The application of evolutionary computation towards the characterization and classification of urothelium cell cultures

Zhen ZHANG

Doctor of Philosophy

UNIVERSITY OF YORK
Electronic Engineering

September 2018

Abstract

This thesis presents a novel method for classifying and characterizing urothelial cell cultures. A system of cell tracking employing computer vision techniques was applied to a one day long time-lapse videos of replicate normal human uroepithelial cell cultures exposed to different concentrations of adenosine triphosphate (ATP) and a selective purinergic P2X antagonist (PPADS) as inhibitor. Subsequent analysis following feature extraction on both cell culture and single-cell demonstrated the ability of the approach to successfully classify the modulated classes of cells using evolutionary algorithms. Specifically, a Cartesian Genetic Program (CGP) network was evolved that identified average migration speed, in-contact angular velocity, cohesivity and average cell clump size as the principal features contributing to the cell class separation. This approach provides a non-biased insight into modulated cell class behaviours.

Contents

List of Tables

List of Figures

Acknowledgements

First, my heartfelt and foremost gratitude goes to my first supervisor, Prof. Stephen Smith. He has been supporting my research by providing lots of helpful advice since my first day as a research student. I could not have imagined having a better advisor and mentor for my Ph.D study.

I would also like to thank my second supervisor, Dr. Steven Johnson - he's given me many useful suggestions and much assistance with my research life.

My thanks also go to Prof. Jenny Southgate and Dr. Dawn Walker for their insightful comments and encouragement.

Finally, I must express my very profound gratitude to my parents and to my girlfriend Xun for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

Declaration of Authorship

I declare that this thesis is a presentation of original work and I am the sole author. This work has not previously been presented for an award at this, or any other, University. All sources are acknowledged as references. Where the thesis is based on work done by myself or jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Sections of the work described in this thesis have been previously published in the following journal article and conference proceeding.

Journal Article

- Zhang, Z., Bedder, M., Smith, S. L., Walker, D., Shabir, S., & Southgate, J. (2016). Characterization and classification of adherent cells in monolayer culture using automated tracking and evolutionary algorithms. Biosystems, 146, 110-121.

Conference Paper

- Zhang, Z., Bedder, M., Smith, S. L., Walker, D., Shabir, S., & Southgate, J. (2015, September). Automated motion analysis of adherent cells in monolayer culture. In International Conference on Information Processing in Cells and Tissues (pp. 185-194). Springer, Cham.

Chapter 1

Introduction

1.1 Classification and characterization of cell cultures

Cell culture is a method of simulating the internal environment (sterile, suitable temperature, pH and certain nutritional conditions, etc.) in vitro to enable it to survive, grow, reproduce and maintain its main structure and function. Cell culture is an essential process, both for the whole of bioengineering and for one of the biocloning technologies, and cell culture itself is the mass cloning of cells. The cell culture technique can consist of a single cell being cultured in large numbers into simple single cells or multicells with very little differentiation. Cell culture is an important and commonly used technique in cell biology research methods, through which it is possible to obtain a large number of cells, but also to study the cell anabolism, cell growth and proliferation.

To characterize complex biological systems requires the integration of experimental and computational research. Currently, ways to understand cell culture include computational modelling[1]–[4] and cell tracking software[5], [6].Computational systems biology resolves fundamental questions in our knowledge of life, but advances here will lead to discoveries in medicine, drug discovery and engineering.

1.2 Opportunitiess for machine learning

Machine learning, is a branch of the field of artificial intelligence whose basic idea is a discipline that builds statistical models based on data and uses

the models to analyze and predict the data.

The most basic approach to machine learning is to use algorithms to parse data, learn patterns from the data, master those patterns, and then make decisions or predictions about events in the real world. Unlike traditional software programs that are hard-coded to solve specific tasks, the core of machine learning is to train with large amounts of data and learn from the data how to accomplish the task through various algorithms

Current experimental molecular biology is now producing the high-throughput quantitative data needed to support research using numerical methods such as machine learning. At the same time, substantial advances in software methods and computational power have enabled the understanding and analysis of the intricate biology.

1.3 Hypothesis

The following hypothesis is used to guide the work presented in this thesis:

" ***Evolutionary Computation is an effective means of characterising and classifying urothelium cell cultures through time-lapse spectroscopy***"

It is proposed that using Evolutionary Algorithms provides a novel method for tracking and characterizing urothelium cell cultures. Specifically, that a Cartesian Genetic Program (CGP) network can be evolved that identifies characteristics of cell migration as the principal features contributing to the classification of cell cultures. This approach not only provides a non-biased and parsimonious insight into modulated cell class behaviours, but can provide a discrete mathematical expression describing these for the parameterization of related computational models.

1.4 Thesis outline

Chapter 2 provides a background to cell culture, urothelium and machine learning. In Section 2.1, the literature of past work on this problem are reviewed and important methods and techniques are introduced. In Section 2.2 Evolutionary Algorithms are introduced including the concept, structure and range of algorithms available. Specifically, Genetic Programming is considered, the algorithms discussed and the related literature reviewed.

Chapter 3 considers the methodology adopted in this work. Firstly, the underlying biological processes of the urothelium and how the cell tracking techniques adopted are applied is described in Section 3.2. In Section 3.3 details of the definition and implementation of features to be extracted from the cell culture time-lapse spectroscopy is explained in depth from both a biological and computational viewpoint. The implementation and configuration of the evolutionary algorithm employed, CGP, is described in Section 3.4. The visualization of the data and preparation of the data sets are considered in Sections 3.5 and 3.6 respectively. The application of the evolutionary algorithm to the data sets is then described in Section 3.9, following consideration of alternative approaches using Principal Component Analysis (Section 3.7) and Support Vector Machines (Section 3.8).

Chapter 4 presents the results and analysis of the experimental work, for each of the methods introduced in Chapter 3.

The thesis conclusion is provided in Chapter 5 summarising the principal findings from the work undertaken, the implications of these and recommendations for future work.

Chapter 2

Literature Review

Contents

2.1 Introduction

This literature review comprises four parts: the first introduces the ***cell*** and ***cell culture***, and importantly, the role of cell culture in biological research. The second part introduces the ***urothelium*** along with the function and significance of urothelial cells. The third reviews current ***technologies*** applied in the analysis and understanding of cell behaviour within cell cultures, including modeling and cell tracking. The final part of this chapter reviews ***Machine Learning*** and comparable numerical methods for characterizing and classifying cell cultures.

2.2 Cell culture and characterization

2.2.1 The Cell

Frequently referred to as the 'building block of life', a cell is the smallest unit of life and forms the elementary structural and biological unit of every identified living organism.

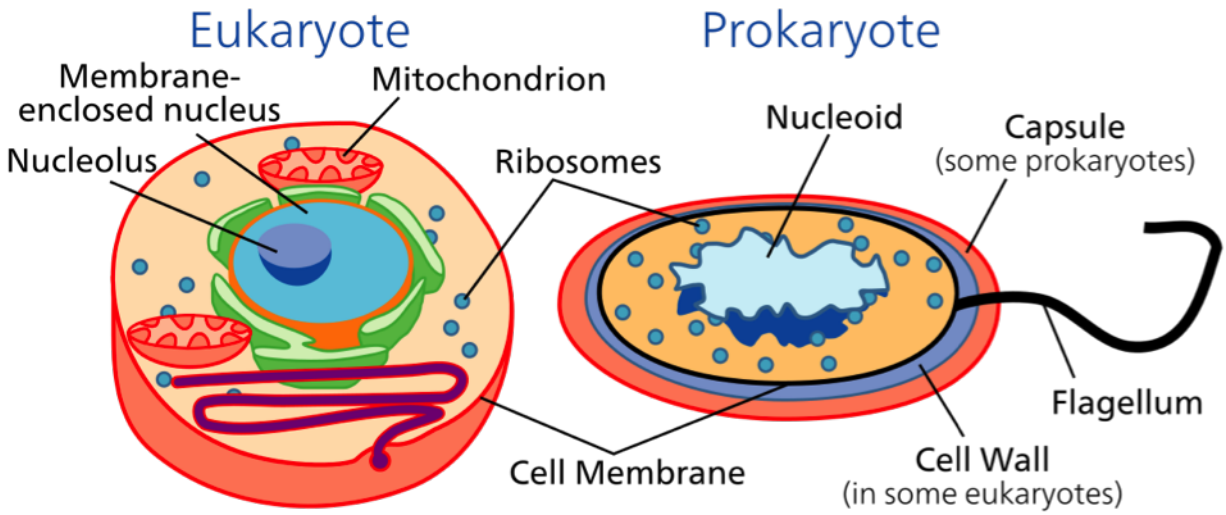


FIGURE 2.1: Cell Structure[11]

Cells comprise a cytoplasm surrounded by a membrane which consists of numerous biomolecules, including proteins and phospholipids.[7] It is possible to categorise organisms as multicellular (including plants and animals) or unicellular (e.g. bacteria).[8] Human beings have in excess of 10 trillion (10^{13}) cells, with this number differing for other plants or animals. It is only possible to observe the majority of plant and animal cells using a microscope, with dimensions ranging between 1 and 100 micrometres.[8]

In 1665, Robert Hooke discovered the cell. The units took their name from their similarity with the cells that Christian monks had inhabited in a monastery.[9] Matthias Jakob Schleiden and Theodor Schwann were the first to formulate Cell Theory in 1839. This asserts that every organism consists of at least one cell; that every cell originates from pre-existing cells; that each cell incorporates the hereditary data required for controlling the ways cells operate and for passing information to subsequent generations of cells; and that cells are the essential unit of structure and function in every living organism.[10] Cells came into existence on Earth more than 3.5 billion years ago.

Cell Types

There are two types of cells. The first, termed *eukaryotic*, can be either single-celled or multicellular and possess a nucleus. This is in contrast to the second type, *prokaryotic* (single-celled organisms), which are devoid of a nucleus.

Prokaryotic cells Prokaryotes, comprising the bacteria and archaea, are two of the three kingdoms and were the first form of life to emerge on Earth. They exhibited crucial biological functions such as cell signaling. In comparison to eukaryotic cells, prokaryotic cells are less complex and not as big. They are devoid of membrane-bound organelles like a nucleus. A prokaryotic cell's DNA comprises a single chromosome that has direct contact with the cytoplasm. The nuclear region in the cytoplasm is referred to as the nucleoid. The majority of prokaryotes are the smallest of all organisms, with their diameter ranging from as little as 0.5 to 2.0 µm.[12]

Eukaryotic cells The following are all eukaryotic: plants, animals, fungi, slime moulds, protozoa, and algae. Such cells are approximately fifteen times wider in comparison to a normal prokaryotic cell; their volume has the potential to be one thousand times the size. It can be observed that compartmentalization is the most notable aspect of eukaryotes in contrast to prokaryotes. Compartmentalization refers to membrane-bound organelles (compartments) being present where particular functions occur. The most significant of such functions is a cell nucleus.[8] This is an organelle storing the DNA of the cell. This nucleus is responsible for the name of the eukaryote ("true kernel (nucleus)").

2.2.2 Cell culture

Cell culture refers to the method of cells being grown within controlled conditions, which ordinarily deviate from their natural environment. Once the relevant cells have been separated from living tissue, it subsequently becomes possible to control them with the necessary conditions. Such conditions differ for every type of cell, although they are typically involve an appropriate vessel with a substrate or medium that provides the required nutrients (amino acids, carbohydrates, vitamins, minerals), additional growth factors and hormones, and gases (CO_2, O_2), and controls on the physiochemical environment (pH buffer, osmotic pressure, temperature). Most cells require a surface or an artificial substrate (adherent or monolayer culture) whilst it is possible to grow others free floating in culture medium (suspension culture). The majority of cells have their lifespan dictated by genetics. However, some culturing cells have been converted into immortal cells allowing them to reproduce indefinitely should the necessary conditions be created.[13]

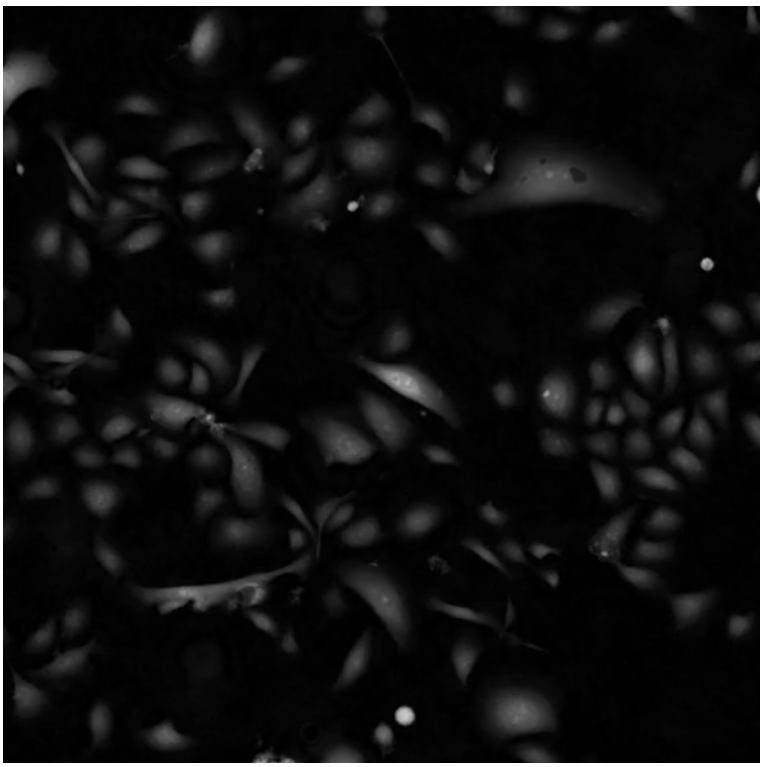


FIGURE 2.2: Normal human urothelial(NHU) cells in culture

The concept "cell culture" practically relates to cells originating from multicellular eukaryotes, particularly animal cells, in comparison to alternative forms of culture that also produce cells. These include plant tissue culture, fungal culture, and microbiological culture (of microbes). The historical evolution and processes of cell culture are strongly linked to the development and methods of tissue culture and organ culture. There is an additional connection to viral culture, with cells as hosts for the viruses.

2.2.3 Application of cell culture

One aspect that is crucial to the formation of viral vaccines and other products of biotechnology is the mass culture of animal cell lines. The culture of human stem cells is utilised in order to broaden the quantity of cells and to categorise the cells into a range of somatic cell types for transplantation.[14] Stem cell culture is additionally utilised in order to harvest the molecules and exosomes produced by the stem cells in order to ensure therapeutic progression.[15]

Biological products generated by recombinant DNA (rDNA) technology in

animal cell cultures include enzymes, synthetic hormones, immunobiologicals (monoclonal antibodies, interleukins, lymphokines), and anticancer agents. Whilst it is possible to generate numerous less complex proteins utilising rDNA in bacterial cultures complex, secreted and glycosylated proteins need to be formed in animal cells. The hormone erythropoietin is a significant instance of such a complex protein. Because of the expense of developing mammalian cell cultures, research into the use of direct gene transfer using particle bombardment and transitory gene expression of insect cells, higher plants, single embryonic cell and somatic embryos has also proved effective. Confocal microscopic observation additionally provides the opportunity of verifying the single cell origin of somatic embryos and the asymmetry of the first cell division, which commences the process.

Additionally, cell culture is an important process for cellular agriculture. The objective of this is to offer innovative products as well as innovative methods of generating existing agricultural goods such as milk, (cultured) meat, fragrances, and rhino horn from cells and microorganisms. Consequently, it is regarded as one way of fulfilling animal-free agriculture. Furthermore, it is a crucial means of providing education regarding cell biology.

2.3 Urothelium

The smooth layers of muscle which encompass the pelvic urethra, the ureter and the bladder require the covering epithelium to give a sonic hedgehog signal from the covering in order that the mesenchyme is distinguished from it. It is surprising that this signal continues to be expressed in the entire life, not being the urothelium specifically.[16].

Contrastingly, the urothelium is, in fact, a highly-specific urinary tract covering the bladder's surface, pelvic urethra, renal pelvis and the ureter which is adjacent to the deferent ducts' outlet in rodents.

2.3.1 Structure and functions

Urothelium, otherwise referred to as "transitional epithelium" comprises one basal cell layer in rodents and more than one in larger mammals like human beings and cattle, in addition to a superficial cell layer and an intermediate cell layer.

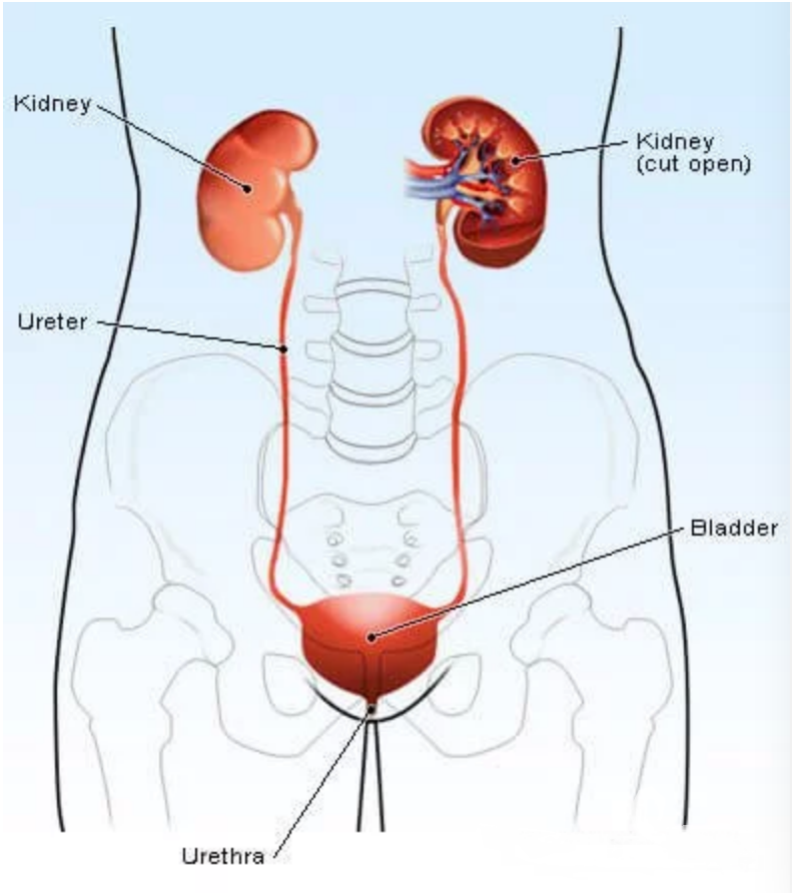


FIGURE 2.3: Structure of Urinary system [17]

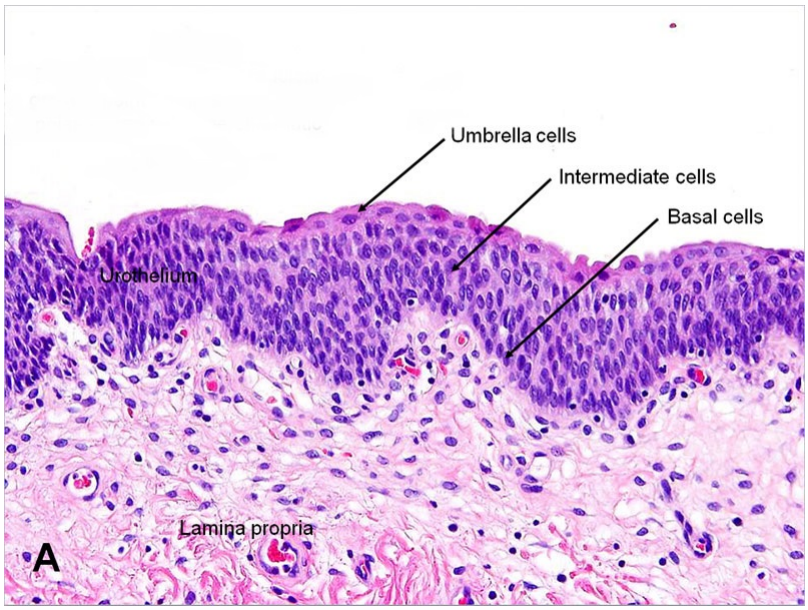


FIGURE 2.4: Umbrella, intermediate and basal cells[18]

The "umbrella" cells which form the highly-distinctive and superficial urothelium layer do not stretch to the basement of the membrane basement. Instead, pseudostratified epithelium conceals the collecting kidney papilla and ducts, [19] , being the ureteric bud's most distal area. Nevertheless, human beings and rodents are different in that the epithelium which conceals the papilla is itself concealed by urothelium instead of pseudostratified epithelium. Despite the fact that the urothelium has a particularly slow turnover rate of 200 days, progenitor cells can be discovered throughout the urothelium.[20] Nevertheless, there is a regional difference in their density which is about twice as high in the base of the bladder than it is in its dome.[21]

Two of the of the urothelium's physiological attributes are its capability of accommodating stretch and its particular electrical resistance which is of a high transepithelial nature. [20]. Both of these properties are attributed to the so called "urothelial plaques". These are two-dimensional concave-shaped crystalline frameworks of 0.3-1mm diameter which occupy 90 percent of the apical surface of the urothelium. These plaques are hexameric 16-nm particle aggregates which subsequently comprise four different uroplakin sub-units. These asymmetrics are unit membranes which are outwardly concave and connected by constricted "hinge" areas.[20]. At different anatomical sites, urothelia differ quantitatively. [19] For example, the ureteral urothelium has a lower content of cytoplasmic fusiform vesicles and uroplakin proteins than does the bladder urothelium.[20] However, it is not yet settled whether the urothelium varies qualitatively within different areas. Furthermore, the plaques possess clinical suitability since they permit uropathogenic Escherichia coli to adhere to their surface.[20]

The late fetal and early postnatal period in rodents is characterised by extreme remodelling of the lining of the bladder. During this period of time, proliferation of the urothelium, desquamation and programmed cell death are involved in far greater activity than at any other time. [22] . Cytokeratin and uroplakins are in continuous existence within the urothelium. However, the passage of cytokeratin 18 into cytokeratin 20 expression[22] as well as the emergence of polyploidy ker (1958) urothelial cells coexist with the commencement of the remodelling period at the final prenatal day, and uroplakin-positive cells continually co-express cytokeratin 20.[19]. The

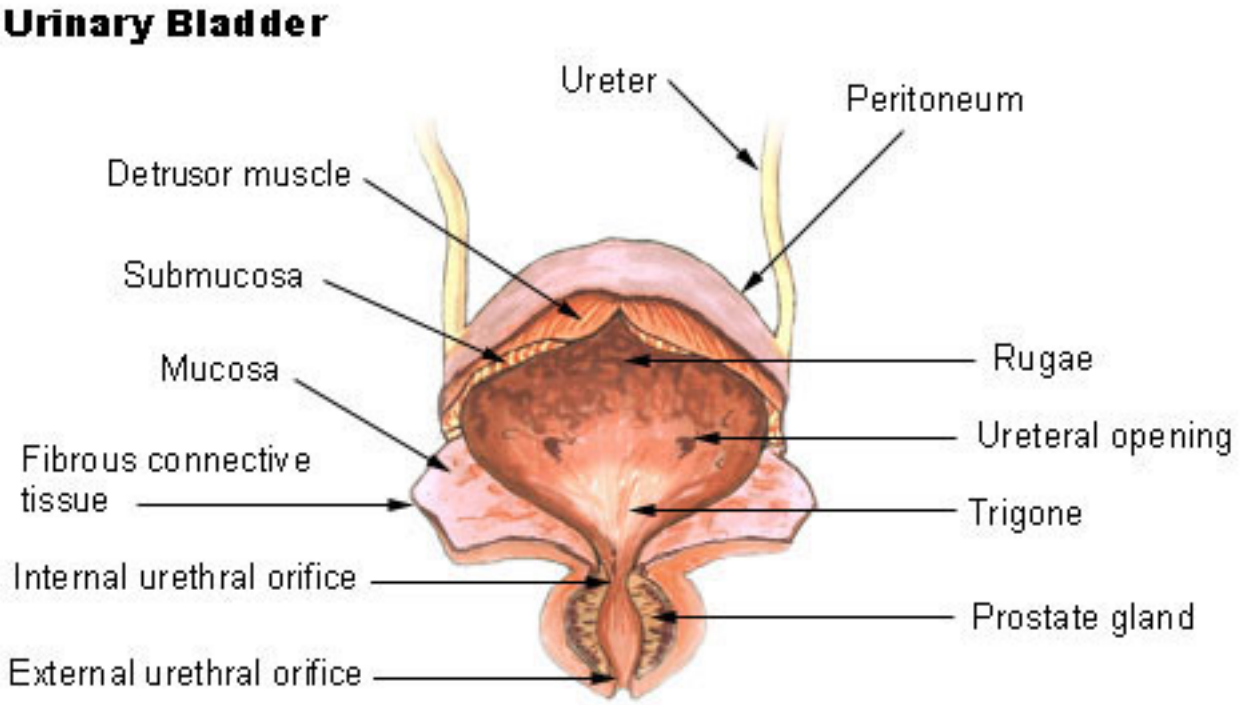


FIGURE 2.5: Bladder[23]

urothelium expresses several kinds of receptors, among which are cholinergic receptors. The activity of these receptors, having a sensory role, is controlled by the activity of the sub-urothelial nerve plexus, and consequently, local variations in vascular perfusion and/or reflex contractions of the bladder. In the trigone area, the sub-urothelial nerve plexus is especially well advanced.

2.3.2 Significance

In response to physical or any other type of impairment, the urothelium transfers quickly and transiently from a stable and mitotically-quiescent and obstacle into a state which is extremely proliferative. This switch is facilitated by mechanisms which are key to the bladder's pathophysiology, but are not well understood. It is reported that the urothelium responds to chemical and mechanical stimulation by releasing soluble factors which include adenosine triphosphate (ATP), which are expected to occupy a function in mediating neuronal signalling.[24] Furthermore, the urothelium indicates purinergic P2X and P2Y channels and receptors which respond to ATP which is released from paracrine or autocrine origins.[25] The consequences of this signalling process are not fully comprehended, since it may

possess a feedback function in the modulation of neuronal signalling or it may also occupy a more direct function in the repair of an urothelial obstruction.[25] Furthermore, it has been advocated that aberrant indication of receptors and/or mediators which the urothelium releases is involved in dysfunctional bladder diseases which include and interstitial cystitis and idiopathic detrusor instability.[26] Irrespective of the indicated expression of these receptors and channels by the urothelium, agreement has been confounded by the contradictions in experimental techniques including the species specificity of reagents and the characteristics of the tissue development.[27]

A cell culture technique to examine usual human urothelial (NHU) tissues and cells in vitro has been developed. In previous work, the application of this culture technique demonstrated that promptin P2 receptors with exogenous ATP advanced the repair of scratch wounds in addition to the ecto-ATPase inhibitor ARL-67156. This avoids a failure of autocrine-generated ATP. In contrast, whether ATP was present or absent, the scratch wound repair[25] was inhibited by the blockade of P2X activity. This shows ATP to be among the main elements released in the case of urothelial impairment which will probably contribute to urothelial barrier repair.

Urothelium is the transitional epithelium which lines the bladder and is related to the role of the urinary tract as a stable, yet self-repairing, urinary obstruction. The urothelium forms a complete barrier when the superficial cells differentiate and acquire surface membrane plaques consisting of uroplakins[29] and well-developed tight intercellular junctions[30]. The separate cells inside the urothelium are sustained in a state which is mitotically-quiescent until, following impairment to the obstruction, cells from every layer move into a proliferative phenotype for the purpose of effecting barrier repair.[31] The balance between differentiation and the paradoxical procedures of regeneration is crucial to sustaining an efficient urinary obstruction. However, the mechanisms controlling urothelial tissue homeostasis and the exchange between regenerative (self-repair) and quiescent phenotypes are not well comprehended.

A simple, replicable method for culturing separated normal human urothelial (NHU) cells has been developed which allows functionally-differentiated and regenerative obstruction states to be obtained. [32] In most basic situations, NHU cells grown in serum- free, low-calcium conditions assume a “basal” epithelial cell phenotype in which cell growth is in migratory,

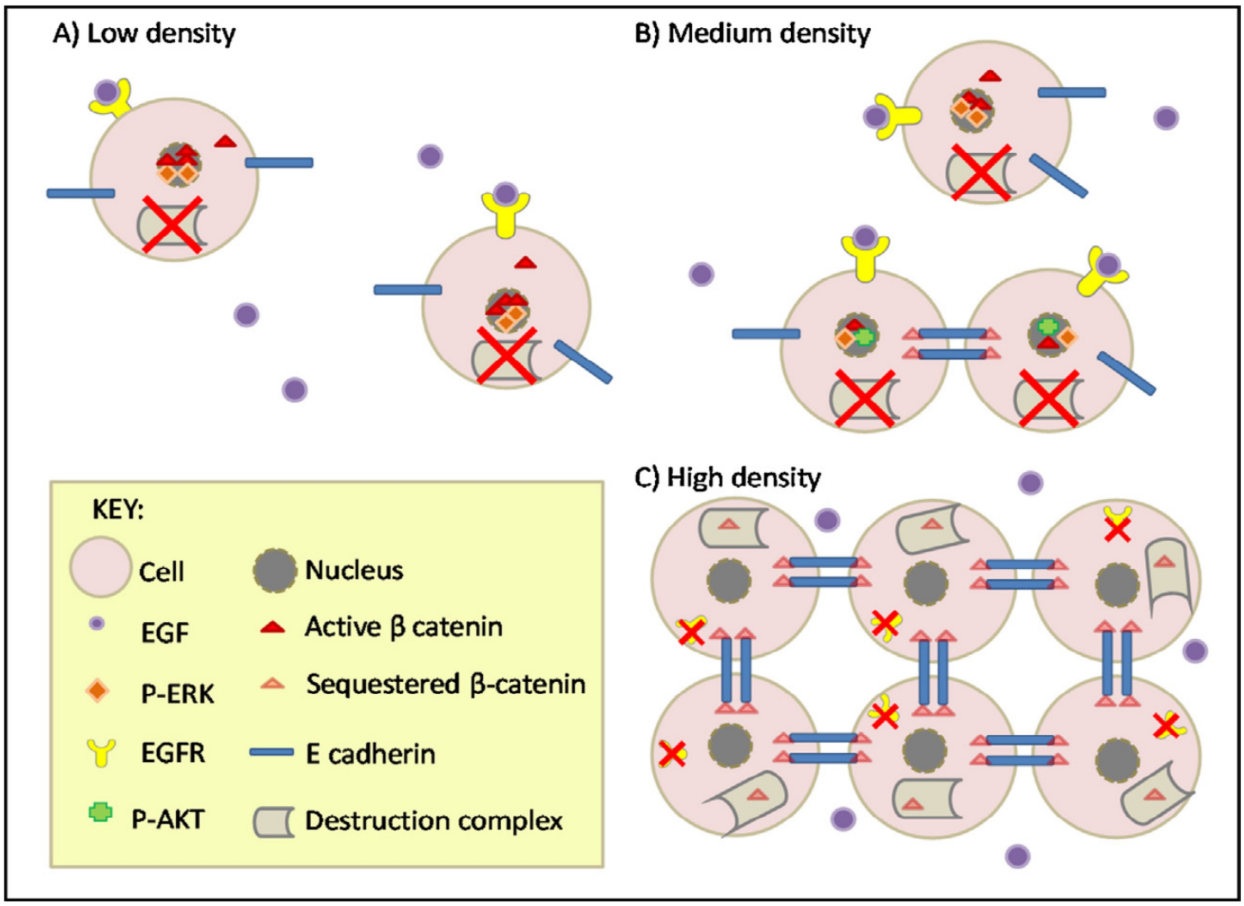


FIGURE 2.6: Schematic model of Wnt/-catenin signalling crosstalk with EGFR/ERK and cell:cell contact-mediated -catenin regulation in NHU cell proliferation.[28]

non-stratified cultures which are very proliferative and which at confluence become contact-inhibited. [33]. By manipulating exogenous calcium from low [0.09 mM] to near-physiological [2 mM], the E-cadherin (adherens) contacts can be altered. This may also be achieved by retroviral transduction of a negative E-cadherin (H-2Kd-E-cad) construct. Furthermore, it has been demonstrated that population growth can be encouraged by manipulating the stability of E-cadherin cell–cell contacts. This is differentially controlled through the Phosphatidylinositol 3-Kinase (PI3-K)/AKT signalling pathway and the Epidermal Growth Factor Receptor (EGFR)/Extracellular Signal-Regulated Kinase (ERK) pathway. [34] The EGFR/ERK pathway is down-regulated by stable adherens junctions while they induce the activity of the PI3-K/AKT pathways which in turn advances intensification at low-cell density. The proliferative capacity of NHU cells is impeded by E-cadherin. Functional inactivation of these stable contacts has been shown to be calcium mediated and also reduces E-cadherin-mediated PI3-K/AKT induction. However, it improves the proliferation of NHU cells by advancing the autocrine-driven EGFR/ERK pathway, thereby activating b-catenin-TCF signalling through GSK3 phosphorylation and inactivating the inhibition. Moreover, in this case when the EGFR is blocked, the NHU cells respond to canonical Wnt signalling, provided the exogenous Wnt ligand is supplied endogenously and the cells are cultured with palmitic acid. This enables post-translational palmitylation of autocrine-produced Wnt ligands.[28]

Therefore the inherent self-repair capacity is controlled by a complicated interrelationship between a minimum of three intracellular autocrine-controlled pathways which interact by the control of the availability or activity of components of these major pathways (e.g. E-cadherin, EGFR, pERK, b-catenin, and pAkt, as shown in Figure. 1). The E-cadherin-defined adherens junction, in practice, serves as a principal regulator through which urothelial cells separately “sense” and initiate neighbour regulation for the purpose of controlling the growth of a population of cells. This is particularly self-regulating and context-specific within the local population as the cell density and the tendency of cells to generate cell-cell bonds influence the activation of downstream pathways and receptor availability. Additional regulation is also achieved by both positive and negative feedback between the signalling pathways. Population homeostasis disturbance; for example, by scratch-wounding a confluent contact-inhibited culture, results in the cells

having a local response. These are complicated interactions and, as previously shown, the PI3K pathway contribution[34] may be informed from modelling.

2.4 Technologies in cell characterization

Over the past two decades, bioimaging technologies have progressed considerably, permitting the dynamic processes of living cells to be investigated at high temporal and spatial resolutions .

2.4.1 Cell tracking

Cell tracking is a key technology that facilitates quantitative analysis of intracellular dynamic processes, such as cell migration, which are characteristic in normal tissue development and tissue disease. Speed and migration type, and the morphological transformations experienced by the cell in the course of movement are distinctly associated with the biomechanical properties of the surrounding environment.[35], [36] Consequently, precise quantification of both is important for comprehending the complicated mechanobiology of cell migration.

A hierarchical cell-tracking technique which comprises the two functions of cell tracking and detection is sometimes referred to as a *Hift*. The input to the Hift is a sequence of raw microscopy images or frames from a time-lapse spectroscopy video. A filtering technique is first applied within the Hift to eliminate noise and improve the contrast of the raw images or video frames. The cells within each resulting image are then located and segmented as part of the detection phase; the cell's centroid being used as a unique marker by which subsequent tracking of the cell can be performed. Consequently, it is possible to abstract each cell as a point, using its located centroid, which is of particular use when tracking the cell in other cultures and contexts. Hence, the proposed Hift provides a common structure which can be used for tracking various cell types.

