiView [159], which allows mapping Genome-Wide Association Study (GWAS) and Hi-C data (count data) on the top of linear genome browser integrating UCSC and Ensembl annotations. Interestingly, recent software such as HUGIn is still making use of linear representations (as histograms) to compare virtual 4C data across tissues or cell lines [160].

My5C was among the first web tools developed to visualize Hi-C chromatin interaction maps as heatmaps [161]. Noticeably, it integrates several programs to analyze these maps. The Hi-C Data Browser is another web tool that was initially developed to display the first Hi-C data [162]. It evolved into a new software called Juicebox that features a real-time zoom allowing users to smoothly adjust the resolution of the display [163], and include a toolbox to analyze Hi-C data [164]. A cloud-based version has also been recently introduced, which enables easy integrations in third-party web browsers [165].

The WashU epigenome browser provides another alternative to the visualization of Hi-C heatmaps [166]. This browser, which has been initially developed to visualize ENCODE data [167], provides an intermediate approach to explore 3D genomics data. Genome annotations are represented using a linear model, but the viewer allows to

map on histone modification profile ChIA-PET interactions and Hi-C heatmaps (Fig. 1.1a).

More recently, an efficient and versatile web tool named HiGlass (Fig. 1.1c) designed for exploring Hi-C data generated by the 4DN consortium has been released [168]. Noticeably, HiGlass allows index searches, has multiscale capacities and features a mechanism enabling simultaneous exploration of multiple independent or linked heatmaps. It has also been used in HiPiler [169], another web tool facilitating the exploration and comparison loops and domains. HiPiler expands the scope of techniques available to analyze interaction matrices, that were previously primarily implementing statistical analysis methods (See HiBrowse [170]). The genome contact map explorer is another recent Desktop-based tool developed to analyze and compare Hi-C contact maps [171]. It has been implemented in Python and offers a broad variety of interfaces including a command line and graphical interfaces and an API.

Nonetheless, new platforms are still being developed to answer specific needs. For instance, the 3D genome browser incorporates heterogeneous 3D (e.g. 3C-based, CHiA-PET) and omics datasets (e.g. RNA-seq), and allow queries to others popular browsers such as the WashU epigenome browser [172]. The HiCExplorer [173] facilitates simultaneous analysis of Hi-C data and other genomics annotations (e.g. genes, ChIP-seq). It has been implemented in Python and runs using a command line interface. Finally, 3DIV integrates almost all features of previous visualization tools, but has the advantage to be coupled with an up-to-date database of chromosomal interaction and epigenomics datasets [174].

An alternate visualization approach was implemented in Capture Hi-C Plotter (CHiCP) [175]. CHiCP uses a circular representation of the sequences, in which long-range chromosomal interactions are represented with arcs (Fig. 1.1b). This display appeared to be convenient to visualize 4C and virtual 4C data (i.e. the list of all interactions with an explicit locus). Although circular representations were previously used in genomics [176, 177, 178], CHiCP has been customized for Hi-C data and dynamic

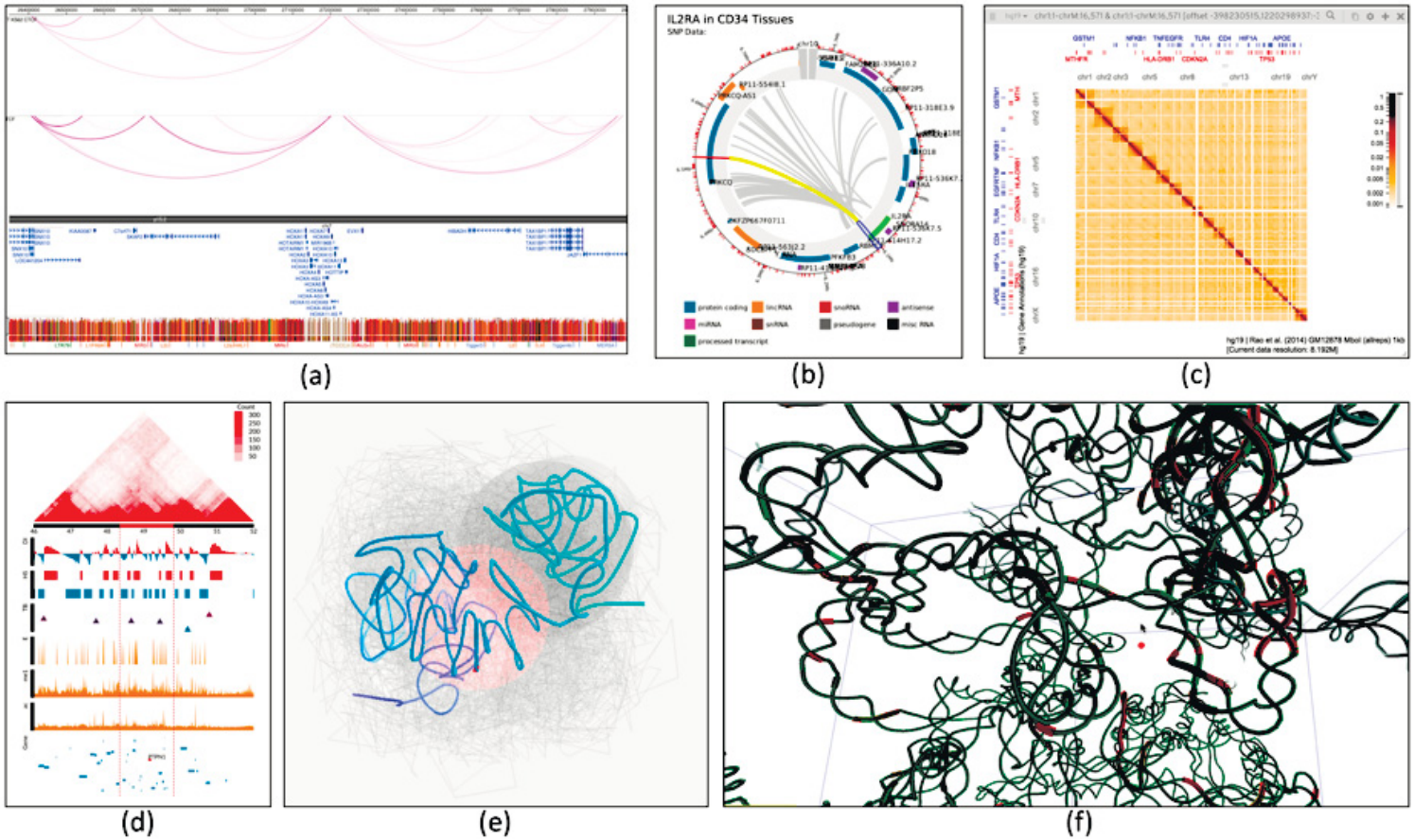

FIGURE 1.1: Overview of 3D genome visualization interfaces. (a) arc representation of ChIA PET interactions within the WashU Epigenome browser, (b) Circular representation of chromosomal interactions and genomics annotations with CHiCP, (c) Hi-C interaction matrix visualization with HiGlass, (d) upper triangular matrix representation of Hi-C data mapped on linear genomic annotation tracks of a TAD region with the 3Disease browser, (e) 3D representation of a TAD region with TADkit, (f) immersive 3D exploration of full 3D genome reconstruction with 3DGB.

manipulation of circular plots. Rondo is another web-based 3D genome browser using this layout [179], which in addition indexes the data to navigate maps at different resolutions. Nonetheless, it is worth noting that the WashU epigenome browser [166], HiBrowse [170] also feature circular genome representations.

1.4.2 3D genome structures

While the representation and manipulation of chromosomal contacts and interaction matrices appear to reach maturity, the visualization and exploration of 3D genome structures are still in its infancy. Many software such as Chimera [180] have been previously developed to visualize 3D molecular structures, but none of them were designed to address the challenges of 3D genomics, including multiscale modeling and the management of numerous heterogeneous genomics annotations.

The 3Disease browser [181] includes partial 3D visualization functionalities. Beside a comprehensive interface integrating multiple tracks showing gene annotations and Hi-C interaction heatmaps (Fig. 1.1d), it can also display the 3D structures of TADs. This platform has been specifically designed to study disease-associated chromosomal rearrangements and integrates the ShRec3D algorithm used to build 3D models from Hi-C data [182].

TADkit is another popular TAD viewer featuring similar functionalities [183] (Fig. 1.1e). It serves as the front-end of a powerful platform for automated analysis of 3D genomics data named TADbit.

The first program specifically designed to explore complete 3D genome architectures appears to be Genome3D [184]. It features multiscale capacities and supports annotations from the UCSC genome browser [185]. However, Genome3D has been implemented as a standalone application running on Windows Operating Systems, which inevitably limits its distribution.

By contrast, GMOL is implemented in Java and offers an alternate, more universal, solution for multiscale visualization of 3D genomes [186]. Is also includes additional

functionalities such as a command line, scripting interpreter and a measurement tool of physical distances.

Finally, 3DGB implements a distinct, web-based, strategy for the exploration of complete 3D genomes [187] (Fig. 1.1f). Unlike the other 3D genome viewers introduced above, 3DGB is implemented in Javascript and runs in a web browser. The deployment of this technology was facilitated by the use of the 3DBG database system, which reduces spatial query latency and thus permits the dynamic loading of the data in the client browser. This infrastructure enables 3DGB to propose an immersive exploration of 3D genome structures. 3DGB includes most of the tools mentioned above (e.g. scripting, distance measurement) except the zoom functionality. It also integrates the Genome Maps web application [188] for navigating linear representations of genome sequences.

This concludes the exert from Waldispühl et al. (2018).

1.5 Thesis Roadmap and Author Contributions

Crowdsourcing is a popular and well-established technique to perform a large number of repetitive tasks. The underlying assumption of the technique is that aggregating sufficiently large number of non-experts answers could approximate expert behaviors [77, 74]. Crowdsourcing strategies focus on efficiently engaging a crowd of non-expert worker to produce useful input for a computational system, which would unlikely be obtained otherwise. In combination with Human-Computer Interaction (HCI) techniques, which study the design of interfaces between human and computer, crowdsourcing touches on a large spectrum of various topics related to computational system including a mechanism of human interacting with a system, players' motivation to contribute to a specific problem, and strategies to aggregate human answers to provide the final result.

Chapter 1 provides the reader with a general background in the three main research topics covered in this thesis: i) crowdsourcing and human computation, ii) abstract

data clustering, and iii) genomic data visualization and interpolation. Recent advances in Artificial Intelligence (AI) allow human-driven analysis to become completely automated. Yet, there are many applications (e.g., ranking or cluster analysis [80]) where AI computer programs require human guidance and intuition to obtain accurate results. Therefore, to overcome this limitation of AI, we provide an overview of major studies in the area of crowdsourcing and human computation as motivation for our human computing games (Chapter 3, Chapter 4) and algorithms that outperform AI in Chapter 5. In thesis, we address the problem of data clustering (Chapter 4, Chapter 5). Thus, we describe the major approaches in the field to this problem. Finally, each chapter of the thesis, directly or indirectly, discusses the data visualization and interpretation techniques. Therefore, the remainder of chapter 1 provides an overview of the data visualization and interpretation techniques in the area of HCI, with a high emphasis on genomic data.

Chapter 2 describes the problem of visualizing three-dimensional (3D) genomic data in an interactive 3D space. To address this problem, we present 3D Genome Browser (3DGB), an immersive web-based 3D Genome Browser interface that is built upon a low latency database. 3DGB enables efficient data browsing of 3D genomic data within any standard web-browser (i.e., Internet Explorer, Safari, Chrome, etc.).

Next, in Chapter 3, the specificity of human-computation systems are explored. Specifically, this chapter aims to understand how humans collaborate, how this phenomenon benefits a system, what incentives are needed to promote collaboration, and how to make an interaction between participants more efficient. To address this problem, we develop the human computing game that simulates a market as the primary tool for collaboration between participants. We examine its effect in context of classical graph theory problem (clique) and demonstrate the importance of a collaborative/competitive environment in the human computation system. In addition to the game environment, we also explore other motivators, such as game challenges and skills, that help direct a player's attention towards poorly explored part of the underlying problem.

In Chapter 4, the application of HCI crowdsourcing techniques in cluster analysis are then explored. We investigate how humans perceive simple two-dimensional (2D) clusters. We describe the mobile online human computing game 'Colony B', a computational framework to collect human input (i.e., crowdsourced) for simple 2D clustering problems. Then, in Chapter 5, we extend our crowdsourced clustering approach to work with higher dimensionality datasets. Two clustering algorithms (hubCLIQUE and CloCworks) are proposed that use aggregated human solutions collected from Colony B players to identify key features in male and female voice samples. Our results demonstrate that the proposed algorithms outperform most automated clustering algorithms considered. Finally, Chapter 6 provides a summary of the preceding four chapters and discussion on possible future works.

This thesis includes the text and figures from several scientific articles. Each article has been published, submitted, or is in preparation for submission to a journal. Alexander Butyaev is either first or co-first author for each article found in Chapter 2-5 and a co-author in Chapter 1. The articles are listed below in their order of appearance throughout the thesis.

Chapter 1

Jérôme Waldispühl, Eric Zhang, Alexander Butyaev, Elena Nazarova, and Yan Cyr. "Storage, visualization, and navigation of 3D genomics data." *Methods* 142 (2018): 74-80. doi: 10.1016/j.ymeth.2018.05.008

J.W. prepared the draft of the manuscript. E.Z., A.B., E.N. helped draft the manuscript. Y.C. consulted for VR systems. All authors reviewed and approved the manuscript.

Chapter 2

Alexander Butyaev, Ruslan Mavlyutov, Mathieu Blanchette, Philippe Cudré-Mauroux, and Jérôme Waldispühl. "A low-latency, big database system and browser for storage, querying and visualization of 3D genomic data." *Nucleic acids research* 43, no. 16 (2015): e103-e103. doi: 10.1093/nar/gkv476

A.B. and R.M. contributed equally to the development of the text. A.B. developed the 3D Genome Browser (3DGB), performed computational analysis, helped with benchmarking

of proposed approach against state-of-art methods, and helped draft the manuscript. R.M implemented the database storage, performed the algorithms comparison, and prepared the first draft of the manuscript. M.B consulted on the available 3D genomic data. P.C.M. and J.W. coordinated the design of the systems and helped draft the manuscript.

Chapter 3

Olivier Tremblay-Savard, Alexander Butyaev, and Jérôme Waldispühl. "Collaborative Solving in a Human Computing Game Using a Market, Skills and Challenges." *In Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play*, pp. 130-141. ACM, 2016. doi: 10.1145/2967934.2968104

O.T.S. and A.B. contributed equally to the development of the text. A.B. developed the computational back end and data management sections of the Market game, contributed to the design of the front end of the game and data analysis, and helped to prepare draft of manuscript. O.T.S. developed the front end of the game, arranged the game sessions, outlined and performed data analysis, prepared the first draft of the manuscript. J.W. coordinated the game design and computational analysis and contributed to the manuscript development.

Chapter 4

Alexander Butyaev, Chris Drogaris, Elena Nazarova, Olivier Tremblay-Savard, and Jérôme Waldispühl. "How do Humans Perceive 2D Clusters? Lessons From a Mobile Crowdsourcing Human-Computing Game." *In submission at Computer-Supported Cooperative Work and Social Computing conference.*

A.B. contributed to the design of the mobile application, developed server side of the `Colony B` *game and data management system, implemented the computational analysis, and prepared a draft for the manuscript. C.D. designed the mobile application for the* `Colony B` *game. E.N. designed the reward system (badges). O.T.S. conceived the original idea of the game and helped draft the manuscript. J.W. contributed to the design of the mobile application and coordinated the computational analysis.*

Chapter 5

Alexander Butyaev, Chris Drogaris, and Jérôme Waldispühl. "Colony B: Multidimensional Data Clustering with Human." *In preparation for submission at Human Computation and Crowdsourcing conference*.

A.B. contributed to the design of the mobile application, developed server side of the `Colony B` *game and data management system, implemented the computational analysis, and prepared the first draft for the manuscript. C.D. designed the mobile application for the* `Colony B` *game. J.W. contributed to the design of the mobile application and coordinated the computational analysis.*

Chapter 2

A low-latency, big database system and browser for storage, querying and visualization of 3D genomic data

Alexander Butyaev [1,†], Ruslan Mavlyutov [2,†], Mathieu Blanchette [1], Philippe Cudré-Mauroux [2], Jérôme Waldispühl [1]

[1] School of Computer Science and McGill Center for Bioinformatics, McGill University, Montréal, Québec, Canada

[2] eXascale InfoLab, University of Fribourg, Switzerland

[†] Equal contributors

2.1 Preface

In biology, the genetic code dictates the rules by which the living cells exist [189]. Although, the release of the first draft of the human genome [154, 155] was an outstanding breakthrough in the area of genomics and gave a great momentum for a development in the field, the primary structure of the genomes does not provide complete information to decipher our genetic code [190].

Recent advances in biotechnologies, such as chromosome conformation capture techniques [12], enabled researchers to grasp the importance of spatial genome organization and make a step toward better understanding of some regulatory mechanisms. It also allowed to generate 3D models of the genomes of multiple living species [162, 191, 192].

While previously the area of genomics was operating with a variety of annotations mapped to the one-dimensional sequence (reference genome) [193, 194], the 3D genome data allowed scientists to identify hidden before relations between genome elements. Current solutions for visualization and interpretation of such data are not capable of providing primary and tertiary genome structure simultaneously. Moreover, often they are OS specific standalone applications that require to install numerous external packages. Therefore, the field needs more user-oriented tools suitable for explorative interaction with 3D genomic data.

This chapter describes the novel database storage 3DBG that efficiently manages 3D genomic data. It introduces interactive web-based 3D genome browser 3DGB capable of visualizing and exploring 3D genome structures served by 3DBG. It presents API to access the database as well as a scripting tool that allows interacting with the database using a web browser.

In this chapter, before designing crowdsourcing systems (Chapters 3,4,5) we aim to gain experience with designing HCI systems that are convenient to use for humans to interpret the data. We choose genomics as it seems a natural field for exploring how to manipulate the data for its better visualization and, therefore, the following

interpretation by a user.

The remainder of text found in this chapter is taken from:

> Alexander Butyaev, Ruslan Mavlyutov, Mathieu Blanchette, Philippe Cudré-Mauroux, and Jérôme Waldispühl. "A low-latency, big database system and browser for storage, querying and visualization of 3D genomic data." *Nucleic acids research* 43.16 (2015).

A.B. and R.M. contributed equally to the development of the text. A.B. developed the 3D genome browser (3DGB), performed computational analysis, helped with benchmarking of proposed approach against state-of-art methods, and helped draft the manuscript. R.M implemented the database storage, performed the algorithms comparison, and prepared the draft of the manuscript. M.B consulted for available 3D genomic data. P.C.M. and J.W. coordinated the design of the systems and helped draft the manuscript.

2.2 Abstract

Recent releases of genome 3D structures have the potential to transform our understanding of genomes. Nonetheless, the storage technology and visualization tools need to evolve to offer to the scientific community fast and convenient access to these data. We introduce simultaneously a database system to store and query 3D genomic data (3DBG), and a 3D genome browser to visualize and explore 3D genome structures (3DGB). We benchmark 3DBG against state-of-the-art systems, and demonstrate that it is faster than previous solutions, and importantly gracefully scales with the size of data. We also illustrate the usefulness of our 3D genome Web browser to explore human genome structures. The 3D genome browser is available at http://3dgb.cs.mcgill.ca/.

Keywords: 3D genome | structure | database | genome browser

2.3 Introduction

2.3.1 Biological motivation

The release of the first draft of the human genome [154, 155] announced the beginning of a data era in genomics. Gathering information does no longer appear as a major bottleneck in molecular biology studies. However, by contrast, storing and mining the massive amounts of data generated by genomics studies becomes increasingly difficult.

The development of efficient computing infrastructures to store and retrieve genomic information is thus an essential part of the discovery pipeline in genomic research. The UCSC genome browser [185, 194] and the Ensembl genome browser [193, 195] were among the first systems specifically developed to address this issue and opened the access of sequencing data to the whole scientific community.

Since then, the complexity and the nature of the data themselves has changed. In particular, the primary structure of genomes does no longer appear to contain all the

information required to decipher our genetic code. Indeed, recent studies suggest that the three-dimensional (3D) structure of genomes is essential to understand some regulatory mechanisms [190].

3D structures and Hi-C data sets of Human [162, 191] or Yeast genomes [192] are now available. Viewers have been developed to visualize complete genome structures [184] and Hi-C data annotations have been integrated in classical genome browsers [166]. However, to date, there is no scalable solution to query simultaneously primary and tertiary genome structures. Moreover, unlike classical human genome browsers, these viewers are currently only available as Operating System (OS)-specific standalone applications that are not embedded into Web browsers.

In this paper, we aim to address these challenges and develop a complete solution for storing and analyzing 3D genomic data. More specifically, we develop a new database system for storing and querying 3D genomic data, and a lightweight 3D genome browser for real-time visualization and exploration of 3D genome structures. We emphasize that the development of efficient database architectures is key for the success of a novel generation of 3D genome browsers.

2.3.2 Database technology

There has been substantial related work on storing and querying 3D data in the context of astronomy, remote-sensing, neurology or more broadly for the storage of spatio-temporal data. Existing approaches can be broadly categorized along three axes:

- Index-based versus cluster-based. Most existing work builds a 3D index over the raw data, but does not attempt to physically reorganize the raw data (index-based) so that co-queried objects are stored near each other (cluster-based).

- Adaptive versus non-adaptive. Adaptive systems adjust storage structures based on the query size and/or the data. They generally require less tuning and can typically handle a broader range of data sets than non-adaptive systems.

- On-line versus off-line. On-line systems change their storage representation as data arrives, whereas off-line systems assume that the indexed data does not change frequently and must recompute their storage layout from scratch as data arrives.

The 'classic' database structure for indexing objects along multiple dimensions is the R-Tree [196]. Unlike 3DBG, R-Trees do not per se cluster data and are optimized for accessing arbitrary spatial objects, rather than large amounts of data organized along 3D trajectories. Of course, it is possible to attempt to physically co-locate (cluster) objects in the same R-Tree rectangle together on disk. Even so, as R-Trees consider nested bounding rectangles to index the objects, it is very likely that if there is much data within a small area, there will be large overlaps in these bounding rectangles, resulting in many I/Os to answer any query.

There have been many optimizations to R-Trees for spatio-temporal data, including TB-Trees [197] and SEB-Trees [198]. TB-Trees are optimized R-Tree indices with special support for temporal predicates. They also do not deal well with very long 3D trajectories that tend to have very large bounding rectangles, and can include a high number of I/Os per lookup. SEB-Trees segment space and time, but are not specifically designed for indexing trajectories. Research on TB-trees and SEB-trees does not explicitly discuss how to cluster data, and both are non-adaptive (i.e. they do not reorganize previously added pages as new data arrives).

To address the concern with very large 3D meshes or trajectories, several systems have proposed segmenting the trajectories to reduce the sizes of bounding boxes and group portions of trajectories that are near each other in space together on disk. Rasetic et al. [199] propose splitting trajectories into a number of sub-trajectories, and then indexing those segments in an R-Tree. They propose a formal model for the number of I/Os needed to evaluate a query, and use a dynamic programming algorithm to minimize the I/O for an average query size. 3DBG also includes an algorithm for

optimally splitting 3D genomic meshes and their associated metadata, but in addition physically clusters those segments rather than just indexing them.

SETI [200] also advocates a segmentation-based approach like 3DBG. It segments incoming 3D meshes/trajectories into sub-trajectories, groups them into a collection of 'spatial partitions' and then runs queries over just the spatial partitions that are most relevant to a given query. The principal differences between 3DBG and SETI are that: (i) the SETI paper does not describe how the size or geometry of partitions is selected, or whether it changes as inserts occur, which is a key contribution of 3DBG, and (ii) SETI does not discuss metadata storage and clustering, read-optimized operations or scalability features.

PIST [201] focuses on indexing individual points rather than 3D meshes. PIST is similar in spirit to 3DBG in that it attempts to optimally partition a collection of points into a variable-sized grid according to the density of the data and query size using a quad-tree like data structure. Unlike 3DBG, PIST is off-line (i.e. it does not adapt to new data being added dynamically).

A number of other systems, such as STRIPES [202], use a dual transformed space to index meshes or trajectories. While such indices are very compelling when indexing the future positions of moving objects, they are known to be suboptimal for answering historical or ad hoc queries [203].

Spatial clustering has been extensively studied (21–23). These approaches focus on generic methods to extract cluster information from large collections of ad hoc data points. Our clustering problem is more specific, since we deal with series of points ordered along 3D genes, and more importantly on the (potentially large) metadata associated to the 3D models.

2.3.3 Contribution

In this paper, we introduce a complete efficient and scalable database system to query genomes in space. The system includes two components: (i) a database 3DBG to store

and query the 3D genomic data, and (ii) a web-based genome browser 3DGB to visualize and navigate 3D genome structures. As far as we are aware, 3DBG is the first database system that is online, adaptive and cluster-based. We designed 3DBG to optimize the speed of searching and accessing genomic annotations from their 3D spatial coordinates in genome structures. We also develop a lightweight 3D genome viewer 3DGB that is fully embedded in Web browsers and accessible to any web user who wishes to browse and query 3D genome structures.

Our system aims to foster the discovery of spatial relationships between genomic elements and accelerate the large-scale analysis of space-dependent regulatory mechanisms. Here, we map data from the 1000 genomes project [204] and experimental Chip-Seq data [194] onto most recent 3D models of the Human genome [162, 191], and use 3DBG to mine these data. We benchmark 3DBG against state-of-the-art systems, and demonstrate that our database system is faster than previous solutions, and more importantly that it scales better with the size of data. We also illustrate the usefulness of our system and use our 3D genome Web browser to explore the 3D neighborhood of the retinoblastoma gene (RB1) and identify potentially interesting genetic relationships between retinoblastoma and sleep disorders.

Our system is freely available at https://github.com/mavlyutovrus/3d_genome_browser and a sample deployment of our 3D genome explorer is accessible at http://3dgb.cs.mcgill.ca/.

2.4 MATERIALS AND METHODS

2.4.1 3D genome database

We have implemented a fully functional database based on YARN (http://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html), currently one of the most promising Big Data processing framework available. 3DBG takes advantage of the lower-level distributed filesystem of YARN (HDFS) to store the data chunks

over large clusters of commodity machines. Our system is based on three main components:

- **a 3D client**, which allows the user to navigate through the 3D genomic space. It dynamically retrieves high-resolution 3D data and genomic metadata from the rest of the system as the user moves through the 3D space and makes queries about certain 3D regions.
- **a sparse, adaptive 3D index**, which dictates how genomic metadata associated to contiguous regions in the 3D space are co-located in the distributed filesystem. The 3D index translates the 3D query posed by the user into a series of data chunks that have to be retrieved from the distributed filesystem.
- **immutable data chunks** that compactly store genomic data and metadata in the distributed filesystem.

Figure 2.1 gives an overview of our database. We implemented our own indices and ancillary data structures to optimize all operations, and bypass the Hadoop NameNode whenever possible to reduce the end-to-end latency of the queries. We do not rely on higher-level Hadoop data structures such as those offered by SpatialHadoop [205] or Impala (http://www.cloudera.com/content/cloudera/en/products-and-services/cdh/impala.html), since these higher-level structures negatively impact the performance of on-line queries. Along similar lines, we do not directly use large-scale batch-processing features a la MapReduce, since they would introduce unreasonably high latencies in our context, but could take advantage of such functionalities for off-line operations such as batch updates or complex analytics.

A detailed description of each component of our database, as well as an explanation on our query insertion and query execution techniques, is available as supplementary material. The full codebase of our current implementation is available online at https://github.com/mavlyutovrus/3d_genome_browser.

2.4.2 Data and web services

We describe the data stored in our database and the syntax of web queries to access them. Additional instructions and scripting packages for javascript users can be found at http://3dgb.cs.mcgill.ca/scripting/.

3D structures

Currently, three complete models of human 3D genome structures are stored in our database. We retrieve these data from [191, 184] and describe them in Table 2.1. It is worth noting that [191] provides individual structures for each chromosome, but no global relative arrangement of all chromosomes. For this reason, we provide independently each chromosome structure.

Origin and description of the 3D models stored in 3DBG

The structures are interpolated with a finite number of points. This number of points depends of the resolution of the model and therefore varies from one model to another.

The volume encompassing the genome structure is segmented in 3D cubic cells. Spatial queries use the coordinates of a cube (starting and ending positions on the x, y and z axis) as an input and return the coordinates of the interpolation points modeling the DNA chains contained within this cell. In particular, it allows us to identify the ranges of DNA subsequences within this volume.

The syntax of a query is http://1kgenome.exascale.info/<mode>?xstart=<x1>&xend=<x2>&ystart=<y1>&yend=<y2>&zstart=<z1>&zend=<z2>, where <*mode*> should be replaced by *js_test* to query the model from [184], or *3D* to query the model from [191]. <*x1*> to <*z2*> indicate the spatial coordinate of the cell. Queries to the structure issued from [191] should also include the chromosome number and the type of the cell (normal or leukemia). In that case, an example of a full query could be http://1kgenome.exascale.info/3d?chr=19&m=normal&xstart=1&xend=2&zstart=1&zend=2&ystart=1&

TABLE 2.1: Origin and description of the 3D models stored in 3DBG

Cell type	Organism	Type	Scale	#fragments	Reference
K562	human	simulated	whole genome	1	[184]
B-cell GM06990	human	real	individual chromosomes	13	[191]
B-cell luekemia	human	real	individual chromosomes	13	[191]

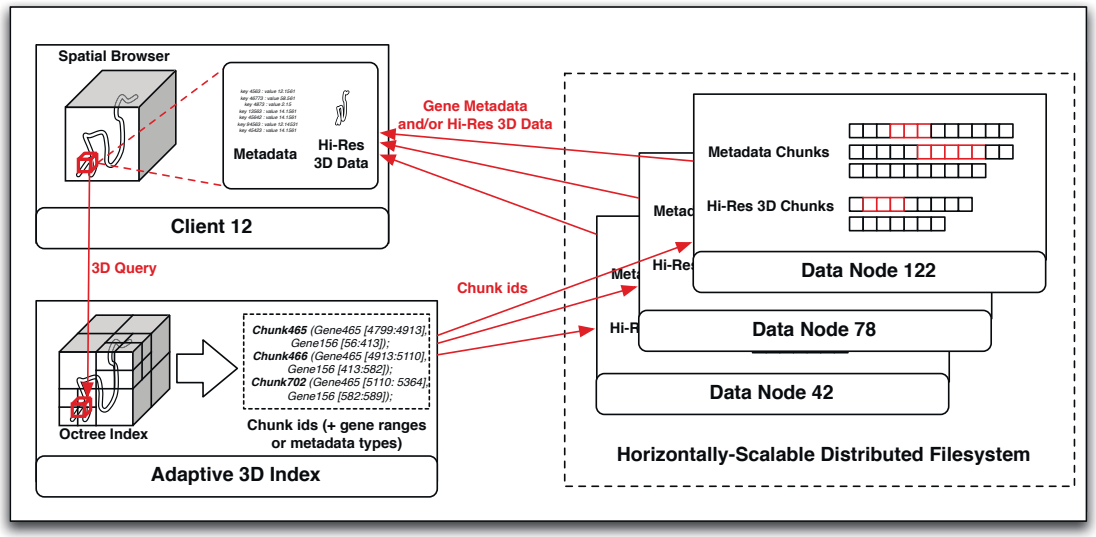

FIGURE 2.1: Overall architecture of 3DBG.

yend=2. The output is represented as an array of arrays, which represent contiguous chains within the volume.

Nucleotide sequences

We use *GRCh38* assembly of the human genome from the UCSC genome browser as our reference human sequence [185, 194]. Nucleotide sequences can be accessed from their chromosomic location. The syntax of a query is http://1kgenome.exascale.info/range?start=<start>&end=<end>&chrid=<chr> , where <*start*> and <*end*> are the first and last index of the subsequence of interest, and <*chr*> is the chromosome number (N.B.: **X** and **Y** chromosomes are identified using letters *X* and *Y* instead of numbers).

Single nucleotide polymorphism

We store the Single Nucleotide Polymorphism (SNP) data from the 1000 Genomes Project [204]. Web users can retrieve SNPs data within a specific range of a chromosome with the following query http://1kgenome.exascale.info/js_snp?chr=<chr>&start=<start>&end=<end>, where <*start*> and <*end*> are the first and last index of the subsequence of interest, and <*chr*> is the chromosome number.

A query returns an array of arrays showing information for each individual SNP found within this interval. This information is represented as a 4-tuple including the SNP position, the SNP ID and the two alleles.

Experimental ChIP-Sequencing data

We recorded experimental ChIP-Sequencing (Chip-Seq) data from the ENCODE project [206] stored in the UCSC genome browser [185, 194]. These data help us to identify transcription factors binding sites (TFBS).

ChIP-Seq data can be retrieved with a query to http://1kgenome.exascale.info/chipseq?chr=<chr>&start=<start>&end=<end>&celline=<cellid>, where the variables <*start*> and <*end*> are the first and last index of the subsequence of interest, <*chr*> is

the chromosome number, and <*cellid*> is the cell line from which we obtained the experimental data. It is worth noting that in practice, Chip-Seq data may not always be available for all 3D structures models and cell types.

The output of such query is an array of 7-tuples that contain basic information on the Chip-Seq data. A 7-tuple stores the chromosome number, starting and ending index of the Chip-Seq peak, the transcription factor name, a normalized value (ranging from 1 to 1000) indicating the magnitude of the binding, the cell lines with similar TFBS and a list of SNPs occurring in this binding site.

Determining single nucleotide 3D coordinates

A key feature of a system for querying genomes in space is its capacity to directly access the 3D coordinates of any nucleotide. However, 3D genome structures are often modeled with (sparse) discrete sets of points corresponding to enzyme cut sites. In that case, it is useful to directly access the closest cut site (in each strand direction) of a model.

This information is accessible with a web query to http://1kgenome.exascale.info/chr_pos?chrid=<chr>&bp=<index>&m=<mode>, where <*chr*> is the chromosome number and <*index*> is the sequence index of the nucleotide. The variable <*mode*> should be set at 'normal' to query the GM06990 cell data or 'leukemia' to query the leukemia cell data (N.B.: these key words are subject to change for more precise acronyms with the addition of new cell types). This argument can be simply ignored if the user wishes to query the K562 data. The query returns an array of triplets indicating the 3D coordinate of the closest interpolation points.

3D genome Web visualization interface

Web users can access and visualize data stored in our servers via a GUI accessible at http://3dgb.cs.mcgill.ca/. Our client, based on dynamic Javascript, mostly allows the user to navigate through the 3D structure in real-time, fetching genomic

data as well as high-resolution 3D meshes representing the DNA backbone from the server. It runs on most common web browsers (Firefox and Chrome). This contrasts with previous viewers that were implemented as standalone applications for specific operating systems. The source code of our browser is freely available at https://github.com/mavlyutovrus/3d_genome_browser.

Before starting to explore 3D genome structures, the users must select a model. The front page of 3DGB allows users to select which model they wish to use. Currently, three complete 3D data sets have been implemented in the database. The first is a simulation of the complete diploid human genome by Asbury et al. [184], while the second and third ones are recent reconstruction of individual chromosomes by Trieu and Cheng [191] for normal B-cells (GM06990) and acute lymphoblastic leukemia cells, respectively. New models will be added to the database as they appear in the literature.

Once a model is selected, users access a search engine that enables them to directly request specific genomic locations (i.e. chromosome number and position), target genes or arbitrary spatial coordinates. Queries re-direct the users to a 3D structure viewer pointing at the desired location. From there, they can explore and navigate the genome structure in real-time. The web client downloads all genomic and structural data in the neighborhood of the query location. More data are dynamically loaded when the user travels in the 3D space. This allows a smooth exploration of the 3D genome structure on any computing device. A screenshot of the 3D genome browser is presented in Figure 2.2.

Web tools

The viewer implements multiple features allowing its users to access and visualize Human genome data stored in the database. At the core of 3DGB resides our ability to define and query a 3D neighborhood, and thus to identify potential spatial relationship between genomic elements. In our viewer, a cursor of points at the center of a box representing a neighborhood to be explored. This neighborhood is represented by

FIGURE 2.2: Sample screenshot of 3DGB.

the red box in Figure 2. The size of the box is adjustable (by scrolling up or down), allowing the users to tune the range of spatial relationships. Once a volume has been selected (directly from a query or following an exploration of the genome structure), the user can retrieve and download the list of all SNPs located within that box, or use hyperlinks to directly access detailed information stored on the NCBI databases [207] for each individual SNP. In addition, it is also possible to access the list of all genes present in the query cell.

Alternatively, the users can switch to a linear mode. In that case, the neighborhood of the query position is defined as a sequence interval. It is equivalent to the viewing frame used in classical (i.e. one dimensional) genome browsers. This mode also allows the users to retrieve all SNPs present in this one-dimensional neighborhood.

The third mode enables the users to highlight transcription factor binding sites in the viewer. TFBSs are represented as colored regions of the DNA chain. The color indicates the intensity of the Chip-Seq experiment (green for low and red for high). The user can access detailed information about the Chip-Seq data by clicking on the TFBS region, or access UCSC genome browser records through a hyperlink.

We linked 3DGB to a 2D genome map viewer [208] to help users navigating the genome. Upon request, users can open a secondary window that displays a linear representation of the genome. The latter highlights the closest 2D genomic position to the center of the current 3D cell. The two windows are dynamically linked. A change of 3D coordinates or 2D position in either window updates automatically the position/coordinates in the other one.