There are three stages to the Hift tracking module that are concerned with reliably tracking the trajectory of cells over successive frames of the time-lapse video. The first stage is concerned with establishing interconnections between every pair of cell centroids in the two neighbouring frames, which

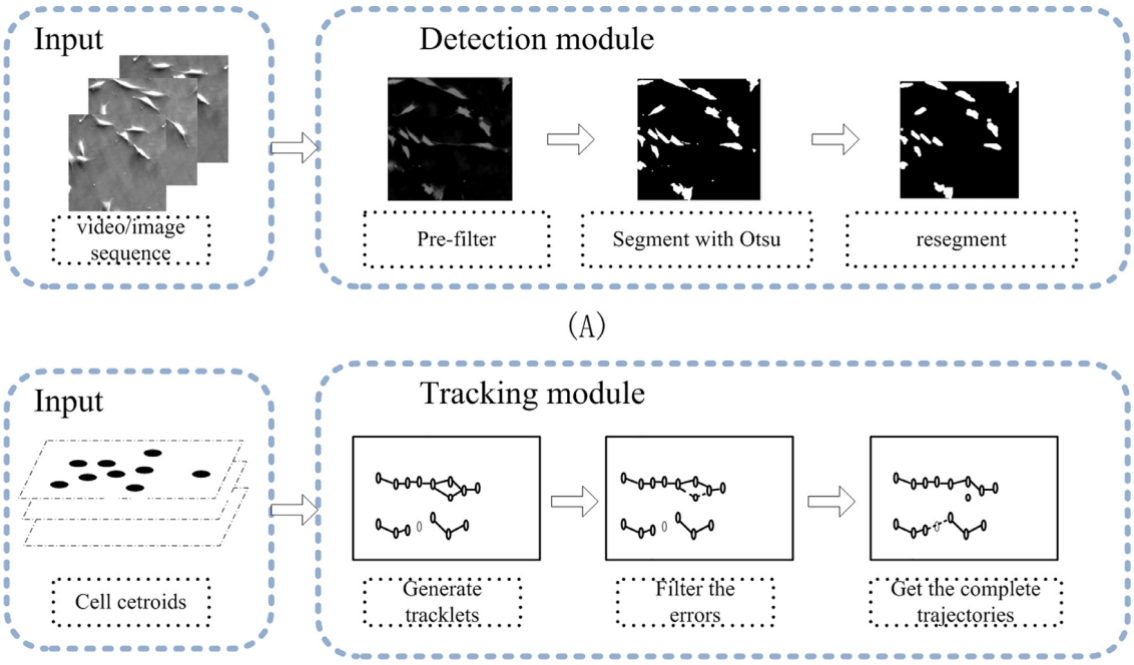


FIGURE 2.7: Overall flowchart of Hift cell tracking system. The upper figure (A) introduces the detection module, and the output of this module is a structure of cell locations in all frames. The below figure (B) is the tracking module, after the tracklet generation step, a bunch of short tracklets are established, which link all the detected cells. The cell motion events will be recognized based on the tracklets to filter the errors and complete trajectories.

are logged for subsequent analysis. The second stage involves the recognition and correction of temporary merge and division cell events according to the tracking analysis previously completed. The third and final stage locates "lost" cells through the application of an optimisation procedure that uses temporal and spatial active field to match the lost cell occurrences and establish their final trajectories. Figure 2.7 depicts the Hift flowchart.

Ctracker

Ctracker is a software system developed by Matthew Bedder (Department of Computer Science at the University of York) that utilises the Open CV computer vision programming library for automated cell tracking from time-lapsed videos.[37] Each frame is processed individually in order to identify candidates for tracking, and subsequently, characterising, cell movement within a cell culture.

The raw time-lapse video is read into Ctracker which pre-processes each frame to identify the position of new cells or track the movement of previously identified cells.

Gaussian blurring is applied to each frame of the movie in order to eliminate noise. Intensity thresholding to a preordained value generates a binary image, effectively segmenting the cells from the background of the video frame image. Subsequently, a distance transform is applied to the binary image, producing a "distance image" in which the centres of cells (or cell groups) are allocated a high value, the edge of cells a low value, and the background a value of zero. The distance images' local maxima are then calculated for the purpose of providing an estimate of the location of the centres of the cells within the frame. Multiple local maxima located within a small area are filtered in order to decrease the number of false cell registrations due to multiple maxima being identified within a single cell's body. The resulting maxima's (x,y) coordinates are subsequently used to estimate the cell locations inside the frame. To confirm this, the position of each respective cell in the previous video frame, relative to to the distance image, is multiplied by a basic Gaussian filter and the maximum value of the pixel in this area used to assess the position of the new cell. Despite the simplicity of this approach, it has been shown to be both effective and efficient. The distance image identifies the centre of the cells and the application of the Gaussian filter confirms that the preferred matches are those adjacent to the original cell location. Despite the efficiency of the cell-tracking procedure, it cannot always recognise and track cell positions for the entire duration of the video. Initially, multiple candidate cell locations are calculated for the purpose of identifying as many cells as possible, many of which rapidly converge to the same locations. Likewise, occasionally, the cell-tracking procedure fails to track the location of cells in subsequent frames, leading to an overall decline in the number of cells tracked over the duration of the video. Such problems in tracking cells may be caused by cell proliferation (production of new cells), cell death (cell loss), occulsion (one cell moving in front of another) and cells travelling in or out of the field of view. A technique was devised to reduce the effect of these issues by periodically restarting the cell recognition procedure, to identify all cell positions without dependency on the location of cells in the previous video frame. The approach was effective in tracking an adequate number of cells for the duration of the video, which sufficiently described the cell population. The entire cell-tracking procedure is outlined in figure 2.8.

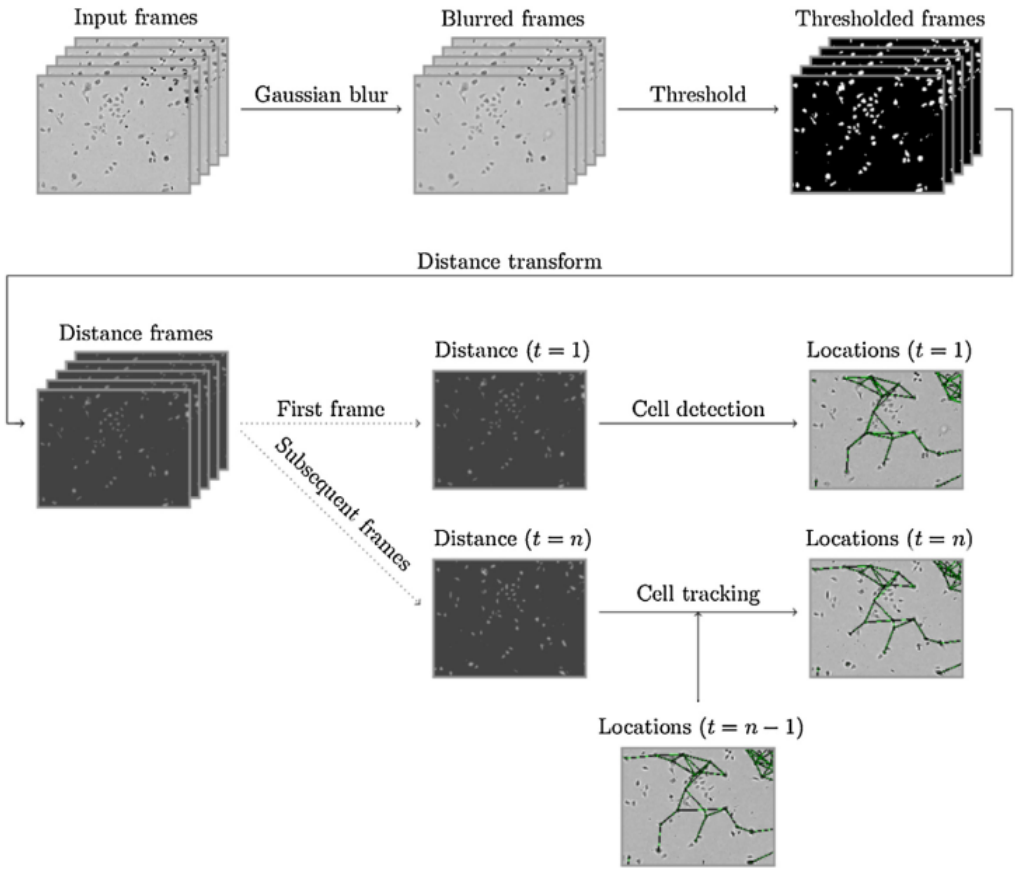


FIGURE 2.8: The process used for detecting cell locations within a video, and tracking of detected cells between video frames.

2.4.2 Modeling

The field of systems biology has, over the past decade, gained considerably from computational paradigms and methods previously used exclusively in computer science to evaluate a program's safety and correctness. The design of a biological paradigm within this setting equates to the development of a computer program. Several programming languages, most of which are biological domain specific, provide a method of defining the instruction series which defines the biological process control flow.

The language syntax defines the methods by which the symbols can be merged to construct well-formed instructions or sentences. Frequently, this specification is represented textually; for example, in a rule-based system or process calculus. However, in many situations, a graphical representation such as state charts or Petri nets are also available. In this way, the user is assisted in the visualisation of the procedure with diagrams which illustrate species flow in the change to their internal states or their reactions. The semantics describe the behaviour of the paradigm and how it ought to be executed by the computer, thereby revealing the interpretation of the syntactically valid instructions. Furthermore, there is a likelihood that a particular paradigm, by the utilisation of a specific language syntax, could be applied by utilising various language sources; for instance: a series of rules on chemical reactions may be applied by utilising ongoing linguistics (ordinary differential equations [ODEs] on concentrations of molecules) or stochastic semantics (number of molecules), dependent upon the standard of complexity and/or approximation [38] that is desired. For instance, COPASI [39], [40] is an instrument for numerical imitation and examination of biochemical networks for both their stochastic and continuous dynamics.

Compartment-based models

It is usual for biological systems to be organised into compartments such as the cell nucleus, cell membrane and organelle. The compartments exchange molecules between one another in accordance with certain rules. Specialised compartment-based paradigms (see figure 2.9) capture many biological properties; for example, the compartments' dynamic rearrangements represent a type of behaviour noted in the mitochondria. The paradigms also capture molecules movement between these.

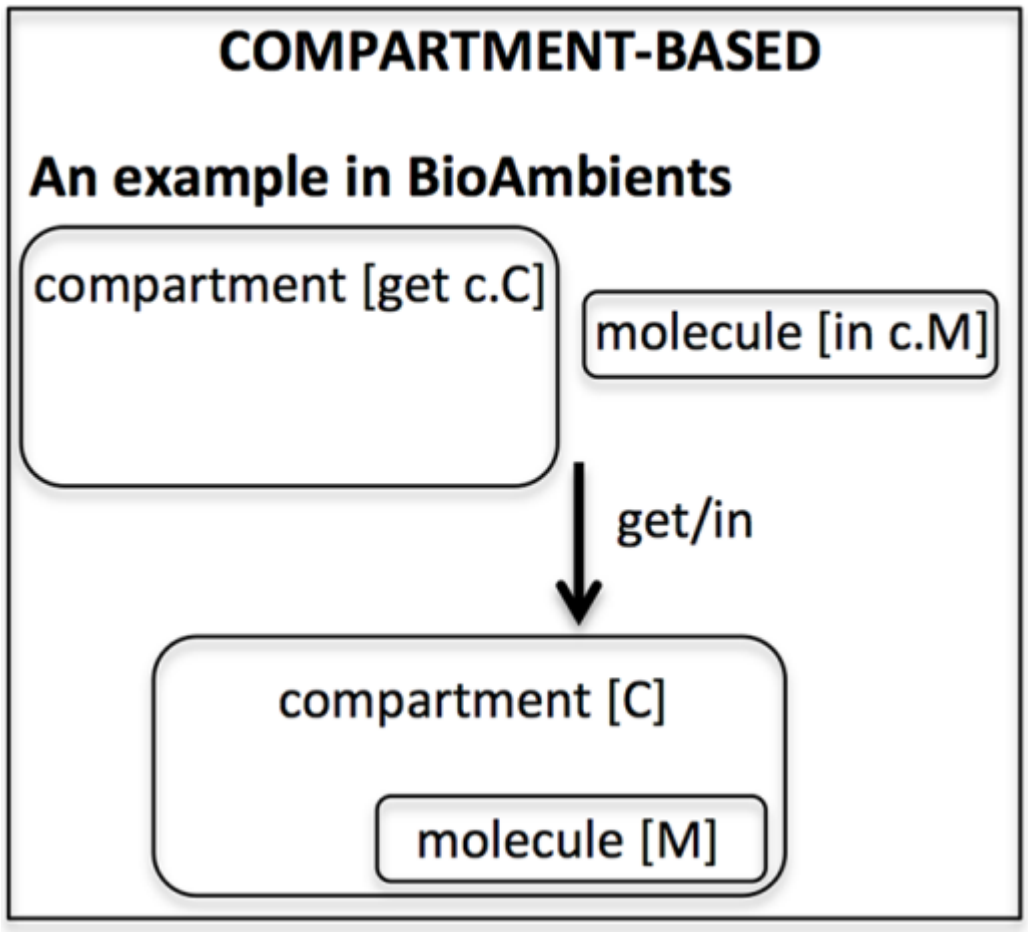


FIGURE 2.9: Compartment Based Model [41]

Moreover, the membranes' study which separates the sections has in computer science implemented a novel field known as membrane computing whose objective is to find new bioinspired computational models, like the P systems.[42] Nevertheless, such paradigms are more suitable for computation theory rather than for modelling in systems biology.

Bioambients [41] is another appropriate modelling structure in which a process algebra enhanced with important operators can particularise splitting and merging as well as communication between biological compartments. BAM [43], [44] is an instrument to execute stochastic bioambients which progressed into Brane calculus [45] and presents a language to define the vigorous membrane behaviour. However, in bioambients, ambient calculus, otherwise the compartment, occupies an active function in stipulating which procedures may enter or leave from this. Moreover, Brane calculus presents another aspect where membranes have control of the coordinators and occupy their function. Currently there is no implementation of this that is available.

Agent-based models

Agent-based models[46], [47] (see figure 2.10) consider a collection of autonomous decision-making entities known as agents. These individually sense the environment and subsequently reach decisions according to a series of rules. Despite that fact that, at its most basic level, an agent-based paradigm comprises a system of agents and the associations between them, it is still able to show complicated behavioural patterns regarding changes and adaptation responding to environmental challenges or to neighbouring agent behaviour; for instance, cooperation or competition.

Potentially, each agent has a unique history and behaviour, because each is explicitly represented in a population. Occasionally, more complicated agent-based models assimilate sophisticated learning and adaptation rules on the basis of evolutionary algorithms, neural networks or other systems.

The single- cell-based paradigms represent a particularly favourable facet, in which agents have several cellular structural and functional elements and behaviour edging towards reality and allowing the recognition of phenomena at various intermediate biosystems' scales.

Cell-based paradigms have the ability to express significant behavioural attributes of a cell like the dynamics of its repetition and information on each

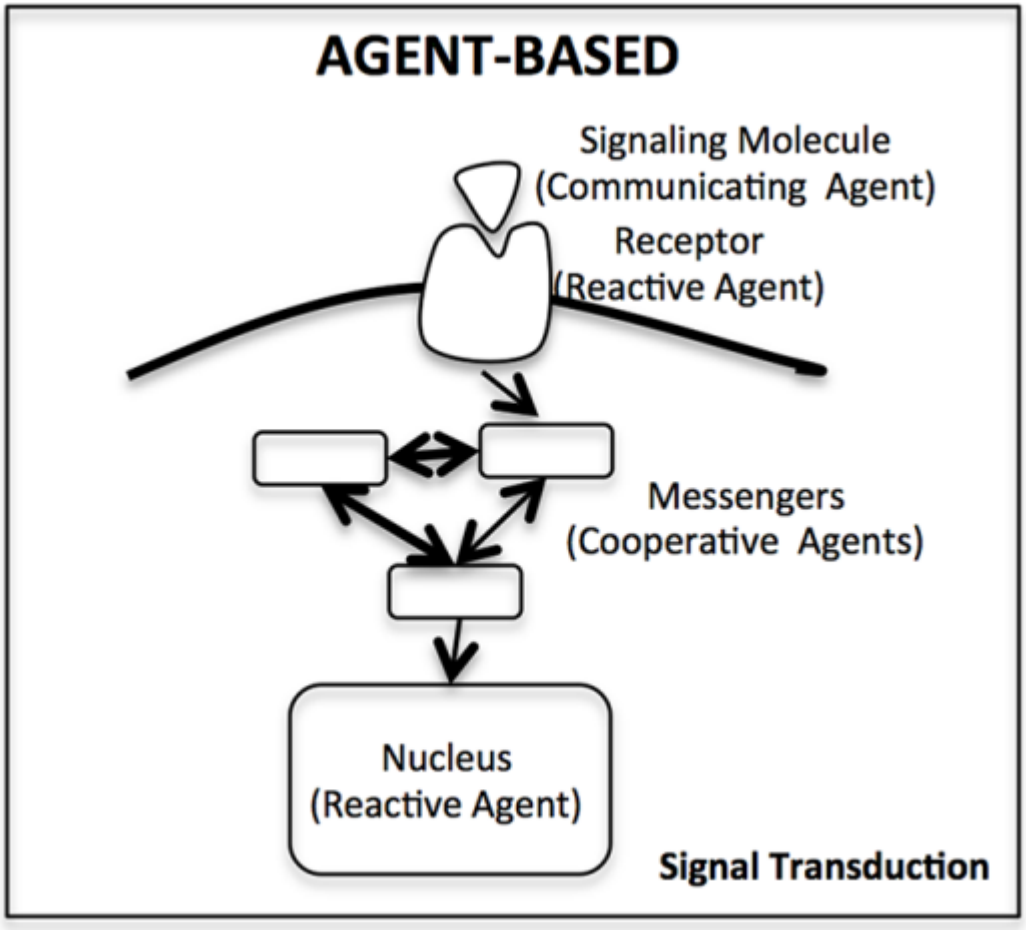


FIGURE 2.10: Agent Based Model[48]

developmental step; for example, size, mechanical properties and cell geometry.

A single cell-based paradigm ought to have the ability to comprehend the way in which stage-dependent cell-cell directions at a microscopic scale will result in interactions in the cell tissue mechanical attributes of the tissue at a macroscopic standard and stage heterogeneity. For instance, paradigms may be introduced by applying FLAME[1], [48] and REPAST.[49], [50]

Lattice-based models

A lattice (see Figure 2.11) which describes a periodically repeated graph, is formulated by n-dimensional closed grid sites and typified by fixed or periodic border conditions in every direction, is specifically relevant for a technique's definition of interlinked procedures at the cellular, tissue, molecular

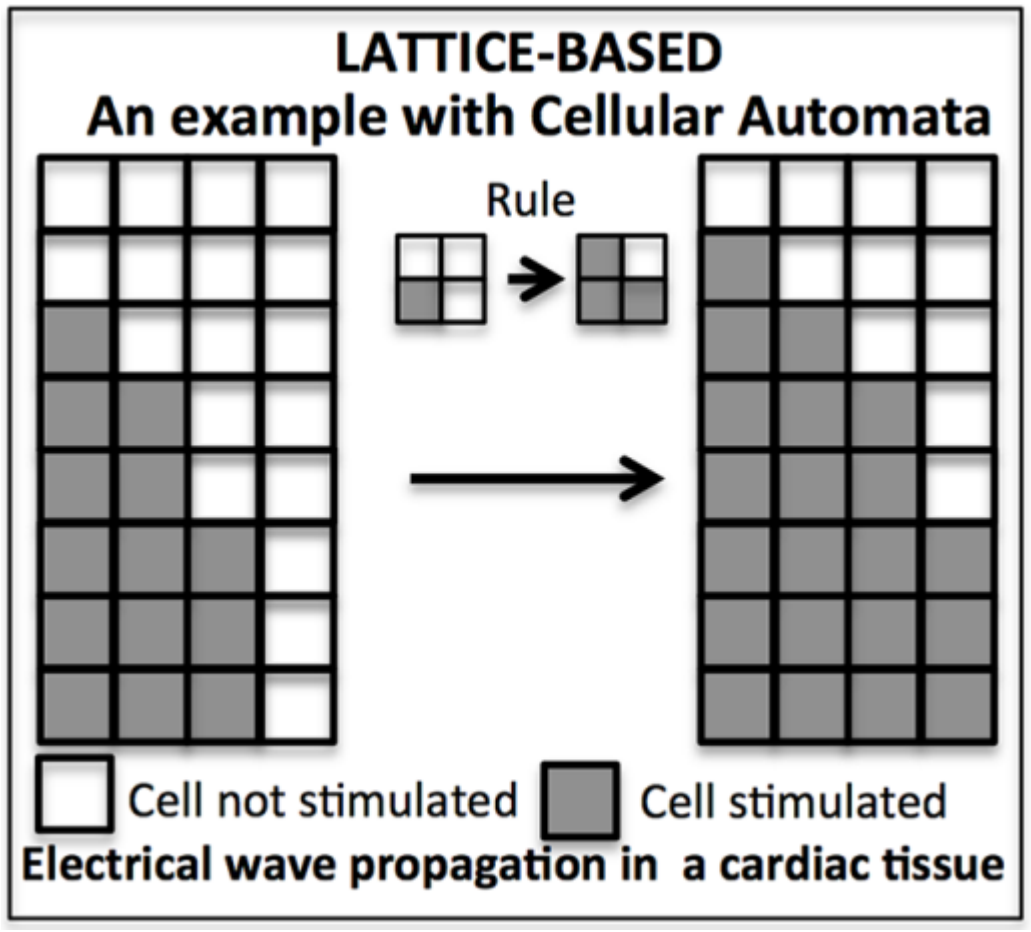


FIGURE 2.11: Latticed Based Model[50]

or organ stage. Such natural stages may be approximately linked to a microscopic (interactions and molecule motion), mesoscopic (cell motion and division, cell-cell interactions and cell-matrix), and macroscopic scale (organ and tissue mechanical attributes) respectively. Cellular automata [3] can be defined as discrete dynamic techniques, in time, space and state. Cellular pattern formation emanates from short range (for example, adhesive forces and cell-cell signalling) and long range (for example, diffusing chemicals or mechanical stress fields) interconnections. A Bethe lattice (or a Cayley tree)[51] is a hierarchically-ordered and cycle-free network having no ends which has been utilised in immunological (idiotypic) networks.

It is possible to observe, in multiscale lattice-based models occurrences at practically every scale ranging from the entire organism to the molecular stage. Nevertheless, placing items together for the purpose of acquiring true comprehension is a considerably more problematic task. This requires homogenisation and scaling up of paradigms over multiple spatial scales and

associated asymptotic methods to analyse multiple time scales. It is possible to surmount this difficulty by applying energetic considerations like the cellular Potts paradigm which is also known as the Glazier Graner- Hogeweg paradigm. This is based upon the stochastic Monte Carlo technique applied to a regular lattice.[2] The objects, which are either discrete generalised or discrete cells (clusters or cells, unicellular organisms, separate cells) or continuous fields (nutrient gradients or small molecules) are linked to an energy definition of procedures like cell-nutrient interaction and cell-cell adhesion. Lattice rearrangements induce the growth of the technique and are controlled by the reduction of the energy of a Hamiltonian function to a minimum.

CompuCell3D is a particular common structure for the advancement of a Potts paradigm [52], [53] and has been applied to model several pathological and anatomical situations at organ, tissue and cell levels. This structure is successful in amalgamating an energetic, rigorous and mechanical treatment of the procedure with a perceptive and intuitive biological definition. Interest in network ensemble approaches is increasing, and multilayer networks, especially multiple networks in various networks share the same nodes. This may be analysed by applying network entropies to quantify and assess the connections between interlinked networks; for instance, biological methods, protein, metabolite and gene networks have powerful connection as well as interdependencies which are unable to be completely depicted as single graphs.

2.5 Machine Learning

Machine learning is a sub-discipline of artificial intelligence that focuses on allowing machines to learn from past experiences, model data uncertainty, and make predictions about the future. Specifically, it is through an optimization method that learns a mapping function whose input is a feature vector of a sample and whose output is a label, this is called supervised learning. Semi-supervised learning, also often referred to as weakly supervised learning, aims to automatically learn parameters in an unlabeled sample through a small number of examples. As for unsupervised learning, people are usually more exposed to clustering problems: combining similar data into clusters by analyzing the similarity of data samples.

2.5.1 Machine learning tasks

It is common to categorise machine learning tasks into two fundamental types according to whether a learning method has a “feedback” or a "signal", and are termed Supervised and Unsupervised learning, respectivelyy.

Supervised Learning

Supervised learning is a machine learning approach that uses previously determined input-output pairs, [54] and implies a labelled training data set comprising a group of training examples[55]. Each instance in supervised learning is a pair which comprises an input object (usually a vector) as well as a desired output value (known as the supervisory signal). The purpose of a supervised learning algorithm is to analyse the training data and to create an inferred function which may be employed to plot novel examples. In an ideal situation, the algorithm will be enabled accurately to establish the class labels for unseen cases.

Unsupervised Learning

Unsupervised machine learning is the machine-learning assignment of deducing a function which defines the “unlabelled” data framework (i.e. uncategorised or unclassified data). There is no simple means of calculating the framework generated by the algorithm because the examples provided for the learning algorithm are unlabelled. Within unsupervised learning, the aim is usually to gather the data into typically varying classifications or clusters.

2.5.2 Artificial neural network

Overview

Artificial neural networks (ANNs) can be defined as computing techniques which are influenced by the biological neural networks of animal brains[56]. Techniques of this type consider examples which are not usually programmed with any particular task rules, thereby enabling them instead to “learn” how to solve problems.

ANNs are centred on a group of associated nodes or units known as artificial neurons which have similarity with the neurons within a biological brain. Each of these is able to transmit signals between artificial neurons in

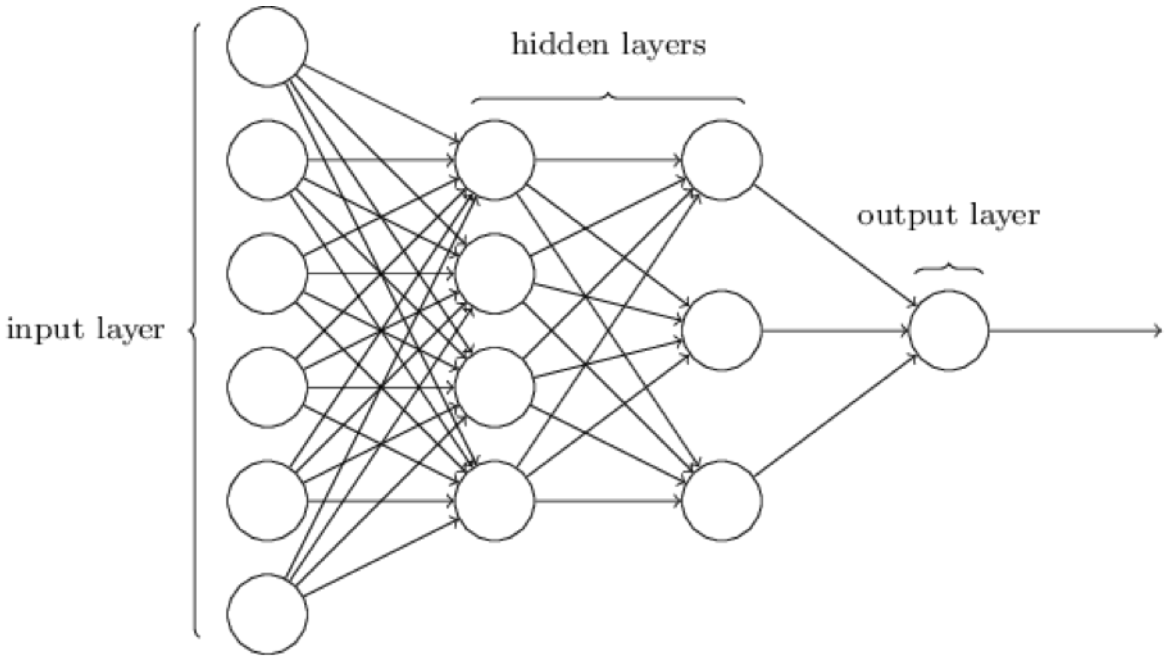


FIGURE 2.12: An ANN network[57]

the same way as the synapses in a biological brain. This obtains a signal and is able to process it and subsequently signal further artificial neurons associated with it.

The signals at connections between neurons in ANNs for general applications is a real number, and the neuron's output is calculated by a non-linear function of its inputs. These connections are known as "edges" and as learning progresses, the weight of edges and artificial neurons is adjusted. Furthermore, the signal strength at a connection is increased or reduced by the weight, and the signal not sent unless the aggregate signal passes a threshold assigned to the neuron. Artificial neurons are generally arranged into layers each undertaking a change on their inputs, from the input layer (the first) to the output layer (the last).

History

A neural network's computational paradigm is known as a threshold logic. Founded upon algorithms and mathematics, it was created by Warren McCulloch and Walter Pitts[58] (1943). This paradigm inspired neural network research into two broad methods, one based upon biological processes inside the brain, and the other based upon the implementation of neural networks to artificial intelligence.

Hebbian Learning During the late 1940s, D.O. Hebb[59] produced a learning hypothesis centred upon a neural plasticity mechanism called "Hebbian learning", unsupervised learning developed into paradigms for long-term potentiation, a persistent strengthening of synapses based on recent patterns of activity. In 1948, researchers began to apply these concepts to computational paradigms with Turing's B-type machines. In order to induce a Hebbian network, computational machines, at that time known as "calculators", were first utilised by Farley and Clark[60] (1954). Furthermore, Rochester, Holland, Habit and Duda (1956)[61] produced additional neural network computational machines. An algorithm for pattern recognition known as the perceptron was produced by Rosenblatt[62] (1958), who with mathematical notation, defined circuitry which was not in the basic perceptron; for example, the exclusive-or circuit which, at that time, was incapable of being processed by neural networks [63].

Nobel laureates, Hubel and Wiesel advocated in 1959 a paradigm founded upon their discovery of two cell types within the primary visual cortex, namely single and complex cells [64], and in 1965, Ivakhnenko and Lapa published the initial functional networks having several layers which became the Group Method of Data Handling[65].

After Minsky and Papert (1969)[66] conducted machine-learning research, neural network research came to a standstill. They discovered two principal difficulties regarding computational machines which processed neural networks. The first of these was the inability of basic perceptrons to process the exclusive-or circuit, and the second was that computers had insufficient processing power for efficient management of the work needed by large neural networks. Consequently, neural network research was delayed until the advent of computers with considerably greater processing power.

Backpropagation Werbos's (1975) backpropagation algorithm was a principal trigger in the revival of interest in neural networks and learning. This essentially resolved the exclusive-or problem and resulted in a general increase in multilayer networks. Backpropagation redistributed the error term back through the layers, by adjusting the weights at each node[63]. Furthermore, parallel distributed processing became popular during the mid 1980s with the term "connectionism", whose utilisation in inducing neural processes was defined by Rumelhart and McClelland (1986)[67]. Neural

networks in machine learning gradually lost popularity to more basic systems like linear classifiers and support vector machines. Nevertheless, the application of neural networks changed certain areas, for example, the forecasting of protein frameworks [68].

Max-pooling was implemented in 1992 in order to assist least shift invariance and deformation tolerance in helping 3D object identification [69]. Backpropagation training through max-pooling was in 2010 accelerated by GPUs and discovered to give a superior performance to that of other pooling variants [70]. The many-layered feedforward networks which utilised backpropagation as well as recurrent neural networks (RNNs) were influenced by the disappearing gradient issue [71]. As errors proliferate from one layer to another, they experience an exponential decrease according to the number of layers, thereby hindering the neuron weights' tuning, based upon such errors, specifically influencing profound networks.

Schmidhuber (1992) applied a multilayer network hierarchy pre-trained by one level in turn, fine-tuned by backpropagation and by unsupervised learning in order to resolve this issue [72]. However, with issues like face localisation and image reconstruction, Behnke (2003) relied exclusively upon the gradient sign (Rprop)[73]. Hinton et al. (2006) advocated the application of successive layers of binary or real-valued latent variables with a restricted Boltzmann machine to model each layer in order to learn a high-level representation [73]. When an adequate number of layers has been learned, the profound architecture may be applied as a generative paradigm by sampling down the paradigm (an "ancestral pass") from the top-level feature activations [74]. Ng and Dean produced a network in 2012 which learned to identify higher-level subjects, such as cats, exclusively from viewing unlabelled pictures on YouTube.[75]

Recurrent neural networks Recurrent Neural Networks (RNNs) are neural networks in which it is possible for outputs from neurons to feed back into its input. Hence, information cycles through a loop in the RNN process, and when a decision has been reached, it considers the current input as well as that that has been learned from the inputs previously obtained. Figure 2.13 depicts the variation within the information flow between a Feed-Forward Neural Network and an RNN.

It is normal for an RNN to possess a short-term memory, but when coordinated with Long short-term memory (LSTM), it can also possess a long-term

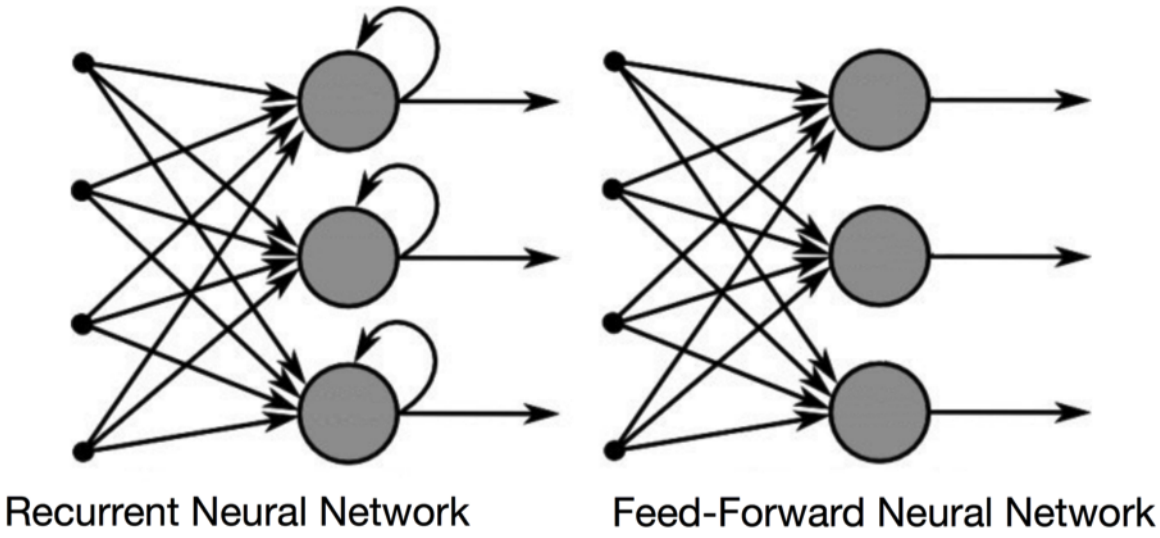


FIGURE 2.13: Difference between ANN and RNN[76]

memory - a matter which will be considered later. An RNN attaches the immediate past to the present; consequently, it has two inputs: present and recent past. This is significant since the data sequence comprises critical information regarding what is to follow, being the reason why an RNN can perform tasks which other algorithms cannot.

2.5.3 Principal component analysis

Principal Component Analysis (PCA) is a technique used to identify a small number of unrelated variables (called principal components) from a large data set. This technique is widely used to emphasize mutation and capture strong patterns in data sets. PCA, invented by Karl Pearson in 1901[77], is a tool used in predictive models and exploratory data analysis.

In statistics, PCA is used to simplify data sets by linear transformation. This transforms the data into a new coordinate system so that the first large variance of any data projection is the first coordinate (called the first principal component) and the second largest variance, the second coordinate (the second principal component), and so forth. Principal component analysis is often used to reduce the dimensionality of the data set, while maintaining

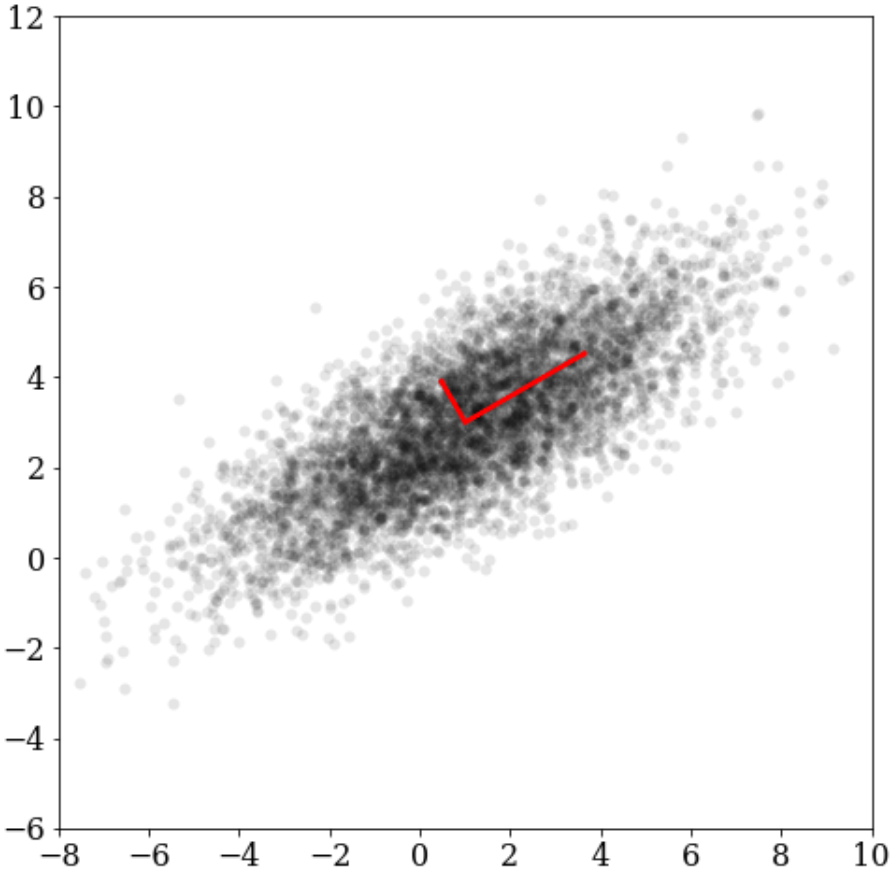


FIGURE 2.14: An example of PCA where the two arrows indicate the highest variance[78]

the features that the variance of the data set contributes the most (see Figure 2.14).

2.5.4 Support Vector Machines

Support Vector Machines (SVMs) as currently used were popularised in 1995[78] as they demonstrated excellent performance in text classification tasks[79], and quickly became the mainstream of machine learning in the 2000s. The theory of SVMs originated in the early 1960s, and the concept of statistical learning theory was established in the 1990s. The technique has been rapidly developed since and a series of improved and extended algorithms have since been derived for application to pattern recognition

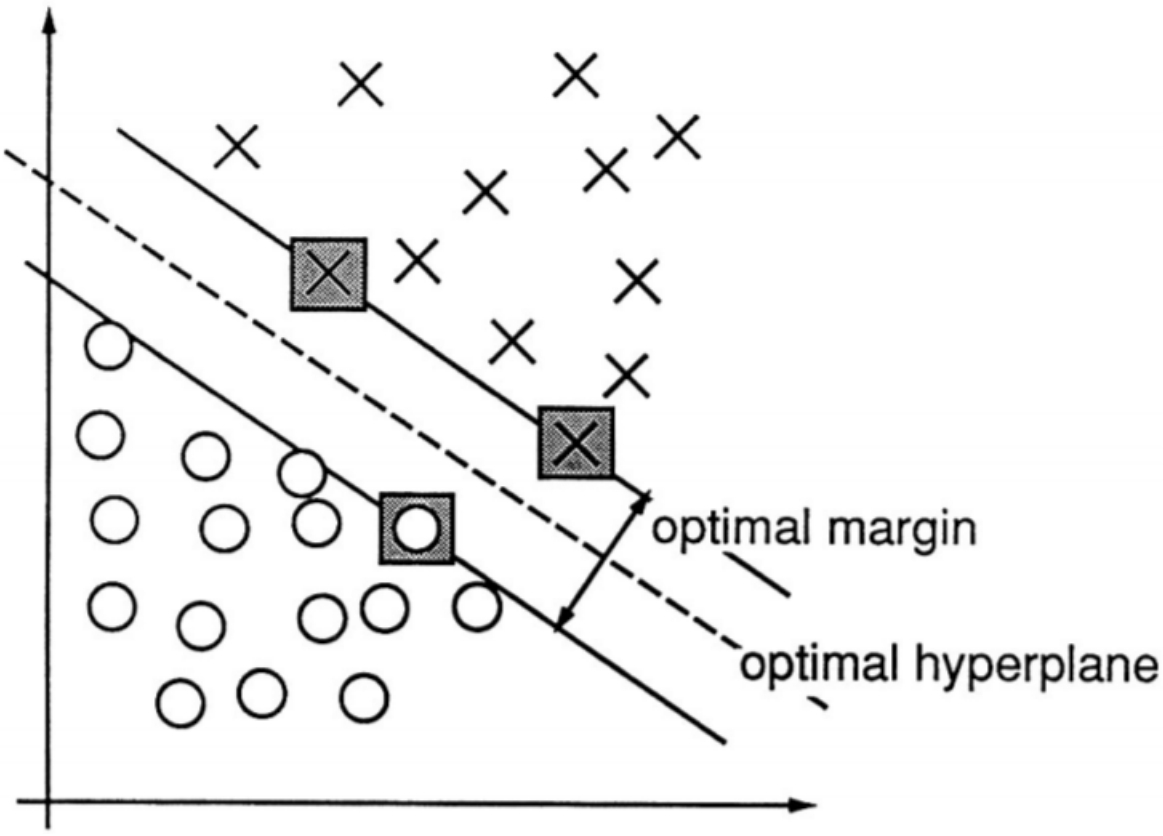


FIGURE 2.15: An example of a classification problem with support vectors marked with gray squares[78]

problems such as portrait recognition and text classification[80], [81]. Figure 2.15 shows an example of SVM where the hyperplane is defined by support vectors.

2.5.5 Evolutionary algorithms

Evolutionary Algorithms (EAs) are often effective in deriving solutions for a wide range of problems as they require no prior information about the underlying fitness landscape. However, when biological evolution is modelled, EAs are usually restricted to exploring microevolutionary processes or planning models based on cellular processes. Moreover, computational complexity is a limiting feature of most real-world applications of EAs. Fitness function evaluation is the reason for such complexity. One way to resolve this difficulty is to use fitness approximation. However, even simple EAs can solve very complex problems. This suggests that, in terms of complexity, there is a direct link between algorithms and problems.

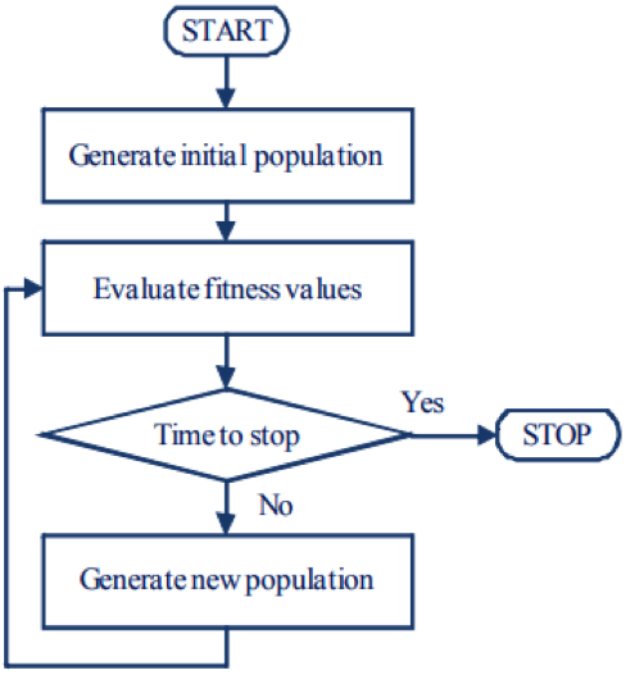


FIGURE 2.16: General form of EAs

Typical EAs in pseudo-code

Evolutionary algorithms involve three primary processes. Initialisation is the first process and involves the random generation of an initial population of individuals in line with a representation of a solution. Directly or indirectly, each individual represents a potential solution. In an indirect representation, every individual must be decoded into a solution. In the second process, each solution is then evaluated in terms of its fitness value. To facilitate selection, these values are then used to calculate the average population fitness or rank individual solutions within the population. The third process involves generating a new population based on solutions in the existing population. These three key processes are presented in Figure 2.16.

As Figure 2.16 shows, an evaluation of the fitness of the population follows initialisation after which the stopping criteria are evaluated. If none of the criteria are met, a new population is generated and the process continues until one or more of the criteria are satisfied. A stopping criterion can be either static or dynamic. A static criterion will allow an algorithm to execute a fixed number of iterations. A dynamic stopping criterion, on the other

hand, will continue the process until the condition is satisfied, for example, a predetermined number of the solutions lies within a given threshold of the optimal solutions derived. Sometimes several stopping criteria are required. The first and most important task when using an evolutionary algorithm to resolve optimisation problems is to determine how to represent the solution given the characteristics of the algorithm. The initialisation and generation of a new population may result in solutions that are not feasible. Therefore, it is essential that a direct or indirect solution representation is chosen that will yield feasible solutions. This is a standard design consideration for all evolutionary algorithms. The primary consideration in terms of design, therefore, is to ensure that it is possible to decode every individual into a feasible solution. In conjunction with a decoding procedure, an indirect representation is often used in complex problems to transform the solution representation into a feasible solution. The fitness function can then be evaluated once the solution is decoded. Two common parameters that must be determined first are the maximum number of iterations and the size of the population. The values chosen for these two parameters have a significant influence on the quality of the solution and the time taken; in practice, these values are usually determined empirically through experimentation.

2.5.6 Genetic Programming

Overview

Genetic programming (GP) may be defined as an evolutionary computation (EC) method which resolves problems automatically without any necessity for the user to be aware of or to particularise in advance the type or framework of the solution. At its most abstract stage, GP is a domain-independent, systematic technique for using computers to resolve difficulties automatically beginning from a high-level statement of requirements. GP discovers the efficiency of a program's operation by operating and comparing its behaviour to an ideal. For instance, there may be interest in the efficiency of a program which controls an industrial procedure or predicts a time series, which comparison is quantified to provide a numeric fitness value. Programs which give a good performance are selected to breed and subsequently provide new programs for the coming generation. The initial genetic operations which are applied for the creation of new programs from current ones are mutation and crossover.

History

It is probable that the initial instance of the suggestion to develop programs is that designed by Alan Turing [82] during the 1950s. Nevertheless, about three decades passed until Richard Forsyth [83] showed the successful growth of small programs which were compared to trees to undertake categorisation of evidence of crime scenes for the UK Home Office.

John Koza patented his GA invention for the growth of a program evolution in 1988 [84] which was succeeded by being published in the International Joint Conference on Artificial Intelligence IJCAI-89. [85], [86] Subsequently, Koza produced 205 "Genetic Programming" (GP) publications, being a name adopted by David Goldberg, who was also one of John Holland's PhD students. [87] Nevertheless, genetic programming was really established in 1992 when Koza began his series of four books [88] with accompanying videos.[89] A tremendous growth of publications followed as the genetic programming bibliography exceeded 10,000 entries.[90]. Koza listed 77 instances in 2010 [91] in which genetic programming was human competitive.

Koza inaugurated the annual Genetic Programming Conference in 1996.[92] This was followed by the annual EuroGP Conference in 1998 [93], together with the first book[94] in a GP series which Koza edited; furthermore, the first GP text- book was published in 1998. [93] GP continued to thrive, resulting in the publication of the first specialist GP journal.[95] After another three years, the annual Genetic Programming Theory and Practice (GPTP) workshop was established by Rick Riolo in 2003. [96]There was continuing publication of genetic programming papers at several conferences and in similar journals. There are currently 19 GP books which include many specifically written for students.[94]

Implementation

In this section, an introduction is provided on how trial solutions are represented in the majority of GP techniques, how an initial random population may be constructed, and how selection, mutation and crossover are applied to prepare new programs.

Representation It is usual for GP programs to be shown as syntax trees instead of code lines; for instance, Figure 2.17 depicts the tree representation

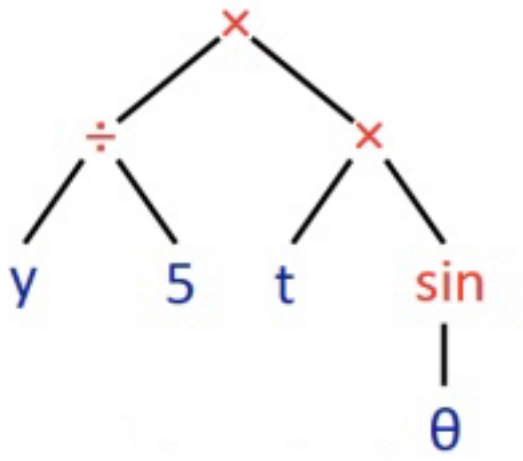


FIGURE 2.17: GP syntax tree representing $(y/5) \times (t \times sin(\theta)$

of the program $(y/5) \times (t \times sin(\theta)$.

In the program, 3, x and y, the constant and variables, are represented by the tree's leaves. In GP, these are known as terminals, whereas the internal nodes arithmetical operations (+, * and max) are referred to as functions. The groups of permitted terminals and functions jointly comprise a GP system's primitive set.

In more advanced GP formats, programs may comprise multiple components, in which case the representation applied to GP is a series of trees united under a specific root node which serves as "glue". These "sub-trees", referred to as branches, definee the program's architecture.

Initialising the Population In GP, the individuals in the first population are usually randomly generated as in other evolutionary algorithms; however, it is not necessary for the initial population to be random. If there is some knowledge concerning potential attributes of the solution which is desired, trees having these attributes may be utilised to seed the first population to speed evolution.

Selection Genetic operators in GP apply to individuals that are probably chosen on the basis of fitness, as in the majority of evolutionary algorithms. This implies that better individuals have a greater potential of having a

greater number of child programs than inferior individuals do. In GP, the most frequently adopted technique for choosing individuals in GP is selection by tournament. This is considered in the following section, and fitness proportionate selection is subsequent; however, any basic evolutionary algorithm selection mechanism may be applied.

In tournament selection, a certain number of individuals is selected randomly. After comparison, the best individual is selected as the parent, but utilisation of the crossover requires two parents; therefore, there are two selection tournaments. It should be noted that tournament selection looks only at the better program, but it is not necessary to be aware of which one is better. In practice, this rescales fitness automatically, resulting in constant selection pressure on the population. Therefore, it is not possible for one exceptionally good program to swamp the subsequent generation's offspring. However, if this happened, it would result in a rapid loss of diversity with a likelihood of poor subsequent performance. Contrastingly, small differences in fitness tournament selection are amplified, resulting in a preference for a better program, even if it is only slightly better when compared with other persons in a tournament. There is a degree of noise in tournament selection as a result of the random choice of candidates. Therefore, although the best individual is preferred, there is no guarantee in tournament selection that even programs which are of average quality have likelihood of having offspring. As it is not difficult to introduce tournament selection and this supplies automatic fitness rescaling, it is frequently applied in GP. Given that choice has often been defined in the literature on evolutionary algorithms.

Recombination and Mutation Since its introduction, of the operators of mutation and crossover GP differs considerably from other evolutionary algorithms, the type which utilised most frequently being the subtree crossover. Where there are two parents, the subtree chooses a crossover point (a node) in each of the parent trees independently and randomly. Subsequently, it exchanges the respective subtrees in order to generate the offspring. The subtree is grounded at the crossover point with a copy of the initial parent, together with a subtree copy which is grounded at the crossover point in the second parent, as shown in figure 2.19.

Frequently, the choice of crossover points has no uniform probability. A characteristic primitive GP group results in trees having only an average

branching factor, being the number of offspring of each of the nodes, having a minimum of two in order that most of the nodes will be leaves. Therefore, the uniform choice of crossover points results in crossover operations which exchange minute amounts of genetic material such as small subtrees. In fact, several crossovers may decrease or reduce simply to exchange two leaves. In order to balance this, Koza (1992) advocated the extensively applied method of selecting functions for 90 percent of the time and leaves for 10 percent of the time. There are several additional types of mutation and crossover of GP trees.

The most frequently applied mutation type in GP (otherwise called subtree mutation) makes a random selection of a mutation point within a tree and subsequently replaces the subtree which is grounded there with a randomly created subtree, as shown in Figure 2.18. Occasionally, subtree mutation is introduced as a crossover between a newly-created random program and another program, a process otherwise referred to as "headless chicken" crossover [97].

Point mutation is a further well-known type of mutation, being GP's approximate bit-flip mutation equivalent as utilised in genetic algorithms [86]. In this mutation type, a random node is chosen which substitutes the primitive which is stored there with another random primitive having the same arity as the primitive group. If no other primitives are connected with that arity, then that node is unaffected, although other nodes may remain mutated. Subtree mutation utilisation concerns the modification of just a single subtree, whereas from another perspective, point mutation is normally used on a basis that is per node, meaning each node being regarded consecutively, and with some level of probability, is adjusted as aforementioned. This permits the independent mutation of multiple nodes in one point mutation application, the selection of which the aforementioned operators ought to be utilised to generate a probabilistic offspring. It is usual for operators in GP to be mutually exclusive, which does not resemble other evolutionary algorithms in which offspring are occasionally acquired through a composition of operators, in which case the probability application is known as operator rates. It is usual for crossover to be used with the greatest probability, with the crossover rate frequently being at least 90 percent. Contrastingly, the mutation rate is considerably less, usually being around one percent. Furthermore, where the total of the rates of mutation and crossover

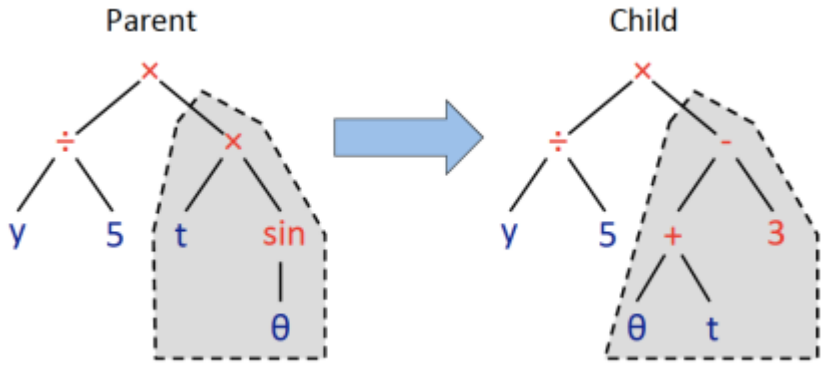


FIGURE 2.18: Subtree mutation example[98]

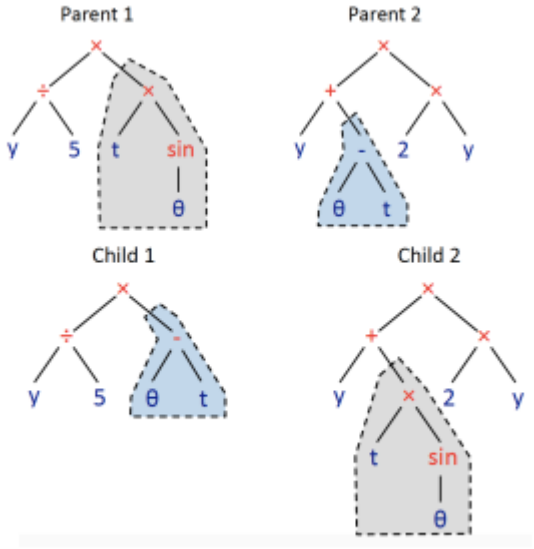


FIGURE 2.19: GP crossover example[98]

reach a p value below 100 percent, an operator known as reproduction is applied with a 1-p rate. Basically, replication concerns an individual's choice on the basis of fitness as well as the inclusion of a copy of this in the succeeding generation.

2.5.7 Cartesian Genetic Programming

Overview

This type of programming progressed from a technique of evolving digital circuits which was developed in 1997, by Miller et al. [99] Nevertheless, the

expression "Cartesian genetic programming" initially arose in 1999, [100] and in 2004 was advocated as a common form of genetic programming. [101] The reason for naming it "Cartesian" is that it represents a program which uses a two-dimensional grid of nodes.

Implemention

Representation Programs in CGP adopt a directed acyclic graph format with a two-dimensional grid of computational nodes. The genes which comprise the genotype in CGP are integers which indicate from where each node obtains its data (inputs), the operation performed by each node upon that data and from where the output to the program can be acquired. Upon the decoding of the genotype, some nodes may be disregarded, a situation which occurs when there is no application of nodes in the computation of output data. In this situation, the nodes and their genes are known as "non-coding". The program resulting from the genotype decoding is known as a phenotype, which in the case of CGP is of a set length. Nevertheless, the phenotype size with regard to the number of computational nodes is able to occupy a range from zero to the number of nodes which are defined in the genotype. If every program output was directly linked to the program inputs, a phenotype would possess zero nodes. Furthermore, when every node in the graph was needed, a phenotype would possess an identical number of nodes as described in the genotype, and a defining characteristic of this is the genotype–phenotype mapping which is utilised in CGP.

The user determines the computational node functions to be used which are listed in a function look-up table. In CGP, every node within the directed graph is encoded by a number of genes and represents a specific function. One gene, known as the function gene, is the address of the node function's operator within the function set look-up table. The remaining node genes determine from which other nodes the current node obtains its data. Nodes receive their inputs in a feed-forward method from the node output in a previous column of the Cartesian two-dimensional graph or from a program input. The number of a node's connection genes is determined as the maximum number of inputs, frequently referred to as the arity, which is possessed by any function within the look-up table. Input node addresses range from zero to ni-1, in which ni is the number of program inputs, are provided by the program data inputs. Addresses are given to the data outputs of the nodes in the genotype. Such addresses are provided sequentially,

column by column, beginning from ni to ni + Ln - 1, in which Ln is the user-determined higher bound of the number of nodes. Figure 2.20 depicts the basic format of a Cartesian genetic program.

The output(s) of the program are indicated by the final gene(s) of the genotype. Generally a number of output genes (Oi) define the program outputs, of which each is a node address from where the output data of the program is obtained. It is nor permitted for nodes within the same columns to be linked with one another, since the graph is feed-forward and directed; the inputs of a node may be linked either to input nodes or to another node's output as long as it resides in a previous column, as shown in Figure 2.21. Every node function fi gene is an integer address in a look-up function table. All connection genes are Ci,j data addresses, assuming values between zero and the node's address is at the base of the previous nodes' column. Three parameters need to be predetermined which are: the number of rows and columns in the Cartesian representation and the levels-back parameter which signifies the number of preceding columns any node may seek data to fulfil its inputs, respectively represented by nc, nr and l. The maximum number of computational nodes permitted (Ln = ncnr) is determined by the product of the first two parameters (rows and columns), and the connectivity of the encoded graph is controlled by the levels-back parameter. The levels-back parameter constrains from which column a node may obtain its inputs. If I = 1, a node may obtain its inputs exclusively from a primary input or from another node within the column immediately to its left. If l = 2, the inputs of a node may be linked to the outputs of any of the nodes in a primary input or to the immediate two left columns. If there is a desire to permit nodes to link to any other nodes located to their left, then l = nc (levels-back is set to the number of columns). If such parameters are varied, then many types of graph topologies may result. When a small number of columns is selected and a great number of rows is selected, the resulting graphs will be thin and tall. Selecting a large number of columns and a small number of rows will result in wide, short graphs. The selection of levels-back generates graphs which are highly-layered in which computations are performed column by column. One special case is significant - this concerns the situation of these three parameters when the number of rows is selected to be one and levels-back is set to be the number of columns. In this situation, the genotype may indicate any directed bounded graph in which the number of columns determines the upper bound.

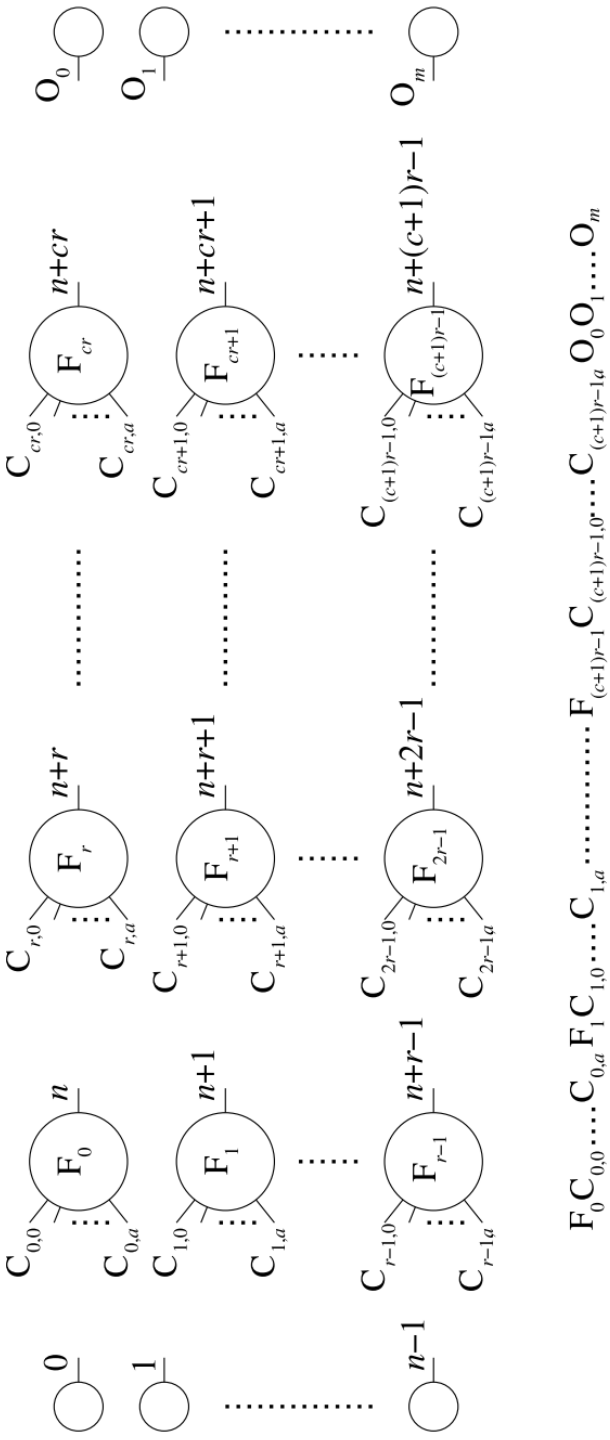


FIGURE 2.20: General form of CGP. A grid of nodes whose functions are chosen from a set of primitive functions. The grid has nc columns and nr rows. The number of program inputs is ni and the number of program outputs is no. Each node is assumed to take as many inputs as the maximum function arity a. Every data input and node output is labeled consecutively (starting at 0), which gives it a unique data address which specifies where the input data or node output value can be accessed (shown in the figure on the outputs of inputs and nodes).[101]

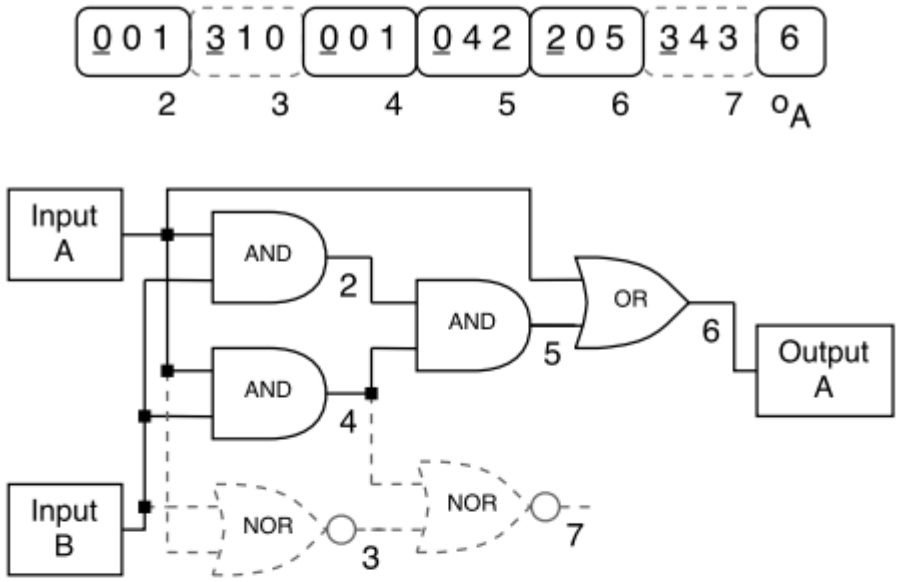
Initialising the Population The initial population is normally populated with randomly-generated chromosomes in the same way as in other evolutionary algorithms. The user defines the number of program inputs and outputs, internal nodes and the arity of each node. The chromosomes are then generated randomly by applying these given restrictions. For example, each specified chromosome node will be provided with a random function gene from the available function set, and each input of each node will be provided a random connection gene. Lastly, each output gene will be set as a randomly-selected program input or node within the graph.

Recombination and Mutation It is usual for CGP to apply only mutation in generating the children from the chosen parents, by utilising probabilistic mutation operators. In CGP a point mutation operator is commonly adopted in which a point mutation, an allele at a randomly chosen gene location is transformed into another valid random value. Where a function gene is selected for mutation, a valid value is the address of any function in the set. For an input gene, a valid value is the output address of the output of any program input or of any previous node in the genotype. Finally, a valid value for a program output gene is the address of a program input or the output address of any node in the genotype. The user determines the number of genes in the genotype that can be mutated each generation, normally a probability referred to as the mutation rate, represented by the symbol μ_r . The actual number of gene sites which can be mutated in the genotype of a given length L_g, is μ_g, such that$\mu_g = \mu_r \ L_g$.

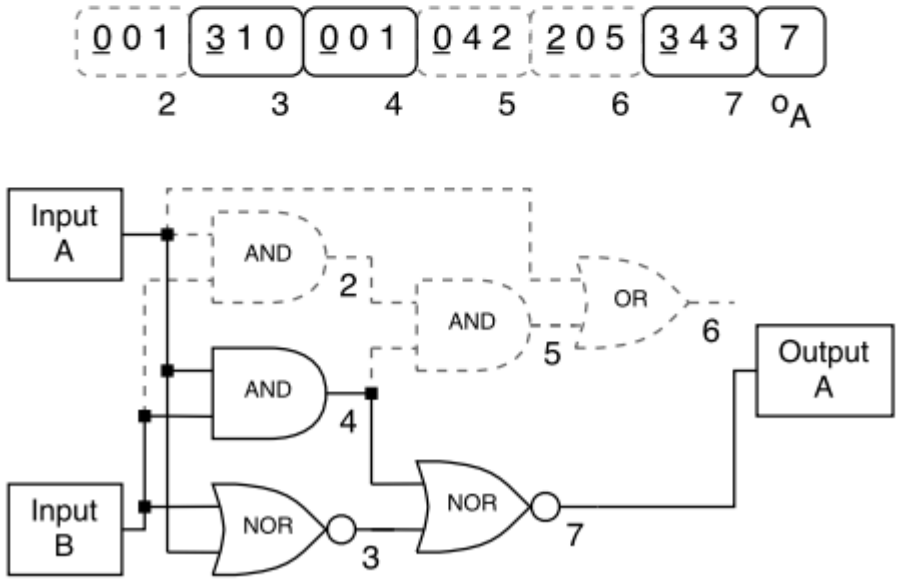
2.5.8 Advantages of Cartesian Genetic Programming

Despite the fact that CGP has not been adopted to the same degree as the more popular tree-based GP [102], it has many beneficial attributes which makes it an attractive alternative:

- CGP does not experience program bloat.
- CGP is well suited to promoting neutral genetic drift.
- CGP is naturally supports multiple input, multiple output tasks.
- CGP permits internally-created sub-structures to be reused.



(a) Before mutation



(b) After mutation

FIGURE 2.21: An example of the point mutation operator before and after it is applied to a CGP genotype, and the corresponding phenotypes. A single point mutation occurs in the program output gene (OA), changing the value from 6 to 7. This causes nodes 3 and 7 to become active, whilst making nodes 2, 5 and 6 inactive. The inactive areas are shown in grey dashes.[101]

No bloat

Bloat is defined as "program growth with no significant return regarding fitness"[103]. Bloat is a serious matter for tree-based GP [104] because it frequently leads to programs of a very large size which is computationally inefficient, unless it is actively prevented by imposing restrictions or fitness penalisation on program size. CGP possesses a natural resilience to bloat as it has a defined maximum number of nodes that can be used[105], therefore the entire matter of bloat does not arise. However, more precisely, CGP utilises a fixed-sized genotype in which nodes can be active or inactive in the phenotype; therefore the phenotype size can increase or decrease. Despite this being not usually considered for tree-based GP (as the phenotype and genotype are practically equivalent), the phenotype bloat is a matter of concern. Consequently, there is a likelihood that bloat does occur in CGP, which would appear as an increase in the number of active nodes, or equivalently the phenotype size.

Heightened Neutral Genetic Drift

The encoding adopted by CGP normally results in a high proportion of the genes being inactive,[106] thereby allowing a considerable amount of neutral generic drift to occur. Since neutral generic drift is considered to be advantageous to the evolutionary search[107], [108], thereby permitting easier escape from local optima, this is considered to be a principal benefit of CGP.

Multiple-Input Multiple Output

Tree-based GP is usually only able to encode problems with multiple inputs and a single output, but CGP architecture permits Multiple-Input Multiple Output. Furthermore, since CGP involves directed graphs, it is not restricted by a tree structure whilst still having the ability to evolve tree structures when the evolutionary pressure is applied to achieve this.

Reuse of Internally Created Sub-Structures

During the evolution of tree-based GP, it is impossible for a given node's output to be used by more than one other node, which is a significant restriction if the same sub-structure functionality is required on multiple occasions to generate an efficient solution. For example, consider a GP function set with neither trigonometric functions, nor with the value of π. If an

approximate value of π is found, this will probably be advantageous to the evolutionary search and utilised through the course of the evolved program. If, in tree-based GP, a value of π wad needed on multiple occasions, then it needs to be rediscovered each time because an internally-computed value can be used only once. However, CGP permits the outputs of any node to be utilised by any other node in the program on the condition that it abides by the rules of acyclic connectivity, meaning that if π were discovered, the same value could be utilised on multiple occasions.[109]

Applications

Despite the fact that CGP was initially developed to implement and optimise digital circuits[95], [110], it has subsequently been utilised in other domains. For example, EAs can be utilised as an optimiser in combination with a feature extractor. The Cartesian representation of CGP and also the predefined number of inputs to the network makes it particularly suitable to process raw data values or previously extracted features.[109]

Furthermore, CGP has been used for image processing [111] in which it has been proved to be a particularly strong method by which the function set comprises image processing functions which included domain knowledge in the search. This method was referred to as CGP Image Processing and it has been successfully applied to edge detection[112], image filtering[101], scale and rotation, medical imaging as well as real time object detection with varying lighting, This has included several medical applications such as classification of mammograms containing benign or malignant tumours[4], [113], [114] as well as the assessment of Alzheimer's disease.

CGP's basic and flexible directed graph framework supports implicit modularity and also averts bloat, thereby providing a clear advantage over traditional GP. Furthermore, the representation is suitable for customisation to an implicit context representation which surmounts the positional dependence of the encoding of CGPs encoding. The capability of defining the network's geometry permits solutions to be customised in a way that is advantageous if there is need for a real-time or hardware implementation. In addition to creating high-performance classifiers, the capability of decoding the network as a discrete mathematical expression provides insight to the system under evaluation.