Finally, we also implemented a distance calculation tool that enables the users to automatically determine the physical distance between two points in space. We intentionally did not use physical units, but instead rely on the model coordinates. Indeed, the determination of physical distances requires to interpret experimental data and make approximations which are often subject of discussions. By contrast, we believe that arbitrary units allow the users to estimate relative differences and leave them the freedom to interpret the experimental data used to obtain the 3D model.

Visualization of custom genotyping data

An important feature of our viewer is to enable users to map their private genotyping data onto reference 3D architectures, and allow them to visualize the data within our browser. This functionality is intended to provide users with tools to identify geometrical dependencies in custom genotyping data sets. The query interface allows users to upload a local file containing genotyping data. In order to prevent any formatting issues, we implemented a program to validate and convert most standard genotyping data file.

Once uploaded, the users can browse and query the 3D genome as described above. In addition to the reference data stored in our database, the users can now access simultaneously the reference SNPs collected from [204] together with those stored in the local file. To prevent any privacy issues, user data are stored locally and not transmitted to our server. A similar solution has been adopted by the UCSC genome browser [209].

2.5 Results

2.5.1 Experimental setting

To evaluate the performance of our system, we used sequence read alignments of chromosome 11 available from the 1000 Genomes project [204]. This data consists of short (around 100 bases) DNA sequence reads, mapped onto the Human reference genome. We used 1.5 billions of records, which constitute 250GB of raw data.

All data have been stored in a cluster (Hadoop version 2.2.0) of 10 machines. Worker nodes were commodity machines with Quad-Core Intel i7-2600 CPUs @ 3.40GHz, 8GB of DDR3-1600 RAM, 500GB Serial ATA HDD, running Ubuntu 12.04.2 LTS. The index node was similar, but with 16GB RAM. The replication factor was set to 3.

The main metric we take into account is response time (latency). As a matter of fact, execution time depends on the amount of records to be returned. In our context,

we considered simple, uniform and fixed-size cube queries returning from 100 to 1000 records.

2.5.2 Benchmarking against the PostGIS database

The performance of storage systems can be characterized by their speed to access the data (i.e. by the average time needed to execute a query) and the influence of the size of the output on the time required for returning a response. In this section, we evaluate the performance of 3DBG compared to the PostGIS database (http://postgis.net/) to store and query 3D neighborhoods of a genome.

We uploaded in the database a data set of ~1.5 GB (gigabytes) that contains the 3D coordinates of reference points of a simulated model of the human genome [184]. All these positions were indexed in the database using the spatial index (the description of PostGIS's spatial index can be found at http://revenant.ca/www/postgis/workshop/indexing.html). Then, we measured the speed of reaching the data through the Java application using the PostGIS JDBC driver, and the influence of the size of the output (results of query) on the processing time.

In our experiments, we queried for all different reference points available in the model from [184], and called the database to get all points that were stored in the cube centered around the current reference point, with a constant edge size (100, 200, 300 and 400 base units). Our results are shown in Figures 2.3 and 2.4.

Figure 2.3 shows the speed of accessing the data. Here, PosGIS has on average a query execution time well above 300 ms, and thus well above the time to gracefully retrieve and visualize data dynamically for on-line 3D browsing. By contrast, when we run the same experiment with 3DBG, the access time is clearly below this threshold. This observation demonstrates that 3DBG performs satisfactorily to visualize the 3D space at high resolution, while the latency of standard solutions such as PosGIS is too high, even for relatively small data sets.

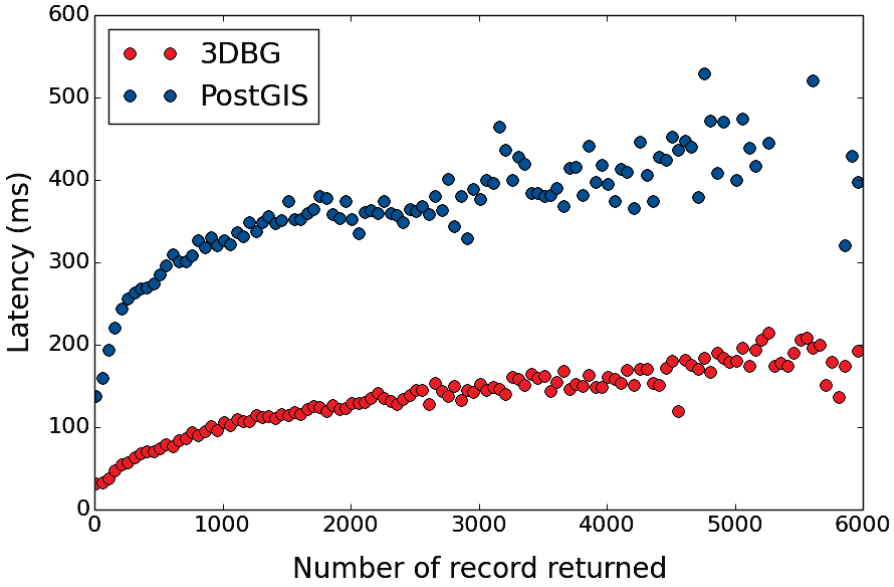

FIGURE 2.3: Comparison of 3DBG and PostGIS query latencies. The x-axis shows the number of records returned and the y-axis shows the latency in milliseconds (ms). Red dots are 3DBG data and blue dots PostGIS data.

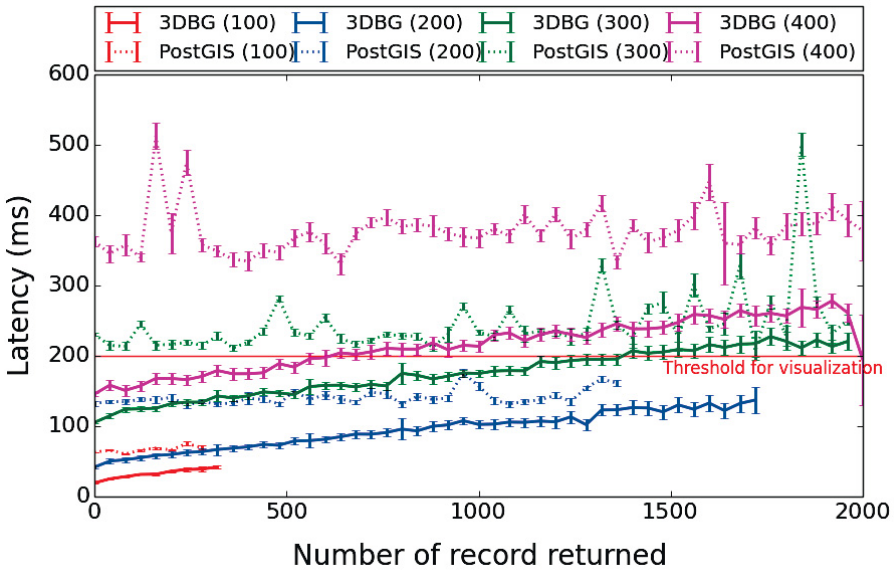

FIGURE 2.4: Dependencies of 3DBG and PostGIS latencies with query size. The x-axis shows the number of records returned and the y-axis shows the latency in milliseconds (ms). 3DBG data are represented with full lines, and PostGIS data with dotted lines. The colors of the curves are associated with the different sizes of the query (edge sizes of the cube varying from 100 to 400 base units). The latency threshold for real-time visualization (200 ms) is indicated with a horizontal red line.

Next, for each edge size (size of the neighborhood delimited by the cube query), we plot in Figure 2.4 the relation between the processing time (i.e. latency), the size of the neighborhood that we wish to explore and the number of records returned. Experiments were repeated five times to obtain the variances. Here again, we observe that PostGIS yields unsatisfactory latencies, which rapidly grow as we retrieve more data and increase the size of the query. In particular, PostGIS latencies for large queries (edge size of 400 units) exceed the threshold required for real-time visualization (~200 ms), while 3DBG performs satisfactorily. This is an important aspect because a comfortable browsing (volume and resolution of data retrieved) requires large query (in our 3D genome browser 3DGB, we use cubes of 400 units for the 3D structure from [184]). Finally, it is worth noting that the data returned by 3DBG are also already sorted, which is not the case for PostGIS.

2.5.3 Benchmarking against the Hbase database

To compare the performance of our back-end solution with Hbase (http://hbase.apache.org/), we installed an Hbase cluster (version 0.96.1) on our experimental infrastructure. We also split all data according to our index for HBase, but used a standard HBase database rather than our own chunk storage. The caching for the Hbase cluster was switched off to ensure valid results. Figure 2.5 shows the results. As can be observed, the execution times of our system are much lower than those of HBase. 3DBG outperforms HBase for relatively small queries (left of the graph), thus ensuring a smooth navigation from the client side. Overall, both systems scale gracefully, thanks to the indexing and clustering offered by our adaptive 3D index.

2.5.4 Using 3DGB to explore the 3D neighborhood of a gene

We illustrate the usefulness of 3DGB with an exploration of the 3D neighborhood of the Retinoblastoma 1 gene (RB1) - a tumor suppressor gene that has been associated with many types of cancer. This experiment does not necessarily intend to provide

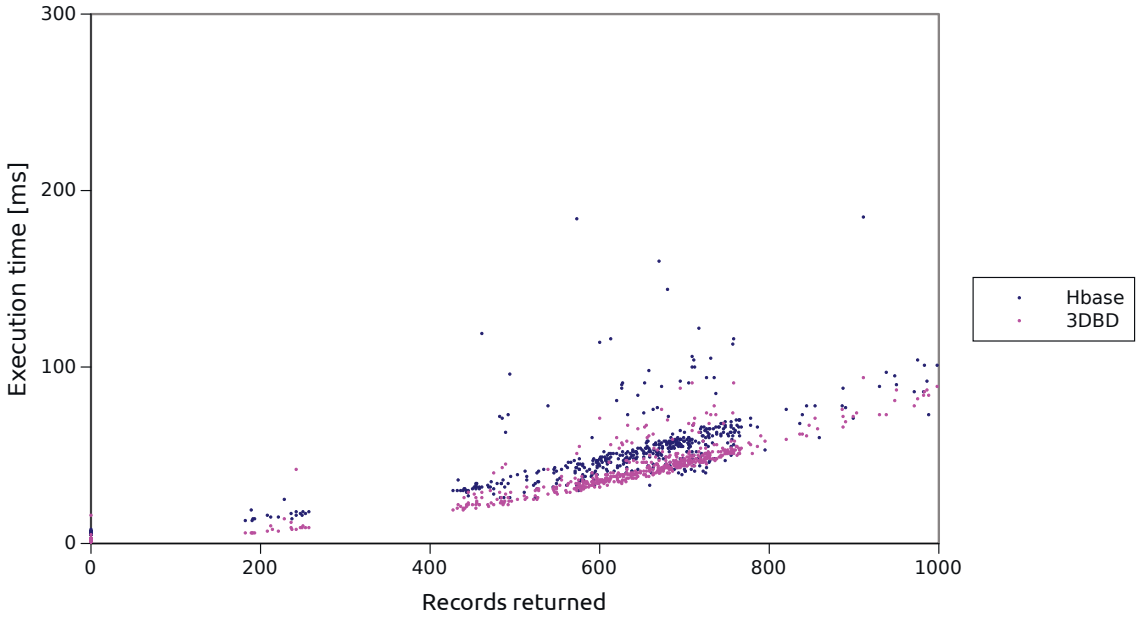

FIGURE 2.5: Comparison of 3DBG and HBase query latencies. The x-axis shows the number of record returned and the y-axis the latency in milliseconds (ms).

new insights into RB1 regulatory mechanisms, but aims to demonstrate what type of novel information can be obtained with the use of 3DGB.

We started our investigation by exploring a 3D neighborhood centered on the promoter of the RB1 gene in the 3D structure of chromosome 13 in normal B-cells (GM06990) [191]. We retrieved the list of SNPs found in the promoter region of RB1, and in other DNA strands that are not in the immediate sequence neighborhood of RB1 promoter.

2.5.5 Distribution of SNPs in the 3D neighborhood of RB1

In addition to the promoter region, we found three other strands in the spatial vicinity of RB1: $S1(44762907, 45379923)$, $S2(57184305, 57531250)$ and $S3(58747059, 59087738)$. These strands are located in a radius $R = 0.2$ of RB1 transcription start site, which corresponds approximately to 88Å.

A total of 1199 SNPs were identified in this 3D neighborhood, for which we retrieve their associated phenotype from [195]. A complete list of these SNPs with associated phenotypes is available in the supplementary data. As expected we identified many SNPs related to various types of cancer. However, another interesting finding has been to detect the occurrence of one SNP (rs10492604) related to sleep disorders in the strand $S3$. Importantly, we found only two SNPs related to sleep disorders in the whole chromosome. Moreover, with a distance of 230 Åfrom RB1 transcription start site ($R = 0.53$), this other SNP (rs10492507) is also in the vicinity of RB1 gene.

Previous studies have identified that children with hereditary retinoblastoma have also an increased risk of developing trilateral retinoblastoma [210]. Trilateral retinoblastoma is the combination of retinoblastoma (usually bilateral) and pineoblastoma (a tumor in the brain's pineal gland). The pineal gland secretes multiple hormones (including melatonin) that are implicated in the regulation of sleep patterns in seasonal and circadian rhythms [211].

Although our finding does not imply any causation, it suggests possible interesting genetic relationships between retinoblastoma and sleep disorders. The scripts used in

this experiment are available at http://3dgb.cs.mcgill.ca/scripting.html.

2.6 DISCUSSION

We presented 3DBG, a novel storage paradigm and database system to store and query genomic data in a 3D space, and developed a lightweight 3D genome browser to visualize and navigate these data from any internet browser.

We compared 3DBG to existing systems and demonstrated that our technology enables us to significantly lower the latency of spatial queries. Importantly, we also showed that our system scales gracefully when handling more data. This technology aims to develop the infrastructure needed to mine big data sets generated by new large-scale genomic studies, and to prepare the next generation of genome browsers.

Although this paper focuses on the technical description of the database system and the evaluation of its performances, we designed 3DBG to permit complex queries in the 3D space. In particular, we also aim to use our system to extract spatial relationship between genomic elements in genome-scale studies, for example using efficient batch-oriented operations a la MapReduce on top of our data chunks (implementing such features is easy, as our whole system is based on the lower levels of the Hadoop / Yarn stack). An example of such queries could be to retrieve all pairs of enhancers / promoters that are co-localized in the the 3D genome structure.

Finally, even though we did not specifically tailor 3DBG to optimize the storage space, 3DBG is already at least as efficient as existing systems. Further versions of the database will integrate compression techniques in order to reduce the space requirements and further reduce query latencies.

2.7 FUNDING

Genome Canada and Genome Québec (Bioinformatics and Computational Biology competition, in part); Canadian Institutes of Health Research [CIHR BOP-130836 to

J.W. and M.B.]; Natural Sciences and Engineering Research Council of Canada Discovery [NSERC RGPIN 386596-10 to J.W.]; Swiss National Science Foundation [200021_143649 to PCM]. Funding for open access charge: Genome Canada / CIHR funding.

Chapter 3

Collaborative Solving in a Human Computing Game Using a Market, Skills and Challenges

Olivier Tremblay-Savard [2], Alexander Butyaev [1], Jérôme Waldispühl [1]

[1] School of Computer Science and McGill Center for Bioinformatics, McGill University, Montréal, Québec, Canada

[2] University of Manitoba, Winnipeg, Canada

3.1 Preface

Data visualization helps users to understand and interpret data. Although, there exist many semi-automated approaches for data mining [212, 213, 214], the task is still difficult for a computer program solely [11]. Combined with the recent growth of computational resources available for researchers, nowadays it is easier than ever to collect a large amount of scientific data. For example, recent advances in high-throughput genomics (e.g., Hi-C technique) allowed to generate enormous amount of genomic data [215, 216]. Hence, the bottleneck is no longer gathering data but its processing. While algorithms are yet helpful, there are many tasks where human still exceeds a computer. Whereas expert data analysis is often preferred, a manual expert annotation is not scalable to large datasets. In contrast, crowdsourcing provides a pool of non-expert workers that individually contribute to the solution of a task. Yet, a better understanding of how people collaborate on a large scale is needed.

Although, the crowdsourcing and human computation techniques are not novel (e.g., Math Table Project, 1938 [217]), they gained momentum at the beginning of the XIX century due to the rapid development of Internet that gave researchers an access to its constantly growing active user base [80]. The techniques employ human workers to fulfill the gap in capabilities of automated solutions using various motivators to achieve the goal. While money was proved to be a very efficient stimulus for a user [29, 27, 218, 28]; knowledge [49, 18], curiosity [42, 46], entertainment [74, 71, 77], and the competition [53, 56, 58] were shown to improve the audience retention while having a positive effect on the produced results. Besides, most of the crowdsourcing and human computing systems follow the divide-and-conquer strategy (split a problem to the microtasks and distribute it to participants) [80], which often fosters groupthink [219] and rarely promote collaborative solving of the problem.

This chapter investigates the collaborative aspect of crowdsourcing systems. In particular, we focus on understanding how humans collaborate, how this phenomenon benefits a system, what incentives are needed to promote collaboration, and how to

make an interaction between participants more efficient. We design the crowdsourcing game that creates a collaborative environment for solving a predefined puzzle using multiple gamified elements including a Market, Skills, and Challenges. It presents qualitative and quantitative analyses of the players' behavior during the game sessions as well as their interaction both with other players and the game elements.

The remainder of text found in this chapter is taken from:

> Olivier Tremblay-Savard, Alexander Butyaev, and Jérôme Waldispühl. "Collaborative Solving in a Human Computing Game Using a Market, Skills and Challenges." *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play.* ACM, 2016.

O.T.S. and A.B. contributed equally to the development of the text. A.B. developed the back end and data management sections of the Market game, contributed to the design of the front end of the game and data analysis, helped to prepare draft of manuscript. O.T.S. developed the front end of the game, arranged the game sessions, outlined and performed data analysis, prepared the draft of the manuscript. J.W. coordinated the game design and computational analysis

3.2 Abstract

Using a human computing game to solve a problem that has a large search space is not straightforward. The difficulty of using such an approach comes from the following facts: (i) it would be overwhelming for a single player to show him or her the complete search space and at the same time, (ii) it is impossible to find an optimal solution without considering all the available data. In this paper, we present a human computing game that uses a market, skills and a challenge system to help the players solve a graph problem in a collaborative manner. The results obtained during 12 game sessions of 10 players show that the market helps players to build larger solutions. We also show that a skill system and, to a lesser extent, a challenge system can be used to influence and guide the players towards producing better solutions.

Keywords: Human computing | Collaboration | Crowdsourcing | Graph problem | Game | Market | Trading game | Skills | Challenges

3.3 Introduction

Human-computation and crowd-sourcing are now perceived as valuable techniques to help solving difficult computational problems. In order to make the best use of human skills in these systems, it is important to be able to characterize the expertise and performance of humans as individuals and even more importantly as groups.

Currently, popular crowd-computing platforms such as Amazon Mechanical Turk (AMT) [220, 221] or Crowdcrafting [43] are based on similar divide-and-conquer architectures, where the initial problem is decomposed into smaller sub-tasks that are distributed to individual workers and then aggregated to build a solution. In particular, these systems prevent any interaction between workers in order to prevent groupthink phenomena and bias in the solution [222].

However, such constraints are necessarily limiting the capacity of the system to harness the cognitive power of crowds and make full benefit of collective intelligence.

For instance, iterative combinations of crowdsourced contributions can help enhancing creativity [223]. The usefulness of parallelizing workflows has also been suggested for tasks accepting broad varieties of answers [224].

The benefits of developing recommendation systems or coordination methods in collaborative environments has been demonstrated [225, 226, 227]. Therefore, in order to gain expressivity and improve their performance, the next generation of human-computation systems will certainly need to implement mechanisms to promote and control the collaboration between workers. Nonetheless, before transitioning to this model, it is important to first estimate the potential gains in productivity, and quantify the usefulness of the mechanisms and incentives to promote collaborative solving and prevent groupthink.

Historically, computation on graphs has proven to be a good model to study the performance of humans in solving complex combinatorial problems [228]. Experiments have been conducted to evaluate the dynamics of crowds collaborating at solving graph problems [229] but still, little is known about the efficiency of the various modes of interaction.

In this paper, we propose a formal framework to study human collaborative solving. We embed a combinatorial graph problem into a novel multiplayer game-with-a-purpose [230, 231], which will be used to engage participants and analyze collective performances. More precisely, we design a market game in which players can sell and buy solutions or bits of information, and couple this platform with (i) a skills system to enhance the efficiency of specific gaming strategies and (ii) a challenge systems to guide the work of the crowd. We use this game to investigate the validity of the following hypotheses.

3.3.1 Hypotheses

The development of the game with its three main features, *i.e.* the market, the skills and the challenge system, was based on those four hypotheses:

1. A market system will help the players build better solutions.
2. A skill system is useful to orient the players into doing specific actions that are beneficial to the game and other players.
3. A challenge system is effective in encouraging the players to do a specific action in the game.
4. The collected solutions are better when all the three features are present in a game session, independently of the players' personal skills.

To answer these questions, we conducted a study on 120 participants using different variants of our market game. Our results confirm the benefits of using a trading platform to produce better solutions. Interestingly, we also found that a skills system helps to promote actions that are favorable to the collective solving process, but that the efficiency of a skill is reduced if it is designed to help solve one of the primary objectives of the game. Finally, we observed that a precise parametrization of challenges (i.e. finding an appropriate difficulty, nor too easy, nor too difficult) is required to result in an improvement of the quality of the collective work.

Our game is freely available at `URL:TBA`, and can be used as a platform for further independent studies.

3.4 Problem

The game was implemented to solve a graph problem, which is the problem of finding maximal cliques in a multigraph. Let $G(V, E)$ be a multi-colored graph, where each vertex $v \in V$ has a set of colors $c(v)$. There is a colored edge $e = (v, u) \in E$ between the vertices v and u for every color in $c(v) \cap c(u)$ (*i.e.*, one for every color that they have in common). In other words, there is no colored edge between two vertices v and u for which $c(v) \cap c(u) \neq \emptyset$. Let $|C|$ be the total number of colors in the graph. The problem is then the one of finding maximal cliques for each possible n number of colors (where $1 \leq n \leq |C|$), *i.e.* cliques in which all the edges (and vertices) have

the same n colors. A simple exact algorithm can solve the problem in $O(|V|2^{|C|})$. We make the conjecture that it is also the worst time complexity of the problem.

This problem was chosen for two reasons. First, it can be solved quickly by a computer when the number of colors is small, thus making it possible to compute the exact solution and measure the percentage of the solution that is found by the players in a game session. Second, this problem can easily be translated into a color matching game, which takes advantage of the ability of human perception. Indeed, since the colored edges between the vertices are given implicitly by the colors of the vertices, it is possible to show the players only the colored vertices. To solve the problem, the players have to find the largests sets of circles with colors in common, for all possible subsets of colors.

Note that it is not our goal to compare the performance of players with the performance of computers in solving this problem. With a limited number of colors (like six in our tests), the exact algorithm can solve the problem in seconds. For this study, we required a problem that was structured enough so that we could easily calculate the optimal solution and evaluate the performance of the players depending on what features were on or off and also the effect of the different features on the quality of solutions.

3.5 Presentation of the game

3.5.1 Goal of the game

The main objective of the game is to build *sequences* (*i.e.* sets) of circles (circles represent vertices of the graph) that (i) are as long as possible and (ii) contain as many colors in common as possible. Circles used by the players to build the sequences either come random packages bought from the system or they come from another player through the market. The sequences can then be sold to the system for a certain amount of game

money, which is determined by a scoring function that takes into account the length and the number of colors in common of the sequence.

3.5.2 Scoring function

The score of a sequence sold to the system is equal to $baseScore_n * seqLength^2$, where $baseScore_n$ is a base score depending on the number of colors in common (see Table 3.1) and $seqLength$ is the length of the sequence. The base scores were calculated based on the exact solution for the graph that was generated for the tests (see section Generating the graph for a description of the graph that was used) in such a way to give a reward that is proportional to the difficulty of building the sequence. More precisely, we calculated the average length L_n of all solutions for each n number of colors. The base score is simply the reciprocal of this average $(1/L_n)$ multiplied by a balancing factor (505 in our case). The balancing factor was chosen in order to get a score of 500 for a sequence of length 10 with only one color in common, which is exactly the price of two random packages of circles. Also notice that the value of a sequence is exponential in relation to its length, which is to encourage players to build the longest possible sequences.

3.5.3 Game interface

The game client and the server were built in Java 1.7. As shown in Figure 3.1, the game interface can be divided into 3 parts: the player information panel, the game panel and the market panel.

A: Player information panel

This panel simply contains information on the player's wallet, the current level of the player and has three buttons, allowing the player to open dialogs showing information on the current challenge, the skills (see section Skills for a description of the available skills) and the leaderboard. One experience point is given to the player for each game

TABLE 3.1: Value of the base score depending on the number of colors in common

Number of colors	Base score
0	0
1	5
2	14
3	26
4	40
5	55
6	72

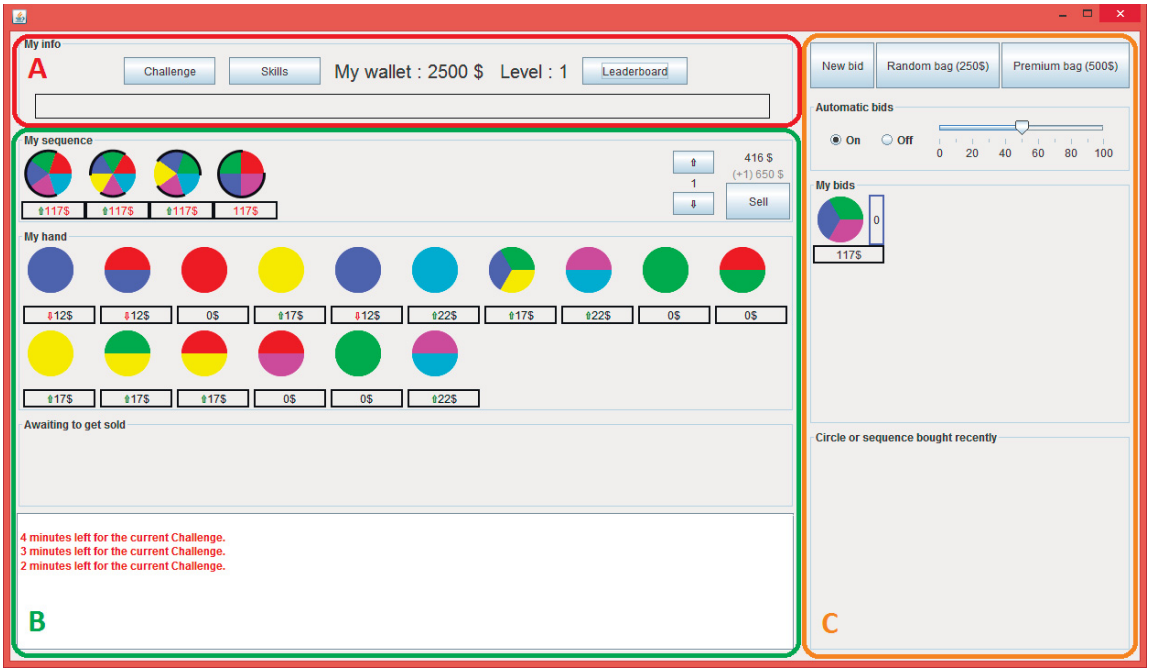

FIGURE 3.1: The game interface, separated in three panels. Panel A (inside the red box) is the player information panel.Panel B (inside the green box) is the game panel. Panel C (inside the orange box) is the market panel.

dollar that he/she wins. The player can lose game money, but cannot lose experience points (experience points can only go up). Every time a player levels-up, he/she gets a skill point that can be used to improve any of the skills.

B: Game panel

The first component of the game panel is the 'My sequence' panel, which shows the current sequence that is being built by the player. The maximum size of a sequence is 10. Colors in common in the sequence are indicated by a thick black border surrounding the colors in the circles. Players can use the arrows to switch between the different sequence slots (2 sequence slots are available at the start of the game). The current value of the sequence is shown at the right, and the price for adding one more circle with the same colors in common is shown right below in gray. Finally, the sell button allows the player to sell the current sequence to the system: the sequence then disappears and the money is given to the player. Selling a sequence is equivalent to submitting a solution to the system.

The second component is the 'My hand' panel, which can contain up to 20 circles. Players can add a circle to the sequence by clicking on it. Circles are represented by their colors and by a price label (in a black box). The price corresponds to the current value of the circle on the market. Clicking on the price label sells the circle to the highest bidder on the market. Circles that are bought from a random package or from other players are sent to the hand.

The 'Awaiting to get sold' is where the circles are sent just before being sold to the highest bidder. If the bid disappears before the transaction is completed, the 'sold' circle will stay there. The player can then click on it to cancel the selling and put it back in the hand.

Finally, the bottom panel is a news feed, showing information on the game state, like the remaining time to complete the challenge and the last transactions completed by the player for example.

C: Market panel

At the top of the market panel, buttons allow the player to create bids for circles or to buy random packages (or bags) of circles. The 'Random bag' costs $250 and contains 5 circles with fewer colors. The 'Premium bag' costs $500 and contains 5 circles with a higher chance of getting circles with many colors.

Right below the buttons is the 'Automatic bids' panel, which allows the player to get automatic bids for circles corresponding to the sequences that he or she is building. A percentage of profit for the price of the automatic bids can be set with the slider. The profit is defined as the money the player would make by adding one more circle with the same colors in the current sequence (difference between the gray and black prices above the Sell button).

The 'My bids' panel shows all the bids that the player currently has on the market. The bid price is shown below the circle (in the black box). On the right side of the circle is the number of sequences with the same colors that the player can buy from other players (in the blue box). Clicking on the blue box opens a window showing the list of sequences that can be bought. Buying a sequence from another player is called a 'buyout' (see the following subsection for a more detailed description of buyouts).

Finally, the last panel at the bottom shows the last circle or sequence that was bought by the player.

3.5.4 Market

The market has three functions: (i) allow the players to exchange circles through a bidding system, (ii) allow players to buy sequences built and sold by other players so that they can be improved, and (iii) merge together sequences of length 10 to create super circles that are then put back in the game.

For every subset of colors, the server has a list of all the bids that are currently on the market. The value of the highest bid on the market is shown below every circle

under the possession of the players. When a circle is sold by a player, it is sent through the server to the highest bidder.

Buyouts work differently. Players cannot bid on sequences, but the server holds for two minutes all the sequence that have been sold by the players. During those two minutes, other players can buy the sequences for a price that is equal to 150% of the initial score of the sequence. When a buyout is made, the bonus game money is sent to the player who initially sold the sequence to the system.

Finally, the game system creates a super circle every time a sequence of length 10 is sold by a player. A super circle of level 2 (representing 10 circles) counts as two circles when put in a sequence. Super circles can be of any level (a sequence of 10 super circles of level 2 form a super circle of level 3, and so on). The idea behind the creation of the super circles was to remove the limitation of the maximum sequence size imposed by the game interface.

3.5.5 Skills

Four different skills were implemented in the game. One skill point is awarded to a player when he or she levels up, which can then be put in any of the four skills. The maximum level of each skill is equal to six (there are six levels of bonuses). Each skill was put in the game as a way to guide the player in doing actions that are beneficial to the system or to the other players:

- *Buyout King*: lowers the price of buying a sequence from another player (goal: encourage buyouts);
- *Color Expert*: gives a bonus to selling sequences that have more than one color in common (goal: push players to build more multicolored sequences);
- *Sequence Collector*: gives an additional sequence slot (goal: give more space to encourage the creation of longer sequences with more colors in common);

- *Master Trader*: gives a bonus to selling circles to other players (goal: promote the selling of individual circles).

3.5.6 Challenge system

We implemented a challenge system that analyzes the recent actions of the players and creates a new challenge every five minutes. The five challenge types are:

- *Sell/buy circles*: requires the players to sell or buy circles;
- *Buyout sequences*: requires the players to buy sequences from other players;
- *Minimum number of colors*: requires the players to sell sequences with at least a certain number of colors in common;
- *Minimum sequence length*: requires the players to sell sequences with a minimum sequence length;
- *Specific colors in common*: requires the players to sell sequences with a specific subset of colors in common.

Basically, the system continuously monitors the activities of the players and decreases or increases the probabilities of each challenge type. The next challenge is then selected using a multinomial sampling on these probabilities. The number of times T that the challenge-related action must be completed is selected randomly between 2 and 4. The prize that is awarded for completing the challenge is equal to $\$1500 * T$.

3.6 Experiments

3.6.1 Independent and dependent variables

In the context of this study, there were three independent variables: the market (present; not present), the skills (present; not present) and the challenges (present; not present).

Instead of trying all 8 possible combinations of independent variables, we decided to focus on four game conditions:

1. All features present (or A)
2. Everything except the market, hereafter referred to as "No Market" (or NM)
3. Everything except the skills, hereafter referred to as "No Skills" (or NS)
4. Everything except the challenges, hereafter referred to as "No challenges" (or NC)

Focusing on those four playing conditions allowed us to repeat each experiment more times with different groups of players. Moreover, the goal was to evaluate the importance of every game feature by removing them one at a time and evaluating the effect on the results obtained by the players.

As for the dependent variables, we were interested in measuring the following:

1. Percentage of the problem solved
2. Total experience points earned by the players
3. Average sequence length of the sequences created by the players
4. Average number of colors in common of the sequences created by the players
5. Proportion of sequences of more than one color in common created by the players
6. Number of circles sold individually to another player
7. Number of sequences bought from other players (buyouts)

3.6.2 Game sessions

We recruited 120 people in total to test our game. We divided the participants into groups of 10 and repeated three times each of the four game conditions presented in the previous subsection. Each participant was playing the game for the first time,

except for some people that were invited as replacements to deal with last minute cancellations. Before starting each game session, the players were shown a document explaining the rules of the game and the interface. They were also asked to fill in a questionnaire so that we could get information on the participants, such as their age, their abilities at puzzle solving and their experience with video games for example. For all the experiments, the game session lasted 45 minutes.

3.6.3 Generating the graph

We generated one random colored multigraph that we used for all the 12 tests. Since the edges in the graph depend entirely on the colors of the vertices, it is sufficient to generate only the colored vertices. For the tests, a graph containing 300 vertices and 6 different colors was generated. To randomly select the number of colors for each vertex, a geometric distribution of parameter $p = 0.5$ was used, so that the vertices with a lot of colors are rarer. Once the number of colors was selected for the vertex, the set of colors was selected uniformly.

3.7 Results and Discussion

3.7.1 Testing hypothesis 1: the efficiency of the market

The market system we implemented in the game allows the players to exchange circles and partial solutions (in the form of buyouts). The main goal of the market is to help the players in building longer sequences.

As shown in Figure 3.2, the three game sessions in which we had the lowest average of sequence lengths (for all the sequences sold by all the players) are the ones that were played without the market, with averages of 4.40 for NM, 4.19 for NM-2 and 4.63 for NM-3. Even if we consider the super circles (the special circles that are actually 10 circles combined into one), the average sequence lengths for those three sessions are still the lowest ones, with values of 4.90 for both NM and NM-2, and 5.40 for NM-3.

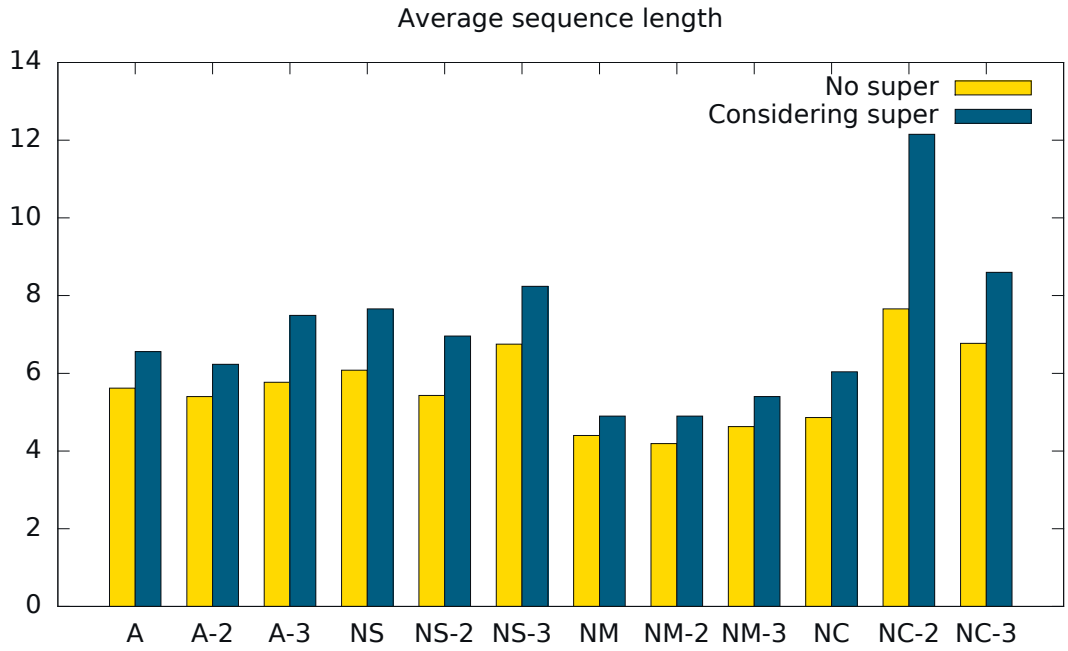

FIGURE 3.2: Average sequence length for every game session, not considering the super circles and considering the super circles (e.g. a super circle of level 2 in a sequence represents 10 circles in the solution). 'A', 'A-2' and 'A-3' represent the tests with all the features on; 'NS', 'NS-2' and 'NS-3' represent the tests without the skills; 'NM', 'NM-2' and 'NM-3' represent the tests without the market; 'NC', 'NC-2' and 'NC-3' represent the tests without the challenges.

Since the distribution of the lengths for all the sequences sold to the system during a game session do not follow a normal distribution, we used a non-parametric test (Kruskal-Wallis) to verify if the sequence lengths of the different game sessions seem to come from the same distributions. The Kruskal-Wallis test revealed a significant effect of the game conditions on the sequence lengths without considering super circles ($\chi^2(11) = 1391.7$, $p < 2.2E-16$) and also when considering super circles ($\chi^2(11) = 1388.4$, $p < 2.2E-16$).

We then made a post hoc test (Dunn's test) to do pairwise comparisons between all the groups. With or without considering super circles, all the game conditions were shown to be significantly different ($p < 0.01$), except a few shown in table 3.2. Note that the strongest similarities are found between the three 'All' groups and the three 'No market' groups. Some of the 'No skills' experiments are found to be similar to the 'All' groups, which could indicate that the presence of the skills have a very limited effect on the sequence length. The NC experiment is found to be similar to two 'No market' groups, but that can be explained by the fact the players for the NC experiment were very weak (as can be seen by the total experience gained during that session in Figure 3.11).

3.7.2 Testing hypothesis 2: the benefits of a skill system

We implemented the skill system for two reasons: (i) to give the players more incentive to accumulate experience points as fast as possible, because the reward for leveling-up is an additional skill point, and (ii) to influence indirectly the players into doing actions that are either improving the solutions collected by the system or helpful to the other players (which in the end will also improve the solutions). In our game, two skills were related to the market (*Buyout King* and *Master Trader*) and two skills were related to building sequences (*Color Expert* and *Sequence Collector*). In the following paragraphs, we will analyze how those four skills affected the strategies and actions of the players. Note that when some players lost all their money in the game, they

TABLE 3.2: Similar groups of sequence length distributions, as reported by Dunn's test. An 'n' in the table represents a similar pair when not considering super circles, and an 's' in the table represents a similar pair when considering super circles.

	A-2	A-3	NS	NS-2	NS-3	NM	NM-2	NM-3
A	n/s	n		n/s				
A-2		n		n/s				
A-3			n/s					
NC						n		n/s
NC-3					n/s			
NM							n/s	n

had to start a new game. In our results, we count both new games as if they were played by different players, since the players who restart might choose a different set of skills the second time. That explains why the total number of players is larger than 120. Players of the 'No skills' game condition were considered and put automatically in the without skill group.

Buyout King

The *Buyout King* skill allows the players to reduce the price of buying a sequence from another player (which we call a buyout). The idea behind this skill was to encourage the players to buy small sequences built by other players so that they could improve them before selling them back to the system. In other words, a buyout is the action of buying a partial solution made by another player in order to improve it.

Figure 3.3 shows statitics for the players who have put at least one skill point in the *Buyout King* skill and the players who did not use the skill at all. We were interested in the number of buyouts that the players with the skill were making compared to the rest of the players. Note that since this skill is related to the market, we did not consider the 'No market' sessions for these results.

The median value for players who spent a skill point in the *Buyout King* skill is 15, while the median value for the players without the skill is 1.5, indicating that half of the players without the skill did not use the buyout at all or used it only once. Since the distribution of the number of buyouts is not following a normal distribution (the Shapiro-Wilk test rejected the null hypothesis with $p = 2.6E - 10$), we used a Mann-Whitney's U test to compare the medians of the two groups. We found a significant effect of the presence of this skill on the medians ($U = 1629.5$, $p = 0.004$, effect size $r = 0.28$).

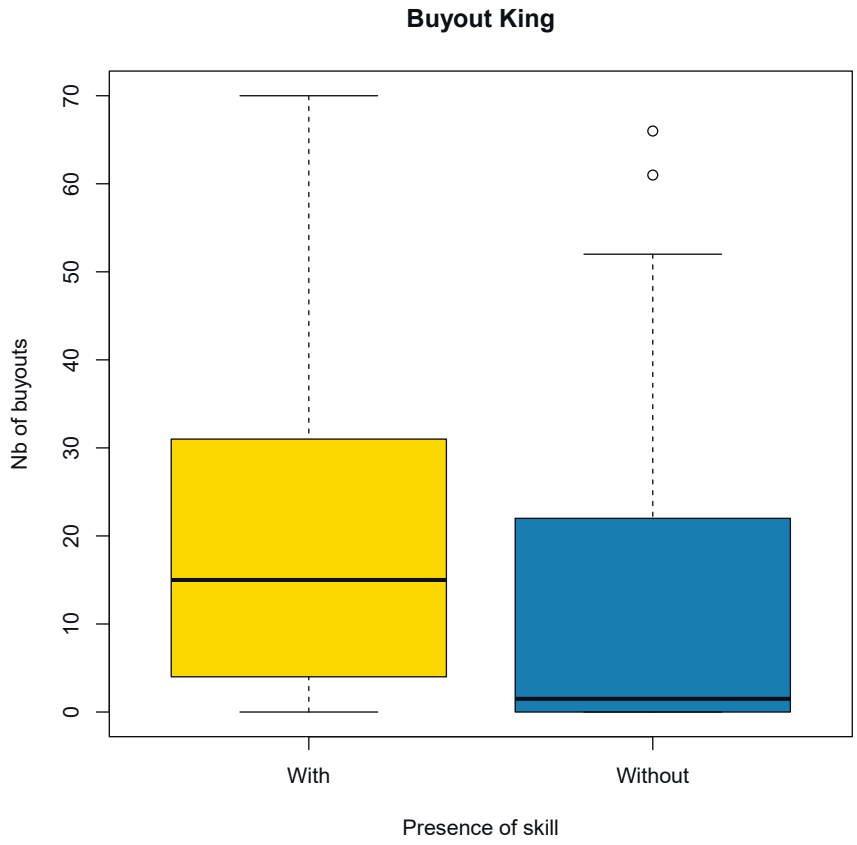

FIGURE 3.3: Boxplot of the number of buyouts made by players with (37 players) and without (66 players) the *Buyout King* skill.

Master Trader

The *Master Trader* skill allows the players to get bonus money in addition to the regular market price for each circle they sell individually. This skill was put in the game in order to increase the activity on the market by encouraging more players to send the circles that they don't need to players who need it the most.

Figure 3.4 shows statistics for the players who have put at least one skill point in the *Master Trader* skill and all the other players. We were interested in comparing the number of individual circles that were sold to another player for the two different categories. Once again, since this skill depends on the presence of the market, we did not consider the 'No market' experiments in the results shown.

The median value for the players who had selected the *Master Trader* skill (73) is more than three times larger than the one for the rest of the players (21.5). Since the distribution of the number of circles sold individually is not following a normal distribution (the Shapiro-Wilk test rejected the null hypothesis with $p = 5.3E - 13$), we used a Mann-Whitney's U test to compare the medians of the two groups. We found a significant effect of the presence of the *Master Trader* skill on the medians ($U = 1633.5$, $p = 7.2E - 4$, effect size $r = 0.33$).

Color Expert

The *Color Expert* skill gives a bonus multiplier to the scoring function for sequences with more than one color in common. This skill was implemented in order to give extra motivation to build sequences with many colors in common, since they are harder to build. Indeed, more focus is needed from the player to match many circles with more than one color in common.

In Figure 3.5, we show the comparison of the proportion of multicolored sequences sold by players with the *Color Expert* skill and players without it. Interestingly, the median values for both groups are almost identical: 0.317 (or 31.7%) for the players with the skill and 0.313 (or 31.3%) for the players without the skill. The distribution

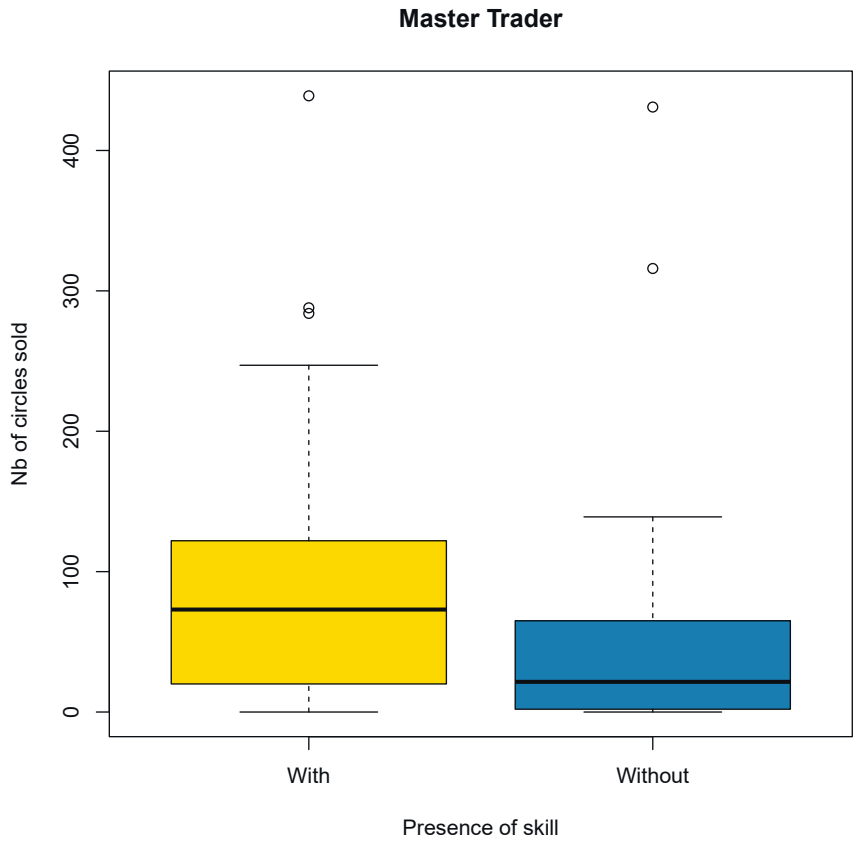

FIGURE 3.4: Boxplot of the number of circles sold individually by players with (33 players) and without (70 players) the *Master Trader* skill.

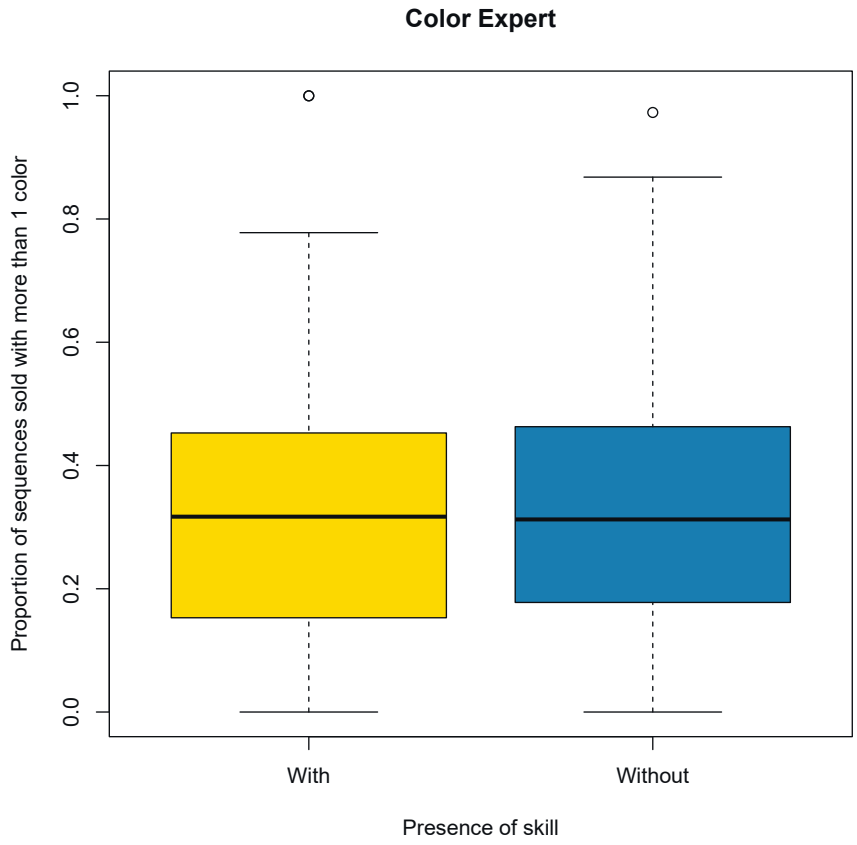

FIGURE 3.5: Boxplot of the proportion of sequences with more than one color in common sold by players with (94 players) and without (49 players) the *Color Expert* skill.

of the proportion of multicolored sequences was not normal (the Shapiro-Wilk test rejected the null hypothesis with $p = 0.3E - 4$), so we did a Mann-Whithney's U test to compare the medians. As expected, the test failed to reject the null hypothesis that the values were sampled from the same distribution ($p = 0.89$).

We conclude that the *Color Expert* skill does not affect the behavior of the players. This can be explained by the fact that one of the main goals of the game is to create sequences with as many colors in common as possible, whether the player selects this skill or not.

Sequence Collector

Every point in the *Sequence Collector* skill gives an additional slot to build a sequence. Because of the limited size of the player's hand and the limited number of sequence slots, it's hard to build long sequences with many colors in common. It is for both the sequence length and the number of colors in common that we added the *Sequence Collector* skill in the game.

We first compared the average sequence length of sequences built by players with the *Sequence Collector* skill and the ones built by the rest of players (see Figure 3.6). While the median value for the players without the skill (5.63) is a little bit larger than the one for the players with the skill (5.12), the averages of both groups are actually similar (5.61 and 5.56 in the same order). Since the distribution of the average sequence lengths were not normal (the Shapiro-Wilk test rejected the null hypothesis with $p = 0.0057$), we did a Mann Whitney's U test to compare the medians of both groups. The test failed to reject the null hypothesis that the values were sampled from the same distribution ($p = 0.69$). Thus, there is no evidence that the *Sequence Collector* skill helps players build longer sequences. This tends to confirm what we mentioned earlier (in Section Testing hypothesis 1): the presence of the skills in general does not seem to affect the length of the sequences built by players. Once again, this can be explained by the fact that selling long sequences is one of the two main goals of the game, and is one of the main components of the scoring function.

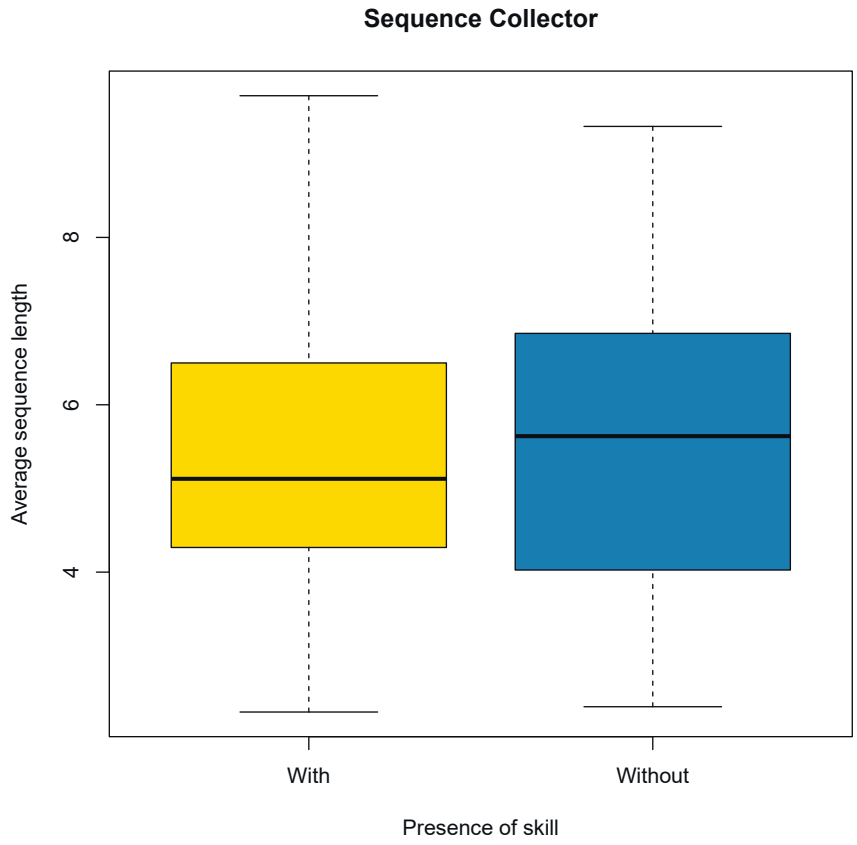

FIGURE 3.6: Boxplot of the average sequence length of sequences built by players with (60 players) and without (83 players) the *Sequence Collector* skill.

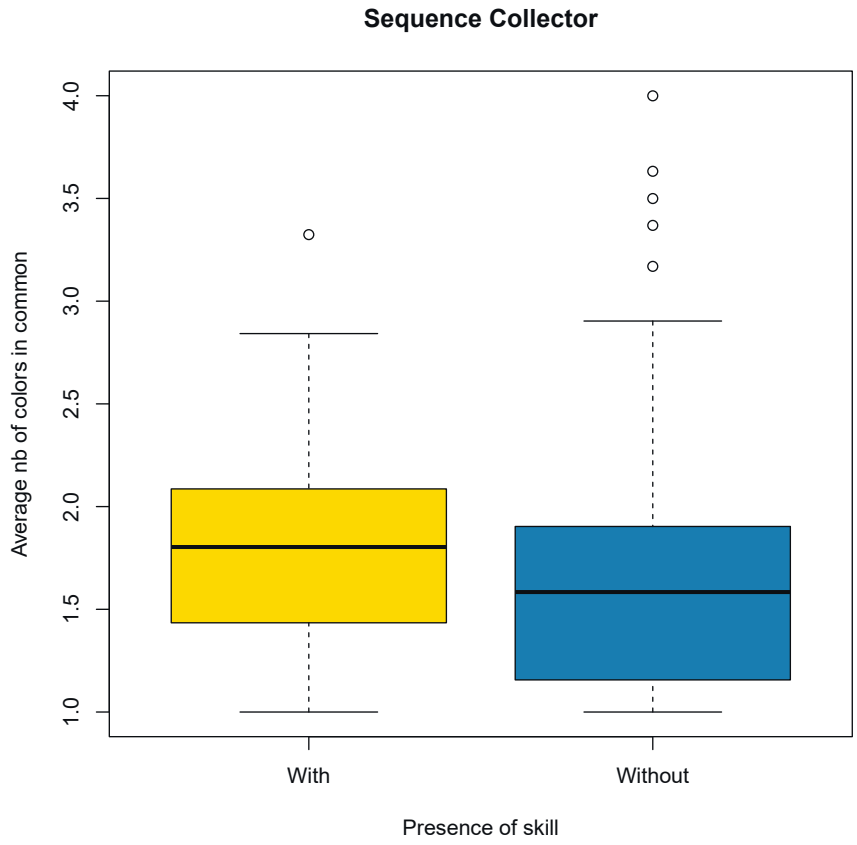

FIGURE 3.7: Boxplot of the average number of colors in common of sequences built by players with (60 players) and without (83 players) the *Sequence Collector* skill.

We then compared the average number of colors in common of the sequences built by players with and without the *Sequence collector* skill (see Figure 3.7). The median value for the players without the skill (1.58) is 12% lower than the one for the players with the skill (1.80). Since the distribution of the average number of colors in common is not following a normal distribution (the Shapiro-Wilk test rejected the null hypothesis with $p = 1.2E - 7$), we used a Mann-Whitney's U test to compare the medians of the two groups and we found a significant effect of the presence of this skill on the medians ($U = 3113$, $p = 0.01$, effect size $r = 0.21$). The *Sequence collector* skill is thus helping players to build sequences with more colors, by allowing them to store unfinished sequences of multiple colors in the additional slots until they are able to complete them.

3.7.3 Testing hypothesis 3: the usefulness of the challenge system

The challenge system was implemented to analyze the current state of the game and guide the players towards doing actions that are currently needed. As mentioned previously, five different challenge types were implemented in the game (see Section Challenge system for the complete list). In order to analyze the effect of the challenges on the way the participants were playing, for each challenge type, we compared the relevant statistics of the game during the challenge with the rest of the game session (when a different challenge was available).

Note that we are considering here only the nine sessions in which the challenges were present and that the Sell/buy and Buyout challenges were disabled during the session without the market.

Minimum number of colors challenge

To measure the effect of the *Minimum number of colors challenge* on the game, we compared the average number of colors of the sequences built by the players when the

challenge was active and when it was not. The different averages for each game session are presented in Figure 3.8. In all the game sessions except A-3 and NM, the average number of colors in common is higher when the challenge is active.

The distribution of the averages of the number of colors in common for all the game sessions considered here is normal (Shapiro-Wilk $p = 0.79$), allowing us to use a Welch's t-test to compare the means for both groups, *i.e.* 1.96 colors in common during the challenge and 1.76 during the rest of the time. The test confirmed a significant effect of the presence of the challenge on the average number of colors in common ($t(16) = 2.19$, $p = 0.04$, Cohen's $d = 1.03$).

Minimum sequence length challenge

In order to analyze the effect that the *Minimum sequence length challenge* had on the game, we compared the average sequence length during the challenge and when a different challenge was active for all the game sessions. As shown in Figure 3.9, the presence of this challenge increased the average sequence length in all the game sessions except the three sessions with all the features.

The means of all the average sequence lengths during the challenge and for the rest of the time are 5.38 and 5.08 respectively. Since the distribution of the averages of sequence lengths is normal (Shapiro-Wilk $p = 0.27$), we used a Welch's t-test to compare those means, but the test wasn't able to prove that those means are significantly different ($t(16) = 0.79$, $p = 0.44$).

Although there is not a statistically significant difference between the two groups, we can generally see a small effect for six of the nine groups with challenges. The fact that we observe the opposite effect in the three game sessions with all the features is very surprising, but hard to explain. One possible explanation could be that when all the features are present, the players have more to think about and check the challenges a little bit less.

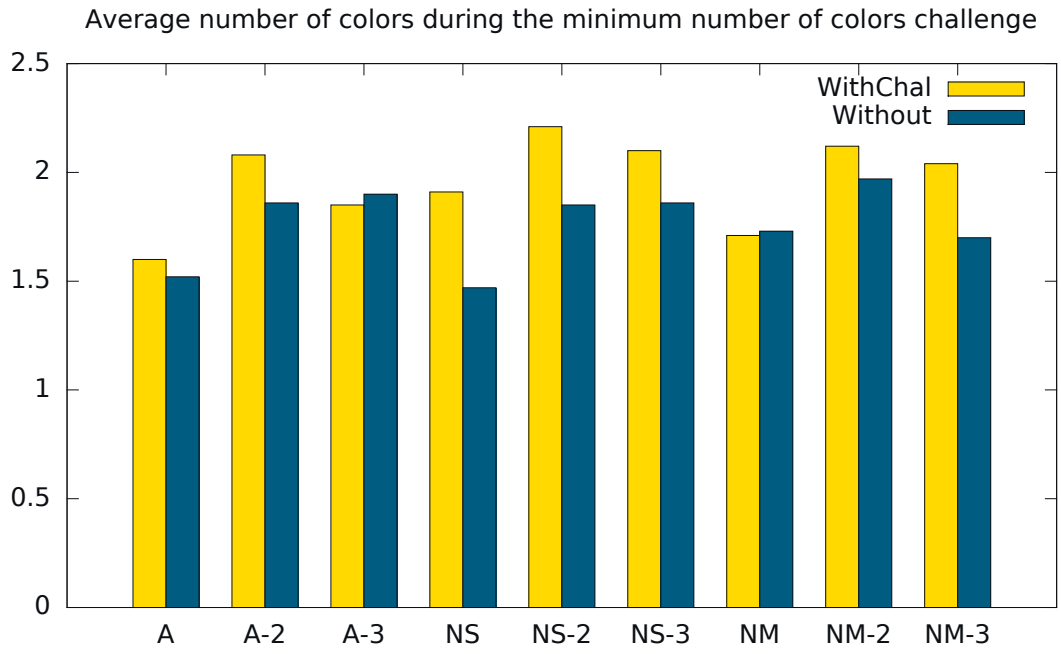

FIGURE 3.8: Average number of colors in the sequences with and without the *Minimum number of colors challenge* active. 'A', 'A-2' and 'A-3'represent the tests with all the features present, 'NS', 'NS-2' and 'NS-3' represent the tests without the skills, and 'NM', 'NM-2' and 'NM-3' represents the tests without the market.

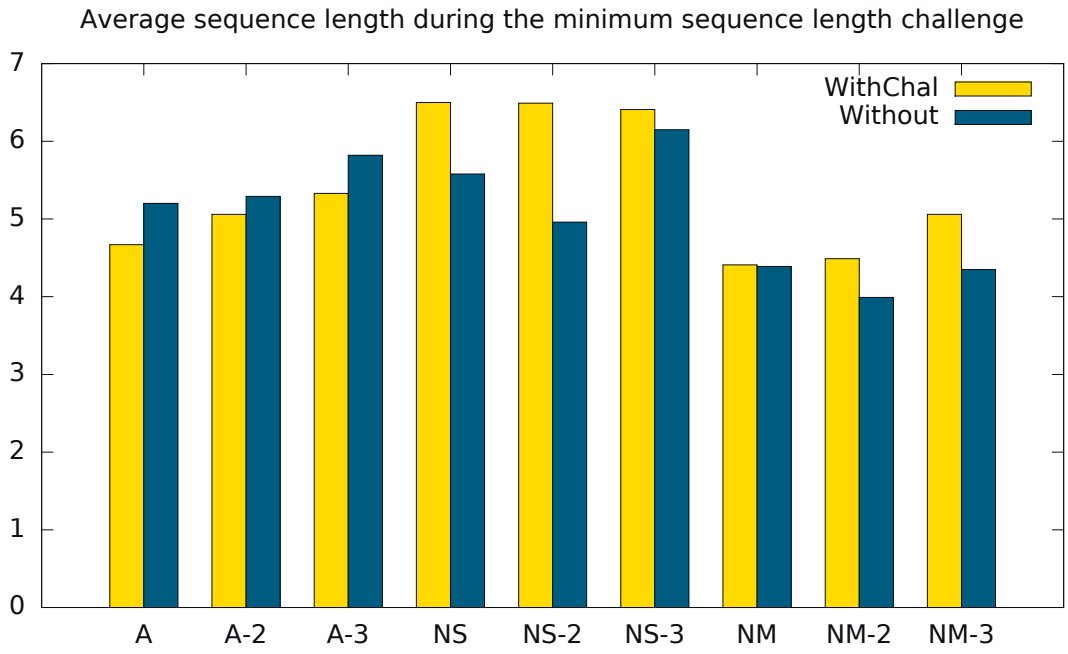

FIGURE 3.9: Average sequence length with and without the *Minimum sequence length challenge* active. 'A', 'A-2' and 'A-3' represent the tests with all the features present, 'NS', 'NS-2' and 'NS-3' represent the tests without the skills, and 'NM', 'NM-2' and 'NM-3' represents the tests without the market.

Sell/buy challenge

For the *Sell/buy challenge*, we were interested in comparing the number of individual circles sold on the market per minute when the challenge was active and when it was not. The results, presented in Figure 3.10, don't show a clear trend. Indeed, in half of the game sessions, the number of circles sold per minute is higher during the challenge, while it's the opposite for the other half of the game sessions.

Once again, the numbers of circles sold per minute in the six different game sessions follow a normal distribution (Shapiro-Wilk $p = 0.26$), so we used a Welch's t-test to compare the means of both groups, which are 13.18 during the challenge and 12.73 during the rest of the time. The t-test failed to reject the null hypothesis that both means are the same ($t(10) = 0.11$, $p = 0.91$).

We believe that the main reason why there doesn't seem to be any difference between the two groups is that most people were able to complete this type of challenge without really changing anything to their normal behavior. This challenge was simply too easy, because most of the players are always selling or buying (through the bids) at least 2 or 4 circles every five minutes (the length of a challenge).

Buyout challenge

The *Buyout challenge* appeared only once in total in all the three gaming session with challenges and with the market. Thus, we don't have a significant amount of data to analyze the effect of this challenge. The reason why this challenge almost never appeared is because players were always using the buyout, which greatly reduced the probability of showing this challenge.

Specific colors in common challenge

The *Specific colors in common challenge* is also difficult to analyze because it was completed only 8 times in total during the nine sessions with challenges, despite appearing 11 times throughout those nine experiments. This can be explained by the fact that it

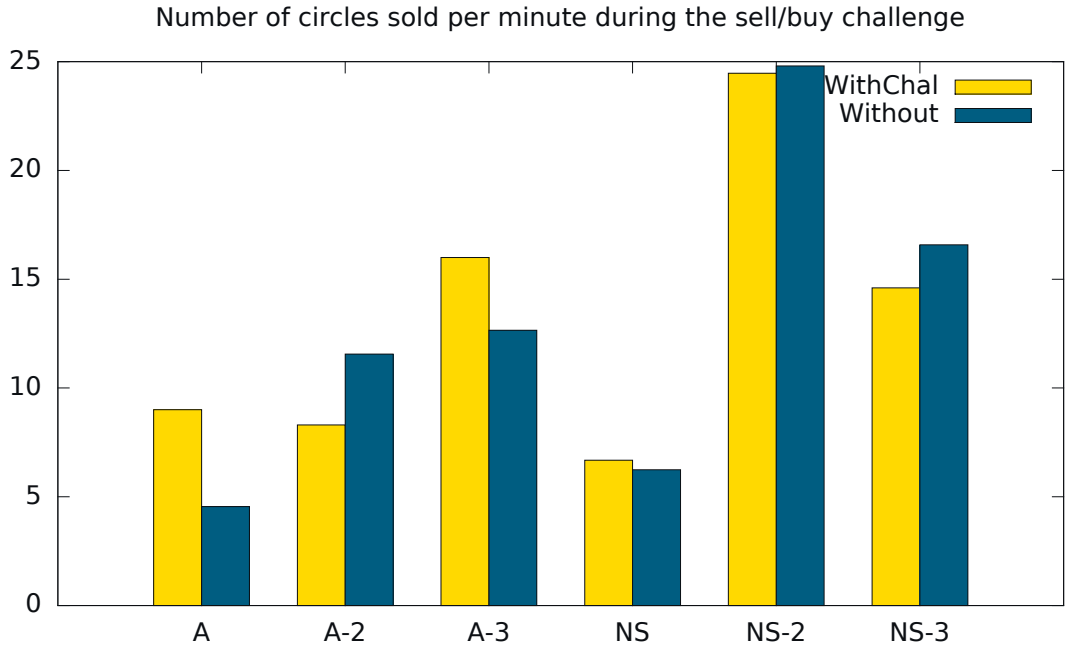

FIGURE 3.10: Number of individual circles sold on the market per minute with and without the *Sell/buy challenge* active. 'A', 'A-2' and 'A-3' represent the tests with all the features present, and 'NS', 'NS-2' and 'NS-3' represent the tests without the skills.

was the hardest challenge. All the other challenges are more general and can be completed by doing actions that are not specific to a certain subset of colors. Even if the market should be helpful in finding circles with the required subset of colors, it seems highly probable that the players felt that this type of challenge was too hard and almost never tried to complete it.

3.7.4 Testing hypothesis 4: percentage of the problem solved as a measure of the importance of different game features

One of the research goals was to measure the impact of each feature by analyzing how much of the problem can be solved by the players in each of the game sessions. Our initial hypothesis was that players who have access to all the game features should be able to solve more of the problem.

Interestingly, we observed a larger than expected variance in the participants' personal skills which made it sometimes difficult to compare one game session with another in terms of the percentage of the problem that was solved. Indeed, some players quickly understood all the rules of the game and how to maximize their score, while others struggled to make points during the whole session, even with our help.

As shown in Figure 3.11, the percentage of the problem that was solved varies from 48% to 75% in all the different experiments. In particular, the differences observed for experiments with the exact same game conditions (sometimes up to a 18% difference) demonstrates that we cannot simply use the percentage of the exact solution found as a way to measure the impact of a feature. Moreover, the top five sessions in terms of percentage solved (all sessions with more than 65%) come from the four different game conditions.

We used linear regression to test if the percentage of the problem solved is, to some extent, directly proportional to the total experience points accumulated by all the players during a session, which is a good way to measure the skills of the players during

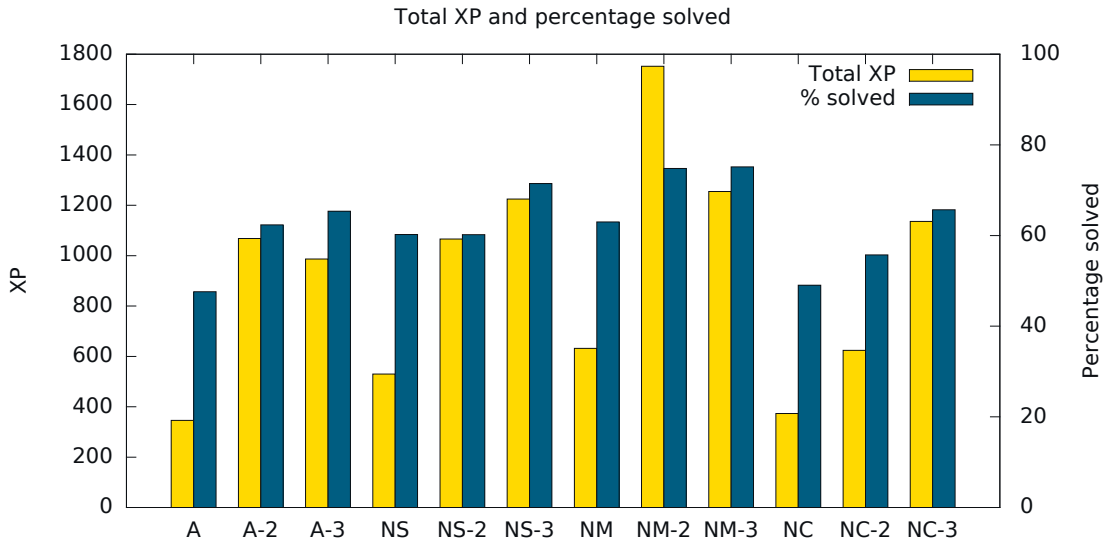

FIGURE 3.11: Total game experience and percentage of the problem solved for each of the 12 game sessions. 'XP' represents experience points. 'A', 'A-2' and 'A-3' represent the tests with all the features on; 'NS', 'NS-2' and 'NS-3' represent the tests without the skills; 'NM', 'NM-2' and 'NM-3' represent the tests without the market; 'NC', 'NC-2' and 'NC-3' represent the tests without the challenges.

each session. The linear function obtained (graph not shown) had a coefficient of correlation $r = 0.89$ and a coefficient of determination $r^2 = 0.79$, which shows a certain level of correlation. The different game conditions are obviously creating some of the observed variance. Another reason for the variance is the fact that, in the current state of the game, players can be selling sequences that correspond to a solution that was already found earlier. While it would be possible to lower the score of a solution (sequence) that already exists, it would be hard to explain to inexperienced players why one sequence is worth less than another with exactly the same length and number of colors in common. That is why we decided to not take into account the existing (*i.e.* already found) solutions in the scoring function.

3.7.5 Understanding what makes a good player

Based on the questionnaire filled by the players before playing the game, and the global leaderboard of all the players from all the sessions put together, we tried to find similarities between the top players. Table 3.3 shows the most interesting differences between the top 12 players and the rest of the players. In the questionnaires, players had to indicate their age category (between 21 and 25 for example), their own evaluation of their puzzle solving abilities and a range of hours spent playing video games every week.

The average age of the two groups of players was calculated by taking the middle point of the age categories. The average age of the top 12 players was about 2.5 years younger than the one of the other players. For the puzzle solving self evaluation, the players could choose a level between 1 and 5 (5 being the strongest). The average level of the top 12 players was 3.67, compared to 2.90 for the others. As we did with the age categories, we computed averages of time spent playing video games every week using the middle point of the categories. The top 12 players were playing roughly 2.5 times more every week than the rest of the players.

TABLE 3.3: Average statistics on the top 12 players vs the others

	Top 12 players	Others
Age	23.42	25.99
Self evaluation	3.67	2.90
Game time	10.00	4.11

3.8 Conclusion

We implemented a human computing game that uses a market, skills and challenges in order to solve a problem collaboratively. The problem that is solved by the players in our game is a graph problem that can be easily translated into a color matching game. The total number of colors used in the tests was small enough so that we were able to compute an exact solution and evaluate the performance of the players. We organized 12 game sessions of 10 players with four different game conditions (three times each).

Our tests showed without a doubt that the market is a useful tool to help players build longer solutions (sequences, in our case). In addition, it also makes the game a lot more dynamic and players mentioned that they really enjoyed this aspect of the game.

Our results also showed that skills in general are helpful to influence and guide the players into doing specific actions that are beneficial to the system and other players. We have found that skills are more efficient in their role of guiding the players if they are not directly related to the main goal of the game: the *Color Expert* skill for example did not affect the proportion of multicolored sequences built by the players.

The results on the challenges indicate that they can be useful to promote an action in the game (*Minimum number of colors in common* for example), but in order to be effective, the difficulty needs to be well-balanced. Challenges that are too easy (*Sell/buy challenge* for example) or too hard (*Specific colors in common challenge* for example) do not affect the game significantly.

Although the great variability in the participants' personal skills made it very difficult to make direct comparisons between the different game conditions in regards to the percentage of the solutions found, we showed that the percentage solved is to a certain extent proportional to the total experience gained by all players during a game session. Therefore, the percentage of the problem solved is clearly not only dependent on the features present in the game, but also on the participants' ability to be good at

the game.