2.6 Conclusion

This chapter has detailed cell and cell culture along with their roles in biological research, this chapter then introduced urothelium, a kind of cell form urinary tract which is a stable and self-repairing tissue. Two technologies used in characterize cell culture behavior then presented which are cell tracking software and cell modeling, cell tracking softwares provided accurate means of measurement of cells and modelings gives way to understand cell behaviors. Last section of this literature review is machine learning, a method of data analysis that automates analytical model building, this section also gives reason why machine learning is a novel method for understanding cell behaviors.

Chapter 3

Methodology

Contents

3.1 Introduction

This chapter considers the methodology adopted for the experimental investigations of this work. The first stage is data acquisition including time-lapse video capture and cell tracking. The following stage is the data prepossessing and extraction of features which are chosen based on the underlying biology scientific questions; section 3.4 presents biological motivation for the features extracted and how they are calculated. After consideration of data visualization and preparation for analysis, the classification of the acquired data is considered using Principal Component Analysis (section 3.7), Support Vector Machines (section 3.8) and finally, Evolutionary Algorithms (section 3.7).

3.2 Data acquisition

The analysis of cell-motion within cell cultures requires capturing images of the culture at regular time intervals using time-lapse spectroscopy. Custom-written software is then used to analyse the resultant images as described below.

3.2.1 Video acquisition

Normal Human Urothelial (NHU) cells were cultured in a serum-free medium, as described in [33]. The cultures were seeded within twelve well plates were open to $100\mu M$ PPADs ($pyridoxalphosphate-6-azophenyl-2\prime,4\prime-disulphonicacid$) or 0.1% DMSO (as vehicle control) for 10 min, before addition of 0, 10 or $50\mu M$ ATP ATP by repeating this addition four times. Cultures were examined by applying a fourfold target by differential interference as compared with videomicroscopy (Olympus IX81 microscope) within an environmental chamber at an automated mechanical level. Time-lapse videos composed from separate images were captured digitally at five-minute intervals during a time span of 24 hours. Figure 3.1 depicts a basic frame of one video of this type.

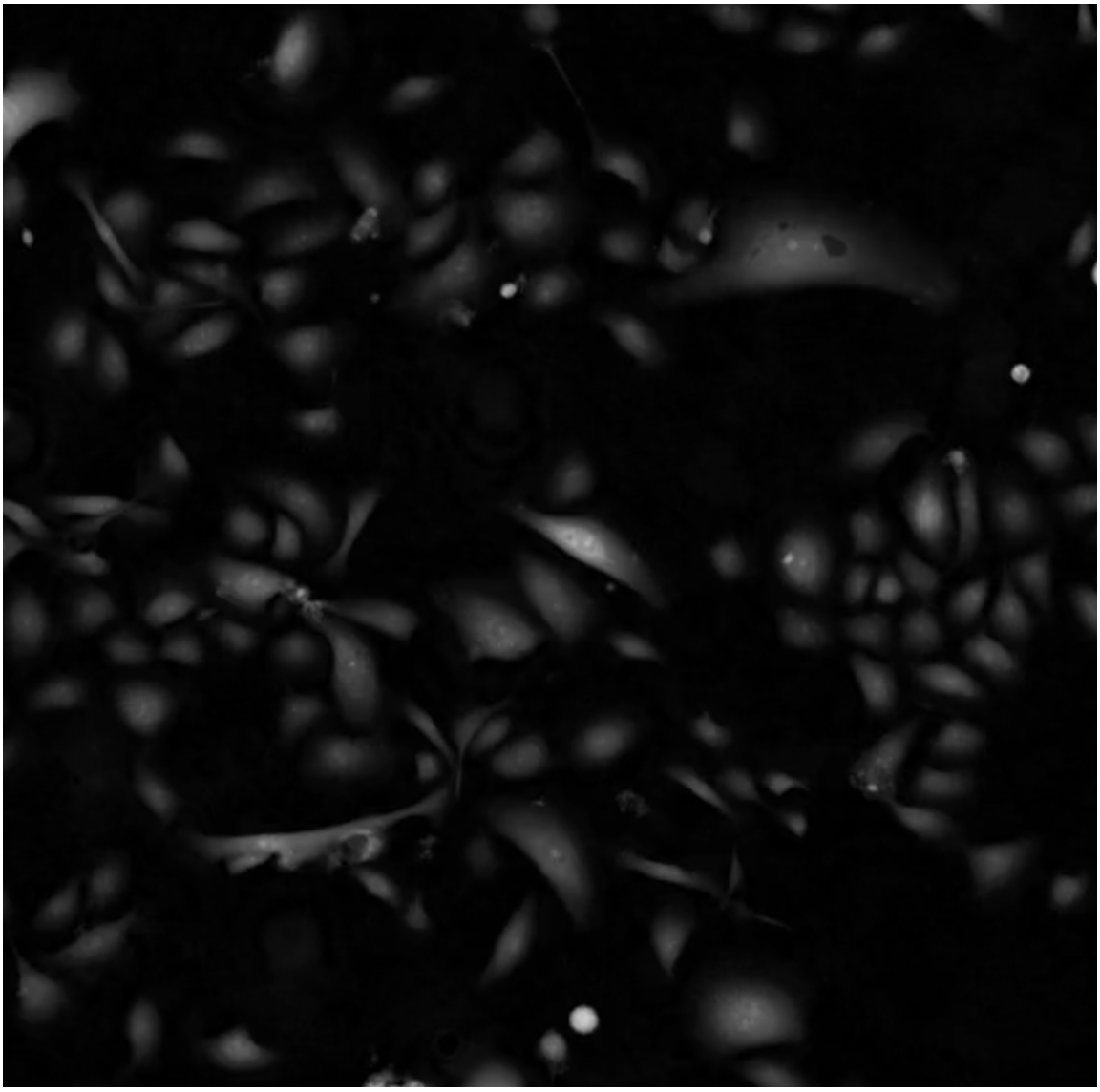


FIGURE 3.1: Sample frame from time-lapse video of NHU cells in culture.

3.2.2 Cell tracking software

Using the OpenCV computer vision programming library, custom software was developed for automated cell tracking.[37]. In order to track the relative movement of cells from frame to frame, each video time points undergoes processing to identify the likely locations of cells. This process takes the raw videos as an input, performs image pre-processing to each frame, and then either tries to identify the possible new cells, or track the location of previous time point located cells. This process was explained in section 2.4.1, as previously reported by the authors[115].

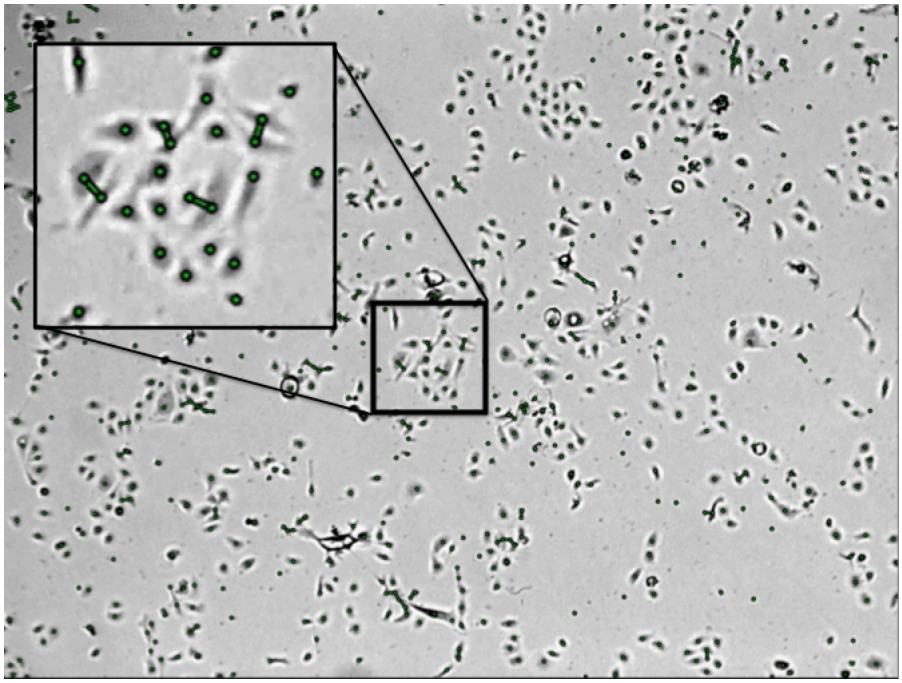


FIGURE 3.2: Example frame showing how ctracker tracks cells

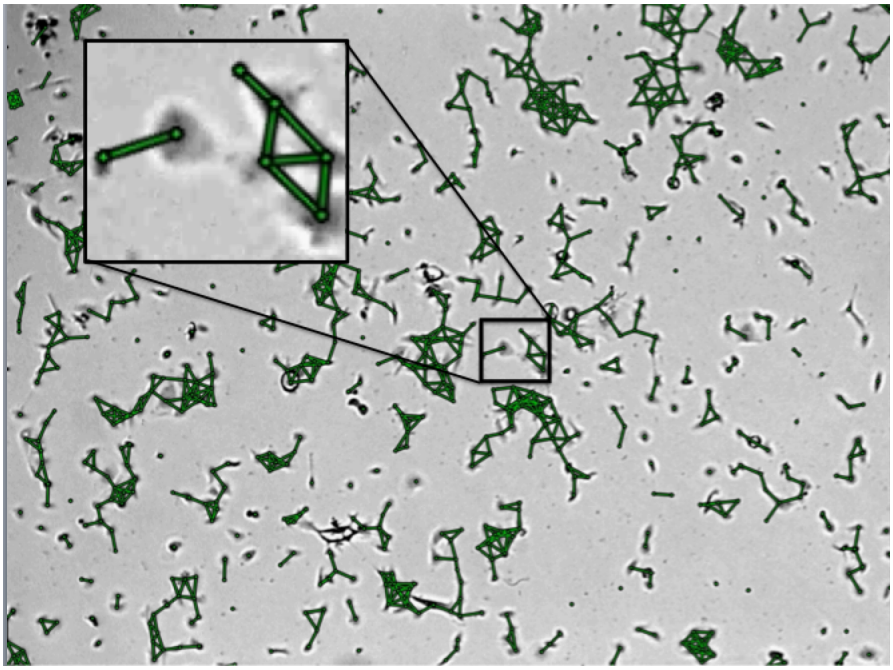


FIGURE 3.3: There is a link between cells when there distence between two cells under parameter L

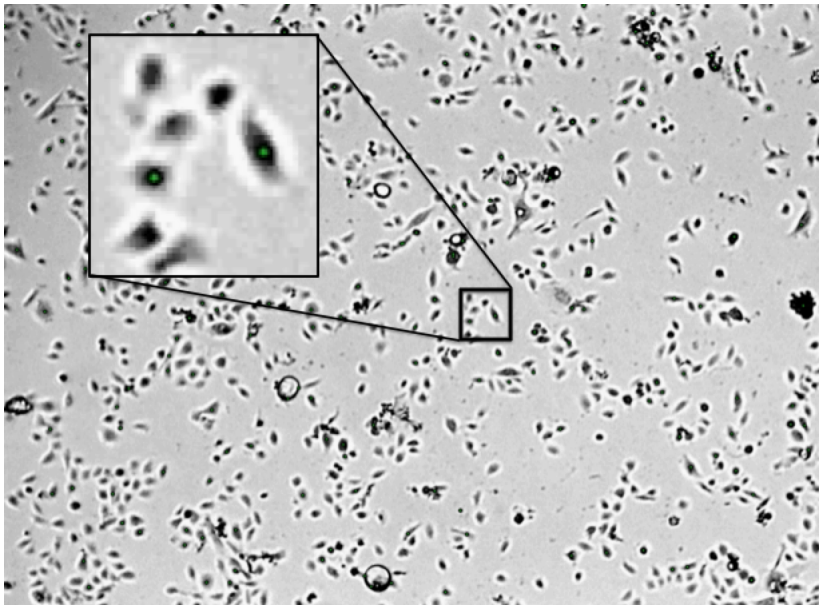


FIGURE 3.4: Example Frame with large Regen, where a lot of cells are losing tracking

Parameters for tracking software

Several parameters have been defined to characterise the software's operation. However, in order to obtain optimal performance and information from the videos, it is necessary to understand the nature and function of these.

Line Length

Line length sets the maximum distance between two points through which a straight line can be drawn. As depicted in figure 3.3, when the distance between two cells is less than 25 pixels, a green link between them is shown. This parameter affects only the video which is generated by the software and not the data file which contains the cell coordinates. Therefore, the generated video can provide some subjective information about the distances between cells in terms of pixels. The video generated by Ctracker is only for illustrating cell tracking and does not affect the cell tracking result nor analysis.

Invert

Depending on the microscope imaging technique adopted, cells can appear with differing colourings in images. There are two main types, the first is

	Less than 20 frames Tracked	20 - 40 Frames	40 - 80 Frames	80 - 150 frames	More than 150 Frames
With Regen 500	964	166	320	184	362
With Regen 60	4536	524	916	546	510
With Regen 20	8734	1138	1146	776	572

TABLE 3.1: Number of frames tracked for each cell with different ***Regen*** values

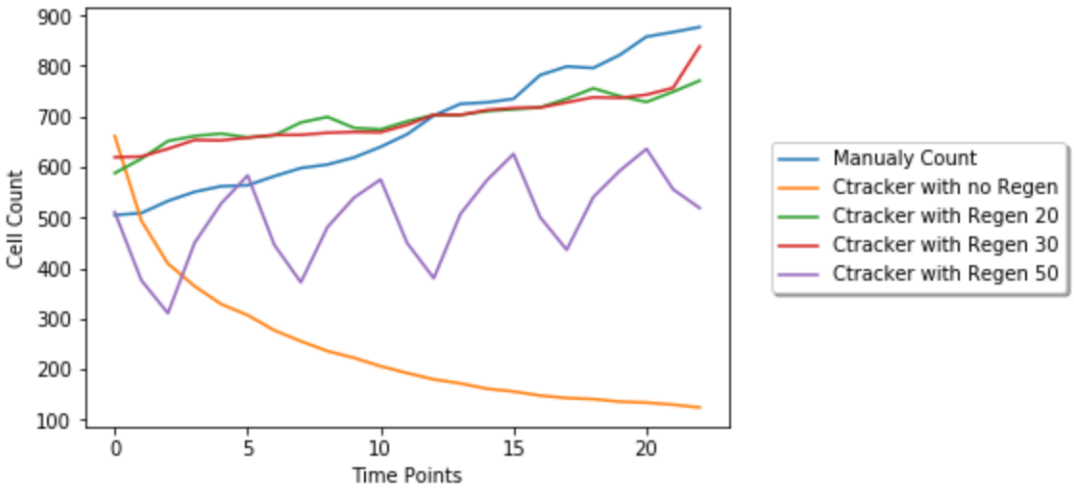


FIGURE 3.5: Compares number of cells tracked with different Regenerate Time

Brightfield Microscopy where the cells are dark and the surrounding is a bright field, the other is Fluorescence Microscopy where the specimen is usually stained, showing light against a dark background. The ***invert*** parameter determines whether or not the input video ought to be inverted. This flag is applied in the current work since all videos have a bright background with dark cells.

Regenerate Time The *Regen* parameter sets how often data points are regenerated. A value of one indicates that cell positions are regenerated for each frame. However, the value needs to be chosen carefully, as table 3.1 shows, the number of cells tracked differs considerably with differing values of ***Regen***. A ***Regen*** value of every 20 frame was found to be most suitable; if the number is too larger then it will look like figure 3.4 in which tracking of many cells is lost.

As shown in figure 3.5 the blue line reflects manual counting of the number of cells which can be regarded as the ground truth. Ctracker with no

Regenerate time means Ctracker only locates cells once at the beginning of the video, meaning track of cells following cell division and other occurrences such as occlusion by other cells will be lost for the remainder of the video. A regenerate time of 50, as shown by the purple line in figure 3.5, produces a wave form indicating how cells are "re-found" each time. The remaining two plots are regenerate every 20 frame and 30 frame, it can be seen that the trends and values are close to ground truth. But the plot with regenerate time 20 is more stable compared with 30 frames. Regenerate time with 10 frames is also tried, but there is no clear advantage of doubling the total amount of data to that achieved with regenerate with 20. Hence the parameter is chosen was 20.

Number of points

This parameter determines the desired number of points to track in any given frame. This number may diminish as Ctracker becomes unable to track cells within a window. In order to track all possible cells, this parameter needs to be sufficiently large, but not too large, because if the parameter exceeds the number of cells in the video, the software will track the same cells many times with different indices. Furthermore, the parameter cannot be too small because this will result in some cells not being tracked by the software. Therefore, it is important to have a general idea of the number of cells. By manually counting, there are around 600 cells at the beginning of videos and depending on different cell culture treatments, the number of cells grows to 800-1000; for cell culture without inhibitor there are more cells and for cell culture with inhibitor there are less cells. For this reason this parameter was choose to be 1000 for cell culture with inhibitor and 1200 for cell culture without inhibitor.

As shown in 3.6 when this parameter is much higher than the actual cell count number then Ctracker will track the same cell multiple times and lose them in next few frames because Ctracker regenerate this number of cells the plot with 2000 number of cells is shown in a wave form. This led to difficulties for Ctracker to track cells continuously. The plot with parameter set to 1000 shows the same trend.

Search Size SEARCHSIZE sets the size of the search window for tracking cells between frames. In two continuous frames Ctracker is looking for the likely position of cells in next frame within the searching window; if the

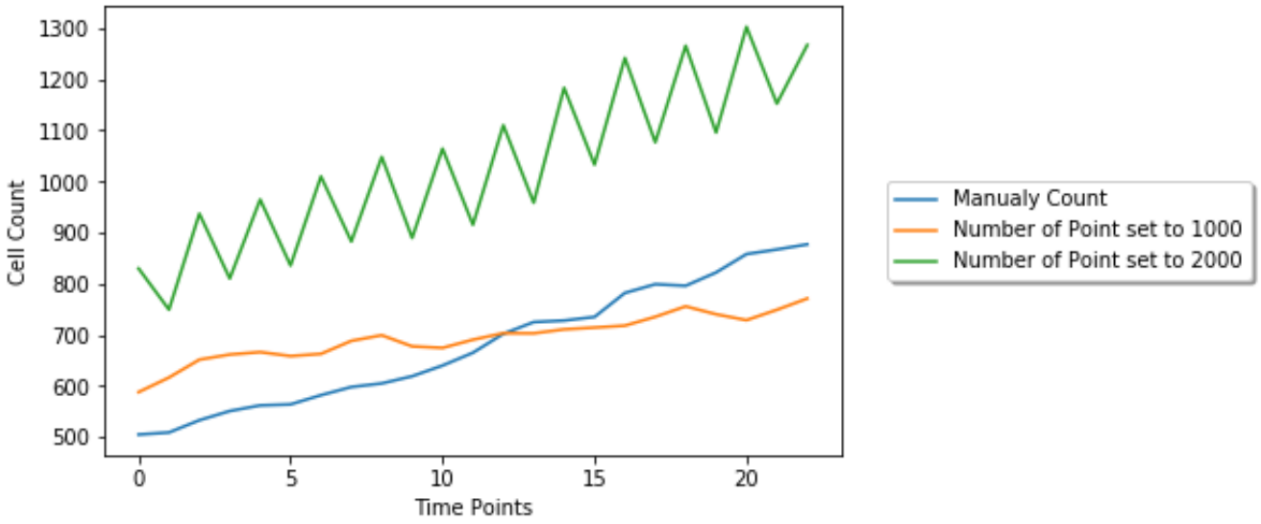


FIGURE 3.6: Compare Number of cells tracked with different parameter value

searching window is too large, Ctracker can easily confuse two cells. On the other hand if the tracking window is too small then Ctracker is likely to lose track of cells. Through observation and experimentation, the cell displacement between two continuous frame was found to be between three and 10 frames. Hence, the searching window parameter is set to 10, so that Ctracker can track cells that have travelled less than 10 pixels distance.

Output Data

Ctracker provides positions of tracked cells in terms of (x,y) coordinate pairs. For each individual cell, ideally there are 245 time points and so 245 pairs of coordinates but Ctracker might lose track of cells at some point and re-track them when *regenerate* occurs and gives the cell a new index within 20 frames. In addition, new cells may appear as a result of cell division, so most of the cells do not have 245 pair of coordinates. For each frame Ctracker is able to track individual cells; figure 3.7 shows comparison between real cell culture capture (bottom figure) and simulation of output coordinates from Ctracker (top figure). It can be seen that most cells are successfully tracked but some large debris can confuse the system and generate multiple pairs of cell coordinates. Therefore, the data needs to be processed to remove debris and large cells that are tracked as multiple cells.

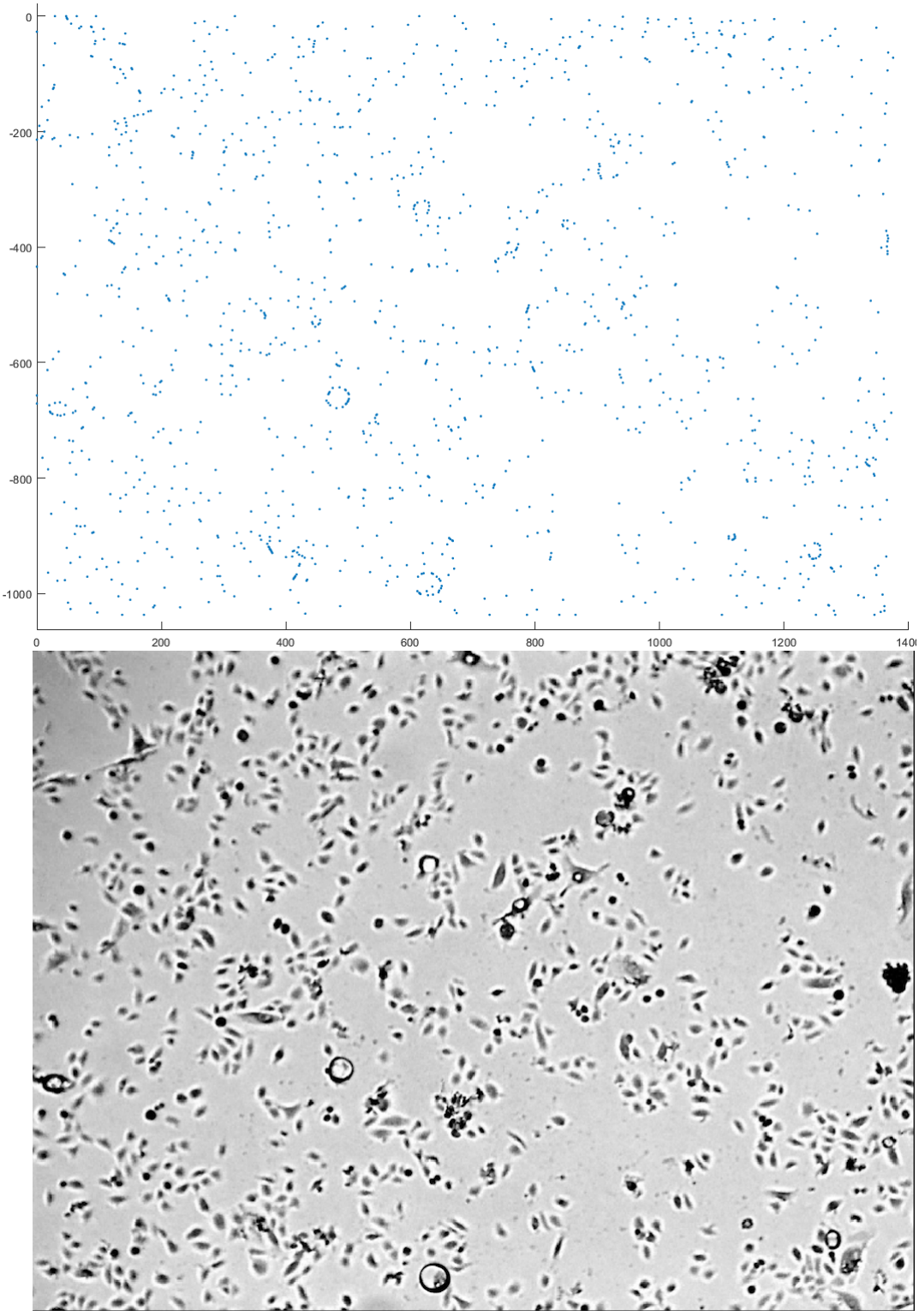


FIGURE 3.7: Comparison between video and data extracted from tracking software

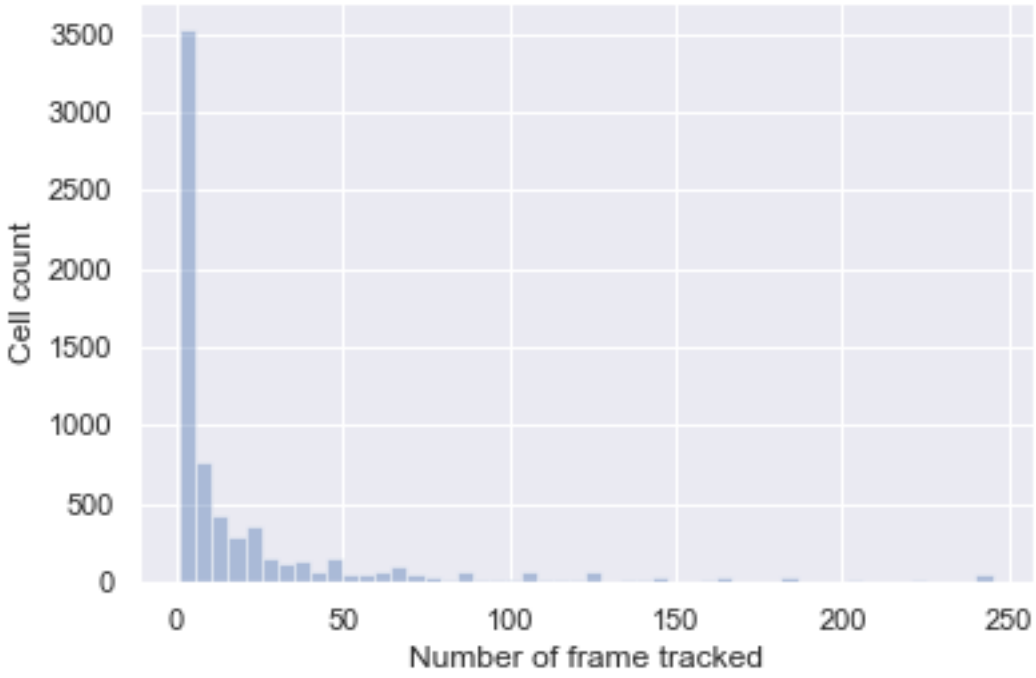


FIGURE 3.8: Histogram showing cell count with number of frames tracked with each index

3.3 Data Preprocessing

Because Ctracker regenerates data points every 20 frames a lot of new cells are tracked with new indexes generated, but some of these are duplicates of original cells which means some cells are tracked more than once. Consider the following example: the output from Ctracker generates a 245 by 13952 matrix representing 245 time points and 6976 cell indexes (13952 divided by 2). This is much larger than real cell number which is less than 1200. Figure 3.8 shows there are around 3500 cell indexes that only have one pair of co-ordinates and another 750 cell indexes with only two pairs of coordinates. This means tracking of these cells was lost after only one or two frames of the video, the reason for this is being that more than one index existed for each cell. In order to remedy this behaviour, a threshold was applied to the number of frames tracked and the number of cell indexes was reduced to 2227. There are still small number of cells with more than one index but due to the nature of Ctracker it is unlikely the cells have more than one index at the same time. Figure 3.9 shows the cell count after this data cleaning.

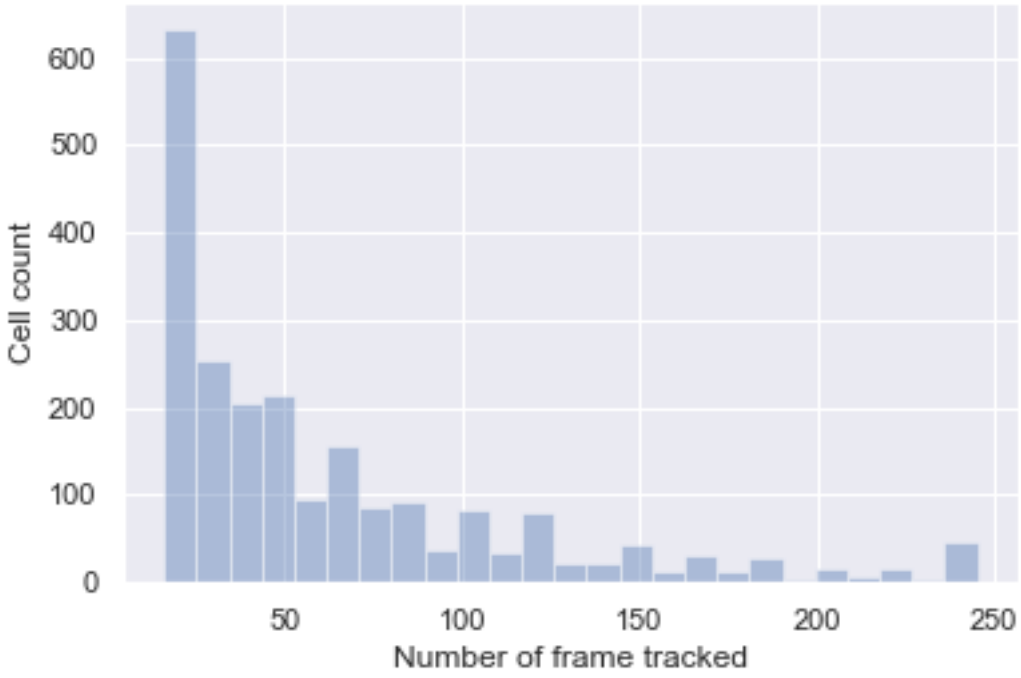


FIGURE 3.9: Histogram showing cell count with number of frames tracked with each index after cleaning

3.4 Feature extraction

Following the detection of the position of each separate cell within each of the video frames (generating the (x, y) coordinate pairs' format as shown in figure 3.10), features were extracted with the aim of characterising the behaviour of the cell population. The MATLAB programming environment was used to perform and illustrate this process. Furthermore, this study considers a single cell from an analysed video as an example of how features of interest are computed. An NHI cell-culture video having $50\mu M$ exogenous. ATP was used to illustrate tracking of a single cell. It can depict the path of this cell graphically (see Figure 3.15) because it was tracked successfully from the video's start to its conclusion. Here it is possible to observe the transformation in the behaviour of the cell from the commencement of the tracking process (adjacent to the graph's centre) to the conclusion of the tracking (the point at which the cell exits the reference frame's lower left-hand side). The cell begins its course unpredictably; however, it stabilises at a later stage. The other features extracted from the tracking data which characterise behavioural changes in the cells in response to environmental

factors include intercellular adhesion, interactions with other cells and cell division.

3.4.1 Feature consideration

The features to be extract from the videos need to describe cell behaviour but also practical to implement. Generally, features to be extracted can be classified into two groups: (i) those which define the behaviour of a cell during the period when it was successfully tracked, and (ii) those that define behaviour when the cell interacts or binds with other cells, defined as being either in-contact or post-contact. It is interesting to consider the number of cells which share the clump size or contact. For the purpose of this work, informed by visual observation of the videos, a clump is defined as a group of at least five cells. The features selected are outlined in 3.2. Definition of the cells' average angular velocity and average migration speed are provided below.

Feature	Feature Name	Description
1	Average Migration Speed	Average speed of cells over the period of tracking
2	Post-Contact Migration Speed	Average migration speed of cells over three frames after leaving a clump of cells
3	In-Contact Migration Speed	Average migration speed of cells when in contact with five or more other cells (a clump)
4	Pre-Contact Migration Speed	Average migration speed of cells before move in a clump or more other cells (a clump)
5	Average Angular Velocity	Rate of change in migration direction of cells between two video frames
6	Post-Contact Angular Velocity	Rate of change in migration direction of cells after leaving a clump
7	In-Contact Angular Velocity	Rate of change in migration direction of cells when in a clump
8	Pre-Contact Angular Velocity	Rate of change in migration direction of cells before move in in a clump
9	Cohesivity	Average number of contacts per cell
10	Average Cell Clump Size	Average size of clump in number of cells
11	Average Contact Duration	Average duration of cell contact with clump
12	Cell Count	Difference between the maximum number of cells tracked during the video from the number tracked at the beginning

TABLE 3.2: Summary of features extracted from cell culture videos.

Average migration speed

An object's speed can defined as the rate of change of its position. The objective, in this case, is to use a video to attain a cell's migration speed. This may be established by computing how many pixels are traversed over a specified interval of time - in this case, the time interval between two consecutive video frames, the frame rate being one for every five minutes. Consequently, the migration speed can be calculated by simply computing the Euclidean distance between a cell's respective coordinates between consecutive frames. Figure 5 provides a graphical representation of this in which the cell's starting position is located at coordinates (74,32) in one frame, and at coordinates (75,33) in the subsequent frame. Consequently, the speed of the cell is calculated from the distance it travels in dt (five minutes). The same technique is applied to compute the migration of all cells for the whole video.

Average angular velocity (or migratory persistence)

Biologists have a particular interest in persistence with regard to cell migration. This feature may be explained as the trend of cells to change their direction. Consequently, it is necessary to acquire the cell's direction of travel in each video frame in order to compute migration persistence. The means by which the vector angle, as determined by the cell's coordinates in consecutive frames of the video, may be applied to establish the direction of travel is illustrated in Figure 6. In this case, angular velocity is defined as the rate of change in a cell's direction of travel across subsequent video frames. An example of such a calculation across two consecutive frames is depicted in Figure 3.13.