Finally, it seems that younger players who play video games on a regular basis and have a strong self evaluation of their puzzle solving skills are able to understand the rules of the game and find winning strategies faster than the average participant.

Chapter 4

How do Humans Perceive 2D Clusters? Lessons From a Mobile Crowdsourcing Human-Computing Game

Alexander Butyaev [1], Chris Drogaris [1], Elena Nazarova [1], Olivier Tremblay-Savard [2], Jérôme Waldispühl [1]

[1] School of Computer Science and McGill Center for Bioinformatics, McGill University, Montréal, Québec, Canada

[2] University of Manitoba, Winnipeg, Canada

4.1 Preface

Law and von Ahn [232] define the term 'human computation' as *"the idea of using human effort to perform tasks that computers cannot yet perform, usually in an enjoyable manner."* During the first decade, the concept found great success in different applications (e.g., image labeling [233], natural language annotation [29], sound annotation [65], and spam detection [27]). Millions of reCAPTCHA queries are submitted as a mean of security step daily in order to digitize the books [48]. Games are a particularly interesting example of human computation. Powered by the human desire to be entertained, it helps to solve an underlying problem efficiently [62, 74, 71, 69].

Such human computing systems generate large annotation databases that later will be used to train ML models in solving a particular task. For example, the ESP game, social computing game for image annotation, produced the dataset containing over 100,000 images with English labels [14], which later was used to build a multimodal learning model [234]. In turns, the players of the crowdsourcing game TagATune created the largest database of music annotations [64]. However, while using these annotations as a training and testing dataset, Hamel et al. [235] demonstrated an approach that performs accurate automatic sound annotation. Thus, recent advances in machine learning allow human-driven analysis to become completely automated [236, 237].

While ESP game or TagATune addressed the classification problem, mathematically abstract problems (e.g., data ranking and data clustering) are still hard for a computer but intuitive for a human [80]. Multiple human computing systems investigate the problem of text clustering. Parent et al. [238] used AMT to cluster dictionary definitions. The Cascade [83] and Deluge [85] systems were proposed to categorize text tips and produce their taxonomy. Chang et al. [86] introduced Alloy, the hybrid approach for text clustering, which combines human judgment with ML algorithms. Multiple studies also approach image clustering problem. Gomes et al. [239] introduced the term 'crowdclustering' referring to image data clustering and then using the proposed model by Gomes, Yi et al. [240] designed semi-crowdsourced clustering method

that combines manual annotations with the low-level feature of an image. Kleiman et al. [241] show an efficient crowdsourced image similarity estimation method.

However, in its most fundamental form, abstract data clustering is still an open-ended problem. Since the term 'cluster' does not have a precise definition, no universal clustering approach suits all clustering scenarios. There are a lot of clustering algorithms that optimize different objective functions. Nevertheless, the presence (or absence) of a cluster ultimately results from an agreement between multiple individuals, and preferentially data analysis experts. Surprisingly, cluster analysis has not adopted human computation techniques yet and therefore requires further investigation.

Human (in most of the cases) has binocular vision [242]: two eyes with an overlapping field of view help an individual to perceive depth and, therefore, single three-dimensional picture. Nevertheless, human eyes contribute two distinct 2D images that later will be processed in visual cortex and merged into the 3D representation of the environment [243]. Therefore, the dimensionality of data is a main restricting factor of human's visual perception. While high dimensional data is beyond the limits of humans comprehension, their low-dimensional data analytical skills (e.g., cluster analysis) create new opportunities for human computation systems [244].

This chapter explores the human perception of simple 2D clusters. It describes the mobile human computing game `Colony B` as a framework to collect human player solutions for 2D data clustering problem. It then proposes the human-based clustering approach that utilizes accumulated game play data. Finally, the chapter discusses the method's application and reports its performance in comparison with conventional clustering algorithms.

The remainder of text found in this chapter is taken from:

> Alexander Butyaev, Chris Drogaris, Elena Nazarova, Olivier Tremblay-Savard, and Jérôme Waldispühl. How do Humans Perceive 2D Clusters? Lessons From

a Mobile Crowdsourcing Human-Computing Game. *In submission at Computer-Supported Cooperative Work and Social Computing conference*.

A.B. contributed to the design of the mobile application, developed server side of the `Colony B` *game and data management system, implemented the computational analysis, and prepared a draft for the manuscript. C.D. designed the mobile application for the* `Colony B` *game. E.N. designed the reward system (badges). O.T.S. outlined an original idea of the game and helped draft the manuscript. J.W. contributed to the design of the mobile application and coordinated the computational analysis.*

4.2 Abstract

Abstract data clustering is a challenging computational task that involves identifying groups within a distribution of data. Many algorithms and metrics have been developed to solve this problem, but the existence of clusters more naturally follows a collective agreement between experts rather than a precise mathematical definition of distance or density. In this paper, we design a mobile crowdsourcing human-computing game as a tool to collect 2D clustering data solutions from players. We analyze the data collected through a large-scale year-long gaming session involving thousands of volunteers that clustered real and simulated datasets. First, we present strategies enabling us to circumvent biases observed in cluster annotations collected through mobile devices. Then, we use a variety of cluster validation techniques to evaluate the performance of our crowdsourcing system. Our results suggest that although the quality of clusters generated by humans and computers is similar, clusters obtained from aggregated human solutions are significantly larger than the ones generated by computer programs.

Keywords: Data Clustering | Game | Crowdsourcing | Data Analysis

4.3 Introduction

Data clustering is a central task in many computer analysis techniques, that is routinely used in data mining and unsupervised machine learning. However, the definition of a cluster remains ambiguous and subject to interpretation. There is no single universally accepted mathematical definition that unambiguously decides the existence of a cluster. Indeed, clusters may have very different shapes and densities within and across datasets. Among the various measures previously introduced to estimate the quality of a cluster annotation [245], Silhouette [118], Dunn [119], SDbw [120], and Modularity [246] emerge as the most popular. This situation resulted in the development of a broad variety of algorithms and metrics [247] but as mentioned above

without consensus. The presence (or absence) of a cluster ultimately results from an agreement between multiple individuals, and preferentially data analysis experts.

Unfortunately, a manual expert annotation is not scalable to large datasets. Moreover, it is unclear how human clustering annotation would compare to those obtained from state-of-the-art algorithms.

Crowdsourcing is now a popular and well-established technique to perform a large number of repetitive tasks, with the underlying assumption that the aggregation of a sufficiently large number of answers from non-experts could approximate expert behaviors [77, 74]. Such an approach opens the doors to large-scale experiments allowing us to calibrate the human perception of clusters against the performance and behavior of most popular clustering algorithms.

In this paper, we developed a mobile application to crowdsource cluster annotations performed by humans. We then compare these results to clusters calculated by state-of-the-art algorithms using various popular metrics. Importantly, we address a generic version of the clustering problem, where the data is represented as clouds of point distributed on a 2D plane. This representation enables us to reduce contextual/cultural bias and focus on the human perception of visual patterns.

Importantly, clustering algorithms generally optimize cost functions designed to estimate the quality of the clusters [245]. However, cost functions might suffer of overfitting: the total error is minimized with large number of small clusters [91]. Consequently, the results of the clustering algorithm often is not informative for the cluster analysis. Although, researchers proposed some methods to address this issue (e.g., elbow method [248], average silhouette method [118], and gap statistics [249]), the size of clusters generated by automated clustering algorithms can still be hard to interpret. In this paper we also propose the hypothesis that the clusters aggregated from human annotations are significantly bigger than the ones generated by automated clustering algorithms and try to address it by conducting two experiments asking participants to cluster simulated and real world datasets.

The paper is organized as follows. In section 4.4 we introduce the online human

computing game that we designed primarily for clustering data collection of the human worker. Next, section 4.5 describes our methods, explains the design of multiple scores and proposes the tool for measuring clusterability of a dataset. In section 4.6 we describe our experiment on simulated and real-world datasets. In particular, we find that humans identify significantly larger clusters than automated methods at the same level of accuracy. Moreover, although humans compare favorably to classical algorithms on artificial datasets with clearly distinct clusters, their performance appears to be more impacted by noisy datasets. Finally, section 4.7 concludes this manuscript with a discussion of our results.

4.4 Mobile Human Computing Game

To address a problem of real-world data clustering, we implement online human-computing game `Colony B`. It is a gamified environment that utilizes crowd intelligence and intuition to tackle the problem of the relative positioning of data points on the mobile screen. Since a manual expert annotation is not scalable to a large dataset, in our approach regular player acts as an expert that manually annotates a partition of the dataset. Then all accumulated human solutions are merged to obtain a single annotation for the entire dataset.

Real-world dataset rarely has only two features (dimensions) so that it can be sampled and showed to the user. We extend our game to deal with multidimensional data. We use various techniques such as dimensionality reduction and random projection techniques, to limit the amount of data to a series of 2D datasets, which then will be played separately by the crowd.

The game is designed around an idea of simple actions from a player - encircling a group of similar points (in context of the shown dataset at the screen) both on a single screen or series of screens (the reference to multidimensional dataset). Thus, the players are responsible for choosing their clustering criterion, which, in turns, can be considered as a simplified data review process by an expert.

Therefore, every game is a consecutive set of 2D screens. Each screen is a set of 2D points, which are to be grouped/clustered together. An example of such a screen is shown in Figure 4.1A. For each screen, a player is allowed to create a single group of dots, which in the context of the given screen, is the best possible cluster (based on the player's judgment).

Since the process of solving puzzles might be monotonous, it requires various gamified elements as well as game dynamics progression called the flow [250] to keep players attention. While the latter is still an open-ended problem (we discuss it more in the Conclusion section), we implement multiple engagement boosting features in the game.

We introduce time-bonus (time slider on top of the screen Figure 4.1A) to speed up gameplay and improve its dynamics. Our end game screen contains an animated indicator of the progress towards the next thematic badge, total and accumulated per game scores, the performance of the player with respect to the crowd (Figure 4.1B).

`Colony B` is also an educational game. We use thematic badges to grow the interest of our players to microbiome studies as well as to provide score related rewards (Figure 4.1C, 4.1E). Our badges are the most known bacteria (such as *Escherichia coli*, *Bifidobacterium longum*, etc.) that become available when a player reaches a specific milestone (score).

An essential part of the game is a leaderboard. We have noticed that new players most of the time get discouraged when they compare a game score to (for example) the top ten players. Thus, we created a system of leagues - partitions of the leaderboard based on multiple score ranges (Figure 4.1E). A player starts from the reasonable distance from the leader of the league and when steps out from the initial league range, goes to the next league.

One of the biggest challenges for any game-with-purpose is to provide enough information for people to produce quality results [251]. We designed an interactive tutorial (Figure 4.1F) that step by step explains each aspect of the game with immediate feedback on every action of the player.

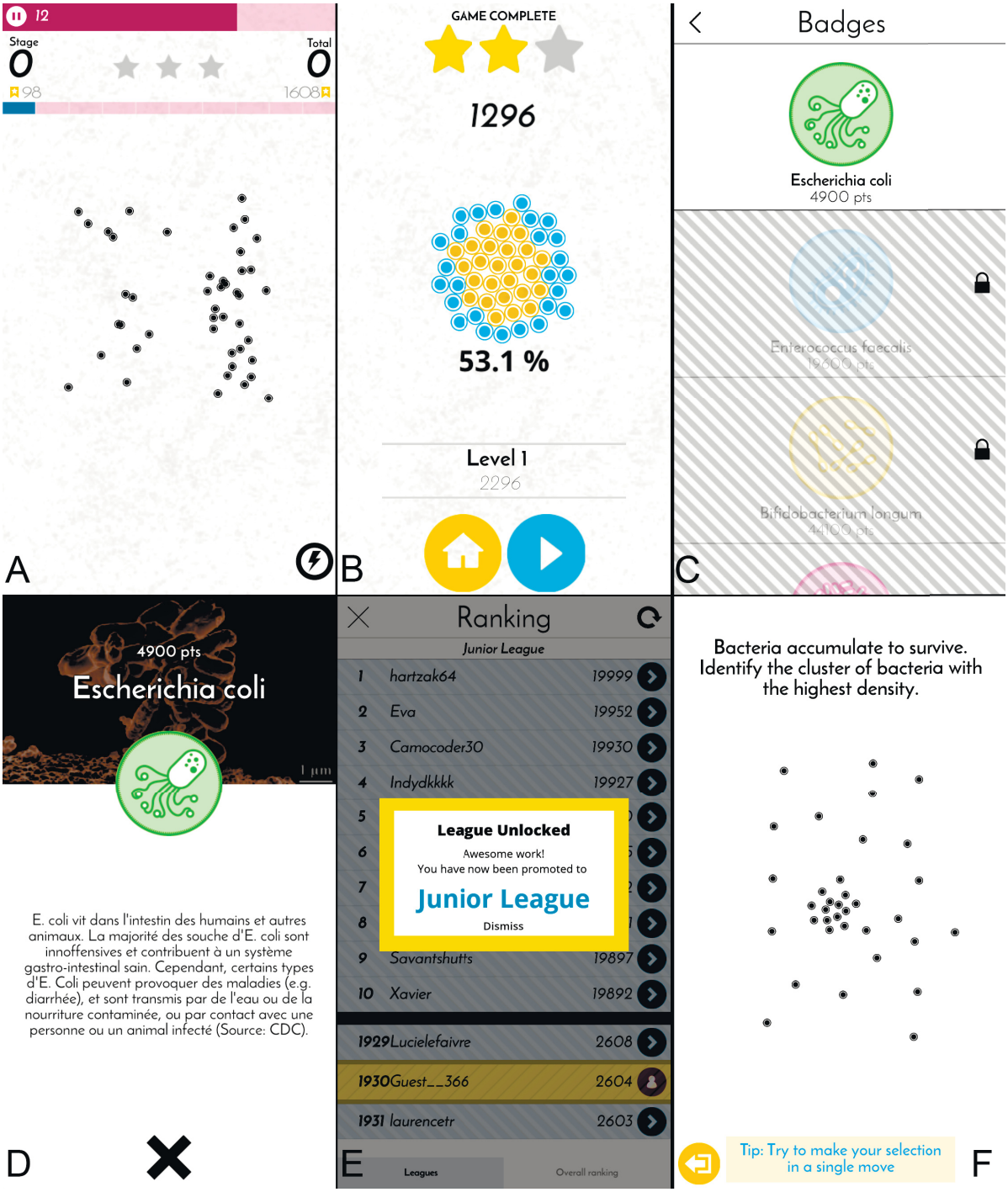

FIGURE 4.1: Illustration of multiple scenes of `Colony B`game: A) clustering panel with stage / total scores and puzzle progress; B) End game screen with a status in new badge discovery; C) List of available thematic badges; D) Educational information about badge; E) Leader board with multiple leagues; F)Interactive tutorial.

Although Colony B is an online game, it allows a player to play while offline and access multiple puzzles, loaded during the last online gaming session.

Since our goal is to involve people in the gamified process of clustering regardless of their knowledge about the underlying problem, we made our game publicly available on all major mobile platforms. Therefore, we do accept all players without a selection process and give them a choice to register with email or start playing right away as a guest user. In case of email registration, we do not store any personal information. We explicitly guarantee the confidentiality of collected information and state that it can only be used to enable a user to reset the password or send feedback about player's contribution to science. We describe the usage of the player's solutions in the mobile application (information tab) as well as on the official Colony B website.

The game consists of multiple stages (screens). At every stage (screen) user is allowed to encircle single group of dots that is the most representative for clustering for particular player in current situation. The main flow of the game (first two stages with player's actions and system reaction) is illustrated in the Figure 4.2.

4.4.1 Game Play Scoring

We design a scoring system as a tool for both providing a reward for a player's action and introducing a competition factor to the game as a boost for crowd engagement [251]. Since such system must not bias a player towards particular solution we compile our scoring function as a superposition of three different clustering validation indices: (i) Silhouette index [118], (ii) SDbw index [120], and (iii) Dunn index [119]. All three indices find cohesion and separation in its own way and then calculate a final index value. That is, Silhouette index computes inter / intracluster distances, while Dunn index operates with diameters of each cluster as well as minimal inter-cluster distance. In contrast, SDbw index computes scattering (sum of standard deviations of each cluster over the standard deviation of the dataset) and density at the midpoint of the cluster centers with respect to the cluster center densities. The maximum over

FIGURE 4.2: The example of the game flow with first two stages and the process of puzzle solving. A) The player is presented with initial stage data. B) The player makes selection of the most representative group of dots on the mobile screen. C) Feedback from the game. D) The player is presented with the next stage data. It also shows in blue the points selected by the player on the previous stage. The Player also encircles suitable group of dots on the screen. E) Feedback from the game.

these three (normalized) values for particular screen will be denoted as its *quality* score $Qual(\mathcal{S}_i)$, where $\mathcal{S}_i$ is a state of the game at ith screen.

Since a series of screens represents a dataset in different dimensions, we introduce a *conservation* score $Cons(\mathcal{S}_i, \mathcal{S}_{i+1})$, the number of points conserved between two consecutive screens. To emphasize this feature in the game and show potential conservation between screens we highlight points that were selected by a player in the previous screen.

Thus, the preliminary score formula can take the following form:

$$QCScore(\mathcal{S}_i) = \alpha * Qual(\mathcal{S}_i) + (1 - \alpha) * Cons(\mathcal{S}_{i-1}, \mathcal{S}_i)$$

We use $\alpha = 0.8$ to balance *quality* and *conservation* scores.

As a final step, we introduce a density correction coefficient $\mathcal{DCC}$ to shift a player's attention towards significantly dense areas. It also aims at reducing the spontaneous behavior of a player. The final scoring function for a player's stage evaluation is:

$$Score(\mathcal{S}_i) = QCScore(\mathcal{S}_i) * \frac{1 + \mathcal{DCC}}{2}$$

To define $\mathcal{DCC}$ we utilize Kernel Density Estimation technique [252]. It is a non-parametric estimation of probability density function (PDF) of a random variable to the given screen dataset. Using PDF, we estimate density in every data point - value in the range $[0, 1]$ that can be used as a weight of each point.

Consider set of points $\mathcal{P}$ for screen $\mathcal{S}_i$. Denote $\mathcal{P}_{selected}$ as a set of selected points and, oppositely, $\mathcal{P}_{notselected} = \mathcal{P} - \mathcal{P}_{selected}$. Let d_k to be the densities of points in $\mathcal{P}$, and $\vec{D}$ is a vector of densities, $|\vec{D}| = |\mathcal{P}|$. Following calculations assumes densities to be in range $[0, 1]$.

Define *balance density* $nm = min(median(\vec{D}), mean(\vec{D}))$, and transform densities as follows $d_k := d_k - nm$. It splits a range of densities in positive and negative intervals, which will be used as a base for award and penalization weights respectively. Finally,

we can write Density Correction Coefficient as follows: $\mathcal{DCC} = \frac{Nom}{Denom}$, where

$$Nom = \sum_{p_k \in \mathcal{P}_{selected}} d_k - : PC * \sum_{p_k \in \mathcal{P}_{notselected}} d_k$$

$$Denom = \sum_{d_k \in \vec{D}, d_k > 0} d_k - : PC * \sum_{d_k \in \vec{D}, d_k < 0} d_k$$

where PC is a parameter controlling a strength of penalization in the functions. For example, if the player's selection contains two dense areas, then, according to $\mathcal{DCC}$, the densities of the most dense group of points will be balanced by the densities of the least dense group of points (by the factor of PC). While we do not intend to penalize significantly player's score and want selection's imperfections to contribute to a stage score, we apply correction factor PC with small values (e.g., 0.1).

4.4.2 Human interaction with mobile device

The crowd interacts with a `Colony B` game using mobile application (available both for iOS and Android devices). It speeds up data collection, and, therefore, improves the quality of the solution.

However, it introduces a significant challenge to data representation. For example, people holding a smartphone in right/left hand tend to keep paying more prominent attention to the smaller area closer to a thumb [253]. To eliminate this problem we randomly mirror screen datasets along X axis only, Y axis only, or combined X and Y axes.

4.5 Methods

In this section, we describe our data analysis pipeline applied to the data accumulated with the `Colony B` game. We consider data as a set of individual solutions. Instead of analyzing distance, density, or other parameters in Euclidean space, we use the space of solutions.

Such space considers every solution as a data point where position vector is a binary vector of the same size as a dataset. Each element of the vector defines a state of the particular point: 1 if the point is selected, 0 otherwise. In the following discussion we use the following formalism: $\vec{V}_i$ denotes an individual binary vector from set of all solutions $\mathcal{V}$; $\vec{V}_{i,k}$ is a k^{th} binary vector of i^{th} cluster.

To transform the multiclass output of regular clustering algorithms into the solution space, we use *One-vs.-Rest* [254] strategy. It results in a set of binary vectors, which must be treated as stand-alone solutions.

In solution space, we group solution vectors and evaluate the strength of the obtained consensus.

Finally, we compare the performance of our approach with multiple clustering algorithms based on such metrics as Silhouette, Dunn, SDbw cluster validation indices. Besides, we use modularity metrics [246] (measure commonly used to evaluate partition of the network as a tool for community search). It can be adapted to our situation by sparsification - trimming the furthest edges (we choose 0.5 of the dataset diameter), a simple transformation of resulted sparse distance matrix to the similarity matrix, and then computing modularity.

4.5.1 Base Pipeline: Search of Vector Groups

Unlike most of the clustering algorithms, the key to our approach is a search of groups of vectors in a set of vectors $\mathcal{V}$. That is clustering in the space of solutions. Here we propose a generic pipeline that can be applied to the general set of binary vectors:

(i) accumulate all binary vectors $\vec{V}_i$ of $\mathcal{V}$ observed from processing particular dataset;

(ii) find the pairwise distance between binary vectors $\vec{V}_i \in \mathcal{V}$ (we use Manhattan distance due to its simplicity);

(iii) given distance matrix, filter out outliers using Local Outlier Factor (LOF) technique [255]. In solution space, the outlier is a solution that is distant from the rest and cannot be grouped with another solution;

(iv) given cleaned distance matrix (filtered outliers), use the community search approach to identify groups of binary vectors. This technique relies on the similarity matrix, which can be obtained by using Radial Basis Function (RBF) kernel [256] or simply subtracting elements of distance matrix from the all-ones matrix of an appropriate size. For community search algorithm we choose Louvain algorithm [114] due to performance on relatively large networks (must be scalable with the number of solutions).

The result of the pipeline described above is a set of vector groups that can be explored independently. Next, we propose an alternative measure of clusterability of a dataset as an additional step to the base pipeline.

4.5.2 Measure of Clusterability

Usually, clusterability is defined by measuring separation and cohesion of the clusters in Euclidean space [257]. In this section, we design an alternative version of this measure, which is also suitable for indicating the presence of clusters in a given dataset.

As discussed above, we obtained distance matrix DM, clusters $\mathcal{C} = \{C_1..C_n\}$, where C_i is a set of binary vectors $\{\vec{V}_{i,1}, .., \vec{V}_{i,k}\}$. Every such vector $\vec{V}_{i,j}$ has size P - number of data points in dataset. Let us define DM_i as an intra cluster C_i distance matrix and DM_i^U - upper triangle of distance matrix DM_i . We compute an average number of points that been selected per cluster $Sel(C_i) = \frac{\sum_{k=1}^{|C_i|} sum\{\vec{V}_{i,k}\}}{|C_i|}$, where $sum\{\vec{V}_{i,k}\}$ is a sum of all the elements of vector $\vec{V}_{i,k}$. Here, $Sel(C_i)$ balances designed score to the average selection and explicitly control an importance of clusters with small / large ratio of selected points. Thus, for particular cluster C_i individual score will take form:

$$iScore = mean(DM_i^U) * \left(1+ : mult * \left(\frac{2 * Sel(C_i) - P}{P* : drop\_point\_ratio}\right)^{:degree}\right)$$

Parameters *mult*, *drop_point_ratio*, and *degree* control the weight increase for the under- / over-represented clusters. Since we are interested in clusters with size in range [0.15, 0.85] of the dataset size - we choose the following values for parameter: 4, 0.85, and 12 respectively.

It is important to mention that the *iScore* function does not depend on the number of solutions contributed to the cluster. To take into account the number of solutions, we factor the *iScore* with a coefficient that weights down clusters with few solutions:

$$iqScore = iScore * \left(1 + \left(1 - : qMult * \frac{|C_i|}{|\mathcal{V}|}\right)^{:qDegree}\right)$$

Parameters *qMult* and *qDegree* control the degrading speed of the coefficient. We choose 1 and 12 respectively to lower importance of clusters containing less than 20% of vectors from $\mathcal{V}$.

Since the primary purpose of the measure of clusterability is to indicate the presence of cluster for a given dataset, we argue that knowledge about the maximum value of the *iScore / iqScore* overall existent clusters is enough to make a decision about a particular dataset.

4.6 Experiments

In this section, we describe experiments arranged to illustrate the behavior of proposed approach on two (artificially created and real world) datasets, compare its performance with different clustering algorithms in terms of multiple cluster validation indices as well as modularity and ground truth labels (for real-world dataset).

It is important to mention that since the game is open to all mobile device users, we do not control the number of players participating in the experiments as well as their knowledge of data clustering topic. Since puzzles were assigned randomly, we explicitly report the number of the players contributed to a particular experiment.

4.6.1 Evaluation Dataset

We test our approach on the artificially created dataset, which at the same time was used to evaluate the performance of `Colony B` players. The dataset was compiled manually suggesting 10 different datasets with 40 points that provide various clustering situations: different number of clusters of varying shape, density, size and location on the mobile screen as well as simple noise dataset without clear clusters. The dataset was played by 3188 participants with 5503 solutions submitted.

We compared the performance of our algorithm (denoted as "human") with multiple clustering algorithms (Agglomerative [98], Gaussian Mixtures models (GMM) [105], KMeans [91], DBSCAN [101], Louvain [114] and also explicitly designed accumulative algorithm that collects all possible solutions for different clustering algorithms and uses them as a stand-alone solution). We used the *elbow* method to identify the number of clusters as well as parameters of the algorithms (e.g., *eps* and *min_samples* for DBSCAN). For comparison, we use four metrics: modularity and Silhouette, Dunn, SDbw cluster validation indices. This experiment showed that according to the four metrics used, the human-based approach is on a par with or outperformed other clustering algorithms. The detailed description of the experiment can be found in Supplementary Material: Evaluation Dataset section.

4.6.2 Voice Recognition Dataset

We also report performance on the real world dataset, Voice Recognition dataset [258] (later, is referred as VRD) which initially contains 21 features, but was processed with dimensionality reduction technique (Principal Component Analysis [259], variance retained ration is 90%) to six-dimensional dataset to fit requirements of our `Colony B`. This dataset was played by 848 participants with over 350 players completing more than ten games. In total, 15515 solutions were submitted with an average 100 games per puzzle.

We propose a hypothesis that players of `Colony B` and therefore human-based approach tends to select bigger clusters than other clustering algorithms. We also discuss other hypotheses related to algorithms performance comparison on various size clusters and clusterability of the data in Supplementary Materials.

We compare cluster sizes for human-based approach and clustering algorithms using four metrics discussed above. To proceed we filter results for stages where either clustering algorithm finds multiple clusters with more than two data points, or human based approach evaluates it as clusterable. The result is shown in Figure 4.3. We need to clarify that cluster size depends on the metric used as for every stage we select representative cluster - cluster with the best score in particular metrics.

Since the distributions of the cluster sizes (all of them) are not following the normal distribution (the Shapiro-Wilk test rejected the null hypothesis with $p = 3.61621e-6$ as for human-based approach using SDbw metrics), we use Wilcoxon signed-rank test to compare medians of two paired groups. For modularity, the significance test rejected the null hypothesis comparing cluster sizes of human-based approach with other clustering methods ($p = 5.9058e-10$). A significant effect ($p < 0.05$) is found with 2% of the samples for GMM, KMeans, Louvain, and multialgo algorithms (40); and 5.5% of the samples for Agglomerative algorithm (110).

For Silhouette, Dunn and SDbw indices tests rejected the null hypothesis ($p < 1.0e-20$). A significant effect ($p < 0.05$) is observed with 2% of all the samples (40).

This result supports our hypothesis 1 that human solutions in general and the results of human-based approach tends to make larger clusters.

We do not emphasize the results of testing the hypothesis 1 on Evaluation Dataset as the dataset lacks a structure uncertainty and primarily targets the performance comparison between human based algorithm and popular clustering algorithms using screens with clear clustering situation. As expected, the experiment does not show any significant difference between these algorithms.

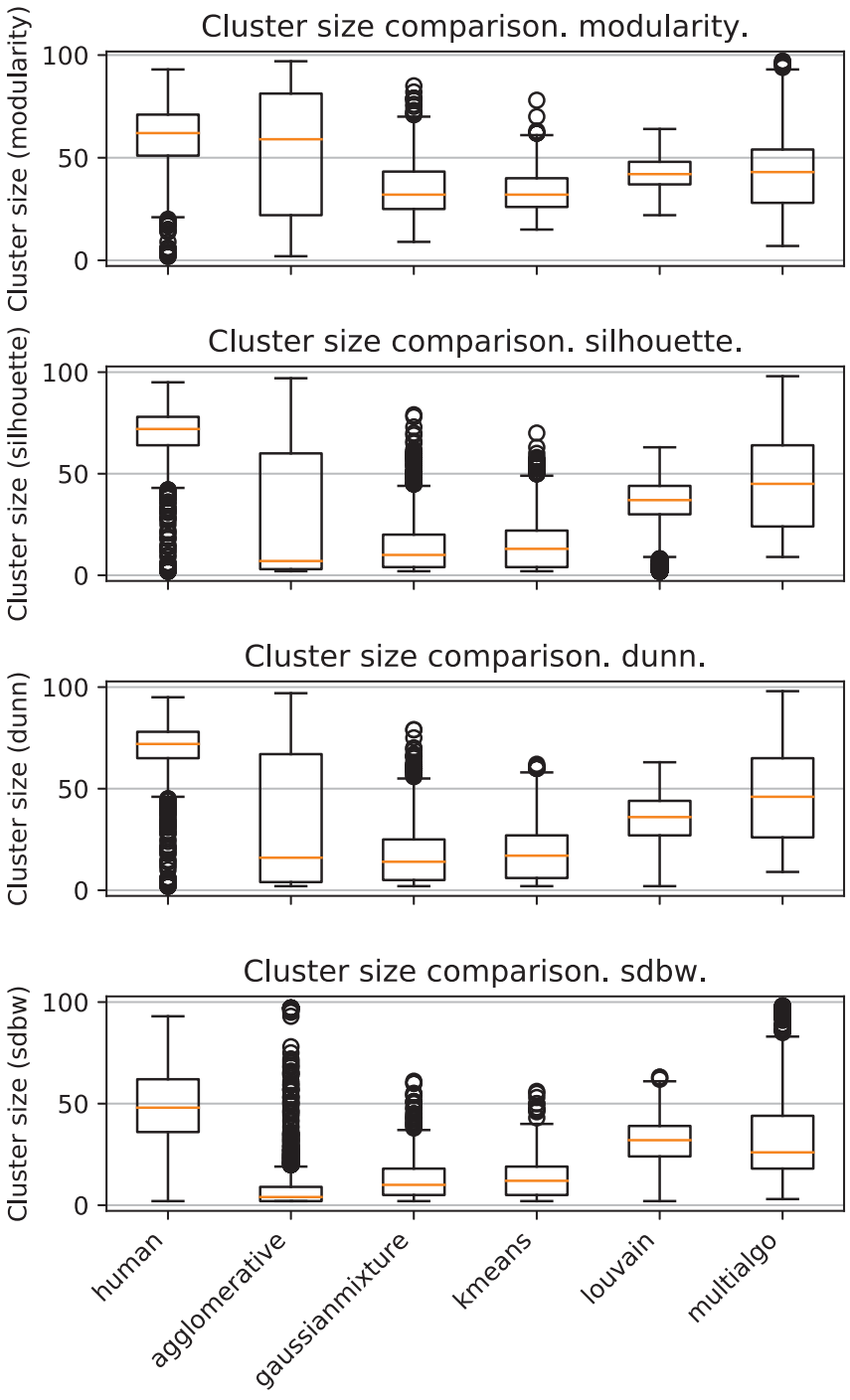

FIGURE 4.3: Comparison of cluster sizes for VRD for human based approach and clustering algorithms based on the 4 metrics: Modularity, Silhouette, Dunn, SDbw.

4.6.3 Mobile Screen Bias

As discussed in section 4.4.2, the mobile device user interacts with smartphones in several ways that can affect the result of our approach. Therefore, it is important to test if each player groups data using the entire clustering panel.

To arrange this experiment, we analyzed individual solutions of every player that completed at least ten games for VRD. We mirrored data as to oppose to actions described in section 4.4.2 and first studied left and right partition of the dataset ($[0, 0.45]$, $[0.55, 1]$ partitions of the screen along X axis). Let l be the number of hits in the left part of the screen and r - in the right part. Then we are interested in the relative difference of two sides $|l - r| / (l + r)$. We found that the difference is at most 0.205 with average 0.047 ± 0.037. It indicates that there is practically no bias in clustering patterns along the X axis for players caused by the usage of a mobile device.

We also analyzed the pattern along the Y axis in the same way. We found that relative difference is in average 0.77 ± 0.26. This implies that there is a strong bias of people using the lower part of the screen to perform clustering. This phenomenon can be explained by the size of the screen of modern mobile devices as well as bad reachability of top of the screen.

We report the results of the experiment as centre of mass of human clusters on a mobile screen (Figure 4.4). The experiment shows a strong skew in the human clustering preferences towards the bottom of the screen.

This experiment suggests that user interface must be adjusted in such a way that action panel is located on the bottom of the screen, and all the game stats are moved to the top. It also justifies the necessity of mirroring data along the Y axis.

4.7 Conclusion

We implemented an online human computing Game `Colony B` as a complex framework to collect human player input in order to solve the problem of data clustering.

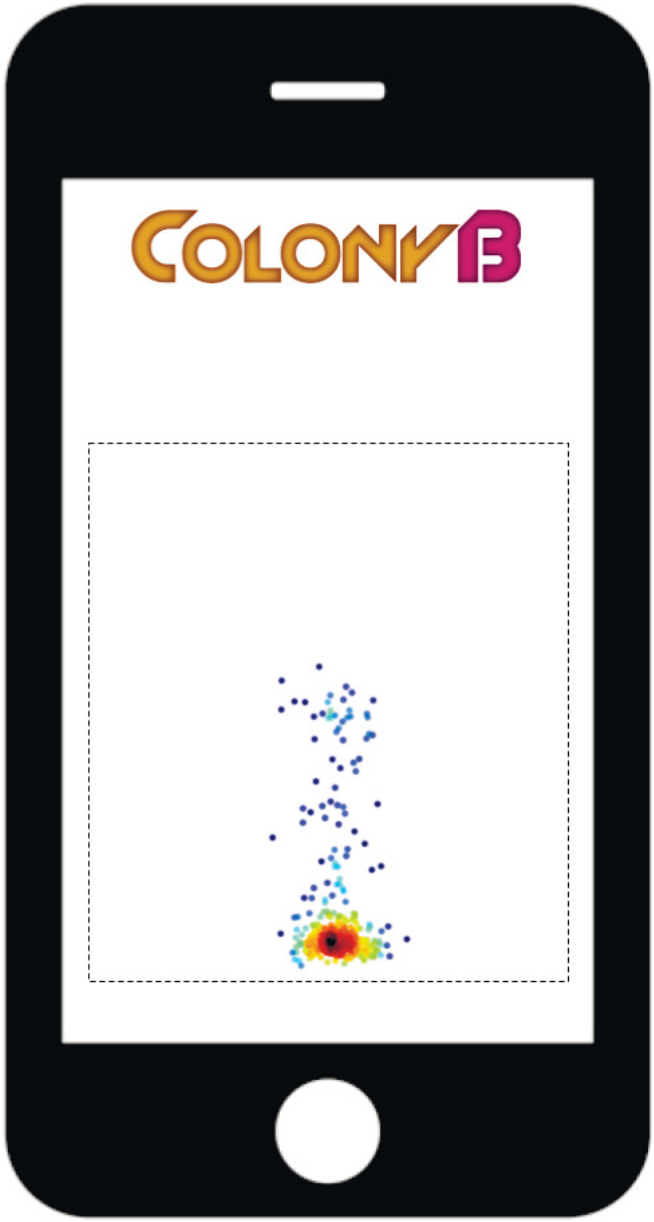

FIGURE 4.4: Centre of mass of human clusters on a mobile screen. Data point represents a measure of the clustering preference for each user. Color illustrates data point density.

We conducted a year-long experiment with over 4000 participants (in total) using two different datasets: an artificially created evaluation set featuring various clustering scenarios, and a real-world voice recognition dataset. Participants to this experiment (i.e., gamers) performed single clustering annotation using a mobile application. The players were rewarded using a simple point rewarding scheme.

We analyzed the collected data and proposed a pipeline that aggregates human solutions. We compared the performance of our human-based approach with major clustering algorithms such as Agglomerative, Gaussian Mixtures models, KMeans, Louvain algorithm. We also compared performance with multialgo - a customized algorithm that aims to mimic human behaviors using results from multiple runs of various clustering algorithms instead of human solutions. It is worth noting that we initially included the DBSCAN algorithm with automatic estimation of parameters using Elbow method in our benchmark. However, this method performed poorly on both datasets (either it identifies single cluster or treats the majority of data points as noise), and thus we decided to exclude DBSCAN from our benchmark to as it was not representative of the quality of the algorithm.