Clump definition

Cell-cell adhesion is critical if multicellular organisms are to advance and survive[116]–[118]. There is an expression of the Ca2+ cadherin superfamily, being dependent on cell–cell adhesion proteins in the majority of tissues and organs of both vertebrates and invertebrates. Various cadherins are shown in cell layers, particular tissues as well as neuronal types of cell which are compatible with functions in clear cellular detection as well as in sorting procedures [117], [119]–[124] . Currently people have little knowledge of the molecular or physical dynamics of compaction (maximisation of

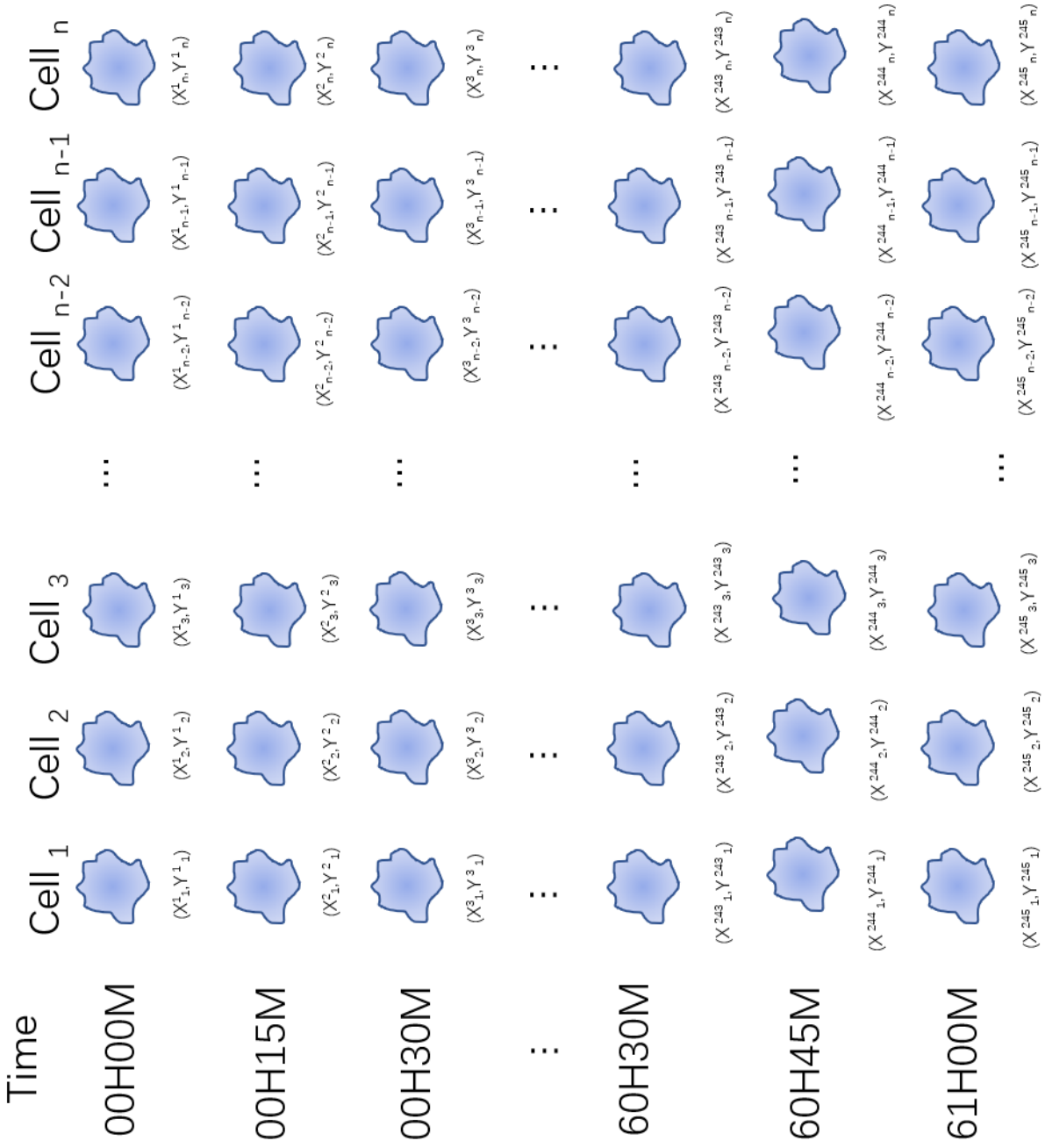


FIGURE 3.10: Data in the form of coordinate pairs before feature extraction

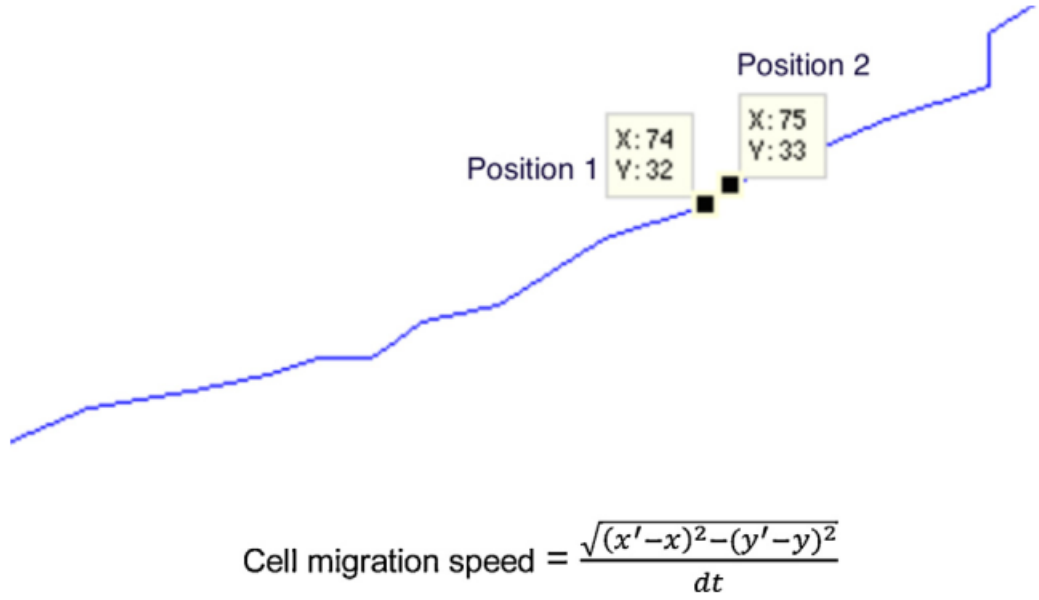


FIGURE 3.11: Example calculation of cell migration speed (pixels/frame)

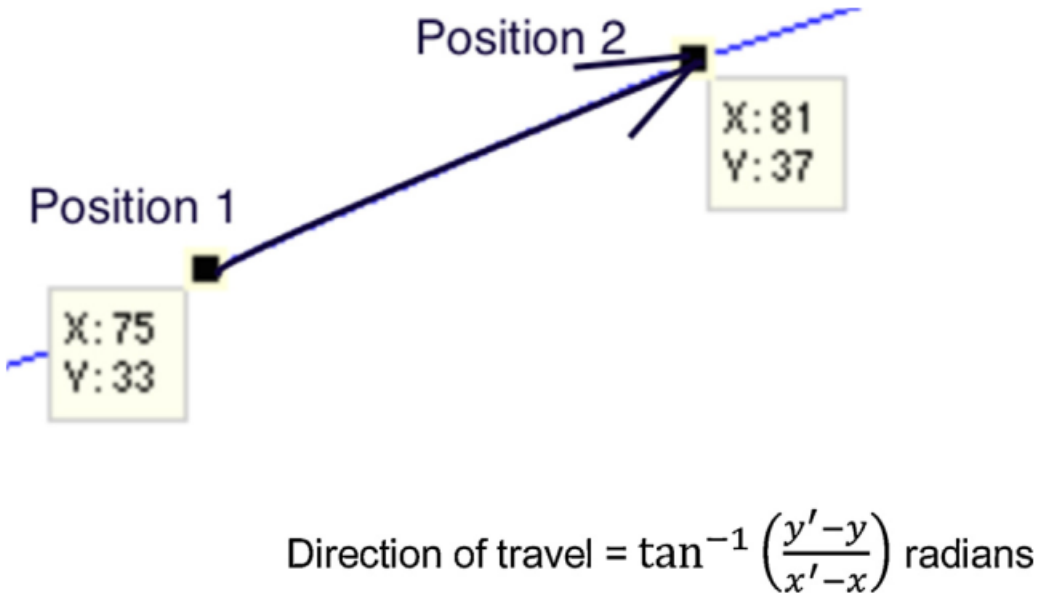


FIGURE 3.12: Direction of travel of cell migration.

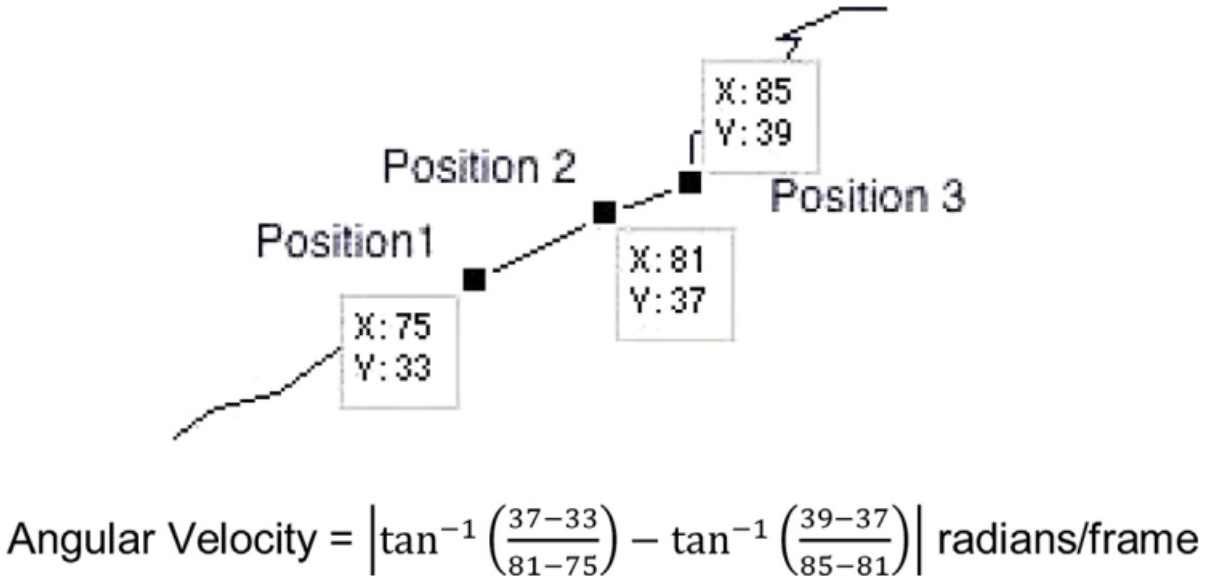


FIGURE 3.13: Example calculation of cell migration persistence over two consecutive frames.

adhesive contacts), cell–cell adhesion (cell stickiness) or condensation (aggregation of large cell colonies) in the course of the creation of tissues. This is despite the endeavours of studies to clarify the mechanical and molecular attributes of cadherin-mediated adhesion[125], [126] .

Cell-cell adhesion can be described in three phases. [127] The first phase involves the membrane contacts between two cells instituting the amalgamation of greatly-mobile E-cadherin cell surface pool into immovable punctuated accumulations on the line of contacting membranes. Such E-cadherin accumulations coincide spatially with membrane attachment sites for filaments of actin which branch from circumferential actin cables and restrain each cell. Phase Two involves these actin cables adjacent to cell–cell contact sites becoming divided with the remaining two cable ends moving outwards towards the contact's perimeter. Subsequently, E-cadherin puncta subsets are driven to the contact's margins at which location they amalgamate large E-cadherin plaques. This leads to the creation of a circumferential actin cable which restrains the two cells. It is also implanted into each E-cadherin plaque at the margin of the contact. Both cells, at this point, attain maximum contact, known as compaction. In the case of further single cells attaching themselves to large cell groups, such changes in the actin and E-cadherin disseminations are relocated. Phase Three of the adhesion process is initiated when further cells attach themselves to groups of four or more

cells. When this occurs, it is apparent that circumferential actin cables which are connected to E-cadherin plaques on bordering cells are retrained in a purse-string action, leading to greater amalgamation of separate plaques into the multicellular contacts' vertices. The process of restructuring actin and of E-cadherin leads to cells condensing into colonies. The formation of a paradigm in order to clarify how actin and E-cadherin cables coordinate to reform initial cell–cell contacts into the concluding condensation of cells into colonies by means of compaction and strengthening.

Post and Pre Contact Behaviour

When a cell is formed by combining with other cells, it is known as an E-cadherin-compromised cell. This cell examines the emergent behaviour caused by the interaction of single and sub-populations of E-cadherin-compromised cells with unaffected normal cells within a monolayer environment. It is necessary to extract movement behaviour from different cell stages which are pre-contact, in-contact and post-contact migration speed, pre-contact angular velocity, in-contact angular velocity and post-contact angular velocity.

FIGURE 3.14: Cell movement in and out of clumps three times

In figure 3.15, the red shading indicates the period when the cell is within a clump; it is therefore an easy task to extract features before, after and in contact with other cells. If the cell moves in and out of a clump more than once as shown in 3.14, where horizontal line is time axis and red period indicates cell i is in a clump, there are three separate red periods, so for cell i the value of in-contact behaviour is the average of these time periods (and similarly for for pre- and post- contact behaviours).

Cell Growth

Cell division is an important part of the cell cycle and possibly impacts loss of E-cadherin function upon the growth attributes of epithelial cell populations, which otherwise would be normal. Furthermore, a comparatively small cell population subset would have a noticeable impact upon the total population growth dynamics, and it could be anticipated that the divergent population would have an impact upon the normal population growth, or if the converse would apply.

In this situation, the ***Cell Growth*** facet indicates cell count from the first frame to the last of the video.

Contact Duration and size

Furthermore, it is particularly interesting to see how the loss of the E-cadherin function affects the duration and size of the cell-cell bonding; these two features are defined as below:

- ***Contact Duration***: Duration of cell-cell binding in terms of frams(five minuts).
- ***Contact Size*** : Number of cells in each group of cell.

3.5 Data Visualization

After computing all features, each cell index has a 1 x 11 feature vector and an output that indicates if the respective cell culture applied an inhibitor. Each feature vector is an instance of the respective dataset; figure 3.16 shows the first five instances. Figure 3.17 shows the total number of instances for two classes labelled 0 and 1, a similar number for each.

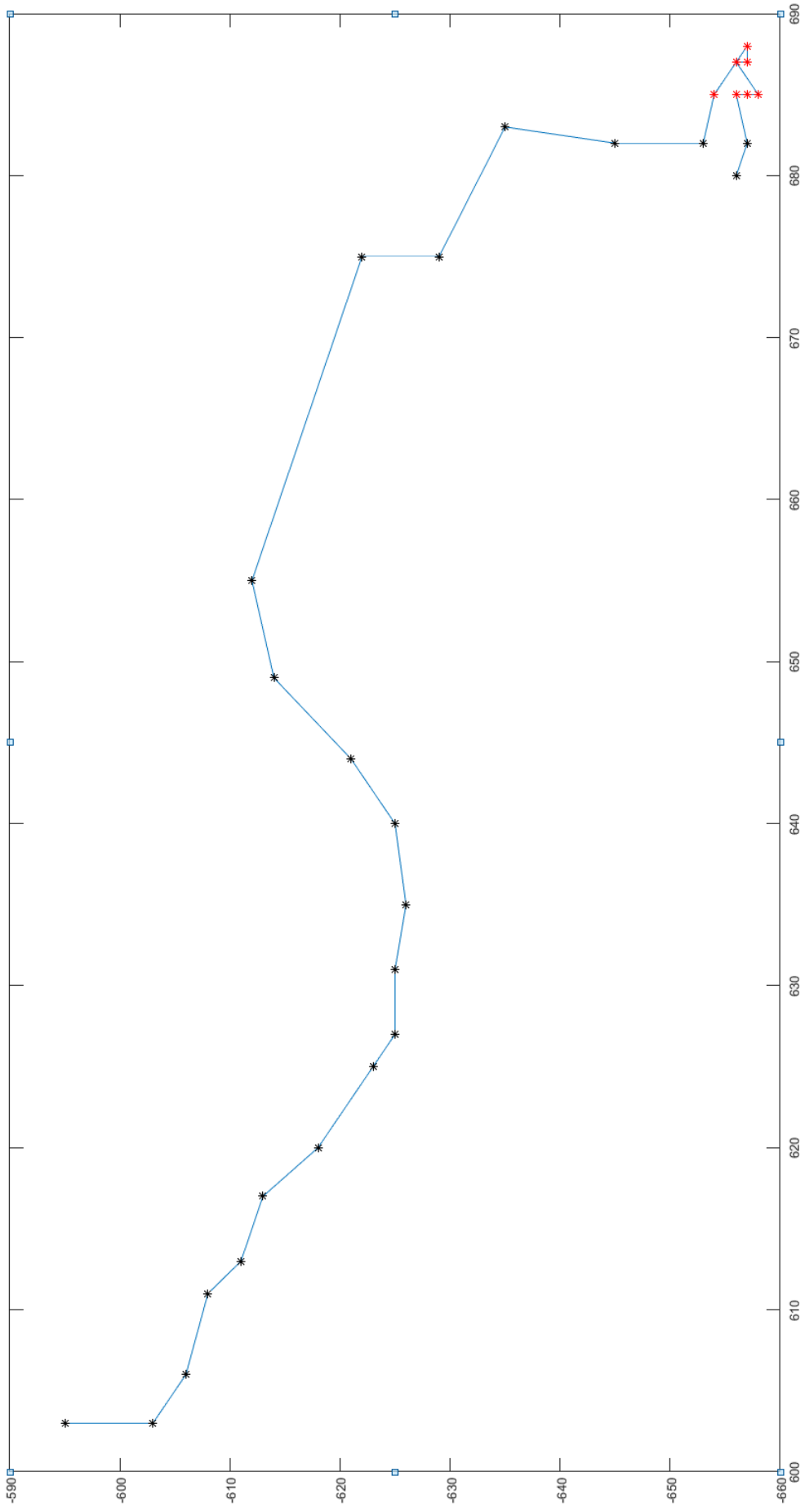


FIGURE 3.15: Cell movement path

	Average Speed	Average Angular Velocity	Pre-contact Speed	Pre-contact Angular Velocity	Post-contact Speed	Post-contact Angular Velocity	In-contact Speed	In-contact Angular Velocity	Contact Duration	Clump Size	Number of Contacts	Output
0	3.4013	1.55640	2.29540	0.63085	5.70590	0.73353	5.2348	1.54830	2	6.50	1	0
1	3.7487	0.53659	0.62781	0.22605	0.80474	0.39270	2.3357	0.52635	8	6.25	10	0
2	3.0424	1.57040	0.80474	1.04720	1.60950	2.84660	3.0559	1.62280	7	7.00	9	0
3	4.5159	1.21240	3.92420	1.89250	6.94770	0.38662	4.3953	2.26380	5	6.40	8	0
4	5.0185	1.30120	3.48260	0.27628	8.46220	0.24058	5.7334	0.35294	4	7.00	13	0

FIGURE 3.16: First five instance of a dataset

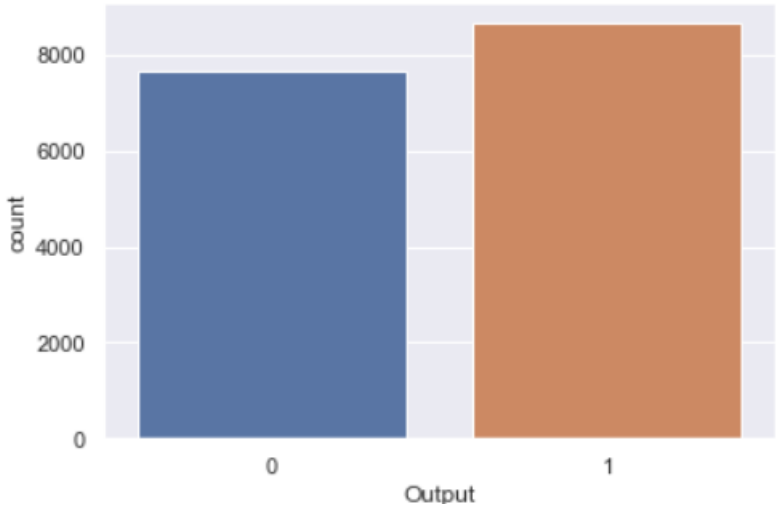


FIGURE 3.17: Output Category Count

3.5.1 Comparing training and test sets

Having established a similar number of instances in both classes of a dataset, it is useful to visualize the distribution of each extracted feature, as previously defined. Figure 3.18 shows each feature summary of all feature vectors in the dataset. The summary includes basic statistical measures such as the mean, standard deviation, minimum, maximum and the quantiles of the data. Also, it can be seen that there are no missing values as all feature counts are the same.

	Average Speed	Average Angular Velocity	Pre-contact Speed	Pre-contact Angular Velocity	Post-contact Speed	Post-contact Angular Velocity	In-contact Speed	In-contact Angular Velocity	Contact Duration	Clump Size	Number of Contacts
count	16304.000000	16304.000000	16304.000000	16304.000000	16304.000000	16304.000000	16304.000000	16304.000000	16304.000000	16304.000000	16304.000000
mean	3.163763	1.449139	2.386853	1.071346	2.815108	1.164245	3.259542	1.458334	7.937316	6.733462	5.014046
std	1.151903	0.497033	1.722979	0.969263	2.035520	1.097402	1.691285	0.982233	5.487116	0.982153	3.540876
min	0.235290	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	6.000000	1.000000
25%	2.368675	1.097825	1.079800	0.349070	1.276100	0.350812	2.070000	0.680007	3.000000	6.000000	2.000000
50%	3.102950	1.433650	1.939100	0.761750	2.305200	0.785400	3.040500	1.332600	7.000000	6.428600	4.000000
75%	3.887725	1.778700	3.273775	1.570800	3.896100	1.692950	4.182900	2.028675	12.000000	7.000000	7.000000
max	8.335600	4.516800	11.800000	5.965300	14.999000	7.789100	14.142000	6.141300	20.000000	15.842000	24.000000

FIGURE 3.18: Feature summary

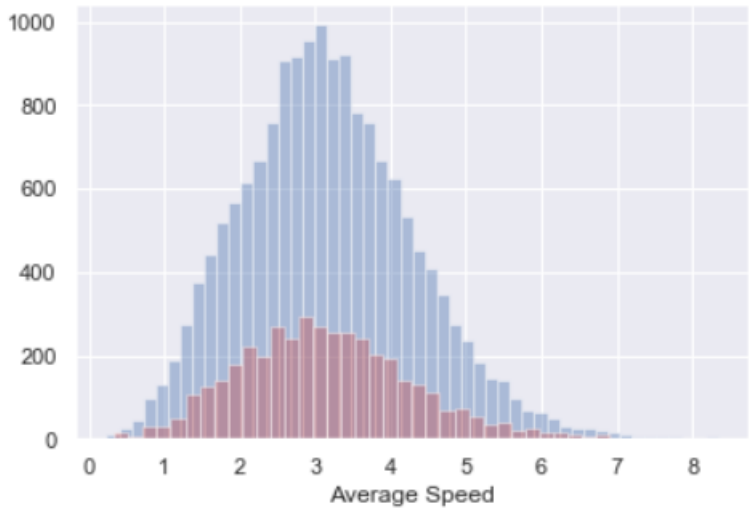


FIGURE 3.19: Distribution of Average Speed in Training and Test sets where blue is the training set and red is the test set

After viewing the summary for each feature, the next step is to check the distribution of features in what will be the training and test sets. Figure 3.19 shows distribution of *average speed* in both sets; it can be seen that average speed is showing a bell shape in both the training and test sets. A plot of the remaining features can be found in the appendix and shows that all features have the same distribution for training and test sets.

3.5.2 Correlations between features

The correlation of two features refers to the degree of closeness between them; normally high correlation indicates two features have a linear relationship. Two features with high correlation have a greater linear relationship, so their impact on a training classifier is almost the same. Therefore, when a pair of features are highly correlated, one of them can be removed. Figure 3.20 shows all eleven features correlations with each other.

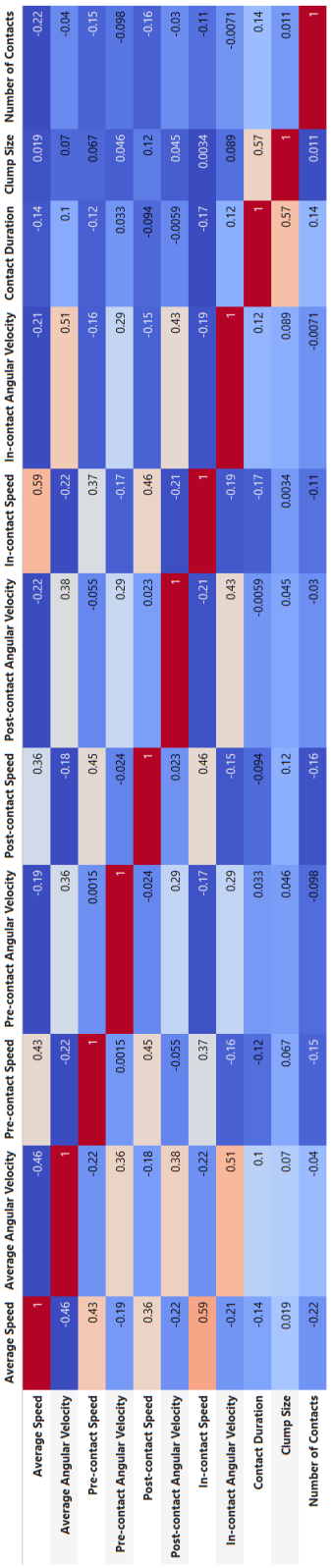


FIGURE 3.20: Variable correlations

From figure 3.20, it can be seen that all *Speed* related features have weak correlations with each other but none greater than 0.59; the same as *AngularVelocity* related features. Other than these, the highest correlation is between *Clumpsize* and *ContactDuration* which has a value of 0.57. There is no pair of features has correlation greater than 0.9, and therefore no features need to be removed.

3.6 Dataset preparation

3.6.1 Cell culture dataset

Each feature vector within the original dataset represents the behaviour of a cell at a certain time, but the characteristics of the cell culture are also very interesting and may provide useful information. $Cell_i$ has a feature vector of ${f_i}^n$ at time point n, and feature vector of $Cell_i$ during a period of d ${f_i}^d$ can be compute by following equation 3.1

$$ {f_i}^d = \frac{\sum_{i=n}^{n+d} f_i}{d} \tag{3.1} $$

In order to obtain features of each cell culture behaviour, each of the 24 videos was divided into 10 to give 240 short videos in total. Feature extraction was performed on all cells in each of these 240 short videos. The average value of features for all individual cells in each short video is the feature vector F, define in equation 3.2 where m is total number of cells in this video.

$$ F = \frac{\sum_{i=1}^{m} {f_i}^d}{m} \tag{3.2} $$

Therefore, there are two datasets, one consists of each individual cell feature vectors f_i and the other consists of feature vector F , which is the average behaviour of videos with 24 frames.

Time Period\Treament	Period 1	Period 2	Period 3
Control Vs Control+PPADS	Control P1	Control P2	Control P3
10μMATP Vs 10μMATP+PPADS	10μMATP P1	10μMATP P2	10μM ATP P3
50μMATP Vs 50μMATP+PPADS	50μMATP P1	50μMATP P2	50μM ATP P3

TABLE 3.3: Dataset splits into 9 smaller single cell data, each of them consist cell culture with same time period and ATP concentration which including cultures with PPADS and without PPADS.

3.6.2 Single cell dataset

The *cell culture dataset* is used to describe the cell movement within each population. On this basis, an additional dataset is computed, which is called a *single cell dataset*. Definition of the features in the single cell data set is the same as that of cell culture dataset. The difference lies in that the single cell dataset does not illustrate the average behaviour of all cells in one cell culture, as it is usually computed from a much smaller number of frames. Hence, the single cell dataset is primarily for describing single cell behaviours during 24 frame, which relates to 4 hours of real time behaviour.

The reason why a 24 frames time period is selected is because the time period cannot be too small, as otherwise, some features cannot be effectively calculated, such as the average duration of each contact and number of total contacts. Keeping tracking long enough is required to measure cell behaviours for the average contact duration and the number of total contacts, since the average contact duration is greater than 10 frames, while average contact is greater than one if the tracking window is larger than 24 frames.

The tracking windows cannot be too large either, given the features are typically described in the form of average behaviour over this time. If the tracking window is too large, there will be certain information lost during averaging process. Therefore, the tracking window is set as 24 frames in the dataset, and three time periods within the video are specified, which are the beginning 24 frames, the middle 24 frames and the last 24 frames of the video.

In summary, there are primarily two types of datasets. The first is a cell culture dataset, which can be used to characterize the cell culture behaviours. The second one is a single cell dataset, which can be used to characterize single cell behaviour and possibly identify sub-populations. During further experiments, all experiments will be based on both types of dataset.

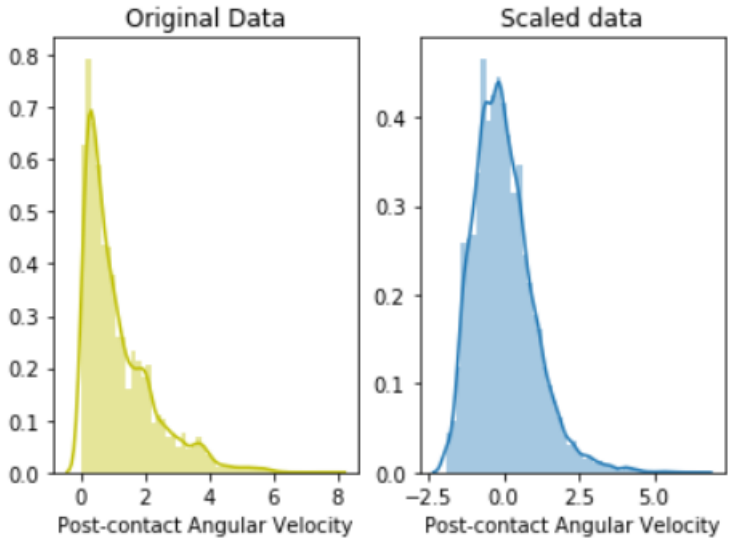


FIGURE 3.21: Left hand side showing feature distribution before standardization and right hand side after standardization

3.7 Application of PCA

The large number of features extracted from the data results in a high dimensionality, which can be visualized individually, as considered in the previous section, but difficult to assess in combination. By using an orthogonal transformation such as Principal Component Analysis (PCA) the dimensionality of the data can be reduced and the correlation between features can be assessed using the two or three principle components.

Before applying PCA to the data, one important issue to resolve is that PCA is affected by the numerical scales used to represent the data. In this case, the scales used are considerably different for some of the features. For example, the range of values for *speed* is between 0 and 10, but *angular velocity* is always less than 6.28 (2π). Figure 3.21 shows data before and after standardization. Following standardization, all feature have a mean of zero and a variance of one, which help optimise the performance of PCA.

After standardization of all features, 80% of the total data was used as a training set and remaining 20% used for a test set. A python open source package called scikit-learn was used for data processing and performing PCA.

Figure 3.22 shows visualization of two principle components; it is important to note that component one and two are abstract, they are a transformation of the original 11 features. From the figure it can be seen that the two classes

are not separated. The related accuracy is shown in Table 3.4; standardization improves the result by 3% , but is still only 56.3%.

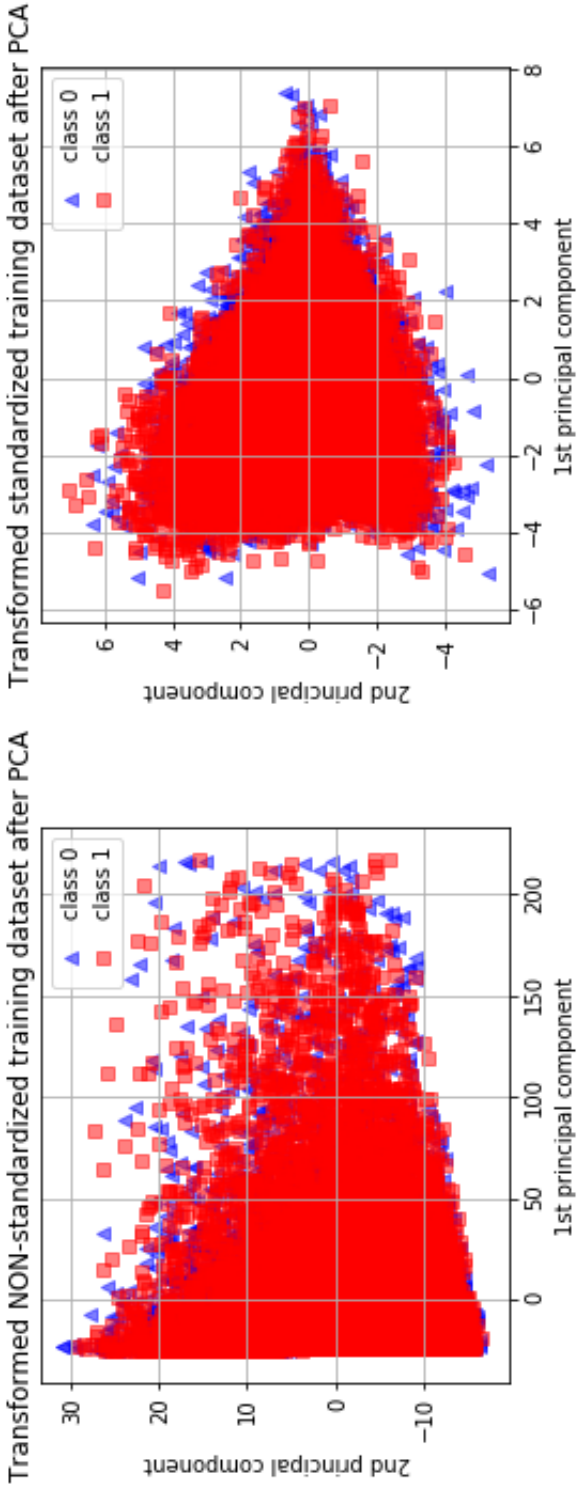


FIGURE 3.22: PCA output visualization

Prediction accuracy	PCA without standardization	PCA with standardization
Training dataset	53.44%	56.83%
Test dataset	53.01%	56.3 %

TABLE 3.4: Accuracy of PCA

3.7.1 Importing necessary libraries

The first step in the implementation of PCA using Python is declaring all libraries needed, the first one is *pandas* which is used for manipulating the dataset, the next one is *sci-kit* learning packages which provides data pre-processing and a PCA implementation. The classifier following PCA is also provided by sci-kit learning library. The following code shows which packages are imported:

```
#import nesscery libarys, including data manipulation liabries pandas and numpy
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
```

3.7.2 Pre-processing

In machine learning and other techniques it is necessary to avoid classifier overfitting on the training data. A common solution to this problem is to retain some of the available data and use it as a test set. Scikit-learn provides a function named *traintestsplit* that can randomly split the dataset into two parts that are used for training and testing. Therefore, a dataset can be easily created retaining 30% of the whole data to test the classifier and the rest for training , The script below imports the datasets and then takes the first eleven columns of data as features which is Xdf and the last column as labels which is Ydf. Next, the dataset is divided into training and test sets:

```
df = pd.read_csv('TotalDataset.csv')
df.describe()
X_df = df.values[:,0:11]
Y_df = df.values[:,11]
x = StandardScaler().fit_transform(X_df)
```

```
X_train, X_test, y_train, y_test = train_test_split(X_df, Y_df,
    test_size=0.30)
```

3.7.3 Feature transform by using PCA

The original feature vector has eleven dimensions; in this step PCA is used to project these features onto two dimensional Cartesian coordinates. The remaining two dimension of features are have no great relevance as they are a projection of the original features that have the least variance between classes. The purpose of PCA is in reducing the dimensionality of the feature vectors; as a result, the number of principal components retained is defined accordingly as detailed in the following code:

```
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(x)
```

3.7.4 Training classifier

As described above PCA is used to reduce the dimensionality of feature vectors, but it does not perform the subsequent classification of data. In this implementation a standard decision tree classifier is used as shown in the following code.

```
classifier = DecisionTreeClassifier()
classifier.fit(principalComponents, y_train)
Test_transformed = pca.transform(X_test)
pred_labels = classifier.predict(Test_transformed)
```

3.8 Application of SVM

Support Vector Machine (SVM) is a supervised machine learning algorithm that is used in many classification problems. The python package scikit-learn implementation of SVM has also been used for this work.

3.8.1 Importing libraries

The following lines of code detail the implementation of SVM by first importing the required libraries which including Sci-kit learning and pandas.