We discussed the performance of humans and algorithms using four popular metrics (Modularity, Silhouette, Dunn, and SDbw) on a simulated (i.e., reference) and real-world datasets.

First, our result on the reference dataset demonstrates that the human annotation outperforms (or is on a par with) other clustering algorithms. Importantly, this result is obtained when users agree on the cluster.

By contrast, on the more noisy Voice Recognition dataset we observed that, according to the four metrics used, the performance of humans is slightly more deteriorated than those of computer algorithms. However, we showed that human-based approach results in significantly larger clusters comparing to automated approaches. Besides, to verify this result, we tried to detect some bias in the way how people are selecting clusters. We used information about relative screen location of each player's cluster in combination with mirroring data statistics on the player screen and found that along

the X axis there are no significant skews toward some parts of the screen. However, a similar experiment with the Y axis showed a bias of players using mainly the bottom part of the screen. This is expected behavior due to the size of the mobile device and reachability of the top part of the screen and justifies our strategy of mirroring data along the Y axis.

We noticed that a lot of accumulated solutions take up to 100% of the screen. Although it might be caused by the noise of the data, it might be a drawback of the designed scoring function and will be addressed in future work.

We also mentioned in the description of the game the need of the flow - a natural game progression that increases challenge level according to the skill level of the player. Whereas we have added multiple game elements that advance with the player's level (score, leaderboard ranking, badges), this is still an open-ended problem for the `Colony B` game (as well as for many games-with-purpose) since the underlying problem has no distinct and clearly defined solution. In particular, ranking puzzles by their complexity is an equally hard problem and will be studied further in future work.

4.8 Declarations

Ethics approval and consent to participate

Ethical approval was not needed for this study.

Consent for publication

Not applicable.

Availability of data and materials

The `Colony B` game is available on Google Play (https://play.google.com/store/apps/details?id=pontax.cs.mcgill.ca) and Apple Store (https://itunes.apple.com/us/app/colony-b/id988892383?mt=8).

The voice recognition dataset is available at the Kaggle website (https://www.kaggle.com/rohankale/voice-recognition).

Funding

This work was supported by the Genome Canada, CIHR, BCB competition.

Acknowledgements

We would like to thank members of our laboratories for meaningful advice and recommendations regarding the `Colony B` game design and data analysis methodology, all `Colony B` players for their invaluable contributions to science.

Chapter 5

Crowdsourcing Multidimensional Data Clustering

Alexander Butyaev [1], Chris Drogaris [1], Jérôme Waldispühl [1]

[1] School of Computer Science and McGill Center for Bioinformatics, McGill University, Montréal, Québec, Canada

5.1 Preface

Clustering is a complex computational task of dividing datasets into groups of similar objects while maximizing dissimilarity of objects between the groups. Objects are usually represented as points in a multidimensional space, where each dimension represents a distinct attribute describing the object. For instance, the attribute might represent the spatial coordinate, the measurement of an experiment, the property of the object represented by a point, etc.

Unlike the classification problem where objects in datasets are labeled in advance so the accuracy of the algorithm can be evaluated from the labels, the data clustering problem is ambiguous since the term 'cluster' does not have a precise definition. Steinbach et al. [122] lists five commonly used definitions, including center-based, density-based, similarity-based, etc. These definitions assume a specific similarity/dissimilarity function to be optimized. Many conventional clustering algorithms use a distance or similarity measure to group together objects that are, in general, close to each other. However, as the dimensionality of a dataset increases, the distance or similarity function becomes less useful [260]. This loss in data usefulness is commonly known as the curse of dimensionality (CoD). CoD is defined as the increase in dataset sparsity caused by a dramatic increase in dimensionality [122].

Feature selection [127] and feature extraction [124] techniques are commonly used strategies to address the problem as a data preprocessing step. While the former is a computational technique to filter out non-informative features (or dimensions), the latter projects a dataset to a lower dimensional space and, therefore, generates a completely new set of features. While both approaches might seem appealing, resulting features can be non-informative or lack interpretability.

Besides, multiple methods have been proposed to address clustering in high dimensional space [149, 150, 147, 151, 152], including subspace and projected clustering algorithms. The former employs bottom-up strategy first considering the primitive subspaces and then exploring higher dimensionality subspaces. In contrast, the latter

considers an entire set of dimensions and then searches for optimal partitions of the feature set to represent the clusters.

Nevertheless, regardless of the dataset dimensionality, the presence of a cluster results from an agreement between multiple individuals. While it is preferred to have an expert manually analyzing data, such a strategy is not scalable and often expensive. In contrast, crowdsourcing is a well-established technique to engage non-expert workers to solve (micro-)tasks that are still hard for a computer program. In particular, the clustering problem has no precise definition and requires human assistance to assess the problem [80].

Multiple studies have approached complex textual data clustering problem using human computing techniques [83, 85, 86], including Cascade, DELUGE, and Alloy. Crowdsourced image clustering was also explored in [240, 261]. Yet, an application of human computation to abstract data clustering problem is poorly described in the literature.

In Chapter 4, we described `Colony B`, a mobile human computing game that approaches the simple 2D clustering problem. Since the first release of the game in 2016, over 130,000 puzzles were completed by over 4,400 unique players. Over 200 players submitted more than 100 solutions each with a 10,000 solutions submitted by top three players.

This chapter describes the `Colony B` game as a tool for data clustering in high dimensional space. It presents two crowdsourced clustering methods that accurately compile collected human player solutions into multidimensional clustering results. We also analyze microbiome data from American Gut Project [262] and give an intuition on the game modifications that allow the system to use the human computing technique more efficiently.

The remainder of text found in this chapter is taken from:

Alexander Butyaev, Chris Drogaris, and Jérôme Waldispühl. Colony B: Multidimensional Data Clustering Using Human Computing Techniques (2019) *In preparation for submission to the Human Computation and Crowdsourcing (HCOMP) conference*.

A.B. contributed to the design of the mobile application, developed server side of the `Colony B` *game and data management system, implemented the computational analysis, and prepared a draft for the manuscript. C.D. designed the mobile application for the* `Colony B` *game. J.W. contributed to the design of the mobile application and coordinated the computational analysis.*

5.2 Abstract

Clustering is a central task in many data analysis applications. However, a 'cluster' does not follow any precise mathematical formulation but results from a consensus from several experts. The problem is even amplified when analyzing high-dimensional data sets where classical distances become uninformative and an agreement even harder to reach. In this paper, we design a mobile human-computing game as a tool to collect human input for the multidimensional data clustering problem. We propose two crowdsourcing multidimensional clustering algorithms that partially or entirely rely on aggregated human answers. We report the results of two experiments conducted on a synthetic and real-world data set. Using accumulated solutions, we benchmark the proposed algorithms against most popular automated clustering algorithms and demonstrate that our methods perform on par or better than fully-automated clustering algorithms. We also analyze the microbiome dataset obtained from the American Gut Project and draw some intuition on our game's modification needed for more effective use of crowdsourcing technique.

Keywords: Data Clustering | Crowdsourcing | Human computing | Game-with-a-purpose | Algorithms | Multidimensional data

5.3 Introduction

Clustering is a complex computational task of dividing datasets into groups of similar objects while maximizing dissimilarity of objects between the groups. The problem definition is somewhat ambiguous since similarity/dissimilarity metrics significantly differ. Cluster analysis encounters many algorithms that emphasize the cluster's shape [97, 91, 87, 98], density [101, 102], or the underlying distribution model [105]. It also provides a number of cluster validation indices and metrics [247]. However, none of them can guaranty to produce satisfying results and often require data expert assistance.

Furthermore, in the context of multidimensional data, traditional similarity metrics, which are used in conventional clustering approaches, are usually not meaningful (i.e., the distance function becomes less efficient due to the curse of dimensionality [123]). To address this problem, customized strategies such as subspace and projected clustering were proposed [150, 147, 153, 149]. Nevertheless, due to the complexity of the task, expert supervision remains required.

Crowdsourcing is a well-known technique to employ human workers to perform repetitive work in data annotation and interpretation which is still hard for a computer [80]. Although the majority of participants are non-expert, sufficient amount of aggregated answers were shown to produce the expert-level results in different fields [263, 264]. Multiple systems address the text clustering problem using crowdsourcing techniques [83, 85, 86]. However, the abstract data clustering is yet to be studied.

In this paper, we present `Colony B`, a human computing game [265], as a tool to collect human input for multidimensional data clustering problem. We introduce two clustering algorithms, hubCLIQUE and CloCworks that either assist with or entirely rely on assembled crowd solutions to compile a clustering result. We arrange two short term experiments using synthetic and real-world datasets. Using accumulated answers of the `Colony B` players, we benchmark both algorithms against most popular automated clustering methods and show that hubCLIQUE and CloCworks performed on par with or outperformed other approaches. We also examine the microbiome data from American Gut Project [262], one of the largest citizen science microbiome project, and draw some intuition on the modification of the `Colony B` game for more effective use of crowdsourcing technique.

5.4 Colony B

Colony B is a mobile game that addresses the problem of real-world data clustering using crowdsourcing and human computing techniques [265]. This mobile game involves simple actions from a player to provide clustering of data points - encircling a group of similar points (in context of the shown on the mobile screen dataset) both on a single screen or series of screens (the reference to multidimensional dataset). Thus, the clustering criteria is entirely dependent on player choices. Since typical approach for the data clustering routine is an iterative process of data analysis by an expert, the game can be considered as a simplified data review process by a human worker (not necessary an expert).

Every game of Colony B consists of a consecutive set of 2D screens. Each in-game screen is a set of 2D points that the player is able to be group/cluster together. The player is only allowed to create a single group or cluster per screen, which is meant to be the best possible cluster (based on the player's subjective opinion).

Besides, Colony B also has multiple features that encourage crowd engagement (interactive tutorials, theme badges, score, multilevel leaderboard) as well as monitor the performance of the players and direct their efforts towards yet unsolved problems (evaluation puzzles) [265].

5.5 Colony B in multidimensional space

Colony B was designed to capture a signal for any datasets regardless dimensionality. In this section, we discuss specific features of the game that help to address the problem of data clustering in multidimensional space.

5.5.1 Game Data Structure

To make data management and analysis more transparent in the case of a multidimensional dataset, we propose the following data structure (listed in order of biggest to

lowest structural unit):

Puzzle The portion of a multidimensional dataset that will be shown to a `Colony B` player during their game session. Each multidimensional dataset usually contains one or more 'Puzzles'.

Stage The formal term for the 2D screen used in `Colony B` and defines the 2D, orthogonal projections of the data encapsulated into the 'Puzzle'. Multiple 'Stages' make up each 'Puzzle'.

Point An orthogonal projection of a multidimensional data point to 2D space (i.e., 'Stage'). Each 'Stage' is represented as a collection of multiple 'Points'.

Thus, we have defined and build a hierarchical structure for manipulating data found within the game of `Colony B`. It is also important to note that the points from two different Stages within a single 'Puzzle' represent the same multidimensional data points in different 2D orthogonal projections.

5.5.2 Data Adaptation and Interpretation

To create a `Colony B` Puzzle a researcher begins by uploading a multidimensional dataset to our system. Here, we consider a dataset that contains N data points in D dimensional space (i.e., represented as D features).

Since a mobile screen is used as an interface to interact with a player, the game is limited to two dimensions (orthogonal projections) at a time. However, it is rarely feasible to process all $\binom{D}{2}$ distinct 2D projections of the dataset. After experimenting with a different number of the stages per game, we found that the optimal number of stages to be 15. Therefore, we limit dimensionality of the dataset to six.

To accommodate this restriction, many dimensionality reduction techniques (PCA [137], PCoA [138]), feature selection techniques, or iterative random sampling of features exist to address multidimensional datasets. Though, the choice of the strategy must be made in context of the nature of the data, distance metrics used, and other data related properties.

Since we target mobile platforms, we are limited by computational resources and rendering capabilities of a mobile device. We pre-generate a set of 'Puzzles', each of them contains fixed number of randomly selected from a dataset points. We define the number of points as maximum possible number that allows a user smoothly interact with our mobile application using average mobile device. While single puzzle often is a very sparse representation of a dataset, by adjusting the number of puzzles we control its overall coverage.

5.5.3 Game Play Scoring

Each solution collected from a Colony B player contains human clustering result for every stage of a puzzle. In this section, we design multi-phase scoring strategy that takes into account individual stage solutions as well as information extracted from transitions between stages. It is non-penalizing strategy enabling us to filter valid clusters.

First, each stage is evaluated using three clustering validation indices: (i) Silhouette [118], (ii) SDbw [120], and (iii) Dunn [119]. Since our scoring function requires comparison of the indices, we propose a heuristic to normalize them. We observe distribution of values for each index while training on multiple available datasets (see section **Experiments**) . This step is done once, so the produced distributions are then available for all datasets uploaded to the Colony B system.

Thus, we evaluate human solution with three indices and using the reference distributions we determine the portion of the tests, which showed worse result than one evaluated from a human solution. This procedure equally scales all three index measures in the interval $[0, 1]$.

The best value obtained after normalization step is assigned to quality score $Qual(\mathcal{S}_i)$ for the stage $\mathcal{S}_i$. To incorporate dimensionality aspect in the score we introduce a *conservation* score $Cons(\mathcal{S}_i, \mathcal{S}_{i+1})$, the number of points conserved between two consecutive stages. We use a density correction coefficient $\mathcal{DCC}$ to shift a player's attention

towards significantly dense areas. It also aims at reducing spontaneous behavior of a player. The final scoring function for a player's stage evaluation is:

$$Score(\mathcal{S}_i) = \big(\alpha * Qual(\mathcal{S}_i) + (1-\alpha) * Cons(\mathcal{S}_{i-1}, \mathcal{S}_i)\big) * \frac{1+\mathcal{DCC}}{2}$$

Where $\mathcal{DCC}$ is a shifted density value obtained using kernel density estimation [252]. The magnitude of a density shift defines the sensitivity of the score to multiple dense areas within a single selection.

A final score is then summed across all the stages in a given 'Puzzle'.

5.6 Methods

In this section, two novel, semi-automated clustering algorithms for multidimensional data are presented: i) **hu**man-**b**ased **CL**ustering **I**n **QUE**ue (hubCLIQUE) algorithm and ii) **Cl**ustering **O**f **C**rowd-sourced net**works** (CloCworks). hubCLIQUE supplements an available dataset with an additional feature/dimension provided from a `Colony B` human player before performing subspace clustering. In comparison, CloCworks relies entirely on networks resulting from `Colony B` player data and then partitions these networks into clusters.

5.6.1 hubCLIQUE

Many clustering algorithms have shown to produce accurate results in low dimensional space; however, only a few perform adequately in multidimensional space where the curse of dimensionality becomes noticeable [266]. A subspace clustering technique has often been found to yield the best results by identifying clusters that are hidden in specific subspace(s) while presented with noise from other dimensions. Thus, we designed hubCLIQUE as a bottom-up subspace clustering approach guided by crowdsourced solutions collected from the `Colony B` game.

As the core of our method, we chose to use one of the first subspace clustering algorithms, CLIQUE (**CL**ustering **I**n **QUE**ue) [149]. CLIQUE uses grid-based and density-based approach to identify dense areas. Such a strategy has enough flexibility to incorporate additional information extracted from a human input.

CLIQUE adaptations

To adapt the CLIQUE algorithm to human input from `Colony B`, we define a primitive subspace in 2D space (opposed to the 1D space defined by authors in the original paper). We also extend the density term by supplying it with weights observed from human solutions of the `Colony B` game.

We describe two generic ways to encode human input as points weights to affect the density of data points:

Point Frequency Consider a set of multidimensional points $x \in X$. Denote a frequency of players selecting a point x as $f_{clustered}(x)$ and the number of times a point x appeared on a screen of a player as $N_{appeared}(x)$. Then, we define the weight of point x as

$$W(x) = \frac{f_{clustered}(x)}{N_{appeared}(x)}$$

Transition Conservation Frequency Define a frequency of players selecting a point x in two consecutive (in a puzzle) stages as $f^{A \to B}_{clustered}(x)$ and the number of times a point x appeared on two consecutive screens as $N^{A \to B}_{appeared}(x)$. Then the weight of the weight x can be defined as

$$W^{A \to B}(x) = \frac{f^{A \to B}_{clustered}(x)}{N^{A \to B}_{appeared}(x)}$$

Within this paper, we refer to these alternations of CLIQUE algorithm as *All Solutions CLIQUE* (asCLIQUE) and *Conservation CLIQUE* (consCLIQUE), respectively.

Since there are many parameters we can extract from human solutions, we filter out solutions from individuals that did not understand the instructions of the game. Therefore, we can extend the asCLIQUE solution with the following filter criteria:

Confidence Level Evaluation puzzles are used to measure a given player's level of pattern recognition skill (confidence level) at specific moment. Then, we define a player's confidence level as the ratio of a player's score obtained for the evaluation puzzle over its maximum possible score. Confidence levels are specific to each evaluation puzzle. Game solutions are only retained when the player's confidence level exceeds a specified threshold.

Average Cluster Size During the development of `Colony B`, we observed many players selecting larger clusters. We believe players are assuming that there is a proportional relation between the size of the cluster and a score. In actuality, this relationship is somewhat the opposite. Therefore, to correct for this large cluster bias, we estimate an average clusters size (ACS) over all solutions for each player. Then, a player's game solutions are filtered out based on the player's ACS.

These two algorithms are referred to as *Confidence CLIQUE* (confCLIQUE) and *Average Cluster Size CLIQUE* (acsCLIQUE), respectively.

The list of criteria is not exhaustive. Using the same methodology, multiple filters could be defined to filter solutions that come from players, which tend to select one particular part of a screen, have a large ratio of requested puzzles over solved ones, etc. Nevertheless, we do not recommend using some features as a restriction factor. For example, a gameplay score introduces a bias towards a set of specific solutions; the number of games played by a player shows a participation statistics and unlikely demonstrates the quality of player's solutions.

We note that asCLIQUE, consCLIQUE, confCLIQUE, and acsCLIQUE are strategies to assign weights to data points, whereas `hubCLIQUE` is the complete algorithm

that employs a particular weight identification tactic resulting in the final multidimensional clustering vector.

After weights for all points of a 'Puzzle' are computed, subspaces are generated following the original CLIQUE strategy defined in [149]. The process produces multiple subspaces. Each $K+1$ dimensional subspace has at most the same number of points covered by dense areas preserved from each of the two K dimensional subspaces. Therefore, for multidimensional subspaces, low coverage of data points is a frequent phenomenon. This phenomenon makes it problematic to gather and interpret results afterward. For instance, the original CLIQUE algorithm uses a Minimal Description Length (MDL) Pruning [267] that searches for optimal split based on the encoded coverage and its code length. Similarly to the MDL Pruning technique, it is possible to use Elbow criterion [268] (which is often used to find the number of clusters) on the coverage values to identify subspaces to be filtered.

Nevertheless, both strategies propose a set of subspaces and a set of clustering vectors that rarely can represent a single clustering vector for a dataset. Thus, we argue that to compile a final vector we should not prune subspaces with low coverage. Despite the low coverage, high dimensional subspaces contain clusters that have their extension in lower dimensional subspaces and, therefore, must be considered as a base for a cluster compilation routine.

First, we extract single clusters from a set of collected clusters while preserving information about its original dimensions. Then, we sort subspaces based on 1) the number of dimensions and 2) size of the cluster in descending order. It helps to initiate the analysis from potentially the base of a cluster and significantly improves the quality of the result.

We iteratively traverse a sorted list of subspaces to identify overlapping (in terms of containing points) subspaces and then final clustering vector.

Consider a dataset that contains N data points, list S of sorted subspaces of length L. Denote s_i an element of S that contains point information s_i^{points} as well as dimensions of its origin s_i^{axes}.

The algorithm contains two phases. First, we search for the cluster candidates and their maximum axes (dimensions) by measuring the overlap of each element $s_i \in S$ with the rest elements $s_j \in S, s_j \neq s_i$. Then, we finalize the result. We iterate over steadily decreasing set of cluster candidates and remove overlapping elements (if any) from the following candidates. The remaining set of cluster candidates is sorted after every iteration based on the number of dimensions, portion of points conserved between the base candidate (*current*) and other candidates, and the number of points covered by a non-base candidate. The algorithm terminates when the procedure passes over the entire set of cluster candidates or the *current* candidate does not contain any non-overlapping with previous candidates elements. Algorithm 1 illustrates our approach.

Our algorithm produces a single clustering vector that consists of labels of each point in a dataset. Along with non-negative integers, it contains -1 label denoting non-labeled data points.

5.6.2 CloCworks

Previously, we described hubCLIQUE- the clustering algorithm that supplements the dataset with human input. By contrast, in this section, we propose CloCworks - clustering algorithm that leverages human cluster annotations to explore multidimensional datasets. It does not require original dataset and therefore directly shows the behavior of the crowd.

CloCworks operates with data represented as a network. It use community search algorithm to find pseudo optimal partitions of the network (as it is NP hard problem [113] and requires heuristics). We choose Louvain community search algorithm [114] mainly because of its performance (modularity score) and speed in comparison with other related algorithms [115, 116].

Result: Single Clustering vector
candidateClusters ← new List
candidateClustersAxes ← new List
Phase 1
for $i \leftarrow 0$ *to* $L-1$ **do**
  if s_i *is used already* **then** continue
  mark s_i as used
  pointCounter ← new Dictionary
  updateUsageClusters ← new List
  for $j \leftarrow i+1$ *to* L **do**
    if s_j *is used OR* s_j^{axes} *not in* s_i^{axes} **then** continue
    overlap ← $s_i^{points} \cap s_j^{points}$
    if *len(overlap)* > 0 **then**
      pointCounter ← Count each point appearance in s_j^{points}
      if *len(overlap)* $> 0.8 * len(s_j^{points})$ **then** mark s_j as used
      else add j to updateUsageClusters
  if *len(pointCounter)* > 0 **then**
    cutFreq ← average of (values of pointCounter)
    consensus ← keys of pointCounter s.t. value $\geq$ cutFreq
    remove consensus points from S_k where $k \in updateUsageClusters$
    append consensus to candidateClusters
    append s_i^{axes} to candidateClustersAxes
Phase 2
for $i \leftarrow 0$ *to len(candidateClusters)* - 1 **do**
  current ← candidateClusters[i]
  augmented ← new List
  for $j \leftarrow i+1$ *to len(candidateClusters)* **do** add len(current ∩ candidateClusters[j]) to augmented
  Sort candidateClusters[i+1 :] **begin**
    let f ← index of element
    1: candidateClustersAxes[i+1+f] (**desc**)
    2: (len(current) + augmented[f]) / (len(current) + len(candidateClusters[i+1+f])) (**asc**)
    3: len(candidateClusters[i+1+f]) (**desc**)
  for $j \leftarrow i+1$ *to len(candidateClusters)* **do**
    candidateClusters[j] ← candidateClusters[j] \ candidateClusters[i]
clusters ← new List(N)
fill clusters with -1
for *clusterID* $\leftarrow 0$ *to len(candidateClusters)* **do** clusters[candidateClusters[clusterID]] ← clusterID
return clusters

Algorithm 1: CLIQUE results are compiled into single clustering vector for complete dataset.

Design

The algorithm first compiles a network for every pair of dimensions in a dataset from `Colony B` players' solutions. Each node of the resulted network is a data point whereas an edge, and its weight represents similarity between pairs of points and its strength, respectively.

Denote $S_k^{i,j}$ a stage in 2D space $\{i, j\}$. The algorithm analyzes all accumulated solutions for $S_k^{i,j}$ and counts the frequency of two points being selected together by a `Colony B` player over the number of times the stage was shown to a player. The result is recorded as a similarity matrix. Then the procedure is repeated for all the stages in $\{i, j\}$ subspace. All resulting similarity matrices are averaged based on the frequency of two points being assigned in the same stage (see *Supplementary Materials*: Fig. 5.5).

Next, we partition the resulted network using Louvain community search algorithm. All partitions of networks (aggregated for each pair of dimensions) are considered as a list S of single clusters s_i. In contrast to the similar list operated by hubCLIQUE (algorithm 1) that contains subspaces of different dimensionality, CloCworks utilizes only 2D clusters. Thus, we modify the algorithm 1. Since we have less information about consensus for higher dimensional subspaces, we relax the following conditions: i) we iterate over S while non-used clusters exist, ii) after each iteration we sort S based on the $length(s_i^{points})$, iii) we use double deviation from the mean of the *pointCounter* values as *cutFreq*. These modification allows us to take into account combinations of clusters (if any) and later try to split them using supplementary information from other projections.

CloCworks can be considered as a modification of the correlation clustering approach applied to general weighted graphs [269]. However, instead of $\pm$ labeled edges with a weight denoting the confidence of the labeling, the proposed algorithm operates only with pairwise point similarities encoded as edge weights. Also due to very sparse dataset coverage by puzzles of the `Colony B` game, very few network clustering

algorithms can be used. For example, algorithms operating with the clustering coefficient and transitivity [270] will perform poorly since the chance of finding triangles in the network observed from `Colony B` is very low. Also spectral clustering [271] algorithm will generate multitude of small clusters that in our case is not representative.

5.7 Experiments

In this section, we benchmark our algorithm against standard clustering algorithms as well as original CLIQUE algorithm. We use the synthetic dataset as well as the real-world dataset to assess the accuracy of the proposed algorithm. Both datasets were played by `Colony B` players for two weeks. Since the `Colony B` game can be played on most mobile devices, we do not restrict or control both players' registration and their contribution to science. Therefore, the number of players played particular dataset and the number of submitted solutions can vary and will be explicitly reported in the description of experiments.

5.7.1 Synthetic dataset

First, we generate a synthetic dataset with high-density clusters in specific subspaces. Unlike the "Synthetic data generation" procedure described in [149], we avoid using predefined hyper-rectangles and their connectivity. Instead, we define dimensionality (for simplicity we choose six to comply with requirements of `Colony B` game) and the approximate size of a dataset, number of clusters, and ratio of additional noise. The following procedure is randomized. For each cluster, it randomly (with replacement) selects its dimensions from all the possible combinations of size in the range $[2, 6]$ as well as its size, coordinates of the center (mean), shape and orientation (covariance). All these parameters are then used to draw random samples from a multivariate normal distribution. We restrict the minimum Manhattan distance between means of the clusters that share at least one dimension:

TABLE 5.1: Synthetic Dataset information

Subspace	Cluster ID	Size
{0,3,4,5}	0	338
{1,2,4,5}	1	328
{0,1,3,4,5}	2	334
{0,1}	3	340
{0,4}	4	335
{0,4}	5	339

$$dist_{Man}(\vec{mean}_1, \vec{mean}_2) > dist_{Man}(\vec{0}, \sqrt{\vec{cov}_1} + \sqrt{\vec{cov}_2}),$$

where $\vec{mean}_x$ and $\vec{cov}_x$ are the mean and covariance vectors for elements of cluster x in the shared subspace, respectively.

For our experiment, we choose to generate a 6D dataset with a size of approximately 2000 data points, which are randomly distributed over six clusters of various dimensionality. Also, we added extra points ('noise' points) randomly distributed over the search space (5% of the dataset size). Table 5.1 shows generic information about the synthetic dataset.

To measure the accuracy of the algorithm, we use F1 score with micro averaging [272], which globally counts true positives false positives and false negatives rates to compute the average metric. This approach allows us to understand which clustering result was the closest one to the true labels.

We compare the performance of four proposed extension of hubCLIQUE algorithm (asCLIQUE, consCLIQUE, confCLIQUE, acsCLIQUE) as well as CloCworks with the most popular clustering algorithms. Using `Colony B` game we collected human input for the proposed algorithms: 25 players submitted over 400 solutions during two-week period.

We include in the test three categories of algorithms: i) original CLIQUE [149] algorithm; ii) algorithms that require a prespecified number of clusters (KMeans [91], Affinity Propagation (AP) [273], Hierarchical clustering using Ward's minimum variance method (Ward) [98], Gaussian Mixture Models (GMM) [105]); iii) as well as those

that do not (DBSCAN [101], MeanShift [92]).

We note that the GMM might converge to arbitrarily incorrect local maxima even in perfect conditions [106]. Therefore, we report average of F1 score for 1000 runs instead.

Since our hubCLIQUE algorithms do not require the number of clusters, for the second category we assume two different cases: i) the number of clusters is estimated using elbow method (4); ii) the correct number of clusters is given (6). We use the former for the main comparison while the latter as a reference only.

Also our test dataset consists of clusters in both high and low dimensional subspaces. Thus, we separately report the accuracy for both groups independently as well as for whole dataset. Results are shown on Figure 5.1.

The underlying technique in all hubCLIQUE algorithms allows us to emphasize multidimensional clusters since it does benefit from the solutions collected from various projections of the dataset. This is clearly illustrated in Fig. 5.1C. Point weights obtained from human input help to eliminate points that otherwise get selected by original CLIQUE algorithms, which, in turns, significantly improve the result (increase of 12.3%, 13.4%, 12.5% and 6.5% in micro F1 score for asCLIQUE, consCLIQUE, confCLIQUE, acsCLIQUE, respectively).

It is interesting that DBSCAN shows average performance for high dimensional clusters, whereas it performs poorly for low dimensional ones. Detection of those clusters is a very challenging task for all clustering algorithms since the process is heavily affected by the imperfections of the data mapping on the subspaces that do not include the clusters. hubCLIQUE algorithms show one of the best results among the tested algorithms.

Our experiment also shows that the GMM in average demonstrates superior performance for both cases. However, it yields not stable results (we report only average score over 1000 runs) so that its accuracy can significantly degrade both for low and high dimensional clusters.

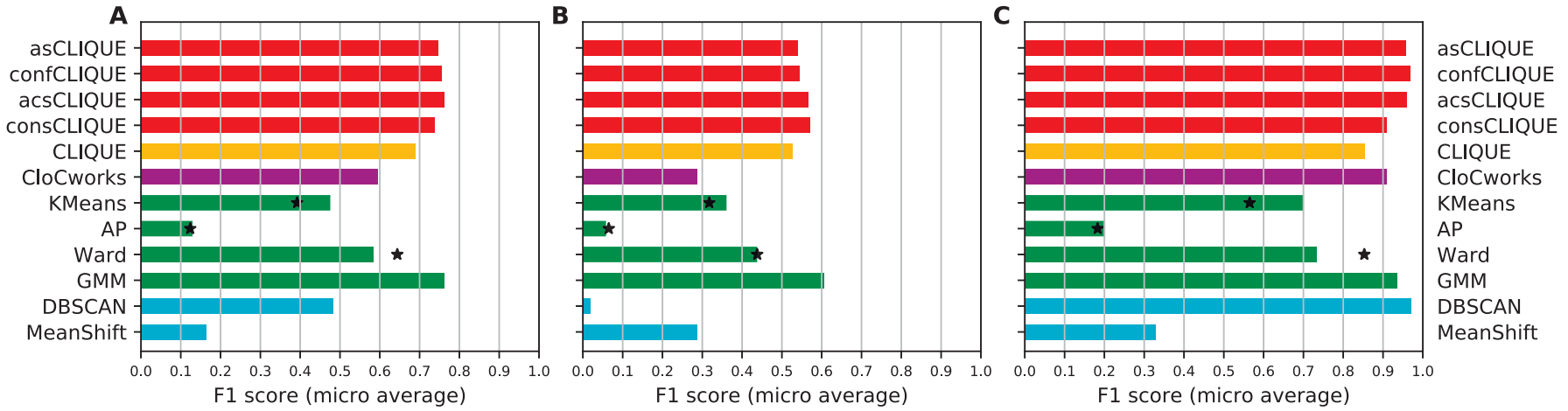

FIGURE 5.1: Performance comparison of the human-based algorithms with automated clustering approaches applied to the synthetic dataset for A) all clusters; B) low dimensional clusters; C) High dimensional clusters. The color scheme is used to separate groups of algorithms: human-based CLIQUE algorithms (red), automated CLIQUE (orange), CloCworks (purple), algorithms that require known number of clusters/components (green), algorithms that do not require the known number of algorithms (cyan). GMM is a special case for which we report average of the metric over 1000 runs. For algorithms that require known number of clusters, we report the performance of an algorithm with number of clusters estimated by elbow heuristic (4). An asterisk (star) shows its performance with correct number of clusters (6).

Although, CloCworks significantly differs from the other algorithms and considers human input as a main and only source of information about the dataset, it performs on par with the best performers in this test for high dimensional clusters. For low dimensional clusters, in contrast to the hubCLIQUE approaches, it shows a less accurate result. Nevertheless, the algorithm detects the signal while suffering from the sparsity of the networks as well as the imperfection of the human solution. Both aspects of the algorithm will be addressed in future work.

5.7.2 Synthetic networks

Since CloCworks does not deal with dataset directly, it is challenging to compare its accuracy with other clustering algorithms. Instead, we investigate its performance within a controlled environment. We generate multiple sets of `Colony B`-style networks corresponding to various configurations (different distribution of edges and their weights) using the Stochastic Block Models (SBM) [274] network generator. Similar to [274], we consider n vertices, which are split into k communities of variable size. We connect nodes independently with probability $p_{i,j}$ where $i, j \in [k]$. Instead of generating equally weighted edges, we assign the weights $W_{i,j} \pm \sigma_{i,j}$ where $i, j \in [k]$ and $\sigma_{i,j}$ is an expected standard deviation.

Similar to the previous experiment, for each set we build 15 networks with the same layout (Table 5.1). For each network, the weight distribution follows the rule $0.8 \pm \sigma / 0.4 \pm \sigma / 0.1 \pm \sigma$ for intra-cluster, inter-cluster and noise-related edge, respectively. We choose an expected standard deviation $\sigma_{x,y} = 0.1$ regardless the xy clusters relationship. The only variable is the probability of observing the intra cluster edge p_{intra}, which varies in range $[0.005, 1]$. Inter cluster and noise-related edges are generated with probabilities $p_{inter} = \frac{1}{2} p_{intra}$ and 0.2 (constant), respectively.

We also report performance of CloCworks for high dimensional clusters (Cluster ID $\{0, 1, 2\}$), low dimensional clusters (Cluster ID $\{3, 4, 5\}$), as well as for entire dataset (total score). The result of the experiment is shown in Figure 5.2.

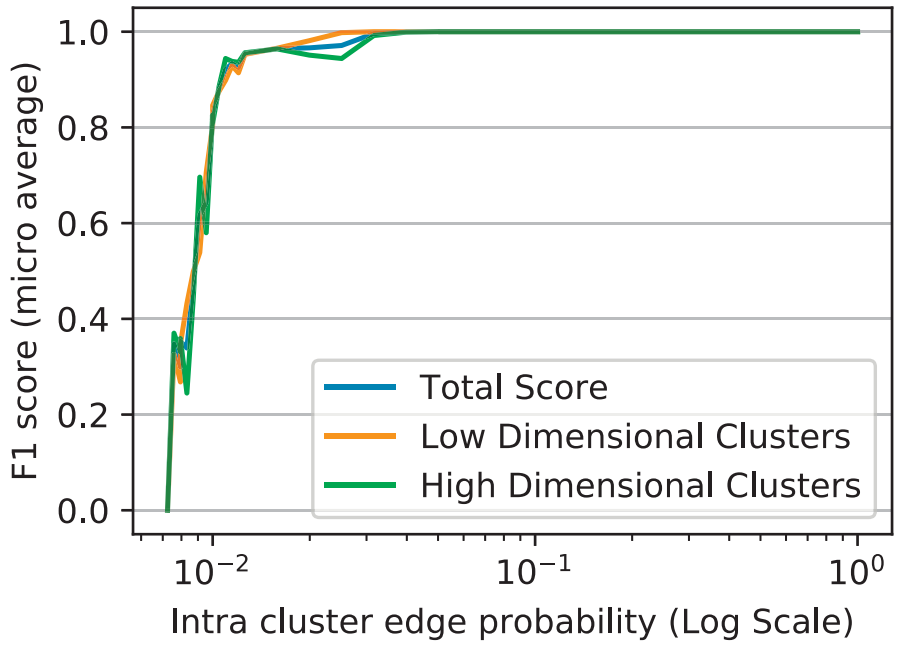

FIGURE 5.2: Analysis of the CloCworks algorithm's performance applied to a series of `Colony B`-style simulated networks using Stochastic Block Models (SBM) data generator. Networks were generated with the probability of observing intra-cluster edge p_{intra} in range $[0.005, 1]$, inter-cluster edges - with probability $p_{inter} = p_{intra}/2$, and noise-related edges - with probability $p_{noise} = 0.2$ (constant). The weight distribution follows the rule $0.8 \pm \sigma / 0.4 \pm \sigma / 0.1 \pm \sigma$ for intra-cluster, inter-cluster, and noise-related edges, respectively. σ is an expected standard deviation and is assigned to 0.1. CloCworks was applied to every set of networks. Micro F1 score was used to estimate total, low dimensional cluster, and high dimensional cluster scores.

With $p_{intra} = 0.01$ we can see F1 score of 0.88 for all three measures. This is expected result since in this case for the abstract cluster containing 300 points (nodes) in average each node would connect with only three other nodes. Reducing the intra-cluster probability will cause an appearance of disjoint parts of networks, and, therefore, can result in multitude of small false clusters. Such an accuracy shows that it is possible to recover the majority of communities (clusters) from the dataset uploaded to `Colony B` given that the game puzzles cover a dataset extensively.

5.7.3 Voice Recognition Dataset

To examine the performance of proposed algorithms applied to the real-world dataset, we apply our techniques to a voice recognition dataset (VRD) [258], which consists of over 20 measures of either male or female voice parameters.

As a preprocessing step, we use a random forest classifier [275] (fitting 25 trees) on VRD dataset with predefined *male/female* labels as a target to identify six features that can be used for the `Colony B` game. Extracted features (*meanfun*, *IQR*, *Q25*, *sd*, *sp.ent*, and *sfm*) are then used to create `Colony B` puzzles. VRD related puzzles were played by 75 volunteers with over 700 solutions submitted during a two-week period.

Similar to the synthetic dataset experiment, we benchmark our hubCLIQUE, CloCworks approaches against CLIQUE, KMeans, Ward, GMM, and DBSCAN (Fig. 5.3). We intentionally skipped AP and MeanShift due to their average performance in both tests. For algorithms that require known number of clusters (green category) we report an algorithm's performance with the number of clusters estimated with elbow method (3) as a bar value, and a metric value obtained with correct number of clusters given (2) as an asterisk (star). We also used elbow method to find the required DBSCAN parameters (*eps* and *min_samples*).

Our results show that hubCLIQUE approaches in general performs on par with CLIQUE algorithm and also algorithms that were provided with correct number of clusters. Though, we can see significant decrease in performance for Ward and KMeans

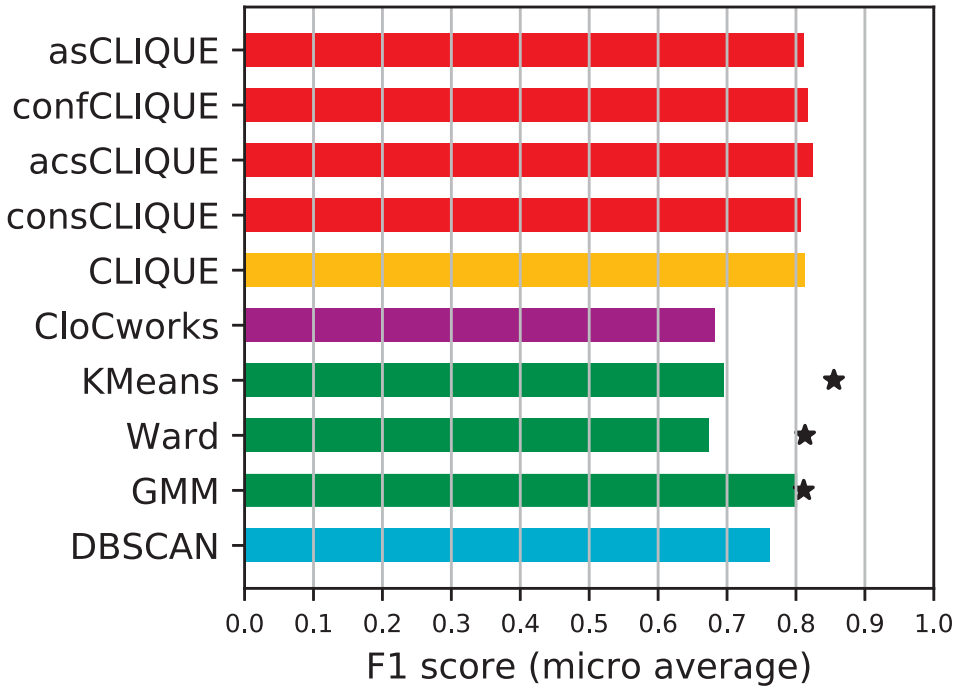

FIGURE 5.3: Performance comparison of the human-based algorithms with automated clustering approaches applied to the Voice Recognition dataset (VRD). The color scheme is used to separate groups of algorithms: hubCLIQUE algorithms (red), automated CLIQUE (orange), CloCworks (purple), algorithms that require known number of clusters (green), algorithms that do not require the known number of algorithms (cyan). For the green category the bar value shows an algorithm's performance with the number of clusters estimated with elbow method (3). An asterisk (star) marks the metric value obtained using correct number of clusters (2).

with the estimated number of clusters. In is interesting that GMM performed well in both cases. The reason is that the misclassified points for the case of correct number of clusters (2) were mostly redistributed in the extra cluster added in case of estimated number (3).

By contrast, performance of DBSCAN is average. Mostly because it identified single major cluster with a lot of outliers. This result illustrates that the density based clustering approach, which considers a high dimensional dataset as is, might present a challenge to identification of sub-optimal parameters for the algorithm to find desired clustering result.

5.7.4 American Gut Dataset

Initially, we designed the `Colony B` game as a proof of concept of the usefulness of human intelligence in multidimensional data clustering applied to the microbiome dataset collected by the American Gut Project (AGP) [262]. AGP is a large citizen science microbiome research that collects, compares, and analyzes human microbiome specimens from all over the world [276]. Aiming to build a complete map of the human microbiome, it attempts to learn how lifestyle (later referred to as metadata) affects gut microbiome (later referred to as microbiome data) using citizen science effort.

In contrast with our previous experiments, the dataset does not have given labels to validate the performance of our approaches. However, recently McDonald et *al*. [262] showed a positive correlation between microbiome data and metadata for 9,511 individual participant samples (weighted/unweighted UniFrac [277] effect sizes). This result inspired us to explore the microbiome data given the top (in terms of effect size) metadata attribute to be the clusters.

Due to the difficulty of interpretation of OTU [278] tables, we decided to work with PCoA [138] reduced dataset provided by AGP, which is routinely used for their analysis [277]. We aimed to find the best set of dimensions that can be used for the `Colony B` game. First, we filtered the best hundred dimensions based on the variance (data

contains approximately 40% dimensions with zero variance). Then, following AGP strategy, for every metadata attribute, we examine every pair of attribute values and define their "separation" in the related microbiome data. For example, we explore the 'AGE_CAT' attribute with values ('BABY', 'CHILD', 'TEEN', '20S', '30S', '40S', '50S', '60S', '70+'). For each pair of values, such as 'CHILD' and '60s', we extract relevant samples from microbiome data and explore the distinction of these two groups. To achieve this, for every small experiment we employ random forest classifier as a binary classifier compiled with 500 individual trees. Using trained models, we extract feature importance, estimate AUC score, and for those whose score is higher than 0.55 we find the most frequently used six dimensions. We suggest that the resulting dimensions are good candidates to explain the majority of the metadata attribute value pairs.

Since the PCoA provides us with dimensions already sorted by importance, we expected that the chosen dimensions to be the first six dimensions in the microbiome dataset. Surprisingly, we observed the lack of strong domination of first dimensions in microbiome data with a slight variability in the results (sorted by rank dimensions $[2, 1, 0, 8, 4, 6]$).

Next, we eliminate (partially or completely) metadata attributes, whose values cannot be explained by any of the resulting dimensions. We also try to filter out samples that do not mainly operate with the preserved attribute values. Since the strict deletion of points with unimportant attribute values would reduce any relevant data point from the dataset, for each point we estimate the number of times it uses one of the preserved attribute values. We use the median value of the frequencies as a threshold to filter out the points with only few preserved attribute values.

Then, we visually examined multiple top rated attributes ranked by AGP (based on the effect size). One of the illustrations related to AGE_CAT metadata attribute is presented in Figure 5.4. To visualize separation of the data (if any), we transformed the 6D to a 2D dataset using tSNE [140], mapped data points to the corresponding metadata attribute values. We chose distinct colors to represent the variety of the attribute

values for AGE_CAT metadata attribute (Fig. 5.4A). The plot shows that there is close to no separability between all the attribute values for particular metadata attribute. It suggests that if the `Colony B` players are provided with (not transformed) microbiome data orthogonal projections during the game (Fig 5.4B), apart from demotivating them, collected cluster annotations will likely be random.

We also repeated the same analysis showing only two attribute values at the same time. For the illustration, we chose 'CHILD' and '60S' values of AGE_CAT metadata attribute as AGP reported the strongest effect size for these attribute values (Fig 5.4C, 5.4D).

We tested our results for the separation of two groups of points ('CHILD', '60S') using two tail Kolmogorov-Smirnov test [279]. The null hypothesis is that two samples are drawn from the same distribution. For the tSNE-transformed dataset (Fig. 5.4C), the test rejected the null hypothesis with $p < 0.0015$, where significant effect is obtained with 8.5% of the samples (30). For the $\{0, 1\}$ projection (Fig. 5.4D) of original dataset, the test rejected the null hypothesis with $p < 0.0002$, where the effect is found with 14.3% of the samples (50).

5.8 Discussion

In this paper, we implemented an online human computing game `Colony B` as a complex framework to collect human player input in order to solve the problem of multidimensional data clustering.

We introduced two crowdsourced multidimensional data clustering approaches (hubCLIQUE and CloCworks) that utilize human pattern recognition skills to compile final multidimensional clustering result. While the hubCLIQUE uses collected human annotations as an assisting information for a density correction within CLIQUE subspace clustering algorithm, the CloCworks uses Louvain community search algorithm to find pseudo optimal partition of the network built entirely on aggregated human solutions.

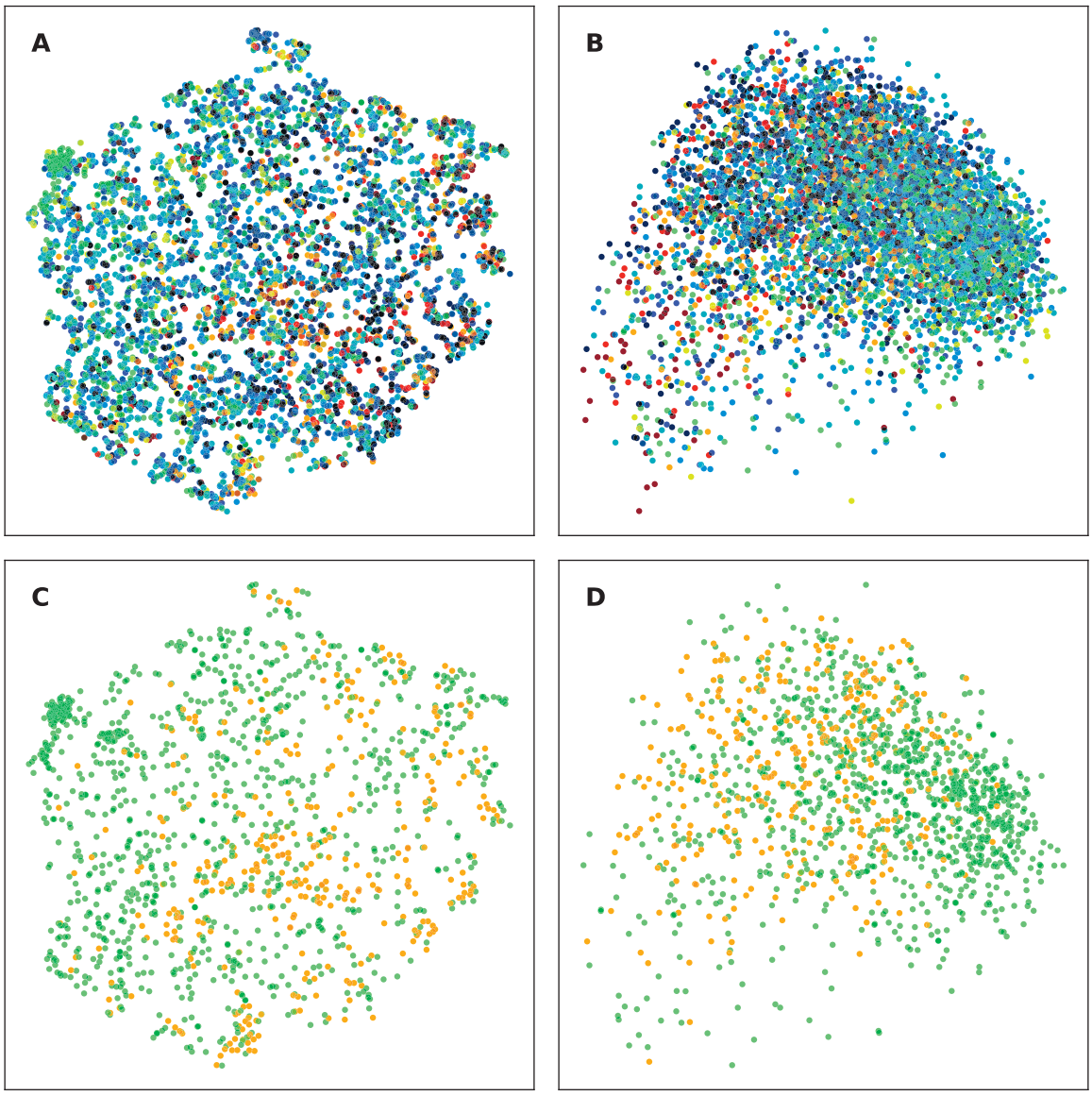

FIGURE 5.4: Separation of the microbiome data points for AGE_CAt metadata attribute values. A) Distribution of the AGE_CAT attribute values shown on tSNE ($6D \rightarrow 2D$) transformed dataset; B) Distribution of the AGE_CAT attribute values shown on the $\{0,1\}$ projection of the microbiome dataset; C) Distribution of two AGE_CAT attribute values ('CHILD', '60S') shown on tSNE ($6D \rightarrow 2D$) transformed dataset; D) Distribution of two AGE_CAT attribute values ('CHILD', '60S') shown on the $\{0,1\}$ projection of the microbiome dataset. ALL) Distinct colors represent different attribute values; if not specified, attribute values shown are 'BABY', 'CHILD', 'TEEN', '20S', '30S', '40S', '50S', '60S', '70+'.

We conducted two short term experiments using synthetic dataset and real-world dataset that involved 100 volunteers with over 1100 solutions submitted. Based on the accumulated results we benchmark both our proposed algorithms against most popular automated clustering algorithms.

For the synthetic dataset, we showed that the hubCLIQUE approach outperformed its competitors in identifying both high and low dimensional clusters. The CloCworks algorithm also showed competitive results for high dimensional clusters. It appeared to be less efficient with low dimensional clusters. However, since the algorithm considers human input as a main and only source of information about the dataset, it performed on par with the rest algorithms. Besides, we observed similar results for the real-world dataset for both hubCLIQUE and CloCworks.

We note that hubCLIQUE benefited from human input much more in case of the synthetic dataset as the game stages showed less structural uncertainty.

As stated before, these results were observed using the data accumulated during two-week period with out `Colony B` game. It illustrates that it is possible to analyze a dataset using players with unknown knowledge of the underlying problem in very short amount of time given that i) the game rules are transparent; and ii) the game preserves a natural game progression balancing challenge level with the skill level of the player [250].

Also, we tested CloCworks algorithm by applying it to a set of `Colony B` simulated networks generated using Stochastic Block Model network generator [274]. We examined its performance on a wide variety of configurations of the networks (from sparse to fully connected) and identified that the algorithm starts to misclassify data points in very sparse networks (probability of observing intra-cluster edge $p_{intra} < 0.01$). This result confirms that the puzzles' coverage of the data points in the `Colony B` game can be rather sparse, yet, it still preserves a network structure for clustering in multidimensional space.

Nevertheless, we found that the CloCworks algorithm is sensitive to the outliers and can be confused by a misleading human solution. This limitation should be addressed

in future work.

Besides, we analyzed datasets provided by the American Gut Project (microbiome and metadata). We found that while microbiome data clustering still remains an open-ended problem and states a challenge to extract data that can be refined by the `Colony B` players, it is possible to separate pairs of metadata attribute values both on a data projection as well as a tSNE transformed dataset.

This phenomenon inspired us for designing a prototype of the two-player mode for the `Colony B` game (section 5.9.3). This concept will be implemented in future iterations of the `Colony B` game and its effect should be further studied in future work.

Funding

This work was supported by the Genome Canada, CIHR, BCB competition.

Acknowledgements

We would like to thank members of our laboratories for meaningful advice and recommendations regarding the `Colony B` game design and data analysis methodology, all `Colony B` players for their invaluable contributions to science. Also we are grateful for the help from the American Gut Project team.

5.9 Supplementary Materials

5.9.1 `Colony B`: Normalization of Clustering Validation Indices

The `Colony B` scoring function, first, compares three clustering validation indices (Silhouette [118], SDbw [120], and Dunn [119]), and then chooses the best result as a quality score for a human solution. However, these indices can not be compared directly as their mathematical definitions differ. We propose a simple heuristic to normalize cluster validation indices.

First, we modify indices in such a way that they monotonously increase with a quality of a solution.

Silhouette score $V_{Sil} := max(0, V_{Sil})$, since negative value indicates that points from different groups are more similar than points within the same group;

SDbw score $V_{SDbw} := 1/(1 + V_{SDbw})$, as the score is monotonously decreasing while solution's quality improves;

Dunn score remains unchanged because it satisfies our requirements

Then, the idea of our approach is to align distributions of the indices' scores observed from trials of multitude of various clustering situations. For a given index score V_{index} we use the probability of observing score worse than V_{index} ($P_{index}(v < V_{index})$) as a normalized score. For a training purposes, we use available puzzles from the database and their stages as datasets for KMeans [91] and Spectral clustering [280] algorithms to provide various partitions of a data set. Each partition is then evaluated using all three indices. Collected statistics for each index is then used as a reference distribution to evaluate $P_{index}(v < V_{index})$. Thus, this procedure equally scales all three index measures in the interval $[0, 1]$ and, therefore, enables us to compare different indices.
Example. Consider human solution $\mathcal{S}_i$. Let v_{Sil}, v_{SDbw}, and v_{Dunn} are the measures of Silhouette, SDbw and Dunn indices, respectively. Then, using previously obtained reference distributions we identify the quality score of a solution as follows:

$$max\big(P_{Sil}(v < v_{Sil}), P_{SDbw}(v < v_{SDbw}), P_{Dunn}(v < v_{Dunn})\big)$$

.

5.9.2 CloCworks: Human Input Aggregation

Figure 5.5 illustrates CloCworks approach of generalization of Human annotations collected with the `Colony B` game. First, human solutions (Fig. 5.5A) are transformed into

networks, where the presence (or absence) of the edge between two nodes (points) determines if the nodes appeared in the same selection. (Fig. 5.5B). Next, individual networks are averaged based on the frequency of two points being selected in the same stage (Fig. 5.5C).

5.9.3 `Colony B`: Two-Player Mode

Inspired by our analysis of the AGP dataset (5.7.4) we design a prototype of a two-player mode for the `Colony B` game.

We present to two players playing simultaneously one 2D dataset containing two groups of points or any other thematic objects (related to the metadata attribute values) colored red and blue. We also assign one of these colors to each player. The objective of the game is then to catch as many points of the opponent's color as possible with the least possible number of player's color. The players take turns and are limited by circular shape selection. During the game, players can be exposed to various restrictions (e.g., based on time or the number of turns). We employ a simple scoring function: the number of opponent's points minus the number player's points covered by a player's selection.

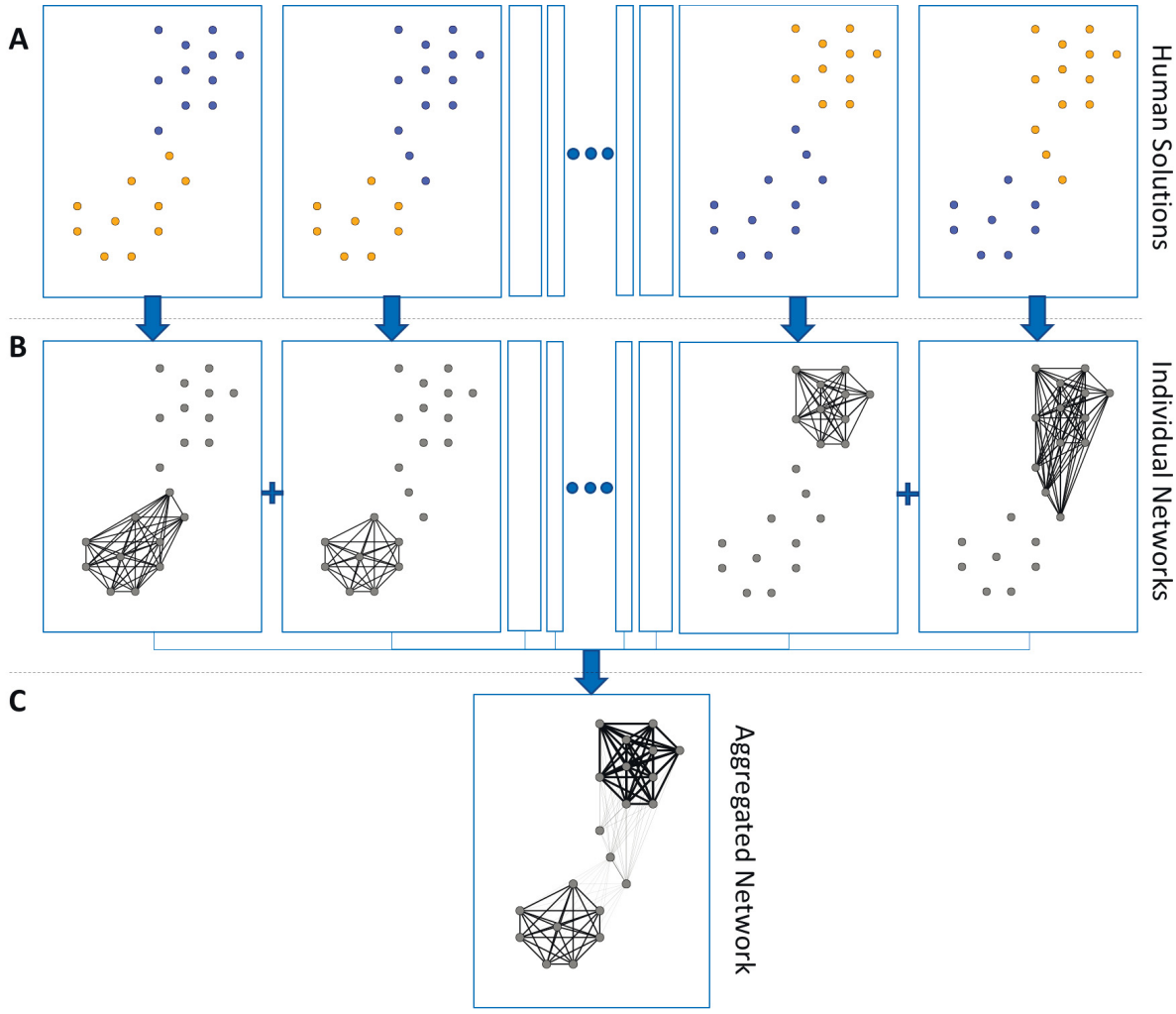

FIGURE 5.5: CloCworks. The process of generalization of Human annotations collected with the `Colony B` game. A) Set of human annotated clusters (orange), shown similar to the in-game data representation style. B) Each human annotation is converted into a network, where an edge between two points indicates points' appearance in the same cluster. C) Individual networks are averaged based on the frequency of two points being assigned in the same stage.

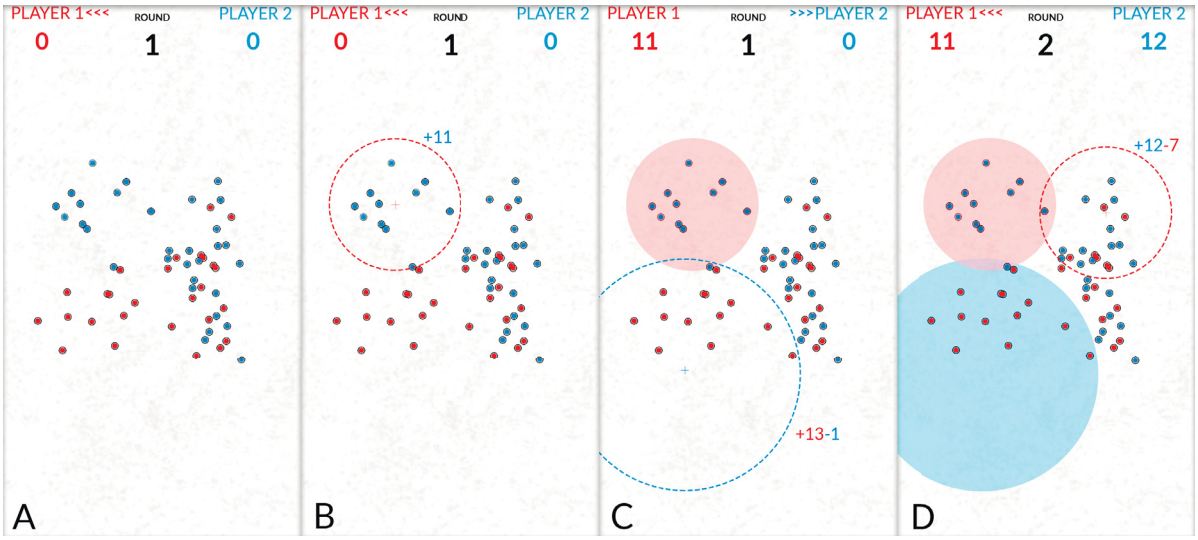

FIGURE 5.6: Series of screens for two-player mode of the `Colony B` game (concept): A) the initiation of the game (player 1 is active); B) "catching" points of player 2 (player 1 is active); C) "catching" points of player 1 (player 2 is active); D) "catching" points of player 2 (player 1 is active)

Chapter 6

Conclusion

Recent advances in machine learning allow human-driven analysis to become completely automated. However, computer programs increasingly need human guidance and intuition to assess some problems (e.g., data ranking, clustering [80]).

Human computation is a well-established technique that helps researchers to engage human participants efficiently to solve a given problem partially or entirely. In part, this technique imposes data-related (e.g., problem decomposition, solution assembly strategy) and human-related (e.g., player motivation, data visualization and interpretation from the perspective of HCI) questions. Answers to those type of questions will help to create a more beneficial system for both researcher and a participant.

In this thesis, first, we present a novel 3D genomic data visualization strategy (Chapter 2). Then, we explore human computing systems in the context of i) the player motivation in collaborative solving (Chapter 3), ii)system design and problem decomposition (Chapter 4), and iii) different assembly techniques for accumulated human answers (Chapter 5).

6.1 Summary of Contributions

In Chapter 2 we describe the problem of visualizing three-dimensional (3D) genomic data such that it is interactive in a 3D space. We present 3D Genome Browser (3DGB), an immersive web-based 3D genome browser interface, which is built upon a low latency database. 3DGB enables smooth data browsing of the 3D genomic structures

while in real-time obtaining high-resolution 3D meshes that represent the DNA backbone within any standard web-browser with support of dynamic Javascript (i.e., Internet Explorer, Safari, Google Chrome, etc.). By request, 1D standard genome browser [208] bound to the current representation is shown to a user for in-depth investigation of a particular segment of the human genome. Additionally, we design a system to dynamically map custom genotyping data to the 3D structures available in 3DGB and visualize it for further analysis. Besides, we describe multiple web-services that provide with access to i) 3D structures, ii) nucleotide sequences, iii) nucleotide coordinates, iv) single nucleotide polymorphisms, and v) experimental ChIP-Seq data. Then we design the 'scripting' user interface that allows a user to explore the data within a web browser using JavaScript. We demonstrate the use case of 3DGB scripting tool by exploring the 3D neighborhood of Retinoblastoma 1 gene (RB1) transcription start site and identifying the only SNPs found previously in the chromosome 13 related to sleep disorders.

Thus, the work presented in Chapter 2 describes a novel HCI visualization system that provides with a set of tools to simplify access to the complex 3D genomic data. By designing this system, we aim to gain an experience with designing HCI systems that are convenient to use for humans to interpret the data.

Next, we focus on the design of human computing systems. In Chapter 3, we approach the problem of large scale collaboration in HCI human computational systems. We present the 'Market game', a multi-player game-with-a-purpose that uses market simulation to create a competitive yet collaborative environment for players to solve a classical graph theory problem (clique). This work aims to understand how humans collaborate and what incentives are needed to promote collaboration within a human computation system. We analyze the impact of the multiple game mechanisms on the performance of the system. Our results highlight the importance of a collaborative environment in the game, as well as other motivators such as skills and challenges that help to shift players' attention towards underexplored parts of the underlying problem. By our knowledge, this study is the first application of real-time collaborative

problem solving involving a large number of participants.

Next, we investigate an application of human computation techniques in cluster analysis. In Chapter 4, we aim to characterize the collective perception of 2D clusters by humans. To address this problem, we present `Colony B`, a mobile online human computing game that is used as a framework to collect human annotations for 2D clustering problems. We present strategies enabling us to circumvent biases observed in cluster annotations collected through mobile devices. Also, we demonstrate that for mobile human computing games due to the recent tendencies of smartphones with a large screen, the interactive panel must be located at the bottom of the screen whereas the relevant game information must be placed on top. We conduct a long-term experiment asking hundreds of participants to cluster points from real and simulated datasets. Our analysis shows that aggregated human solutions are significantly larger than the ones generated by computer programs while providing approximately the same quality of clusters.

Finally, in Chapter 5, we approach the problem of higher dimensionality data clustering. We use our mobile human computing game `Colony B` to collect players' clustering solutions for simple 2D datasets, which are essentially a series of orthogonal (axis-parallel) 2D projections of the original dataset. We present two clustering algorithms (hubCLIQUE and CloCworks) that use aggregated human solutions collected from `Colony B` players to identify key features in real-world and synthetic high dimensional datasets. We conduct two small-scale experiments to collect human clustering solutions and then to benchmark hubCLIQUE and CloCworks against conventional clustering approaches. Our results demonstrate that the proposed algorithms perform on par or better than most automated clustering algorithms considered. Moreover, we analyze the high dimensional microbiome dataset obtained from the American Gut Project [262] and discuss possible modifications of the `Colony B` game to use human computation technique efficiently.

6.2 Perspectives on Future Work

Several directions can be considered to improve for both genome visualization tool and human computing approaches described in this thesis.

The current scope of data available for 3DGB to be visualized is somewhat limited [184, 191]. Recently multiple 3D genomic models were released [281], which can be directly integrated into the database. Besides, 3D genome structure reconstruction algorithms, such as 3DMax [282] and LorDG [283], provide with a tool to utilize available HiC generated datasets [284]. Similarly, the genome structure can also be approximated from ChIA-Pet data using the approach proposed by Tang et al. [285]. Thus, these techniques might allow us to use raw experimental data instead of relying on the previously compiled 3D genomic models. Moreover, a chromatin interaction map presents a critical viewpoint on the genome architecture [172] and, therefore, is an essential part of a 3D genome browser. Such a feature can be shown in 3DGB as an alternative visualization mode available for a user.

Due to memory management, 3DGB has a limitation on the amount of information shown to the user at a time. Space in 3DGB is represented as a cubed grid containing spatial genomic information. Thus, a user is able to observe data within two cells from the current cell, whereas outside of this neighborhood it is erased. While each model is scaled manually for each portion of data to be informative, there is a lack of flexibility in scaling the model (i.e., zoom in/out) that needs to be addressed.

Virtual Reality (VR) is an emerging technique that allows a user to explore genomic structure from different angles in the immersive environment [286]. We are working on VR mode for 3DGB [287] that will be available both as web-based and stand-alone application.

The study presented in Chapter 3 shows the importance of a collaborative environment in human computing systems using the market game. The game is designed as a proof-of-concept and currently limited to offline use (i.e., during game sessions). Also, since during our tests, we have noticed that having more players is beneficial to

the system, an implementation of the online version of the game will allow reaching a broader audience and thus improve the efficiency of the system.

The `Colony B` game described in chapters 4 and 5 makes two assumptions/limitations on the underlying dataset. First, to explore all orthogonal (i.e., axis parallel) projections of the dataset, we reduce the number of dimension to six. While this arrangement allows the system to examine a dataset the most, feature selection/extraction technique might eliminate meaningful for signal detection dimensions. The consideration of a entire feature set of a dataset during the game often is not feasible (the number of 'Stages' for a single game would be quadratic to the dimensionality of a dataset). In Chapter 5 we showed that the human clustering results obtained during a short term experiment are sufficient for exploring a portion of a dataset. Therefore, iterative exploring a random six-dimensional projection of a dataset might provide better coverage of a high dimensional dataset. The second limitation of the `Colony B` game is a lack of global context in the game (i.e., a player sees only a random sample of data points [86]). Increase the sample size, which implies optimizing the mobile application, can help to eliminate cluster artifacts and consequently improve the accuracy of collected results. `Colony B` also requires more elaborated game flow [250], a game progression that balances challenge level with the skill level of a player. This also raises an interesting question about the ranking of `Colony B` puzzles that need to be addressed in future work.

In addition, in Chapter 5 we propose two-player mode for `Colony B` that presents the same dataset to two players simultaneously. This mode will be used to promote collaboration between `Colony B` players in solving the data clustering problem in the context of supplementary metadata.

Taken together, the methods described in this thesis provide comprehensive analyses of HCI systems in the context of genomic data visualization and interpretation as well as of human computing techniques. While many questions in HCI and human computation remain unexplored, proposed methods advance the field forward and present new exciting challenges to be addressed in future work.

Appendix A

Chapter 4: Supplementary Materials

A.1 Evaluation Dataset

We test our approach on the artificially created dataset, which was also used to evaluate the performance of `Colony B` players. It contains ten screens with various clustering situations: different number of clusters of varying shape, density, size and location on the mobile screen as well as simple noise dataset without clear clusters. Besides, it has been labeled by the expert. We use it as a reference dataset to evaluate the performance of a player of our `Colony B`. Also since the launch of the game, for this dataset, we accumulated over 5000 solutions per stage.

To show performance of our approach (later referenced to as "human") we use the following clustering algorithms to compare with: Agglomerative, Gaussian Mixtures models, KMeans, DBSCAN, Louvain and also explicitly designed accumulative algorithm that collects all possible solutions for different clustering algorithms (using iterative approach for different parameters) and use them as a stand-alone solution. It mimics our human-based approach with machine-based solutions. For clustering algorithms, we process cluster labels using One-vs.-Rest technique, compute modularity for each resulting binary vector and report the maximum one.

We note that DBSCAN is not shown in most report plots as it often groups the majority of data points in a single cluster and, therefore, is not useful in our comparison (we use elbow heuristics to estimate ϵ and $min\_samples$). We benchmark our

approach against clustering algorithms using four metrics: modularity and Silhouette, Dunn, SDbw cluster validation indices (Figure A.1).

For Modularity metric we are interested in comparison with Louvain approach since it is based on modularity optimization process and therefore is expected to produce maximum result. Our results illustrate that human-based approach performs on par with the Louvain algorithm (0.355 against 0.363). Besides, the drop in the measure for all approaches in our test can be explained by the presence of the stages with unclear clustering structure.

For Silhouette and Dunn metrics, the human-based approach outperforms other algorithms.

We note that SDbw metrics is reversed. That is an optimal clustering is assumed to hit the minimum value of the index. In our test, on average, the human-based approach performs similarly to the other clustering algorithms.

A.2 Voice Recognition Dataset

We propose two additional (to the main paper) hypotheses regarding the accumulated data:

1. The human-based approach performs differently (in terms of the metrics discussed in the main paper) for stages where an "optimal" cluster (defined by used metric) contains minor ($< 50\%$) or major ($> 50\%$) portion of the data points.

2. Clusterability defined in main paper is proportional to the consensus quality.

Of note, due to the noise in the dataset, we do not expect a stage to have clear and meaningful clustering structure.

We also examine submitted solutions to understand if there is any bias in the size of a player's selection. We compute the diameter of a solution (distance between furthest points in a player's selection) and compare it against the diameter of the data shown to a player (Fig. A.2). The result shows that there are three major sizes of the clusters:

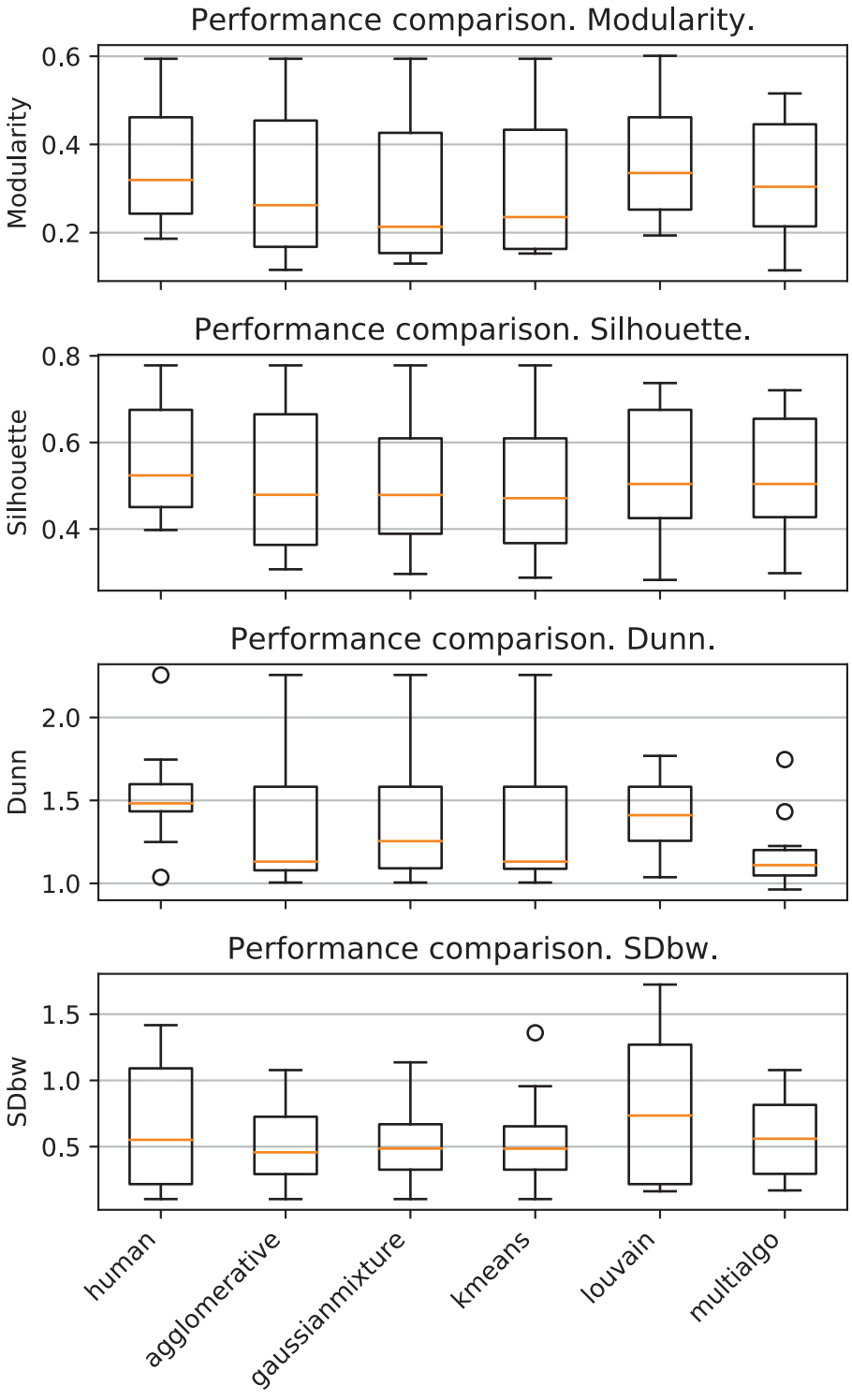

FIGURE A.1: Performance of different approaches for Evaluation Dataset based on the 4 metrics: Modularity, Silhouette, Dunn, SDbw.

i) $55-60\%$, ii) $80-85\%$, and iii) $95-100\%$ of the screen. The latter can be ignored as it signalizes that nearly full stage was selected and is not informative for our discussion. Therefore, we split our discussion into two populations of solutions: $0-60\%$ of the screen selected, and $60-90\%$ of the screen selected.