Scikit-learn provides an implementation of SVM and pandas allows algorithm to manipulate data. To read data in the csv file, there is a function called read csv that can read files separated by comma and store as matrix. The script below details the importing of the data set which is stored as a data frame so that it is easy to process in later steps.

```
#import SVM function in Sci-kit learn libary
from sklearn.svm import SVC
import pandas as pd
#import dataset
```

3.8.2 Pre-processing

The same data preprocessing is performed as for PCA; the data is split into two parts, of which 30% of the whole data is retained for testing and the rest (70%) for training.

```
df = pd.read_csv('TotalDataset.csv')
df.describe()
X_df = df.values[:,0:11]
Y_df = df.values[:,11]
x = StandardScaler().fit_transform(X_df)
X_train, X_test, y_train, y_test = train_test_split(X_df, Y_df,
    test_size=0.30)
```

3.8.3 Training the algorithm

As previously stated, the data has been partitioned into testing and training sets. The training of the SVM is achieved by using Scikit-Learn which consists of the SVM library including built-in classes for various types of SVM algorithms. Given that tasks will be classified for execution, Support Vector Classifier (SVC) classes from the Scikit-Learn SVM library will be used. The SVC function requires a parameter specifying the kernel type. The fundamental kernel type is a linear one, but for some classification tasks the data is not linearly separable, so different types of kernels are available for SVM that are able to classifies more complex data set such as that considered in

this thesis where there are eleven dimensions of feature vectors. Scripts below only shows SVM with a linear kernel but other type of kernels have also been utilised including polynomial and gaussian.

The *fit* method of the SVC class is used to train the algorithm by applying the training dataset, passed as a parameter to the fit method as illustrated in the following code:

```
#Define kernal type
        svclassifier = SVC(kernel='rbf')
        #Train classifier
        svclassifier.fit(X_train, y_train)
```

3.8.4 Making predictions

In terms of making predictions on test set, the *predict* method of the SVC class can be utilized. The following code is used to obtain the predicted output of the test set:

```
        #Prediction of trained classifier
        y_pred = svclassifier.predict(X_test)
```

3.8.5 Evaluating the trained algorithm

The most usually adopted metrics for classification tasks encompasses recall, Confusion matrix, precision, and F1 measures. Scikit-Learn's metrics library provides classification report and confusion matrix methods. Here is the code for obtaining these metrics:

```
        #Use output report libary to evaluate output results
        from sklearn.metrics import classification_report, confusion_matrix
        print(confusion_matrix(y_test,y_pred))
        print(classification_report(y_test,y_pred))
```

3.9 Application of evolutionary algorithms

Cartesian Genetic programming (CGP) is the evolutionary algorithm adopted for the work described in this thesis. A non-cyclic directed graph, CGP is a

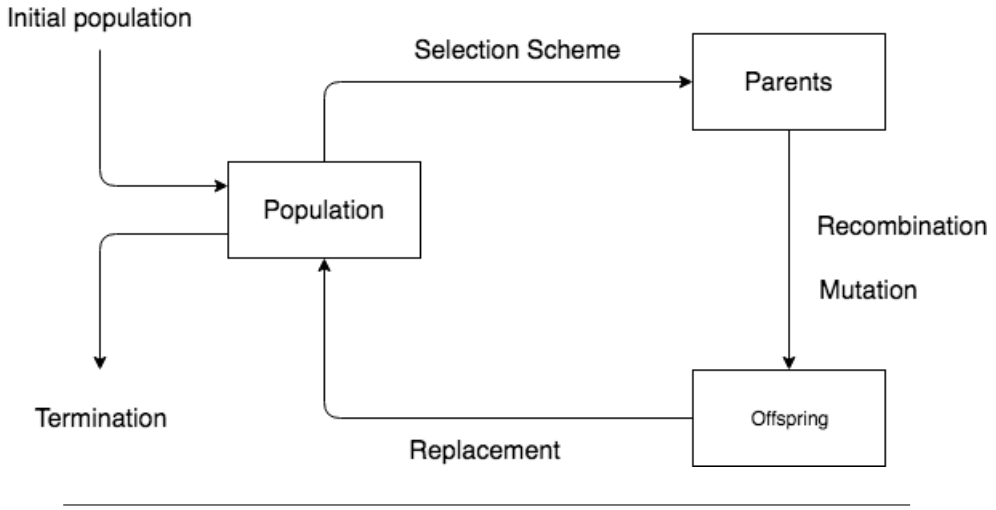


FIGURE 3.23: CGP flow chart

arrangement of processing nodes which may be reconfigured in the course of the evolutionary cycle by two basic means. These are the selection of a new node function from a predefined list and, stipulating the connection to other nodes through each input and output. CGP is advantageous over other categorisation methods in two ways. Firstly, it has been demonstrated that CGP develops high-performance classifiers for highly non-linear complex data [128]. Secondly, when a high-performance classifier has been evolved, it can be easily and fully defined as a mathematical expression by decoding the respective CGP network. This process is dissimilar to most other machine learning methods. As aforementioned, this is able to supply a helpful awareness of the biological behaviors acquired from cell-culture dynamics which occupy a defining function in its categorisation.

3.9.1 Dataset

As previously mentioned, the dataset which is used as input to CGP is as depicted in figure 3.2 with the exception of *average contacts per cell*. The average number of contacts per cell feature is omitted from 24 frames because this it is too short to obtain a meaningful value.

LISTING 3.1: Example Dataset

```
9,1,178,
3.35,3.56,3.59,1.32,1.21,1.50,6.29,8.58,2272,0,
3.30,3.79,3.82,1.38,2.80,1.55,6.86,9.03,2194,0,
2.46,2.39,3.14,1.73,2.26,2.17,9.65,11.13  2998,1,
2.58,2.21,3.23,1.80,1.68,2.21,9.35,10.68  2624,1,
```

The data set output is either '1' or '0', thereby giving the expected classification outcome. An output of '1' shows that the training example is with PPADS and an output of '0' shows the training example to be without PPADS.

The input and output number is shown by the first data line, in this example there are nine inputs and one output. The number of examples utilised as training data is shown by the last number, and there are 178 such examples in this data set.

A further 57 examples were utilised as test data to assess the CGP output performance, such data being unseen for CGP during the training stage.

3.9.2 Fitness function

The fitness function evaluates the performance of each individual classifier evolved. From a technical aspect, this process or function allocates a quality evaluation of the genotype. This function is normally produced from a quality evaluation in the inverse representation and the phenotype space.

The objective of this study is to successfully classify two groups of cell cultures. The selection scheme and the fitness function are set to select the greatest-precision chromosome from the population.

3.9.3 Parameters

The Table below shows the example parameters which are utilised in a CGP run, where **Evolutionary Strategy** is the number of chromosomes in the population and the number of chromosomes which are required to be kept for the next generation. **Inputs** and **Outputs** are dependent upon data types. The **Node** is the number of function nodes in each chromosome, including both active and inactive nodes. The **Node Arity** is the number of connections for each function node. The **Mutation rate** is the percentage of node mutation. In this example, there are 500 nodes and five per cent of the mutation rate; therefore, 25 nodes will be mutated.

Parameters	
Evolutionary Strategy :	(1+4) − ES
Inputs :	9

```
Nodes:                                          500
Outputs:                                        1
Node Arity:                                     3
Mutation rate:                                  0.050000
Function Set: add sub mul div sin and nand or
                                    nor not abs pow 1 0 (14)
```

3.10 Conclusion

This chapter presents the methodology used to classify and characterize cell cultures by using CGP, PCA and SVM. First of all, acquisition of time-lapsed video of cell cultures was described, followed by cell tracking which provided cell position data. The next section in this chapter presents the results of extracting features from the cell tracking data, including individual cell movement and cell-cell behaviours.

Chapter 4

Results and analysis

Contents

4.1 Introduction

This chapter presents the experimental results and their analysis. Section 4.2 presents the statistical analysis of the extracted features where data have been averaged for the whole population over a 24 hour period. Section, 4.5, presents results for the evolutionary algorithm used, CGP, including results and analysis for CGP with different parameters. The following section introduces model evaluation, comparing features extracted using the algorithm presented in chapter 3 with features of real cell culture.

4.2 Statistical analysis of results

The work presented in this thesis considers the analysis of three main cell cultures: (i) a control culture without ATP; (ii) a culture having $10\mu M$ ATP; and (iii) a culture having $50\mu M$ ATP. Each video's average angular velocity and average cell migration speeds were computed and subsequently shown in Table 4.1. The utilisation of analysis of variance (ANOVA), as depicted in figure4.1 shows that the separation between the three migration speed classifications was statistically significant. These outcomes were verified by comparison with the manual tracking of 15 random cells for each experimental setup as depicted in figure 4.2. Similarly, angular velocity results are depicted in figure 4.5, and good separation between the three sets of culture conditions is also shown.

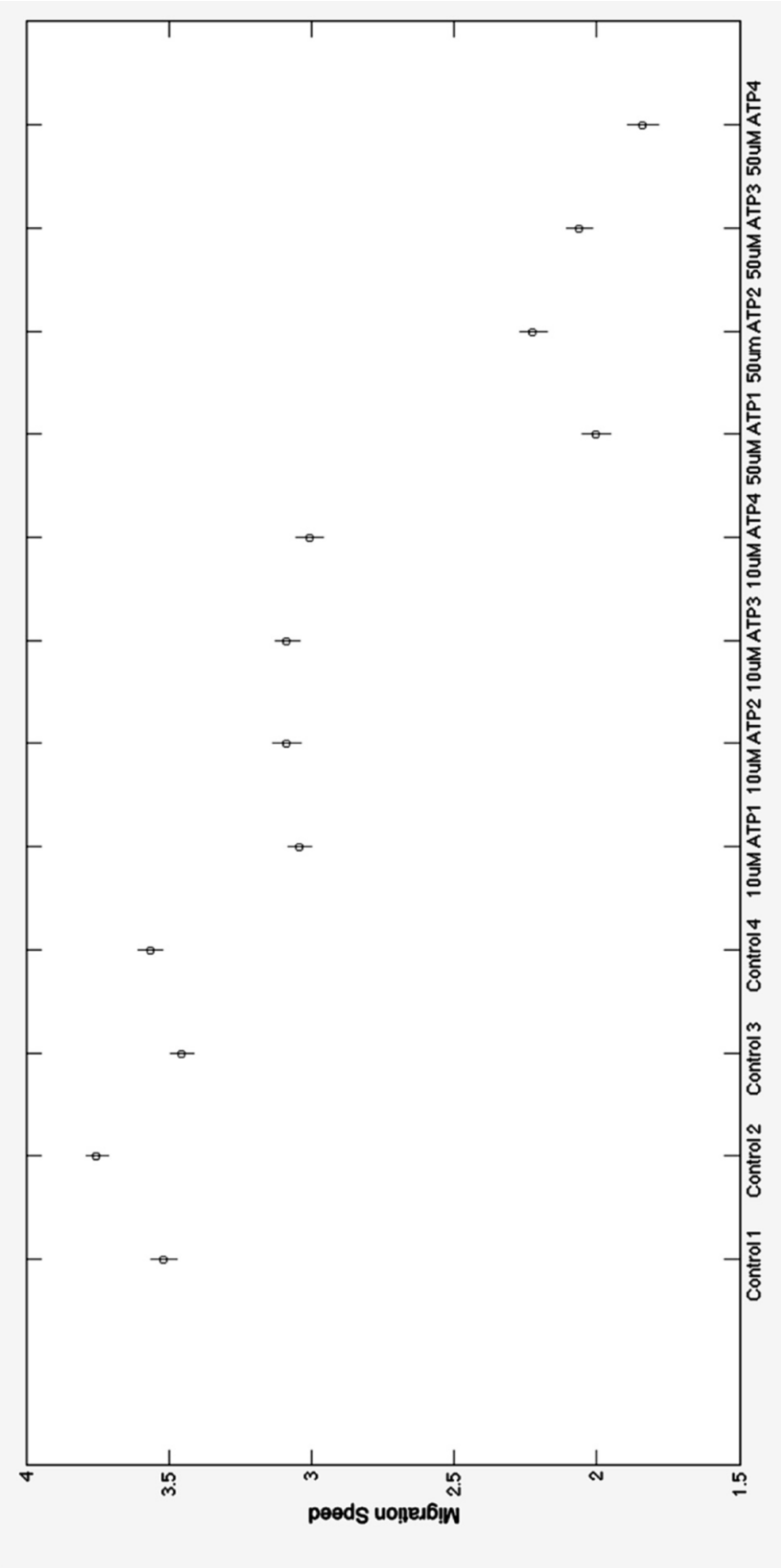


FIGURE 4.1: Automated calculation of cell migration persistence. Average migration speeds are shown in F-distribution form for Control ,10/muM ATP and 50/muM Videos: small circles mark the mean of the group and the bars the 95% confidence interval.

Cell Culture Video	Average Migration Speed (pixels/frame)	Average Angular Velocity (rads/frame)
Control1	3.52	1.31
Control2	3.75	1.35
Control3	3.45	1.37
Control4	3.56	1.36
10uM ATP1	3.04	1.52
10uM ATP2	3.09	1.39
10uM ATP3	3.08	1.52
10uMATP4	3.01	1.46
50uMATP1	2.00	1.79
50uMATP2	2.22	1.74
50uMATP3	2.06	1.77
50uMATP4	1.83	1.85

TABLE 4.1: Average migration speed and average angular velocity values for a control culture with no ATP, a culture with 10uM ATP and a culture with 50uM ATP.

4.2.1 Additional features

The characteristics of connections between the cells are of special interest as well as the migratory persistence and cell migration speeds. This is associated with the physical extent of the contact which develops between them and the extent to which interacting cells make transient or more highly-maintained contacts. The features, previously considered in Chapter 3, demonstrate the behaviour of cells whilst in contact and post-contact in each of the videos shown in figures 4.4 4.8 4.3 4.6 4.9 4.7.

Furthermore, the aspects which define the general population behaviour such as ***Cell growth*** are depicted in figure 4.11.

4.3 PCA results

PCA can provide a meaningful reduced dimensional distribution of data points. The following section provides results for the use of PCA with a classifier, as previously described in Chapter 3, for both datasets.

4.3.1 Single cell dataset

Table 4.2 shows predictive results for the single cell dataset, where the data has been divided into 80% for training and 20% for testing.

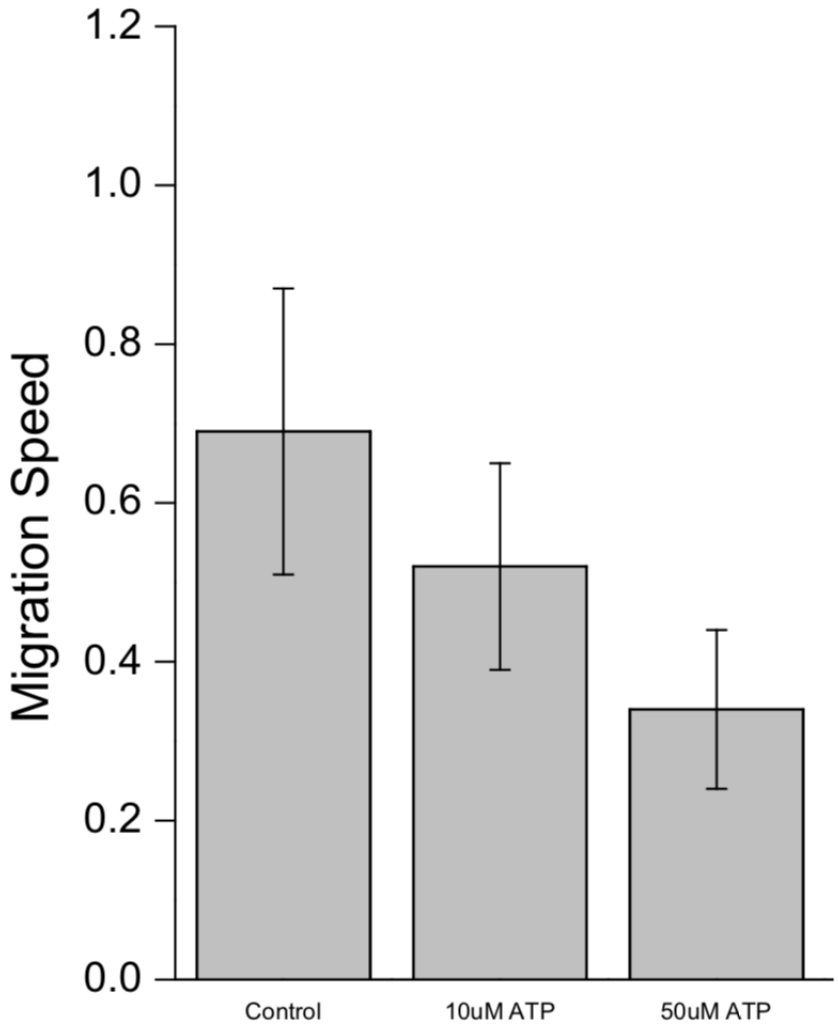


FIGURE 4.2: Calculation of cell migration persistence using manual tracking

Cell Culture\Accuracy	Period 1	Period 2	Period 3
Control	64%	62%	59%
$10\mu M$	54%	61%	63%
$50\mu M$	50%	56%	57%

TABLE 4.2: PCA classification result for single cell data

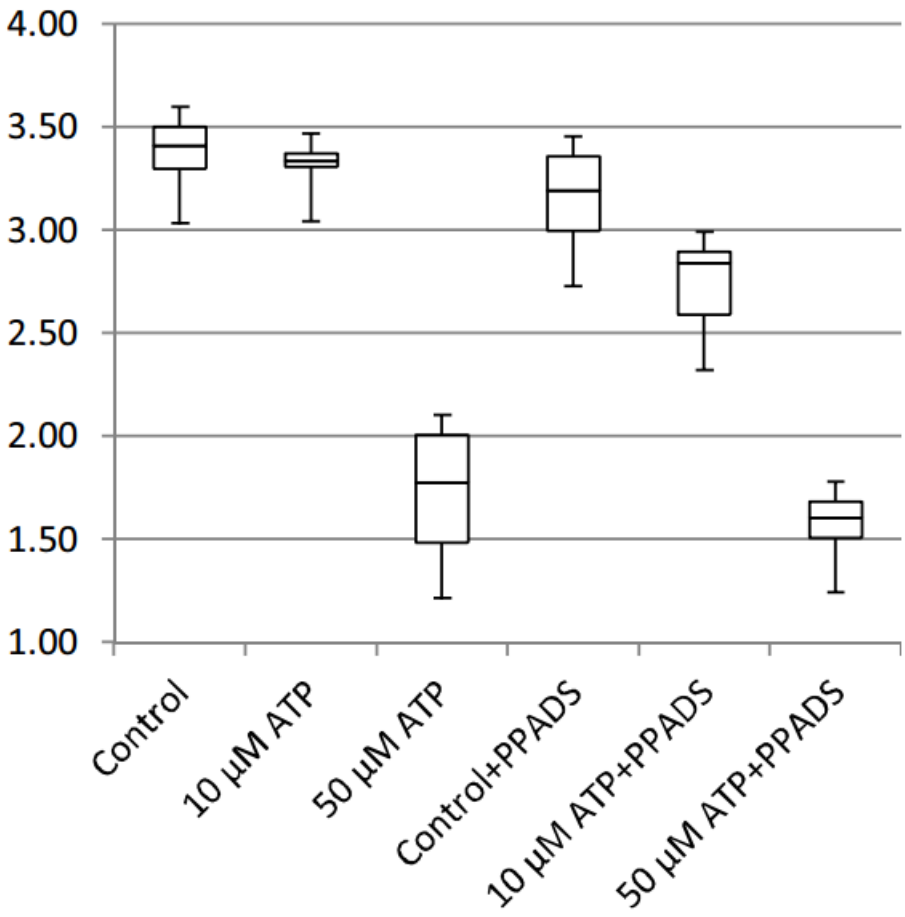


FIGURE 4.3: Post contact speed

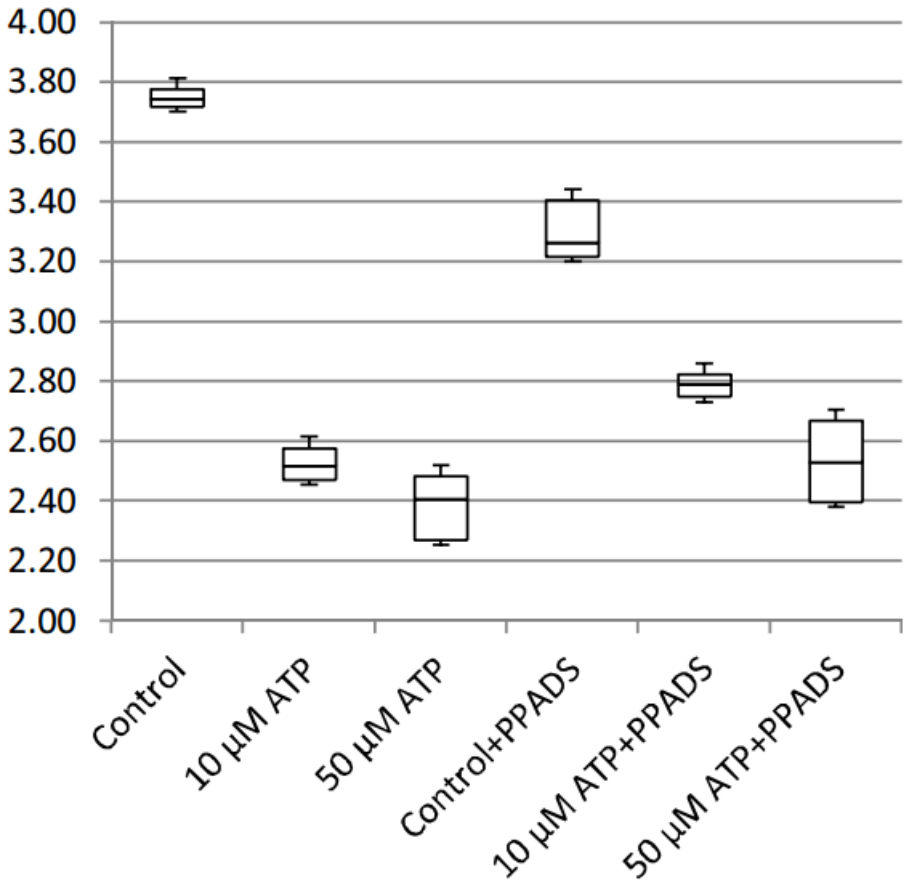


FIGURE 4.4: In-contact speed

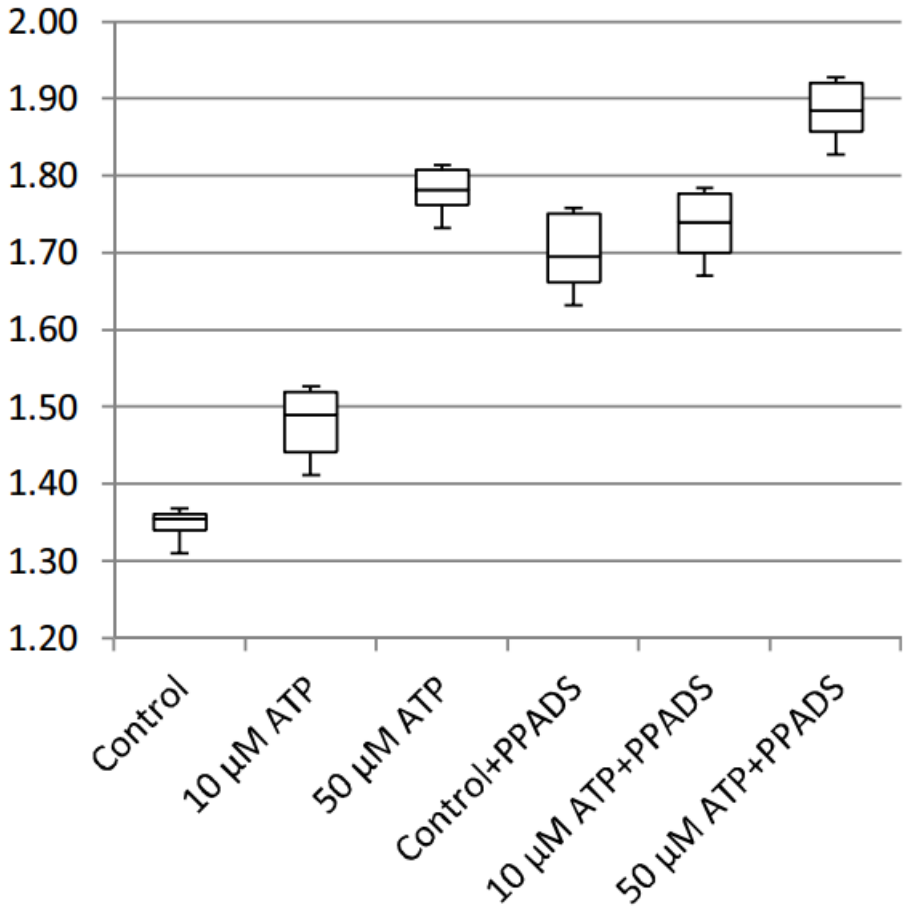


FIGURE 4.5: Angular velocity

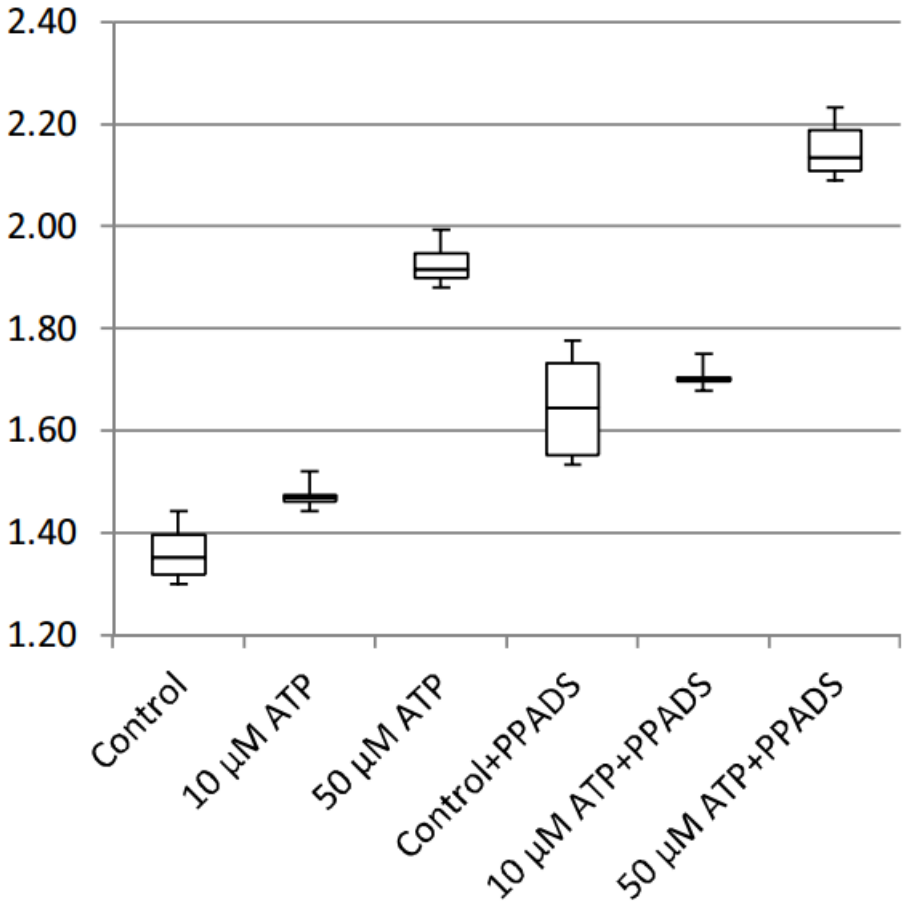


FIGURE 4.6: Post contact angular velocity

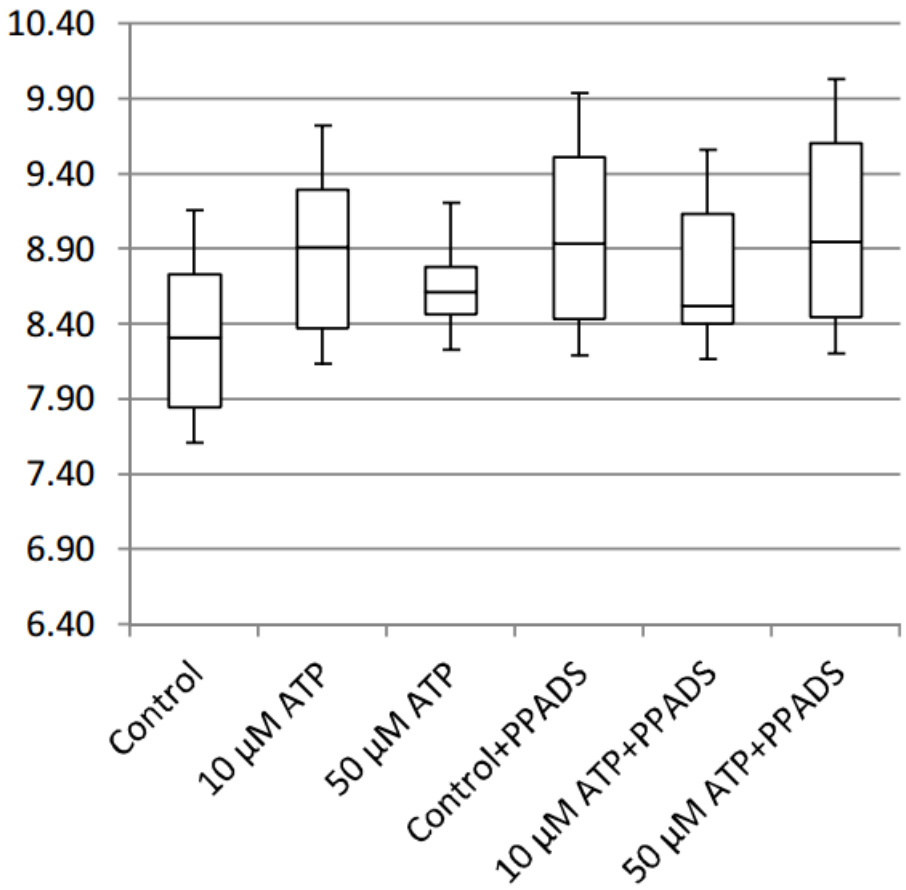


FIGURE 4.7: Average clump size

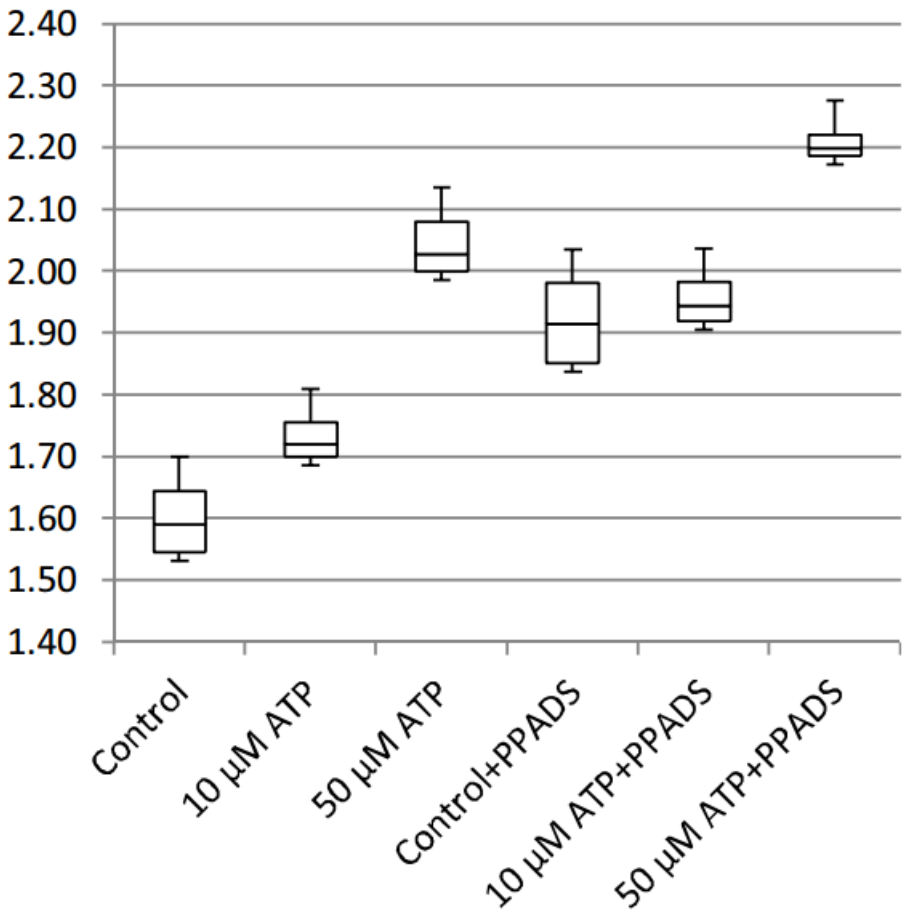


FIGURE 4.8: In-contact angular velocity

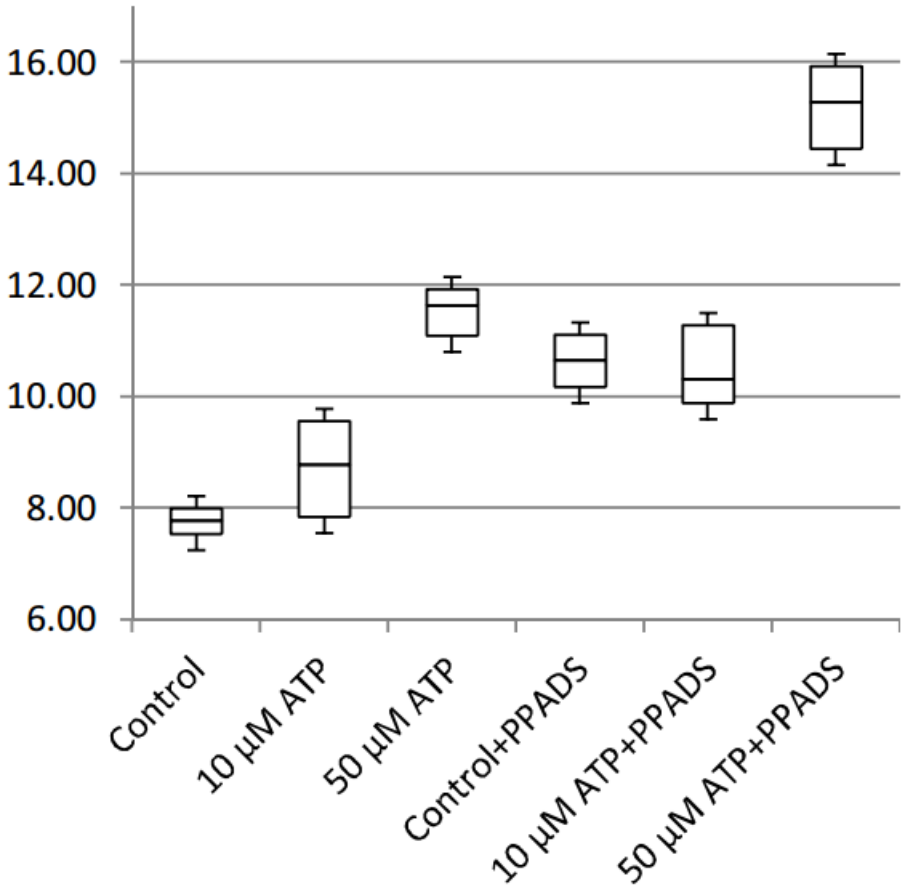


FIGURE 4.9: Average contact duration

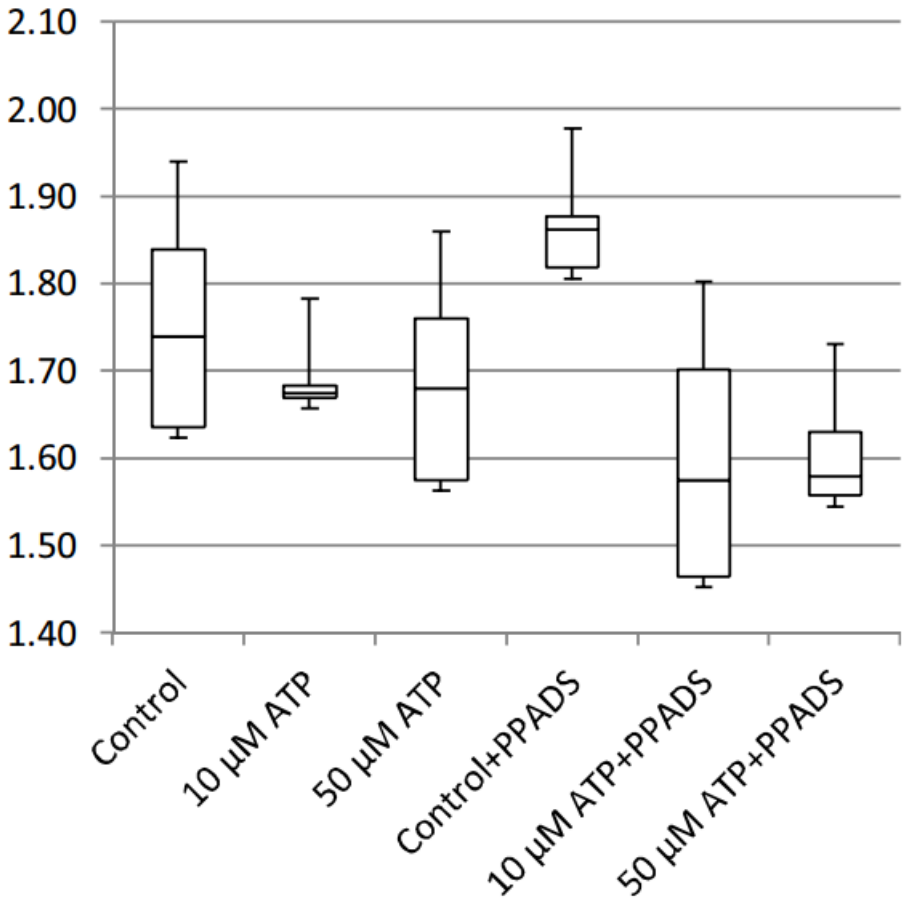


FIGURE 4.10: Cohesivity

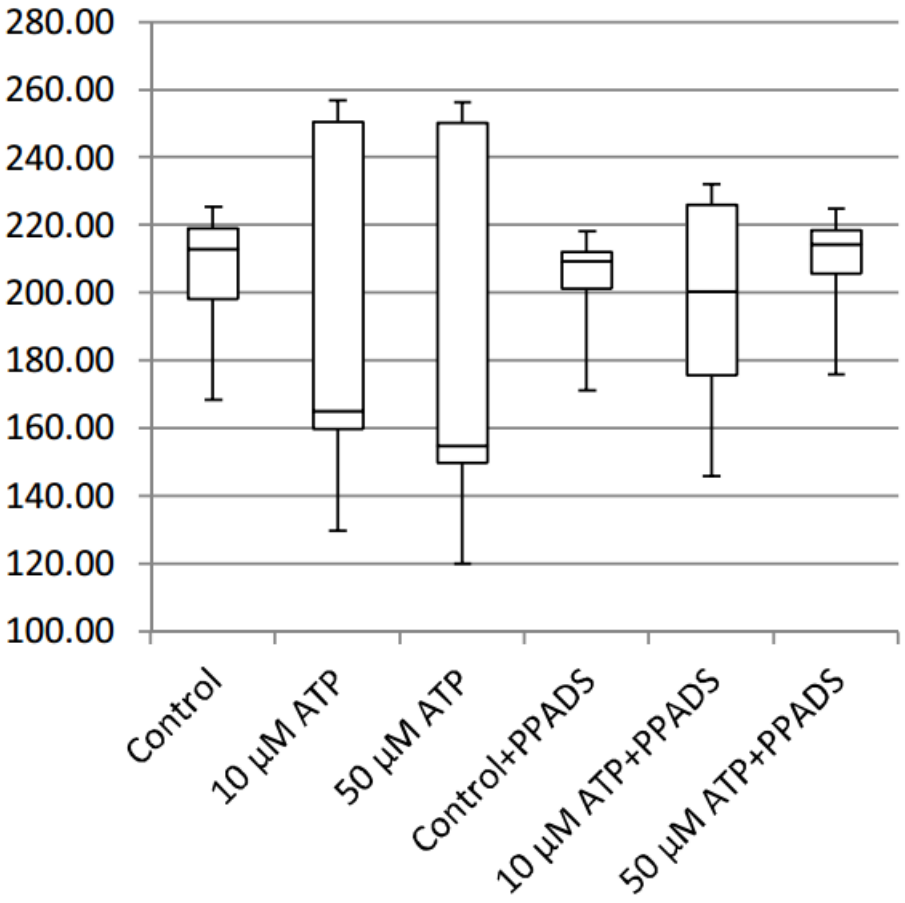


FIGURE 4.11: Cell count

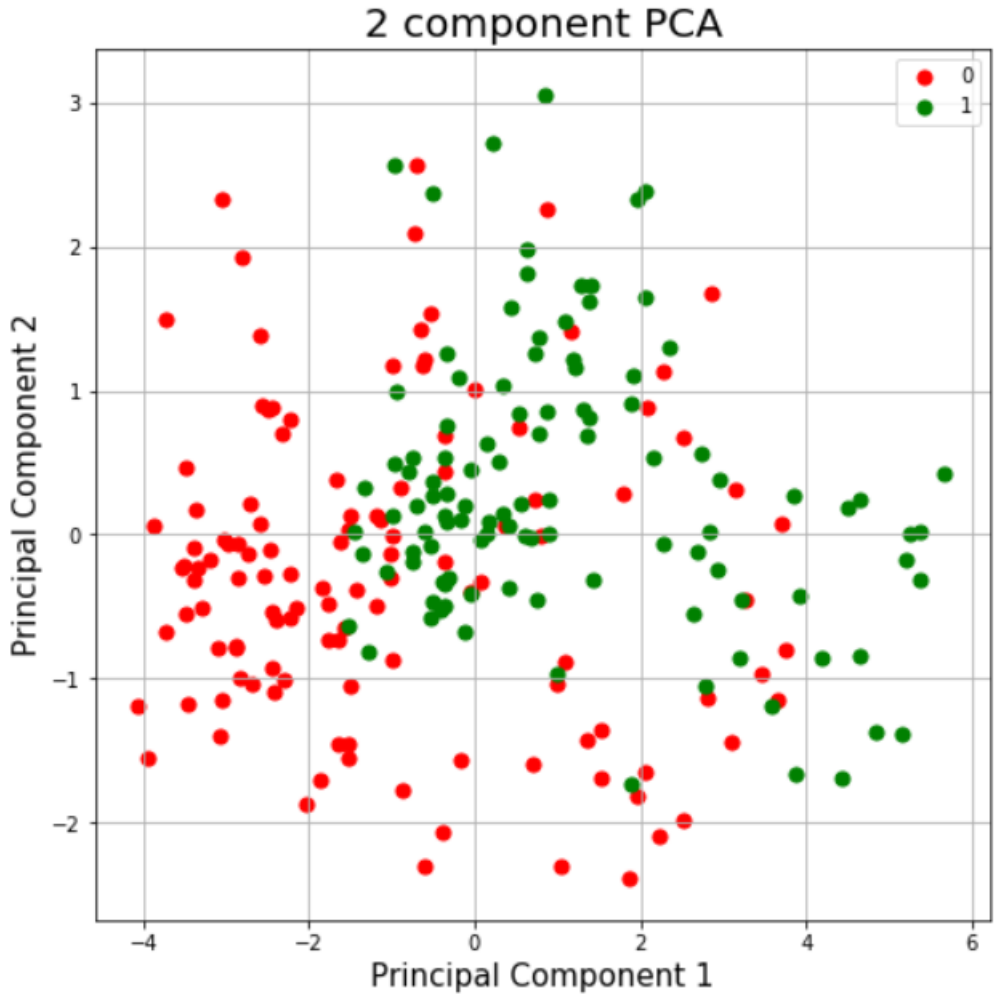


FIGURE 4.12: PCA result visualization of Cell culture dataset

4.3.2 Cell culture dataset

Figure 4.12 shows result of PCA and classification applied to cell culture dataset with inhibitor PPADS. The predictive accuracy for the training dataset is 75.35% and for the test dataset, 69.38%.

4.4 SVM results

Different kernel functions were evaluated to determine which is best for the target classification problem. Because the SVM classifier is black box and dataset is almost balanced between the two classes, the resulting metrics can be used to evaluate different kernel functions.

Cell Culture\Accuracy	Period 1	Period 2	Period 3
Control	68%	67%	65%
$10\mu M$	57%	64%	63%
$50\mu M$	60%	59%	59%

TABLE 4.3: SVM classification results for single cell data

Kernal Function	Linear	Quadratic	Cubic	Fine Gaussian	Medium Gaussian
Prediction Accuracy	85.3%	86.0%	86.9%	83.0%	87.8%

TABLE 4.4: SVM classification result for cell culture dataset with different kernel functions

4.4.1 Single cell dataset

Table 4.3 shows results for all nine single cell datasets, even though all available kernel functions were tested, the results are similar and SVM with medium Gaussian gives highest accuracy. Consequently, all following results for SVM employ the medium Gaussian kernel SVM.

4.4.2 Cell culture dataset

Table 4.4 shows SVM results for the cell culture dataset; five different kernel functions were used SVM, but again, the medium Gaussian SVM kernel gives the best predictive accuracy.

4.5 CGP results

Experiments were run to investigate parameterisation of CGP classifiers with respect to the following areas:

- ***Number of Generation***: number of generations specified for evolution of the classifier.
- ***Number of Nodes*** : Number of nodes in each chromosome.

This section discusses the results of each of these in turn, each factor with three different configurations of parameters and each experiment repeated twenty times. Performance of the classifier is evaluated in each case by classification accuracy with unseen test data set.

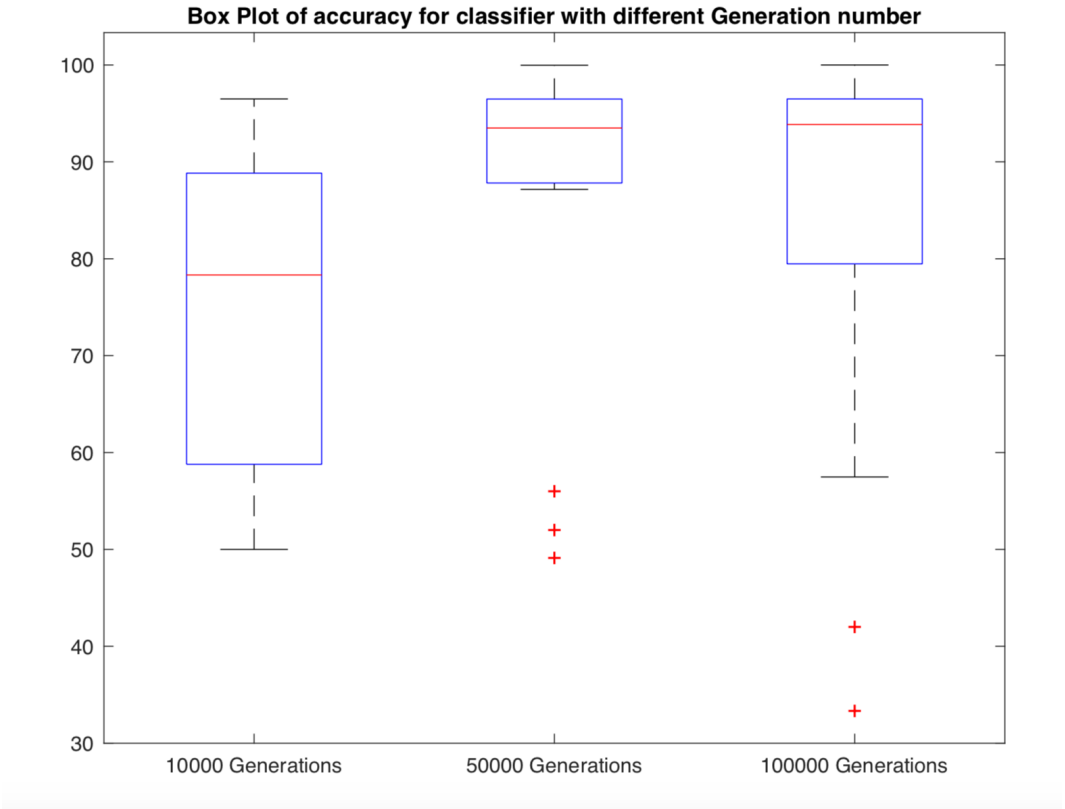


FIGURE 4.13: Box Plot of accuracy for classifiers with different Generation numbers

	10000 Generations	50000 Generations	100000 Generations
Mean Accuracy(%)	75.30	87.62	84.79
Standard Deviation	16.38	15.71	19.35

TABLE 4.5: Mean and standard deviation of accuracy of classifiers with different CGP generation

4.5.1 Choice of generation

In order to determine the best generation number for CGP, three different choices of generation numbers are considered, which are ten thousand, five thousand and one hundred thousand. For each choice of generation number CGP is run twenty times, and the classification accuracy of unseen test data recorded for each.

Table 4.2 shows average classification accuracy and standard deviation for three generation parameters; classifiers using a number of generations parameter of 50,000 generations has highest accuracy and lowest standard deviation.

The evolutionary nature of CGP suggests that with increasing number of

generations, the training set accuracy should continue to improve. However, higher training accuracy does not always mean a better classifier, because there is the possibility that the classifier has learnt the specific patterns which are evident in the limited training data rather than any overall relationships evident in the wider population. By increasing the number of generations to 100,000 , it can be seen that some classifiers have an accuracy of 98% or more but their corresponding test classification accuracy is much lower than the average value, which is a typical sign of overfitting. So in terms of experiment generation number choice, it is not always the larger number that generates the best overall result.

4.5.2 Preventing overfitting

In the process of creating classification networks, the most important issue is the performance of their response to unseen sets of data. There is effectively minimal benefit if a classifier is able to make a precise recognition without forecasting any new data accurately. The expression "overfitting" is given to this phenomenon which happens when the classifier has learnt particular patterns which are observable within the training data, which will not effectively generalise to unseen data. The separation of the data set into a test set and a training set is a basic and efficient technique of evaluating overfitting, with a typical performance of the test set being less than that of the training set as it evolves. Furthermore, it has the ability to attain a close precision on this unobserved data, thereby generalizing to a good standard. Nevertheless, since this system is unable to ensure that the test accurately represents that with unseen data, a basic, random assignment of the data into test and training sets may lead to a further bias factor in the outcome. The application of k-fold cross-validation is one technique of obtaining a reduction in this selection bias. The technique employs k equally-sized "folds" which represent portions of the dataset which are used to form the training and validation sets. Typically, one fold is used as the validation set and also the remaining k – 1 folds as the training set. The folds are then rotated for different runs of the evolutionary algorithm so that ultimately, each of the folds is used in both the training and validation sets. The final step is to average the validation fold scores of the fitness function, thereby enabling more comprehensive evaluations of the evolved classifiers.

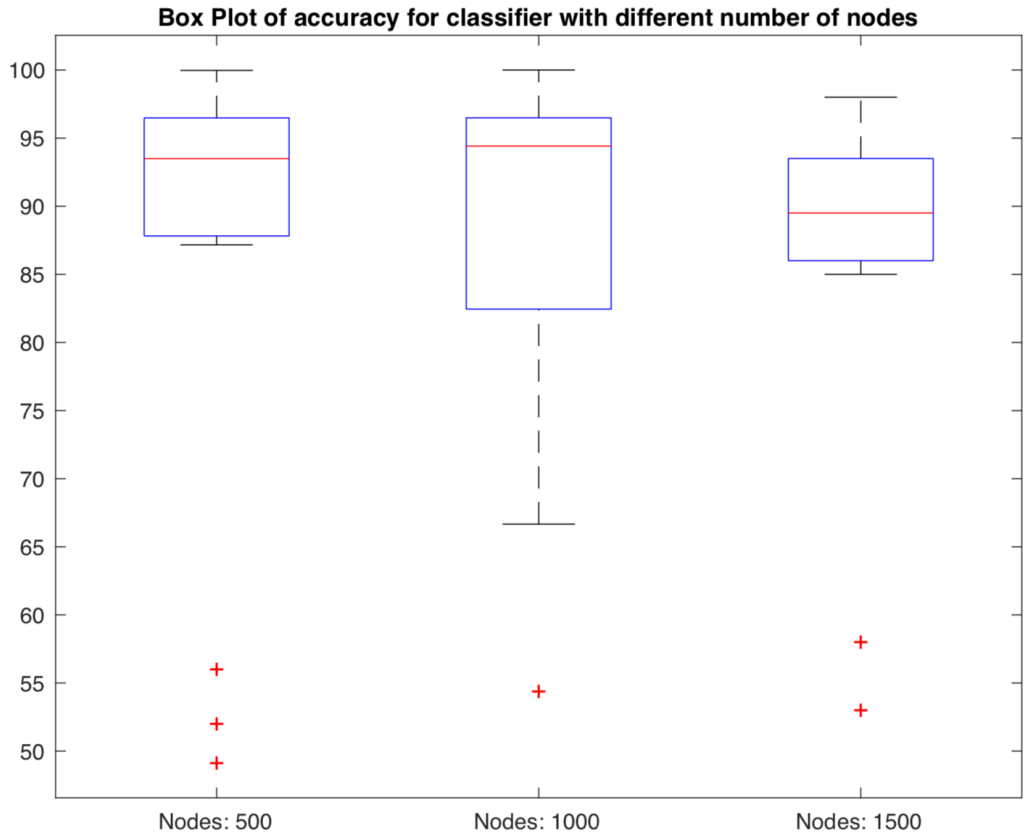


FIGURE 4.14: Box Plot of classifier accuracy with different number of Nodes

	500 Nodes	1000 Nodes	1500 Nodes
Mean Accuracy(%)	87.61	85.23	85.8
Standard Deviation	15.71	5.54	10.82

TABLE 4.6: Mean classification accuracy and standard deviation of classifiers with differing number of nodes.

4.5.3 Choice of number of nodes

Figure 4.14 shows the classification rate of classifiers with differing number of nodes. There is no big difference between three choice of node numbers because on average there are 97 active nodes in each evolved network; all three choices of number of nodes provides enough spare nodes to support effective mutation within each generation.

4.5.4 Analysis of evolved networks

The use of GCP has an additional benefit that the evolved classifier network can be easily defined as a conventional mathematical expression, which may

provide useful understanding of the role of the input data (and hence, ultimately, the biological processes) used in the classification procedure. For instance, in the example depicted in figure 4.15, It can be observe that this specific classifier depends upon inputs: (5),(6)and(8) which, with reference to Table 3.2, can be equated to features: in-contact angular velocity, post-contact angular velocity and average cell clump size, respectively.

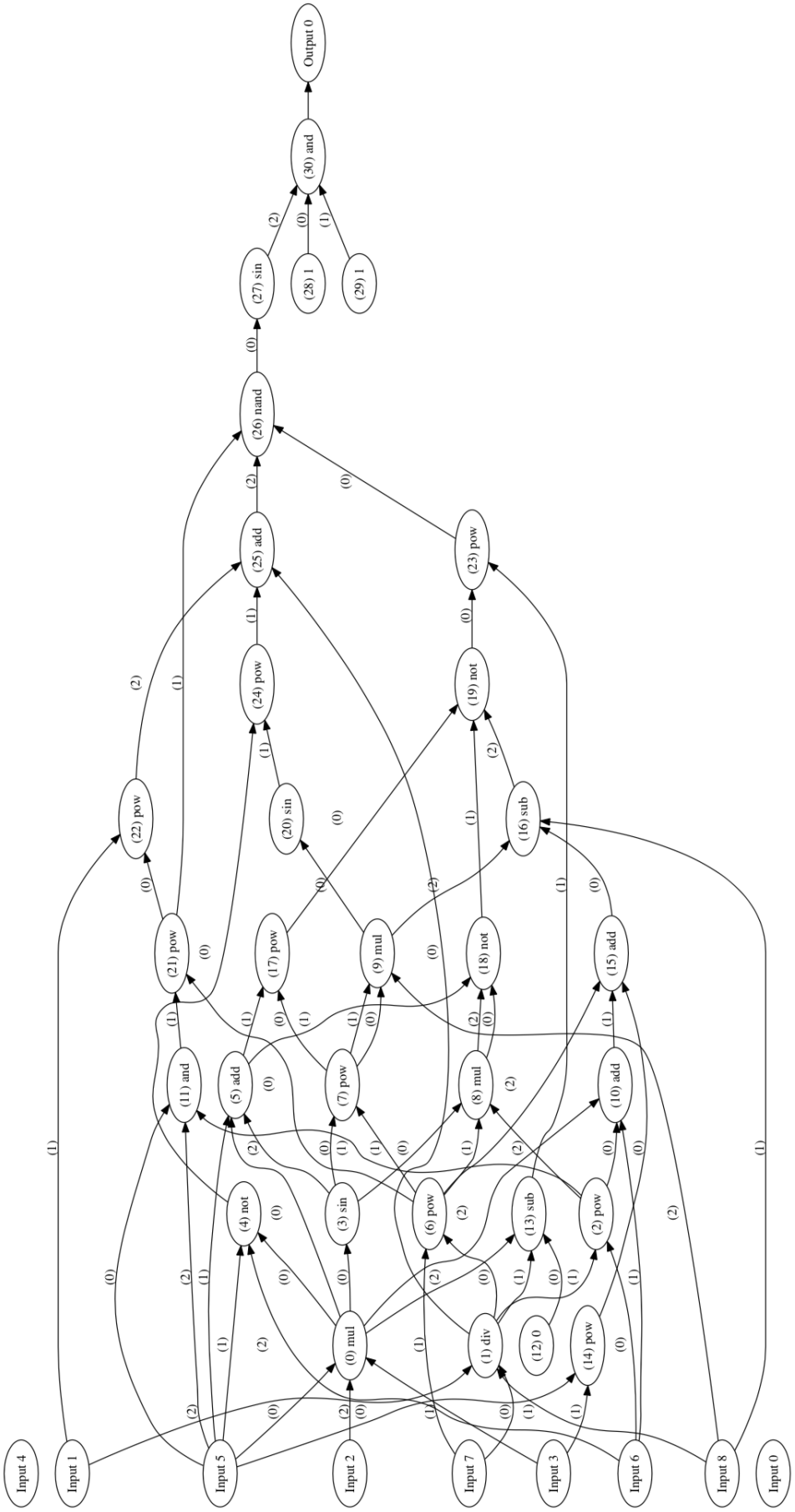


FIGURE 4.15: Example CGP

	Average Migration Speed	Post-Contact Migration Speed	In-Contact Migration Speed	Average Angular Velocity	Post-Contact Angular Velocity	In-Contact Angular Velocity	Average Cell Clump Size	Average Contact Duration	Cell Count	Accuracy of classifier
Mean	6.22	4.89	4.37	6.48	3.77	5.30	6.02	5.12	3.78	95.45
Standard Deviation	3.08	2.35	2.62	3.53	1.86	2.37	3.60	2.69	2.24	3.32

TABLE 4.7: Percentage of connections between input nodes and other nodes out of the total number of total nodes.

Table 4.7 shows the average and standard deviation of each input nodes as a percentage of all nodes in 20 classifiers with at least 90% accuracy. As can be seen in the table ***Average Migration speed ,Average Angular velocity*** and ***Average cell clump size*** are connected with at least six percent of total nodes directly.

4.6 Comparison of different Methods

In this section, the three classification algorithms employed, PCA, SVM and CGP are compared and contrasted.

When comparing the prediction accuracy, as shown by the respective values in table 4.8, it can be seen that there are certain close prediction accuracies between SVM and CGP for all ten datasets, where both of them are higher than PCA. In comparison, there is no clear difference that can be seen between SVM and CGP.

Other than the classification results, there is also another advantage of GP. After training, it can be theoretically inverted to the original functional expression, which is symbolic regression[129]. Therefore, the GP method can be used for the extrapolation and prediction of data. While traditional regression techniques such as SVM attempt to optimize the parameters for pre-specified model structures or kernel tricks, CGP avoids imposing a priori assumptions, and instead infers the models from the data. In other words, it aims to discover the model structure and the corresponding parameters. On the other hand, SVM has limited number of kernels, thus there is limited information that can be provided other than the classification.

Because of the nature of the two methods, CGP requires more computational calculations to evolve a network, whereas SVM possesses pre-defined kernels tricks and structures that requires less computational resources as compared with CGP.

4.7 Comparison of different datasets

Even though single cell datasets have much more training instances than the cell culture dataset, the results in section 4.6 shows that the cell culture dataset can achieve better classification results among all three methods, where the difference between two types of dataset lies in a relatively large

Dataset\Algorithm	PCA	SVM	CGP
Control P1	64%	68%	67%
Control P2	62%	67%	66%
Control P3	69%	65%	67%
10uM P1	54%	57%	59%
10uM P2	61%	64%	64%
10uM P3	63%	63%	63%
50uM P1	50%	60%	61%
50uM P2	56%	59%	61%
50uM P3	57%	59%	62%
Cell Culture	69%	87%	95%

TABLE 4.8: Overall results from each Machine learning algorithm where CGP result is average of 10 runs in order to get average performance.

margin. The reason behind this is that E-Catherin acts as both receptor and ligands during the cell development cycle[130]. Furthermore, E-Catherin can only be expressed through specific functions. In the cell, different cells may be deferentially expressed within the cell culture [131]. Therefore, the cells may undertake different roles and result in correspondingly different behaviours, simultaneously, even within the same cell culture, which means there are subcultures within a cell culture. While on the other hand, the cell culture dataset can characterize the whole cell behaviour.

4.8 Discussion

One major advantage of GP, and therefore CGP, is that only evolutionary parameters, such as the size of population, number of generations and the geometry of the network, are required to generate effective classifiers without *a priori* information regarding the feature space, other parameters of the objective function, human bias or missing input data. There is also the flexibility for new features to evolve through intrinsic connections between network nodes that is easy to specify mathematically open to human interpretation. Section 4.6 presents a close prediction performance between SVM and CGP, but CGP possesses the advantage of providing a white box classifier, as compared to the black box classifier of SVM. The goal of this research is not only focused on the classification, but also to understand the underlying characteristics of cell cultures. In summary, CGP can help establish a mathematics relationship among different features. These features

are not abstract as they have their own biological motivations and meanings. Therefore, CGP can provide more information for the characterization of urothelium cell cultures.

4.9 Conclusion

This chapter has presented the experimental results of the thesis, specifically, statistical analysis of extracted features from time-lapse spectroscopy, verified by manual tracking. Subsequent application of a classifier, CGP, demonstrated how evolutionary algorithms can be applied with a high classification rates for the cell culture dataset under investigation. This classification network consisted of nine inputs which are fed by features describing cellular physical attributes. The advantage of this type of network is it can be defined as mathematical equation which can help with understanding classification process and underlying biological behaviours.

Chapter 5

Conclusions and Further Work

Contents

5.1 Conclusion

In this dissertation, a new technique for the classification and the characterization of cell cultures has been presented by applying cell tracking to time-lapse videos. This thesis presents a method of describing cell behaviour in relation to extracted features such as migration persistence and migration speed, both in isolation and in the context of interaction with other cells, pre-contact, in-contact and post-contact. The methods developed provide a unique opportunity to deduce behaviour related to cell interactions by a totally automated means. Furthermore, the thesis has demonstrated how evolutionary algorithms may be utilised to successfully classify cell cultures , even if this is not clearly demonstrated by consideration of the extracted features themselves.

Through the consideration of normal human urothelial (NHU) cells which were studied in this thesis, it was mentioned that in the case of scratch assays, in which the repair of the injury caused to a confluent cell population is observed regarding the wound closure, the impact of the exogenous Adenosine triphosphate (ATP) was to improve the repair of the wound. It was demonstrated, in support of such observations, that the restraint of ATP breakdown improved the repair rate, but PPADs, being a selective antagonist of the P2X receptor, which was ATP-activated was preventative [25]. Such observations result in a prediction of the positive effect of ATP upon cell migration. Furthermore, this thesis did not anticipate that in sparse cell cultures, the impacts of exogenous ATP would cause a reduction in the migration speed. Such observations, together with the P2X's equivocal impact antagonist, PPADS, implies that the NHU cells' response to ATP could be more dependent upon context than was previously thought; furthermore, that other urothelial- expressed ATP-modulated receptors like the P2YG protein-coupled receptors [25] may have some relevance. The new analytical technique offered in this dissertation supplies the method by which by which predominantly influenced parameters as well as the responsible mechanisms may be totally analysed objectively and automatically.

5.2 Contributions

5.2.1 Feature Extraction

This thesis proposes feature extraction from time-lapsed spectroscopy based on biological motivation, including migration speed, angular velocity and population growth. in order to investigate the function of E-cadherin in urothelium cell cultures the thesis also proposed extracting dynamic features of cells particularly with respect to cell clumping and following contact with a cell clumping. Features were also extracted to describe the size of cell clumping and frequency of cells forming a clump.

The extracted average migration speed (figure 4.1) successfully separated cell culture classes with different concentration of Calcium. Other features extracted provide motivation for future research.

5.2.2 Application of CGP

The application of CGP in this work generated high accuracy classifiers (average accuracy of 95.45% from 20 networks). Unlike the other machine learning algorithms, CGP not only provides high performing classifiers, but can define the underlying network as discrete mathematical expressions. This means this approach not only classifies cell culture but also provides a means to characterise and understand the cell culture's physical properties.

5.3 Hypothesis Revisited

The hypothesis presented at the beginning of this thesis stated the following:

" ***Evolutionary Computation is an effective means of characterising and classifying urothelium cell cultures through time-lapse spectroscopy.***"

This work presented in thesis has provided the theoretical and experimental evidence to support the hypothesis and, particularly, the specific benefits of CGP to provide insight to the underlying biological mechanisms which will benefit future research.

5.4 Further work

The majority mathematical models of biochemical pathways take either signaling events that occur within isolated individual cells, or average cells that are thought to represent a population of cells. Similarly, experimental measurements are usually averaged over populations made up of hundreds of thousands of cells. This approach overtakes the fact that even within a genetically-homogeneous population, local conditions can affect cell signaling and result in phenotypic heterogeneity. Previous studies demonstrate that modeling of mixed cell populations can provide an insight into biological processes that may be non-intuitive. Eventually, it is excepted that understanding diverse cellular interactions will enhance our knowledge and understanding of the development of neoplasia in epithelial tissues and provide new insight into the design of treatment regimens.

Future work based on techniques presented in thesis, along with additional morphological and dynamic features provides the opportunity of building a model that can accurately characterise and describe cell culture behaviours and therefore help understand the biology behind those behaviours.