A.2.1 VRD. Population 1

In this section, we take into account only solutions that cover less than 60% of a mobile screen (population 1).

First, we apply our pipeline as well as other clustering algorithms discussed above to each stage dataset. Then, we calculate the 'optimal' cluster size for each stage for a particular metric. To do so, we find the maximum metric score among the scores found by all approaches and identify the size of the cluster using the approach that gave the maximum value. By no means, this solution must be considered as an optimal clustering partition for a stage, and it is used as a reference point only. Next, denote "small" and "large" clusters the selections that contain less than 50% and more than 50% of the points per stage, respectively. We employ this logic as a classification criterion for stages with small/large 'optimal' cluster (later referred to as SOC and LOC, respectively). We expect to see the performance of human-based approach degrade with the 'optimal' cluster being small, whereas it should match or even outperform other approaches otherwise.

Figure A.3 shows the performance comparison for both SOC (blue) and LOC (green). For modularity section, since individual groups do not follow the normal distribution (the Shapiro-Wilk test rejected the null hypothesis with $p < 0.007$), we use a Mann-Whitney's U test to compare medians of SOC and LOC groups. We primarily aim to detect a change in the score for human-based algorithms. Significance test did not reject the null hypothesis ($p = 0.2333$). This illustrates that there is no significant difference in modularity scores for the human-based approach. We observe similar results

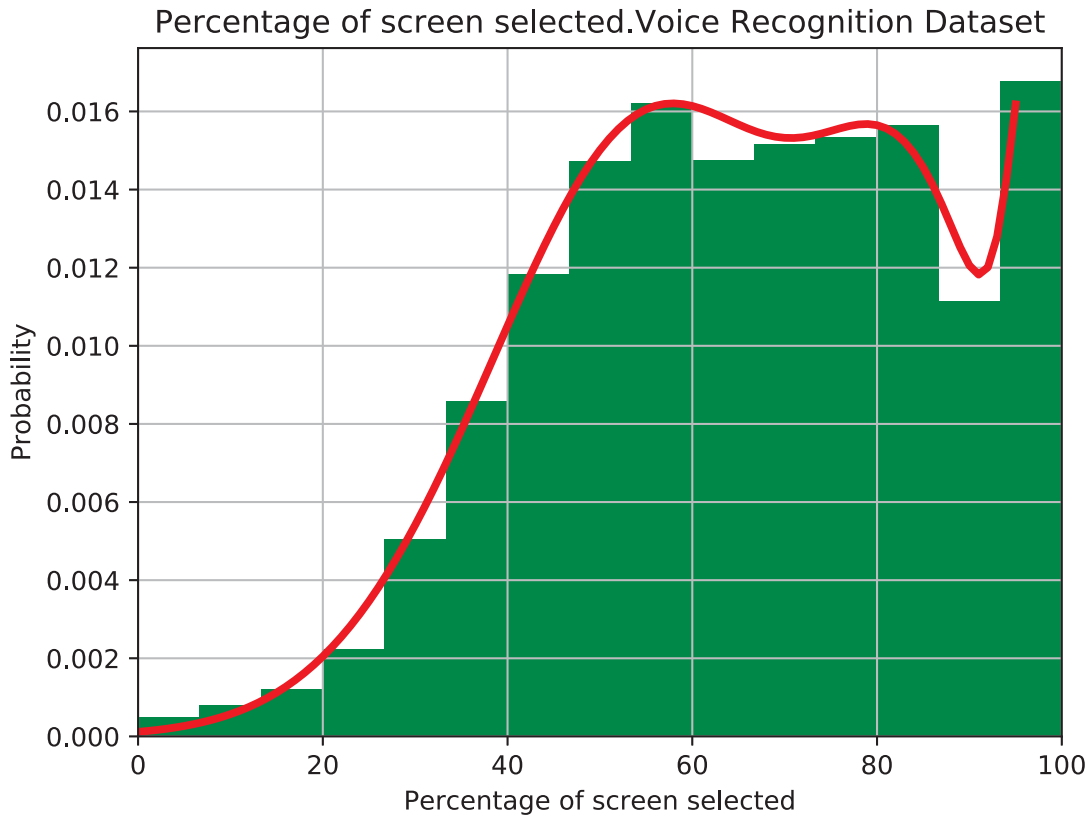

FIGURE A.2: Percentage of a mobile screen selected by the `Colony B` players for Voice Recognition Dataset (VRD).

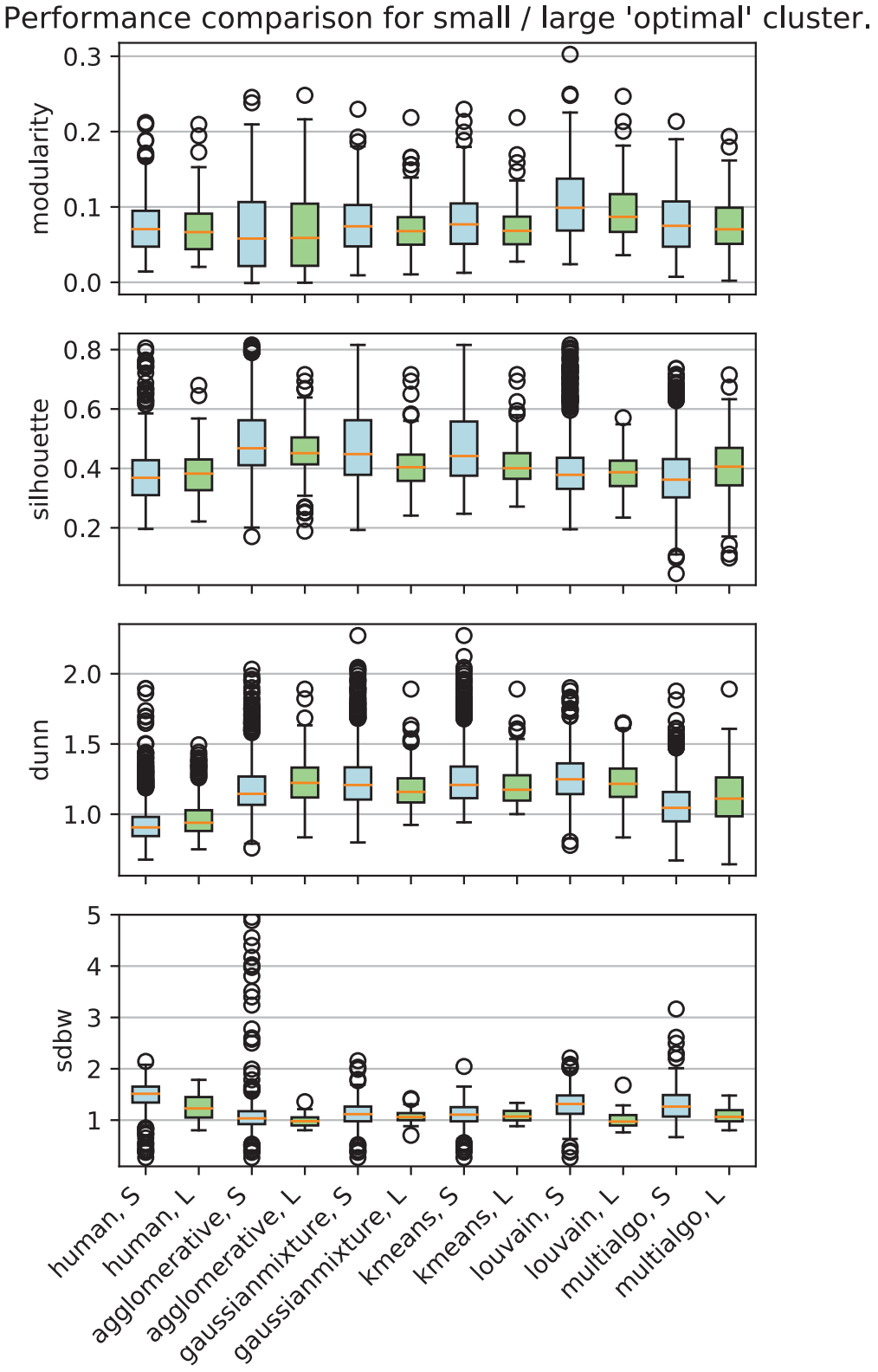

FIGURE A.3: Performance comparison for small (blue)/large (green) 'optimal' cluster for VRD (population 1) using four metrics: Modularity, Silhouette, Dunn, and SDbw.

for Silhouette metric, where Mann-Whitney's U test did not reject the null hypothesis ($p = 0.0512$).

In contrast, Dunn and SDbw related results for the human-based approach show a significant difference between SOC and LOC groups. The Mann-Whitney's U test rejects the null hypothesis with $p = 5.3758e - 08$ and $p = 0.0002$, respectively. Therefore, the values of the Dunn and SDbw indices for SOC and LOC groups support the hypothesis that the performance of the human-based approach improves for stages with larger clusters.

Comparing the human-based approach with other algorithms, for modularity metrics our approach performs on par with the other approaches (as expected, outperformed by Louvain algorithm), however for Silhouette, Dunn and SDbw metrics it performs less efficient both for SOC and LOC stages.

A.2.2 VRD. Population 2

In this section, we consider only solutions that cover $60 - 90\%$ of a mobile screen (population 2). Then, we repeat the experiment described in the previous section.

First, we study the performances of all the algorithms for both SOC and LOC groups of stages (Fig. A.4).

The result demonstrates that regardless of the metrics used, there is an improvement in the score for LOC over SOC stages. Since the scores for SOC and LOC groups observed using the human-based approach do not follow the normal distribution (the Shapiro-Wilk test rejected the null hypothesis with $p < 0.007$), we use a Mann-Whitney's U test to compare medians of SOC and LOC group scores. The test rejects the null hypothesis with $p < 0.0028$. Then, we measure the difference between the medians of SOC and LOC groups for all metrics. Our results show 11.1%, 9.9%, 13.3%, and -29.5% of growth from SOC to LOC stages for Modularity, Silhouette, Dunn, and SDbw metrics respectively. This fact supports our hypothesis that regardless of the

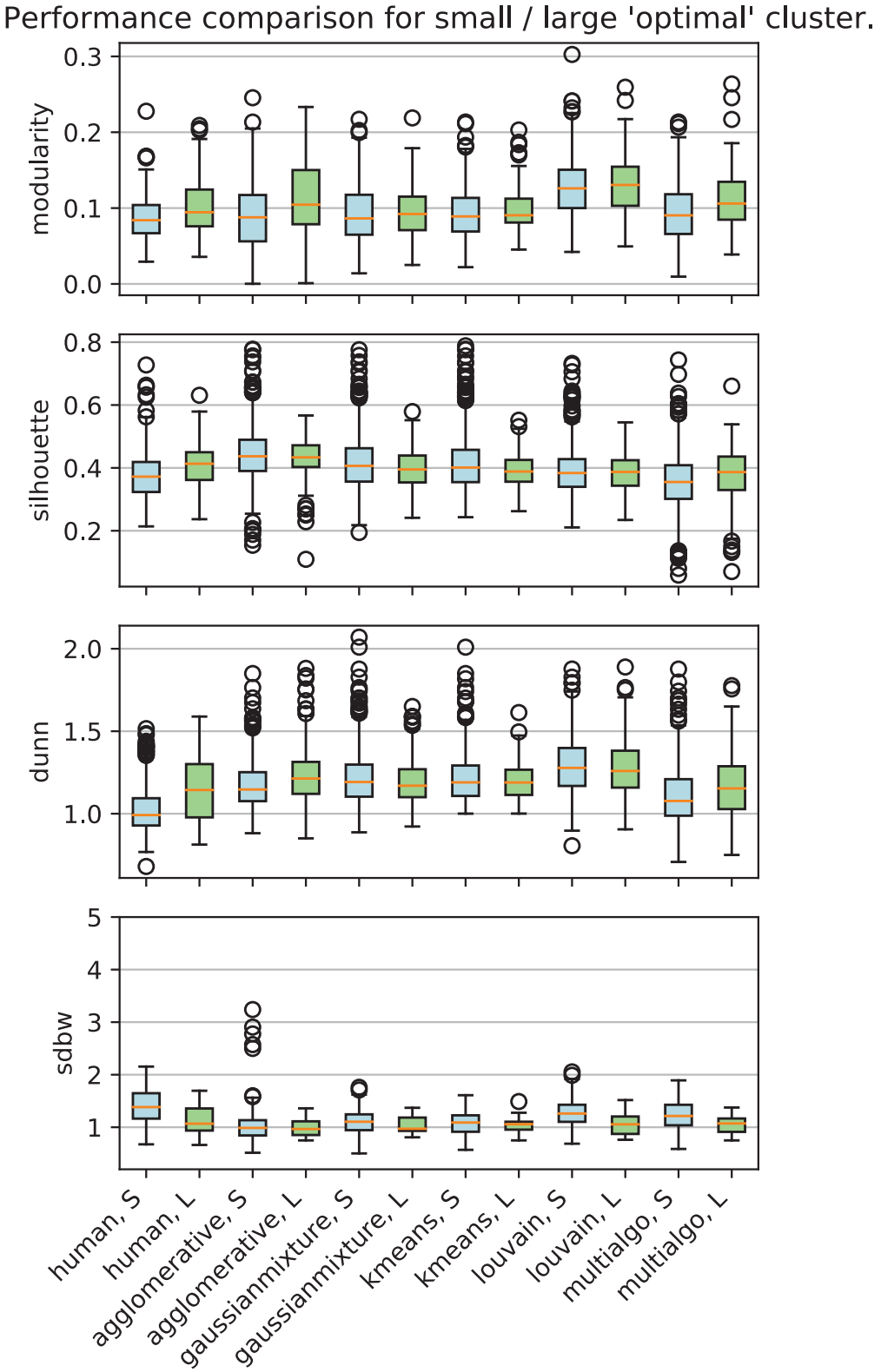

FIGURE A.4: Performance comparison for small (blue)/large (green) 'optimal' cluster for VRD (population 2) using four metrics: Modularity, Silhouette, Dunn and SDbw.

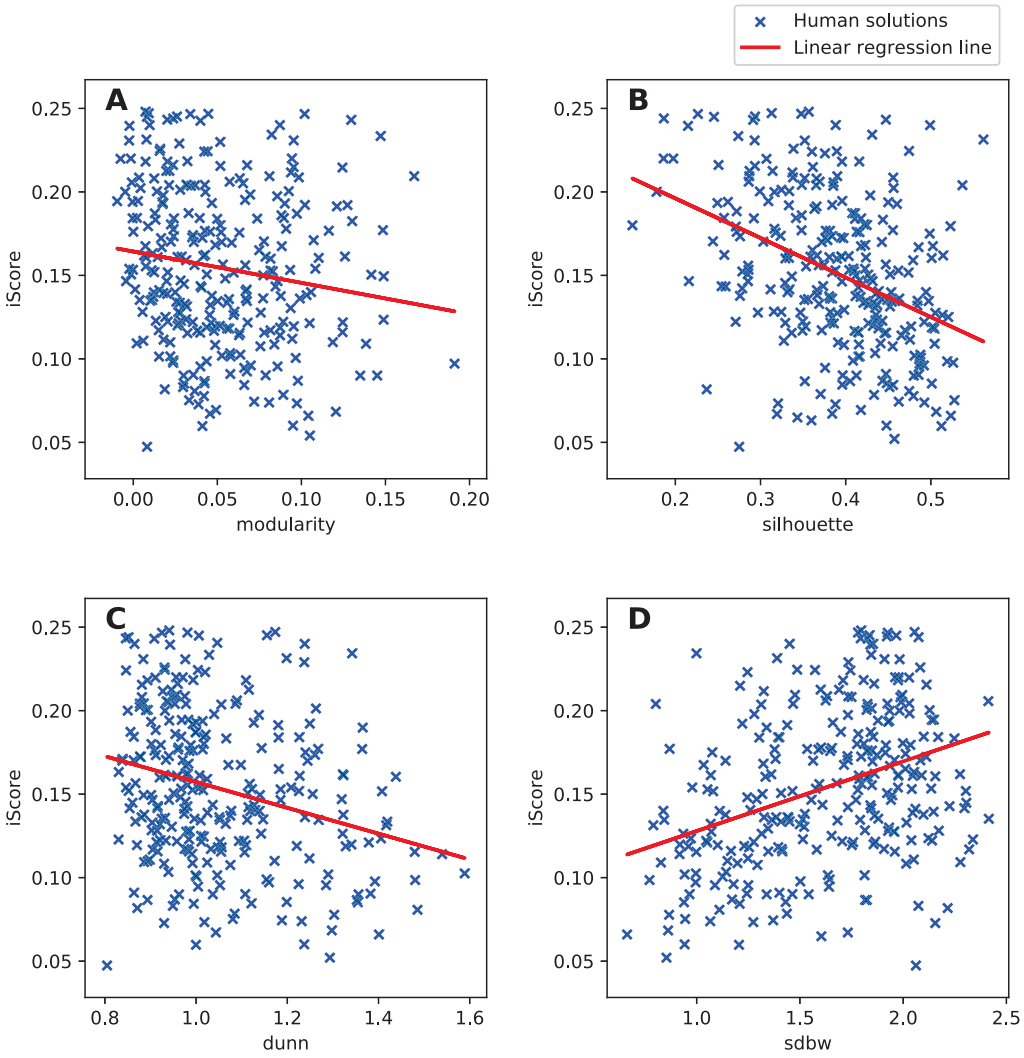

FIGURE A.5: Correlation between clusterability (iScore) and quality of consensus measured with A) Modularity, B) Silhouette, C) Dunn, D) SDbw indices. Each point represents a consensus of the `Colony B` players' solutions for a particular stage. Linear regression line shows a direction of the correlation between the measure of clusterability and a metric.

metric used, the performance of the human-based approach for LOC stages are higher than one for SOC stages.

We also compare the performance of the human-based approach with other clustering algorithms both for SOC and LOC stages (Fig. A.4). For Modularity and Silhouette metrics, the performance of the clustering algorithms mostly reduces between SOC and LOC stages while opposite for the human-based approach. Thus, for LOC stages the human-based approach performs on par with other algorithms. For Dunn and SDbw metrics, our results show that despite the lack of the improvement for clustering algorithms between SOC and LOC stages, the proposed algorithm also performs better on LOC than on SOC stages. Similar to the other metrics, the performance of the human-based algorithm on LOC stages is on the same level with other algorithms.

Besides, since population 2, in general, performs in our tests better than population 1, we choose this population to test hypothesis 2. We compare the measure of clusterability (section 4.5.2) with a quality of the consensus of accumulated human solutions.

According to the hypothesis 2 we expect clusterability to be somehow inversely proportional to Modularity, Silhouette and Dunn metrics score (clusterability is inverse measure with the best value at 0), and proportional to SDbw metrics score. It is important to mention that we do not expect a significantly strong correlation for any of the metrics due to the differences in underlying criteria of evaluation. Nevertheless, this experiment can provide us with an idea which metrics (or combination of metrics) is able to approximate human behavior.

To filter out the noise of the VRD, we utilize ground truth labels (Male/Female). We select only those stages, where the human-based approach was in the top three methods to identify true labels for a particular stage. We illustrate correlation between clusterability and each one of the metrics individually (Fig. A.5).

The results show that there is close to no correlation between the measure of clusterability and such metrics as Modularity (Spearman's rank correlation is -0.19) and

Dunn (Spearman's rank correlation is -0.26). In contrast, Silhouette and SDbw metrics better correlate with iScore (Spearman's rank correlation is -0.41 and 0.34). Also, the signs of the correlations agree with the proposed earlier relations between measures. Thus, the result indicates that there is a distinguishable relationship between the clusterability and some metrics discussed above.

References

1. Sells, S. B. & Fixott, R. S. Evaluation of research on effects of visual training on visual functions. *American journal of ophthalmology* **44,** 230–236 (1957).
2. Card, S. K. *The psychology of human-computer interaction* (CRC Press, 2018).
3. Rolls, E. T., Critchley, H. D., Mason, R. & Wakeman, E. A. Orbitofrontal cortex neurons: role in olfactory and visual association learning. *Journal of neurophysiology* **75,** 1970–1981 (1996).
4. Small, D. M. & Prescott, J. Odor/taste integration and the perception of flavor. *Experimental brain research* **166,** 345–357 (2005).
5. De Araujo, I. E., Rolls, E. T., Kringelbach, M. L., McGlone, F. & Phillips, N. Taste-olfactory convergence, and the representation of the pleasantness of flavour, in the human brain. *European Journal of Neuroscience* **18,** 2059–2068. https://onlinelibrary.wiley.com/doi/abs/10.1046/j.1460-9568.2003.02915.x (2003).
6. Eng, H. Y., Chen, D. & Jiang, Y. Visual working memory for simple and complex visual stimuli. *Psychonomic bulletin & review* **12,** 1127–1133 (2005).
7. Kimball, R. & Harslem, B. V. E. Designing the Star user interface. *Byte* **7,** 242–282 (1982).
8. Jacko, J. A. *Human computer interaction handbook: Fundamentals, evolving technologies, and emerging applications* (CRC press, 2012).
9. Preece, J. *et al. Human-Computer Interaction* ISBN: 0201627698 (Addison-Wesley Longman Ltd., Essex, UK, UK, 1994).
10. Wickens, C. D., Gordon, S. E., Liu, Y., *et al.* An introduction to human factors engineering (1998).

11. Howe, J. The rise of crowdsourcing. *Wired magazine* **14,** 1–4 (2006).

12. De Wit, E. & De Laat, W. A decade of 3C technologies: insights into nuclear organization. *Genes & development* **26,** 11–24 (2012).

13. Nielsen, C. B., Cantor, M., Dubchak, I., Gordon, D. & Wang, T. Visualizing genomes: techniques and challenges. *Nature methods* **7,** S5 (2010).

14. Von Ahn, L. & Dabbish, L. *Labeling images with a computer game* in *Proceedings of the SIGCHI conference on Human factors in computing systems* (2004), 319–326.

15. Wikipedia. *The final release of Wikistats-1 dump-based reports* Wikipedia. https://stats.wikimedia.org/EN/TablesWikipediaEN.htm.

16. Alexa. *wikipedia.org Competitive Analysis, Marketing Mix and Traffic* Alexa. An Amazon.com company. https://www.alexa.com/siteinfo/wikipedia.org.

17. Comscore. *Comscore Ranks the Top 50 U.S. Digital Media Properties for July 2015* Comscore inc. https://www.comscore.com/Insights/Rankings/comScore-Ranks-the-Top-50-US-Digital-Media-Properties-for-July-2015.

18. Bryant, S. L., Forte, A. & Bruckman, A. *Becoming Wikipedian: transformation of participation in a collaborative online encyclopedia* in *Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work* (2005), 1–10.

19. Haklay, M. & Weber, P. Openstreetmap: User-generated street maps. *IEEE Pervasive Computing* **7,** 12–18 (2008).

20. O'Connor, P. User-generated content and travel: A case study on Tripadvisor. com. *Information and communication technologies in tourism 2008,* 47–58 (2008).

21. Nasehi, S. M., Sillito, J., Maurer, F. & Burns, C. *What makes a good code example?: A study of programming Q&A in StackOverflow* in *2012 28th IEEE International Conference on Software Maintenance (ICSM)* (2012), 25–34.

22. Mason, B. & Thomas, S. A million penguins research report. *Institute of Creative Technologies, De Montfort University, Leicester, United Kingdom* (2008).

23. Buhrmester, M., Kwang, T. & Gosling, S. D. Amazon's Mechanical Turk: A new source of inexpensive, yet high-quality, data? *Perspectives on psychological science* **6,** 3–5 (2011).

24. Paolacci, G., Chandler, J. & Ipeirotis, P. G. Running experiments on amazon mechanical turk. *Judgment and Decision making* **5,** 411–419 (2010).

25. Heer, J. & Bostock, M. *Crowdsourcing graphical perception: using mechanical turk to assess visualization design* in *Proceedings of the SIGCHI conference on human factors in computing systems* (2010), 203–212.

26. Alonso, O., Rose, D. E. & Stewart, B. *Crowdsourcing for relevance evaluation* in *SIGIR forum* **42** (2008), 9–15.

27. Ott, M., Choi, Y., Cardie, C. & Hancock, J. T. *Finding deceptive opinion spam by any stretch of the imagination* in *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1* (2011), 309–319.

28. Shank, D. B. Using crowdsourcing websites for sociological research: The case of Amazon Mechanical Turk. *The American Sociologist* **47,** 47–55 (2016).

29. Snow, R., O'Connor, B., Jurafsky, D. & Ng, A. Y. *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* (2008), 254–263.

30. Marge, M., Banerjee, S. & Rudnicky, A. I. *Using the Amazon Mechanical Turk for transcription of spoken language* in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing* (2010), 5270–5273.

31. Mohammad, S. M. & Turney, P. D. *Emotions evoked by common words and phrases: Using mechanical turk to create an emotion lexicon* in *Proceedings of the NAACL HLT 2010 workshop on computational approaches to analysis and generation of emotion in text* (2010), 26–34.

32. Bonfanti, A. & Brunetti, F. Crowdcrafting as a new manufacturing model: the experience of Berto Salotti. *Sinergie* **98** (2015).

33. Gardlo, B., Ries, M., Hoßfeld, T. & Schatz, R. *Microworkers vs. facebook: The impact of crowdsourcing platform choice on experimental results* in *2012 Fourth International Workshop on Quality of Multimedia Experience* (2012), 35–36.

34. De Winter, J., Kyriakidis, M., Dodou, D. & Happee, R. Using CrowdFlower to study the relationship between self-reported violations and traffic accidents. *Procedia Manufacturing* **3,** 2518–2525 (2015).

35. UnikScripts, I. *Crowdsourcing web service RapidWorkers* RapidWorkers. http://rapidworkers.com/.

36. MiniJobz. *Crowdsourcing web service MiniJobz* MiniJobz. http://jinijobz.com/.

37. Peer, E., Samat, S., Brandimarte, L. & Acquisti, A. Beyond the Turk: An empirical comparison of alternative platforms for crowdsourcing online research. *ACR North American Advances* (2015).

38. Downs, J. S., Holbrook, M. B., Sheng, S. & Cranor, L. F. *Are Your Participants Gaming the System?: Screening Mechanical Turk Workers* in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (ACM, New York, NY, USA, 2010), 2399–2402. ISBN: 978-1-60558-929-9. http://doi.acm.org/10.1145/1753326.1753688.

39. Ross, J., Irani, L., Silberman, M, Zaldivar, A. & Tomlinson, B. *Who are the crowdworkers?: shifting demographics in mechanical turk* in *CHI'10 extended abstracts on Human factors in computing systems* (2010), 2863–2872.

40. Ipeirotis, P. G., Provost, F. & Wang, J. *Quality Management on Amazon Mechanical Turk* in *Proceedings of the ACM SIGKDD Workshop on Human Computation* (ACM, New York, NY, USA, 2010), 64–67. ISBN: 978-1-4503-0222-7. http://doi.acm.org/10.1145/1837885.1837906.

41. Ipeirotis, P. G. Analyzing the amazon mechanical turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students, Forthcoming* (2010).

42. Law, E., Dalton, C., Merrill, N., Young, A. & Gajos, K. Z. *Curio: a platform for supporting mixed-expertise crowdsourcing* in *First AAAI Conference on Human Computation and Crowdsourcing* (2013).

43. Lombrana, D. *et al. http://crowdcrafting.org/* http://crowdcrafting.org/ (2015).

44. Di Pasquale, A., McCann, R. S. & Maire, N. Assessing the population coverage of a health demographic surveillance system using satellite imagery and crowdsourcing. *PloS one* **12,** e0183661 (2017).

45. Willis, C. G. *et al.* CrowdCurio: an online crowdsourcing platform to facilitate climate change studies using herbarium specimens. *New Phytologist* **215,** 479–488 (2017).

46. Law, E. *et al. Curiosity killed the cat, but makes crowdwork better* in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (2016), 4098–4110.

47. Von Ahn, L., Blum, M., Hopper, N. J. & Langford, J. *CAPTCHA: Using hard AI problems for security* in *International Conference on the Theory and Applications of Cryptographic Techniques* (2003), 294–311.

48. Von Ahn, L., Maurer, B., McMillen, C., Abraham, D. & Blum, M. recaptcha: Human-based character recognition via web security measures. *Science* **321,** 1465–1468 (2008).

49. Von Ahn, L. *Duolingo: learn a language for free while helping to translate the web* in *Proceedings of the 2013 international conference on Intelligent user interfaces* (2013), 1–2.

50. Zook, M., Graham, M., Shelton, T. & Gorman, S. Volunteered geographic information and crowdsourcing disaster relief: a case study of the Haitian earthquake. *World Medical & Health Policy* **2,** 7–33 (2010).

51. Mizushima, K., Sugihara, T. & Ikawa, Y. *Knowledge management in a volunteer community at the time of disaster* in *2012 Proceedings of PICMET'12: Technology Management for Emerging Technologies* (2012), 2274–2282.

52. Riccardi, M. T. The power of crowdsourcing in disaster response operations. *International Journal of Disaster Risk Reduction* **20,** 123–128 (2016).

53. Lakhani, K. R., Garvin, D. A. & Lonstein, E. Topcoder (a): Developing software through crowdsourcing. *Harvard Business School General Management Unit Case* (2010).

54. Lakhani, K. R. *et al.* Prize-based contests can provide solutions to computational biology problems. *Nature biotechnology* **31,** 108 (2013).

55. Narayanan, A., Shi, E. & Rubinstein, B. I. *Link prediction by de-anonymization: How we won the kaggle social network challenge* in *The 2011 International Joint Conference on Neural Networks* (2011), 1825–1834.

56. Graham, B. Kaggle diabetic retinopathy detection competition report. *University of Warwick* (2015).

57. Taieb, S. B. & Hyndman, R. J. A gradient boosting approach to the Kaggle load forecasting competition. *International journal of forecasting* **30,** 382–394 (2014).

58. Iglovikov, V., Mushinskiy, S. & Osin, V. Satellite imagery feature detection using deep convolutional neural network: A kaggle competition. *arXiv preprint arXiv:1706.06169* (2017).

59. Brabham, D. C. Moving the crowd at iStockphoto: The composition of the crowd and motivations for participation in a crowdsourcing application. *First monday* **13** (2008).

60. Good, B. M. & Su, A. I. Crowdsourcing for bioinformatics. *Bioinformatics* **29,** 1925–1933 (2013).

61. Von Ahn, L. Games with a purpose. *Computer* **39,** 92–94 (2006).

62. Von Ahn, L., Liu, R. & Blum, M. *Peekaboom: a game for locating objects in images* in *Proceedings of the SIGCHI conference on Human Factors in computing systems* (2006), 55–64.

63. Von Ahn, L., Ginosar, S., Kedia, M. & Blum, M. *Improving image search with phetch* in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07* **4** (2007), IV–1209.

64. Law, E. L., Von Ahn, L., Dannenberg, R. B. & Crawford, M. *TagATune: A Game for Music and Sound Annotation.* in *ISMIR* **3** (2007), 2.

65. Turnbull, D., Liu, R., Barrington, L. & Lanckriet, G. R. *A Game-Based Approach for Collecting Semantic Annotations of Music.* in *ISMIR* **7** (2007), 535–538.

66. Bell, M. *et al. EyeSpy: supporting navigation through play* in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2009), 123–132.

67. Prandi, C., Salomoni, P., Roccetti, M., Nisi, V. & Nunes, N. J. *Walking with GeoZombie: A pervasive game to engage people in urban crowdsourcing* in *2016 International Conference on Computing, Networking and Communications (ICNC)* (2016), 1–5.

68. Blasco, A. *et al.* Advancing Computational Biology and Bioinformatics Research Through Open Innovation Competitions. *bioRxiv,* 565481 (2019).

69. Mavandadi, S. *et al.* Distributed medical image analysis and diagnosis through crowd-sourced games: a malaria case study. *PloS one* **7,** e37245 (2012).

70. Luengo-Oroz, M. A., Arranz, A. & Frean, J. Crowdsourcing malaria parasite quantification: an online game for analyzing images of infected thick blood smears. *Journal of medical Internet research* **14,** e167 (2012).

71. Kawrykow, A. *et al.* Phylo: a citizen science approach for improving multiple sequence alignment. *PloS one* **7,** e31362 (2012).

72. Kwak, D. *et al.* Open-Phylo: a customizable crowd-computing platform for multiple sequence alignment. *Genome biology* **14,** R116 (2013).

73. Singh, A., Ahsan, F., Blanchette, M. & Waldispühl, J. *Lessons from an online massive genomics computer game* in *Fifth AAAI Conference on Human Computation and Crowdsourcing* (2017).

74. Cooper, S. *et al.* Predicting protein structures with a multiplayer online game. *Nature* **466,** 756 (2010).

75. Khatib, F. *et al.* Crystal structure of a monomeric retroviral protease solved by protein folding game players. *Nature structural & molecular biology* **18,** 1175 (2011).

76. Kleffner, R. *et al.* Foldit Standalone: a video game-derived protein structure manipulation interface using Rosetta. *Bioinformatics* **33,** 2765–2767 (2017).

77. Lee, J. *et al.* RNA design rules from a massive open laboratory. *Proceedings of the National Academy of Sciences* **111,** 2122–2127 (2014).

78. Cireşan, D., Meier, U. & Schmidhuber, J. Multi-column deep neural networks for image classification. *arXiv preprint arXiv:1202.2745* (2012).

79. Mazumdar, A. & Saha, B. Clustering via crowdsourcing. *arXiv preprint arXiv:1604.01839* (2016).

80. Law, E. & Ahn, L. v. Human computation. *Synthesis lectures on artificial intelligence and machine learning* **5,** 29–311 (2011).

81. Gokhale, C. *et al. Corleone: hands-off crowdsourcing for entity matching* in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data* (2014), 601–612.

82. Verroios, V., Garcia-Molina, H. & Papakonstantinou, Y. *Waldo: An adaptive human interface for crowd entity resolution* in *Proceedings of the 2017 ACM International Conference on Management of Data* (2017), 1133–1148.

83. Chilton, L. B., Little, G., Edge, D., Weld, D. S. & Landay, J. A. *Cascade: Crowdsourcing taxonomy creation* in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2013), 1999–2008.

84. André, P., Kittur, A. & Dow, S. P. *Crowd synthesis: Extracting categories and clusters from complex data* in *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing* (2014), 989–998.

85. Bragg, J., Weld, D. S., *et al. Crowdsourcing multi-label classification for taxonomy creation* in *First AAAI conference on human computation and crowdsourcing* (2013).

86. Chang, J. C., Kittur, A. & Hahn, N. *Alloy: Clustering with crowds and computation* in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (2016), 3180–3191.

87. Park, H.-S. & Jun, C.-H. A simple and fast algorithm for K-medoids clustering. *Expert Systems with Applications* **36,** 3336 –3341. ISSN: 0957-4174. http://www.sciencedirect.com/science/article/pii/S095741740800081X (2009).

88. Kaufman, L. & Rousseeuw, P. J. *Clustering by means of medoids* (eds Y, I. D. & editor) Amsterdam: 1987.

89. Ng, R. T. & Han, J. CLARANS: A Method for Clustering Objects for Spatial Data Mining. *IEEE Transactions on Knowledge and Data Engineering* **14,** 1003–1016 (2002).

90. Zhang, B., Hsu, M. & Dayal, U. *K-Harmonic Means - A Spatial Clustering Algorithm with Boosting* in *Proceedings of the First International Workshop on Temporal, Spatial, and Spatio-Temporal Data Mining-Revised Papers* (Springer-Verlag, London, UK, UK, 2001), 31–45. ISBN: 3-540-41773-7. http://dl.acm.org/citation.cfm?id=645905.672960.

91. Lloyd, S. Least Squares Quantization in PCM. *IEEE Trans. Inf. Theor.* **28,** 129–137. ISSN: 0018-9448. http://dx.doi.org/10.1109/TIT.1982.1056489 (Sept. 2006).