Appendix A

Appendix I

A.1 Model Evalution results

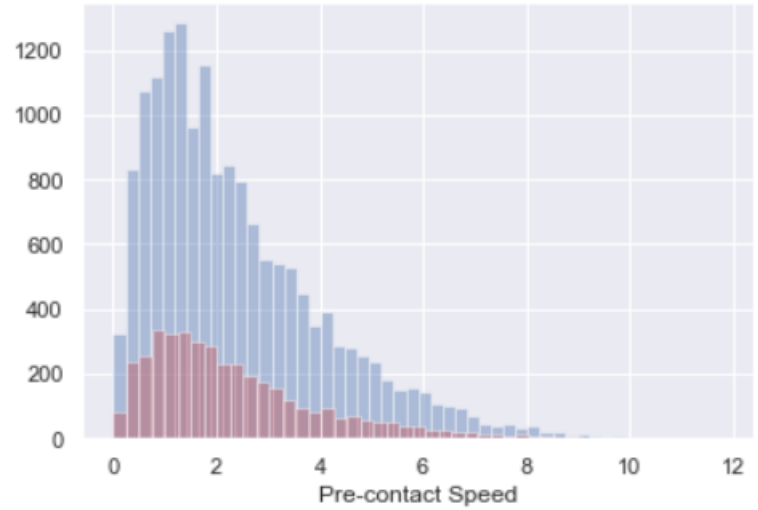


FIGURE A.1: Distrobuction of Pre-contact Speed in dataset where blue is training set and red is test set

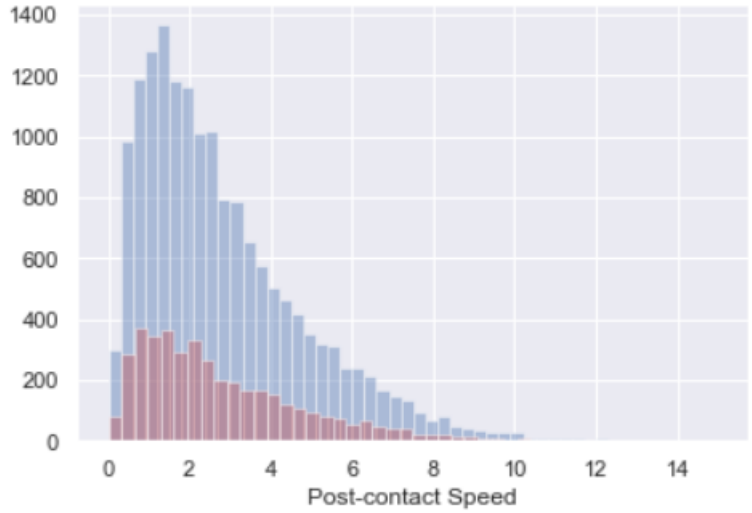


FIGURE A.2: Distrobuction of Post-contact Speed in dataset where blue is training set and red is test set

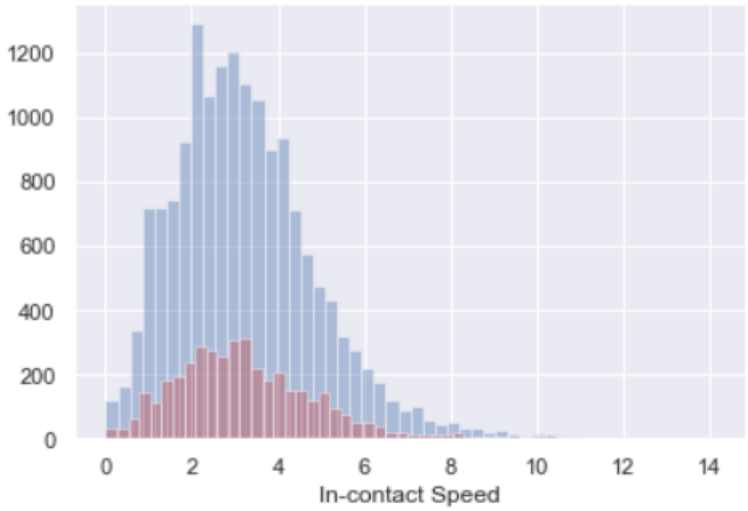


FIGURE A.3: Distrobuction of In-contact Speed in dataset where blue is training set and red is test set

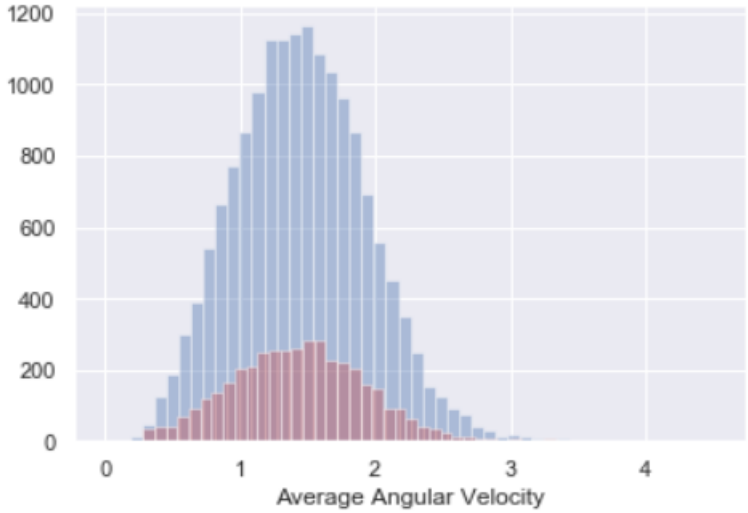


FIGURE A.4: Distrobuction of Average Angular Velocity in dataset where blue is training set and red is test set

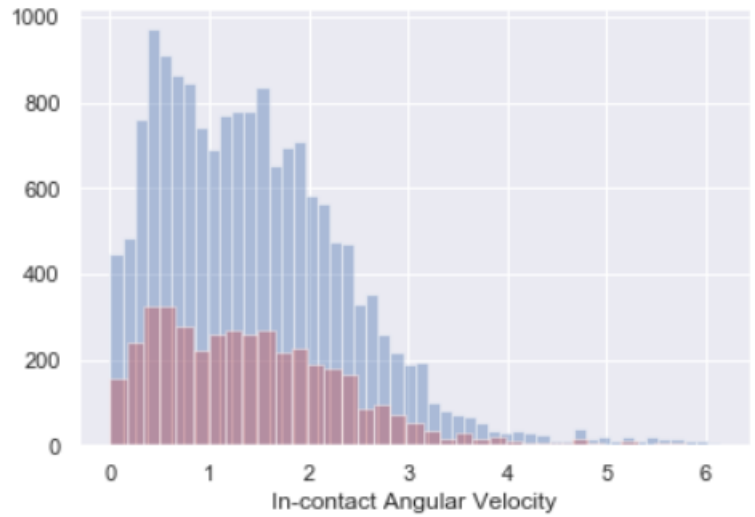


FIGURE A.5: Distrobuction of In-contact Angular Velocity in dataset where blue is training set and red is test set

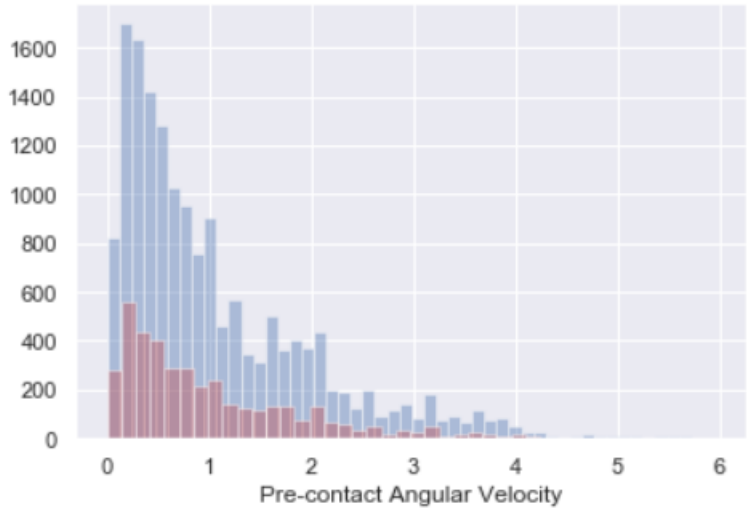


FIGURE A.6: Distrobuction of Pre-contact Angular Velocity in dataset where blue is training set and red is test set

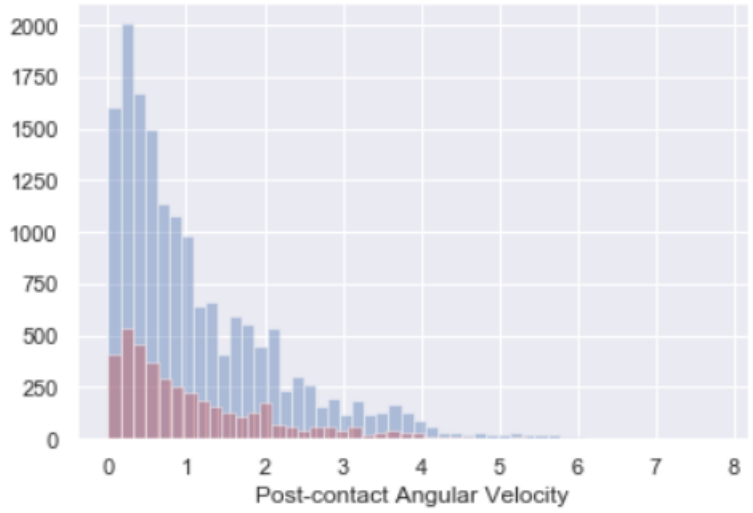


FIGURE A.7: Distrobuction of Post-contact Angular Velocity in dataset where blue is training set and red is test set

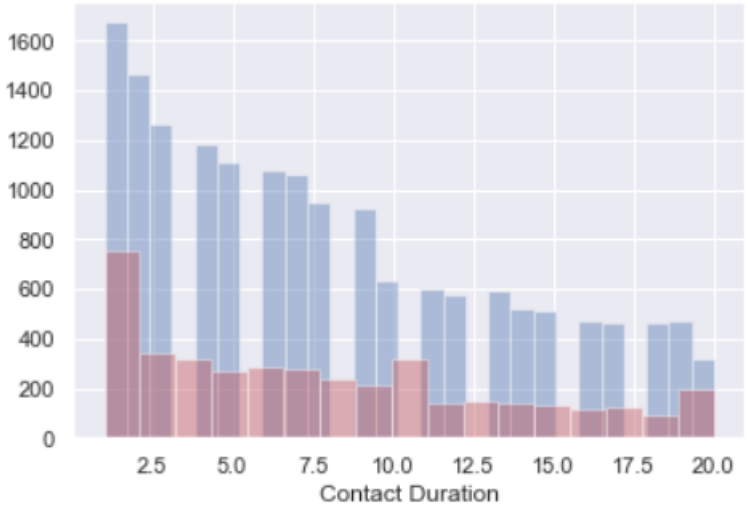


FIGURE A.8: Distrobuction of Contact Duration in dataset where blue is training set and red is test set

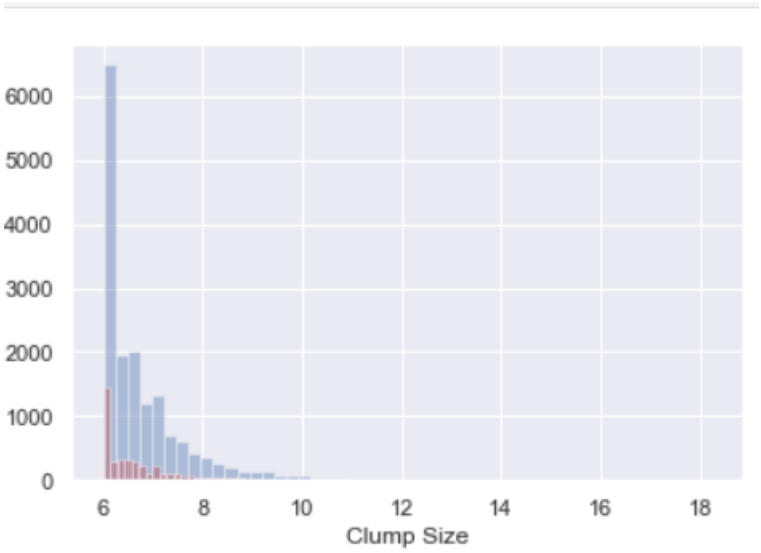


FIGURE A.9: Distrobuction of Clump Size in dataset where blue is training set and red is test set

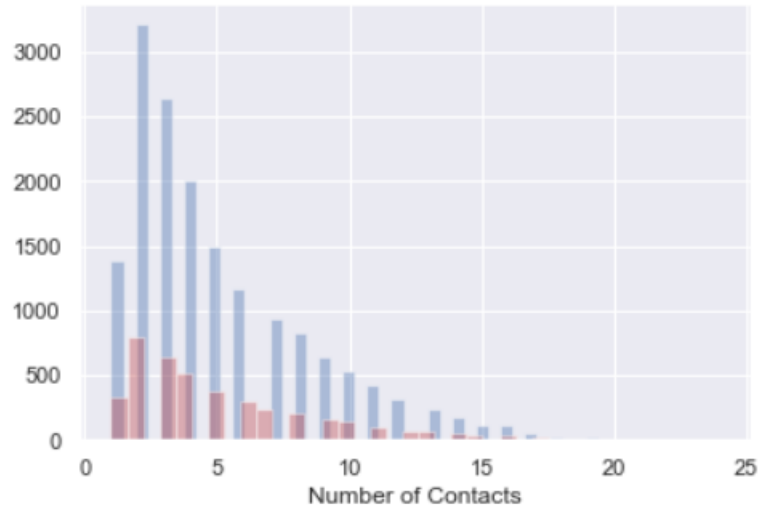


FIGURE A.10: Distrobuction of Number of Contacts Speed in dataset where blue is training set and red is test set

List of Abbreviations

LAH	**L**ist **A**bbreviations **H**ere
WSF	**W**hat (it) **S**tands **F**or
MATLAB	**MAT**rix **LAB**oratory
NHU	**N**ormal **H**uman **U**rothelial
ATP	**A**denosine **T**ri**P**hosphate
CGP	**C**artesian **G**enetic **P**rogramming
PPADS	**P***yridoxal***P***hosphate* − 6 − **A***zophenyl* − 2′,4′ − **D***isulphonicacid*
CCD	**C**harge **C**oupled **D**evice
ANOVA	**AN**alysis **O**f **VA**riance
DNA	**D**eoxyribo**N**ucleic **A**cid
AKT	Protein kinase B
ANN	**A**rtificial **N**eural **N**etwork
RNN	**R**ecurrent **N**eural **N**etwork
LTSM	**L**ong **S**hort-term **M**emory
EA	**E**volutionary **A**lgorithms
GP	**g**enetic **P**rogramming
GPTP	**G**enetic **P**rogramming **T**heory and **P**ractice

Reference

[1] M. Burkitt, D Walker, D. M. Romano, and A. Fazeli, "Modelling sperm behaviour in a 3d environment", in *Proceedings of the 9th International Conference on Computational Methods in Systems Biology*, ACM, 2011, pp. 141–149.

[2] M Scianna and L Preziosi, *Cellular potts models: Multiscale developments and biological applications*, 2013.

[3] A. Deutsch and S. Dormann, "Cellular automaton modeling of biological pattern formation", *FASEB*, vol. 23, p. 12, 2005.

[4] J. A. Walker, K. Völk, S. L. Smith, and J. F. Miller, "Parallel evolution using multi-chromosome cartesian genetic programming", *Genetic Programming and Evolvable Machines*, vol. 10, no. 4, p. 417, 2009.

[5] R. Chatterjee, M. Ghosh, A. S. Chowdhury, and N. Ray, "Cell tracking in microscopic video using matching and linking of bipartite graphs", *Computer methods and programs in biomedicine*, vol. 112, no. 3, pp. 422–431, 2013.

[6] Z. Zhang, M. Bedder, S. L. Smith, D. Walker, S. Shabir, and J. Southgate, "Characterization and classification of adherent cells in monolayer culture using automated tracking and evolutionary algorithms", *Biosystems*, vol. 146, pp. 110–121, 2016.

[7] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter, "Cell movements and the shaping of the vertebrate body", 2002.

[8] A. Bruce. (). What is a cell, [Online]. Available: https://web.archive.org/web/20130507094245/http://www.ncbi.nlm.nih.gov:80/About/primer/genetics_cell.html.

[9] G. Karp, *Cell and molecular biology: concepts and experiments*. John Wiley & Sons, 2009.

[10] A. Maton, D. Lahart, J. Hopkins, M. Q. Warner, S. Johnson, and J. D. Wright, *Cells: Building blocks of life*. Pearson Prentice Hall, 1997.

[11] W. Commons, *File:celltypes.svg — wikimedia*, [Online; accessed 24-September-2018], 2018. [Online]. Available: https://commons.wikimedia.org/w/index.php?title=File:Celltypes.svg&oldid=314722701.

[12] Wikipedia contributors, *Cell (biology) — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Cell_(biology)&oldid=849739515, [Online; accessed 11-July-2018], 2018.

[13] ——, *Cell culture - Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Cell_culture&oldid=847497293, [Online; accessed 11-July-2018], 2018.

[14] L. Qian and W. M. Saltzman, "Improving the expansion and neuronal differentiation of mesenchymal stem cells through culture surface modification", *Biomaterials*, vol. 25, no. 7-8, pp. 1331–1337, 2004.

[15] G. Maguire, *Therapeutics from adult stem cells and the hype curve*, 2016.

[16] M. Cao, B. Liu, G. Cunha, and L. Baskin, "Urothelium patterns bladder smooth muscle location", *Pediatric research*, vol. 64, no. 4, p. 352, 2008.

[17] S. N. Nabili, *Kidney infection (pyelonephritis) signs, symptoms & home remedies*. [Online]. Available: https://www.emedicinehealth.com/kidney_infection/article_em.htm#what_is_kidney_a_infection_pyelonephritis.

[18] D. St-Onge, *Analyse de la valeur pronostique du contexte immunologique du cancer superficiel de la vessie*. [Online]. Available: https://corpus.ulaval.ca/jspui/bitstream/20.500.11794/27009/1/32491.pdf.

[19] P. Kreft and K. Jezernik, "Urothelial injuries and the early wound healing response: Tight junctions and urothelial cytodifferentiation", *Histochemistry and cell biology*, vol. 123, no. 4-5, pp. 529–539, 2005.

[20] X.-R. Wu, X.-P. Kong, A. Pellicer, G. Kreibich, and T.-T. Sun, "Uroplakins in urothelial biology, function, and disease", *Kidney international*, vol. 75, no. 11, pp. 1153–1165, 2009.

[21] M. Nguyen, D. Lieu, L. Degraffenried, R. Isseroff, and E. Kurzrock, "Urothelial progenitor cells: Regional differences in the rat bladder", *Cell proliferation*, vol. 40, no. 2, pp. 157–165, 2007.

[22] M. Erman and K. Jezernik, "Superficial cell differentiation during embryonic and postnatal development of mouse urothelium", *Tissue and Cell*, vol. 38, no. 5, pp. 293–301, 2006.

[23] W. Commons, *File:illu bladder.jpg — wikimedia*, [Online; accessed 24-September-2018], 2016. [Online]. Available: \url{https://commons.wikimedia.org/w/index.php?title=File:Illu_bladder.jpg&oldid=226940891}.

[24] L. A. Birder, "Urothelial signaling", in *Urinary Tract*, Springer, 2011, pp. 207–231.

[25] S. Shabir, W. Cross, L. A. Kirkwood, J. F. Pearson, P. A. Appleby, D. Walker, I. Eardley, and J. Southgate, "Functional expression of purinergic p2 receptors and transient receptor potential channels by the human urothelium", *American Journal of Physiology-Renal Physiology*, vol. 305, no. 3, F396–F406, 2013.

[26] L. A. Birder and W. C. De Groat, "Mechanisms of disease: Involvement of the urothelium in bladder dysfunction", *Nature Reviews Urology*, vol. 4, no. 1, p. 46, 2007.

[27] W. Yu and W. G. Hill, "Defining protein expression in the urothelium: A problem of more than transitional interest", *American Journal of Physiology-Renal Physiology*, vol. 301, no. 5, F932–F942, 2011.

[28] N. T. Georgopoulos, L. A. Kirkwood, and J. Southgate, "A novel bidirectional positive-feedback loop between wnt–catenin and egfr–erk plays a role in context-specific modulation of epithelial tissue regeneration", *J Cell Sci*, jcs–150 888, 2014.

[29] P. Hu, S. Meyers, F.-X. Liang, F.-M. Deng, B. Kachar, M. L. Zeidel, and T.-T. Sun, "Role of membrane proteins in permeability barrier function: Uroplakin ablation elevates urothelial permeability", *American Journal of Physiology-Renal Physiology*, vol. 283, no. 6, F1200–F1207, 2002.

[30] N. J. Smith, J. Hinley, C. L. Varley, I. Eardley, L. K. Trejdosiewicz, and J. Southgate, "The human urothelial tight junction: Claudin 3 and the switch", *Bladder*, vol. 2, no. 1, e9, 2015.

[31] C. Varley, G. Hill, S. Pellegrin, N. J. Shaw, P. J. Selby, L. K. Trejdosiewicz, and J. Southgate, "Autocrine regulation of human urothelial cell proliferation and migration during regenerative responses in vitro", *Experimental cell research*, vol. 306, no. 1, pp. 216–229, 2005.

[32] S. C. Baker, S. Shabir, and J. Southgate, "Biomimetic urothelial tissue models for the in vitro evaluation of barrier physiology and bladder drug efficacy", *Molecular pharmaceutics*, vol. 11, no. 7, pp. 1964–1970, 2014.

[33] J. Southgate, K. Hutton, D. Thomas, and L. K. Trejdosiewicz, "Normal human urothelial cells in vitro: Proliferation and induction of stratification.", *Laboratory investigation; a journal of technical methods and pathology*, vol. 71, no. 4, pp. 583–594, 1994.

[34] N. T. Georgopoulos, L. A. Kirkwood, D. C. Walker, and J. Southgate, "Differential regulation of growth-promoting signalling pathways by e-cadherin", *PLoS One*, vol. 5, no. 10, e13621, 2010.

[35] P. Friedl and D. Gilmour, "Collective cell migration in morphogenesis, regeneration and cancer", *Nature reviews Molecular cell biology*, vol. 10, no. 7, p. 445, 2009.

[36] S. Ripke, A. R. Sanders, K. S. Kendler, D. F. Levinson, P. Sklar, P. A. Holmans, D.-Y. Lin, J. Duan, R. A. Ophoff, O. A. Andreassen, *et al.*, "Genome-wide association study identifies five new schizophrenia loci", *Nature genetics*, vol. 43, no. 10, p. 969, 2011.

[37] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.", 2008.

[38] S. Gay, S. Soliman, and F. Fages, "A graphical method for reducing and relating models in systems biology", *Bioinformatics*, vol. 26, no. 18, pp. i575–i581, 2010.

[39] S. Hoops, S. Sahle, R. Gauges, C. Lee, J. Pahle, N. Simus, M. Singhal, L. Xu, P. Mendes, and U. Kummer, "Copasi—a complex pathway simulator", *Bioinformatics*, vol. 22, no. 24, pp. 3067–3074, 2006.

[40] S. Sahle, R. Gauges, J. Pahle, N. Simus, U. Kummer, S. Hoops, C. Lee, M. Singhal, L. Xu, and P. Mendes, "Simulation of biochemical networks using copasi-a complex pathway simulator", in *Simulation Conference, 2006. WSC 06. Proceedings of the Winter*, IEEE, 2006, pp. 1698–1706.

[41] A. Regev, E. M. Panina, W. Silverman, L. Cardelli, and E. Shapiro, "Bioambients: An abstraction for biological compartments", *Theoretical Computer Science*, vol. 325, no. 1, pp. 141–167, 2004.

[42] G. Păun and G. Rozenberg, "A guide to membrane computing", *Theoretical Computer Science*, vol. 287, no. 1, pp. 73–100, 2002.

[43] V. A. Muganthan, A. Phillips, and M. G. Vigliotti, "Bam: Bioambient machine.", in *ACSD*, Citeseer, 2008, pp. 45–49.

[44] H. Pilegaard, F. Nielson, and H. R. Nielson, "Pathway analysis for bioambients", *The Journal of Logic and Algebraic Programming*, vol. 77, no. 1-2, pp. 92–130, 2008.

[45] L. Cardelli, "Abstract machines of systems biology", in *Transactions on Computational Systems Biology III*, Springer, 2005, pp. 145–168.

[46] E. Merelli, G. Armano, N. Cannata, F. Corradini, M. d'Inverno, A. Doms, P. Lord, A. Martin, L. Milanesi, S. Möller, *et al.*, "Agents in bioinformatics, computational and systems biology", *Briefings in bioinformatics*, vol. 8, no. 1, pp. 45–59, 2006.

[47] E. Bartocci, D. Cacciagrano, N. Cannata, F. Corradini, E. Merelli, L. Milanesi, and P. Romano, "An agent-based multilayer architecture for bioinformatics grids", *IEEE transactions on Nanobioscience*, vol. 6, no. 2, pp. 142–148, 2007.

[48] P. Richmond, D. Walker, S. Coakley, and D. Romano, "High performance cellular level agent-based simulation with flame for the gpu", *Briefings in bioinformatics*, vol. 11, no. 3, pp. 334–347, 2010.

[49] N. Paoletti, P. Lio, E. Merelli, and M. Viceconti, "Multilevel computational modeling and quantitative analysis of bone remodeling", *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, vol. 9, no. 5, pp. 1366–1378, 2012.

[50] M. J. North, N. T. Collier, and J. R. Vos, "Experiences creating three implementations of the repast agent modeling toolkit", *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol. 16, no. 1, pp. 1–25, 2006.

[51] H. A. Bethe, "Statistical theory of superlattices", *Proc. R. Soc. Lond. A*, vol. 150, no. 871, pp. 552–575, 1935.

[52] J. A. Izaguirre, R. Chaturvedi, C. Huang, T. Cickovski, J Coffland, G Thomas, G. Forgacs, M Alber, G Hentschel, S. A. Newman, *et al.*, "Compucell, a multi-model framework for simulation of morphogenesis", *Bioinformatics*, vol. 20, no. 7, pp. 1129–1137, 2004.

[53] S. D. Hester, J. M. Belmonte, J. S. Gens, S. G. Clendenon, and J. A. Glazier, "A multi-cell, multi-scale model of vertebrate segmentation and somite formation", *PLoS computational biology*, vol. 7, no. 10, e1002155, 2011.

[54] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited, 2016.

[55] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*. MIT press, 2012.

[56] M. van Gerven and S. Bohte, *Artificial neural networks as models of neural information processing*. Frontiers Media SA, 2018.

[57] Techsmosh, *A digestible overview of neural networks*, 1970. [Online]. Available: https://dev.to/techsmosh/a-digestible-overview-of-neural-networks-2mg9.

[58] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity", *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[59] D. O. Hebb, *The organization of behavior: A neurophysiological approach*, 1949.

[60] B. Farley and W Clark, "Simulation of self-organizing systems by digital computer", *Transactions of the IRE Professional Group on Information Theory*, vol. 4, no. 4, pp. 76–84, 1954.

[61] N. Rochester, J Holland, L Haibt, and W Duda, "Tests on a cell assembly theory of the action of the brain, using a large digital computer", *IRE Transactions on information Theory*, vol. 2, no. 3, pp. 80–93, 1956.

[62] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.", *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[63] P. Werbos, "Beyond regression:" new tools for prediction and analysis in the behavioral sciences", *Ph. D. dissertation, Harvard University*, 1974.

[64] D. H. Hubel and T. N. Wiesel, *Brain and visual perception: the story of a 25-year collaboration*. Oxford University Press, 2004.

[65] J. Schmidhuber, "Deep learning in neural networks: An overview", *Neural networks*, vol. 61, pp. 85–117, 2015.

[66] M. Minsky and S. Papert, *Perceptron expanded edition*, 1969.

[67] D. E. Rumelhart and J. L. McClelland, "Parallel distributed processing: Explorations in the microstructure of cognition. volume 1. foundations", 1986.

[68] B. Rost and C. Sander, "Prediction of protein secondary structure at better than 70% accuracy", *Journal of molecular biology*, vol. 232, no. 2, pp. 584–599, 1993.

[69] J. J. Weng, N. Ahuja, and T. S. Huang, "Learning recognition and segmentation using the cresceptron", *International Journal of Computer Vision*, vol. 25, no. 2, pp. 109–143, 1997.

[70] D. Scherer, A. Müller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition", in *Artificial Neural Networks–ICANN 2010*, Springer, 2010, pp. 92–101.

[71] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber, *et al.*, *Gradient flow in recurrent nets: The difficulty of learning long-term dependencies*, 2001.

[72] J. Schmidhuber, "Learning complex, extended sequences using the principle of history compression", *Neural Computation*, vol. 4, no. 2, pp. 234–242, 1992.

[73] S. Behnke, *Hierarchical neural networks for image interpretation*. Springer, 2003, vol. 2766.

[74] G. E. Hinton, "Deep belief networks", *Scholarpedia*, vol. 4, no. 5, p. 5947, 2009.

[75] Q. V. Le, "Building high-level features using large scale unsupervised learning", in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, IEEE, 2013, pp. 8595–8598.

[76] *Recurrent neural networks*, 2018. [Online]. Available: https://machinelearningblog.com/2018/02/21/recurrent-neural-networks/.

[77] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space", *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[78] C. Cortes and V. Vapnik, "Support-vector networks", *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[79] T. Joachims, "Making large-scale svm learning practical", Technical report, SFB 475: Komplexitätsreduktion in Multivariaten ..., Tech. Rep., 1998.

[80] J. Qin and Z.-S. He, "A svm face recognition method based on gabor-featured key points", in *2005 international conference on machine learning and cybernetics*, IEEE, vol. 8, 2005, pp. 5144–5149.

[81] A. Sun, E.-P. Lim, and W.-K. Ng, "Web classification using support vector machine", in *Proceedings of the 4th international workshop on Web information and data management*, ACM, 2002, pp. 96–99.

[82] A. M. Turing, "Computing machinery and intelligence", in *Parsing the Turing Test*, Springer, 2009, pp. 23–65.

[83] R. Forsyth, "Beagle—a darwinian approach to pattern recognition", *Kybernetes*, vol. 10, no. 3, pp. 159–166, 1981.

[84] J. R. Koza, *Non-linear genetic algorithms for solving problems*, US Patent 4,935,877, 1990.

[85] ——, "Hierarchical genetic algorithms operating on populations of computer programs.", in *IJCAI*, vol. 89, 1989, pp. 768–774.

[86] D. E. Goldberg and J. H. Holland, "Genetic algorithms and machine learning", *Machine learning*, vol. 3, no. 2, pp. 95–99, 1988.

[87] D. E. Goldberg *et al.*, "Computer-aided gas pipeline operation using genetic algorithms and rule learning part ii: Rule learning control of a pipeline under normal and abnormal conditions", 1985.

[88] J. R. Koza, "Genetic programming as a means for programming computers by natural selection", *Statistics and computing*, vol. 4, no. 2, pp. 87–112, 1994.

[89] J. R. Koza and J. P. Rice, *Genetic Programming:The Movie*. Cambridge, MA, USA: MIT Press, 1992.

[90] T. Hu, W. Banzhaf, and J. H. Moore, "The effects of recombination on phenotypic exploration and robustness in evolution", *Artificial Life*, vol. 20, no. 4, pp. 457–470, Oct. 2014, Ten thousandth GP entry in the genetic programming bibliography, ISSN: 1064-5462. DOI: doi:10.1162/ARTL_a_00145.

[91] J. R. Koza, "Human-competitive results produced by genetic programming", *Genetic Programming and Evolvable Machines*, vol. 11, no. 3/4, pp. 251–284, Sep. 2010, Tenth Anniversary Issue: Progress in Genetic Programming and Evolvable Machines, ISSN: 1389-2576. DOI: doi:10.1007/s10710-010-9112-3. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.297.6227.

[92] J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, Eds., *Genetic Programming 1996: Proceedings of the First Annual Conference*, Stanford University, CA, USA: MIT Press, 1996. [Online]. Available: http://www.genetic-programming.org/gp96proceedings.html.

[93] W. B. Langdon, *Genetic Programming and Data Structures: Genetic Programming + Data Structures = Automatic Programming!*, ser. Genetic Programming. Boston: Kluwer, 1998, vol. 1, ISBN: 0-7923-8135-1. DOI: doi:10.1007/978-1-4615-5731-9. [Online]. Available: http://www.cs.ucl.ac.uk/staff/W.Langdon/gpdata.

[94] W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone, *Genetic Programming – An Introduction; On the Automatic Evolution of Computer Programs and its Applications*. San Francisco, CA, USA: Morgan Kaufmann, Jan. 1998, ISBN: 1-55860-510-X. [Online]. Available: http://

www.elsevier.com/wps/find/bookdescription.cws_home/677869/description#description.

[95] J. F. Miller, D. Job, and V. K. Vassilev, "Principles in the evolutionary design of digital circuits—part i", *Genetic programming and evolvable machines*, vol. 1, no. 1-2, pp. 7–35, 2000.

[96] R. L. Riolo and B. Worzel, Eds., *Genetic Programming Theory and Practice*, vol. 6, Genetic Programming, Series Editor - John Koza, Boston, MA, USA: Kluwer, 2003, ISBN: 1-4020-7581-2. DOI: doi : 10 . 1007 / 978 - 1 - 4419 - 8983 - 3. [Online]. Available: http : //www . wkap . nl / prod/b/1-4020-7581-2.

[97] P. J. Angeline, "Subtree crossover: Building block engine or macromutation", *Genetic Programming*, vol. 97, pp. 9–17, 1997.

[98] S. Smith, *Biologically inspired computation*.

[99] J. F. Miller, P. Thomson, and T. Fogarty, *Designing electronic circuits using evolutionary algorithms. arithmetic circuits: A case study*, 1997.

[100] J. F. Miller, "An empirical study of the efficiency of learning boolean functions using a cartesian genetic programming approach", in *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation-Volume 2*, Morgan Kaufmann Publishers Inc., 1999, pp. 1135–1142.

[101] J. F. Miller and P. Thomson, "Cartesian genetic programming", in *European Conference on Genetic Programming*, Springer, 2000, pp. 121–132.

[102] D. R. White, J. Mcdermott, M. Castelli, L. Manzoni, B. W. Goldman, G. Kronberger, W. Jaśkowski, U.-M. O'Reilly, and S. Luke, "Better gp benchmarks: Community survey results and proposals", *Genetic Programming and Evolvable Machines*, vol. 14, no. 1, pp. 3–29, 2013.

[103] R. Poli, W. B. Langdon, N. F. McPhee, and J. R. Koza, *A field guide to genetic programming*. Lulu. com, 2008.

[104] S. Silva and E. Costa, "Dynamic limits for bloat control in genetic programming and a review of past and current bloat theories", *Genetic Programming and Evolvable Machines*, vol. 10, no. 2, pp. 141–179, 2009.

[105] J. Miller, "What bloat? cartesian genetic programming on boolean problems", in *2001 Genetic and Evolutionary Computation Conference Late Breaking Papers*, San Francisco, California, USA, 2001, pp. 295–302.

[106] J. F. Miller and S. L. Smith, "Redundancy and computational efficiency in cartesian genetic programming", *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 2, pp. 167–174, 2006.

[107] V. K. Vassilev and J. F. Miller, "The advantages of landscape neutrality in digital circuit evolution", in *International Conference on Evolvable Systems*, Springer, 2000, pp. 252–263.

[108] T. Yu and J. Miller, "Neutrality and the evolvability of boolean function landscape", in *European Conference on Genetic Programming*, Springer, 2001, pp. 204–217.

[109] S. L. Smith, J. A. Walker, and J. F. Miller, "Medical applications of cartesian genetic programming", in *Cartesian Genetic Programming*, Springer, 2011, pp. 309–336.

[110] J. F. Miller, D. Job, and V. K. Vassilev, "Principles in the evolutionary design of digital circuits—part ii", *Genetic programming and evolvable machines*, vol. 1, no. 3, pp. 259–288, 2000.

[111] S. Harding, J. Leitner, and J. Schmidhuber, "Cartesian genetic programming for image processing", in *Genetic programming theory and practice X*, Springer, 2013, pp. 31–44.

[112] L. Sekanina, "Image filter design with evolvable hardware", in *Workshops on Applications of Evolutionary Computation*, Springer, 2002, pp. 255–266.

[113] D. Hope, E Munday, and S. Smith, "Evolutionary algorithms in the classification of mammograms", in *Computational Intelligence in Image and Signal Processing, 2007. CIISP 2007. IEEE Symposium on*, IEEE, 2007, pp. 258–265.

[114] K. Völk, J. F. Miller, and S. L. Smith, "Multiple network cgp for the classification of mammograms", in *Workshops on Applications of Evolutionary Computation*, Springer, 2009, pp. 405–413.

[115] Z. Zhang, M. Bedder, S. L. Smith, D. Walker, S. Shabir, and J. Southgate, "Automated motion analysis of adherent cells in monolayer culture", in *International Conference on Information Processing in Cells and Tissues*, Springer, 2015, pp. 185–194.

[116] P. L. Townes and J. Holtfreter, "Directed movements and selective adhesion of embryonic amphibian cells", *Journal of experimental zoology*, vol. 128, no. 1, pp. 53–120, 1955.

[117] M. Takeichi, "Cadherin cell adhesion receptors as a morphogenetic regulator", *Science*, vol. 251, no. 5000, pp. 1451–1455, 1991.

[118] H. O. Steinberg, H. Chaker, R. Leaming, A. Johnson, G. Brechtel, and A. D. Baron, "Obesity/insulin resistance is associated with endothelial dysfunction. implications for the syndrome of insulin resistance.", *The Journal of clinical investigation*, vol. 97, no. 11, pp. 2601–2610, 1996.

[119] A. Nose, K. Tsuji, and M. Takeichi, "Localization of specificity determining sites in cadherin cell adhesion molecules", *Cell*, vol. 61, no. 1, pp. 147–155, 1990.

[120] M. S. Steinberg and M. Takeichi, "Experimental specification of cell sorting, tissue spreading, and specific spatial patterning by quantitative differences in cadherin expression", *Proceedings of the National Academy of Sciences*, vol. 91, no. 1, pp. 206–209, 1994.

[121] A. M. Fannon and D. R. Colman, "A model for central synaptic junctional complex formation based on the differential adhesive specificities of the cadherins", *Neuron*, vol. 17, no. 3, pp. 423–434, 1996.

[122] N. Uchida, Y. Honjo, K. R. Johnson, M. J. Wheelock, and M. Takeichi, "The catenin/cadherin adhesion system is localized in synaptic junctions bordering transmitter release zones.", *The Journal of cell biology*, vol. 135, no. 3, pp. 767–779, 1996.

[123] S. Martinek and U. Gaul, "Neural development: How cadherins zipper up neural circuits", *Current Biology*, vol. 7, no. 11, R712–R715, 1997.

[124] A. Nagafuchi, Y. Shirayoshi, K. Okazaki, K. Yasuda, and M. Takeichi, "Transformation of cell adhesion properties by exogenously introduced e-cadherin cdna", *Nature*, vol. 329, no. 6137, p. 341, 1987.

[125] W. M. Brieher, A. S. Yap, and B. M. Gumbiner, "Lateral dimerization is required for the homophilic binding activity of c-cadherin.", *The Journal of cell biology*, vol. 135, no. 2, pp. 487–496, 1996.

[126] A. S. Yap, W. M. Brieher, M. Pruschy, and B. M. Gumbiner, "Lateral clustering of the adhesive ectodomain: A fundamental determinant of cadherin function", *Current Biology*, vol. 7, no. 5, pp. 308–315, 1997.

[127] C. L. Adams, Y.-T. Chen, S. J. Smith, and W. J. Nelson, "Mechanisms of epithelial cell–cell adhesion and cell compaction revealed by high-resolution tracking of e-cadherin–green fluorescent protein", *The Journal of cell biology*, vol. 142, no. 4, pp. 1105–1119, 1998.

[128] M. A. Lones, S. L. Smith, J. E. Alty, S. E. Lacy, K. L. Possin, D. S. Jamieson, and A. M. Tyrrell, "Evolving classifiers to recognize the movement characteristics of parkinson's disease patients", *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 559–576, 2014.

[129] D. A. Augusto and H. J. Barbosa, "Symbolic regression via genetic programming", in *Proceedings. Vol. 1. Sixth Brazilian Symposium on Neural Networks*, IEEE, 2000, pp. 173–178.

[130] B. M. Gumbiner, "Regulation of cadherin-mediated adhesion in morphogenesis", *Nature reviews Molecular cell biology*, vol. 6, no. 8, p. 622, 2005.

[131] L. A. Taneyhill and A. T. Schiffmacher, "Should i stay or should i go? cadherin function and regulation in the neural crest", *genesis*, vol. 55, no. 6, e23028, 2017.