92. Comaniciu, D. & Meer, P. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis & Machine Intelligence,* 603–619 (2002).

93. Shamos, M. I. & Hoey, D. *Closest-point problems* in *16th Annual Symposium on Foundations of Computer Science (sfcs 1975)* (1975), 151–162.

94. Sibson, R. SLINK: an optimally efficient algorithm for the single-link cluster method. *The computer journal* **16,** 30–34 (1973).

95. Defays, D. An efficient algorithm for a complete link method. *The Computer Journal* **20,** 364–366 (1977).

96. Sokal, R. R. A statistical method for evaluating systematic relationship. *University of Kansas science bulletin* **28,** 1409–1438 (1958).

97. Ward Jr, J. H. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association* **58,** 236–244 (1963).

98. Murtagh, F. & Legendre, P. Ward's hierarchical agglomerative clustering method: which algorithms implement Ward's criterion? *Journal of classification* **31,** 274–295 (2014).

99. Zhang, T., Ramakrishnan, R. & Livny, M. BIRCH: An Efficient Data Clustering Method for Very Large Databases. *SIGMOD Rec.* **25,** 103–114. ISSN: 0163-5808. http://doi.acm.org/10.1145/235968.233324 (June 1996).

100. Guha, S., Rastogi, R. & Shim, K. CURE: An Efficient Clustering Algorithm for Large Databases. *SIGMOD Rec.* **27,** 73–84. ISSN: 0163-5808. http://doi.acm.org/10.1145/276305.276312 (June 1998).

101. Ester, M., Kriegel, H.-P., Sander, J. & Xu, X. *A Density-based Algorithm for Discovering Clusters a Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise* in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (AAAI Press, Portland, Oregon, 1996), 226–231. http://dl.acm.org/citation.cfm?id=3001460.3001507.

102. Ankerst, M., Breunig, M. M., Kriegel, H.-P. & Sander, J. OPTICS: Ordering Points to Identify the Clustering Structure. *SIGMOD Rec.* **28,** 49–60. ISSN: 0163-5808. http://doi.acm.org/10.1145/304181.304187 (June 1999).

103. Karami, A. & Johansson, R. Choosing dbscan parameters automatically using differential evolution. *International Journal of Computer Applications* **91,** 1–11 (2014).

104. Gupta, M. R. & Chen, Y. Theory and Use of the EM Algorithm. *Found. Trends Signal Process.* **4,** 223–296. ISSN: 1932-8346. http://dx.doi.org/10.1561/2000000034 (Mar. 2011).

105. Rasmussen, C. E. in *Advances in Neural Information Processing Systems 12* (eds Solla, S. A., Leen, T. K. & Müller, K.) 554–560 (MIT Press, 2000). http://papers.nips.cc/paper/1745-the-infinite-gaussian-mixture-model.pdf.

106. Jin, C., Zhang, Y., Balakrishnan, S., Wainwright, M. J. & Jordan, M. I. *Local Maxima in the Likelihood of Gaussian Mixture Models: Structural Results and Algorithmic Consequences* in *NIPS* (2016).

107. Kruskal, J. B. *On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem* in *Proceedings of the American Mathematical Society, 7* (1956).

108. Prim, R. C. Shortest Connection Networks and some Generalizations. *The Bell Systems Technical Journal* **36,** 1389–1401 (1957).

109. Sharan, R. & Shamir, R. *CLICK: A Clustering Algorithm with Applications to Gene Expression Analysis* in (AAAI Press, 2000), 307–316.

110. Sozio, M. & Gionis, A. *The community-search problem and how to plan a successful cocktail party* in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* (2010), 939–948.

111. Newman, M. E. J. & Girvan, M. Finding and evaluating community structure in networks. *Physical Review* **E 69** (2004).

112. Brandes, U. *et al.* On Modularity Clustering. *IEEE Trans. on Knowl. and Data Eng.* **20,** 172–188. ISSN: 1041-4347. http://dx.doi.org/10.1109/TKDE.2007.190689 (Feb. 2008).

113. Brandes, U. *et al. Maximizing Modularity is hard* 2006. http://arxiv.org/abs/physics/0608255.

114. Blondel, V. D., Guillaume, J.-L., Lambiotte, R. & Lefebvre, E. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* **2008.** http://stacks.iop.org/1742-5468/2008/i=10/a=P10008 (2008).

115. Clauset, A., Newman, M. E. J. & Moore, C. Finding community structure in very large networks. *Phys. Rev. E* **70,** 066111. https://link.aps.org/doi/10.1103/PhysRevE.70.066111 (6 Dec. 2004).

116. Pons, P. & Latapy, M. *Computing Communities in Large Networks Using Random Walks* in *Computer and Information Sciences - ISCIS 2005* (eds Yolum, p., Güngör, T., Gürgen, F. & Özturan, C.) (Springer Berlin Heidelberg, Berlin, Heidelberg, 2005), 284–293. ISBN: 978-3-540-32085-2.

117. Wakita, K. & Tsurumi, T. *Finding community structure in mega-scale social networks* in *Proceedings of the 16th international conference on World Wide Web* (2007), 1275–1276.

118. Rousseeuw, P. Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis. *J. Comput. Appl. Math.* **20,** 53–65. ISSN: 0377-0427. http://dx.doi.org/10.1016/0377-0427(87)90125-7 (Nov. 1987).

119. Dunn, J. C. A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. *Journal of Cybernetics* **3,** 32–57. https://doi.org/10.1080/01969727308546046 (1973).

120. Halkidi, M. & Vazirgiannis, M. *Clustering Validity Assessment: Finding the Optimal Partitioning of a Data Set* in *Proceedings of the 2001 IEEE International Conference on Data Mining* (IEEE Computer Society, Washington, DC, USA, 2001), 187–194. ISBN: 0-7695-1119-8. http://dl.acm.org/citation.cfm?id=645496.657864.

121. Desgraupes, B. Clustering indices. *University of Paris Ouest-Lab Modal'X* **1,** 34 (2013).

122. Steinbach, M., Ertöz, L. & Kumar, V. in *New directions in statistical physics* 273–309 (Springer, 2004).

123. Bellman, R. Dynamic programming. *Science* **153,** 34–37 (1966).

124. Fodor, I. K. *A survey of dimension reduction techniques* tech. rep. (Lawrence Livermore National Lab., CA (US), 2002).

125. Miao, J. & Niu, L. A survey on feature selection. *Procedia Computer Science* **91,** 919–926 (2016).

126. Kelkar, B. A. & Rodd, S. F. in *Data Management, Analytics and Innovation* 209–220 (Springer, 2019).

127. Liu, H. & Motoda, H. *Feature selection for knowledge discovery and data mining* (Springer Science & Business Media, 2012).

128. Guyon, I. & Elisseeff, A. An introduction to variable and feature selection. *Journal of machine learning research* **3,** 1157–1182 (2003).

129. Almuallim, H. & Dietterich, T. G. *Learning With Many Irrelevant Features.* in *AAAI* **91** (1991), 547–552.

130. Kohavi, R. & John, G. H. Wrappers for feature subset selection. *Artificial intelligence* **97,** 273–324 (1997).

131. Choi, J. Y., Ro, Y. M. & Plataniotis, K. N. Boosting color feature selection for color face recognition. *IEEE transactions on image processing* **20,** 1425–1434 (2011).

132. Zhao, Z. & Liu, H. *Spectral feature selection for supervised and unsupervised learning* in *Proceedings of the 24th international conference on Machine learning* (2007), 1151–1157.

133. Goltsev, A. & Gritsenko, V. Investigation of efficient features for image recognition by neural networks. *Neural Networks* **28,** 15–23 (2012).

134. Cai, D., Zhang, C. & He, X. *Unsupervised feature selection for multi-cluster data* in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* (2010), 333–342.

135. Dy, J. G. & Brodley, C. E. Feature selection for unsupervised learning. *Journal of machine learning research* **5,** 845–889 (2004).

136. Kim, Y., Street, W. N. & Menczer, F. *Feature selection in unsupervised learning via evolutionary search* in *KDD* (2000), 365–369.

137. Abdi, H. & Williams, L. J. Principal Component Analysis. *WIREs Comput. Stat.* **2,** 433–459. ISSN: 1939-5108. https://doi.org/10.1002/wics.101 (July 2010).

138. *Principles of Multivariate Analysis: A User's Perspective* (ed Krzanowski, W. J.) ISBN: 0-198-52211-8 (Oxford University Press, Inc., New York, NY, USA, 1988).

139. Balasubramanian, M. & Schwartz, E. L. The isomap algorithm and topological stability. *Science* **295,** 7–7 (2002).

140. Maaten, L. v. d. & Hinton, G. Visualizing data using t-SNE. *Journal of machine learning research* **9,** 2579–2605 (2008).

141. Wang, Y., Yao, H. & Zhao, S. Auto-encoder based dimensionality reduction. *Neurocomputing* **184,** 232–242 (2016).

142. Hartigan, J. A. Direct clustering of a data matrix. *Journal of the american statistical association* **67,** 123–129 (1972).

143. Pontes, B., Giráldez, R. & Aguilar-Ruiz, J. S. Biclustering on expression data: A review. *Journal of biomedical informatics* **57,** 163–180 (2015).

144. Chi, E. C., Allen, G. I. & Baraniuk, R. G. Convex biclustering. *Biometrics* **73,** 10–19 (2017).

145. Bozdağ, D., Parvin, J. D. & Catalyurek, U. V. *A Biclustering Method to Discover Co-regulated Genes Using Diverse Gene Expression Datasets* in *Bioinformatics and Computational Biology* (ed Rajasekaran, S.) (Springer Berlin Heidelberg, Berlin, Heidelberg, 2009), 151–163. ISBN: 978-3-642-00727-9.

146. Alakwaa, F. M., Solouma, N. H. & Kadah, Y. M. Construction of gene regulatory networks using biclustering and bayesian networks. *Theoretical Biology and Medical Modelling* **8,** 39. ISSN: 1742-4682. https://doi.org/10.1186/1742-4682-8-39 (Oct. 2011).

147. Aggarwal, C. C., Wolf, J. L., Yu, P. S., Procopiuc, C. & Park, J. S. Fast Algorithms for Projected Clustering. *SIGMOD Rec.* **28,** 61–72. ISSN: 0163-5808. http://doi.acm.org/10.1145/304181.304188 (June 1999).

148. Parsons, L., Haque, E. & Liu, H. Subspace clustering for high dimensional data: a review. *Acm Sigkdd Explorations Newsletter* **6,** 90–105 (2004).

149. Agrawal, R., Gehrke, J., Gunopulos, D. & Raghavan, P. Automatic subspace clustering of high dimensional data. *Data Mining and Knowledge Discovery* **11,** 5–33 (2005).

150. Cheng, C.-H., Fu, A. W. & Zhang, Y. *Entropy-based subspace clustering for mining numerical data* in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining* (1999), 84–93.

151. Goil, S., Nagesh, H. & Choudhary, A. *MAFIA: Efficient and Scalable Subspace Clustering for Very Large Data Sets* tech. rep. (1999).

152. Chang, J.-W. & Jin, D.-S. *A New Cell-based Clustering Method for Large, High-dimensional Data in Data Mining Applications* in *Proceedings of the 2002 ACM Symposium on Applied Computing* (ACM, New York, NY, USA, 2002), 503–507. ISBN: 1-58113-445-2. http://doi.acm.org/10.1145/508791.508886.

153. Aggarwal, C. C. & Yu, P. S. Finding Generalized Projected Clusters in High Dimensional Spaces. *SIGMOD Rec.* **29,** 70–81. ISSN: 0163-5808. http://doi.acm.org/10.1145/335191.335383 (May 2000).

154. Lander, E. S. *et al.* Initial sequencing and analysis of the human genome. *Nature* **409,** 860–922 (2001).

155. Venter, J. C. *et al.* The sequence of the human genome. *science* **291,** 1304–1351 (2001).

156. Labrinidis, A. & Jagadish, H. V. Challenges and Opportunities with Big Data. *Proc. VLDB Endow.* **5,** 2032–2033. ISSN: 2150-8097. http://dx.doi.org/10.14778/2367502.2367572 (Aug. 2012).

157. Chen, C. P. & Zhang, C.-Y. Data-intensive applications, challenges, techniques and technologies: A survey on Big Data. *Information Sciences* **275,** 314 –347. ISSN: 0020-0255. http://www.sciencedirect.com/science/article/pii/S0020025514000346 (2014).

158. Yardımcı, G. G. & Noble, W. S. Software tools for visualizing Hi-C data. *Genome biology* **18,** 26 (2017).

159. Xu, Z. *et al.* HiView: an integrative genome browser to leverage Hi-C results for the interpretation of GWAS variants. *BMC research notes* **9,** 159 (2016).

160. Martin, J. S. *et al.* HUGIn: Hi-C unifying genomic interrogator. *Bioinformatics* **33,** 3793–3795 (2017).

161. Lajoie, B. R., van Berkum, N. L., Sanyal, A. & Dekker, J. My5C: web tools for chromosome conformation capture studies. *Nature methods* **6,** 690 (2009).

162. Lieberman-Aiden, E. *et al.* Comprehensive mapping of long-range interactions reveals folding principles of the human genome. *science* **326,** 289–293 (2009).

163. Durand, N. C. *et al.* Juicebox provides a visualization system for Hi-C contact maps with unlimited zoom. *Cell systems* **3,** 99–101 (2016).

164. Durand, N. C. *et al.* Juicer provides a one-click system for analyzing loop-resolution Hi-C experiments. *Cell systems* **3,** 95–98 (2016).

165. Robinson, J. T. *et al.* Juicebox. js provides a cloud-based visualization system for Hi-C data. *Cell systems* **6,** 256–258 (2018).

166. Zhou, X. *et al.* Exploring long-range genome interactions using the WashU Epigenome Browser. *Nature methods* **10,** 375 (2013).

167. Bernstein, B. E. *et al.* The NIH roadmap epigenomics mapping consortium. *Nature biotechnology* **28,** 1045 (2010).

168. Kerpedjiev, P *et al. HiGlass: Web-based visual comparison and exploration of genome interaction maps. bioRxiv 121889* 2017.

169. Lekschas, F., Bach, B., Kerpedjiev, P., Gehlenborg, N. & Pfister, H. HiPiler: visual exploration of large genome interaction matrices with interactive small multiples. *IEEE transactions on visualization and computer graphics* **24,** 522–531 (2018).

170. Paulsen, J. *et al.* HiBrowse: multi-purpose statistical analysis of genome-wide chromatin 3D organization. *Bioinformatics* **30,** 1620–1622 (2014).

171. Kumar, R., Sobhy, H., Stenberg, P. & Lizana, L. Genome contact map explorer: a platform for the comparison, interactive visualization and analysis of genome contact maps. *Nucleic acids research* **45,** e152–e152 (2017).

172. Wang, Y. *et al.* The 3D Genome Browser: a web-based browser for visualizing 3D genome organization and long-range chromatin interactions. *Genome biology* **19,** 151 (2018).

173. Ramírez, F. *et al.* High-resolution TADs reveal DNA sequences underlying genome organization in flies. *Nature communications* **9,** 189 (2018).

174. Yang, D. *et al.* 3DIV: A 3D-genome Interaction Viewer and database. *Nucleic acids research* **46,** D52–D57 (2017).

175. Schofield, E. *et al.* CHiCP: a web-based tool for the integrative and interactive visualization of promoter capture Hi-C datasets. *Bioinformatics* **32,** 2511–2513 (2016).

176. Krzywinski, M. I. *et al.* Circos: an information aesthetic for comparative genomics. *Genome research* (2009).

177. An, J. *et al.* J-Circos: an interactive Circos plotter. *Bioinformatics* **31,** 1463–1465 (2014).

178. Laird, M. R., Langille, M. G. & Brinkman, F. S. GenomeD3Plot: a library for rich, interactive visualizations of genomic data in web applications. *Bioinformatics* **31,** 3348–3349 (2015).

179. Taberlay, P. C. *et al.* Three-dimensional disorganisation of the cancer genome occurs coincident with long range genetic and epigenetic alterations. *Genome research,* gr–201517 (2016).

180. Pettersen, E. F. *et al.* UCSF Chimera—a visualization system for exploratory research and analysis. *Journal of computational chemistry* **25,** 1605–1612 (2004).

181. Li, R., Liu, Y., Li, T. & Li, C. 3Disease Browser: a web server for integrating 3D genome and disease-associated chromosome rearrangement data. *Scientific reports* **6,** 34651 (2016).

182. Lesne, A., Riposo, J., Roger, P., Cournac, A. & Mozziconacci, J. 3D genome reconstruction from chromosomal contacts. *Nature methods* **11,** 1141 (2014).

183. Serra, F. *et al.* Automatic analysis and 3D-modelling of Hi-C data using TADbit reveals structural features of the fly chromatin colors. *PLoS computational biology* **13,** e1005665 (2017).

184. Asbury, T. M., Mitman, M., Tang, J. & Zheng, W. J. Genome3D: a viewer-model framework for integrating and visualizing multi-scale epigenomic information within a three-dimensional genome. *BMC bioinformatics* **11,** 444 (2010).

185. Kent, W. J. *et al.* The human genome browser at UCSC. *Genome research* **12,** 996–1006 (2002).

186. Nowotny, J. *et al.* GMOL: an interactive tool for 3D genome structure visualization. *Scientific reports* **6,** 20802 (2016).

187. Butyaev, A., Mavlyutov, R., Blanchette, M., Cudré-Mauroux, P. & Waldispühl, J. A low-latency, big database system and browser for storage, querying and visualization of 3D genomic data. *Nucleic acids research* **43,** e103–e103 (2015).

188. Pavlopoulos, G. A. *et al.* Visualizing genome and systems biology: technologies, tools, implementation techniques and trends, past, present and future. *Gigascience* **4,** 38 (2015).

189. Crick, F. H. The origin of the genetic code. *Journal of molecular biology* **38,** 367–379 (1968).

190. Mercer, T. R. *et al.* DNase I–hypersensitive exons colocalize with promoters and distal regulatory elements. *Nature genetics* **45,** 852 (2013).

191. Trieu, T. & Cheng, J. Large-scale reconstruction of 3D structures of human chromosomes from chromosomal contact data. *Nucleic acids research* **42,** e52–e52 (2014).

192. Duan, Z. *et al.* A three-dimensional model of the yeast genome. *Nature* **465,** 363 (2010).

193. Hubbard, T. *et al.* The Ensembl genome database project. *Nucleic acids research* **30,** 38–41 (2002).

194. Karolchik, D. *et al.* The UCSC genome browser database: 2014 update. *Nucleic acids research* **42,** D764–D770 (2013).

195. Flicek, P. *et al.* Ensembl 2014. *Nucleic acids research* **42,** D749–D755 (2013).

196. Guttman, A. R-trees: A Dynamic Index Structure for Spatial Searching. *SIGMOD Rec.* **14,** 47–57. ISSN: 0163-5808. http://doi.acm.org/10.1145/971697.602266 (June 1984).

197. Pfoser, D., Jensen, C. S. & Theodoridis, Y. *Novel Approaches to the Indexing of Moving Object Trajectories* in (2000), 395–406.

198. Song, Z. & Roussopoulos, N. *SEB-tree: An Approach to Index Continuously Moving Objects* in *Mobile Data Management* (eds Chen, M.-S., Chrysanthis, P. K., Sloman, M. & Zaslavsky, A.) (Springer Berlin Heidelberg, Berlin, Heidelberg, 2003), 340–344. ISBN: 978-3-540-36389-7.

199. Rasetic, S., Sander, J., Elding, J. & Nascimento, M. A. *A Trajectory Splitting Model for Efficient Spatio-temporal Indexing* in *Proceedings of the 31st International Conference on Very Large Data Bases* (VLDB Endowment, Trondheim, Norway, 2005), 934–945. ISBN: 1-59593-154-6. http://dl.acm.org/citation.cfm?id=1083592.1083700.

200. Chakka, V. P., Everspaugh, A., Patel, J. M., *et al. Indexing large trajectory data sets with SETI.* Citeseer, 2003.

201. Botea, V., Mallett, D., Nascimento, M. A. & Sander, J. PIST: An Efficient and Practical Indexing Technique for Historical Spatio-Temporal Point Data. *Geoinformatica* **12,** 143–168. ISSN: 1384-6175. http://dx.doi.org/10.1007/s10707-007-0030-3 (June 2008).

202. Patel, J. M., Chen, Y. & Chakka, V. P. *STRIPES: An Efficient Index for Predicted Trajectories* in *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data* (ACM, Paris, France, 2004), 635–646. ISBN: 1-58113-859-8. http://doi.acm.org/10.1145/1007568.1007639.

203. Porkaew, K., Lazaridis, I. & Mehrotra, S. *Querying Mobile Objects in Spatio-Temporal Databases* in *Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases* (Springer-Verlag, London, UK, UK, 2001), 59–78. ISBN: 3-540-42301-X. http://dl.acm.org/citation.cfm?id=647227.719101.

204. Altshuler, D *et al.* A 785 map of human genome variation from population-scale sequencing. *Nature* **786,** 467 (2010).

205. Eldawy, A. & Mokbel, M. F. A Demonstration of SpatialHadoop: An Efficient Mapreduce Framework for Spatial Data. *Proc. VLDB Endow.* **6,** 1230–1233. ISSN: 2150-8097. http://dx.doi.org/10.14778/2536274.2536283 (Aug. 2013).

206. The ENCODE (ENCyclopedia Of DNA Elements) Project. *Science* **306,** 636–640. ISSN: 0036-8075. http://science.sciencemag.org/content/306/5696/636 (2004).

207. Sayers, E. W. *et al.* Database resources of the National Center for Biotechnology Information. *Nucleic Acids Research* **39,** D38–D51. http://dx.doi.org/10.1093/nar/gkq1172 (2011).

208. Medina, I. *et al.* Genome Maps, a new generation genome browser. *Nucleic Acids Research* **41,** W41–W46. http://dx.doi.org/10.1093/nar/gkt530 (2013).

209. Haeussler, M. *et al.* Navigating protected genomics data with UCSC Genome Browser in a Box. *Bioinformatics* **31,** 764–766. http://dx.doi.org/10.1093/bioinformatics/btu712 (2015).

210. De Jong, M. C. *et al.* Trilateral retinoblastoma: a systematic review and meta-analysis. *The Lancet Oncology* **15,** 1157 –1167. ISSN: 1470-2045. http://www.sciencedirect.com/science/article/pii/S1470204514703365 (2014).

211. Macchi, M. M. & Bruce, J. N. Human pineal physiology and functional significance of melatonin. *Frontiers in Neuroendocrinology* **25,** 177 –195. ISSN: 0091-3022. http://www.sciencedirect.com/science/article/pii/S0091302204000196 (2004).

212. Fayyad, U., Piatetsky-Shapiro, G. & Smyth, P. The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM* **39,** 27–34 (1996).

213. Jain, A. K., Murty, M. N. & Flynn, P. J. Data clustering: a review. *ACM computing surveys (CSUR)* **31,** 264–323 (1999).

214. Gaber, M. M., Zaslavsky, A. & Krishnaswamy, S. Mining data streams: a review. *ACM Sigmod Record* **34,** 18–26 (2005).

215. Marx, V. *Biology: The big challenges of big data* 2013.

216. Forcato, M. *et al.* Comparison of computational methods for Hi-C data analysis. *Nature methods* **14,** 679 (2017).

217. Grier, D. A. The math tables project of the work projects administration: The reluctant start of the computing era. *IEEE Annals of the History of Computing* **20,** 33–50 (1998).

218. Mao, A. *et al. Volunteering versus work for pay: Incentives and tradeoffs in crowdsourcing* in *First AAAI conference on human computation and crowdsourcing* (2013).

219. Solomon, M. Groupthink versus the wisdom of crowds: The social epistemology of deliberation and dissent. *The Southern Journal of Philosophy* **44,** 28–42 (2006).

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

221. Paolacci, G., Chandler, J. & Ipeirotis, P. G. Running Experiments on Amazon Mechanical Turk. *Judgment and Decision Making* **5,** 411–419 (2010).

222. Lorenz, J., Rauhut, H., Schweitzer, F. & Helbing, D. How social influence can undermine the wisdom of crowd effect. *Proc Natl Acad Sci U S A* **108,** 9020–5 (2011).

223. Yu, L. & Nickerson, J. V. *Cooks or cobblers?: crowd creativity through combination* in *Proceedings of the International Conference on Human Factors in Computing Systems, CHI 2011, Vancouver, BC, Canada, May 7-12, 2011* (2011), 1393–1402. http://doi.acm.org/10.1145/1978942.1979147.

224. Little, G. *Exploring iterative and parallel human computation processes* in *Proceedings of the 28th International Conference on Human Factors in Computing Systems, CHI 2010, Extended Abstracts Volume, Atlanta, Georgia, USA, April 10-15, 2010* (2010), 4309–4314. http://doi.acm.org/10.1145/1753846.1754145.

225. Kittur, A. & Kraut, R. E. *Harnessing the wisdom of crowds in wikipedia: quality through coordination* in *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work, CSCW 2008, San Diego, CA, USA, November 8-12, 2008* (2008), 37–46. http://doi.acm.org/10.1145/1460563.1460572.

226. Dow, S., Kulkarni, A. P., Klemmer, S. R. & Hartmann, B. *Shepherding the crowd yields better work* in *CSCW '12 Computer Supported Cooperative Work, Seattle, WA, USA, February 11-15, 2012* (2012), 1013–1022. http://doi.acm.org/10.1145/2145204.2145355.

227. Zhang, H. *et al. Human computation tasks with global constraints* in *CHI Conference on Human Factors in Computing Systems, CHI '12, Austin, TX, USA - May 05 - 10, 2012* (2012), 217–226. http://doi.acm.org/10.1145/2207676.2207708.

228. Kearns, M., Suri, S. & Montfort, N. An experimental study of the coloring problem on human subject networks. *Science* **313,** 824–7 (2006).

229. Kearns, M. Experiments in social computation. *Commun. ACM* **55,** 56–67. http://doi.acm.org/10.1145/2347736.2347753 (2012).

230. Von Ahn, L. & Dabbish, L. *Labeling images with a computer game* in *Proceedings of the 2004 Conference on Human Factors in Computing Systems, CHI 2004, Vienna, Austria, April 24 - 29, 2004* (2004), 319–326. http://doi.acm.org/10.1145/985692.985733.

231. Ho, C., Chang, T. & Hsu, J. Y. *PhotoSlap: A Multi-player Online Game for Semantic Annotation* in *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada* (2007), 1359–1364. http://www.aaai.org/Library/AAAI/2007/aaai07-215.php.

232. Law, E. & von Ahn, L. *Input-agreement: A New Mechanism for Collecting Data Using Human Computation Games* in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (ACM, Boston, MA, USA, 2009), 1197–1206. ISBN: 978-1-60558-246-7. http://doi.acm.org/10.1145/1518701.1518881.

233. Deng, J. *et al. Imagenet: A large-scale hierarchical image database* in *2009 IEEE conference on computer vision and pattern recognition* (2009), 248–255.

234. Goodfellow, I. J. *et al. Challenges in representation learning: A report on three machine learning contests* in *International Conference on Neural Information Processing* (2013), 117–124.

235. Hamel, P., Bengio, Y. & Eck, D. Building musically-relevant audio features through multiple timescale representations (2012).

236. Chui, M., Manyika, J. & Miremadi, M. Where machines could replace humans—and where they can't (yet). *McKinsey Quarterly* **30,** 1–9 (2016).

237. Obermeyer, Z. & Emanuel, E. J. Predicting the future—big data, machine learning, and clinical medicine. *The New England journal of medicine* **375,** 1216 (2016).

238. Parent, G. & Eskenazi, M. *Clustering dictionary definitions using amazon mechanical turk* in *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk* (2010), 21–29.

239. Gomes, R. G., Welinder, P., Krause, A. & Perona, P. *Crowdclustering* in *Advances in neural information processing systems* (2011), 558–566.

240. Yi, J., Jin, R., Jain, S., Yang, T. & Jain, A. K. *Semi-crowdsourced clustering: Generalizing crowd labeling by robust distance metric learning* in *Advances in neural information processing systems* (2012), 1772–1780.

241. Kleiman, Y., Goldberg, G., Amsterdamer, Y. & Cohen-Or, D. Toward semantic image similarity from crowdsourced clustering. *The Visual Computer* **32,** 1045–1055 (2016).

242. Ogle, K. N. Researches in binocular vision. (1950).

243. Smith, A. T. Binocular vision: joining up the eyes. *Current Biology* **25,** R661–R663 (2015).

244. Tatu, A., Bak, P., Bertini, E., Keim, D. & Schneidewind, J. *Visual quality metrics and human perception: an initial study on 2D projections of large multidimensional data* in *Proceedings of the International Conference on Advanced Visual Interfaces* (2010), 49–56.

245. Rendón, E. *et al. A Comparison of Internal and External Cluster Validation Indexes* in *Proceedings of the 2011 American Conference on Applied Mathematics and the 5th WSEAS International Conference on Computer Engineering and Applications* (World Scientific, Engineering Academy, and Society (WSEAS), Puerto Morelos, Mexico, 2011), 158–163. ISBN: 978-960-474-270-7. http://dl.acm.org/citation.cfm?id=1959666.1959695.

246. Newman, M. E. J. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences* **103,** 8577–8582. ISSN: 0027-8424. http://www.pnas.org/content/103/23/8577 (2006).

247. Estivill-Castro, V. Why So Many Clustering Algorithms: A Position Paper. *SIGKDD Explor. Newsl.* **4,** 65–75. ISSN: 1931-0145. http://doi.acm.org/10.1145/568574.568575 (June 2002).

248. J. KETCHEN, D. & L. SHOOK, C. The Application of Cluster Analysis in Strategic Management Research: An Analysis and Critique. *Strategic Management Journal* **17,** 441 –458 (June 1996).

249. Tibshirani, R., Walther, G. & Hastie, T. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **63,** 411–423 (2001).

250. Chen, J. Flow in games (and everything else). *Communications of the ACM* **50,** 31–34 (2007).

251. Baaden, M. *et al.* Ten simple rules to create a serious game, illustrated with examples from structural biology. *PLoS Computational Biology* **14.** https://doi.org/10.1371/journal.pcbi.1005955 (2018).

252. Turlach, B. A. *Bandwidth Selection in Kernel Density Estimation: A Review* in *CORE and Institut de Statistique* **19** (1993), 1–33.

253. Hoober, S. *How Do Users Really Hold Mobile Devices?* UXmatters. https://www.uxmatters.com/mt/archives/2013/02/how-do-users-really-hold-mobile-devices.php.

254. Rifkin, R. & Klautau, A. In Defense of One-Vs-All Classification. *J. Mach. Learn. Res.* **5,** 101–141. ISSN: 1532-4435. http://dl.acm.org/citation.cfm?id=1005332.1005336 (Dec. 2004).

255. Breunig, M. M., Kriegel, H.-P., Ng, R. T. & Sander, J. LOF: Identifying Density-based Local Outliers. *SIGMOD Rec.* **29,** 93–104. ISSN: 0163-5808. http://doi.acm.org/10.1145/335191.335388 (May 2000).

256. Chang, Y.-W., Hsieh, C.-J., Chang, K.-W., Ringgaard, M. & Lin, C.-J. Training and Testing Low-degree Polynomial Data Mappings via Linear SVM. *J. Mach. Learn. Res.* **11,** 1471–1490. ISSN: 1532-4435. http://dl.acm.org/citation.cfm?id=1756006.1859899 (Aug. 2010).

257. Ackerman, M. & Ben-David, S. *Clusterability: A Theoretical Study* in *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, AISTATS 2009, Clearwater Beach, Florida, USA, April 16-18, 2009* (2009), 1–8. http://www.jmlr.org/proceedings/papers/v5/ackerman09a.html.

258. Kale, R. *Kaggle Voice Recognition* Kaggle. https://www.kaggle.com/rohankale/voice-recognition.

259. Wold, S., Esbensen, K. & Geladi, P. Principal component analysis. *Chemometrics and intelligent laboratory systems* **2,** 37–52 (1987).

260. Bellman, R. & Kalaba, R. Dynamic programming and statistical communication theory. *Proceedings of the National Academy of Sciences of the United States of America* **43,** 749 (1957).

261. Chang, J. C., Amershi, S. & Kamar, E. *Revolt: Collaborative crowdsourcing for labeling machine learning datasets* in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (2017), 2334–2346.

262. McDonald, D. *et al.* American Gut: an Open Platform for Citizen Science Microbiome Research. *mSystems* **3** (eds Greene, C. S. *et al.*) https://msystems.asm.org/content/3/3/e00031-18 (2018).

263. Zaidan, O. F. & Callison-Burch, C. *Crowdsourcing translation: Professional quality from non-professionals* in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1* (2011), 1220–1229.

264. See, L. *et al.* Comparing the quality of crowdsourced data contributed by expert and non-experts. *PloS one* **8,** e69958 (2013).

265. Butyaev, A., Drogaris, C., Nazarova, E., Tremblay-Savard, O. & Waldispühl, J. *How do Humans Perceive 2D Clusters? Lessons From a Mobile Crowdsourcing Human-Computing Game* 2019.

266. Rodriguez, M. Z. *et al.* Clustering algorithms: A comparative approach. *PloS one* **14,** e0210236 (2019).

267. Rissanen, J. *Stochastic Complexity in Statistical Inquiry Theory* ISBN: 9971508591 (World Scientific Publishing Co., Inc., River Edge, NJ, USA, 1989).

268. Bholowalia, P. & Kumar, A. EBK-means: A clustering technique based on elbow method and k-means in WSN. *International Journal of Computer Applications* **105** (2014).

269. Demaine, E. D., Emanuel, D., Fiat, A. & Immorlica, N. Correlation clustering in general weighted graphs. *Theoretical Computer Science* **361,** 172–187 (2006).

270. Saramäki, J., Kivelä, M., Onnela, J.-P., Kaski, K. & Kertesz, J. Generalizations of the clustering coefficient to weighted complex networks. *Physical Review E* **75,** 027105 (2007).

271. Krzakala, F. *et al.* Spectral redemption in clustering sparse networks. *Proceedings of the National Academy of Sciences* **110,** 20935–20940. ISSN: 0027-8424. https://www.pnas.org/content/110/52/20935 (2013).

272. Goutte, C. & Gaussier, E. *A probabilistic interpretation of precision, recall and F-score, with implication for evaluation* in *European Conference on Information Retrieval* (2005), 345–359.

273. Frey, B. J. & Dueck, D. Clustering by Passing Messages Between Data Points. *Science* **315,** 972–976. ISSN: 0036-8075. http://science.sciencemag.org/content/315/5814/972 (2007).

274. Abbe, E. & Sandon, C. *Community detection in general stochastic block models: Fundamental limits and efficient algorithms for recovery* in *2015 IEEE 56th Annual Symposium on Foundations of Computer Science* (2015), 670–688.

275. Ho, T. K. *Random decision forests* in *Proceedings of 3rd international conference on document analysis and recognition* **1** (1995), 278–282.

276. McDonald, D., Birmingham, A. & Knight, R. Context and the human microbiome. *Microbiome* **3,** 52 (2015).

277. Lozupone, C., Lladser, M. E., Knights, D., Stombaugh, J. & Knight, R. UniFrac: an effective distance metric for microbial community comparison. *The ISME journal* **5,** 169 (2011).

278. Li, H. Microbiome, metagenomics, and high-dimensional compositional data analysis. *Annual Review of Statistics and Its Application* **2,** 73–94 (2015).

279. Fasano, G. & Franceschini, A. A multidimensional version of the Kolmogorov–Smirnov test. *Monthly Notices of the Royal Astronomical Society* **225,** 155–170 (1987).

280. Ng, A. Y., Jordan, M. I. & Weiss, Y. *On spectral clustering: Analysis and an algorithm* in *Advances in neural information processing systems* (2002), 849–856.

281. Yildirim, A. & Feig, M. High-resolution 3D models of Caulobacter crescentus chromosome reveal genome structural variability and organization. *Nucleic Acids Research* **46,** 3937–3952. ISSN: 0305-1048. https://doi.org/10.1093/nar/gky141 (Feb. 2018).

282. Oluwadare, O., Zhang, Y. & Cheng, J. A maximum likelihood algorithm for reconstructing 3D structures of human chromosomes from chromosomal contact data. *BMC genomics* **19,** 161 (2018).

283. Trieu, T. & Cheng, J. 3D genome structure modeling by Lorentzian objective function. *Nucleic acids research* **45,** 1049–1058 (2016).

284. Sauerwald, N. & Kingsford, C. Quantifying the similarity of topological domains across normal and cancer human cell types. *Bioinformatics* **34,** i475–i483. ISSN: 1367-4803. https://doi.org/10.1093/bioinformatics/bty265 (June 2018).

285. Tang, Z. *et al.* CTCF-Mediated Human 3D Genome Architecture Reveals Chromatin Topology for Transcription. *Cell* **163,** 1611 –1627. ISSN: 0092-8674. http://www.sciencedirect.com/science/article/pii/S0092867415015044 (2015).

286. Todd, S. *et al.* CSynth: A Dynamic Modelling and Visualisation Tool for 3D Chromatin Structure. *BioRxiv,* 499806 (2019).

287. Waldispühl, J., Zhang, E., Butyaev, A., Nazarova, E. & Cyr, Y. Storage, visualization, and navigation of 3D genomics data. *Methods* **142.** 3D genome mapping and analysis methods, 74 –80. ISSN: 1046-2023. http://www.sciencedirect.com/science/article/pii/S1046202317302359 (2018).